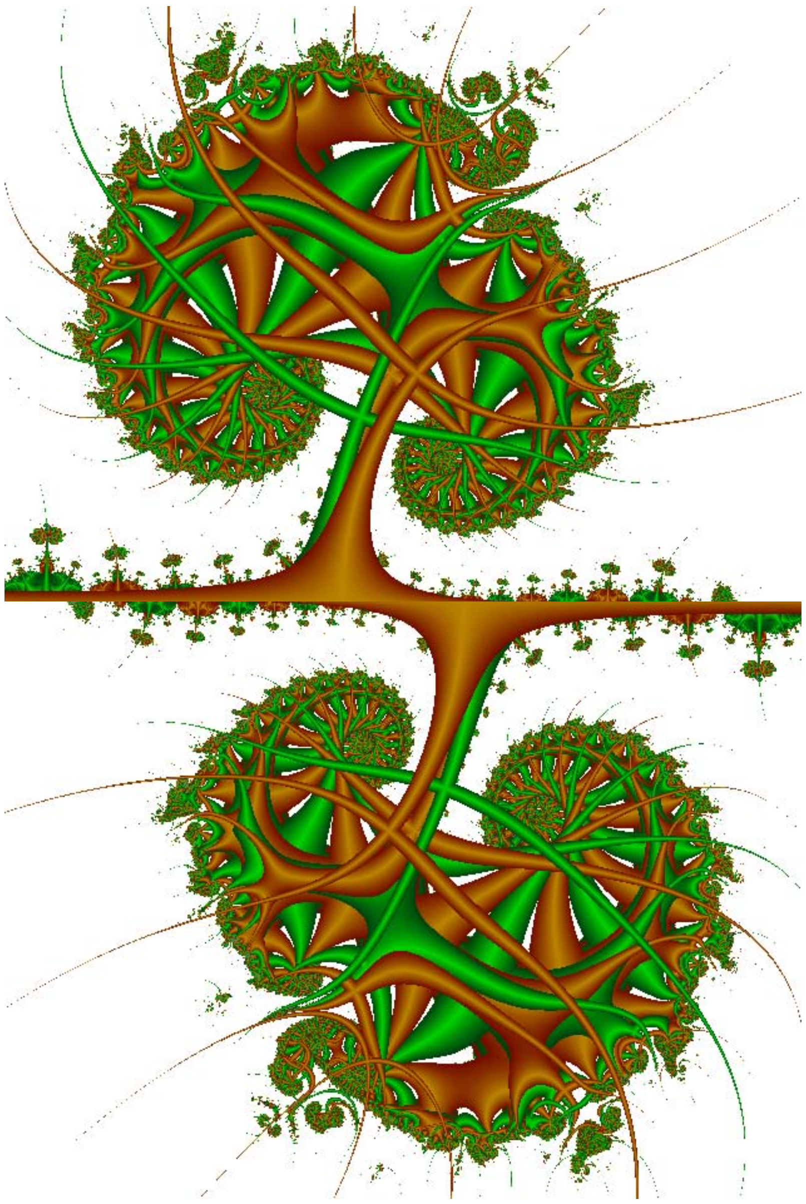
A Formal Theory of Granularity

Toward enhancing biological and applied life sciences information systems with granularity



C. Maria Keet

A Formal Theory of Granularity

Toward enhancing biological and applied life sciences information systems with granularity

Freie Universität Bozen
Libera Università di Bolzano
Free University of Bozen · Bolzano

| Fakultät für Informatik | Facoltà di Scienze e Tecnologie informatiche | Faculty of Computer Science |

KRDB Research Centre for Knowledge and Data

KRDB Dissertation Series DS-2008-01

A Formal Theory of Granularity

Toward enhancing biological and applied life sciences information systems with granularity

TESI PER IL DOTTORATO DI RICERCA IN INFORMATICA

DOKTORARBEIT IN INFORMATIK

PH.D. THESIS IN COMPUTER SCIENCE

KRDB DS-2008-01

Catharina Maria Keet

Relatore - *Doktorvater* - Faculty Advisor: Prof. Enrico Franconi
Correlatore - *Zweitbetreuer* - Faculty Co-Advisor: Dr. Alessandro Artale

Commisione Giudicatrice - *Prüfungskommission* - Examination Committee:
Prof. Dr. Brian Fitzgerald
University *of* Limerick, Ireland
Prof. Dr. Werner Nutt
Free University of Bozen-Bolzano, Italy
Prof. Dr. Fabio Casati
University of Trento, Italy
Dr. Claudio Masolo
Laboratory for Applied Ontology LOA-CNR, Italy
Mr. Marcello Federico
Fondazione Bruno Kessler, Trento, Italy

Data di consegna - *Abgabedatum* - Submission date: 21-12-2007
Data di ammissione all'esame finale - *Datum der Zulassung zur Doktoratsprüfung* -
Accepted to the final examination on: 21-1-2008
Data della discussione pubblica - *Datum der offentliche Doktoratsprüfung* -
To be defended in public on: 28-4-2008

Keywords: Granularity, Part-whole Relations, Abstraction, Ontologies, Conceptual Data Modeling, Life science applications

ACM categories: Knowledge Representation Formalisms and Methods (I.2.4), Knowledge Engineering Methodologies (M.1), Knowledge Modeling (M.4), Life and Medical Sciences (J.3)

Cover design by Maria Keet, with thanks due to the ChaosPro v3.02 fractal generator and Daniele Gobbetti.

Printed and bound by DigiPrint, Bolzano.

Contents

List of Figures

List of Tables

Acknowledgments

I would like to express my first words of gratitude to my scientific advisors Enrico Franconi, for giving me the freedom to pursue my interest and explore ideas and to have the open-mindedness and confidence in me to see it out, and Alessandro Artale, who with his to the point valuable feedback and good humour was an excellent motivator. I am very grateful to Rosella Gennari for her introduction to a wider scope on logics, the human-side of doing research, and the moral support.

I am indebted to the critical feedback from Barry Smith on article drafts that helped me to focus better and be more careful in word choice, Diego Calvanese's open-door policy was of great benefit to me for getting to grips with the $\mathcal{DLR}$s, and Claudio Masolo for his detailed constructive criticism he gave as reviewer of the draft.

I would like to thank my co-authors of publications for the fruitful collaboration in the diverse topics it spanned: Alessandro Artale for the shared interest in, primarily, part-whole relations and his persistence on the details, Anand Kumar with whom I developed a first domain granularity framework that helped scoping a range of problems, Scott Marshall and Marco Roos for the discussions on requirements for automated reasoning over bio-ontologies, Mariano Rodríguez for his input on the DL-Lite family and LaTeX tricks, and Christina Tziviskou who introduced me to WSML and meta-modelling for ontologies.

The dynamic environment at the KRDB contributed to a stimulating workplace and, in addition to above-mentioned colleagues, I particularly wish to thank Paolo Dongilli, Daniele Gobbetti, Andrei Lopatenko, and Davide Martinenghi for the pleasant working atmosphere.

Being introduced to and reminded of domain experts' requirements, desires, and real problems from the fields of biology, bio-ontologies, and bioinformatics that are in need of solutions, is both useful and stimulating. I would like to take the opportunity to thank my main points of contact and interested user-base from whom I received input that I hope to have addressed and solved at least in part, a small part. They are Santiago Garcia-Vallvé, Benjamin Good, Kristina Hettne, Scott Marshall, Mark van Passel, and Marco Roos. I particularly wish to thank Fabio Porto for the invitations to visit the Laboratoire de Bases Données at EPFL and the enjoyable interesting discussions on databases, ontologies, rules, and bio.

Last, the XXth PhD cycle was the first one to commence at FUB, which asked for some creativity and flexibility to transform written regulations into a smoothly working implementation. I wish to thank Federica Cumer for her efforts and positive attitude to try to make it work.

Doing a PhD was not quite what I had expected, but a very valuable experience. Also, with logic alone, one can easily avoid the real world altogether, and I would like to express my appreciation to Kenneth, Giuseppe, Alex, Machiel, Heidi, and Dominik for keeping me in touch with life and what's happening in the world.

Bolzano
December 18, 2007.

Maria Keet

Abstract

Granularity—the ability to represent and operate on different levels of detail in data, information, and knowledge—is indispensable for managing the huge amounts of data generated by, among others, scientists and demands representations at widely ranging levels of analysis. In particular, managing biological databases, knowledge bases, and ontologies effectively and efficiently cannot be solved by adding more one-off software applications, but requires new foundational methodologies to push implementations to the next phase of *in silico* biology. Recent requests from domain experts include adequately addressing vertical integration of information systems across levels of biological granularity and cross-granular querying and reasoning. However, these kind of enhancements have been investigated in a fragmented manner and encompass both subject domain semantics ranging from molecules up to ecosystems and computational applications for qualitative and quantitative granularity irrespective of the subject domain. Coherently and consistently representing granularity and subsequently using and reusing it is a new foundational methodology that can enhance management of information systems and exploit its data more effectively. To realise representation of and reasoning with granularity, there are multiple issues to solve along three main dimensions. Current informal approaches have inconsistent granulation hierarchies that are not or only to a limited extent usable and reusable for computation. Formal and ontological approaches for granularity offer partial, data-oriented, theories that do not address precisely what granularity is, what its components are, how to use it, or if the theory is proven to be consistent. Engineering solutions are limited to difficult to reuse, data-centric, implementation-level solutions; this counts for both the levels and granulation hierarchies and the lack of transparency of hard-coded functions.

To solve these issues, we move from a data-centric and underspecified treatment of granularity to the ontological and logical layers, where informally defined components of granularity have become ontologically-motivated modelling constructs proper. This is achieved through the analysis and formalisation of three key elements:

I. The foundational semantics of granularity are disambiguated and structured in a taxonomy of types of granularity. This taxonomy makes explicit both the ways of granulation and representation, and how entities are organised within a level of granularity, thereby also simplifying representing and using granularity.

II. The static components of granularity—such as levels, indistinguishability, how finer- and coarser-grained levels and entities relate, and granulation criteria—were subjected to an ontological analysis and formalised in a consistent and satisfiable logical theory, the Theory Of Granularity (TOG). The TOG has a model-theoretic semantics so that an unambiguous meaning is ensured, interesting properties could be proven, and satisfiability was computationally demonstrated;

III. An extensible set of domain- and implementation-independent functions were defined for both the TOG components to enable granular querying and reasoning over the theory that can be used across multiple implementation scenarios, and for moving between entities (/types) residing in different levels through abstraction and expansion functions.

Effectively, granularity is both lifted up to a higher layer of abstraction alike conceptual modelling does for software design and database schemas and precisiated in a formal theory with model-theoretic semantics, thereby having made the representation of granularity domain- and implementation independent. Hence, reusability across implementations can be ensured, which in turn facilitates interoperability among information systems, such as granulation of ontologies, knowledge bases, databases, data warehouses, and biological and geographical information systems. The modelling of a particular domain granularity framework—an instantiation of the TOG—serves not only as a novel methodology for structuring data, information, and knowledge, but also as an additional query- and inferencing layer to recombine the source data by posing granular queries and retrieve implicit or novel information hitherto hidden in the data source. A granular information system facilitates level selection and granular querying of the information source to retrieve desired facts more precisely and facilitates zooming in & out on sections of large ontologies tailored to a user's interests. It thereby alleviates the Ontology Comprehension Problem through enabling granular information retrieval for a range of different domain expert whilst having the full resource still available for reasoning across granular levels.

As encompassing architecture, there are four connected components. First, the *foundational semantics of granularity* is structured in a taxonomy of types of granularity. Second, the *domain- and implementation-independent theory of granularity* formally characterises granularity components such as granular perspective, granulation criterion, granular level, and the relation between levels. The TOG also makes use of aspects intertwined with granularity, such as indistinguishability, part-whole relations, and abstraction. When applied to databases, this TOG is positioned orthogonally at the conceptual data modelling layer, or, when applied to conceptual data models or logic-based type-level ontologies, resides at its meta-level. Third, the TOG is instantiated (at the instance or type-level, respectively) for specification of a *domain granularity framework*, such as a granular perspective for human structural anatomy with granular levels such as cell, tissue, and organ. Fourth, this domain granularity framework is then applied to the *data source*, which is, in the scope of the thesis, delimited to databases, knowledge bases, and ontologies. The combination of these four components results in a *granulated information system* that can be used for, among other scenarios, cross-granular querying and automated reasoning. Functions for the applied domain granularity are specified at the goal-oriented conceptual and logical layers and include selections of levels, of their contents (entity types or instances), combinations thereof, and abstraction and expansion operators to move from coarser to finer-grained levels and back. Possible use is demonstrated with the infectious diseases domain and other examples from the biology and applied life sciences domains. Feasibility of computational implementations was tested with the Gene Ontology and Foundational Model of Anatomy, thereby substantiating advantages of automation when applying granularity.

Last, the theory and implementation scenarios are compared with other formal and informal models and usages of granularity. It is demonstrated that the TOG entails extant theories, hence serves as a more generic, unifying, theory which is more comprehensive and can be better scalable and reusable than the extant partial theories and technology-dependent implementations. There is, however, ample room for further research, such as realising granularity-enabled linking of OWL-formalised ontologies and querying across the levels of biological granularity, rough/fuzzy extensions for level specification, and adding the granularity components as modelling constructs to conceptual data modelling languages so that the granularity may propagate automatically to the data in databases.

Part I

Introduction

Chapter 1

Introduction

Although the use of computer science and information technologies for, in, and with biology is not new, the now infamous biological data explosion has pushed this combination to the forefront. The prolific data generation and the desire to analyse the experimental results still runs ahead of the facilitating technologies to accomplish this demand effectively. A natural next step of data analysis is management of the information extracted from the data, and, consequently, knowledge management. Both biology and computer science & information technology, however, face multiple challenges how to improve information and knowledge management. The phase of information and knowledge management concerns *what* to represent, *how* to represent it, and *what* can be done with it. For any subject domain, the emphasis is on what piece of reality to represent, whereas for computer science the emphasis lies on how to achieve this and what can be do with a representation. Research in the latter falls in the sub-discipline of knowledge representation and reasoning. The former—how to achieve representation of knowledge—focuses on conceptual data modelling, ontologies, and logics to provide a formal approach to improve conceptual data models and ontologies, primarily by enabling to be more precise in what is modelled to permit making the implicit explicit. The latter, reasoning, comprises enhancing reasoning services over the theories to automatically derive knowledge implicit in the representation, and to improve the theory and technologies for querying data, information, and knowledge.

Exploration and usage of conceptual data modelling and ontologies over the past 20 years has resulted in improvements for data- and information management, but both methodologies take each modelled aspect as equally relevant and important, such that one still cannot see the wood for the trees. Effective information- and knowledge management must facilitate a zooming in or zooming out to a section of interest for diverse types of users, abstracting away details when it is not needed, and focussing on a level of detail relevant for the domain experts' information needs. In short, accessing and using information and knowledge at the *optimum level of granularity*. As such, taking into account granularity in the representation of information and knowledge amounts to adding a *novel way of representation* to aforementioned existing modelling methods, and, consequently, to have new possibilities for manipulating, re-combining, and reorganising existing information and knowledge. Such new options equip domain experts to take a fresh look at the structured information, uncover hitherto hidden knowledge, and discover more about the world we live in. However, we are far from this granularity-enhanced situation. It is not clear what granularity is, what a 'level of detail' is and contains, how to abstract away details, how exactly information systems can be enhanced with granularity in a usable and, moreover, *reusable* way, and what kind of novel information may be extracted using granular queries and reasoning services.

In order to shed light on the types of problems coming from domain experts' requirements regarding subject domain specifics and technological issues, §1.1 commences with an overview of typical issues in biology and the applied life sciences (§1.1.1) and subsequently generalises that to its relevant issues in computer science in §1.1.2, which have as common denominator combining different levels of detail and selecting a desired level of detail. This is followed by a summary of the state of the art on granularity research and engineering and closely related topics such as abstraction and modularisation (§1.1.3), which is synthesised in a list of significant problems in §1.1.4. The research aims, questions, and overview of the proposed solution with its benefits are provided in §1.2, which is followed by the research methodology and the outline of the organisation of the remainder of the thesis (§1.3). A summary of this chapter was published and presented at the *KnowledgeWeb PhD Symposium* (Keet, 2006f) and a more comprehensive treatment on the combination of biology and computer science can be found in the *CSBio Reader* (Keet, 2005c; Keet and Franconi, 2005).

1.1 Background and motivation

The relatively recent focus on systems biology in molecular biology and medicine leads to a more pressing demand of information integration across disciplines, not only horizontally among software systems covering the same or similar subject, such as genebanks for mice genomes and single nucleotide polymorphisms (SNPs) of human genes, but in particular a *vertical* integration from genes, to metabolisms, to organ systems, organisms, and populations. Moreover, biological data integration is only a prelude to conducting more complex *in silico* biology and medicine. This pushes research and development of computer science and Information Technology that will have to meet the complex requirements, but are not yet doing so. Problems encountered extend to other subject domains and, in fact, knowledge management in general. Solving those problems, in turn, will push biology and medicine, in particular regarding clear and unambiguous specification of the subject domain semantics—*i.e.*, achieve better precision and accuracy of the biomedical-ontologies with respect to the piece of nature it is supposed to represent—and deploying reusable solutions. Major advantages of the bio-ontologies and bioinformatics areas are that they are comparatively active, open to experimentation with new technologies, formal and informal ontologies are created and used both for formulating shared agreements within a broad research community of users and for Ontology-driven Information Systems (roughly conforming to usage as suggested by Guarino (1998)), and biology & applied life science provide motivating examples to illustrate real-life relevance of problems to solve. Because of the initial push-characteristic by the subject domain and prospects for potential early-adoption of possible solutions, several essential characteristics and problems biology and applied life sciences are highlighted first in §1.1.1, which is followed by corresponding and their more generic computer science topics in §1.1.2. Subsequently, we look at what granularity specifically can offer to contribute solving those issues (§1.1.3) and what its main limitations are, which are summarised in §1.1.4.

1.1.1 The Universe of Discourse: biology and applied life sciences

One of biology's and applied life sciences' recent developments is systems biology, which applies systems thinking by laying bare the underlying web of reciprocal relationships (Richmond, 1991) as opposed to a reductionist approach. Its recent popularisation is largely due to its 'discovery' by molecular biologists, but has been applied before in ecology, developmental biology, and metabolic control analysis (Atauri *et al.*, 2004; Westerhoff and Palsson, 2004). A biological system may be limited to the metabolome or genotype-phenotype interactions (Sontag, 2004), but also can have grander aims such as integrated omics layers to facilitate drug discovery (Nicholson *et al.*, 2004). For drug discovery, comparative genomics and metabolomics across species are needed to find commonalities that can be exploited for increasing the understanding of an organism's (mal)functions. This stretches medicine into basic biology, which brings about renewed assessment which sections of biological data are usable for medicine and which not, preferably carried out automatically and hidden from the user to avoid overloading biomedical researchers with for them irrelevant data. In addition, traditional boundaries of the typical research topics of the basic life sciences are crossed, where a particular item at another level of detail is of interest but not all details of the adjacent discipline; for instance, that a plant scientists needs to know details about structures of plant cells but not animal cells or microorganisms. For a biomedical scientist, relating genealogy, epidemiology, geographical healthcare statistics, and human physiology to discover a gene or gene complex that causes a disease or contributes to an aspect of longevity requires vertical integration of domain knowledge from genotype up to environmental living conditions. For instance, a researcher wants to reason over a section of an ontology or query appropriate databases to retrieve cross-species DNA sequences for its corresponding transcriptome and map that to the related metabolome of human. To realise this, she needs to find her way in the more than 1500 biological databases available (Galperin, 2005; Bernal *et al.*, 2001) and have contained in the query answer the information relating to a biochemical pathway of humans, but not, say, xylem composition. Even within pathway analysis—a combination of biochemistry, molecular biology, cell- and organism physiology—problems for dealing with more and less detail are abound. Representation of biological pathways are a simplification compared to large ontologies, because they are just graphs and visualisation is done manually or based on storage and querying of databases; hence, pathway visualisation issues are a variation on the generic problems of managing large ontologies (see §1.1.2), so that a domain- and implementation-independent solution may be able to address them both. Given the analyses of PathwayAssist, PathArt, KeyMolnet, and MetaCore (Hettne and Boyer, 2005), and BioCarta and KEGG (Schmitt and Haaland, 2004), there are two main domain-specific issues. First, huge, detailed, breadth and depth of pathway semantics is not addressed computationally, but manual analyses and comparisons of the figures are carried out. It lacks advanced searching capabilities other than string matching and pre-coded figure zooming, and is far away from reasoning support for pathway comparison, classification, and molecule fact-finding across pathways. Kitano *et al.*'s (2005) efforts toward standardisation is a step forward, but is as of yet insufficient for such tasks. Second, pathway management in context is lacking: even if pathway data is stored in a database, the resultant output is a static figure lacking contextualised information about the location of the pathway; *e.g.*, the BioCarta "FXR and LXR regulation of cholesterol metabolism" does not distinguish between the differences of this pathway located in the intestine and in the liver (Hettne and Boyer, 2005). Organisations do not have the same scope for pathway model representation, resulting in different biases on what and is not is included in the representation concerning the entity types, relations, and inconsistent levels of detail within and between models (Schmitt and Haaland, 2004). Further, graphical icons may be limited to generic items such as "small molecule" and "cellular process" from which no further detail can be retrieved (*e.g.*, in PathwayAssist), whereas it also includes named entities in more detail, thereby mixing different levels of granularity in a static manner. Considering the effort put into development, maintenance and usage of pathway representations stored both in databases and ontologies (*e.g.*, PathwayAssist; BioPAX) and well-known requirements slip of increasingly challenging demands formulated by biology researchers, the urgency for computer scientists to solve these problems rapidly increases.

A new sub-field is metagenomics, which is also called high-throughput molecular ecology or community genomics (DeLong, 2005; Gross, 2007). It combines the micro-level of molecular biology with the macro-level of ecosystems. At present, this is for technological reasons limited to the study of the interactions within microbial communities *in situ*, such as marine microbiology (Schleper *et al.*, 2005; DeLong, 2005), the use of microbes in biotechnology (Lorenz and Eck, 2005), and the microbial community in dental plaque (Kolenbrander *et al.*, 2005). It reveals community and population-specific metabolisms with the interdependent biological behaviour of organisms in nature that is affected by its micro-climate; hence, requiring ecological knowledge to resolve the gaps in understanding of the phenomena. Simplified, it tries to answer questions like 'what lives in my soil sample?' and 'is the physiology of the organisms in the target population uniform?'. Metagenomics also aids clarifying the distinction between the individual, a population of individuals of the same type, and the universal (species) and elucidating the versatility in living and survival strategies of microorganisms. Metagenomics requires not just more and new software-supported analysis tools, but also adequate management of biological knowledge across levels of granularity going from the molecular level up to ecosystem. Among the many outstanding problems in metagenomics are issues such as relating environmental metadata with sequence databases, assessing if specific adaptations to depth, light or redox gradient have occurred among the sampled microbiological community, gene flow through horizontal gene transfer, the sheer size of the data running into terabytes across a grid (DeLong, 2005; Seshadri *et al.*, 2007), and solutions for dense and condensed visualisation of intra-species as well as cross-species gene clusters and proteins for functional, ecological, and evolutionary analyses (Eppley *et al.*, 2007). From the bioscience perspective, this requires horizontal and vertical interdisciplinary approaches, combining biochemistry, microbial ecology, genomics, environmental sciences, and the ecological niche, but also having to deal with both quantitative and qualitative granularity.

1.1.2 Computer science

This section highlights several challenges, where relevant for granularity, in the areas of database and ontology management, reasoning, the Semantic Web, and visualisation.

Database and Ontology Management. A task for computer science and engineering in the endeavours as outlined in the previous section is ultimately to provide methodologies and tools that facilitate this integrative biology approach. Computer science can offer, among others, knowledge bases and ontologies with automated reasoning services, database connectivity, and the highly integrative technologies required for scientific workflow systems and virtual laboratories for e-Science. To develop a software system capable of meeting such requirements for data- and knowledge bases, one at least needs to address:

1. Demarcation of the UoD and identification of perspectives and levels of detail to organise these data sources along the scope of their content;
2. Inventory of existing databases and ontologies and its usability for reasoning, following with dynamic database integration and/or ontology linking, mapping, or integration;
3. Software support so that it indeed can eliminate unnecessary generalities and details, use it for cross-granular querying, fact finding, and hypothesis testing;
4. Sufficient query functions and automated reasoning services to navigate intelligently and quickly through the granular space.

The boundaries of the subject domain are less obvious than it may seem: biology ranges from the molecular level to ecosystems, and medicine and agriculture comprise applied biosciences, technology, and practice. The UoD is huge and diverse and hundreds of knowledge- and databases and informal ontologies are available on the Internet, varying from primary source to boutique data- and knowledge bases and ranging from foundational to cottage-industry and experimental ontologies and other spin-offs (Antezana *et al.*, 2006; Bad Bug Book; BFO, 2007; BioPAX; FMA, 2003; GenBank; Jaiswal *et al.*, 2002; Masolo *et al.*, 2003; OBO Foundry, 2006; POC, 2002; Rosse and Mejino, 2003; UniProt, among many). Customary in the biological science is to 'abstract away' undesired detail or the larger system and to copy only sections of interest from an ontology or database to use it in another database or ontology. Ideally, one would not want to copy such data or information, but when formulating a systems biology query, to have the required databases or ontologies connect *on demand* to answer the question and its nested queries. Hard-coding every possible question is restrictive, inflexible, laborious to create, and difficult to maintain, therefore far from ideal. To realise on demand querying at the desired level of detail, a theory of biological granularity is a prerequisite for organising both the data sources according to topic they cover so as to *meaningfully* link a chemicals ontology to a taxonomy of enzymes, to an ontology of pathways. Further, a human apparently seamlessly shifts between levels and perspectives, *e.g.*, from structural to functional and from gene to gene product in some organ, and alternately emphasises the criterion to granulate and the contents within a level itself. But *how is this informal and ambiguous notion of granularity actually used? What are the relations between granular perspectives and their levels? How can we teach a computer program to use granularity soundly and consistently? When and how can one make the program switch from one perspective to the other and from one level of detail to another? How can we query a granulated information system? Where and how is granularity used, why, and how can it contribute to better knowledge management?* These questions, among others, will be addressed in this thesis. Analogous to ontologies with bioscience (Keet, 2005a; Rosse, 2005), granularity in biology and applied life sciences information systems may contribute to new insights in the biosciences. Several granulation hierarchies combined with ontologies exist, such as human anatomy, infectious diseases, colon carcinomas, and physiology (GO, 2004; Kumar *et al.*, 2004, 2005; Keet and Kumar, 2005; Rosse and Mejino, 2003; Smith *et al.*, 2007). Augmenting this with more and a better representation of domain granularity founded on an ontologically-motivated theory of granularity can enable realisation of items 1, 3, and 4 in the list above, because if we have a full-fledged theory of granularity, granular querying and reasoning can be transparent and formulated at the conceptual *what*-layer as opposed to have one-off functions and queries at the software system's *how*-layer.

The Semantic Web. Research and development of technologies for the Semantic Web adds requirements for scalability and flexibility to common database an ontology management. The recently established Health Care and Life Sciences Interest Group of the World Wide Web Consortium (W3C HCLS IG, 2006) aims to contribute to the creation of a Semantic Web for the Life Sciences (SWLS) by applying existing Semantic Web Technologies. To realise the SWLS, one needs link the deep Web to enable finding the right information at the right level of detail quickly and at once, which again requires *vertical* integration of domain knowledge, but also on demand. For such a flexible and dynamic approach, one could take Semantic Web agents that can negotiate about the meaning of the data sources through services represented and implemented with the WSxx suite (WSDL, 2001; WSML, 2005; WSMO, 2006; WSMX)—provided the semantic descriptions are available. This generation and use of comprehensively annotated biological resources is a crucial open problem. Ontologies represented in the W3C Standard OWL (OWL, 2004; RDF, 2004) are useful to deploy, but they neither provide a mechanism to address levels of (biological) granularity nor take into account the more generic case of modularisation and context so that domain experts with different UoD foci can use subsections of the same online sources. Granularity is already a pressing problem to address in biological data management in absence of a SWLS, and worsens as biologists keep rapidly producing more data without better data- and information management, thereby complicating their own research. However, Semantic Web Technologies may be used for developing granularity-based solutions and to integrate granular information management for exploiting existing data better and bring more structure to the 'biological data anarchy' on the Web. However, biological granularity has not been subjected to a formal analysis yet, the nature of biological information is more diverse than operational data of enterprises and research toy-examples, and how granularity can be applied as an additional ontology-like formal structure and be used for meaning negotiation by Semantic Web services, querying, reasoning, and information visualisation is underspecified by biologists, ontologists, and Semantic Web engineers alike.

Reasoning. Each of the previously mentioned themes probably could be addressed by a one-off application with many hard-coded functions, but this would amount to adding yet another tool to the myriad of software applications with difficult to maintain code, methods, and predefined queries. What is lacking are *generic methodologies* to address the issues, of which one is the use of automated reasoners (including expressive query languages).

On demand combining and querying information systems at different levels of granularity may require new reasoning services; *i.e.*, it remains to be seen if automated reasoners such as Fact++ and Pellet (Fact++, 2007; Mace4 & Prover9, 2007; Pellet, 2006) offer sufficient reasoning services to deal adequately with granularity. This, of course, depends on both the language required for representing granularity comprehensively and the application scenarios. For instance, querying the Foundational Model of Anatomy (FMA), which is stored in a relational database, on parts of tissues that are types of cell uses the taxonomy, partonomy, and recursive queries, whereas querying for the location of macrophages in any organ requires also a path query of arbitrary, but finite, length. This can be done with StruQL over the instances in OQAFMA (Mork *et al.*, 2003). Conversely, with the FMA represented in OWL-DL (in the TBox), one can check satisfiability and spot inconsistencies (Zhang *et al.*, 2006) using automated reasoners such as Fact++ and Pellet, but it cannot do aforementioned path queries across the informal levels of granularity. Adding granularity to a data source as an additional layer of logic could simplify the queries for one would then be able to represent the hitherto implicit knowledge about granularity and use that in query formulation and evaluation. With such granularity-enhanced management through formal ontologies and conceptual data models and their databases, automated reasoning over the data to derive new information as opposed to the current time-consuming manual *ad hoc* analyses conducted by domain experts may be realised then, too.

Information retrieval from large corpora. Solving some of the previously discussed problems by using granularity is the focus in the remainder of the thesis, which is not to say granularity would not be able to contribute to a solution in other areas. Extending the topic of the Semantic Web in a different direction, we arrive at information retrieval from the Internet. The amount of online available publications has increased tremendously (Graham and Dayton, 2002) and the famous PubMed/MEDLINE has indexed well over 12 million articles covering about 25% (Fangerau, 2004) of published articles in biology and (bio)medicine. When within a few years about 60 000 articles are published on a single topic like apoptosis (Lazebnik, 2002), it is impossible to read all topical literature; hence, (i) selections have to be made for narrowing down searches to retrieve a readable amount of relevant papers and (ii) somehow scan by other means closely related literature. Two approaches are taken. First, text mining the literature to extract what is known in publications but not by the individual researcher, *e.g.*, to find (semi-)automatically the metabolic and signalling pathways (Daraselia *et al.*, 2004). Second, development of intelligent querying capabilities to address point (i). A basic implementation of the latter is GoPubMed (Doms and Schroeder, 2005; GoPubMed, 2007), where the user can specify a query in the usual way with PubMed, but the result is compared with and sorted along the Gene Ontology (GO), showing the relevant GO terms and number of articles found within the original PubMed result. With this additional GO tree structure, one can drill down to the desired level and assess the abstract on relevance, including more or less specific GO terms than the keywords of the query. The idea might be useful for creating a granular view to find the desired scientific articles more quickly and along different dimensions, as well as find relevant articles that are related but that the domain expert may not have been aware of.

Visualisation. Although visualisation is a separate field in engineering, it is related to advanced database and ontology management, because appropriate data retrieval is required to feed graphical layout algorithms with data before it can graphically structure this information from ontologies and conceptual data models and it can be used for visualisation of query formulation. Therefore, techniques to manage which data should, and should not, be shown in a graphical representation to make it comprehensible for the user, is relevant.

The main problems with current graphical interfaces to depict ontologies is that they are not scalable to ontologies larger than about 50 entity types, thereby contributing to a database & ontology comprehension problem (Campbell *et al.*, 1996; Keet, 2005a). The ezOWL plugin for Protégé is least scalable, but this is primarily because it shows all defined relations, whereas OWLViz (OWLViz, 2005) still produces a readable visualisation up to about 70 entity types, but has the disadvantage that it shows the taxonomic structure only (see Keet, 2004b, for test results of the comparison). GrOWL offers both visualisation of OWL and basic features for visual querying (GrOWL; Krivov and Villa, 2005; Krivov, 2005), which is better scalable compared to the other tools, but still considers only the most detailed level of an ontology. The SEWASIE Ontology Design Tool (Jarke *et al.*, 2005) uses data from a database to visualise the entities and relations and, like GrOWL, allows ontology creation and updates through a graphical interface. However, the underlying database cannot be queried via that interface, and it does not scale up well either. Large ontologies such as the GO and the FMA are visualised only as a tree in the customary fashion of the directory tree of an operating system (*e.g.*, in Protégé, DAG-Edit, other tools (Khatri and Draghici, 2005), and web interfaces). The tree structure enforces depicting types of relation in one way only and GO's multiple inheritance is dealt with by including those entities multiple times in separate branches. The FMA (FMA, 2003) uses this same approach with different subtrees for different types of relations (*is_a*, *part_of* etc.) so that one looses the oversight: upon selecting an entity, one can see only a partial representation of its direct neighbours but never the interconnectedness of the entity from different perspectives. In addition, when repeatedly clicking to an entity deep in the hierarchy, the higher level parent entity is outside the view or one has to browse sequentially in the various subtrees. Taking into account that biologists are more visually oriented in representing their knowledge, adequate behind-the scenes techniques for effective visualisation is an imperative.

1.1.3 Existing approaches and solutions for granularity

This section contains a summary of existing approaches and solutions, which range from informal usage of granularity without software support to formal partial theories for either qualitative or quantitative aspects of granularity. A comprehensive literature review and analysis is deferred to Chapter 5 to make possible a comparison with the solution that will be proposed in the upcoming chapters.

The literature can be assessed along three dimensions, each with their own criteria:

- ⋆ *Formal characterisations of granularity and ontological approaches*: (i) Coverage of the theory, which aspects of granularity it considers and which not; (ii) Knowledge Representation language used for the formal characterisation; (iii) Ontological rigour taken into account for the development of the theory; (iv) Amenability for computational implementation and usage with automated reasoning services.
- ⋆ *Engineering solutions* that mainly emphasise the quantitative aspects of granularity: (i) Usability and usefulness of the solution to solve generic problems independent of the UoD; (ii) Reusability outside the application for which it was developed; (iii) Possibility to scale-up the approach; (iv) Amenability to formalisation.
- ⋆ *Informal approaches to (biological) granularity*: (i) Usefulness to solve a particular problem within the UoD; (ii) Reusability of the informal description of biological granularity and subject domains they cover; (iii) Amenability to formal specification for constructing a subject domain granularity framework.

The first two items can be grouped as granular computing with the aim of *structured thinking* and *structured problem solving*, respectively, where the latter can be divided into methodologies and processes (Yao, 2005b, 2007b). When the former is clear, the latter ought to fall in place with ease to, in turn, solve the methodologies and processes. Conversely, the former has to match reality sufficiently to be applicable, thereby closing the loop. Such informal considerations will have to be taken into account when developing a generic theory of granularity.

Looking first at informal approaches, Tange *et al.* (1998) constructed granular perspectives and levels for medical practice based on term usage in literature that was intended for text mining and categorisation of scientific literature. This is as of yet the only *computational* usage of shallow biomedical granularity—consisting of three perspectives (history, physical examination, and progress notes) and three levels with, for example, Physical examination - Lungs - Auscultation—that informally combines a process, structural part of the human body and "type of observation", which have to be organised more clearly and preferably in an ontologically consistent manner. Other informally established granular levels include the omics spaces genomic, transcriptomic, proteomic, metabolomic, and phenomic levels (Toyoda and Wada, 2004), which are, roughly, related through time displacement and causality. Other informal approaches in biology and medicine include, among others, Elmasri *et al.* (2007); Fent *et al.* (2005); Grizzi and Chiriva-Internati (2005); Hunter and Borg (2003); Ribba *et al.* (2006); Salthe (1985).

On the border of biomedicine and formal approaches is Kumar *et al.*'s granularity for human structural anatomy and relatively simple $gran$ function (Kumar *et al.*, 2004, 2005; Keet and Kumar, 2005). They have GR as the ordered set of levels of granularity applicable to a domain and U denoting the set of biological universals, so that "gran : $u \rightarrow gran(u),$ for $u \in U$ and $gran(u) \in GR$" returns the level of granularity where the entity type of interest resides. The idea of this function may be useful in retrieving information at which level the entity type of interest resides, but this assumes that granulated domain knowledge already exists and it requires patchwork in the logic, design, and implementation. For instance, $gran$ returns only one level at a time, which, if used with more than one perspective (granulation hierarchy), requires introduction of modifiers for perspectives x and y (and any other) to get functions $gran\text{-}x$ and $gran\text{-}y$ and so forth. Modifying $gran$ in this way (Keet and Kumar, 2005) suggests the functions are different, whereas they perform the exact same task. Alternatively, $gran$ has to be redesigned to allow for an answer that contains more than one level; using standard database querying may be a more robust option. Alternatively, and still without underlying formal representation of granularity, different IDs have to be assigned with an ID management mechanism in place to prevent incorrectly mixing different hierarchies. With contextual information, *i.e.*, proper management of granular perspectives, the same entity type can reappear in different perspectives, thereby avoiding inconsistencies in the software system. A separate issue is its bottom-up development of granular levels limited to human beings (Kumar *et al.*, 2004, 2005; Rosse and Mejino, 2003), which are not reusable in an expanded subject domain. For computational implementations, however, an underlying domain-independent logically consistent theory of granularity is an imperative to meet requirements such as reusability, flexibility, and interoperability. Similar issues can be observed for conceptual data modelling for multi-representation geo-spatial databases in geographical information systems (Fent *et al.*, 2005; Fonseca *et al.*, 2002; Parent *et al.*, 2006a). Other granulated information systems are an architecture for software-supported individual student-tailored educational study feedback (McCalla *et al.*, 1992) and computer games (Zukerman *et al.*, 2000). Interestingly, Tange *et al.* (1998) and Zukerman *et al.* (2000) achieved better performance with less levels in more hierarchies than with fewer perspectives that had more levels.

Formal approaches motivated by engineering usefulness are restricted to a partial account of granularity and incorporate modelling decisions suitable for the engineering scope, such as data warehouse design (Franconi and Kamble, 2004; Luján-Mora *et al.*, 2002, 2006; Malinowski and Zimányi, 2006), UML (Abelló *et al.*, 2006), and databases as linguistic corpus (Fagin *et al.*, 2005), and therefore are not easily transportable to other implementation scenarios such as Geographic Information Systems (GIS) and ontologies. Also, they have specified a large set of one-off functions and data manipulation operators only at the design or implementation layer; compare *e.g.*, $\mathcal{GMD}$, MSD, MADS, and MultiDimER (Franconi and Kamble, 2004; Fagin *et al.*, 2005; Parent *et al.*, 2006a; Malinowski and Zimányi, 2006), or see Euzenat and Montanari (2005) for an overview on theories of and functions for time granularity. To the best of my knowledge, there is no known to be consistent, satisfiable domain- and implementation-independent theory of granularity, be it proven by hand or with an automated model searcher.

Hobbs (1985) has introduced several core components of granularity and Bittner and Smith (2003) have developed an ontologically-motivated formal "theory of granular partitions" (TGP) based on mereology. The TGP is relatively comprehensive and useful for granular levels, but it is limited to mereology, does not address the types of aggregation commonly used with data mining and conceptual data modelling, has no functions, no mechanism to deal with multiple granulation hierarchies for different perspectives, and does not allow for the kind of granularity and abstraction commonly used in biology or Mani's (1998) folding operations in linguistics. An earlier, philosophy-oriented, version of the TGP (Smith and Brogaard, 2002) aims to be more general by referring to set theory, spatial granularity, and time granularity, but is data-centric like the TGP and, whilst thus covering many topics in the text, has comparatively few formal characterisations. There are few contributions on granularity from philosophy, which is addressed mostly within themes such as hierarchical systems and emergent properties (Cariani, 1997; Edmonds, 2000; Salthe, 1985, 2001; Wimsatt, 1995; Yao, 2005b) where the main emphasis is on use of levels of detail to demarcate models to achieve better scientific explanations of natural phenomena and to address the limitations of those models with different theories for different levels of detail. Thus, it does not focus specifically on the ontological status or nature of what granularity is, and what it is not.

Other types of implementations exist in different research disciplines, such as data mining and clustering techniques, which are grouped recently under the term Granular Computing, which focuses on computational problem solving aspects. It combines efforts primarily from AI, machine learning, database theory and data mining, and (applied) mathematics with fuzzy logic and rough sets (Yao, 2005b), *e.g.*, Peters *et al.* (2002); Reformat *et al.* (2004); Yao (2004a); Zhang *et al.* (2002); Zadeh (1997). Lin (2006) summarises several of the many example usages and (Bargiela and Pedrycz, 2006; Yao, 2007a) describe background and trends. In this context, the comprehensive description of granule and granulation by Zadeh (1997) is useful for grouping together several notions about granular computing: "Informally, granulation of an object A results in a collection of granules of A, with a granule being a clump of objects (or points) which are drawn together by indistinguishability, similarity, proximity or functionality... In general, granulation is hierarchical in nature.". The similarity and equivalence relations have been well investigated with set-based approaches (Bittner and Stell, 2003; Chen and Yao, 2006; Hata and Mukaidono, 1999; Mencar *et al.*, 2007; Peters *et al.*, 2002; Skowron and Peters, 2003; Yao, 2004a), but not indistinguishability. In addition, this set-based approach has issues that may be better addressed with mereology (Abelló *et al.*, 2006; Bittner and Smith, 2003; Varzi, 2004a). Major themes addressed for computational problem solving are quantitative granularity and—like with DWHs and GIS (Chen and Yao, 2006; Ning *et al.*, 2002; Stell and Worboys, 1998; Yao, 2004a)—it takes a data-centric approach toward granularity, whereas for conceptual data modelling, ontologies, and the Semantic Web, there is also the need to deal with both qualitative aspects of granularity and with the conceptual modelling and ontological analysis layers.

Topics closely related to granularity. Modularisation, context, and abstraction address aspects of dividing things in subsections and/or dealing with less or more detail. These topics overlap with granularity so that several aspects of modularisation and abstraction are useful to consider. *Modularisation* is a mechanism to achieve goals such as reusability of large conceptual data models, program code, or ontologies and splitting them up into manageable chunks to fit the input of groups of domain experts, modellers and programmers. Due to a more liberal scope, divisions are made arbitrarily and for convenience with respect to the subject domain semantics, thereby contradicting the goal of reusability. In fact, the former tends to end up as design patterns if it is indeed a reusable piece of information, whereas the latter is in UML modelling realised through packages. Such named packages are high-level elements that group together a coherent set classes, their attributes and associations of a UML (class) diagram and packages can contain other packages (OMG UMLSpec, 2005). This practice does not guarantee modularisation equates with granulation, because it can involve dividing the content at the same level of granularity although it possibly could be used for it. Different from the arbitrary manual modularisation is automated modularisation based on the connectedness of concepts in a DL-based ontology (Cuenca Grau *et al.*, 2006, 2007). Although the metrics and algorithms process only the syntax and the computed isolated areas in a logical theory do not imply coarser- or finer-grained-ness of those parts of the ontology, it might be feasible to amend the algorithms to 'find' levels. *Abstraction* is different from granularity in that one goes from a detailed to simplified representation where one chooses to ignore undesired aspects; that is, granularity acknowledges increasing levels of detail *and* generality, whereas abstraction only reduces the scope toward the salient semantics captured in a, mostly, conceptual data model. Further, granularity is static, whereas abstraction is the process to go from one level to another and thereby could provide a dynamic component to granularity. However, what the process of abstraction is and what an abstraction function is supposed to do is not always clear or consistent throughout the proposals (Ghidini and Giunchiglia, 2003; Campbell *et al.*, 1996; Mani, 1998; Tavana *et al.*, 2007). There are divergent goals, hence also solutions, for abstraction: existing approaches comprise manual and semi-automatic abstractions, syntax-focused formalisation of abstraction, syntactical abstraction augmented with semantics, and several others (Campbell *et al.*, 1996; Ghidini and Giunchiglia, 2003; Giunchiglia *et al.*, 1997; Jäschke *et al.*, 1993; Keet, 2005b; Mani, 1998; Pandurang Nayak and Levy, 1995). Common to these solutions for abstraction is that they exhibit three main problems. First, the focus on contents of a level and thereby abstraction lacks a surrounding *framework* and notion of 'abstraction levels'. Second, similar to the ambiguous use of the $gran$ function, a general abstraction function abs does not reveal what it is abstracting unless more detailed distinctions between the *types* of abstraction are made. Third, current proposed solutions are mainly theoretical and not developed for & assessed on scalability.

1.1.4 Summary of significant problems of granularity

Merging requirements of domain experts with the state of the art in granularity, the type of shortcomings and significant problems can be divided into three groups: problems with informal & bottom-up approaches, limitations of the existing formal approaches to granularity, and engineering shortcomings.

- *Issues with the informal approaches.* They are not or only to a very limited extent usable for computation and reasoning, ontologically inconsistent, underspecified, and the granulation hierarchies are not reusable within the same domain but in another application. People do 'abstract away' things, but how this is done and how to translate that process to a computationally usable and flexible representation remains to be resolved.

- *Limitations with the formal and ontological approaches.* They are more or less compatible *partial* theories that neither address precisely what granularity is nor what its components are—such as levels, granules, indistinguishability, how finer- and coarser-grained levels and entities relate, and granulation criteria—nor how to use it, therefore they are of limited use. No subject domain and implementation independent theory of granularity exist that can ensure any domain-granularity specification is logically sound and consistent.

- *Limitations of engineering solutions.* By virtue of being engineering approaches, they are limited to data-centric implementation-level solutions that are not reusable in the current format outside the (type of) software application each one is designed for; this counts for both the granules and the lack of transparency of the myriad of hard-coded functions. With the increase in distributed software environments and corresponding integration challenges, the absence of an underlying unifying framework for dealing with granularity hampers interoperability, and vertical integration complicates further the already difficult to scale-up *ad hoc* laborious manual translation efforts to let software components interoperate.

The existing formal, informal, and engineering approaches are largely intuitive, but formalizing intuition does not necessarily lead to a 'good' theory that incorporates notions of the ontology of granularity.

1.2 Objectives and proposed solution

The objectives are structured according to the central aim of the thesis, research questions, and tasks. Subsequently, the proposed solution and its benefits are outlined.

1.2.1 Aim, research questions, and tasks

The aim of this research is to develop a formal, domain- and implementation-independent theory of granularity that can be used for computational reasoning in different subject domains. This theory of granularity is added to or integrated with data- and knowledge bases and ontologies to enhance data, information and knowledge management, including querying and reasoning across levels of granularity. The theory will be sufficiently comprehensive to be useful in the subject domain of biology.

The main research question can thus be formulated at a high-level: *Why, how, and where will usage of granularity improve knowledge representation and knowledge management?* This requires a more precise demarcation and division into four sections and further subtopics:

1. Can a subject domain-independent reusable theory of granularity be defined and formalised?
   (a) What are the characteristics of different 'types' of granularity?
   (b) What are the key requirements for and components of granularity?
   (c) What are the characteristics of those components of granularity, such as granular level and perspective? How are they related to each other? How are levels of granularity related, what are the (primitive?) relations?
   (d) Based on the types of granularity, where and how does this influence a reusable theory of granularity?

2. How does a meta-level theory of granularity constrain domain specific granularity?

3. How to relate domain data to a domain granularity framework and apply a domain granularity framework to data?
   (a) What are the design decisions, if any, to make a theory of granularity implementable (w.r.t. decidability, complexity, current technologies)?
   (b) Where does it make a difference in proposed solution using a database versus knowledge base or ontology as type of data source?

4. What reasoning tasks require, or can benefit from, granularity? Where and how will this affect the following types of tasks?
   (a) For usage of the structure: *e.g.*, level selection, fact-finding, ontology browsing
   (b) What are the ways for moving between levels (abstraction and expansion) and which functions do they involve?

These questions are specified in more detail later, where each corresponding paragraph in chapters 2-4 contain more specific research questions relevant to each sub-topic under investigation.

Based on the research questions, several tasks can be formulated that have to be addressed satisfactorily if granularity is to be applied in a knowledge management system. These are:

Task 1. Perform a selection of levels from a particular domain granularity framework d_i, *i.e.*, $task^1(d_i^f) \rightarrow lss_i$, where lss_i is the selected subset of levels within that domain granularity framework.

Task 2. To retrieve contents of at least one level, we have $task^2(gl_i) \rightarrow \mathcal{E}$, where $\mathcal{E}$ denotes the contents of a granular level, that is, entities or types and, where applicable, any further structure other than an unordered set.

Task 3. A formalised relationship or transformation rule is required between a data source, $\mathcal{DS}$, and domain granularity framework applied to the data source, $\mathcal{DG}$, to utilise them both for some particular reasoning task. Therefore, $task^3(\mathcal{DS}, \mathcal{DG}) \rightarrow \mathcal{DS}\ related\_to\ \mathcal{DG}$, where the "$related\_to$" has to be specified. Likewise, the relation between $\mathcal{DS}$ and its selected granulated subsets, *i.e.* $\mathcal{DG}^1$, ..., $\mathcal{DG}^n$, has to be specified.

Task 4. Use a combination of levels of the same or different perspectives on the data source to which a granularity framework has been applied, $\mathcal{DG}$. The result is a subset of $\mathcal{DG}$, $\mathcal{DG}'$; thus, the task is $task^4(\mathcal{DG}, \mathcal{DL}) \rightarrow \mathcal{DG}'$, where $\mathcal{DG}' \subseteq \mathcal{DG}$. Note this is an extension of $task^1$ and $task^2$.

Task 5. $task^4$ has ex^2 as sub-task or can be performed vice versa, therefore they can be combined into one more complex operation: $task^5(d_i^s, d_i^f, \mathcal{DL}) \rightarrow \mathcal{DG}'$.

Note that details and abbreviations for the granularity components are omitted and rendered in natural language text (the tasks are specified precisely and in more detail in Chapters 4 and 5). To examine the interactions between the meta-layer, domain granularity framework, and the data source, a least two types of experiments should be carried out.

Experiment 1. Use the domain D for constructing a particular subject domain granularity framework d_i^f with its perspectives and levels, thus $ex^1(D) \rightarrow d_i^f$, where the meta-level characterisation constrains a particular domain granularity.

Experiment 2. Take the perspectives and levels that have been defined in a domain granularity framework, d_i^f, and apply it to, integrate it with, a particular data source, d_i^s; hence, the second task is to granulate a data source: $ex^2(d_i^s \cup d_i^f) \rightarrow \mathcal{DG}$, where $\mathcal{DG}$ has the original structure of d_i^s with an additional granulation structure and thus that d_i^s and $\mathcal{DG}$ will have the same entities (/types) but additional relations between them.

A birds-eye view of the proposed solution with which the research questions, tasks, and experiments can be addressed will be introduced in the next section.

1.2.2 The proposed solution and its benefits

The intention is to put the Theory Of Granularity (TOG) to use, not as one-off representation for one particular subject domain, but based on a domain- and implementation-independent logical foundation to keep the theory generally applicable to facilitate reuse and interoperability between different knowledge representations and software systems. To achieve this, the high-level overview of the architecture contains four components, as depicted in *Figure 1.1*. On the far right-hand side we have ***foundational semantics of granularity***, structured in a taxonomy of types of granularity. Second from right is the formal ***domain- and implementation-independent theory of granularity*** that formally characterises granularity components such as granular perspective, granulation criterion, granular level, and the relation between levels. When applied to databases, this TOG is positioned orthogonally at the conceptual data modelling layer, or, when applied to conceptual data models or logic-based type-level ontologies, resides at its meta-level. The TOG is instantiated (at the instance or type-level, respectively) for specification of a ***domain***

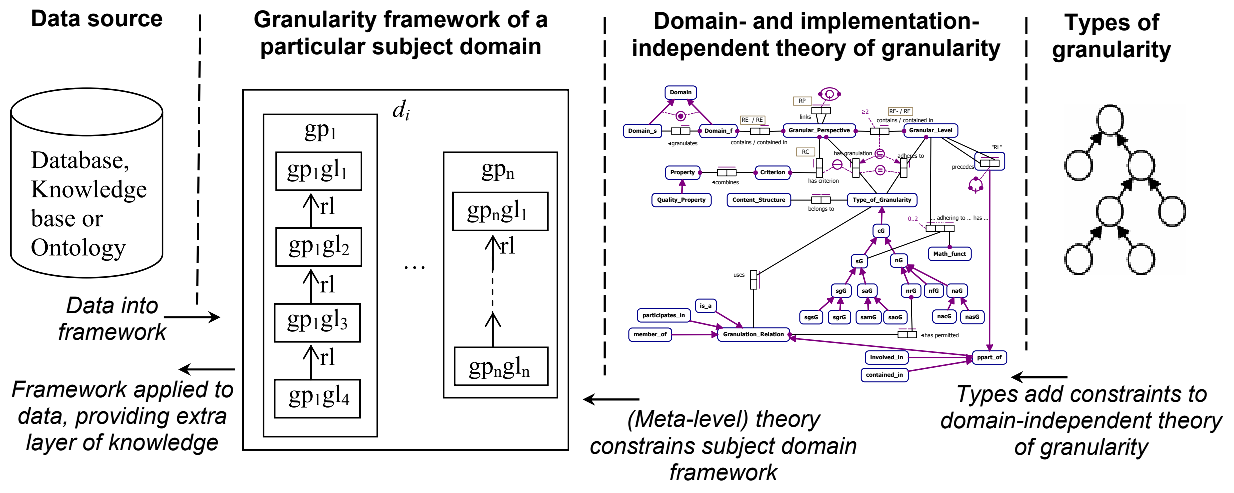


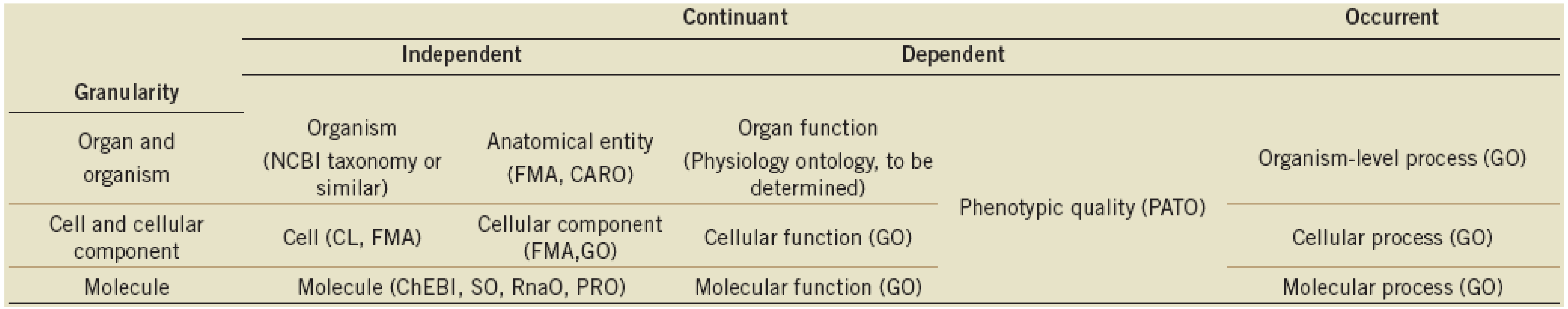
Figure 1.1: Main components of, and related with, granularity. In the domain- and implementation-independent theory of granularity we have components such as D for domain, GP granular perspective, GL granular level of a perspective, and RL as relation between two adjacent granular levels. Subsequently, we have d_i for a subject domain with a particular subject domain granularity framework, gp_i: particular perspective, gp_1gl_1 ... gp_ngl_n: levels defined for each corresponding perspective, and rl_i as relation between two levels in the domain granularity framework.

granularity framework, such as a granular perspective for human structural anatomy with granular levels such as cell, tissue, and organ. This domain granularity framework is then applied to the ***data source***, which is in the scope of the thesis delimited to databases, knowledge bases, and ontologies. The combination of the latter two, an *applied domain granularity framework*, results in granulated information system that can be used for further cross-granular querying and automated reasoning. *Example 1.1* presents a brief illustration.

Example 1.1. Let us start with a practical *domain granularity framework* for the subject domain of organisms, d_1 = Organisms, and put it in the context of the ambitious aims of the OBO Foundry (Smith *et al.*, 2007) whose informal ordering is depicted in *Figure 1.2-A*. d_1 is denoted with a solid rectangle in *Figure 1.2-B*. Within d_1, we can identify several granular perspectives, such as for structural anatomy, gp_1, functional anatomy, gp_2, and processes in organisms, gp_3. (These perspectives may be different for different types of organisms, so that $gp_1, ..., gp_3$ requires further precisiation into "human structural anatomy", "zebrafish structural anatomy" and so forth.) For each perspective, several levels are identified, which are denoted with gp_igl_i, such as the Cellular component-level gp_1gl_4. The *data sources* used for granulation and for populating the granular levels, *i.e.*, that contribute the contents of the levels, are ontologies for this example and indicated between the braces after each level name in *Figure 1.2*, such as the Gene Ontology (GO) and Protein Ontology (PRO). Further, there are relations between these levels and between the perspectives (not drawn). Each d_i, gp_i, gp_igl_i, and the relations between them are constrained by the *domain-independent* TOG so that each domain granularity framework as instantiation of the TOG has to be consistent with respect to the TOG. The TOG, in turn, has constraints coming from the *foundational semantics of granularity* so that each perspective and its levels adhere to one mechanism of granulation, such as anatomical entities granulated by $part\_of$ or by is_a subtyping.
An example for the infectious diseases UoD with sample contents of the granular levels can be found in *Appendix A* and its formal specification in (Keet, 2006c). ♢

These four components together will result in a robust characterisation, reusable implementation, and usage of granularity.

A



	Continuant				Occurrent
	Independent		Dependent		
Granularity					
Organ and organism	Organism (NCBI taxonomy or similar)	Anatomical entity (FMA, CARO)	Organ function (Physiology ontology, to be determined)	Phenotypic quality (PATO)	Organism-level process (GO)
Cell and cellular component	Cell (CL, FMA)	Cellular component (FMA,GO)	Cellular function (GO)		Cellular process (GO)
Molecule	Molecule (ChEBI, SO, RnaO, PRO)		Molecular function (GO)		Molecular process (GO)

B

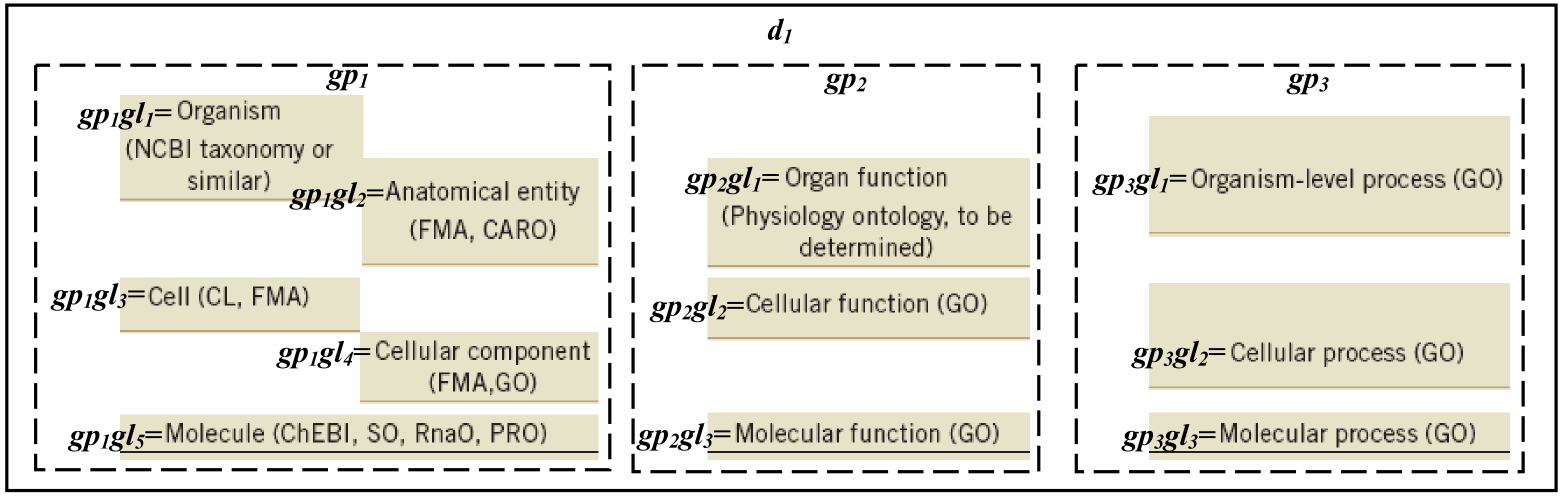

d_1
gp_1
gp_1gl_1= Organism (NCBI taxonomy or similar)
gp_1gl_2=Anatomical entity (FMA, CARO)
gp_1gl_3=Cell (CL, FMA)
gp_1gl_4=Cellular component (FMA,GO)
gp_1gl_5=Molecule (ChEBI, SO, RnaO, PRO)
gp_2
gp_2gl_1= Organ function (Physiology ontology, to be determined)
gp_2gl_2=Cellular function (GO)
gp_2gl_3=Molecular function (GO)
gp_3
gp_3gl_1= Organism-level process (GO)
gp_3gl_2= Cellular process (GO)
gp_3gl_3=Molecular process (GO)

Figure 1.2: A: original OBO Foundry table of ontologies and current coverage of ontologies at different levels of granularity, divided by level of granularity (rows), perspective (columns), and levels' contents that are the ontologies listed between the braces after each level's name (Smith *et al.*, 2007); B: clearer distinctions of the levels (shaded boxes with gp_igl_i) that are contained in the perspectives (indicated with dashed rectangles and gp_i) that, in turn, are contained in the domain granularity framework d_1 (solid rectangle). See *Example 1.1* for further detail. Abbreviations: CARO = Common Anatomy Reference Ontology, ChEBI = Chemical Entities of Biological Interest, CL = Cell Type, FMA = Foundational Model of Anatomy, GO = Gene Ontology, NCBI = US National Centre for Biotechnology Information, PRO = Protein Ontology, RnaO = RNA ontology, SO = Sequence ontology.

1.2.2.1 Contributions to the solution of the problems

Highlighting the three groups of shortcomings in existing approaches described in §1.1.4 to address the problems outlined in §1.1.1-1.1.3, the solution proposed in §1.2.2 shall solve them as follows.
A. *The informal approaches* are not usable or reusable for computation and reasoning, are ontologically inconsistent, and underspecified.

- ***Solution***: Development of a domain- and implementation-independent theory of granularity, the TOG, comprising disambiguation between scale- and non-scale-dependent (quantitative and qualitative) types of granularity and ontologically motivated modelling decisions necessary for the development of a generic, ontologically sound formal foundation of granularity, which enables both precise specification of domain granularity and computational implementation.

B. *The formal and ontological approaches* are more or less compatible partial data-centric theories that address neither what granularity is nor what its components are nor how to use it.

- ***Solution***: Foundational semantics, formal representation, and functions are distinct but connected through a unifying theory that maps to a model-theoretic semantics usable to specify and constrain domain granularity frameworks and necessary for computation and interoperability. By moving from a data-centric and informal-ontological treatment of granularity to the conceptual and logical layers, the granularity components such as level, perspective, and granulation criterion, will be defined, ontologically-motivated modelling constructs proper.

C. *The engineering solutions* are not reusable in the current format beyond the difficult to scale-up software application each one is designed for and are not-interoperable.

- ***Solution***: The domain- and implementation-independent TOG ensures its genericity and widest possible applicability such that multiple divergent uses have a shared common well-founded framework. This can be deployed for multiple application scenarios, such as (biological) database management, cross-granular querying and reasoning, ontology and pathway management with 'browsing in context', fact finding, and provides support for other computational implementations like information retrieval from large corpora. In addition, the foundational semantics simplify consistent implementation of content retrieval functions. Transparent, modular, and extensible functions enable querying and reasoning to manage granulated data sources, such as databases, ontologies stored-in-databases, knowledge bases, and OWL-ontologies, thereby providing a novel and coherent method to analyse information.

1.2.2.2 Advantages of the proposed solution

The additional logic layer provided by the TOG may be integrated with Semantic Web technologies, ontologies, knowledge bases and databases and thereby offers a *new foundational methodology* to analyse data, information, and knowledge though an unambiguous representation of granularity and its accompanying functions for granular querying and automated reasoning. Thus, it also enables recombination of the information to retrieve more knowledge from the same data source and find hidden or implicit information and discover gaps and errors in the data source, that, in turn can provide an impetus for wet-lab research. In addition, it can make the huge amounts of data and large ontologies in the Semantic Web for the Life Sciences manageable and understandable because the user will be able to zoom in to the desired section and level(s), thereby hiding the for the user irrelevant information, yet at the back-end this is still linked and usable for automated reasoning and accessible for other users with different foci.

A combination of maximum expressivity and computability takes advantage of both foundational ontology aspects and engineering usefulness to develop implementations. Domain experts who can benefit include, but are not limited to, researchers in interdisciplinary fields such as translational medicine, metagenomics, and the planned "virtual human" project that intends to simulate *in silico* a functioning human body across all relevant levels of granularity. Data warehouse development, database integration, and ontology linking engineers can use the TOG for structured, *meaningful* connectivity of information sources and multidimensional models to achieve vertical integration and transparency for maintenance of the software systems. With the same theory, GIS hierarchies, among others, may be aligned, implemented in a consistent manner, and have their granularity conversion functions simplified, resulting in easier knowledge and information system management. The TOG makes explicit in an formal unambiguous manner hitherto implicit and underspecified assumptions about granularity, which therefore can also facilitate further ontological investigation.

1.3 Research methodology

1.3.1 Approach

The research methodology consists of four components, depicted in *Figure 1.3*: exploration phase, theory development, experimentation, and evaluation to assess the problems solved. The iterative process has been separated in the explanation below to indicate the conceptual distinction between the types of research activities.

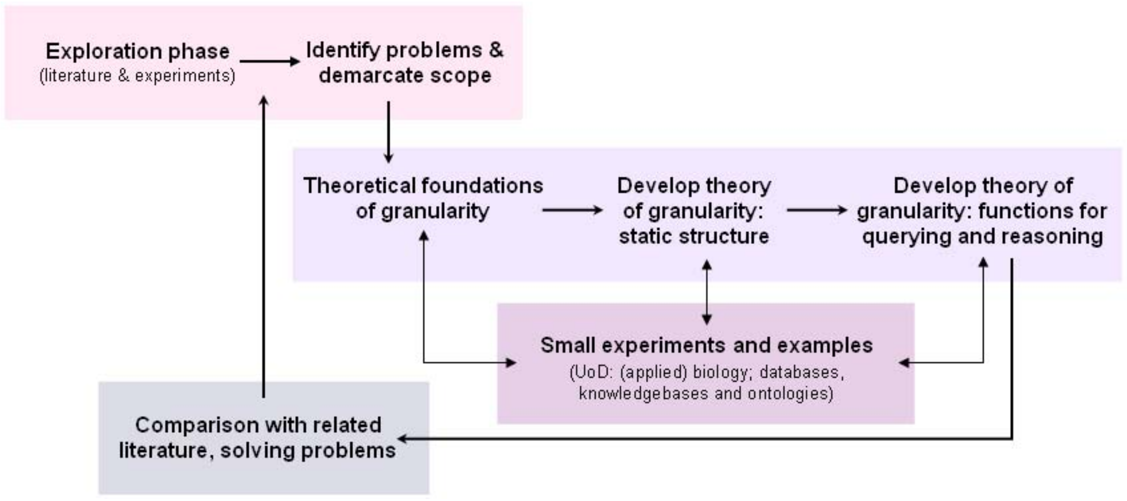

Exploration phase
(literature & experiments)
Identify problems & demarcate scope
Theoretical foundations of granularity
Develop theory of granularity: static structure
Develop theory of granularity: functions for querying and reasoning
Small experiments and examples
(UoD: (applied) biology; databases, knowledgebases and ontologies)
Comparison with related literature, solving problems

Figure 1.3: Overview of the research methodology, which is depicted in sequential and iterative steps and grouped into four main blocks: introductory assessments, theoretical foundations, experimentation, and solving problems.

Exploration phase. The preliminary investigation involves three activities. First, searching for existing literature and applications of granularity to biological domains. Second, experimenting with identifying granularity in smaller subject domains such as human infectious diseases (Keet and Kumar, 2005), blood and its components, the Second Messenger System, and a few examples in ecology. Third, based on the literature and experimentation, assessing reuse and amenability of the existing generic $\mathcal{GMD}$ model and contextual reasoning for constructing a formal granularity framework. However, set-theory and propositional logic appeared to be suboptimal for representing necessary components of granularity. The idea of compartmentalisation in contextual reasoning is useful and returns in the TOG; likewise, $\mathcal{GMD}$'s algebra provided useful input regarding requirements of granular operations. The three activities brought afore additional research questions and aided scoping of the topic.

Development of the theory of granularity. The development of the TOG comprises three phases: examining the foundational semantics of granularity, modelling considerations to construct and develop a formal theory in First Order Logic, and its functions to enable usage of the TOG.

Aspects of the TOG that require further investigation before formalising the theory can be investigated through two principally distinct approaches: bottom-up and top-down. The former takes some subject domain, such as infectious diseases, that is used to (intuitively) specify its perspectives and the levels for each perspective by taking into account relevant existing knowledge bases and/or ontologies, from which one subsequently can generalise to how a domain-independent theory of granularity may look like. The latter has its point of departure theory, including domain-independent ontological investigations, where, after formalisation, it has to be tested with a real subject domain to create the perspectives and their corresponding levels. A caveat with the bottom-up approach, however, is that the specification of granularity may incorporate aspects that are, in hindsight, domain dependent, hence that during testing with a second subject domain this comes to light and that it requires adjustment of the granularity framework. On the other hand, starting from a top-down approach, one either never may make it to define granularity of a subject domain, or have a sound theory that is difficult to apply in practice. I will not take a pure top-down or bottom-up approach, but follow one that is mainly top-down with several bottom-up examples taken primarily from the biology subject domain. The rationale for using motivating examples taken from the biology domain is that (i) granularity is a pressing problem to address in biological data management; (ii) biological granularity has not yet been subjected to a formal analysis; (iii) the nature of the data and information is more diverse, hence challenging, than other subject domains such as operational data of enterprises; and (iv) the examples illustrate and discuss knowledge about reality as published in the biosciences scientific literature, instead of small, sometimes fictional, toy examples.

After formally characterising the TOG, one can define the functions required to use a domain granularity framework effectively, which concerns both the granularity components and the granulated entities (/types). This will be interleaved with experimentation to test the functions on usability and sufficiency for an applied domain granularity framework.

Testing and evaluating the proposed solution. Implicitly, the aims contain the assumption that there exists a 'granular view' on reality—or, more strongly: that reality is granular—with common, domain independent, underlying characteristics that can be represented in one encompassing theory. Consequently, falsification of the TOG may be shown if the theory appears to require exceptions for each subject domain—suggesting that there is no underlying framework after all or that the theory is insufficient—and by inadequacy of applicability to any subject domain—meaning the TOG is not properly grounded. To provide intermediate feedback during definition of the theory and applicability of the theory, the (intermediate) theory is examined with diverse subject domains to test ideas and feasibility manually and computationally. Where applicable, the encountered issues will be used to improve the theory. Further, a comprehensive comparison of the TOG with the main other types of solutions will be conducted to ensure the TOG is a proper generalisation and solves issues encountered with those approaches.

1.3.2 Organisation of the thesis

After this introductory chapter, there are three core chapters, a comprehensive comparison with related literature and implementation considerations, and conclusions. Chapters 2 and 3 can be read relatively independently, but they both precede Chapter 4. The related research (§5.1 and §5.2) and conclusions in Chapter 6 can be read before or after Chapters 2, 3, and 4. A bio-ontologist or bioinformatician may prefer to consult first §1.1.1 or §1.1.2, respectively, related research about biological granularity (§5.3) or engineering solutions (§5.2.3 and §5.2.2.1), a sample of domain granularity (Keet and Kumar, 2005) or experimentation (Keet, 2006b), and read Chapters 2, 3, and 4 afterward.

Chapter 2 focuses on uncovering the foundational semantics of granularity, with four principal dimensions along which one can vary a theory. The result of this investigation is a categorisation of types of granularity organised in a top-level taxonomy (§2.2), which is depicted in *Figure 2.3* and the distinguishing characteristics at the branching point are summarised in *Table 2.1*. This taxonomy lays bare the principal mechanisms of granulation. From the categorisation follows the corresponding structure of the entities (/types) for each leaf type of granularity (§2.3). Several leaf types are illustrated in §2.4, which at the same time introduce some of the issues on representing granularity and problems of granular querying that will be elaborated on and solved in the successive chapters.

Chapter 3 addresses the second of the three main theoretical parts of the thesis: the investigation with modelling considerations and formalisation of the structural components of the TOG—domain, perspective and criterion, and level (§3.2-3.4)—and the relations between them (§3.5-3.7), which is summarised and depicted in *Figure 3.1*. The definition and constraints for each component is preceded by an ontological analysis and justification *why* it is represented in this way and proven in lemmas and theorems where possible. The formal characterization in first order logic is presented in §3.8. Afterward, it will be demonstrated through computational experimentation that the TOG with indistinguishability as primitive relation is satisfiable (§3.8.3) and that the TOG meets the 12 key requirements (§3.10) that were identified in §3.1.

Chapter 4 deals with the dynamic aspects for the TOG, *i.e.*, it has the goals and formal specification of the functions to query TOG-components and retrieve entities (/types) residing in granular levels (§4.2), which are augmented with abstraction and expansion modes and functions in order to move from contents in one granular level to another (§4.3). For these functions to behave as intended, we also need to reason over relational hierarchies, and to this end, the RBox Compatibility Service for automated reasoners will be defined in §4.4. The last section (§4.5) integrates the TOG structural components and the functions by means of an extended granular reasoning example on hormones in the liver, which will be worked out for three types of users. Last, §4.6 summarises the answers to the requirements for research question 4 and tasks 1-4.

Chapter 5 is divided into two main components. First, it presents the comparative assessment of the TOG against recent options considering other formal and ontologically-motivated theories (§5.2), subject domain granularity (§5.3), and abstraction (§5.4); this is summarised in *Table 5.1*. It will be demonstrated that the TOG entails several existing proposals, hence serves as a more generic, unifying, theory that is more scalable and reusable than the partial models and technology-dependent implementations. Second, it discusses limitations of the TOG and casts a look ahead toward opportunities for implementations (§5.5), which goes into some detail addressing task 5 and the two types of experiments.

Chapter 6 concludes the thesis where the research questions from §1.2.1 are answered and tasks solved (§6.1), which includes the answer to the central question "why, how, and where will usage of granularity improve knowledge representation and knowledge management?" on page 207. Current and future research and engineering directions are summarised in §6.2.

Last, there are three appendices. *Appendix A* contains an informal granulation of the subject domain of infectious diseases, which will be analysed and specified precisely throughout the running examples in the successive chapters. *Appendix B* contains the satisfiable first order logic of the TOG as Mace4 input file and two computed proofs to demonstrate feasibility of automation. *Appendix C* has a brief introduction to Description Logics, $\mathcal{DLR}_{ifd}$, and $\mathcal{DLR}_{\mu}$. Due to space limitations, the experimentation with a manual comprehensive and formal description of a domain granularity framework for infectious diseases and computational experimentation of granular querying over the Gene Ontology and Foundational Model of Anatomy have been phased-out into two technical reports and a conference paper (Keet, 2006b,c; Keet and Kumar, 2005).

Part II

Theory of Granularity

Chapter 2

Foundational semantics of granularity

Granularity deals with organising data, information, and knowledge in greater or lesser detail that resides in a *granular level* or *level of granularity* and which is granulated according to certain *criteria*, which thereby give a perspective—also called view, context, or dimension—on the subject domain, henceforth called *granular perspective*. A lower level within a perspective contains knowledge—types, concepts, relations, constraints, or their respective instances—that is more detailed than the adjacent higher level. Conversely, a higher level 'abstracts away', simplifies, or makes indistinguishable, finer-grained details. A granular level contains one or more entities, that is, representations of entity types or their instances; note that granular level is sometimes called granule, but we reserve *granule* to denote a cell or 'part of the pie'. Ideas about what granularity comprises can differ between research disciplines that tend to emphasise one aspect or the other. Several interpretations of granularity and diagrammatical representations are shown in *Figure 2.1*, capturing subtle, but essential, differences in interpretation, representation, and/or emphasis. These differences in types of granularity and their representation are discussed in §2.1, which will be structured in a taxonomy of types of granularity in §2.2. In §2.3 we look at the structures of the contents of a granular level for each of its leaf types. Several examples and issues to resolve are presented in §2.4. An earlier version of §2.1-2.3 was published and presented at the *IEEE International Conference on Granular Computing* (Keet, 2006a).

2.1 Granularities

The possible interpretations of *Figure 2.1* will be discussed in this section. Successively, the emphasis will be on entity types & instances, the relation between levels and their contents, the perspective & criteria for granulation, and, last, on a consequences of choosing a particular formal representation. The main distinctions are summarised in §2.1.5.

2.1.1 Emphasis on entity types and their instances

Figure 2.1: A1-A5. The circles A1-A4 in *Figure 2.1* are examples where the circles can represent the subject domain or a granular level. This gives four possible interpretations.

i. If it represents a subject domain, then the four respectively five parts in A1 (A2) are finer grained than the circle, *i.e.* each one provides more detail about the domain than a plain circle (C1) and with $\prec$ denoting a strict order, then A1 $\prec$ C1 and A2 $\prec$ C1 hold.
ii. If it represents a granular level, it shows the four (A1) respectively five (A2) granules resulting from granulating the contents of a level where each level is disjoint exhaustively granulated (fully divided). Without further clarification, it cannot be excluded that one of the granules denotes Everything else, or, if there is always one entity (/type) (A6) or possibly more (A7) entities in each granule (see also below on A6 and A7).

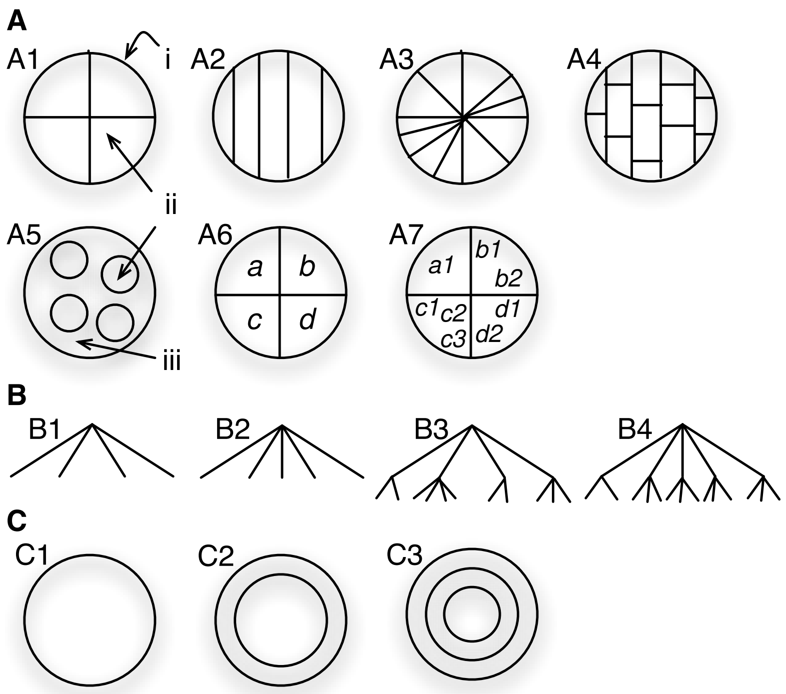


Figure 2.1: Several graphical representations of granularity. A1/A5: (i) is the domain or a level with (ii) a partition and (iii) possible non-included rest depending on the interpretation. B1-B4 may be alternative representations of A1-A4. See text for explanation.

iii. If the circles A1 and A2 are the same domain or granular level, then a different grid corresponds to granulation according to different perspectives or criteria on the same domain.
iv. If A3 (resp. A4) is at a lower level of granularity compared to A1 (A2), then A3 $\prec$ A1 (A4 $\prec$ A2, respectively) and the granules of A1 (A2) are fully divided into more granules in A3 (A4), thereby representing finer-grained divisions that can be made when more details are taken into account, but which are indistinguishable at the level of A1 (A2).

From the possible interpretations assigned to A1-A4, A5 suggests that it contains four granules and empty space that falls outside the four nested smaller circles. However, it equally may be an inappropriately used ER diagram or Venn diagram requiring additional clarification to disambiguate its exact meaning regarding levels and granulation, such as if the four circles are disjoint exhaustive or not. Important to realise is that *Figure 2.1-A* has *implicit* the granular perspective with its criterion how to granulate: there is some criterion x why A1 has four parts and another criterion y such that A2 has five, but x and y are assumed in the representation and the criterion for granulation—hence also the granular perspective—is omitted.

Figure 2.1: A6 and A7. The two circles containing $a, ..., d$ can represent a fundamental distinction on how to model granularity. Both A6 and A7 represent a populated A1, but depending on the interpretation of the figure, $a, ..., d$ in A6 can denote entity types or instances, and the indexed $a_1, ..., d_2$ in A7 then denote instances.

i: If a, b, c, and d are entity types, then
  1) *without* granulation as in C1, $\{a, b, c, d\}$ is an unordered set of entities of the domain of interest, thus $a, b, c, d \in D$ although in C1 they are indistinguishable;
  2) *with* granulation, as in A6, $\{a, b, c, d\}$ are distinguishable and found to be distinct. Moreover, there is *exactly one* entity type in each granule, which can be either by design—one granule, one entity—or accidental in that there may be an e that also fits in the granule where, *e.g.*, a resides but is not included due to either unintended

omission or known incomplete coverage of the domain.

ii: If a, b, c and d in A6 are instances, then either
   1) the current 4 granules are accidental in the sense that at a time $t_1 > t_{present}$ there may be more or less than 4 granules because at least one of the objects may have ceased to exist or a new one added, or
   2) at t_1 where an object has ceased to exist, there is an empty granule; hence, one commits to the original granulation for a set of instances. Thus, it can be that at time t_2, where $t_2 > t_1$, that granule is not empty anymore.

   In both options, the outcome of granulation is dependent on the instances present at the time of identifying or creating the granules.

iii: If it is the case of i-2, then either
   1) A7 shows the corresponding instances of the entities in A6, or
   2) there was an unordered set of instances $\{a_1, ..., d_2\}$ that were grouped according to some criterion. Based on their similarity, they are grouped into their corresponding classes as in A6 that may or may not correspond to universals.

iii-1 and iii-2 are different only in the starting point, one being the entity types and the other instance-motivated. Some of the points in this and the previous paragraph will be illustrated with an example; the new topic it introduces about measurement scales will be discussed afterwards.

Example 2.1. Let the circles A1 and A2 each represent a human population, where its parts (granules) are labelled with, respectively:

a1: (Single, Married, Divorced, Widowed)
a2: (Newborn, Child, Adolescent, Adult, Elderly)

Hence, A1's criterion is Marital status and A2's criterion can be Life stage. Considering A7, let $a_1, ..., d_2$ be (not necessarily exhaustive) instances of months, then each granule could represent the quarters:

a7: (Quarter1, Quarter2, Quarter3, Quarter4)

This indicates that the domain or coarser-grained level C1 is Year; if there were semesters, then an intermediate level x where A6 $\prec$ x $\prec$ C1, with two granules for Semester1 and Semester2. More importantly, this granulation uses a 'smallest element': in this closed-world assumption, Month is chosen as the type of *Urelement* that is aggregated in such a way that each aggregate denotes a set extension of a class that is also a universal. Another example of this concerns phone points as *Urelement*, where the phone point instances are granulated into Cell, Land line, Direct line, and PABX where a class was created from sets of phone points (Kamble, 2004). Thus, this relies on set theory for representing granularity, where in the case of Kamble (2004), a class neither has to be the set-extension of a universal nor has to have defined necessary and sufficient properties.
Another aspect of the figures in A is, *e.g.*, granulating temperature using a measurement scale in a lower grain size of integer degrees 19, 20, 21, for isotherms and moving up to a higher level where Isotherm20 suffices with coarser-grained rounding off (see also *Example 2.3*). In both cases, the *same* thing is granulated with more or less detail. This interpretation is prevalent in GIS for making a grid over land plots (see §5.3.2 and references therein). Analogous is the case with spatial and time scales that for, *e.g.*, humans cover factor differences of 10^{15} for spatial and 10^9 for time, ranging from proteins (in nm) to height of humans (in m) and from μs for Brownian motion to decades for lifespan of a human, respectively (Hunter and Borg, 2003). ♢

It is essential to note that when granulating according to a *scale*, one defines a *smallest unit* or a *standard unit* from which other levels are generated using a mathematical formula, according to which the domain is to be granulated. This is another, less problematic, granulation compared to trying to figure out the relation between Tissue and Cell or Cell and Organelle, if and how developmental stages of an organism have granularity, or characterising the type(s) of components of the Second messenger system that comprise distinct objects, its parts, processes, events etc. The second section of *Example 2.1* above deals with larger or smaller parts of arbitrary scales, but each level still concerns values according to the same arbitrary scale. Non-scale-dependent finer grains involve *other* types of entities, as, for instance, a biological cell is not equal to a tissue slice of 0.05mm thin. The latter puts a higher emphasis on the criterion for granulation and its levels than on the entities and instances one may find at a certain level of detail. This is especially useful for biological granularity, because of the incomplete knowledge of the domain that prevents disjoint exhaustive categorisations of its contents and its emphasis on qualitative information and knowledge as opposed to quantitative data. Granularity comprises both methods, but they involve fundamentally different granularity between coarser and finer levels.

In addition, while a1 and a7 in *Example 2.1* may seem alike, they are not: members of a population are different from elements in a set. With the latter, granularity *depends* on its instances: with another set of instances, the levels of granularity, ordering of the elements in a level, and perspectives may turn out to be different, and therefore can be time-inconsistent. In contradistinction, granulation involving an entity type identified with a collective noun like (human) population and how one can group the *members* of the population: from time t_0 to a later time t_1 the instances (members of the human population) have changed, but this does neither affect the principle/criterion nor the levels.

2.1.2 Emphasis on relation between entities and levels

Continuing with the possible interpretations, we proceed to ***Figure 2.1: B1-B4***. Two first basic observations are that

i. B1-B4 correspond to A1-A4, where the top equals the circle and each edge leads to a node (cell) at the end of each line. The tree structure is favourable when depicting multiple granular levels, because it is more concise than the figures in A (compare A3 and A4 to B3 and B4, respectively).
ii. The lines in B emphasise the relation between levels of granularity, or at least between its entities (/types) residing in coarser- and finer-grained levels.

Point ii highlights the point of departure or focus—the relations involved—but is ignorant about which types of relations are relevant for granularity, both regarding the relation between the entities in different levels and how granular levels relate to each other. Committing to one type of relation or the other can imply an ontological commitment how one perceives granularity (see §2.1.4); in particular, partonomic versus taxonomic (generalisation/specialisation) granulation that are used or considered regarding informal biological granularity (a.o., Degtyarenko and Contrino, 2004; Fonseca *et al.*, 2002; Zhang *et al.*, 2002; Pandurang Nayak and Levy, 1995; Kiriyama and Tomiyama, 1993). Such deliberations for one type of relation or the other is a distinct issue from using arbitrary scales and puts in the background the entities (/types) in each level and how the contents is allocated to a level. In addition, it may be that there is a taxonomic division for contents within a level, as depicted in *Figure 2.5*, in the sense of two orthogonally positioned granular perspectives. Using the is_a relation for granulation means that each layer in the tree with the same depth should correspond to a granular level. However, this does not necessarily hold for granularity as perceived by domain experts. For instance, 'folding' deals with polysemy and underspecification in language and the so-called black-box usage in biology, which is illustrated for cell physiology and book ordering in *Example 2.2*.

Example 2.2. Combining different types of entities and relations between granular levels may be useful in particular for abstracting biological complex types like Second messenger system or MAPK cascade. With the former, its processes such as Activation, GTP-GDP

exchange, α-subunit release, states like Activated, and components such as Hormone receptor, G_s protein, and cAMP, collapse together into one entity type Second messenger system (Stryer, 1988). MAPK cascade is already used as a module in systems biology that at a higher level of abstraction is treated as a black box, containing (sub-)processes, inputs/outputs, parameters and their values, etc. (Sontag, 2004).
A variant not uncommon in hierarchical modeling of conceptual data models is to have, *e.g.*, an entity type Book order, where the ordering consists of several procedural processes and entities involving, among others, Billing, Paying, Supplier, and Shipment. $\diamondsuit$

As the example shows, that what is a type of endurant at the higher level of granularity, is composed of a combination of endurant parts, processes, and states. For an implementation, it is possible to separate the different types of components into different granular perspectives and levels, but this does not capture what is meant with the higher-level entity type like Second messenger system. Put differently: if separated in a granular perspective of structural components and another one for processes, the "Second messenger system" at the higher level in each perspective is only a *partial* representation of the entity type. If one allows relating levels of granularity by folding with type shifting (Mani, 1998), then this complicates what the parts are and how they relate to the whole, but on the other hand, saves integrating or linking granular perspectives. Either way, the relation-view between levels and between the entities (/types) is there; which relations can be used with granularity and how will be addressed in §3.6.

2.1.3 Emphasis on the perspective and criteria for granulation

Last, ***Figure 2.1: C1-C4*** show three levels of granularity where a smaller circle denotes a finer-grained level. This is unlike the Russian dolls analogy, where a similar smaller doll is contained in the larger one, but alike dissecting an organism to see what organs are inside, zooming in on parts of the organ, the tissue, cells and so forth. Thus, the parts are different types of entities and one uses, *e.g.*, human structural anatomy to identify finer- and coarser-grained levels that contain, respectively, all types of organs, tissues, and cells. In addition, each level has its distinguishing characteristic, that is, the property of a level is emphasised. In contrast with the first approach analysed in §2.1.1, here one looks first at the property or properties, decides on the levels, and subsequently allocates entities to the levels based on the pre-selected properties.

Less explicit is how these properties relate to each other, except that it must relate in some way to both the finer- and coarser-grained level. I call the unifying rationale that links these properties the *criterion*. For instance, within the domain of human anatomy, one can granulate according to different criteria, such as structural anatomy, functional anatomy, or the processes they are involved in. Subsequently, one can identify granular levels according to certain properties that have to do with structural aspects or with containment and so forth. Observe that this entails a commitment to a granulation relation.

Because of the property-focus, C1-C3 do not bear any information if the cascaded granulation of the contents in each level is disjoint or complete. It does suggest that each level of granularity has one *type* of entity, such the outer circle representing the Cell-level containing cell types, with a smaller circle the Organelle-level containing entities such as Endoplasmatic reticulum and Lysosome. Although these examples may indicate the physical size is a criterion, this is not necessarily the case. For example, if one were to represent the phylogenetic tree in the diagrammatic representation of B or C, a Mammal-level has no physical size associated with its definition. More generally, with the emphasis on the perspective and criteria for granulation, this approach is more useful for non-scale dependent granularity.

2.1.4 Emphasis on formal representation

The difference between scale and non-scale dependency mentioned in the previous sections roughly fits with Sowa's (2000) epistemic and intentional granularity. Sowa bases his three types of granularity on Peirce's three categories of Firstness, Secondness and Thirdness.

1. Firstness maps to actual granularities with axioms for discrete, continuous or lumpy aggregates (Sowa (2000) and below) and concerns the entities that populate a level.
2. Secondness for granularity uses epistemic logics involving measurements, including error ranges, or axioms & measurements (Sowa, 2000) and corresponds to the scale-dependent granularity[1].
3. The Thirdness for granularity, corresponds to intentional, which requires a three-place predicate relating "an agent *a* to an entity *x* for a reason *r*" (Sowa, 2000), where a reason *r* depends on the perspective on takes.

Depending on how one uses granularity in a subject domain, devising *levels* does not require asking oneself questions if entity *x* has at least one atom as part, if there is an infinite regress of parts that is cut at the lowest level defined, or if the entity is lumpy (point 1 above), but the allocation of entities to a given level does use aggregates and entities. More precisely, in mereology an $Atom$ is an entity that has no proper parts (2.1).

$$Atom(x) \triangleq \neg\exists y(y < x) \tag{2.1}$$

Then, there are three kinds of aggregates (with "$\leq$" as part-of and "$<$" as proper-part-of). First, *Discrete*: everything has at least one atom as part (2.2); thus, that things can be subdivided up to the point where nothing is left but atoms.

$$\forall x\exists y(Atom(y) \land y \leq x) \tag{2.2}$$

Second, *Continuous*: everything has at least one proper part (2.3), which permits indefinite subdivision, implying that there are no atoms,

$$\forall x\exists y(y < x) \tag{2.3}$$

Third, *Lumpy*: some things are atoms, some are continuous (2.4). (Sowa, 2000).

$$\exists x Atom(x) \land \exists y\forall z(z \leq y \rightarrow \exists w(w < z)) \tag{2.4}$$

Thus, representing granularity using mereology may have *but does not require* atoms as 'ultimate part' or Urelement that is normally used for set theory-based granularity. Observe also that Urelement can, in fact, be defined in terms of atoms, where Urelement is the "atom at the finest-grained level in a granulation hierarchy", provided that 'granular level' and 'granulation hierarchy' are defined properly (see Chapter 3). Both set theory and mereology have their advantages and disadvantages for representing granularity that better approximates reality. Ease, difficulty, or even impossibility, to identify an Urelement is illustrated in the following example.

Example 2.3. Taking calendar entities and set-theory based granularity, entities like Week, Month, Quarter, and Year may be built from a chosen Urelement Day and can be represented by distinct sets of days. However, taking isotherms, then what can or should be chosen as Urelement? For instance, one could use Degree as smallest element and build up coarser-grained isotherms. However, with a set as the extension of Isotherm20, like $\{15, 16, 17, 18, 19, 20, 21, 22, 23, 24\}$, and where Isotherm20 is a subtype of Isotherm that has other subtypes like Isotherm30 etc., there are some problems: the extension is not the entity type and the numbers are not degrees but integers (see also Johansson, 2004b).
With biology, identifying or choosing a smallest element is even more challenging. Say, a general practitioner who does not take into account smaller entity types than tissue and makes Tissue the Urelement (atom) for the lowest level, then this means that all higher-level structures are composed of tissue *only*: we *know* this is biologically incorrect and

[1]Sowa also mentions belief, which is intentionally omitted here.

thereby not a good representation of nature. In addition, if one takes the lowest level of the FMA (2003)—*i.e.*, Biological macromolecule, which does not include other molecules without which a human body cannot survive, such as H_2O—and deem that the coarser-grained level Body is the set of all its macromolecules, then a body changes identity each time a molecule is synthesised/metabolised, which happens continuously, resulting in the situation that a body has no enduring identity but is in flux[2]. Likewise, entomologists study the same ant colony over time, even though ants were born and died. More generally, regardless if a set-theoretic logical theory or model is logically valid and corresponding knowledge base in a legal state, basing reasoning on represented knowledge that is not adequately grounded in the reality it aims to represent can lead to undesirable outcomes for patients, ecosystems and the like. $\diamondsuit$

Both ways of representing granularity, through is_a with set theory and mereological $part\_of$, are from a logical viewpoint mostly interchangeable (Pontow and Schubert, 2006), but not from an ontological viewpoint as the intended meaning captured in a formalisation is distinct. This difference has been recognised earlier by Salthe (2001) and are not considered to be *competing* interpretations of granularity, but *both* considered as distinct, valid ways of understanding granularity. One does not have to force one type of granularity in the straightjacket of the other; doing so anyway always will deprive another type of granularity from representing nature as accurate as possible. Capturing granularity in one's preferred version of logic is restrictive, unless clear semantics of the intended meaning is provided with it, *i.e.* the formalism is *onto*logical in nature instead of only a logical theory (LOA, 2005).

Moving to thirdness, reason *r* (point 3 above) may be useful for granular perspectives in non-scale dependent granularity: although it is not necessarily modelled as a triadic predicate, separating and reusing the reason, or criterion, benefits scaling up the granularity framework. For instance, we have Wine as an amount of matter (continuous aggregate) versus being a composition of different of molecules to extract the different types of tannins from the wine (discrete aggregate). The fundamental difference of the latter with arbitrary scales may be clear. Such differences in types of granularity have, at the meta-level, a major effect on granulation relation between entities (/types) residing in different granular levels, because scale-dependent levels are identified and ordered according to a combination of a property and an arbitrary scale whereas non-scale-dependent levels are ordered according to a combination of properties where level identification is less straightforward. Properties will be analysed in detail in the next chapter.

2.1.5 Main differences concerning approaches toward granularity

An attempt to merge the graphical representations depicted in *Figure 2.1* is shown in *Figure 2.2* for two granular perspectives, where the top ellipses are coarse-grained granular levels with less detail in larger cells—that is, conceptually more encompassing entities—than the two finer-grained granular levels.
Thus, one can identify the main differences in types of granularity, and the perception thereof:

1. Arbitrary scale versus non-scale-dependent granularity;
2. How levels, and its contents, in a perspective relate to each other;
3. Difference in emphases, being entity-, relation-, or criterion-focused;
4. The perception and (mathematical) representation, such as based on set theory versus mereology.

[2]This view is not entirely uncommon (Hawley, 2004), but topics like the four-dimensionalism of perdurantists is outside the scope.

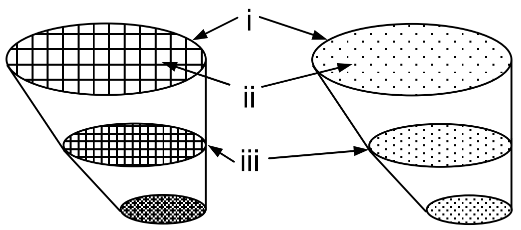


Figure 2.2: Merging emphases on aspects of granularity (A-C of *Figure 2.1*): top ellipse (i) is a coarse-grained granular level granulated with less detail in larger cells or coarser-grained entities (ii) than in the finer-grained granular level (iii).

These differences do not imply one cannot switch from one to the other, represent one way into another, or let them work together orthogonally. When analysing of some subject domain, one apparently seamlessly shifts perspectives and alternately emphasises the criterion used for granularity and the partitioning within a level itself, or taking a type versus instance-inspired approach. Teaching a computer program to do so, however, requires a formal approach to implement it in a consistent manner that can be used and reused across different types of software applications.

2.2 Taxonomy of types of granularity

Given the types of granularity informally introduced in the previous paragraphs, they will be structured into a taxonomy of types of granularity now, which can be seen as a meta-layer in the TOG, which is positioned on the right-hand side in *Figure 1.1*. Hereby it is emphasised that there is not one granularity, but several types—mechanisms of granulation—that have additional constraints extending the **c**ore **G**ranularity, **cG**, as root. *Figure 2.3* shows the top-level taxonomy where the meaning behind the labels of the types are important (it might benefit from meaningful names other than mnemonics as labels). The types are summarised in this section and the characteristics for distinguishing between the types of granularity at each branching point are presented in *Table 2.1*. The structures among the entities within a level of each leaf type will be described in more detail in §2.3.

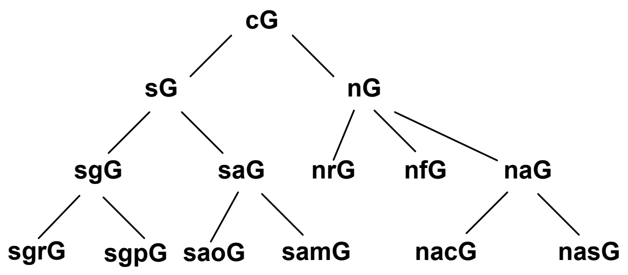


Figure 2.3: Top-level taxonomy of types of granularity.

- **cG**: **c**ore **G**ranularity, consisting of the basic characteristic common to all considered types of granularity and basic constraints.
- **nG**: **n**on-scale-dependent **G**ranularity, where other types of entities reside in each finer-grained level; subtypes have additional constraints and granulation relations.

Table 2.1: Distinguishing characteristics at the branching points in the taxonomy of types of granularity depicted in *Figure* 2.3.

Branching point	**Distinguishing feature**
sG – nG	scale – non-scale (or, roughly: quantitative – qualitative)
sgG – saG	grain size – aggregation (or: scale *on* entity – scale *of* entity)
sgrG – sgpG	resolution – size of the entity
saoG – samG	overlay aggregated – entities aggregated according to scale
naG – nrG – nfG	semantic aggregation – one type of relation between entities in different levels – different type of relation between entities in levels and relations among entities in level
nacG – nasG	parent-child not taxonomic and relative independence of contents of higher/lower level – parent-child with taxonomic inheritance

- **nrG**: levels of **n**on-scale dependent **G**ranularity are ordered according to one type of relation in a perspective; *e.g.*, (structural-)$part\_of$, (spatially-)$contained\_in$. The primary types of granulation relations will be identified in §3.6.2.
- **nfG**: levels of **n**on-scale dependent **G**ranularity are ordered by simultaneous **f**olding ≥ 2 different (types of) entities, such as folding events and states, and consequently folding relations between those entities, upon going to a coarser-grained level; *e.g.*, the 'black boxes' in biology such as the Second messenger system, the Abstraction Hierarchy, ER clustering.
- **naG**: **n**on-scale-dependency with some form of **a**ggregation.
- **nasG**: **n**on-scale-dependency using **a**ggregation of the **s**ame collection of instances of one type that subsequently can be granulated using semantic criteria. The class at a lower level is a subtype of the class at the coarser-grained level; *e.g.*, a collection of phone points and at the finer-grained level we have land-line and mobile phone points.
- **nacG**: **n**on-scale dependency using **a**ggregation attributed to the notion of an entity generally labelled with a **c**ollective noun, has an existing semantics, the instances of the aggregate are different from instances of its members, and a change in its members does not affect the meaning of the whole; *e.g.*, Population with Organisms of type x, or Team as aggregate of its Players.
- **sG**: **s**cale-dependent **G**ranularity where the contents is structured according to a more or less obvious arbitrary scale; for **cG** and additional constraints. For instance, calendar hierarchy, rounding off of altitude lines on a cartographic map.
- **sgG**: **s**cale dependency with relation to **g**rain size, or resolution, scale-based zooming.
- **sgrG**: **s**cale dependency, taking into account **g**rain size with respect to **r**esolution; *e.g.*, Cell wall represented as line, as lipid bi-layer, and as three-dimensional structure, or a building on cartographic maps as polygon or as point depending on the resolution of the map.
- **sgpG**: **s**cale dependency, with **g**rain size and **p**hysical size of the entities; *e.g.*, sieves with different pore sizes that retains the entities or lets them through, a Euro coins separator, or two objects touching each other (e.g. wallpaper and the wall).
- **saG**: **s**cale dependency with some form of **a**ggregation and its immediate parts are of one type.
- **samG**: **s**cale dependency and using **a**ggregation of the **s**ame collection of instances of the same top type or Urelement that subsequently can be granulated in various ways at lower

levels of detail using a mathematical function; *e.g.*, Second, Minute, and Hour, where 60 seconds go in a minute.

- **saoG**: **s**cale-dependency the carving up of the same entity at each level according to a coarser or finer grid of which the cells can be **a**ggregated and lay **o**ver the representation of a material entity[3]. For instance, the earth with its isotherms, where the isotherms are in steps of 10 degrees, 5 degrees, 1 degree detail[4].

The current version of the taxonomy of types of granularity roughly fits the distinction between quantitative and qualitative features and added versus inherent granularity. At some branching point in the taxonomic structure more than one desideratum is used to distinguish between the subtypes, which can be remedied by introducing 'fillers' to ensure only one desideratum at a time is added. However, these fillers are not used anyway and unnecessarily enlarge the top-level structure, and therefore have been omitted. Other categorisations of types of granularity are conceivable, but these are much less consistent and structured. For instance, aggregation versus 'granularity by other means', instead of the (non) scale dependency because one has both **saG** and **naG** each with their subtypes. However, using aggregation emphasises the internal structure of a level, how entities and instances relate, or is implementation-driven, but it does not take into account the properties *how* to make the distinction between types because having a remainder group of types of granularity does not capture the semantics adequately and then the same desiderata would re-appear in both branches thereby creating redundancy. In addition, using aggregation as distinguishing criterion *implicitly* makes a distinction between set theory and mereology, but this should be a representational issue only. Last, aggregation is underspecified, both with respect to its ontological nature and variants in implementations.

There are two other options to categorise the types, being entity-focussed and human-centred, which might aid understanding of the types. However, because several types can be categorised twice, it does not serve to provide unambiguous classification[5]. In contrast, the proposed taxonomy takes a purely *semantic*, ontological, approach, thereby also separating restrictions of (formal) representation and implementation from the intended meaning.

The basic taxonomy of types of granularity will be included in the formal characterisation of the TOG in Chapter 3. Deploying different types of granularity for conceptually elegant granular querying and information retrieval will be addressed in Chapter 4.

2.3 Structure of the contents for different types of granularity

A consequence of different types of granularity is the influence on the structure of the contents, independent of the actual data source. For instance, completeness and disjointness of the subject matter: a grid is automatically disjoint and, depending on the level and implementation decisions, complete, which does not necessarily hold for contents of levels that has a **nG**-type of granulation. It also affects the loading of the contents and reasoning over it, which will be addressed in Chapter 4. A first formalization of the structural aspects of the contents of a level for each leaf type was presented in (Keet, 2006a). The drawback of that formalization is that it requires many primitives and, as it turns out, we can avail of the TOG to provide a more elegant formal characterization; however, the TOG will only be formally characterised in the next chapter. Put differently,

[3]The grid is over the representation of a *material* entity, because one cannot put a grid over the representation of a non-material entity like an organisation, but one can do this with e.g. a lake—that is, with GIS objects such as representations of entities on cartographic maps.

[4]This does not consider roughness or fuzziness of the measurement, which is an orthogonal issue.

[5]For instance, entity-focussed: i) Different real-world entities in different levels with **sgpG**, **nrG**, **nfG**, **nacG**, ii) The same real-world entities in different levels, reordered/restructured with **sgrG**, **samG**, **saoG**, **nasG**, **nrG**. Human-centred: i) Human-imposed granularity with **sgrG**, **sgpG**, **samG**, **saoG**, **nacG**, **nasG**, and ii) Not necessarily human-imposed granularity with **sgpG**, **nrG**, **nfG**, **nacG**, **nasG**

- The TOG does not have as prerequisite a comprehensive formalization of the taxonomy of types of granularity;
- A precise characterisation of the types of granularity is very useful to grasp several modeling decisions that will be made for the TOG, but not mandatory.
- Only an implementation of comprehensive granular reasoning requires both.

Therefore, at this stage, we only formalise aspects where possible and clearly indicate which kind of predicates one would need in a theory of granularity (section §2.3.2). To be precise about the formalisation in this chapter as well as successive chapters, we first need a few preliminaries about first order logic, universals and particulars, and several basic functions and relations, which are given in §2.3.1.

2.3.1 Preliminaries

Both the structure of the content as well as the remainder of the TOG is formalised in first order predicate logic (FOL). Therefore, the basics of FOL is given first, including what is meant by a *theory* in the logical sense and model theoretic semantics. Put differently, if all axioms of the TOG in this and next chapter would be removed, we still have a theory in the common usage of "theory". On the other hand, with the axioms that adhere to the symbols, syntax, and semantics of FOL, we not only have a *formal* notion of theory, but also—moreover—an unambiguous characterisation of the theory of granularity *and* the machinery to formally prove interesting properties of the theory and a clear path toward computational implementations of the theory.

The FOL introduction is followed by two general conventions that apply to the axioms and definitions and by a few basic relations and functions for content retrieval and level assignment.

First order logic. The lexicon of a first order language contains:

⋆ Parentheses: (and);
⋆ Connectives: $\neg$ (negation), $\rightarrow$ (implication), $\leftrightarrow$ (double implication/equivalence), $\wedge$ (and), and $\vee$ (or);
⋆ Quantifiers: $\forall$ (universal) and $\exists$ (existential);
⋆ Variables: $x, y, z, ...$ ranging over particulars;
⋆ Constants: $a, b, c, ...$ representing a specific element;
⋆ Functions: $f, g, h, ...$, with arguments listed as $f(x_1, ...x_n)$;
⋆ Predicates: $R, S, ...$ with an associated arity.

Later, constants denoting universals will be used (GL for granular level etc.) as well as variables ranging over universals ($\phi, \psi, ...$) that are used as syntactic sugar (see §3.8 for details).

Considering functions, a function in FOL has the form $f : A \mapsto B$, with f denoting the function, A the set of arguments, and B the set with the result; *i.e.*, with input $a \in A$, $f(a)$ yields an output $b \in B$. The set of $a \in A$ that produces an output is also called the *domain* of the function and the set of all $b \in B$ such that $b = f(a)$ is called the *range*. (Hedman, 2004). An alternative notation for functions is to rewrite a n-ary function as an $n+1$-ary predicate; this will be introduced here as well, because it will be used later in this chapter and in chapter 4. Function f has the form $f(A_1, ..., A_n) : R$, where $A_1, ..., A_n$ are the arguments, and R the type of the result, then this corresponds to an (n + 1)-ary predicate $f_{A_1,...,A_n}$, with n arguments and the last one, r, represent the result. This predicate must satisfy:

$$\forall a_1, ..., a_n, r(f(a_1, ..., a_n, r) \rightarrow \bigwedge_{i=1}^{n} A_i(a_i) \wedge R(r)) \tag{2.5}$$

$$\forall a_1, ..., a_n, r, r'(f(a_1, ..., a_n, r) \wedge f(a_1, ..., a_n, r') \rightarrow r = r') \tag{2.6}$$

$$\forall a_1, ..., a_n, r(f(x, a_1, ..., a_n, r) \rightarrow R(r)) \tag{2.7}$$

where (2.5) ensures correct typing of the arguments, (2.6) says that the object with given arguments determines in a unique way the return value, and (2.7) ensures the correct type of result. This said, we can use the simplified way of representing functions and functional relations to avoid repetitive axioms that are essentially the same for the functions introduced in this chapter, where a function like $grain(x)$ is constrained as above.

Regarding the syntax, the following three definitions and rules apply. A *term* is inductively defined by two rules, being

1: Every variable and constant is a term.
2: if f is a m-ary function and $t_1, ...t_m$ are terms, then $f(t_1, ...t_m)$ is also a term.

DEFINITION 2.1. *An* atomic formula *is a formula that has the form $t_1 = t_2$ or $R(t_1, ..., t_n)$ where R is an n-ary relation and $t_1, ..., t_n$ are terms.*

The three rules are:
R1. If ϕ is a formula then so is $\neg\phi$.
R2. If ϕ and ψ are formulas then so is $\phi \wedge \psi$.
R3. If ϕ is a formula then so is $\exists x \phi$ for any variable x.

DEFINITION 2.2. *A string of symbols is a* formula *of FOL if and only if it is constructed from atomic formulas by related applications of rules R1, R2, and R3.*

A *free variable* of a formula ϕ is that variable occurring in ϕ that is not quantified. We then can introduce the definition of *sentence*.

DEFINITION 2.3. *A* sentence *of FOL is a formula having no free variables.*

We now proceed to the semantics of FOL. To say whether a sentence is true or not depends on the underlying set and the interpretation of the function, constant, and relation symbols. To this end, we have structures. A *structure* consists of an *underlying set* together with an *interpretation* of functions, constants, and relations. Given a sentence ϕ and a structure M, M *models* ϕ means that the sentence ϕ is true with respect to M. More precisely,

DEFINITION 2.4. *A* vocabulary *$\mathcal{V}$ is a set of function, relation, and constant symbols.*

DEFINITION 2.5. *A $\mathcal{V}$-*structure *consists of a non-empty underlying set Δ along with an interpretation of $\mathcal{V}$. An interpretation of $\mathcal{V}$ assigns an element of Δ to each constant in $\mathcal{V}$, a function from Δ^n to Δ to each n-ary function in $\mathcal{V}$, and a subset of Δ^n to each n-ary relation in $\mathcal{V}$. We say M is a* structure *if it is a $\mathcal{V}$-structure of some vocabulary $\mathcal{V}$.*

DEFINITION 2.6. *Let $\mathcal{V}$ be a vocabulary. A $\mathcal{V}$-*formula *is a formula in which every function, relation, and constant is in $\mathcal{V}$. A $\mathcal{V}$-*sentence *is a $\mathcal{V}$-formula that is a sentence.*

Thus, when we say that M *models* ϕ, denoted with $M \models \phi$, this is with respect to M being a $\mathcal{V}$-structure and $\mathcal{V}$-sentence ϕ is true in M. With this, we can proceed to the basic notions of model theory, that is, the interplay between M and a set of first-order sentences $\mathcal{T}(M)$, which is called the *theory of* M, and its 'inverse' from a set of sentences Γ to a class of structures.

DEFINITION 2.7. *For any $\mathcal{V}$-structure M, the* theory of *M, denoted with $\mathcal{T}(M)$, is the set of all $\mathcal{V}$-sentences ϕ such that $M \models \phi$.*

DEFINITION 2.8. *For any set of $\mathcal{V}$-sentences, a* model *of Γ is a $\mathcal{V}$-structure that models each sentence in Γ. The class of all models of Γ is denoted by $\mathcal{M}(\Gamma)$.*

We arrive at *theory* in the context of logic.

DEFINITION 2.9. *Let Γ be a set of $\mathcal{V}$-sentences. Then Γ is a* complete *$\mathcal{V}$-*theory *if, for any $\mathcal{V}$-sentence ϕ either ϕ or $\neg\phi$ is in Γ and it is not the case that both ϕ and $\neg\phi$ are in Γ.*

It can then be shown that for any $\mathcal{V}$-structure M, $\mathcal{T}(M)$ is a complete $\mathcal{V}$-theory (for proof, see *e.g.* (Hedman, 2004) p90). Regarding the latter part of *Definition 2.9* we get the following definitions.

DEFINITION 2.10. *A set of sentences* Γ *is said to be* consistent *if no contradiction can be derived from* Γ.

DEFINITION 2.11. *A* theory *is a consistent set of sentences.*

The interpretation as mentioned above in *Definition 2.5* is also denoted with $\mathcal{I}$ where $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ where $\cdot^{\mathcal{I}}$ is an *interpretation function* that assigns to each concept C in $\mathcal{T}$ a subset $C^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$, to each relation R of arity n a subset $R^{\mathcal{I}}$ of $(\Delta^{\mathcal{I}})^n$, and so forth. Then we get concept (& relation) satisfiability and a satisfiable theory as:

DEFINITION 2.12. *Given a theory* $\mathcal{T}$, a concept C (relation R) in $\mathcal{T}$ is satisfiable *if there exists a model* $\mathcal{I}$ *of* $\mathcal{T}$ *such that* $C^{\mathcal{I}} \neq \emptyset$ *(*$R^{\mathcal{I}} \neq \emptyset \times \emptyset$*), i.e.* $\mathcal{T} \not\models C \sqsubseteq \bot$ *(*$\mathcal{T} \not\models \exists R \sqsubseteq \bot$*).*

DEFINITION 2.13. A theory $\mathcal{T}$ is satisfiable *if there is an interpretation* $\mathcal{I}$ *which satisfies every axiom in* $\mathcal{T}$*; in this case* $\mathcal{I}$ *is called a model of* $\mathcal{T}$.

Note that $\mathcal{T}$ logically implies an axiom α, written as $\mathcal{T} \models \alpha$, if α is satisfied by every model of $\mathcal{T}$. We return to theory, models, and satisfiability of the TOG in §3.8.3 and, within the context of DL languages, in §5.5.1 and *Appendix C*.

Universals and particulars. In addition to the foundational notions of model theory, several terminological issues concerning universals and particulars have to be clarified. The theory with its functions is applicable to both universals and particulars, for we do not specify *a priori* the particular data source $\mathcal{DS}$ granularity is applied to. The formalisation uses several components of the DOLCE foundational ontology (Masolo *et al.*, 2003), such as PT for particular and PRO for process, to type the instances—as DOLCE is a foundational ontology for particulars (a section of DOLCE is depicted in *Figure 3.5*). One can read the formalisations to be applicable to entity types (universals), where they are of the type indicated with the DOLCE category. Thus, a particular amount of matter x, such as Mary's saliva, is denoted as $M(x)$—M in DOLCE is the abbreviation for "amount of matter", not as in Model, above, and $\forall x(M(x) \rightarrow PT(x))$—if the data source is a database or the ABox of a DL Knowledge base, whereas if the data source is an ontology of universals or TBox, then the x in $M(x)$ stands for its universal Saliva[6] and then $\forall \phi(M(\phi) \rightarrow U(\phi))$ (or: $\forall x(M_U(x) \rightarrow U(x))$). Furthermore, to clarify usage of the terms concepts, class, universals, instances, particular:

i. There are several definitions for *universal*; for instance, that it is a combination of properties. One can extend this imprecise description with the requirement that a universal must have instances, we get the definition that "[a] universal is an entity which is multiply located in space and time through its instances. It is what these instances share in common with each other" (Smith *et al.*, 2006). One can add further restrictions to the definition, concerning mind-independence and identity: "Universals are a class of mind independent entities, usually contrasted with individuals, postulated to ground and explain relations of qualitative identity and resemblance among individuals. Individuals are said to be similar in virtue of sharing universals." (MacLeod and Rubenstein, 2005). Philosophically, universals are not always considered to be distinct from concepts (Earl, 2005), but, in practice, the term *concept* tends to refer to mind-dependent entities (Smith, 2004). A *class* is a set, which may correspond to a universal or a concept in its intension.

[6]Alternatively, one could choose for a many-sorted logic, but it will be made clear when we deal with instances and when with universals, where the latter is denoted with U and the variable x in $U(x)$ is then a variable ranging over universals cf. particulars PT.

ii. Following a realist approach, then universals are instantiated by at least one individual (/instance/particular/token). However, the reverse is more widely accepted, *i.e.*, that for each individual there must be a universal for it. The latter allows one to say things about, *e.g.*, unicorns and fairies, and store information about extinct species; the former does not. For purpose of generality and widest applicability, the latter, less constrained, version is followed. Formally, we have the definition for instantiation as follows:

DEFINITION 2.14 (Instantiation). *Let a be an instance of universal A, their relatedness is denoted with the* instantiation *relation $inst(a, A)$; that is, for a variable x ranging over instances and ϕ a variable ranging over universals, we have $\forall x \exists \phi (inst(x, \phi))$.*

To ensure that instances cannot have instances, one can add the constraint $inst(x, \phi) \rightarrow \neg \exists y \; inst(y, x)$, which is already implicit in the typing of the $inst$ relation. (This does not rule out second-order logic for instantiating a class from its meta-class as is common in languages such as UML, because the class is not an instance.) Note that this is a similar approach to that of, among others, Bittner and Stell (2003), but their "$TI1$" is explicitly *not* included here. Put differently, it is possible for universals ϕ and ψ to have the same set extension during some time, but this does not imply that $\phi = \psi$ must hold. More precisely,

PROPOSITION 2.1. *Let x be an instance and $inst(x, \phi)$ and $inst(x, \psi)$ denote instantiation of ϕ and ψ by x, then either*

- *ϕ and ψ have a common subsumed universal φ such that $inst(x, \varphi)$, $\forall x(inst(x, \varphi) \rightarrow inst(x, \psi))$, and $\forall x(inst(x, \varphi) \rightarrow inst(x, \phi))$ hold; or*
- *one is subsumed by the other, that is , $\forall x(inst(x, \phi) \rightarrow inst(x, \psi))$ or $\forall x(inst(x, \psi) \rightarrow inst(x, \phi))$; or*
- *$\forall x(inst(x, \phi) = inst(x, \psi)$.*

A refinement will be added in §3.4.1 to narrow down this proposition for certain cases by introducing the most specific universal of an instance (*Definition 3.12*).

Basic granularity functions. For the present purpose, it suffices to let GL denote granular level, which is a unary predicate, U as universal and PT a particular, $\mathcal{DL}$ is the set of levels in the domain granularity framework (D^f) and D^s the subject domain (not to be confused with a specific data source, such as an instance of the FMA database); full formalization of the components of the TOG will be introduced, defined, and formalised in Chapter 3. Given the preliminaries, (2.8) says there is a relation between an entity (/type) x and the level it resides in and with $grain$ (2.9) we retrieve level y where x resides; an alternative notation for (2.9) is $\forall x(grain(x) = y \rightarrow D^s(x) \wedge GL(y))$. $assignGL(x, y)$ assigns an entity (/type) x to a granular level y, where $U(x)$ or $PT(x)$ and $GL(y)$ (2.10, 2.11).

$$\forall x, y(in\_level(x, y) \triangleq ((PT(x) \veebar U(x)) \wedge GL(y) \wedge grain(x) = y)) \tag{2.8}$$

$$\text{grain} : D^s \mapsto GL \tag{2.9}$$

$$\text{assignGL} : D^s \times GL \tag{2.10}$$

$$\forall x, y(assignGL(x, y) \rightarrow GL(y) \wedge (PT(x) \veebar U(x)) \wedge in\_level(x, y)) \tag{2.11}$$

Further, one can enforce that each entity must reside in a granular level with (2.12).

$$\forall x(D^s(x) \rightarrow \exists y(grain(x) = y \wedge GL(y))) \tag{2.12}$$

Put differently, it represents an ontological commitment that the world—or at least the subject domain under consideration—is granular, which may or may not be truthful to reality. Moreover, this constraint may be too restrictive for deployed information systems where for some part of the D^s either the granulation is not known or beyond the interest of the domain experts and software developers. This topic will return in §2.3.2, below, and in §5.5.1.

2.3.2 Content structure for the eight leaf types

The 8 leaf types inherit characteristics from their parent type of granularity. There are several general conditions that the structure of the entities (/types)[7] within a level must satisfy, which all types inherit from **cG**. Based on the previous sections, one can draw a preliminary list of general characteristics for the entities (/types) that are granulated according to any of the subtypes of **cG**.

i. The contents of a level can be either entity types or instances, but not both.
ii. The entities (/types) in a particular level have at least one property (value) in common.
iii. The entities (/types) are disjoint, but not necessarily exhaustive due to our gaps in knowledge of nature. Within a closed world assumption, they are disjoint exhaustive.
iv. Provided an entity (/type) is not an orphan and the subject domain is covered fully with granular perspectives, it must reside in at least one granular level.
v. An entity (/type) never can reside in more than one granular level within the same perspective and that entity (/type) is classified as the same (instance of) universal.
vi. The entity (/type) in a granular level may reside also in ≥ 1 other levels, provided that each level the entity (/type) resides in is contained in a distinct granular perspective.

This will be specified more precisely and, where appropriate, incorporated in the TOG in Chapter 3 or discussed in §5.5.1 why it is not included in the theory. In addition, recollecting the introduction of §2.3, only lists of ingredients are given to formally characterise the eight leaf types, whereas an elegant, comprehensive formalization that relies on both the TOG and a generic foundational ontology, such as DOLCE, is a topic of future work. For instance, the underspecified "$\prec$" over levels that was used in §2.1 can be defined (called RL in the TOG), several of its properties proven, and a clear distinction can be made between the relation between levels and the granulation relation between entities (/types) residing in the levels, as will be shown in §3.6. Further, notions such as endurant, ED, physical region, PR, how they relate, and so forth have precisely defined meaning with constraints in DOLCE, whereas the intuitive primitives that will be suggested for several types may be defined in terms of DOLCE categories and constraints after closer investigation (the bare minimum required for the TOG is included in §3.8.1).

The respective characteristics for the eight leaf types of granularity are then as follows.

saoG. All instances in the ordered set belonging to a particular level are instances of the same type. The amount of whole instances is not necessarily determined by the size of the entity that is granulated (see §2.4.1 for an example with granulating a lake). The instances are automatically disjoint because the granulation results in a grid. In addition, the instances within the same level make up the set-extension of its corresponding universal, such as a set of plots of km^2 where the amount of plots depends on both the entity that is granulated and on the decisions to include or discard 'partial' plots where a cell of the grid covers a larger area than the part of the entity. Combining these constraints, we need to represent, at least:

- a notion of region to represent the cells of the grid (*e.g.*, DOLCE's region R) and that these cells are of the same type, so that if we denote the granulation region with, say, $granR$, then $\forall x(granR(x) \rightarrow R(x))$ and for a particular grid where the shapes are squares (with squares defined the usual way), then we have also that for that particular granular perspective $\forall x(granR(x) \rightarrow Square(x))$ holds;
- a way to relate the boundary of one cell to another and to state these cells are disjoint;
- the entity (/type) to which the grid is applied (*e.g.*, DOLCE's endurant ED);
- that the endurant can be associated to each level in the granular perspective, but granulation regions of certain size are in one level of granularity only and those of a larger measure must be in a coarser-grained level than those of a smaller measure.

[7]The use of the word "structure" in the context of of contents in granular levels refers to its organisation of the entities (/types), such as an ordered list or grid, which can coincide with structure sensu *Definition 2.5* if the TOG is applied to instance data but its underlying idea will also be used where the contents is type-level knowledge.

samG. All instances in the ordered set belonging to a particular level are instances of the same type and they are whole instances at that level. Further, there is an exact, known, number of instances that can be in that level. In addition, the entities and instances at the higher levels are ultimately composed of the chosen Urelement at the lowest granular level. In contradistinction with **saoG**, the set is grouped into particular amounts, like {Hour 1, ..., Hour 24} at the Hour-level y_i, which are ultimately built up from the same Urelement, such as Second at level y_{i+2}. Combining these constraints, we need to represent, at least:

- Urelement defined as the arbitrarily chosen atom at the most fine-grained level in the granular perspective, which may or may not be Atom *sensu* (2.1);
- a function to calculate the amount of elements that have to be aggregated and to relate that function to the level;
- the granulation relation between the aggregates in different levels versus the relation between granular levels;
- that the aggregates in different levels are the set extensions of different universals (as opposed to arbitrary aggregates).

sgpG. This involves a 'zooming in' and 'zooming out' factor, where at a coarser-grained level, *e.g.*, the wall and wallpaper touch each other, but at a greater magnification, there is wall-glue-wallpaper, and again in smaller detail, one looks at the molecules in the paper, glue, and wall. The zooming factor is like a grain size when relating levels of granularity, where *within one* level one can distinguish instances of, *e.g.*, $\geq$ 1mm but instances $<$ 1mm, metaphorically, fall through the sieve and are indistinguishable from each other, but are distinguishable at lower levels of granularity. In practice, this is used, *e.g.*, when filtering substances with filters having different pore sizes and dialysis tubes. With **sgpG**, differences in physical size of the entities (/types) is *the* property for granulation. To characterise the content, we need a function, $size\_of$, which returns a value in, say, length, square or cubic size, which can be categorised as physical regions alike DOLCE's PR. Thus, the instances recorded at some level gl_j are physically smaller than the instances at a higher level (gl_i), with $gl_j \prec gl_i$, and thereby are related to each other at least or possibly only by the relation that they fall within the same physical size range. Combining these constraints, we need to represent, at least:

- access to the measurement of the physical size of the objects, *e.g.*, with a function $size\_of : PT \mapsto PR$ for the measured region (PR) of an object (PT), and comparison of measured regions so that, in rudimentary form, $size\_of(x) < size\_of(y)$ provided we have $in\_level(x, gl_j)$, $in\_level(y, gl_i)$, and $gl_j \prec gl_i$;
- that this measurement is taken by using a direct or indirect measurement property of the entities (/types);
- a value range for each level in the granular perspective and to ensure that those ranges of coarser levels are larger than those of finer-grained levels;
- that, consequently, the (type of) entities in different levels are different—following the first item, then $\forall x, y(inst(x, \phi) \rightarrow \neg inst(y, \psi))$—and can also be different within the same level.

sgrG. The entities in reality associated with the levels are the same in the coarse- and finer-grained levels, but their representations change according to pre-defined resolutions. That is, the real world entity is the same universal or its instance, but one chooses to represent them as if they were instantiating different universals; *e.g.*, Cell wall as circle, lipid bi-layer, three-dimensional structure, or also considering the movements of the lipids and proteins. In this case, the resolution-motivated representation is a figurine where that at a coarser-grained level is, in fact, a proper part of the figurine at the finer-grained level, as, *e.g.*, a point is a proper part of a polygon and circle a proper part of a sphere. This is common in cartography and GIS in general, where at a greater resolution, a street is represented as a single line, two parallel lines, or even more detail. Thus, there are several mappings from the same entity to different figurines, resulting in the situation where ordering the figurines with respect to the resolution (hence, also their attributes) are more important than the actual entity. To capture this multi-representation, we could introduce a primitive $rep\_of(x, y)$ to denote the relation between the real-world entity and its coarser- or finer-grained representation $Rep(y)$. Combining these constraints, we need to represent, at least:

- associate a value to the granular level to represent the resolution applicable to the level;
- the entity (/type) that has a multi-representation (*e.g.*, DOLCE's endurant ED) and thereby that the endurant can be associated to each level in the granular perspective;
- the figurines with, say, $Rep(x)$ (but preferably a more detailed characterisation);
- the association of the entity (/type) with the figurines with, say, $rep\_of(x, y)$ where $ED(x)$ and $Rep(y)$;
- that given any $rep\_of(x, y_i)$ and $rep\_of(x, y_j)$, then they must be in different levels;
- that given the previous item, if $in\_level(y_i, gl_i)$ and $in\_level(y_j, gl_j)$ and $gl_j \prec gl_i$, then $Rep(y_i)$ is a proper part of $Rep(y_j)$.

nrG. The entities in a level are of a different type, but all are of the same category, such as all being non-agentive physical objects ($NAPO$) or processes (PRO) and so forth. For instance, at the Cell-level, there are many *types* of cells, but they are all of the category $NAPO$ structural component (Hemal cell, Leukocyte, ...), or function (Hormone excretor, Insulin excretor, ...), and so forth, or a Protein unit structure-level with items such as α-helices and β-sheets. Thus, the entities are structured in a hierarchy where the direct children are in a lower level of granularity than its supertype. The characteristic of the **nrG** type is the type of relation between entities, which is of the same type throughout. Which types of relation to permit will be analysed in detail in §3.6; here it suffices to denote this with granulation relation *GR*. In the level, without further specification, the entities can be in an unordered set. It may be, however, that the content has some other additional structure within the level alike a **nasG**, or another **nrG** structure, as illustrated in *Example 2.5* and *Figure 2.5*. Alternatively, one can group the unordered set such that it takes into account the additional tree (or other) structure in the level, where each granule correspond to a different branch. Either way, the entities are disjoint thanks to the underlying structure in the data source. Combining these constraints, we need to represent, at least:

- the granulation relation GR between the entities (/types) that relate these entities (/types) residing in adjacent levels;
- that for each granular perspective only one granulation relation is used;
- the permitted granulation relations by which one can granulate the data.

nfG. The entities in a level can be of different kinds, such as folding $NAPO$s with their processes and states, combining types of entities into *one* entity residing in an adjacent higher level. It is not the case that the entities contained in the lower granular level is an (un)ordered set, but the entities (/types) are always related to at least one other entity (/type) within that level. For instance, the hierarchical modeling to improve comprehension of large conceptual data models that was illustrated in *Example 2.2* and different folding operations—that is, what is folded and how—can be identified, concerning perdurants and endurants and some of their subtypes (this will be elaborated on in §4.3.2). Combining these constraints, we need to represent, at least:

- the assertion that the entities (/types) in a level are related to each other;
- the granulation relations between the entities (/types) and their relations in the finer-grained level as the domain of the relations on the one hand and the single entity (/type) they are are folded into in the coarse-grained level as the range on the other hand;
- the permitted granulation relations between the coarse-grained entity (/type) and the finer-grained entities (/types) and relations it expands into.

nacG. Like **samG**, all instances in the set belonging to a particular level are all of the same type and at that level they are whole instances. It is not necessarily the case that the amount of instances in a particular level is known and can be computed. For instance, Sports team does have a predefined amount of instances of Player per team, but sales department members of a company do not have to have always the same amount of members. The instances that are member of such populations change over time but the entity (/type), generally labelled with a collective noun, and its meaning endures. Thus, looking at the structure of the data in a level, it is at least an unordered set but can be an ordered set of instances, and the instances populating the set can vary over time, although the entity (/type) keeps its identity. It might be possible, to have not an (un)ordered set but a taxonomy or other additional aggregation within the level alike a **nasG** or **nrG** structure, such as an employee hierarchy (with Junior sales person, Senior sales person, Trainee, Manager, etc), or aggregated by the organisational unit (teamA1, teamA2, etc). Combining these constraints, we need to represent, at least:

- that the instances in the level instantiate the same type, *i.e.*, $\forall x(inst(x, \phi))$ for a particular level, or, at the type-level, that they are subsumed by a root entity type;
- that this type is, at least, a subtype of endurant ED, such as a social object and not defined by its extension;
- a notion of 'membership' of the entities (/types) in the finer-grained level as members of the entity (/type) in the adjacent coarser-grained level, such as through the meronymic $member\_of$ as granulation relation.

nasG. The structure of the data is like **samG**, but if one combines the subsets at each level, then the amount of unique instances residing in *each* level is always the same amount as they are instances of the chosen counting element. For instance, at level gl_1 there are 100 phone points and in a gl_2, such that $gl_2 \prec gl_1$, the 100 phone points may be divided into three subsets Land line, Mobile, Phone over IP each with, say, 2, 35, 63 elements of the original set, respectively, hence, Mobile $\subset$ Phone point. There may be a gl_3 with Classic cell phone and Skype mobile phone that granulates Mobile phone points and that each have 20 and 15 elements in the set, respectively, which adds up to the 35 elements for Mobile of the higher level gl_2 (assuming that Vodafone $\cap$ O2 = $\emptyset$, although in certain cases there may be a 'rest group'). Thus, at each level there are subsets with instances as elements of the set that, depending on the granulation criterion, are disjoint. Combining these constraints, we need to represent, at least:

- define the counting element as the arbitrarily chosen atom at the most coarse-grained level in the granular perspective, which serves to count the number of instances in all levels;
- that the number of instances at a given time are the same for each level in the granular perspective, hence, are fully partitioned at each level;
- for all instances that are member of a class ϕ in a finer-grained level, their coarser-grained representations are instances of ψ residing in a coarser-grained level, *i.e.*, $\forall x(inst(x, \phi) \rightarrow inst(x, \psi))$ so that taxonomic subsumption (is_a) may be a relation by which to granulate the data;
- following from the previous point and proper taxonomy development, then the instances at some level gl_2 have in their representation either at least one more attribute or more constrained attribute values than their respective representation in the coarser grained-level gl_1.

One may opt for the design decision to demand from the chosen criterion that the sets never overlap, or, for 'just in case', create two subtypes of **nasG** where one does allow overlapping sets and the other subtype does not (an illustration is given in *Example 2.4*). It does not merit a subtyping because the core ontological aspect is the same, but it may be useful for software systems to distinguish between these two cases.

Some types of relation between the entities or instances within a level can be combined, because one does not have to take into account that some are granulated according to arbitrary scale and others are not. (The (non-)arbitrary scale division is relevant for the relations between levels, but do not always act out on the relation between entities/instances contained within a level.) **nasG**, **nacG**, **nrG**, and **sgG** may be unordered sets, **samG** and **saoG** may be ordered sets, and **nfG**, **nrG**, **sgG**, and **nacG** can have a more complex additional orthogonal structure of the data inside the level that itself may be subject to a granular structure. This, among other topics, will be illustrated in the next section.

2.4 Sample contents of a granular level

Although that what is 'inside' the granular level is not a component of the framework, an illustration will benefit understanding of the overall system, regarding both the previous paragraphs and next chapters. The steps how the contents of a level got in there are assumed at this stage—Chapter 4 contains a formal analysis of *how* contents relate to the TOG—-and here will be outlined through examples the several salient characteristics as well as problems that need to be solved.

2.4.1 Content of a level with arbitrary scales

For granularity type **saoG**, each level is granulated alike a grid with cells that may or may not be exhaustive for its contents. Concerning the exhaustiveness, one can think of, *e.g.*, laying a grid on a 'lake' at a higher level as depicted in *Figure 2.4-A* to ensure each part of the lake is covered by the grid. Alternatively, one might want to apply a fuzzy rule alike "when $> 50\%$ of a cell is occupied it must be covered by a grid cell", shown in *Figure 2.4-A′* that consequently discards parts of the lake that occupy $< 50\%$ of a cell. This leads to a second question and a consequence: if the discarded cells in *Figure 2.4-A′* should be taken into account at a lower level. If one does, then one arrives at a granulation as depicted in *Figure 2.4-B*, if one does not (*Figure 2.4-B′*), then the parts at the lower level do not make the whole at the coarser-grained level as can be observed from the difference between moving from A′ to B′ instead of from A to B to B′ (the seven shaded squares would have been discarded moving from A′ to B′). The discarding rule means that granulation is not exhaustive for we have thrown out a remainder. This type of impreciseness is characteristic for any coarse-grained level and applies to scales for features such as surfaces, volumes, isotherms, and isobars. Developments in rough set theory and fuzzy logic might serve as an appealing implementation method, and is further elaborated on in §3.4.1 regarding indiscernibility and §5.2.2 on related works. Yet differently, instead of including part of the shore of the lake, one can divide the lake by making the grid *inside* the coloured area only, but then one would have to deal with incomplete cells, that is, cells of different size within one granular level, which complicates computation and would not solve the aforementioned boundary problems. Another type of (non) exhaustiveness occurs with less obvious scales, which was briefly illustrated in *Example 2.1*. For instance, if it had only (Baby, Child, Adolescent, Adult), then Elderly is omitted, hence, that the granulation is either non-exhaustive or assumed to be included in Adult and thereby meeting the exhaustiveness criterion. Because we have created the scale, we can easily decide one way or the other.

2.4.2 Non-scale-dependent content of a level

One might conceptualise non-scale-dependent, qualitative granularity as squeezing in a grid-structure in a level, but this ignores the relations between parent/children in the hierarchy, be such that each cell contains one entity (/type) only, and would then be a changeable grid and no (pairwise) disjoint tree, because disjointness and exhaustiveness in biology is aimed for but

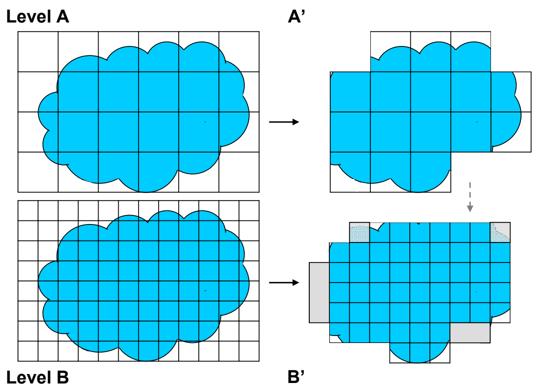


Figure 2.4: Grid with cells partitioning a 'lake' according to different rules.

rarely achieved. This is primarily due to epistemological reasons: there are many things of nature we just do not know enough about, accumulation of knowledge about nature is in flux, and discoveries are not made following a balanced binary tree representation but as they come and where most funding is[8]. In addition, disjointness depends on the categorisation one is accustomed to, where it may be that the types and their instance satisfy more than one category, which is illustrated in *Example 2.4*.

Example 2.4. The 'everything else' group in the Tree of Life phylogeny (Maddison and Schulz, 2004) is The other protists (Patterson, 2000) that contains types of eukaryotes for which no satisfactory place in the taxonomy has been found or decided upon. Omitting those ambiguous species when loading a granularity framework with the taxonomy avoids this problem, but it is exactly those non-textbook cases that are of most interest to scientists and where granularity and its additional reasoning services could be a useful novel analysis methodology.
Instead of phylogeny, the vernacular in scientific communication refers to more or less officially established groups of species, which brings afore the need to address non-disjointness, which can occur in phylogeny and species taxonomies only if one wants to represent alternative trees in one system because different species taxonomies can be constructed by choosing different sorting criteria. Scientific publications use groups of microorganisms; *e.g.*, that a type of bacteriocin can inhibit Gram-positive bacteria. A group contains types that have one or more properties in common. Properties can be, among others, genus (*Listeria* sp.), isolation source (soil bacteria), growth condition (thermophiles), biochemistry (lactic acid bacteria - LAB), or morphology (cocci). Bacteria are often a member of more than one group (a LAB coccus) and some groups can be subtyped further (LAB streptococci, LAB staphylococci etc.); *i.e.*, there are also hierarchies of groups of types of organisms.
There are difficulties with the species concept (*e.g.*, Hey, 2001), which is complicated by horizontal gene transfer (van Passel, 2006; Gogarten and Townsend, 2005; Gevers *et al.*, 2005), favouring fuzzy, overlapping, boundaries over crisp sets even for non-scale-dependent types of granularity. On the other hand, one could use granularity to define clear boundaries and examining their validity. ♢

Proceeding to the issue of relating data to the framework, granularity can be positioned orthogonally to the data source so that it provides an additional layer to infer more knowledge than is possible separately. Loading a domain granularity framework with data is—or should be—structure-preserving with respect to the data source, hence, so that granularity *enhances* the domain data. Advantages of this approach are illustrated in the next example.

[8]There is a disproportionate body of information about a few so-called "model organisms" (*Drosophila* fruitfly, etc.), humans, diseases, staple crop plants, and domesticated animals.

Example 2.5. The Foundational Model of Anatomy (FMA, 2003) uses both is_a and $part\_of$ relations between anatomical entities. Let us take parthood for granulation, then the taxonomic structure can be preserved in the levels, as depicted in *Figure 2.5-B* for cells. The FMA lists that Blood has as parts: Plasma, Erythrocyte, Neutrophil, Eosinophil, Basophil, Lymphocyte, Monocyte, Platelet, B lymphocyte, T lymphocyte, Natural killer cell, Granular leukocyte, and Leukocyte. Relying on this unordered set alone, one cannot know if it is exhaustive: 1) the list was created manually and some entity type may have been omitted by accident, 2) an ontology adheres to the open-world assumption, and 3) the development tool, FMA-Protégé, does not include axioms for disjoint exhaustive. Combining the taxonomy subsumed by Cell and intersecting it with the parts Blood, it is immediately evident that both the parent and child types of Non-granular leukocye are part of blood, but not Non-granular leukocye itself, even though logically it should be. In addition, two cell types that are directly subsumed by Non-granular leukocye are Peripheral blood mononuclear cell and Lymphoblast, but they are not listed as parts of blood. Monocytes are definitely part of blood, whereas lymphoblasts are "immature lymphocytes" and either not non-granular leukocytes or should be subsumed by Lymphocyte. Either way, the structure-preserving loading of granular levels brings afore the ignorance about such transforming entities, which has yet to be resolved ontologically. Obviously, one would want to take advantage of the already encoded structure of the taxonomy and have returned something alike *Figure 2.5-B* instead of *A*. This example is revisited in Keet (2006b), where I demonstrate the advantages of automating analyses compared to this brief manual example. ◇

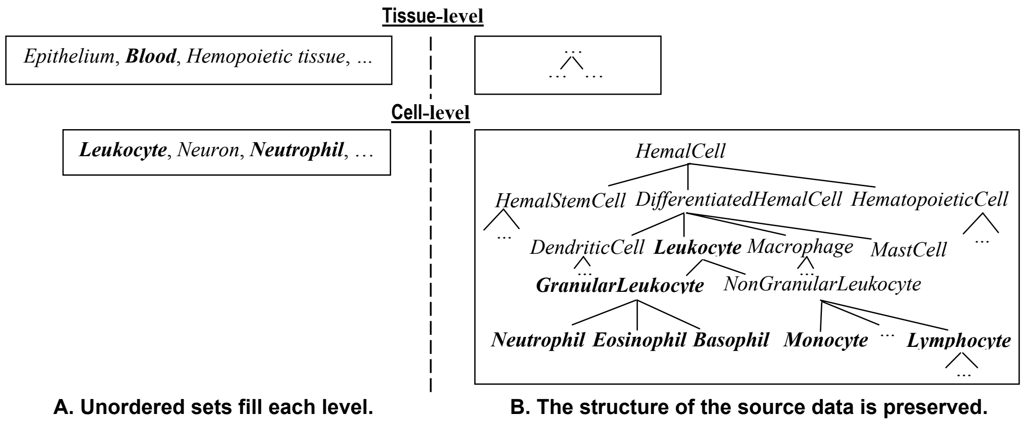


Figure 2.5: Two levels with examples of their contents, unordered as in the FMA versus structure-preserving; entities subsumed by Hemal cell present a section of the FMA (2003) where terms in bold-face are listed (as in A) as part of blood.

Thus, an important advantage of this structure-preserving approach is that when presenting the combination of taxonomy with granularity demarcations, one gets for free the detection of inconsistent or incomplete knowledge in either the taxonomy or in the partonomy. Thus, conflicting information is highlighted, can be used for formulating research questions, and be investigated.

Example 2.5, however, brushes over another issue. Lower levels may contain many more entities—1 human body, 12 organ systems, 300+ cell types, 100000+ proteins (Hunter and Borg, 2003)—which is difficult to interpret if it were represented as an unordered set for each level. This can be pruned through intersections as done in the example, or by selecting an entity type at a higher level of granularity first. For instance, if one searches the contents at the Cell-level combined with a particular selection of, say, Blood at the Tissue-level[9], then the types returned contain only the entity types in the selected levels & type, indicated in bold face in *Figure 2.5-B*; functions to achieve this, among other types of queries, will be introduced in Chapter 4.

A related facet of utilising the structure of the contents compared to an unordered set for each level concerns the 'size' aspect, where it is crucial that the relations between the entities at different levels are not destroyed when applying non-scale-dependent granularity the subject domain. The typical problem it otherwise raises is illustrated in *Example 2.6*, whereas benefits from using the underlying structure at higher levels for reasoning is demonstrated with *Example 2.7* afterward. It will be revisited in Chapter 4 where functions will be integrated with the TOG.

Example 2.6. Take avian anatomy limited to chicken, where the Cell-level is finer-grained than Organ, which is finer-grained than the Body part-level. This neither implies that any organ can be part of any body part nor that all cells are automatically physically smaller in size: the chicken egg in the Cell-level is larger than its head at the Body part-level. This can be ensured when one maintains the partonomy of the chicken's anatomy so that it contains part_of(Chicken egg, Body). Then, using the $part\_of$ relation between entity types in addition to the level hierarchy, it properly prevents an incorrect answer like part_of(Chicken egg, Head). These are different branches that should not be mixed, which can occur with (un)ordered sets, but not if one maintains the structure from the original data source. ♢

Example 2.7. Whereas *Example 2.5* looked at the lower levels, here we also consider coarse-grained levels of Blood. (1) Represents three levels in the mode of transmission perspective for infectious diseases (*Appendix A*), and (2) is taken from the FMA partonomy; thus, Blood is positioned at the intersection of two levels in distinct perspectives, being Mode of transmission and Anatomy, and one can derive (3) from (1) and (2) by traversing the levels 'up'. (4) Is another branch in the FMA: one branch descends to Blood and another one to Skin-associated lymphoid tissue, and both are ultimately part of the Hemolymphoid system. However, one cannot conclude that Skin-associated lymphoid tissue (SALT) is involved in transmission via Direct contact, but it does pose hypotheses on involvement. In fact, SALT *prevents* infectious agents to enter the vascular system (hence, blood). Although the involvement is different, new combinations may be identified and suggest directions for new research.

```
1: Blood involved_in Person-to-person involved_in Direct contact
2: Blood part_of Hematopoietic system part_of Hemolymphoid system
3: Hemolymphoid system involved_in Direct contact
4: Hemolymphoid system has_part Lymphoid System has_part
   Non-lymphatic lymphoid system has_part Skin-associated lymphoid tissue
```

Traversing the partonomy downwards, one can infer that at least one of Blood's 13 parts must be involved in transmission of infectious agents because blood is. This is already supported by scientific evidence: transmission of hepatitis C virus via Erythrocytes (Widell *et al.*, 1996) and West Nile Virus via blood Plasma (Hollinger and Kleinman, 2003). Consequently, one may wonder if it is the whole cell or if one can isolate parts of cells that are involved. The latter has been established manually with, *e.g.*, *Listeria* infections at the Organelle-level (sub-cellular) and nucleation of actin filament polymerization (Rodal *et al.*, 2005; Nature Editorial, 2005). ♢

Note that the assumption in the above example implies a reductionist viewpoint, and philosophically encounters the problem of infinite regress. It is possible that when the implementation predicts involvement of a lower level it either is not known, hence an epistemological issue

[9]Blood is a tissue in the Physiome anatomy ontology. In the FMA, blood is categorised as a body substance—regardless its ambiguous ontological status, the idea may be clear.

where the system generates new research questions, or for good scientific reasons involvement of a lower level is not possible due to a systems-level complex combination of events and substances. Either way, using biological granularity in combination with ontologies can speed up the discovery process because it combines existing information—and gaps therein—in a novel way, thereby offering a new view on the same information.

In this section, I briefly addressed factors involved in granulation, which provide a general idea of the topics and some of its challenges that will receive in-depth attention in Chapters 3 and 4.

2.5 Chapter summary

Foundational characteristics of granularity were investigated, which enabled identifying ontological distinctions between different types of granularity based on differences in emphasis (entity type view, instance view, and property-based), characteristics (scale- and non-scale-dependent types of granularity), and its representation (mathematical or otherwise). Based on the differences uncovered, a top-level taxonomy of types of granularity was developed and it was formalised how content of levels relate for each of the leaf types of granularity. Last, sample contents of a level of granularity were illustrated with examples from several subject domains, which introduced several problems, such as data source structure preservation. These aspects, among others, will be analysed and solved in Chapters 3 and 4.

Chapter 3

A domain- and implementation-independent theory of granularity

3.1 Introduction and key requirements

Overview of this chapter

In this chapter, the static components of the Theory Of Granularity (TOG) will be illustrated, analysed, formalised, and their constraints proven. We move from a data-centric treatment of granularity to the conceptual and logical layers, where oftentimes informally defined components of granularity become ontologically-motivated, formalised, modelling constructs in their own right. The resultant TOG is a consistent and satisfiable logical theory to ensure unambiguous semantics.

The investigation into granularity in Chapter 2 focussed for a considerable part on characteristics of *contents* of granular levels, but also revealed requirements that a theory of granularity must meet to effectively manage granularity in knowledge representation & reasoning and biological information systems. These requirements are listed in §3.1 and will be revisited in the chapter's summary (§3.10) where I will show that the here proposed TOG meets all of them. Sections §3.2-3.7 contain specific research questions, modelling considerations, definitions and propositions, and lemmas and theorems with proofs for the TOG components—domain, granular perspective, granular level, and the relations between them. The main constrains are summarised at the end of each of section and a *summary* of the outcome of the analysis is depicted in *Figure 3.1* with ORM2 notation, which is intended for indicative purpose only. Each ORM Object type in the figure, denoted with roundtangles, will be given a definition in the upcoming sections, and likewise for the relations (rectangles). Constraints, such as mandatory participation (a blob), internal uniqueness (line above the rectangle), external uniqueness ($\ominus$ with dashed lines), and ORM-ring constraints (oval or circle with dots, *e.g.*, acyclicity on "RL") will be proven where possible. To give an intuitive idea, we have, *e.g.*, that each granular level GL is contained in a granular perspective GP that in turn is contained in a domain granularity framework D^f and each granular perspective can be identified by a combination its granulation criterion C and type of granularity TG. For a subject domain (d^s) of calendars with granulation bounded by d^f, then a particular gp_i could be the Gregorian calendar with levels built-up from sets of days using **samG** type of granularity and a particular level gp_igl_j could be Month; more comprehensive examples will pass the revue in the remainder of this chapter.

While *Figure 3.1* gives a useful bird's eye view and §3.2-3.7 go into the details with modeling considerations and justifications, all TOG axioms are put together in §3.8 and its satisfiability is demonstrated with the Mace4 model searcher, whose input file can be read in *Appendix B*, and is discussed in §3.8.3. Granularity in a theory of granularity will be addressed briefly in §3.9.

To demonstrate problems and usages as well as accommodate for non-logician readers, several notions are illustrated with examples taken from biology and biomedicine.

The part-whole relations in §3.5.1 are more comprehensively treated in the articles published at the *Second International Workshop on Object-Role Modelling* (Keet, 2006d), the *Applied Ontology* journal (Keet and Artale, 2007), and the tutorial reader (Keet, 2006e). The analysis of indistinguishability, similarity, and proofs for *Theorem 3.2* in §3.4.1 were published at the *IEEE International Conference on Granular Computing* (Keet, 2007d).

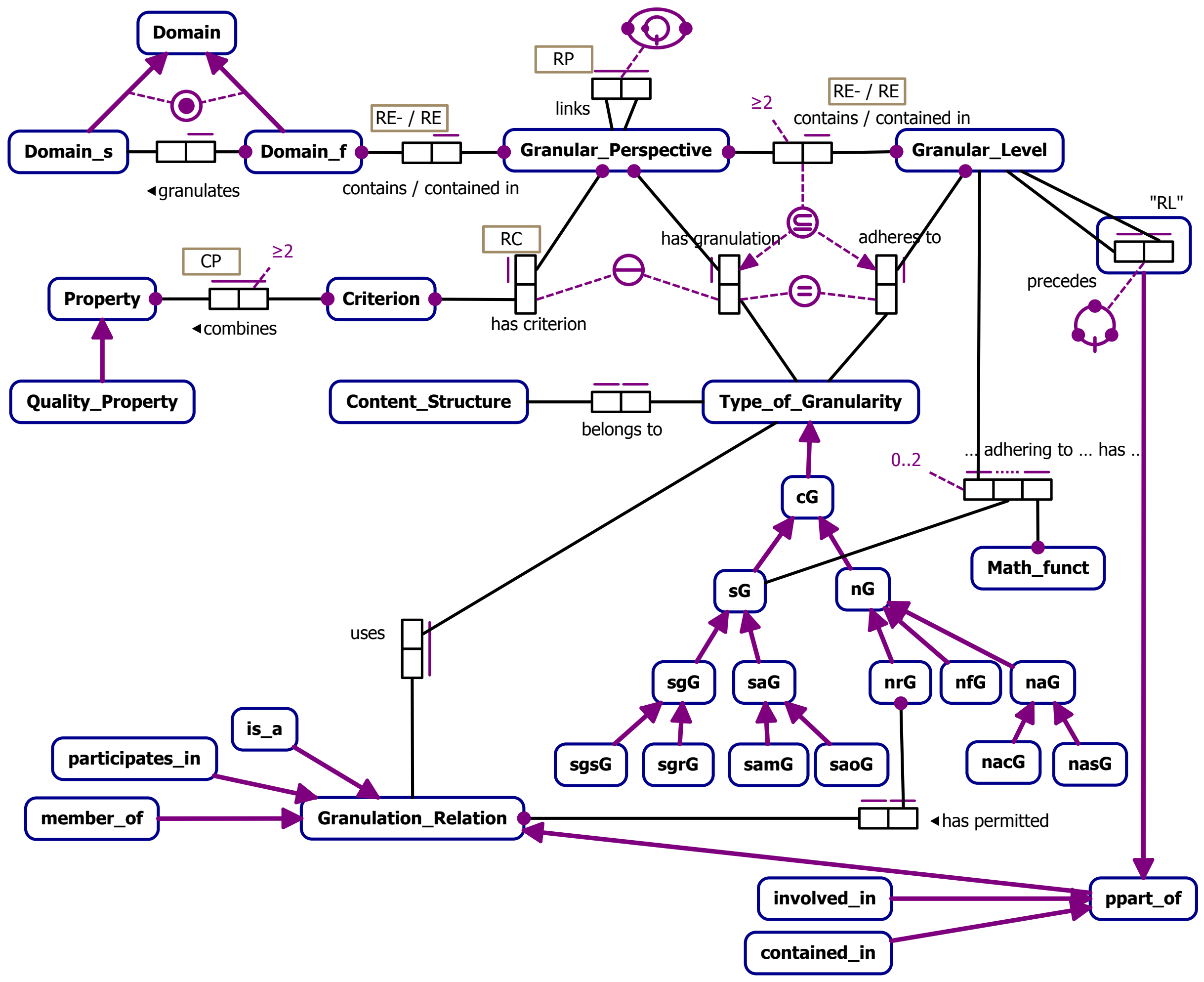


Figure 3.1: High-level graphical rendering of the main components, the relations between them, and constraints on their participation in the TOG (subtypes are disjoint). The Type_of_Granularity section was introduced already in Chapter 2. The remainder of the figure—a refinement of the third column in *Figure 1.1*—is investigated in this chapter.

Key requirements for a theory of granularity

Based on the investigation into the foundational aspects of granularity in the previous chapter, key requirements that any theory of granularity should meet can be formulated now. These requirements concern both the granularity framework and those implied by the constraints on the structure of the contents of a level of granularity and its perspective.

1. A theory of granularity should be usable eventually in a format for contents at the instance level and in a format for defining a domain granularity framework at the type level;
2. A higher level simplifies, makes indistinguishable, the finer-grained details that are indistinguishable at that higher level;

3. Following from the indistinguishability requirement, there have to be at least two levels within a perspective, else there is no granularity;

4. Any theory must be able to accommodate both the quantitative and qualitative aspects of granularity (or: arbitrary scale and non-scale-dependent granularity);

5. The logic-based representation has to permit the two main ways for perceiving and representing granularity, being set theoretical and mereological, therefore the relations between the entities (/types) contained in the levels and the relations between granular levels are, at least, either of the type is_a or $part\_of$;

6. Given that one granulates according to a certain type of granularity, this also means that there has to be one type of relation between granular levels within a particular granular perspective;

7. For ontological correctness and computation, the type of relation between adjacent levels in a perspective has to be transitive for those perspectives that contain $>$2 levels;

8. A type of relation that relates contents in levels of granularity within a particular perspective can have the property of being intransitive, provided there are always exactly 2 levels in any given perspective that contains that type of relation relating the entities (/types);

9. The entities or entity types in a particular granular level have at least one aspect in common, which is a criterion by which to granulate the data, information, or knowledge;

10. An entity (/type) never can reside in more than one granular level within the same granular perspective (this follows from point 3);

11. Given that each level is contained in a granular perspective and granular perspectives contained in a domain granularity framework are disjoint, an entity (/type) in a particular granular level may reside in ≥ 1 levels, provided that each level the entity (/type) is contained in a distinct granular perspective;

12. If there is more than one granular perspective for a subject domain, these perspectives must have some relation among each other.

We return to these key requirements in §3.10.1, where it will be shown that the proposed TOG satisfies these criteria. With an eye on implementations, four desirable features can be identified, which will be assessed in Chapters 4 and 5:

A. The entities (/types) are disjoint;
B. The entities (/types) are not necessarily exhaustive: this may not be possible due to our gaps in knowledge of the natural world. Within a closed world assumption, they are (disjoint) exhaustive;
C. Provided an entity (/type) is not an orphan in the original data source and the subject domain is covered fully with granular perspectives, it must reside in at least one granular level;
D. The purposes of its usage—dynamic aspects with primary distinctions allocating/classifying entities (/types) and information retrieval & reasoning—shall not affect the static structure of a theory of granularity.

Given the key requirements, we now proceed to the analysis and stepwise formalisation of the TOG components.

3.2 The domain D

Question/task. Define the domain D as holder for the outer frame for a subject domain.

D requires identity criteria, or at least an identification, so as to distinguish it from other domains of interest; these may be supplied or carried by the domain. The definition of D has as aim to restrict any definition of a subject domain d_i, where there are two components to defining D:

- ⋆ The outer frame of the granularity, as framework.
- ⋆ A concept, description, or complex entity (type), as subject domain that is to be granulated.

Depending on the analysis and intended use, emphasis is put on one or the other, and therefore both will be analysed in the next two sections.

3.2.1 D as outer framework

D as outer granulation framework, D^f, can be considered as solely a cognitive frame based on human perception that would cease to exist if there are no humans (or other organisms capable of cognition) granulating a subject domain. However, considering the subject domain biology, and reality in general, it may be possible that granularity is not only a cognitive tool deployed by humans to order our understanding of the world around us. For instance, an animal eats only part of its prey and leaves bones and skin, thereby distinguishing between the prey and its parts, or a low enzyme specificity for substrates may considered to be a 'generalising feature' that enzymes use, thereby not caring about the finer-grained structural differences of the types of molecules. One may dismiss these cases as anthropomorphizing non-human entities, but claiming human supremacy on cognition to conceptualise and/or perceive granularity is not without its problems either. Yao identifies these distinct viewpoints as belonging to two extremes: one considers either that "the structural levels of matter [are] determined by the entirely objective laws of nature" or one focuses "on the human subjective multi-level understanding of reality" (Yao, 2005b). This has one main consequence for identifying what D^f is, because one takes either a

1. realist position and revisionary attitude where one considers only the intrinsic nature of the world that exist independent from humans and, if the conceptualizing agent's understanding of the world changes, then so the ontology has to be revised to reflect such change; or a
2. descriptive approach where "categories refer to cognitive artifacts more or less depending on human perception, cultural imprints and social conventions" (Masolo *et al.*, 2003, p7). This, of course, still permits descriptions of—representations of the intrinsic nature of—real world entities.

DOLCE, the Descriptive Ontology for Linguistic and Cognitive Engineering, takes the second approach, which intends to facilitate meaning negotiation of *already formed* conceptualizations (Masolo *et al.*, 2003, p13) and relies on agreement within a community. For instance, to describe a CoNcept CN[1], one has to know its *stable* definition and if something of that changes, then we do not have a revised concept, but a new one. Applying this to D^f—*i.e.*, if we assume $D^f(x) \rightarrow CN(x)$ holds—then we have to consider two cases, being identifying D^f and distinguishing its instances. First, one could determine the identity of D^f by what it contains. Recollecting *Figure 1.1*, this would mean that D^f is either the mereological sum of its granular perspectives or the superset. However, perspectives indicate how to granulate the entities of the subject domain, but neither define the domain framework nor are perspectives subtypes of domain. If, on the other hand, we change the *intension* of D^f, then there is a change in the theory, hence would qualify to be identified as a different concept. With the realist position and revisionary attitude,

[1]see Masolo *et al.* (2004) for an explanation, definition of, and constraints on concept CN and definition DF, which are part of DOLCE.

we would have an 'updated' notion of D^f instead. Given that the formal characterisation forms part of a logical theory, it is *easier* to adopt the descriptive approach, but this is not a good philosophical argument to categorise D^f as a concept. Second, considering D^f's instances, when one changes a particular d_i^f by adding or removing one or more perspectives, this would change a d_i^f into $d_{i'}^f$ if it were a concept, but the amount of perspectives in d_i^f, and, in turn, levels in perspectives, is not necessarily exhaustively declared upfront and also may change in time due to changes in the data source, as was discussed in §2.1.1 under "A6 and A7". This, would require a change in identity even though the concept did not change. Therefore, limiting the definition to the mereological sum is too restrictive and restricting it to superset is ontologically inadequate. (Observe that this does not preclude using mereology for the relations in the TOG, just that it does not suffice for identity.)

A different argument to characterise D^f as outer framework to serve as cognitive demarcation and to let it be a concept CN is as follows. Although the focal subject domain is biological granularity, this does not imply we must use the realist position and demand that a granular framework somehow can be extracted from nature. The main reasons for this ontological choice are twofold. First, the theory of granularity has to be applicable to other subject domains, too, which are not necessarily governed by nature, whereas a realist position in conjunction with laws of nature prevents such a descriptive use of a theory of granularity, hence, would be too restrictive in its applicability. Second, even when granularity is identified in nature, then we, humans, attempt to represent this understanding of granular structures in nature, and use some perspective, frame, or window on reality to demarcate that piece of reality; the frame is the boundary for what is granular or will be granulated. Thus, adopting a descriptive position still permits a realist approach to represent granularity in nature where deemed necessary.

3.2.2 D as subject domain

D as subject domain D^s, on the other hand, does *not* imply $D^s(x) \rightarrow CN(x)$. First, adding another granular level or perspective to granulate a complex concept or biological system such as Second messenger system (SMS) does not change the intended meaning of the SMS: the definition exists, with or without all relevant granular perspectives and levels humans think there are—*i.e.*, D^s can exist *independent* of D^f. In addition, there are two cases. First, let us take d_1^s = SMS to which GTPase is added of which it is already known to play a part in the SMS. This makes the d_1^s less incomplete, thereby remedying a previous hiatus in the used data source, hence, it is not epistemology but an incomplete domain ontology or knowledge base that was used for granulation, which is an implementation issue that does not change the intension of SMS. It is different when an "Undoubtedly Most Important Component" (Lazebnik, 2002) is discovered that significantly changes the understanding of the subject domain. That is, using biology as subject domain—but not necessarily for other subject domains—then a *revisionary* attitude is appropriate along the line of falsification and better approximations of the truth (Popper, 1996; Johansson, 2005), where entities can be reinterpreted or its linguistic expressions changed if it is not defensible on scientific grounds. Thus, it does neither suffice to categorise D^s only as a concept nor to require it to cover subject domains in nature only. D^s can be a CN for subject domains other than biology or categorised to be one of the other DOLCE categories (that DOLCE is intended for descriptive ontologies does not have to prevent one to use its machinery for representing scientific knowledge). For instance, SMS is a non-physical endurant ($NPED$), a d_2 = Human body is a physical endurant (PED), and d_3 = Krebs cycle can be a $NPED$, a process (PRO) or an accomplishment (ACC) that involves multiple enzymes, substrates, other molecules, and processes. To be more precise, one can generalise for D^s that it has a single definition (DF), can be a CN or, more generally, a (complex) particular PT—in case one considers instances—or (complex) universal U when one desires to granulate an ontology of types or scientific theory. This, however, raises an issue for the formal characterisation: D^s should either instantiate instances x, y, ... so that we remain in First Order Logic, or we have ϕ, ψ, ... ranging over universals, which requires Second Order Logic. This was mentioned in §2.3, where the formalisms are the same except for swapping the variables ranging over individuals for variables ranging over universals. We take the same approach in this chapter. This means that for any particular domain granularity framework, it is populated either with universals (or concepts) or instances, but not both at the same time. With this clarification, we now can proceed to the definition of domain.

3.2.3 Defining D

Given the analysis in the preceding two sections, we can formally characterise the domain D and its two subtypes—outer framework D^f and as subject domain D^s—as follows.

DEFINITION 3.1 (Domain)**.** *For each domain x there exist at least a y and exactly one z such that $D(x) \triangleq D^f(y) \vee D^s(z)$, and*

(i) iff $D^f(y)$ is the outer framework of the granularity, $\exists!w$ s.t. it has a definition $DF(y,w)$ and it is a concept $CN(y)$, and

(ii) iff $D^s(z)$ describes the subject domain, $\exists!v$ s.t. it has a definition $DF(z,v)$ and is either a (complex) universal $U(z)$ or a (complex) particular $PT(z)$.

Note that with full integration of the TOG with DOLCE, "$D^s(z) \rightarrow CN(z)$" can be omitted from the definition, because $CN(x) \rightarrow PT(x)$. We can observe the following from this definition and previous explanation.

PROPOSITION 3.1. *For any instance $D(x)$, $D^s(y)$, $D^f(z)$, there is an* inclusive-or *constraint on the subtypes, i.e., for each x there must exist at least a y or a z that are subsumed by x, or both:*

$$\forall x(D^s(x) \rightarrow D(x)) \tag{3.1}$$
$$\forall x(D^f(x) \rightarrow D(x)) \tag{3.2}$$
$$\forall x(D(x) \rightarrow D^f(x) \vee D^s(x)) \tag{3.3}$$

Further, we need a relation between D^f and D^s to be able to state that the former $granulates$ the latter and constraints over this relation because one cannot granulate without having a demarcation of the subject domain.

DEFINITION 3.2 (granulates)**.** *The relation* granulates(x,y) *holds if $D^f(x)$ and $D^s(y)$, that is:* $\forall x,y(granulates(x,y) \rightarrow D^f(x) \wedge D^s(y))$.

PROPOSITION 3.2. *For each $D^f(x)$, there must be a $D^s(y)$ that x granulates:* $\forall x(D^f(x) \rightarrow \exists y\; granulates(x,y))$

We can make *Proposition 3.2* ontologically stronger by asserting that D^f is *one-sided dependent*, $OD(\phi,\psi)$, on D^s [2], *i.e.*, for $OD(x,y)$ we have that x depends on y iff, necessarily, y is present whenever x is present, provided $NPED(x)$ and $PED(y)$ (see also Masolo *et al.*, 2003, Fig.5 p30); hence, this holds if the D^s represents endurants in physical reality[3].

LEMMA 3.1. *D^f is one-sided dependent on D^s only if D^s is a PED and D^f granulates D^s:* $\forall x,y((D^s(y) \rightarrow PED(y)) \wedge granulates(x,y) \rightarrow OD(D^f,D^s))$

Proof. Given that

[2]Note that ϕ and ψ are syntactic sugar (see Common Logic http://philebus.tamu.edu/cl/), with the relevant subsection summarised in (Masolo *et al.*, 2003, p26) and included in §3.8.1.

[3]Observe that this does not imply the granularity frames cannot exist without having it filled with actual data, *i.e.*, there is a difference between the domain granularity framework and the *applied* domain granularity framework in some software system: the latter must contain data (entities (/types)). See also *Definition 4.1* and *4.2* in Chapter 4.

(i) $\forall x(D^f(x) \rightarrow \exists y\ granulates(x,y))$ (*Proposition 3.2*)
where $D^s(y)$ (from *Definition 3.2*),

(ii) one-sided dependence from DOLCE (Dd69-Dd73):
Dd69 $SD(x,y) \triangleq \Box(\exists t(PRE(x,t)) \wedge \forall t(PRE(x,t) \rightarrow PRE(y,t)))$
Dd70 $SD(\phi,\psi) \triangleq DJ(\phi,\psi) \wedge \Box\forall x(\phi(x) \rightarrow \exists y(\psi(y) \wedge SD(x,y)))$
Dd71 $GD(\phi,\psi) \triangleq DJ(\phi,\psi) \wedge \Box(\forall x(\phi(x) \rightarrow \exists t(PRE(x,t)) \wedge \forall x,t((\phi(x) \wedge At(t) \wedge PRE(x,t)) \rightarrow \exists y(\psi(y) \wedge PRE(y,t))))$
Dd72 $De(\phi,\psi) \triangleq SD(\phi,\psi) \vee GD(\phi,\psi)$
Dd73 $OD(\phi,\psi) \triangleq De(\phi,\psi) \wedge \neg De(\psi,\phi)$
where DJ disjoint, PRE being present, and De *De*pendence (D in DOLCE, but the label is changed to avoid confusion with TOG's D).

(iii) $\phi \rightarrow NPED$ and $\psi \rightarrow PED$ for OD (by Ad74 in Masolo *et al.*, 2003),

(iv) $D^f(x) \rightarrow CN(x)$ (*Definition 3.1*),

(v) $CN(x) \rightarrow NPED(x)$ (from DOLCE)

therefore $D^f \rightarrow \phi$ holds. For OD to hold among D^f and D^s, we have (i) and also that D^s must be a ψ, hence, by (iii), then $D^s \rightarrow PED$. □

Observe that this does not imply that D^s can *only* be a PED, but just that the theory entails that if it is a PED, then OD can be derived in addition to mandatory participation of D^f in $granulates$. We can prove the participation constraint by D^s in $granulates$ as follows.

LEMMA 3.2. *For each $D^s(y)$, there is a 1:n participation on $granulates$ with (instances of) $D^f(x)$: $\forall x,y,z(granulates(x,y) \wedge granulates(x,z) \rightarrow y = z)$.*

Proof. Given $D^f(x)$, *Definition 3.2*, *Proposition 3.2*, (3.3), and any $D^s(y)$, there may or may not be a $D^f(x)$: (i) regardless if there is a $D^f(x)$, (3.3) holds because of the "$\vee$", and (ii) *Proposition 3.2* holds if there is either (ii-a) ≥ 1 x, and then there is also a $D^s(y)$ or (ii-b) trivially returns true if there is no x. Thus, multiplicity of D^s:D^f is at least 1:n, where $n \geq 0$. □

Lemma 3.2 says there can be a domain that is not granulated (yet), whereas with *Proposition 3.2* a granularity framework cannot exist without some subject domain it demarcates.

Last, *Proposition 3.3* is added, because there is, realistically, no reason to have some D^f without actually having at least one perspective (hence, levels) granulating the subject domain. To relate D^f to GP, we use the relation RE that will be discussed in detail in §3.5.

PROPOSITION 3.3. *Each $D^f(x)$ contains ≥ 1 $GP(y)$: $\forall x(D^f(x) \rightarrow \exists y\ RE(x,y))$*

To link the previous definitions, propositions, and lemma to the theory in §3.8, we summarise constraints on D and list the relevant axioms of the formal characterisation in §3.8.

A. *Definition 3.1*: The domain has the dual purpose to demarcate the subject domain and as outer frame to contain the granular perspectives (D.1, D.2, A.93, A.94).
B. *Proposition 3.1*: inclusive-or on D^f and D^s (A.89, A.90, A.91).
C. *Definition 3.2* and *Proposition 3.2*: the $granulates$ relation with mandatory participation by D^f (A.98, A.97, A.67, A.66).
D. *Lemma 3.1*: One-sided dependence if D^s is a PED (D.19-D.23, A.98).
E. *Lemma 3.2*: D^s can exist without a granularity framework D^f, hence may have zero or more instances of D^f for that subject domain (A.68).
F. *Proposition 3.3*: D^f contains ≥ 1 GP (A.109).

Observe that these constraints, and subsequent ones in the following paragraphs, are for the case of one domain only; interactions between multiple domains are discussed in §5.5.1.

3.3 The granular perspective GP

Questions and tasks. Define what a granular perspective, GP, is. Is 'perspective' synonymous with other used notions such as dimension and role, and if not, what are their differences and which ontological commitment does it entail for defining GP? Characterise the criterion used to granulate.

When assessing which perspectives exist, this is within domain D^f, hence GP does not need the machinery of what is defined at D^f because that is already taken care of with D^f. Because GP is contained in D^f, it is restricted by the domain. A finite amount of perspectives may be identified, but within one D^f, no two perspectives are the same, because using the same granular perspective in one domain more than once is redundant (see below for justification). Like D^f, GP is a concept CN because the perspective is another part of the framework for granularity. At the other end of the spectrum are the granular levels, GLs, that GP contains, but the perspective does not need to be aware of which levels it contains, just that it contains levels; hence, we can set levels aside for the remainder of this section.

The next step is to analyse what granular perspectives are. In Chapter 2 we identified ways *how to granulate* that will be reused here, but we also look at how to represent the characteristics of *what to granulate* within a demarcated domain. Three possibilities are investigated in §3.3.1, which are dimension, role, and property. Role and dimension are both useful for a perspective but do not suffice for representing the full semantics of a granular perspective. As preliminary, recollect §2.3.1 p.33 items i & ii on concepts and universals, and that several kinds of properties exist (which will be elaborated on below). In addition, given the use of properties for granulating contents, this indicates characteristics that GP must bring in to obtain a coherent hierarchy of levels. That is, in addition to the type of granularity, there is a *criterion for granulation* as property of GP by which one demarcates a particular aspect of the D^s; these aspects will be specified in §3.3.2.

3.3.1 Roles, dimensions, and properties

Roles. A role has a base entity (/type) on which it is dependent. During some time of its life, the entity can move into one or more roles, like the person Francesco Ranieri can be rector of a university in Calabria and member of the Italian parliament; hence there is a relation between the concept and a role it plays (Masolo *et al.*, 2004). A role does not capture an intrinsic property of a concept and can be assigned by social convention, which is different from being x or being function y for biological entities that exist independent of human cognition. This, of course, is taking the philosophical position that (biological) entities exist independent of human thought. With infectious diseases, one has, *e.g.*, participates_in(Blood, Person-to-person transmission) where blood may have the role of transmitter of infectious agents, but blood always has the capacity to do so and there are infectious agents that rely on this function for their survival. Alternatively, Blood is a compound property being a transmitter of infectious agents, where Blood as anatomical entity type is the bearer of the extrinsic property of transmitting infectious agents. Either way, it is possible that a granular perspective is constructed according to the type of role, just *not all* perspectives. Because roles, and extrinsic properties in general, do not fully address the ontological nature of a granular perspective, it is insufficient to be the sole characteristic at the domain-independent level of a theory of granularity.

Dimensions. A perspective and dimension differ in their intended meaning and use. A *dimension* can be synonymous with *one* property or attribute to distinguish universals and their instances, or, in Gärdenfors' (2000) interpretation, quality properties like Colour have *quality dimensions* (hue, brightness, and saturation) where a "conceptual space" is a constellation of one or more quality dimensions, forming an n-dimensional representation of a phenomenologically perceived property (Gärdenfors, 2000); thus, a combination of dimensions makes up a quality, and a quality is a kind of property. When one takes a perspective and orders the entities (/types) according to their level of granularity, one or more properties, hence also dimensions, of the object are selected. That is, by taking a perspective, one chooses one or more properties, *but not all properties* an object instantiates; thus, using 'granular *dimension*' sensu Gärdenfors limits us to granulate by one dimension at a time only, whereas a granular *perspective* can consist of a specific combination of properties and dimensions, hence provides more flexibility to capture the perceived views in ordering, or discovering the order of, the entities of a subject domain. The latter is also used in multi-dimensional modelling for data warehouses. The problem with the data warehousing dimensions in star and cube models, however, is that they are arbitrary combinations of properties (see also §5.2.2). On the other hand, when we take as example the granular perspectives of infectious diseases for humans (Keet and Kumar, 2005), the Source of infection has a further restriction with Mode of transmission of the infectious organism to identify granular levels, and another one for Infectious organism, which are not just arbitrary combinations, but, in this example, a two-step demarcation (see also *Appendix A*). A particular granular perspective in a subject domain may be characterised by these properties, therefore GP requires a relation to the criterion for granulation, C, to delimit what a GP is. C is a combination of properties used to construct a perspective, where the combination is less than the complete pattern of properties that describes the universal and its instances that will populate the granularity framework, and is at least one dimension, although normally more than one. This use of universal corresponds to the least constrained definition, which is easier to map to a computer-understandable representation than the full definition (§2.3.1 p.33). This will be discussed and clarified in the next two sections.

Properties. Both roles and dimensions address the ontological nature of a granular perspective insufficiently. Their common denominator, however, is that they are both *properties*. Thus, the position taken here is that properties exist. Before we consider properties in conjunction with granularity, three points about properties will be highlighted, of which the third item will be explained afterward.

1. *There are multiple kinds of properties*. A summary of properties commonly used in knowledge representation and several typical examples are included in *Table 3.1* (debate about the sense or nonsense of one or the other is beyond the current scope).
2. *Properties have (meta-)properties, which enables categorisation of properties* that brings results of Ontology closer to computational use. Most notably is the formal ontology of properties by Guarino and Welty (2000a,b) that forms the basis of the OntoClean methodology for ontology development (Guarino and Welty, 2004). Their taxonomy of properties is based on the analysis of results from philosophy and the use of meta-properties rigid, non-rigid, anti-rigid, and semi-rigid, carrying versus supplying identity, and dependence. For reasons of brevity, we only include the taxonomy of properties and illustrative examples in *Figure 3.2*; consult Guarino and Welty (2000a) for details.
3. *Knowledge representation is flexible about representing properties*; that is, how properties are formally represented depends on the language used.

Given the variety in properties, they can end up in a logical theory in different ways. It is important to stress that logic does not deal with if some property x should be represented as, say, a UML-class or DL-concept, as a relation, or as an attribute—ontology does. There is the ontology about properties on the one hand, and logic-based knowledge representation on the other. For purpose of transparency and domain-expert understandability, it is beneficial if a knowledge representation language can represent the subject domain semantics in a manner that is as close as possible to a modeller's understanding of that piece of reality, but for efficient computation, less intuitive representations tend to be more effective. *In casu*, recollecting the definitions of universal (page 35), this ends up in UML class diagrams as a class with attributes (see §7.3.7

Table 3.1: A selection of kinds of properties and typical examples (based on Swoyer (2000)).

Kind of property	Comments and examples
Relational	Property with more than one argument place, *e.g.*, being married
Internal relation	Relational analogue of essential property, so that if $R(a, b)$ and a and b both exist, then they must be related through R that relates a and b internally, *e.g.*, being a biological parent of
External relation	A relation that contingently relates their relata (domain and range), not internal, *e.g.*, being married relating your mother and your father (they may not have been married)
Particularizing	Provides principles on identity, also called sortal property, *e.g.*, being a chair, phosphorylation
Characterizing	A type of particularizing property that "do not divide the world up into a definite number of things", *e.g.*, being square redness requires a thing (to be square-shaped or red, respectively)
Mass	A type of particularizing property that refer to amounts of matter and, generally, are not countable, *e.g.*, wine
Natural kind	They "carve nature at its natural joints", *e.g.*, being a protein
Artificial kind	Contrasted with natural kind properties, *e.g.*, being a television
Essential (rigid)	The individual always has that property for the time of the individual's existence, *e.g.*, being dog is an essential property of Lassie
Accidental	A property that a thing just happens to have but could have been otherwise, *e.g.*, Employee, for one can be without a job
Intrinsic	Non-relational property, a property that a thing has independent of its relations to other things
Extrinsic	An object has it because its relation to another thing, *e.g.*, married to
Primary	Objective feature of the world, *e.g.*, size, protein conformation
Secondary	They "somehow depend on the mind", taste, smell

in OMG UMLSpec (2005) for a detailed description), in WSML as concept with attributes (De Bruijn, 2007), and in DL languages as concept with one or more roles (relations) that may have their range defined (see *Appendix C* for an introduction to DL). The DL languages, like ORM, are attribute-free languages, because they have a flat representation with DL-concepts and DL-roles. With UML and WSML, one has attributes representing a property 'inside' the class or concept so that one neither can describe the relation between the concept and its attribute nor reuse the attribute 'outside' that class in another class definition. There are several mappings and transformations between the various knowledge representation languages (among others, Berardi *et al.*, 2005; De Bruijn and Heymans, 2007; Halpin, 2001; Keet, 2007b), and for all practical purposes, the *formal representation of a universal amounts to a non-empty collection of properties*. The following example illustrates this and extracts their respective implicit ontological commitment on representing properties.

Example 3.1. *Figure 3.3* shows the universal Blood with several properties and related entity types according to four different representations that, implicitly, do not adhere to the same ontological commitment.

i. *Figure 3.3-A* represents Blood as DL-concept with several attributes and multiple inheritance. The properties range from characterising properties, such as Viscosity, to the sortal properties BodySubstance, Transmitter of infectious agents, and Chemical transporter. Assessing accuracy of the first versus the alternative representation reveals an issue of point of view and essential versus accidental property. From a functional perspective, the property of being a Chemical transporter is the core function of blood, transporting molecules such as oxygen and metabolic waste compounds. In addition, concerning blood's involvement in the process of transmission of infectious agents such as HIV, then from the viewpoint of a human it may not be a

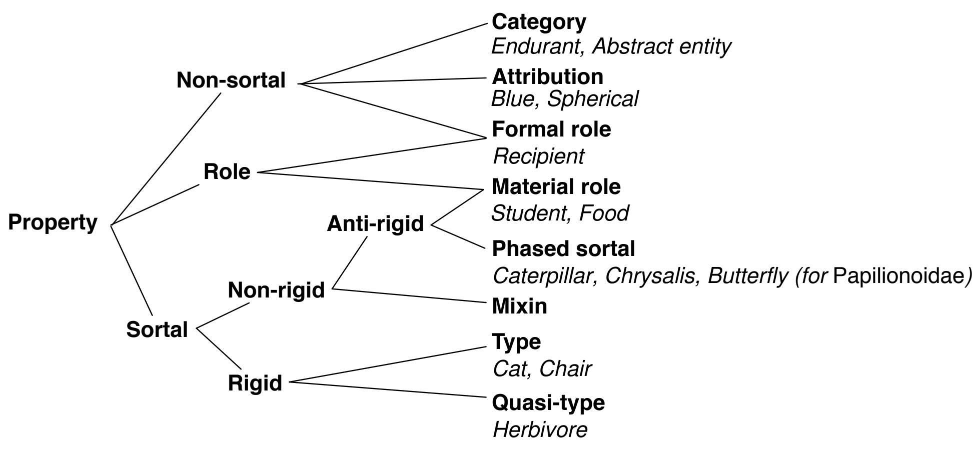


Figure 3.2: Taxonomy of properties with several examples (Guarino and Welty, 2000a).

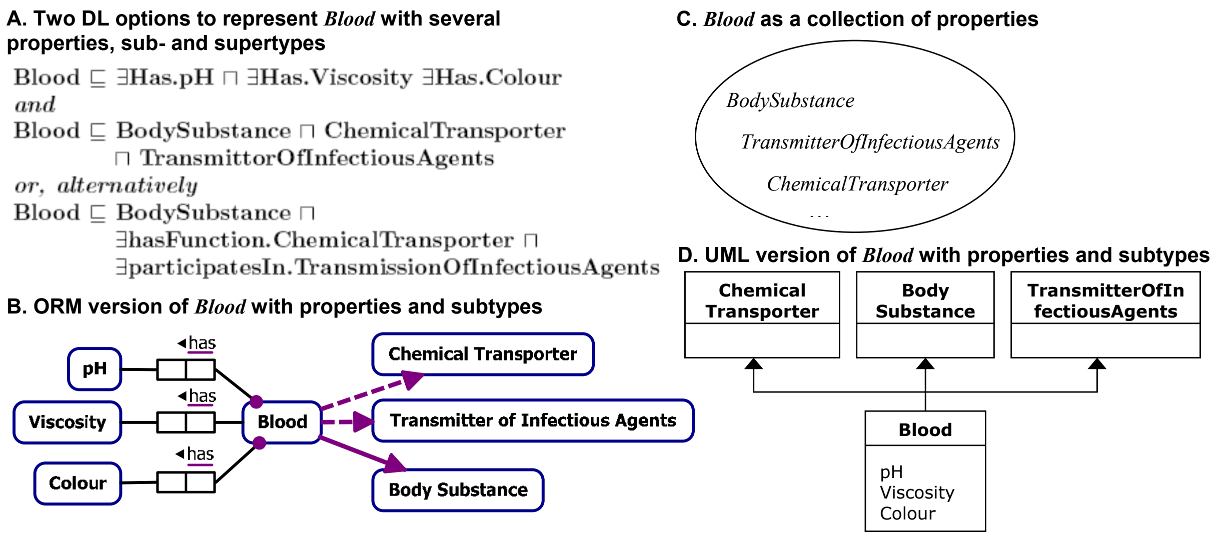

A. Two DL options to represent *Blood* with several properties, sub- and supertypes
Blood ⊑ ∃Has.pH ⊓ ∃Has.Viscosity ∃Has.Colour
and
Blood ⊑ BodySubstance ⊓ ChemicalTransporter ⊓ TransmittorOfInfectiousAgents
or, alternatively
Blood ⊑ BodySubstance ⊓ ∃hasFunction.ChemicalTransporter ⊓ ∃participatesIn.TransmissionOfInfectiousAgents
B. ORM version of *Blood* with properties and subtypes
pH
Viscosity
Colour
has
has
has
Blood
Chemical Transporter
Transmitter of Infectious Agents
Body Substance
C. *Blood* as a collection of properties
BodySubstance
TransmitterOfInfectiousAgents
ChemicalTransporter
...
D. UML version of *Blood* with properties and subtypes
Chemical Transporter
Body Substance
TransmitterOfInfectiousAgents
Blood
pH
Viscosity
Colour

Figure 3.3: Blood with several sortal and characterising properties modelled in three distinct representations. See text for explanation.

natural property of blood to be involved in transmission of infectious agents. However, blood *is* an essential transmission mechanism for HIV and an essential factor in its propagation cycle. Comparing this with *C. tetani* in *Example 3.2*, then the difference is that HIV *relies* on human blood, whereas *C. tetani* is an anaerobic soil bacterium *independent* of humans: *C. tetani* and tetanospasmin were in nature before humans and will be there after *Homo sapiens* cease to exist, whereas HIV infects only humans who are both its host and reservoir.

ii. *Figure 3.3-B* shows an ORM2 model drawn in the NORMA CASE tool. The tool allows only *one* primary subsumption relation for the first is_a drawn, even though none of the underlying FOL formalizations of ORM Halpin (1989); Hofstede and Proper (1998) make such a distinction. Moreover, who decides—and what is the ontological justification for—that Blood is a Body Substance takes precedence over Blood is a Chemical Transporter? When two properties are rigid then that is what they are and not that one is 'more rigid' than the other. Choosing a "primary" preferred property may be an engineering solution (Rector, 2003), but, apart from being ontologically incorrect, it is inflexible and prone to errors, because another software system covering the same domain may just have chosen the primary as secondary and vice versa, thereby hampering integration of the data sources, exactly what ontologies are supposed to solve. In addition, multiple inheritance is distinct from a collection of properties. In contrast with ontology development practices to avoid multiple inheritance, here it is due to features in the conceptual data modelling language and software tool exactly the intention to have the multiple inheritance as a means to represent the properties applicable to Blood.

iii. *Figure 3.3-C* has type Blood as the circle and a collection of properties. To represent fully what it is to be blood, each sortal property should be included; the diagram could be augmented with other kinds of properties. This notion corresponds better to the ontological status of Blood than *Figure 3.3-B* and *D*.

iv. *Figure 3.3-D* shows an UML class diagram with attributes inside the class definition and three superclasses for Blood but without prioritization of superclass.

Approach (iii) provides the underpinning for linking levels (§3.7.2), where selecting a subset of the relevant properties does *not* make it another entity (/type), but only that part of the universal is highlighted. ♢

Relating this information about properties and the examples to granularity and granular perspectives in particular, then what is of most interest are the intrinsic, extrinsic, essential, and accidental properties and, consequently, the taxonomy of properties with the rigid properties, roles, and non-sortals. The next example illustrates the difference between essential and accidental properties.

Example 3.2. The bacterium *Clostridium tetani* is not 'a devious bacterium on earth to infect humans, produce neurotoxins, cause tetanus and death'. *C. tetani* produces a zinc-endopeptidase—as toxin the enzyme is called tetanospasmin—which is involved in normal cellular mechanisms of exocytosis and endocytosis (Montecucco and Schiavo, 1995; Rossetto, 2000), the zinc-endopeptidase is always that throughout its life span *and* essential to the bacterium. That this is not beneficial for humans is coincidental tough luck. Humans attribute the function of being a Toxin producer to *C. tetani*, but it is not a *natural* property of the bacterium to produce the molecule as defence mechanism, competitive advantage, or to intentionally harm humans—metallopeptidases are of themselves functionally not toxins. Thus, *C. tetani* is, *accidentally*, a Toxin producer and the zinc-endopeptidase, *accidentally*, a toxin. ♢

Then, if we granulate infectious diseases by causative organisms or toxins, we would not bother about zinc-endopeptidases and, vice versa, granulating proteins and enzymes on their biological properties, the property of being a toxin is out of view. Put differently, the *granulation occurs by highlighting some, but not necessarily all, properties of an entity (/type)*. The other factor is that it may not matter if a property is accidental or essential when we granulate information and that both could be used, *as long as one is consistent in applying one kind of property*. This is also enforced with the OntoClean methodology for ontology development and those principles apply equally to granulation. Regarding the TOG, without considering the actual data to be granulated, it may not matter, but it does when one declares granular perspectives for a subject domain, suggesting one cannot ignore this at the meta-level. Thus, we return to the ontological commitment of realism with grounding in reality versus the freedoms that come with conceptualizations; but this concerns representing the subject domain, not the granularity framework, and good ontology development practice should prevent mixing accidental and essential properties. Making the kind of property, or properties, for granulation explicit aids ensuring the kind of property used is consistent throughout.

Thus, given the properties and examples, we have the notion of a perspective by *alternating sortal properties* taken from the full set of properties, which prevents arbitrary choices for "primary properties" (Rector, 2003), and the distinction between properties assigned to it by humans and co-existent perspectives with properties that entities have of themselves (*Example 3.2*). This leaves us with intrinsic properties of the entity type itself and extrinsic properties of some object x that is related to y and for y that property is accidental. One can take the liberal modelling decision that *there is no subjective preference order for the sortal properties of the universal.* Ontologically, one might object to treating essential & accidental and intrinsic & extrinsic properties as equals, but for granularity it is more important that one *does* granulate according to specific properties. Given that in each perspective only certain properties of entities (/types) are selected or highlighted, then GP must have some means to cater for and delimit the kind or category of properties according to which the domain is partitioned, levels identified, and subject domain granulated; this is labelled with *criterion for granulation* and the topic of the next section.

3.3.2 The criterion for granulation

Each granular perspective has a criterion associated with it, which provides the properties by which one performs the granulation. Winther (2006) and Chen and Yao (2006) allude to using criteria as well, albeit philosophically in an informal manner or formally but lacking an ontological foundation, respectively. Here I will characterise the criterion for granulation more precisely. The following example describes several examples of granular perspectives taken from the literature that use and discuss biological granularity.

> **Example 3.3.** In the paragraph about dimensions, we had an example where there were two components for the perspective to identify granular levels: Source of infection with either Mode of transmission or Infectious organism. Other examples are Site with modifier Site of entry or Site of effect, and Mode of action in the pathology with Function as particular property for the modes of action of the infectious agent. Eight of the nine granular perspectives identified by Keet and Kumar (2005) had two components for the criterion for granulation; the remaining Predisposing factors perspective needs a further qualification to split up taxonomic and partonomic granulations (Keet, 2006c, and gp'_9 in *Appendix A*). Other examples of criteria in biomedicine are Human structural anatomy and Cancer growth activities at different levels of granularity (Grizzi and Chiriva-Internati, 2005; Kumar *et al.*, 2005; Ribba *et al.*, 2006). Several ecosystem hierarchies have been proposed, which were analysed by Salthe (1985) on consistency in applying a granulation criterion. Salthe proposes a Genealogical hierarchy of nature that takes *time* into account and Ecological hierarchy of nature for energy exchanges in systems of *spatial* extent. Mota *et al.* (1995) aggregate ecological processes at different time granularities and Sorokine *et al.* (2006) combine ecological units with scale. $\diamondsuit$

These examples, however, do not suffice to make criterion C always a combination of two properties as general constraint for all granular perspectives. C is a combination of properties, which could be modelled as a compound property. If it is a compound property, then one has to introduce λ as variable-binding operator in order to construct it from open formulas. For example, with a combination of two properties $A(x) \wedge B(x)$ a one-place complex predicate $\lambda x(A(x) \wedge B(x))$ can be created, such that x is both A and B. This approach requires additional machinery in the form of "property-building" operations and a comprehension schema (Swoyer, 2000), thereby complicating the formalisation and its subsequent use. It is possible to avoid this for two reasons. First, the criterion for granulation may not have two, or more, properties of equal importance with respect to the particular perspective, but a sequence with a step-wise demarcation toward the focal property for granulation. Second, the combination of properties is different between GPs that have as granulation scale- vs. non-scale-dependent types of granularity, which adds to the first point that x does not instantiate both A and B. Characterising the criterion for scale-dependent granularity is different because some scale is *always* involved, which, if it were combined as compound property, would result in an ambiguous ontological status. For a scale-dependent granular perspective, the criterion C is a combination of a sortal property and a quality property. For example, Surface with 'modifier' the Surface metric scale measured in, say, m^2, dam^2 etc for **saoG** type of granularity, or the 3D shape with **sgpG**. In DOLCE terminology, scales are mapped to Abstract Quality (AQ) and their values to Abstract Region (AR). Relating this to the TOG graphical rendering in *Figure 3.1*, we now have completed the path from Granular_Perspective via RC to Criterion, which combines properties of which one is a Quality_Property, provided the type of granularity that is used with Granular_Perspective is of the type sG. Observe that it is not a requirement for the properties of the criterion to be a property of the granulated entities, but only related to a property of the entities (see also §3.4.1).

Concerning characterisation of the criterion of a granular perspective for non-scale-dependent types of granularity, the criterion supplies at least the category to which the properties of the entities (/types) belong. Further, recollect that the properties combined for a single criterion are less than or equal to the full combination of properties that make up the universal or concept, or instances thereof. For instance, Pathological processes of infectious diseases with **nrG**, then we can have a granulation with processes and part-processes—hence, using at least a category—and an entity type, such as Congestion, is involved in Inflammation, and possibly could reside in some other granular perspective as well so that not all properties of Congestion are taken into account in this granular perspective.

It has to be noted that both the selection of which properties to use for granulation and how they are combined to form the granulation criterion is still underspecified and serves further ontological investigation by philosophers, which is outside the scope of the thesis. Therefore, the notion of criterion is liberally formulated as being a combination of either at least two properties, $Prop$, or at least one property and a quality property, Q, that has a measurable region where $\forall x(Q(x) \rightarrow Prop(x))$. C can be defined as:

DEFINITION 3.3 (Criterion). *Each criterion C is a combination of either*

- $\exists^{\geq 2} y(Prop(y) \wedge \neg Q(y))$*, i.e., at least two properties $Prop$ but not a quality property Q, or*
- $\exists y \exists! z(Prop(y) \wedge Q(z) \wedge \neg(y = z))$*, i.e., at least one $Prop$ and exactly one Q.*

which are related to C through the CP relation.

From this definition, four aspects have to be taken into account: (i) a way to relate the properties to the criterion it is used for with CP (*Definition 3.4*) and the participation constraint that follows from *Definition 3.3*; (ii) following from *Definition 3.3* and the preceding analysis, when a Q is part of C then we deal with scale-dependent granularity (*Proposition 3.4*); (iii) a means to record the values of Q and $Prop$ through $has\_value(x, y)$ (*Definition 3.5* and *Proposition 3.5*)[4]; and (iv) a value's upward distributivity from property to its criterion in *Proposition 3.6*. The usefulness of these propositions will be shown in the next section about granular levels with *Lemma 3.12*.

DEFINITION 3.4 (CP). *The relation CP relates a criterion C to the properties it combines:* $\forall x, y(CP(x, y) \rightarrow C(x) \wedge Prop(y))$*, where there are at least two properties participating:* $\forall x(C(x) \rightarrow \exists^{\geq 2} y\, CP(x, y))$

PROPOSITION 3.4. *If a criterion C has at least one $Prop$ and exactly one Q, then this is associated with granulation of type **sG**.*

DEFINITION 3.5 (has_value). *The has_value relation relates a property with its value:* $\forall x, y(has\_value(x, y) \rightarrow Prop(x) \wedge V(y))$.

PROPOSITION 3.5. *Each quality property $Q(x)$ has some value $V(y)$, which are related through the relation $has\_value(x, y)$:* $\forall x(Q(x) \rightarrow \exists y(has\_value(x, y)))$.

PROPOSITION 3.6. *By upward distributivity, value(s) of the property/ies $Prop$ and/or Q of the criterion are also values of the criterion C:* $\forall x, y(has\_value(x, y) \rightarrow \exists z(has\_value(z, y) \wedge C(z)))$.

With these characteristics of the granulation criterion, we now can proceed to defining GP.

[4] $has\_value(x, y)$ corresponds in spirit to "ql" in DOLCE, but then for types and where values are denoted with *R*egion, which is Value here.

3.3.3 Defining GP

Recollecting the introduction of this section, C provides the *what* is to be granulated in addition to the *how* provided by the type of granularity (TG). These two TOG components have to be related to GP before defining granular perspective. The former is done through RC and the latter through $has\_granulation$. The relation between GP and its C, RC, is defined as:

DEFINITION 3.6 (RC). *Relation $RC(x, y)$ holds between perspective $GP(x)$ that has criterion $C(y)$: $\forall x, y(RC(x, y) \rightarrow GP(x) \wedge C(y))$.*

In addition to the basic typing of the RC relation, a mandatory participation can be added to RC, because there is no reason to have a criterion in a domain granularity framework without actually using it.

PROPOSITION 3.7. *Each criterion must participate in a RC: $\forall x(C(x) \rightarrow \exists y\ RC(y, x))$.*

Likewise, one can neither use more than one criterion for one perspective nor use none for then there is nothing to granulate, therefore we add proposition *Proposition 3.8*. The intuition of this proposition is that ontologically, it is nonsense to combine, say, $c_1 =$ Human pathological processes at different levels of granularity with $c_2 =$ Mouse structural anatomy at different levels of granularity to make one single hierarchy of levels.

PROPOSITION 3.8. *Each perspective has exactly one criterion: $\forall x(GP(x) \rightarrow \exists! y\ RC(x, y))$.*

$has\_granulation$ relates a type of granularity to a perspective where the greek letters are syntactic sugar for the eight leaf types (see footnote 2 and §3.8.1).

DEFINITION 3.7 (has_granulation). *The relation* has_granulation(x,ϕ) *holds if $GP(x)$ and $TG(\phi)$ where TG is the type of granularity: $\forall x, \phi(has\_granulation(x, \phi) \rightarrow GP(x) \wedge TG(\phi))$.*

Recollecting Chapter 2, one always uses a type of granularity for granulating the data; hence, we have a mandatory participation of GP in the $has\_granulation$ relation. In addition, one should not mix different ways of granulating data within one perspective lest the hierarchy of levels will be inconsistent; hence, each perspective has a maximum of one TG:

LEMMA 3.3. *Each perspective has exactly one type of granulation:*
$\forall x(GP(x) \rightarrow \exists! \phi\ has\_granulation(x, \phi))$.

Proof. First, if one does not use a type of granularity at all, then one does not granulate, because it would negate any structure among entities (§2.3). Second, each type of granularity (TG) in the taxonomy is disjoint (§2.2) and for each leaf type, the structure of the contents is different (§2.3), hence combining two or more types leads to a contradiction. □

Finally, we arrive at the definition of granular perspective GP.

DEFINITION 3.8 (Granular perspective). *$\forall x \exists! w, y, z, \phi$ such that $GP(x)$ is a concept $CN(x)$, has a definition $DF(x, y)$, relates to its criterion $C(z)$ through the relation $RC(x, z)$, has_granulation type $TG(\phi)$ and is contained in $D^f(w)$.*

Following from the definitions and propositions, *Lemma 3.4*—identifying a path between C and TG through GP—can be proved now, but that was only implicit in Chapter 2.

LEMMA 3.4. *If $C(x)$ has a $Q(y)$ and $RC(z, x)$, then $GP(z)$ has granulation type **sG**:*
$\forall x \exists z, \phi((C(x) \rightarrow \exists! y(CP(x, y) \wedge Q(y))) \wedge RC(z, x) \wedge has\_granulation(z, \phi) \rightarrow (\phi \rightarrow sG))$.

Proof. First, *Definition 3.3* can be formalised as
$\forall x((C(x) \rightarrow \exists^{\geq 2} y(Prop(y) \wedge \neg Q(y))) \veebar (C(x) \rightarrow \exists y \exists ! z(Prop(y) \wedge Q(z) \wedge \neg(y = z)))$
Given we have a Q, then the second part after the exclusive-or in *Definition 3.3* must hold. Second, we have the typing of RC and mandatory constraint
$\forall x, y(RC(x, y) \rightarrow GP(x) \wedge C(y))$ (*Definition 3.6*)
$\forall x(C(x) \rightarrow \exists y\, RC(y, x))$ (*Proposition 3.7*)
therefore, there has to be an instance of GP (first argument in RC). Given this instance
$\forall x, \phi(has\_granulation(x, \phi) \rightarrow GP(x) \wedge TG(\phi))$ (*Definition 3.7*)
$\forall x(GP(x) \rightarrow \exists ! \phi\, has\_granulation(x, \phi))$ (*Lemma 3.3*)
therefore, there must be a ϕ that is a TG. By having Q (first point) and *Proposition 3.4*, then $\phi = sG$, therefore $GP(z)$ has granulation type **sG**. □

From the proof of *Lemma 3.4* it follows immediately that the other half of the definition of C applies to **nG** (*Corollary 3.1*), due to the exclusive-or in *Definition 3.3* and disjoint subtypes in the taxonomy of types of granularity.

COROLLARY 3.1. *If $C(x)$ has ≥ 2 properties $Prop(y)$ and $\neg Q(y)$, then $GP(z)$ has granulation type **nG**.*

Now we add an interesting property of granular perspectives concerning reuse of criteria (*Lemma 3.5*), from which follows that the combination of criterion and type of granulation determines uniqueness (*Theorem 3.1*), hence, together they provide the necessary and sufficient conditions for identity of GP.

LEMMA 3.5. *A criterion C can be used with more than one perspective GP, provided the perspectives have distinct granulation types TG: $\forall x_1, x_2, y, \phi_1, \phi_2(RC(x_1, y) \wedge RC(x_2, y) \wedge has\_granulation(x_1, \phi_1) \wedge has\_granulation(x_2, \phi_2) \wedge \neg(x_1 = x_2) \rightarrow \neg(\phi_1 = \phi_2))$.*

Proof. For each GP we have a $C(y)$ and a $TG(\phi)$, because of
$\forall x(GP(x) \rightarrow \exists ! y\, RC(x, y))$ (*Proposition 3.8*)
$\forall x(GP(x) \rightarrow \exists ! \phi\, has\_granulation(x, \phi))$ (*Lemma 3.3*)
Assume for some y and some ϕ, there is the same x. Let us reuse ϕ for some other perspective, $GP(z)$, *i.e.*, $has\_granulation(z, \phi)$ and assume $z \neq x$ hold. Let us also reuse y for some other perspective, $GP(w)$, *i.e.*, $RC(w, y)$ and assume $w \neq x$ hold. Then we have two cases:

(i) $w = z$: then by *Proposition 3.8* and *Lemma 3.3* either $w = z = x$ (thus contradicting the assumptions $z \neq x$ and $w \neq x$) or there is an elusive property α to negate the equality. There is no α, hence, it must lead to identity of GP with C and TG. Thus,
$\forall x_1, ..., x_4, y_1, y_2, \phi_3, \phi_4(RC(x_1, y_1) \wedge RC(x_2, y_2) \wedge has\_granulation(x_3, \phi_3) \wedge$
$has\_granulation(x_4, \phi_4) \wedge y_1 = y_2 \wedge \phi_3 = \phi_4 \rightarrow x_1 = x_2 = x_3 = x_4)$.

(ii) $w \neq z$: then by *Lemma 3.3*, we have $has\_granulation(w, \phi')$ and $\phi \neq \phi'$, and by *Proposition 3.8*, we have $RC(z, y')$ and $y \neq y'$.

Thus, reuse of criterion y with another TG, ϕ', is demonstrated in point (ii) with $GP(w)$. □

THEOREM 3.1. *The* combination *of some $C(y)$ with a $TG(\phi)$ determines uniqueness of each $GP(x)$.*

Proof. Follows from *Lemma 3.5*, point (i). □

From *Lemma 3.5* and *Theorem 3.1* we get a nice corollary (*Corollary 3.2*) so that the intuition mentioned in the introduction of this section about unique granular perspectives in a domain granularity framework to prevent duplication is ensured, because it follows trivially from the theory. (RE is the relation between perspective and domain, which will be discussed in detail in §3.5.2.)

COROLLARY 3.2. *Granular perspectives are unique within the domain they are contained in:*
$\forall x_1, ..., x_n, y(GP(x_i) \wedge D^f(y) \wedge RE(x_i, y) \rightarrow \neg(x_1 = x_2) \wedge ... \wedge \neg(x_{n-1} = x_n))$.

This can be trivially reformulated into that all GPs contained in a D^f are disjoint; note that one cannot derive a complete coverage unless one were to take a closed-world assumption.

Summarising the constraints on a granular perspective (as before and afterward, the numbers between braces refer to the axioms in §3.8), we have

G. *Definition 3.3*, *3.4*, *Proposition 3.4*, *3.5*, *3.6*, *Lemma 3.4*: Each criterion is composed of at least two properties. Of the properties for **sG** types of granularity, one is a measurable quality property, whereas the criterion of GPs with **nG** granulation have qualitative properties. Quality properties have values and upwards distributivity of the values to the criterion (A.63, A.64, A.65, A.87, A.88, A.99, A.107).

H. *Definition 3.8*: Each granular perspective has one criterion associated with it through the relation RC, which provides the properties by which one granulates (D.3, A.95, A.103).

I. *Definition 3.7*, *Lemma 3.3*: each granular perspective must have exactly one type of granularity (A.82, A.83).

J. *Corollary 3.2*: A granular perspective GP must be contained in some D^f and within one domain, granular perspectives are distinct (A.110).

K. *Lemma 3.5*, *Theorem 3.1*: The criterion can be reused for different perspectives, provided it is used with a different type of granularity, and the combination determines uniqueness of each perspective (A.105, A.106, T.6).

L. *Proposition 3.8*: Each GP has exactly 1 criterion C, related through RC (A.104).

3.4 The granular level GL

Questions and tasks. Define the granular level GL, including how to distinguish between two levels. What is (are) the identity criterion (criteria) of a level? Is there a difference in definition of GL caused by the difference in scale-dependent and non-scale-dependent GPs? What effects do interwoven and/or related concepts such as emergence, indistinguishability, and similarity have on the characterisation of GL? Is it true, and if yes how/why, that a level in one perspective is always different from a level in another perspective within one domain?

Like with the domain and granular perspective, here too it is important to make the distinction between the framework component—the granular level GL—and the instances or universals *inside* it that meet the constraints of its granular level GL. The specification of a level in a particular subject domain, gp_igl_i, is relevant only after defining its granular perspective gp_i and, consequently, demarcation of d_i^f. For instance, devising levels in the 'structure of the body' has a different meaning in biomedicine compared to social organisations. GL delimits what it is to be a level and of a certain level and, analogous to D and GP, has a definition and constraints, and is a concept, too. The criterion for granulation does not have to be re-defined for each granular level, because this is already taken care of by its GP's C, but values of GP's criterion are needed to distinguish between different levels in a perspective and to establish that no two levels are identical in one granular perspective. A level is bound to the perspective within it resides and does not exist on its own. Put differently: if one has a granular level, there *must* be a perspective it is contained in, lest one creates levels freely by combining types of granularity or mixing criteria that would result in inconsistent granulation. Moreover, the criterion C of GP is reused for each level, but where its values differ for each GL. For example, a Surface and Surface metric with three levels gp_1gl_1, gp_1gl_2, and gp_1gl_3 can have the values km^2, hm^2, and dam^2, respectively. Note that $gp_1gl_3 \prec gp_1gl_2 \prec gp_1gl_1$ is valid, which does not imply that there is a subclass relation between either the levels or its contents: dam^2 is not a taxonomic subtype of km^2 but a proper part of km^2. Scales of time durations, intervals, and granularity in calendar entities exhibit this confusion more readily (see also §5.3.3). Although this example with scale-dependent granular levels may seem straightforward, it brushes over indistinguishability, similarity, equivalence, and the interplay between these relations; this will be examined in detail in §3.4.1, where it will be shown that they are key ingredients for identifying granular levels. Further, it will be demonstrated that the indistinguishability relation is a special type of the equivalence relation and we demonstrate the difference in their intended usage: generalising versus specialising, respectively. By availing of similarity as well, we prove that there must be at least two levels of granularity in a granular perspective (*Theorem 3.2*). Last, from the proofs it also follows that there has to be a strict total order between fine- and coarse-grained granules (*Lemma 3.9*). We proceed to the definition of granularity in §3.4.2 and prove some additional properties of a granular level, such as that each level can occur only once in a perspective, that it must adhere to the same type of granularity as its perspective, and that each level adhering to **sG** has a maximum of two functions related to it for conversion between adjacent finer- and coarser levels.

3.4.1 Indistinguishability, equivalence, and similarity

There are two main ideas recurring in the literature in conjunction with granular levels and sets, which are similarity and indistinguishability. They emphasise *why* universals or particulars reside in the same level or not and consequently provide a rationale for allocating entities in a particular level. Similarity and indistinguishability are closely related properties, but their differences are salient when their use is examined not only for quantitative granular computing, but, moreover, for qualitative aspects (semantics) of granularity. Similarity will be summarised first, which is followed by indistinguishability and its parent relation equivalence; subsequently, their influence on granular levels will be defined formally.

3.4.1.1 Similarity

When testing for similarity, one takes two or more objects, compares them property & value by property & value and calculate their closeness, be it in geometrical space with the Euclidean distance, city-block metric (see Gärdenfors (2000) for a comprehensive explanation), or some other comparison technique like those used for clustering in data mining (Mencar *et al.*, 2007). Prerequisite for this similarity matching is to have values of properties that map to a scale to measure the distance between the entities. Gärdenfors' approach to similarity, like clustering algorithms, is useful for grouping instances, but is limited to being similar with respect to one or two values of properties only. If the property chosen for similarity matching is not a sortal property, then no classification occurs (cf. DL Systems and OWL-ontologies, *e.g.*, Wolstencroft *et al.* (2007)) but only an *ad hoc* grouping of the entities or instances. This may be useful for designing a tree to classify the data afterward, but the two operations are distinct; *e.g.*, clustering objects by being red and others purple does not reveal anything about the identity of each object grouped under Red and Purple. A few examples about similarity measures in biology are described in the following example.

Example 3.4. An extension of clustering with similarity is the notion of guilt-by-association (GBA) in mRNA expression analyses, where uncharacterised genes may share the same functional roles as already annotated genes in the same cluster. Zhou *et al.* (2005) describe techniques how to measure similarity adequately for GBA, which are based on trial-and-error and subsequent human analysis without using semantics of granularity. The authors developed an Ontology-based Pattern Identification to achieve meaningful clustering by combining quantitative measures for clustering with qualitative knowledge from the Gene Ontology. A similar approach was taken by Tsumoto (2007), who combined a diagnostic taxonomy about headaches with hospital clinical data for data mining. A similar approach may alleviate the randomly generated protein clusters from SwissProt and TrEMBL (Kaplan *et al.*, 2004, 2005; Shachar and Linial, 2004), but where each cluster ought to be such that it is a set extension of a universal.

Alternatively, if there is no measurable property, one has to invent one that bears a correlation with the property under investigation: access to property a of entity x by means of property b, where b is not necessarily a property of x. Relating quantitative granularity with the qualitative granularity then becomes a necessity. Bender and Glen (2004) discuss many such combined similarity measures for molecules and Rutishauser and Moline (2005) for parameters for traits and genes. $\diamondsuit$

Thus, testing for similarity means that there is *something to compare*. More precisely, we can define similarity in the scope of granularity as follows.

DEFINITION 3.9 (Similarity σ). *Let $x, y, ...PT$ or $x, y, ... \in U$, gl_i a granular level, $in\_level(x, gl_i)$ and $in\_level(y, gl_i)$, then gl_i contains $\{x, y, ...\}$ that are similar to each other, $x\ \sigma\ y$ at level gl_i, with respect to quality property, p_q, with measurable values that are within the defined value space ε valid for p_q of x. σ is reflexive, symmetric, and transitive.*

Observe that in a setting *without* granularity—omitting the assertion that gl_i contains x, y—one cannot assert symmetry and transitivity, because it is possible that $\varepsilon_y < \varepsilon_x$ such that the particular value of x is beyond ε_y. And if $x\ \sigma\ y$ holds when the value of y for p_q is within ε_x, then this does not imply that if $y\ \sigma\ z$ (because z's value for p_q is within ε_y) then z's value still falls within ε_x; transitivity can hold, but not necessarily for all entities. In contrast, in the granular setting (a) ε is set for each level, hence σ is symmetric and transitive with respect to the objects residing in the same level, and (b) if $\varepsilon_y < \varepsilon_x$ (according to the source data or level specification) so that the particular value of x is beyond ε_y then it *must* be that y resides in finer-grained level of granularity compared to the level where x resides. We return to points a and b after addressing indistinguishability and indiscernibility.

3.4.1.2 Indistinguishability and equivalence

Given similarity, it is easier to highlight the important distinction with indistinguishability, which concerns objects being indistinguishable at some level, hence that one *cannot compare* and determine if objects are similar or not. It assumes that at a lower level of granularity the entities are distinguishable from each other. The differences between similarity and indistinguishability are illustrated for coffee, cholera, and whooping cough in *Example 3.5* and analysed afterward on both qualitative and quantitative aspects of granulation.

Example 3.5. Take the meaning of the statement "caffeine and cholera toxin both cause prolonged activation of the Second Messenger System (SMS)", then the effects of both caffeine and the cholera toxin are *similar enough* to be put together at this level gp_igl_i, where finer-grained mechanisms of prolongation of SMS, if any, are *indistinguishable* from each other at gp_igl_i. This is different from claiming both mechanisms are 'exactly similar', the same, because based on the given information, one cannot prove either way. At a finer-grained level gp_igl_j, the mechanisms are distinguishable and found to be distinct, thus not similar enough to be grouped together. More precisely, caffeine inhibits phosphodiesterase activity that otherwise would break down cAMP, whereas the A subunit of the cholera toxin binds to the G protein, thereby impairing G to return to its inactive state and locking G into its active form, resulting in excessive production of cAMP. Thus, they are distinguishable with regards to the *properties* involved, hence looking for similarity *values* becomes irrelevant in this comparison.
The pertussis toxin, produced by the causative agent (bacterium) of whooping cough *Bordetella pertussis*, also interferes with a G protein in the SMS but a different one from cholera toxin. G proteins are signal-coupling proteins with *similar* structural and functional motifs, where the *s*timulatory G_s protein is the target of cholera toxin and the *i*nhibitory G_i protein is the target of the pertussis toxin. G_s and G_i have the same β

and γ subunits but have different α subunits; *i.e.*, the same properties are involved but they have distinct values, hence the α subunits of G_s and G_i can be compared on their similarity on 3D shape or aminoacid sequence. When both are categorised as whole G protein they are *indistinguishable* and it is only at the finer-grained level where one has 'access' to the properties of its structural components that one can distinguish them.[5] $\diamondsuit$

This example illustrates two key issues about indistinguishability: *being* (in-)distinguishable and *how* one moves from distinguishable to indistinguishable (and back). The former involves properties, the latter makes use of multiple types of relations to make things (in-)distinguishable, which needs to be disambiguated. The first step to clarify this, is to analyse the indistinguishability relation as introduced by Hobbs (1985) and followed up by Mani (1998); McCalla *et al.* (1992), among others: the indistinguishability relation "$\sim$" between x and y, slightly extended by McCalla *et al.* (1992), says that x and y are indistinguishable if no *relevant* predicate distinguishes between them, where p is some predicate and **R** a subset of predicates of a first order logical theory T_0 (3.4), or, in natural language: x and y are equivalent for predicate p.

$$x \sim y \leftrightarrow (p \in \mathbf{R})(p(x) \equiv p(y)) \tag{3.4}$$

Analysing (3.4) and Hobbs', Mani's and McCalla *et al.*'s description of the "indistinguishability" relation, it is immediately clear that the axiom is just another rendering of the *equivalence relation*:

DEFINITION 3.10 (Equivalence relation). *Let X be a set, and R a binary relation on X that is a subset of $X \times X$. The equivalence relation $\sim$ (alternative notation: $\equiv$) is a special case of R: an equivalence relation on X is a binary relation on X such that: $x \sim x$ for all $x \in X$ (reflexivity), if $x \sim y$ then $y \sim x$ for all $x, y \in X$ (symmetry), and if $x \sim y$ and $y \sim z$ then $x \sim z$ for all $x, y, z \in X$ (transitivity).*

With this general definition, one can define specific particular equivalence relations. For instance, to state that x and y are equivalent under a given condition, we have $x \sim y$ iff $x \equiv y (\text{mod } 2)$, meaning that x and y have the same value after y is divided by 2; or concerning (3.4), "(mod 2)" is replaced by an arbitrary predicate p. Bittner and Stell (2003) use the equivalence relation in the context of rough sets (rough sets are addressed in the next section), whereas Schmidtke (2005) limits the equivalence relation to equal durations of particular temporal intervals. Ideally, one should be able to extend this equivalence relation also to other subject domains, like that we can say that compatible transplant kidneys are equivalent at the *Organ*-level of granularity, even though its cells have different characteristics. This cannot be represented as such with just the equivalence relation, due to the lack of reference to a property and the absence of the qualifier 'given level g'. Further, if X is, say, a set of toys that has to be granulated and one defines $\sim$ as has the same colour as, then a subset Y of X for the colour blue has as members all blue toys; but the toys still can be identified. However, if one does not have access to properties at a finer-grained level to examine if the toys have, *e.g.*, at least one arm, then a $\sim$ for being a crane—as property p, permitted by Hobbs' description—leaves us in the dark if the objects in subset Y of X are fluffy toy-birds or toy lifting devices. On the other hand, with G_s, $G_i \in Y$, $G \in X$, then $G_s \sim G_i$ in X for properties function and α-subunit. In short, there are differences in *intended semantics and usage* between the equivalence relation and indistinguishability:

- ⋆ The equivalence relation has a 'conversion parameter' to make y equivalent to x with respect to that parameter and it focuses on partitioning sets; that is, making objects in X *distinguishable* in subsets of X;
- ⋆ Before, during, and after creating disjoint subsets with the equivalence relation, one still can distinguish—identify—the objects: identity or identification of the objects is already provided by membership of X.

[5]Part of the biological information for this example is based on Stryer (1988) pp973-985.

- $\star$ Hobbs' (and Mani (1998); McCalla *et al.* (1992)) indistinguishability (a) aims to be more generic to any kind of property (although this is not excluded with the equivalence relation), including *qualitative* properties and (b) focuses on the direction from similar and distinguishable in Y toward indistinguishable—that is, the *opposite* of granulation: a unifying property among objects to make them indistinguishable from each other in set X.
- $\star$ With indistinguishability, one cannot distinguish between the objects at their coarser-grained level: the objects' identity criterion lies with the characteristics of the finer-grained set Y, not with the unified set X at the coarser-grained level.

Of course, $\sim$ itself does not say anything about the direction of usage—partitioning or unifying—but the semantics of indistinguishability is more comprehensive. To substantiate this claim we take a closer look at Hobbs' (and others) *simplification function*, κ, and *articulation*[6] before defining the indistinguishability relation further below. The simplification function collapses a theory T_0 into a coarse-grained simpler theory T_1: $x \sim y \equiv \kappa(x) = \kappa(y)$; κ is only constrained by $\neg(\exists x, y)(x < y \wedge \kappa(x) > \kappa(y))$ (Hobbs, 1985). Thus, if we have two fine-grained entity types A and B (or their respective instances a and b) in $gp_i gl_j$, with two mappings $\kappa(\mathsf{A}) \rightarrow$ C and $\kappa(\mathsf{B}) \rightarrow$ C and C is an entity type in the coarse-grained level $gp_i gl_i$, such that $gp_i gl_j \prec gp_i gl_i$ (and not $gp_i gl_j \prec x \prec gp_i gl_i$), then A $\sim$ B at level $gp_i gl_i$. Using the G proteins of *Example 3.5*, one has A = G_s Protein, B = G_i Protein, and C = G Protein, or, as Hobbs and later also Mani (1998) put forward, a time interval in T_0 maps into an instant in coarser-grained T_1. The inverse procedure of simplification is to be able to distinguish between x and y, what Hobbs calls *articulation*, which has the aim to find (partial) predicates wherein x and y differ, which corresponds to granulation. For substance p that has an $\sim$ determined by the level of granularity of p, then "*a piece of p has proper parts which are of the same substance*, provided it has two distinguishable points" (emphasis added) (Hobbs, 1985):

$$\forall x(p(x) \wedge \exists y_1, y_2(in(y_1, x) \wedge in(y_2, x) \wedge y_1 \nsim y_2) \supset \exists z(part\_of(z, x) \wedge z \neq x \wedge p(z))) \tag{3.5}$$

Continuing now with the equivalence vs. indistinguishability, there are four issues regarding simplification and articulation with respect to granularity. First, "substance" is ontologically ambiguous. Is it (a type of) an amount of matter, as in the DOLCE foundational ontology (Masolo *et al.*, 2003), or any type of entity? If the former, then (3.5) is meaningless. For instance, if we take the molecules in wine to be proper parts of wine, then the molecules are not the same substance as wine, like tannin is not the same kind of stuff as wine is. If, on the other hand, we take a smaller portion of wine, then the smaller portion is made of the same substance as the full amount of wine, but there is never a distinguishable property between the two portions except a difference in the value of the quantities. Alternatively, only the two distinguishable parts are of the same substance.

Second, the formalisation and semantics of articulation is not the inverse of the simplification function. During simplification, one discards a property in the same way as with specialisation/generalisation, but with articulation there is 'something' z new introduced, that is *part of* an existing recognised property x. One goes up to a coarser-grained level by discarding a property and returning down by only identifying a part of a known property that does not necessarily reintroduce the discarded property, except if the property is a compound property and meets the requirement of compositionality. *In casu*, using aforementioned simplification from time interval to instant (Hobbs, 1985; Mani, 1998), then articulating from instant to interval introduces several issues: how is the interval part of the instant, and, more importantly, what are the two distinguishable parts of the same type of the interval? This only can be if the interval itself is split-up in parts to be able to distinguish between, say, the 3rd second and the 5th second in an interval of > 5 seconds, which is at a finer-grained level of granularity than the interval. It is different when we look at a sequence of events or processes occurring *during* the interval

[6]A comprehensive analysis of simplification and articulation—or, more precisely, *abstraction* and *expansion*—together with granular levels is deferred to §4.3.2 and §4.3.3, respectively.

instead of looking at the interval itself, because then we have a straight-forward case of whole process and its process-parts, but this does not imply that the process-parts are of the same type as Hobbs' defined in (3.5).

Third, the generic notion of being indistinguishable represents a coarse-grained meaning of *what* is indistinguishable: it lumps together being indistinguishable caused by subtyping, structural parts, sub-processes, spatial parts and so forth. Further, by considering only an *arbitrary* property of some object, $p(x)$, one can randomly discard and add properties, but ignoring something at the same level of granularity because it has nothing to do with the subject domain is distinct from granularity itself. Likewise, partitioning with the equivalence relation does not imply one generates different levels of granularity. In contrast stands the use of a *non*-arbitrary, *essential*, identity criterion-providing property to make something indistinguishable: having lost the identity criterion at the coarser-grained level has made those objects as they are in the finer-grained level *non-identifiable* as such in the coarser-grained level. Analogously, partitioning with the equivalence relation according to an *essential* identity criterion-providing property generates not just new subsets, but, unlike the Blue toys in the example above, those subsets are the *set extensions of universals*.

The fourth problem follows from the second and third: the underspecified indistinguishability relation (as well as the equivalence relation) and the two functions do not characterise a granular level, let alone granularity, except for highlighting it has something to do with properties. The issue at hand is *what* about those properties and *how* are they involved in specification of a granular level? This will be addressed in the next section.

Summarising the *qualitative* aspects of indistinguishability, we have an indistinghuishable-ising property, an implicit distinction between using either properties or their values for granulation, involvement of coarse- and finer-grained granules, if the resulting fine-/coarse-grained sub-/super-set should be a set extension of a universal. This still lacks the integration of *quantitative* aspects for indistinguishability. Concerning indistinguishability for **sG**, we have to address in more detail a variant of indistinguishability, better known as the indiscernibility relation, which is discussed next.

Indiscernibility. An aspect of indistinguishability is indiscernibility: the property under consideration is the same, but some measurable, numerical value, of the property is indistinguishable at a coarser-grained level. The simplest approach is Hobbs' (1985) similarity measure, or "measure of undefinedness", ε, where f is a restricted case of κ for indistinguishability for measurement values: $x \sim y \equiv \kappa(x) = |f(x) - f(y)| < \varepsilon$. For instance, real numbers are rounded off to its nearest integer, such that two reals, say, 11.1 and 11.2, at level $gp_i gl_j$ are indistinguishable at its higher level of integers, $gp_i gl_i$, where both reals map into the integer 11. Independently, this idea has been developed to a much larger, and usable, scope in the research areas of rough sets, its rough mereological approach, approximation spaces of similarity (Peters *et al.*, 2002; Skowron and Peters, 2003; Yao, 2004a), and fuzzy sets (Reformat *et al.*, 2004; Zadeh, 1997) with "fuzzified equality" (Klawonn and Kruse, 2004). For all these variations, some instance is part of a set to a degree, depending on how set inclusion is defined in the domain of interest and its software application. For the TOG, this notion corresponds to scale-dependent granularity in that a coarser-grained scale has 'rougher' similarity with a larger approximation space between the instances of a set compared to finer-grained sets with instances that are more similar to each other.[7]

[7]An extension to rough sets is measurement taking and allocation of instances to sets according to fuzzy rules, which are relevant for implementations, but rough and fuzzy sets do not have ontological implications for the definition of a granular level in a domain- and implementation independent theory of granularity. This is distinct for *vagueness* where things have fiat boundaries, are considered to be *intrinsically uncertain*, and are inquiry resistant (Sorensen, 2003). One may contend if vagueness in ontological in origin or due to our linguistic limitations (Prinz, 1998), and if it is due to limitation of measurement equipment. Either way, vagueness is not an issue for granularity, because there is always a property to distinguish by at its appropriate level of granularity, *i.e.*, granularity *avoids*

Going into some detail for the approximation spaces and rough mereological approach (Peters *et al.*, 2002), let Ind be the indiscernibility relation, U a set of objects, $P(U)$ its power set, $\mathcal{N}$ an uncertainty function defined on U where $\mathcal{N}(x)$ is a neighbourhood of object x, and ν the inclusion function defined on $P(U) \times P(U)$ that measures the degree of inclusion of sets, then we can define the two operations for lower (AS_l) and upper (AS_u) approximation: $AS_l(X) = \{x \in U : \nu(\mathcal{N}(x), X) = 1\}$ and $AS_u(X) = \{x \in U : \nu(\mathcal{N}(x), X) > 0\}$. $\mathcal{N}(x)$ can be defined by the indiscernibility relation Ind in two ways, either as equivalence relation or as tolerance (or similarity) relation τ, where $\tau \subseteq U \times U$ (in Peters *et al.* (2002)'s terminology). If the former, then $\mathcal{N}(x) = [x]_{Ind}$ where the equivalence class $[x]$ comprises x and its neighbourhood, if the latter, we have $\mathcal{N}(x) = \{y \in U : x\tau y\}$ and thereby making $\mathcal{N}$ equal to the tolerance class τ defined by x. Thus, τ is conceptually equivalent to the $\sim$ in the previous section. y is near enough to x to be grouped together because they are in the same approximation space AS and both objects are included thanks to the inclusion function ν; AS is then defined by $AS = (U, \mathcal{N}, \nu)$ (Peters *et al.*, 2002). Approximation operations, thus also classifying the object into a particular level of granularity, can be adjusted via parameterisation of $\mathcal{N}$ and ν. Note that ν can be generalised to the rough mereological approach with the inclusion relation μ_r such that $x\mu_r y$ denotes that x is part of y to a degree r. Thus, set membership can be determined if the value of a property falls within the defined lower and upper bound for that set. Klawonn and Kruse (2004) use μ to denote fuzzification of set membership "$\in$", represented as an element x that has a membership degree $\mu(x)$ to a fuzzy set μ with the usual further specifications for degrees of membership in fuzzy sets, like taking into account ε for closeness to a point (as Hobbs did as well), and minimum & maximum values. This lower-upper bound usage matches the sieve conceptualisation of **sgG**.

These specifications, ignorant about levels of granularity (or granules), permit two orthogonally positioned usages. One concerns the value or value range for each level within a perspective, such as km^2 and m^2, and the other the precision of each specific level, or the *refinement* of measurements at that level, such as "km^2 $\pm$ 1 m^2" or "km^2 $\pm$ 1 cm^2", where the choice for m^2 or cm^2 is provided by the approximation space that covers both intended and enforced indistinguishability[8]. Examples of degrees of membership are properties such as colour shades, but this is rarely used as a criterion for granulation of universals (although it can be used as an indirect means (Bender and Glen, 2004; Zhou *et al.*, 2005)). It is important to stress the difference between the complementary granularities & indiscernibility: the primary 'axis' involves the scale relevant for the granular perspective and the secondary axis the refinement, amount of impreciseness, that is given in any of the finer-grained quantities at each level. Although the myriad of mathematical representations of indiscernibility (for a collection, see Mencar *et al.* (2007) and references therein), and, tacitly, scale-dependent granular levels, differs from the one formalised in this chapter, the underlying semantics of what it conveys fits within the TOG, hence a mapping of these approaches into the TOG is possible.

3.4.1.3 Indistinguishability for granulation and granular levels

Having addressed both the qualitative and quantitative aspects of indistinguishability, we can proceed to the definition of indistinguishability, where U stands for universal and PT particular,

vagueness. That at a certain level of granularity two objects are indistinguishable or sufficiently similar does not imply vagueness, but 'vagueness' with respect to a model and interpretation. It may be that at a finer level of granularity this 'vagueness' can be resolved because there we do have the necessary tools and methodologies to distinguish the hitherto indistinguishable entities. Consequently, this allows us to define distinctions ourselves. For instance, one can define a cloud's boundary there where the density of droplets in the air is higher than some value x, thereby doing away with fiat boundaries and vagueness.

[8]Klawonn and Kruse (2004) add distinctions for "enforced" and "intended" indistinguishability, which affects representation in software systems, but not the domain- and implementation-independent TOG. The former is caused by limited precision due to noisy data, the equipments itself and indirect measurement taking, the latter corresponds to impreciseness because the measurement-taker does not care about more precise measurements.

Q quality, and V region are DOLCE categories:

DEFINITION 3.11 (Indistinguishability $\eqcirc$). *Let x and y reside in level gl_j, there exists a level gl_i s.t. $gl_j \prec gl_i$ contained in one granular perspective, z resides in gl_i, x and y map into z, and $z \in PT$ or $z \in U$, then x and y are φ-indistinguishable from each other, $x \eqcirc y$, at level gl_i with respect to z. φ denotes the type of relation that relates x, y to z, its distinguishing property, property value, or value range between the two levels,* i.e., *$\varphi \in \mathcal{P}$ or $\varphi \in V$, where $\mathcal{P}$ is the set of binary predicates P on the domain ($PT \times PT$ or $U \times U$), $\eqcirc \neq \varphi$, and V is the set of declared values and value ranges for property Q.*

Observe the difference with the equivalence relation by framing indistinguishability in the scope of granularity and inclusion of a reason for indistinguishability. With the φ, one can define specific versions for a particular indistinguishability relation relevant for the levels in a granular perspective. As will be motivated in §3.6, these relations are restricted to $ppart\_of$, $contained\_in$, $involved\_in$, is_a, $participates\_in$, and $member\_of$. For instance, for qualitative granularity, the φ of φ-indistinguishable with respect to set-based granularity then refers to exactly that additional property of the subsumed entity type compared to its subsumer, whereas with whole-part granulation the φ refers to the very fact that the finer-grained entity type is part-indistinguishable. On the other hand, modifying the definition of the equivalence relation by adding reference to levels is another option. Yao (2004a) takes steps in that direction, but he permits partitioning (granulation) into arbitrary sets or named subsets (refer to §5.2.2 for a discussion). Although a name can be a mere label, it is a step toward *meaningful* subsets as opposed to an arbitrary grouping of objects. To have subject domain semantics for indistinguishability at the *ontological* level, the sets resulting from granulation or unifying should be the set-extensions of universals, which can be met with $\varphi \in \mathcal{P}$. However, the equivalence relation has a widespread use beyond granular computing and modifying the definition may result in confusion, whereas the notion of indistinguishability remains mainly within the context of granulation and therefore makes it easier to assign semantics to the relation.

With the three definitions of equivalence, similarity, and indistinguishability, we can derive several interesting properties regarding a theory of granularity, which are demonstrated with the following four lemmas and theorem. The main outcome (*Theorem 3.2*) is that *there must be at least two granular levels in a granular perspective*. This contradicts Salthe (1985, 2001), who asserts that there must be at least three levels to always have one level above *and* one level below the level of interest. However, there is neither necessarily an infinite chain of granulation steps (see Chapter 2) nor is this inherently demanded from any of the well-established notions of indistinguishability, partitioning, granulation, and equivalence: the only requirement these relations imply is to have one level above *or* one level below the level of interest. It follows from *Theorem 3.2* that there is a strict total order among the granules in any particular hierarchy (*Lemma 3.9*); therefore, levels in such a hierarchy must be disjoint (*Corollary 3.3*). To arrive at these results, we first take the definitions of equivalence, similarity, and indistinguishability, and prove exclusion of co-existing similarity and indistinguishability (*Lemma 3.6*), then that indistinguishability is a subtype of the equivalence relation (*Lemma 3.7*) with its specific properties (*Lemma 3.8*).

LEMMA 3.6. *If x and y are similar, $x\sigma y$ in a level, then they cannot be indistinguishable, $\neg(x \eqcirc y)$.*

Proof. Given *Definition 3.9*, x, y, and $x\sigma y$, then at least their value spaces ε_x and ε_y must either overlap or overlap properly to ensure y falls within ε_x.
First, using overlap from mereology, then (where the usual x, y is substituted with their respective εs)

$overlap(\varepsilon_x, \varepsilon_y) \triangleq \exists z(part\_of(z, \varepsilon_x) \wedge part\_of(z, \varepsilon_y))$

Assume $x \eqcirc y$, which implies $\varepsilon_x \eqcirc \varepsilon_y$. Substituting ε_y for ε_x (or vv.), collapses into identity ($\varepsilon_x = \varepsilon_y$) for p_q and thereby contradicting $overlap$. Take ε_x for x, then x must have been identified prior to determine ε_x, because p_q is a property of x. Given x, one takes another object y to

measure p_q, which is known to be $\neg x$, thereby contradicting $x \Bumpeq y$. (Note: if the measured values are the same, then x and y are 100% similar, because x and y were already identified, hence not $(x \Bumpeq y)$.)
Second, consider proper overlap and its overcross, which are defined as
$p\_overlap(x, y) \triangleq overcoss(x, y) \wedge \neg(overcoss(y, x))$
$overcross(x, y) \triangleq overlap(x, y) \wedge \neg part\_of(x, y)$
Substitute x, y for their respective εs, substitute $overlap$ in $overcross$, and then $overcross$ in $p\_overlap$, which gives
$(\exists z(part\_of(z, \varepsilon_x) \wedge part\_of(z, \varepsilon_y) \wedge \neg part\_of(\varepsilon_x, \varepsilon_y))) \wedge$
$\neg(\exists z(part\_of(z, \varepsilon_y) \wedge part\_of(z, \varepsilon_x) \wedge \neg part\_of(\varepsilon_y, \varepsilon_x)))$
De Morgan in the second part of the $p\_overlap$ definition:
$(\exists z(part\_of(z, \varepsilon_x) \wedge part\_of(z, \varepsilon_y) \wedge \neg part\_of(\varepsilon_x, \varepsilon_y))) \wedge$
$(\neg\exists z(part\_of(z, \varepsilon_y) \wedge part\_of(z, \varepsilon_x)) \vee \neg(\neg part\_of(\varepsilon_y, \varepsilon_x)))$
Either the first quantification "$\exists z(\ldots)$" or the second one with the negation has to be false. The latter can be false but then $\neg(\neg part\_of(\varepsilon_y, \varepsilon_x))$ must be true (thanks to the $\vee$). With elimination of the double negation as well, then
$(\exists z(part\_of(z, \varepsilon_x) \wedge part\_of(z, \varepsilon_y) \wedge \neg part\_of(\varepsilon_x, \varepsilon_y))) \wedge part\_of(\varepsilon_y, \varepsilon_x)$
holds. The combination of $\neg part\_of(\varepsilon_x, \varepsilon_y)$ and $part\_of(\varepsilon_y, \varepsilon_x)$ is *proper* parthood (Varzi, 2004a), hence, $p\_part\_of(\varepsilon_y, \varepsilon_x)$, which implies $\varepsilon_x \neq \varepsilon_y$, and in turn $\neg(\varepsilon_x \Bumpeq \varepsilon_y)$. Given that the respective spaces of x and y can be distinctly identified, x and y must be distinguishable if $x\sigma y$, hence $\neg(x \Bumpeq y)$. □

LEMMA 3.7. *If x and y are indistinguishable, $x \Bumpeq y$, then they are also equivalent, $x \sim y$.*

Proof. Given *Definition 3.10*, we have $\forall x, y(x \sim y \rightarrow R(x, y))$, where $R \subseteq S \times S$, and $x \sim y$ with $x, y \in S$, for any *arbitrary* condition $c \in C$. Given *Definition 3.11*, we have $\forall x, y(x \Bumpeq y \rightarrow R(x, y))$, where R is a subset of the contents S of a granular level, $S \times S$, but $x \Bumpeq y$ (with $x, y \in S$) is *restricted* to φ type of conditions, where $\varphi \in \mathcal{P}$ and $\mathcal{P} \subset C$; hence, $\forall x, y(x \Bumpeq y \rightarrow x \sim y)$. □

LEMMA 3.8. *The indistinguishability relation $\Bumpeq$ has the meta-properties of reflexivity, symmetry and transitivity.*

Proof. From *Lemma 3.7* we have $\forall x, y(x \Bumpeq y \rightarrow x \sim y)$, therefore $\Bumpeq$ inherits the properties of $\sim$. Given *Definition 3.10*, we know that $\sim$ has the meta-properties of being reflexive, symmetric, and transitive, therefore $\Bumpeq$ is also (at least) reflexive, symmetric, and transitive. □

THEOREM 3.2. *A granular perspective GP must contain at least two granular levels GL, formally:*
$\forall x(GP(x) \rightarrow \exists^{\geq 2} y(RE^{-}(x, y) \wedge GL(y)))$

Proof. Given *Definition 3.10* and *Definition 3.11*, let $x, y, ... \in X$, R defined over $X \times X$, $\mathcal{R}$ the set of relations, $\forall x, y(x \sim y \rightarrow R(x, y))$ and $\forall x, y(x \Bumpeq y \rightarrow R(x, y))$.

1. Assume one level: By *Lemma 3.6*, we cannot have one set (granule, level) where both $x\sigma y$ and $x \Bumpeq y$ hold, therefore granulation results in > 1 level.
2. Assume at least one level: suppose $\forall x, y(x \Bumpeq y \leftrightarrow R(x, y))$, i.e. $\{\Bumpeq\} \equiv \mathcal{R}$, then all objects in contained in level gl_i, X (*i.e.*, $in\_level(X, gl_i)$), are indistinguishable and $\{x, y\} \equiv X$. But by *Definition 3.11*, we have $\Bumpeq \neq \varphi$ and $\varphi \in \mathcal{R}$, and by *Lemma 3.6* we have σ, therefore at least $\Bumpeq, \varphi, \sigma \in \mathcal{R}$, therefore there must exist a subset Y s.t. $x', x'' \in Y$, $\neg(x \Bumpeq x')$ and $\neg(x \Bumpeq x'')$ and $x'\sigma x''$ because x' and x'' are made indistinguishable into x with φ (thus also either $x' \neq y$ or $x'' \neq y$ or both), hence $Y \subset X$ for its granules, where $in\_level(Y, gl_j)$, where $i \neq j$, *i.e.*, ≥ 2 levels.
3. Assume at least three levels: with ... $\prec gl_j \prec gl_i \prec gl_h \prec$ ..., then with $in\_level(x, gl_i,$ x must be related to at least two entities, y in gl_j and z in gl_h, which requires a ternary relation $R(x, y, z)$ to span 3 levels. σ, $\sim$, and $\Bumpeq$ are binary relations and, by point 2 above, permit granulation, thereby contradicting a minimum of 3 levels. □

LEMMA 3.9. *There is a strict total order, "$\prec$", between finer- and coarser-grained levels in a perspective.*

Proof. It follows from *Lemma 3.6* ($\sigma \not\Leftrightarrow$ and proper parthood) and *Theorem 3.2* (point 2, $Y \subset X$) that we have not only a strict weak order (if x is incomparable with y, y incomparable with z, then x is incomparable with z, which is ensured with reference to GL) but also trichotomy (one of $x \prec y$, $y \prec x$ or $x = y$ is true), respectively. $\square$

COROLLARY 3.3. *Within one granular perspective, no two levels are the same.*

With these outcomes, two clarifications have to be made regarding **samG** and set-based granularity in the light of key requirement 10 that "the same" entity (/type) cannot reside in more than one level. For the latter, we avail of the *most specific universal*, which is defined as

DEFINITION 3.12 (most specific universal)**.** *Let A denote a universal, ϕ and ψ variables ranging over universals except A, $\neg(\phi = \psi)$, and a denotes an instance such that $inst(a, A)$ holds, then A is the* most specific universal *of a, $minst(a, A)$, iff $\neg\exists\phi(inst(a, \phi) \wedge (\phi \rightarrow A))$ and $\forall\psi(inst(a, \psi) \rightarrow (A \rightarrow \psi))$.*

Observe that the definition entails $minst \rightarrow inst$ and $\neg(A \equiv \top)$. We can be more precise now with key requirement 10 concerning taxonomic subsumption and granularity: "*An entity a or A where $minst(a, A)$ holds never can reside in more than one granular level within the same perspective*". When a or A is not in the coarsest level, then $inst(a, \phi) \wedge (A \rightarrow \phi)$ holds or, in the case the data source is an ontology, $(A \rightarrow \phi)$ holds (the latter means that the amount of properties of A is at least 1 more than that of ϕ, as per usual for a well-defined taxonomy). Further, **samG** takes the same collection of entities of the same Urelement to populate the different levels, which would suggest $\preceq$ instead of $\prec$. However, one has to appreciate the difference between *intension* and *extension* of a universal (see also *Proposition 2.1*). For instance, an hour consisting of 60 minutes and minute being a proper part of hour are different from a set of arbitrary 60 minutes. Therefore **samG** does not contradict $\prec$.

Summarising, indistinguishability and similarity were disambiguated and compared with the well-known equivalence relation. For objects being *similar*, one has some measurable property to compare the objects, whereas objects being *indistinguishable* cannot be compared because they cannot be distinguished for the focal property and *indiscernible* objects are indistinguishable for some measurable property, *i.e.* the obtained values of the objects are the same. Indistinguishability starts from distinguishable objects at a finer-grained level and is generally used to move 'up' toward being indistinguishable, whereas the equivalence relation starts from some set and generally is used to partition objects into finer-grained sets making them distinguishable. Indistinguishability influences the conceptualization of granular level concerning how properties are introduced and removed going from a coarse to a finer-grained level and vice versa, which implies that lower levels have more, or more precisely specified properties compared to levels higher up in the granularity hierarchy. It also affects the process of allocating objects to a certain level of granularity. Indiscernibility and related similarity are particularly relevant for the implementation layer to deal with measurable values of properties and functions and types of queries to perform on the data sources. These outcomes, however, do not address fully granularity relative to a model and measurement method or device, and its (perceived) ontological consequences concerning emergent properties. Such philosophical considerations are addressed in (Keet, 2007e) because after investigation, it appeared that this notion does not affect the definition of granular level, but actually benefits from granularity.

3.4.2 Defining scale- and non-scale-dependent levels

Recollecting the introduction of this paragraph, values of the criterion of GP delimit which entities (/types) reside in a $gp_i gl_i$. For any level that adheres to **sG** type of granularity, the value, or value range is determined by the type of scale used. Regarding **nG**, the amount of properties considered at a finer-grained level increases with **nrG** and **nasG** (with respect to

is_a) and properties are added in mereology-based granular levels (**nrG**) where properties of the finer-grained entity (/type) do not refer to the properties of the whole. Thus, any criterion C does not provide a single obvious property with changing values for non-scale-dependent levels across the hierarchy. Even with a straightforward perspective of human structural anatomy, this still does not have an obvious distinctive value to distinguish between, *e.g.*, gp_2gl_i = Organ and gp_2gl_j = Cell other than the name of the level, if it exists, such as Organ, Tissue, Cell etc. An issue is that for some levels we perceive, there is no readily available term. Introducing a numbering scheme can be sufficient for an implementation in order to administer a hierarchy—but note that identification and identity are separate issues. I use the *label*—which may be just a number—of each level as value of one of the properties in the criterion, which is used in the sense of *looking through* (Johansson, 2005) and does *not* mean the label is all there is and this leaves open the option to adorn GL with other properties (*e.g.*, to take the intension of the root type of a taxonomy, say, Organ, for adorning a level that adheres to **nrG**).

What may have become clear with this assessment, is that, like GP, GL also refers to TG and uses GP's C, hence, granular levels cannot exist in a domain granularity framework on their own, but must be contained in a granular perspective. To this end, $adheres\_to$ and *Proposition 3.9* are introduced.

DEFINITION 3.13 (adheres_to). *The relation* adheres_to(x, ϕ) *holds if* $GL(x)$ *and* $TG(\phi)$*, that is:* $\forall x, \phi(adheres\_to(x, \phi) \rightarrow GL(x) \land TG(\phi))$.

PROPOSITION 3.9. *For all* x*, where* $GL(x)$*,* x *is contained in a granular perspective:* $\forall x(GL(x) \rightarrow \exists y(RE(x, y) \land GP(y)))$.

Likewise that GP must have a TG, GL has to have one, but this time specifically to constrain the structure of the contents of that level.

PROPOSITION 3.10. *Each* GL *must adhere to a* TG*:* $\forall x(GL(x) \rightarrow \exists\phi\ adheres\_to(x, \phi))$.

It does not suffice to only attach a 'how' data is granulated as with GP, but also to replace the variables in the characterisation of the contents structure with the actual data to enforce consistency in the implementations. Therefore, a mandatory constraint is added to the $adheres\_to$ relation. Now we have sufficient ingredients to define granular level. As before, several definitions from DOLCE are used, being concept CN, definition DF, quality Q, and region V. In addition, $has\_value(x, y)$ (*Definition 3.5*) and $RE(x, y)$ (*Definition 3.21*) are reused.

DEFINITION 3.14 (Granular level). $\forall x\exists!v, w, y, z\exists p$ *such that* $GL(x)$ *is a concept* $CN(x)$*, has a definition* $DF(x, y)$*, is related to* $GP(w)$ *with* $RE(x, w)$ *and uses criterion* $C(z)$ *with* $RC(w, z)$ *and* $has\_value(z, v)$ *where the value is in region* $V(v)$ *for any* $GL(x)$ *that* $adheres\_to$ **sG**, $GL^s(x)$*, and* z*'s label for any* $GL(x)$ *that* $adheres\_to$ *type* **nG**, $GL^n(x)$*. Entities residing in* $GL^s(x)$ *are similar to each other with respect to (the value* z *of)* $V(v)$*, entities residing in* $GL^n(x)$ *are similar to each other with respect to (the label of the universal of)* $Prop(p)$ *of* $C(z)$*, and both are* φ*-indistinguishable with respect to its adjacent coarser-grained level.*

Given this definition of granular level and the above-defined and proven characteristics, we can prove several additional properties. The "role subset" and "role equality" constraints shown in the overview diagram in *Figure 3.1* will be proven first (*Lemma 3.10* and *3.11*). From this, it can be proven that GL is contained in exactly one GP (*Theorem 3.3*).

LEMMA 3.10. *For each* $GP(x)$ *and* $GL(y)$ *over their join paths, the following holds: if* $GP(x)$ *contains* $GL(y)$*, then* $GP(x)$ *has granulation some* TG *and* $GL(y)$ *adheres to some* TG*:*

$$\forall x, y(RE(x, y) \land GP(y) \land GL(x) \rightarrow \exists\phi(has\_granulation(y, \phi) \land adheres\_to(x, \phi))) \quad (3.6)$$

Proof. First, given
$\forall x(GL(x) \rightarrow \exists y(RE(x,y) \wedge GP(y)))$ (*Proposition 3.9*)
$\forall x(GP(x) \rightarrow \exists^{\geq 2} y(RE^{-}(x,y) \wedge GL(y)))$ (*Theorem 3.2*)
therefore, if we have a GP, then there must be ≥ 2 instances of GL related to it and if we have a GL that there must be a GP. Assume a, b such that $GP(a)$ and $GL(b)$, then with
$\forall y(GP(y) \rightarrow \exists! \phi\ has\_granulation(y, \phi))$ (*Lemma 3.3*)
$\forall x(GL(x) \rightarrow \exists \phi\ adheres\_to(x, \phi'))$ (from *Proposition 3.10*)
either $\phi = \phi'$ or $\phi \neq \phi'$ so that there must be ≥ 1 TG and therefore (3.6) holds. □

Lemma 3.10 does not ensure GP and its GL use the *same* TG because the "$\exists \phi$" says there is *at least one* of them. To achieve this in *Corollary 3.4*, we first prove the following lemma.

LEMMA 3.11. *For each TG, some $GL(x)$ adheres to that TG if and only if some $GP(y)$ has_granulation that TG: $\forall \phi(\exists y\ has\_granulation(y, \phi) \leftrightarrow \exists z\ adheres\_to(z, \phi))$.*

Proof. Assume GP and GL are (mutually dependent) instantiated so that they must have a TG (*Lemma 3.10*). Given *Lemma 3.3* and §2.3 where each structure of level contents of the leaf types are distinct, then also
$\forall x(GL(x) \rightarrow \exists! \phi\ adheres\_to(x, \phi'))$
must hold, because combining two or more types leads to a contradiction. Further, from *Definition 3.14* we have "uses criterion $C(z)$..." and by
$\forall x(GP(x) \rightarrow \exists! y\ RC(x,y))$ (*Proposition 3.8*)
RE relating GL to its GP, having
$\forall x(GP(x) \rightarrow \exists! y, \phi(RC(x,y) \wedge has\_granulation(x, \phi)))$ (*Theorem 3.1*)
and aforementioned *Lemma 3.3*, therefore, the GL uses the same criterion as its GP, hence $\phi = \phi'$ holds, too. □

The combination of *Lemma 3.10* and *Lemma 3.11* can be formulated in a shorter constraint:

COROLLARY 3.4. *For all $GL(x)$ contained in its $GP(y)$, they have the same $TG(\phi)$ as its $GP(y)$, and vv: $\forall x, y(GP(y) \wedge GL(x) \wedge RE(x,y) \rightarrow \exists! \phi(has\_granulation(y, \phi) \leftrightarrow adheres\_to(x, \phi)))$.*

Alternatively, we can write the right-hand side of the implication
$\exists \phi_1, \phi_2((has\_granulation(y, \phi_1) \leftrightarrow adheres\_to(x, \phi_2)) \rightarrow \phi_1 = \phi_2)$
(see also "conventions" in §3.8). Note that the first implication in *Corollary 3.4* is not bidirectional, because the level may be contained in another perspective. This is possible due to the fact that a type of granularity can be reused for different perspectives, provided it has a different criterion (*Theorem 3.1*).

With these results, we can strengthen *Proposition 3.9* and prove that a level is contained in exactly one perspective:

THEOREM 3.3. *For all x, where $GL(x)$, x is contained in* exactly one *granular perspective.*

Proof. We already have at-least-one GL in GP (*Proposition 3.9*) and need to demonstrate the at-most-one (general version with R an arbitrary relation, then $R(x,y) \wedge R(x,z) \rightarrow y = z$). GL uses the C of GP it is contained in (*Definition 3.14*), which still permits a GL to be reused in another GP. However, GL adheres to the same type of granularity as its GP it is contained in (*Corollary 3.4*)[9]. Given
$\forall x_1, ..., x_4, y_1, y_2, \phi_3, \phi_4(RC(x_1, y_1) \wedge RC(x_2, y_2) \wedge has\_granulation(x_3, \phi_3) \wedge$
$has\_granulation(x_4, \phi_4) \wedge y_1 = y_2 \wedge \phi_3 = \phi_4 \rightarrow x_1 = x_2 = x_3 = x_4)$ (*Theorem 3.1*)
$\forall x_1, ..., x_n, y(GP(x) \wedge D^f(y) \wedge RE(x,y) \rightarrow \neg(x_1 = x_2) \wedge ... \wedge \neg(x_{n-1} = x_n))$ (*Corollary 3.2*)
there cannot be another GP with the same C and TG in one D^f, hence, GL can be at-most-one time in a perspective.Thus, ≥ 1 and ≤ 1 is exactly one, *i.e.*, $\forall x(GL(x) \rightarrow \exists! y RE(x,y))$ □

[9]Note that although the combination of C and TG are necessary and sufficient conditions for GP, they are only necessary for GL.

Now we can proceed to another useful property of granular levels, which revolves around the **sG** type of granularity that levels can adhere to. More precisely, the next two lemmas show that the values of the level's criterion is more encompassing that that of its adjacent finer-grained level and, by having measurements, we can add ≤ 2 mathematical functions to a granular level that takes care of the conversions between these values.

LEMMA 3.12. *The criterion's value $V(x)$ for coarser-grained level adhering to **sG** is larger (more encompassing) than that of its finer grained level in the same perspective.*

Proof. Let $GL(x_i)$, $GL(x_j)$ be in the same $GP(y)$ and $x_j \prec x_i$. Recollect *Definition 3.5* for has_value and upward distributivity (*Proposition 3.5, 3.6*, page 60). Further, with

$\forall x \exists z, \phi((C(x) \rightarrow \exists! y(CP(x,y) \wedge Q(y))) \wedge RC(z,x) \wedge has\_granulation(z,\phi) \rightarrow$
$(\phi \rightarrow sG))$ (*Lemma 3.4*)
$\phi \rightarrow sG$ (constraint in this lemma)

and *Corollary 3.4*, we also must have $adheres\_to(x_i, sG)$ and $adheres\_to(x_j, sG)$ and, thus by *Definition 3.14*, we can use $V(v)$. From *Lemma 3.6* we know $\varepsilon_{x_j} < \varepsilon_{x_i}$ and from *Definition 3.9* that $\forall x(\varepsilon_x \rightarrow V(x))$, hence $V(v_{x_j}) < V(v_{x_i})$ holds for such levels. □

With those values for levels, we can relate a function to a granular level that adheres to **sG** type, which can be used for 'converting' contents of one level into its adjacent coarser level, or vice versa; *e.g.* 60 * 1 minute = 1 hour. Let ϑ denote this mathematical function, and ϑ^- its inverse, where the variable ϑ ranges over functions. We can treat ϑ as syntactic sugar for a finite list of first order axioms and remain within FOL, as proposed by the Common Logic workgroup (see footnote 2, p52 and §3.8.1). $Conv(x,y,\vartheta)$ is defined as in *Definition 3.15* with F the set of functions.

DEFINITION 3.15 (Conversion). *The conversion relation, $conv(x,\phi,\vartheta)$, relates a granular level $GL(x)$ that adheres to a (subtype of) **sG** type of granularity ϕ to function ϑ:*
$\forall x, \phi, \vartheta(conv(x,\phi,\vartheta) \rightarrow GL(x) \wedge (\phi \rightarrow sG) \wedge F(\vartheta))$.

$Conv$ can be used at most twice for each level, which is proved in *Theorem 3.4*; the interested reader may want to consult *Appendix B.3* to read the Prover9-*computed* proof.

THEOREM 3.4. *For each $GL(x)$, there are ≤ 2 functions ϑ.*

Proof. We prove this in two steps.

1. Prove none: the relata for *conv* are constrained to **sG** types (*Definition 3.15*) but, **nG** types can be used for GP too (*Corollary 3.1*), which are disjoint from **sG** (Chapter 2). GL adheres to the TG of its GP (*Corollary 3.4*), so GL can use **nG** too, *i.e.*, $adheres\_to(x,\psi) \wedge (\psi \rightarrow nG)$ can hold. Thus, we can have levels that do not satisfy $conv$ and thus have no conversion function.
2. >2 functions leads to a contradiction: This follows from *Theorem 3.6* (1:1 on RL relating levels), because in a single-line hierarchy a level is related to ≤ 2 levels, being the adjacent lower level and/or the adjacent higher level. □

From this proof we trivially get the mandatory constraint on **sG** (*Corollary 3.5*) and that levels adhering to **nG** types do not have an associated function (*Corollary 3.6*).

COROLLARY 3.5. *If $GL(x)$ adheres to **sG** type of granularity, then it does have a relation to a function ϑ related to it through the conv relation, that is: $\forall x, \phi(adheres\_to(x,\phi) \wedge (\phi \rightarrow sG) \rightarrow \exists \vartheta\ conv(x,\phi,\vartheta))$.*

COROLLARY 3.6. *If some $GL(x)$ adheres to **nG** type of granularity, then that $GL(x)$ does not have a relation to a function ϑ.*

From an engineering point of view, a maximum of two functions for each level may seem prohibiting, but that is, theoretically, all one requires for traversing **sG**-granulated levels. Any other granularity conversion function to, say, skip a level for aggregating data, are extras to, *e.g.*, improve database performance, but this is outside the theoretical need.

Summarising the constraints on a granular level, we have

M. *Lemma 3.9*: There is a strict total order of the levels within a granular perspective (T.1, T.2).

N. *Corollary 3.3*, *Proposition 3.9*, *Theorem 3.3*: Within one granular perspective, no two levels are the same and each granular level must be contained in exactly one perspective (T.8).

O. *Theorem 3.2*: Each granular level in a granular perspective has at least one distinct adjacent granular level, *i.e.*, ≥ 2 levels in a perspective (T.7).

P. *Definition 3.9, 3.10, 3.11*, *Lemma 3.6, 3.7, 3.8*: similarity, equivalence, indistinguishability properties (A.52-A.57, A.58-A.62).

Q. *Definition 3.14*, *Lemma 3.12*: The criterion's value $V(x)$ of coarser-grained level is larger than that of its finer grained level (D.4, A.96, A.108, A.125).

R. *Lemma 3.10*, *3.11*, *Corollary 3.4*: all levels contained in one perspective adhere to the same type of granularity as its perspective (A.84, A.85, A.86).

S. *Definition 3.15*, *Theorem 3.4*, *Corollary 3.5*, *3.6*: ≤ 2 functions for each level, and functions can only be related to levels that adhere to **sG** type of granularity (A.123, A.124, T.9).

T. *Definition 3.11*: Granular levels that adhere to a granulation of **nG** or any of its subtypes, contain entities or instances that are φ-indistinguishable at a coarser-grained level, where φ denotes the type of relation that relates two adjacent levels or their distinguishing properties. Granular levels that adhere to a granulation of **sG** or any of its subtypes, contain entities or instances that are φ-indistinguishable at a coarser-grained level, where φ refers to scale-based measurable values or value ranges of the property under consideration. (D.24, D.25, D.26, D.27).

Having defined the domain, granular perspective, and, in this section, granular level, we proceed to the two principal relations among these framework components: RE and RL.

3.5 Relating levels, perspectives, and the domain

This section addresses the relation RE among the three components—domain, perspective, and level—-and §3.6 deals with RL and the granulation relations. As in previous sections, the list of constraints are summarised at the end of the sections.

3.5.1 Preliminaries: part-whole relations

In order to characterise RE between domain, perspective, and level, RL between levels, and the granulation relations GR properly, we first need to clarify part-whole relations.

3.5.1.1 Introduction

Many ontological and cognitive aspects of the part-whole relation have been discussed (*e.g.*, Artale *et al.* (1996a); Bittner and Donnelly (2005); Gerstl and Pribbenow (1995); Odell (1998); Shanks *et al.* (2004); Varzi (2004a, 2006a); Vieu and Aurnague (2005); Winston *et al.* (1987)) and proposals to include this relation to conceptual data modelling and knowledge representation languages have been suggested, such as (Barbier *et al.* (2003); Guizzardi (2005); Motschnig-Pitrik and Kaasbøll (1999)) for UML, (Bittner and Donnelly (2005); Lambrix and Padgham (2000); Sattler (1995); Schulz and Hahn (2000)) for DLs, Shanks *et al.* (2004) for ER, and Keet (2006a) for ORM. The reader is referred to (Keet and Artale, 2007; Keet, 2006d) for comprehensive analyses of related works on part-whole relations. These communities introduced different types of part-whole

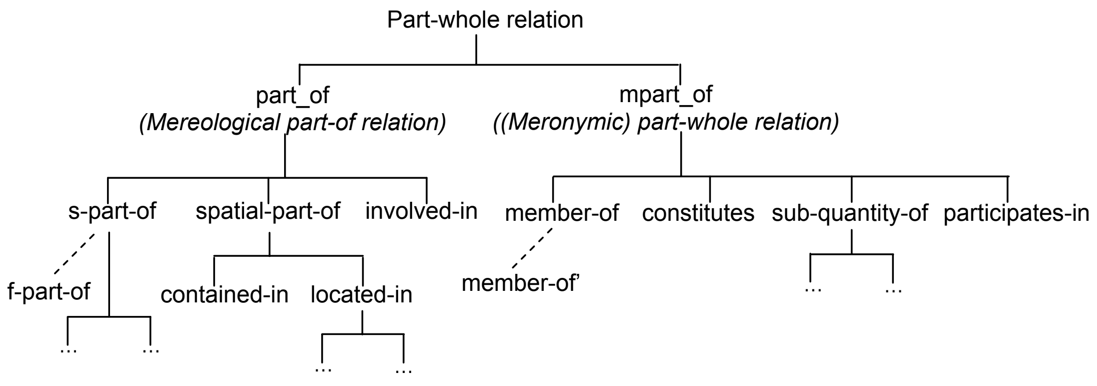


Figure 3.4: Taxonomy of basic mereological and meronymic part-of relations. s-part-of = structural part-of; f-part-of = functional part-of. Dashed lines indicate that the subtype has additional constraints on the participation of the entity types; ellipses indicate several possible finer-grained extensions to the basic part-whole relations.

relations, which sometimes are different types of mereological parthood relations, whereas others appear to be motivated by cognitive and linguistic use of 'part' (meronymy). The aim is to clarify the semantics of part-whole relations by considering the recent results of philosophical analyses on both mereological theories and foundational ontologies. A notable byproduct is the contribution to theoretical foundations of conceptual data modelling as well as facilitating usage of part-whole relations in software systems. To clarify the semantics of part-whole relations, a basic *formal* taxonomy of part-whole relations will be proposed that includes both mereological and meronymic part-whole relations, thereby also unambiguously specifying the hierarchical relations between the various part-whole relations introduced in the literature. The novelty of the approach taken is that the taxonomy is not merely a description of part-whole relations tailored to one specific usage only, but it uses well-known foundational ontology aspects and an implementation-independent, formal characterisation, to enable portability across different conceptual data modelling languages and application scenarios. The obtained taxonomy of part-whole relations and distinctions made will be justified by formally defining them with FOL formulæ, which are introduced and explained in the next subsection.

3.5.1.2 A Formal Taxonomy of Part-Whole Relations

The taxonomy proposed here structures the various part-whole relations, as introduced and discussed by Artale *et al.* (1996a); Barbier *et al.* (2003); Bittner and Donnelly (2005); Gerstl and Pribbenow (1995); Guizzardi (2005); Johansson (2004a, 2006); Keet (2006a); Motschnig-Pitrik and Kaasbøll (1999); Odell (1998); Sattler (1995); Schulz *et al.* (2006); Rector *et al.* (2006); Smith *et al.* (2005); Tan et al. (2003); Vieu and Aurnague (2005); Winston *et al.* (1987), by taking into account ontological distinctions[10], and it builds upon the lowest common denominator of theories of parthood, *Ground Mereology*, where $part\_of$ is partial order that is reflexive ($\forall x part\_of(x, x)$), antisymmetric ($\forall x, y(part\_of(x, y) \land part\_of(y, x) \rightarrow x = y)$), and transitive ($\forall x, y, z(part\_of(x, y) \land part\_of(y, z) \rightarrow part\_of(x, z))$).

The taxonomy is depicted in *Figure 3.4* and is explained in the remainder of this section, including formal definitions for the leaf types of part-whole relations. This taxonomy is a balance between typing ontologically-motivated relations useful for domain ontology development and conceptual data modelling up to the minimum level of distinctions to gain benefit from specify-

[10]The taxonomy does not deal with other facets of parthood relations, such as intra-part relations, the inverse relation *has_part*, and if the parts together are *all* parts that make up the whole; these aspects are beyond the current scope as they do not affect essential components of granularity. For a discussion and various options to address such issues, see, *e.g.*, Barbier *et al.* (2003); Guizzardi (2005); Lambrix and Padgham (2000); Motschnig-Pitrik and Kaasbøll (1999); Opdahl *et al.* (2001).

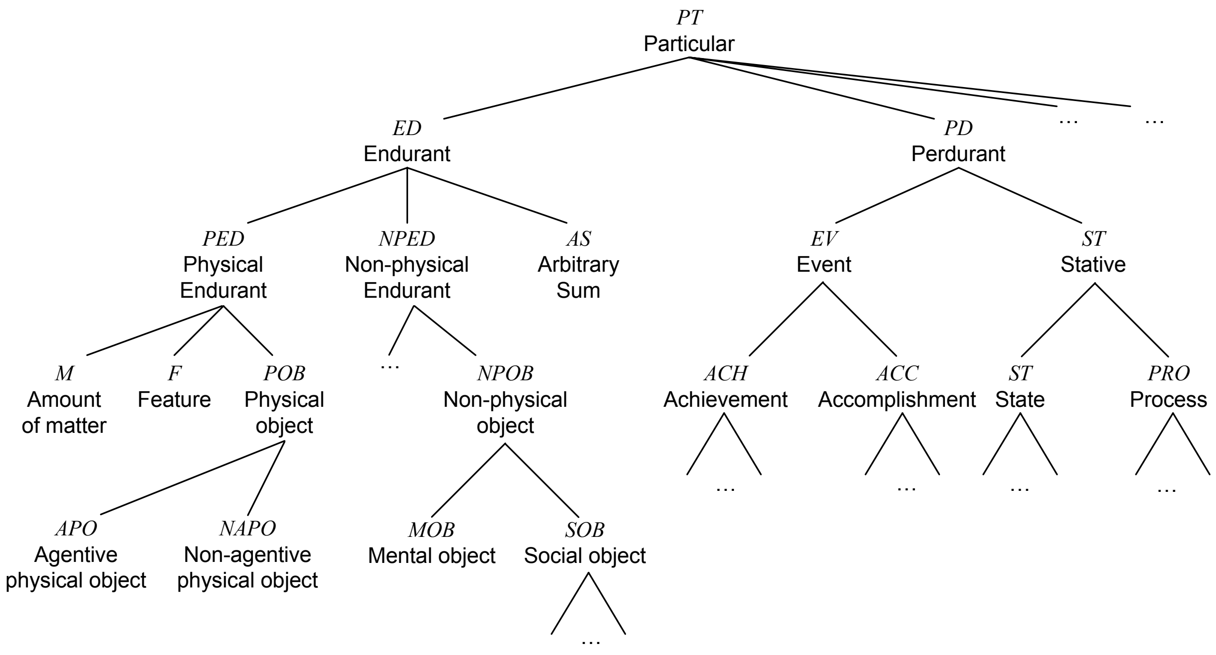


Figure 3.5: Graphical rendering of a section of the foundational ontology DOLCE. In colloquial natural language communication, endurant roughly maps to entity types and perdurant to processes or (objectified) relations/associations in conceptual data models.

ing part-whole relations more precisely, yet avoiding ontological exuberance that would deter modellers from using it in practice during the analysis stage; where applicable, current cut-off points will be justified in the explanation below.

Overview and preliminaries. The first principal distinction in the taxonomy is made between transitive and intransitive part-whole relations. The prime reason why this ontological distinction exists has to do with transitivity of the mereological parthood relation versus other part-whole relations. Successive distinctions between the relations are made based on the categories of the entity types participating in the relation—also called relata or domain and range restriction. The types of part-whole relations are disjoint. Further distinctions may be made on finer-grained categories of participating entity types and on other properties of the relation, such as existential dependence of the part on the whole or vice versa.

To be able to talk about the categories of the entity types involved in part-whole relations, we have to consider foundational ontologies from which we can borrow several top-level categories. Of the several prevalent foundational ontologies, such as DOLCE (Masolo *et al.* (2003)), BFO (BFO, 2007), OCHRE (Schneider, 2003), SUMO, and GFO (Herre and Heller, 2006), we chose DOLCE, the Descriptive Ontology for Linguistic and Cognitive Engineering, because it is the most comprehensively formalised one, has a mapping to OWL (OWL, 2004), and is used across several subject domains for development of ontology-driven information systems[11]. DOLCE includes the afore-mentioned Ground Mereology (as it uses Atomic General Extensional Mereology for atemporal parthood), and the definitions introduced in this section are fully compatible with DOLCE's formal characterisation. A relevant portion of the DOLCE foundational ontology is depicted in *Figure 3.5* (directly subsumed universals are disjoint (Masolo *et al.*, 2003)). In the remainder of this section, the abbreviated names of *Figure 3.5*, such as ED for *e*n*d*urant and POB for *p*hysical *ob*ject are used.

To avoid overloading terms, the non-mereological part-whole relation is labelled with $mpart\_of$. This part-whole relation is included in the taxonomy mainly for structuring pur-

[11]See for an overview: http://www.loa-cnr.it/DOLCE.html.

poses and is not intended for general use. Observe also that $mpart\_of$ is '*non*-transitive', that is, neither transitive nor intransitive[12], because it is not the case that intransitivity holds for *all* its subsuming part-whole relations with *all* their related instances. Thus, its semantics differs from the mereological $part\_of$, which is assumed transitive. Because of this distinction, the top relation in the taxonomy, $part\text{-}whole\text{-}relation$, is necessarily also non-transitive. Both $part\_of$ and $mpart\_of$ inherit from $part\text{-}whole\text{-}relation$ the typing of the relata (domain & range restriction), assumed to be the DOLCE's top-category Particular (PT). For explanatory purposes only, definitions in the meronymic branch contain the subscript "$_{it}$" to denote an *in*transitive relation ($\forall x, y, z(R(x, y) \land R(y, z) \rightarrow \neg R(x, z))$) or "$_n$" for *n*on-transitive.

Leaf types of part-whole relations. The meronymic leaf types are described and formalised first and the mereological parthood relations afterward.

The $member\_of$ relation (in the literature also called member-bunch, collection, collective or aggregation) belongs to the meronymic branch. The whole-side of the relation is generally denoted with a *collective noun*, *i.e.* a social entity with aggregations like Herd or Orchestra. The part-side are physical entities or the role they play, such as Sheep and Musician, respectively. Given that roles are dependent on the bearer, a physical object, we formalise $member\_of$ with the following definition.

DEFINITION 3.16 (member_of). *Let POB be physical object and SOB social object as defined in* DOLCE, *$mpart\_of$ a non-transitive meronymic part-whole relation, then*

$$\forall x, y(member\_of_n(x, y) \triangleq mpart\_of(x, y) \land (POB(x) \lor SOB(x)) \land SOB(y) \quad (3.7)$$

This $member\_of$ constrains the 'part' side of the relata to either physical objects (POB) or their roles as non-physical social objects (SOB) (a simplification of the work by Masolo *et al.* (2004)), while the whole the parts are aggregated into are constrained to be social objects SOB. With (3.7), it is easy to represent Odell (1998)'s "member-partnership" example relation for Husband and Wife in a partnership Marriage. The only addition is that the whole is existentially dependent on the part and vice versa: let ε be existential dependence (see also Guizzardi (2005)), then we can add a subtype *member_of'* which has "$\varepsilon(x, y) \land \varepsilon(y, x)$" added to definition (3.7).

The *material-object* relation, categorised as part-whole relation by Winston *et al.* (1987); Sattler (1995); Gerstl and Pribbenow (1995); Odell (1998), corresponds ontologically to *constitution* where a POB is related to an amount of matter (M) it is made of (see section 3.3.3 in Masolo *et al.*, 2003, for detail and justification)—*e.g.*, Statue and the Marble it is constituted of. Amounts of matter are generally denoted with *mass nouns* that are not countable. In natural language, the inverse $constituted\_of$ is used more often, where the whole is constituted of its material-parts.

$$\forall x, y(constitutes_{it}(x, y) \equiv\ constituted\_of_{it}(y, x) \triangleq mpart\_of(x, y) \land POB(y) \land M(x)) \quad (3.8)$$

The *sub-quantity-of* relation, also called quantity-mass or portion-object (*e.g.*, Sattler (1995); Odell (1998); Guizzardi (2005)), relates a smaller part-amount of matter (M) to a whole-matter (M), where the amounts of matter are either the same type of stuff, *e.g.*, a glass of Wine—in this case we require an additional measure for its quantity like glass & bottle or millilitre & litre—or, similar to Guizzardi (2005), the part-M is a different type of matter than the whole-M—*e.g.*, sub_quantity_of(Salt, Sea water). These examples are ontologically distinct, giving rise, in the first case, to a transitive $sub\_quantity\_of'$ relation and to an intransitive $sub\_quantity\_of''$ relation in the second case. In the second case, a part-M may have undergone a chemical reaction in the whole-M and, strictly speaking, not exist anymore compared to the part-M in isolation or existing in a different state (*e.g.*, the molecules have released a hydrogen atom upon dissolving

[12]non-transitive is not a new property but a short-hand notation for absence of declaring transitivity or intransitivity, which we use when it is known that the relation is neither transitive for all cases nor intransitive.

in the whole-M). Further, specific quantities may not matter for representing basic knowledge in ontologies, such as representing sub_quantity_of(Alcohol, Wine), but this is needed for conceptual data models where recording data on percentages of alcohol in beverages is needed. Therefore, (3.9) has the lowest common denominator, but it can benefit from further disambiguation.

$$\forall x, y(sub\_quantity\_of_n(x, y) \triangleq mpart\_of(x, y) \wedge M(x) \wedge M(y)) \tag{3.9}$$

The last meronymic relation is $participates\_in$ (*Definition 3.17*), a noun-feature/activity in linguistics (Motschnig-Pitrik and Kaasbøll, 1999), which relates an entity of the category endurant (ED) to the process (perdurant PD) it participates in (Masolo *et al.*, 2003)[13]—*e.g.*, an Enzyme that participates in a Catalytic reaction.

DEFINITION 3.17 (participates_in). *Let ED be endurant as defined in* DOLCE *and $mpart\_of$ a non-transitive meronymic part-whole relation, then*

$$\forall x, y(participates\_in_{it}(x, y) \triangleq mpart\_of(x, y) \wedge ED(x) \wedge PD(y)) \tag{3.10}$$

From this characterisation, it is clear *why* meronymic part-whole relations are at least non-transitive, but generally intransitive: the category of the part is usually different from the category of the whole[14], therefore one can neither make a chain with the same relation nor concatenate them with the mereological relations and assume transitivity holds.

Considering the mereology branch of the taxonomy, recollect that all types of relation are transitive and they can be formalised (specialised further) with proper parthood (3.11) as well, which is particularly relevant for the TOG.

$$\forall x, y(ppart\_of(x, y) \triangleq part\_of(x, y) \wedge \neg part\_of(y, x)) \tag{3.11}$$

We first encounter the $involved\_in$ relation, which relates two perdurants through the normal $part\_of$ relation (3.12) and through proper parthood (*Definition 3.18*), where the part-processes are components in sequential, parallel, or cyclic steps in the larger parent process; for instance, Chewing is involved in Eating and the KEGG pathway diagrams (KEGG). This definition concurs with the parthood argument restriction "(Ad2)" in DOLCE, $P(x, y) \rightarrow (PD(x) \leftrightarrow PD(y))$, and hereby is rendered more accessible for domain ontology development and conceptual data modelling.

$$\forall x, y(involved\_in(x, y) \triangleq part\_of(x, y) \wedge PD(x) \wedge PD(y)) \tag{3.12}$$

DEFINITION 3.18 (involved_in). *Let PD be perdurant as defined in* DOLCE *and $ppart\_of$ mereological proper parthood (3.11), then*

$$\forall x, y(involved\_in(x, y) \triangleq ppart\_of(x, y) \wedge PD(x) \wedge PD(y)) \tag{3.13}$$

We distinguish between two mereotopological relations for endurants (mereological relations that take into account space or location), one according to a 3-dimensional containment (3.14), another for 2-dimensional location (3.15), and their respective definitions with proper parthood (*Definition 3.19*). The definitions refer to both the endurant itself and the region (V) it occupies. DOLCE has an elaborate formalisation to refer to the region of an endurant (using quality, quale, and regions), which is abbreviated here with the properties *has_3D* and *has_2D*, respectively, because this attribute-approach generally provides sufficient details in conceptual data models for applications[15]. One can do away with the 2D/3D distinction and just refer to

[13]Observe that the $participates\_in$ relations does not violate DOLCE's Dd63 $PC_C(x, y) \triangleq \exists t(PRE(y, t)) \wedge \forall t(PRE(y, t) \rightarrow PC(x, y, t))$.

[14]The only exception being $sub\_quantity\_of$, which is under-specified.

[15]The relations do not reflect the full range of mereotopological and mereogeometrical complexities (Borgo and Masolo (2007); Varzi (2006a)), some of which could be useful for geographic and biological information systems, but they are less relevant for the more common domain modelling. It is a point of further research how these FOL theories can be transformed and rendered usable in the scope of granularity.

any kind of spatial region, but the distinction is included because it is a recurring relation in the informal discussions about part-whole relations for conceptual data modelling (e.g. Gerstl and Pribbenow (1995); Keet (2006a); Motschnig-Pitrik and Kaasbøll (1999); Odell (1998); Winston *et al.* (1987)) and can be accommodated for easily.

$$\begin{aligned}&\forall x, y(contained\_in(x,y) \triangleq part\_of(x,y) \land V(x) \land V(y) \land \\ &\qquad \exists z, w(has\_3D(z,x) \land has\_3D(w,y) \land ED(z) \land ED(w)))\end{aligned} \tag{3.14}$$

$$\begin{aligned}&\forall x, y(located\_in(x,y) \triangleq part\_of(x,y) \land V(x) \land V(y) \land \\ &\qquad \exists z, w(has\_2D(z,x) \land has\_2D(w,y) \land ED(z) \land ED(w)))\end{aligned} \tag{3.15}$$

DEFINITION 3.19 (Containment). *Let V be region and ED be endurant as defined in* DOLCE *and has_3D the relation between an endurant and its region, then*

$$\begin{aligned}&\forall x, y(contained\_in(x,y) \triangleq ppart\_of(x,y) \land V(x) \land V(y) \land \\ &\qquad \exists z, w(has\_3D(z,x) \land has\_3D(w,y) \land ED(z) \land ED(w)))\end{aligned} \tag{3.16}$$

where $ppart\_of$ is the mereological proper parthood relation (3.11).

For instance, contained_in(John's address book, John's bag). Containment is not always conceptualised as a particular type of parthood (Bittner and Donnelly (2005); Odell (1998); Schulz *et al.* (2006)), but, first, $contained\_in$ always meets the above-mentioned mereological parthood and proper parthood properties. Second, those considerations on conceptualizing parts & space exhibit a hopping back and forth between considering entity only, entity & region it occupies, and region only. With examples such as contained_in(actin filament, cell) and many others in biology, both parthood and containment hold, which means that x & z and y & w coincide exactly. To address these subtle, but important, distinctions, we explicitly include *structural* parthood (see below) in the taxonomy as different from containment. The same argument holds for location. Examples for location are located_in(Amsterdam, North Holland) or located_in(Mont Blanc, Alps). Note that the examples for location permit a finer-grained specification: the former relates city to province, which are entities by social convention, whereas the latter relates two physical entities (mountain and mountain range). Such ontological exuberance is not included here as it may be avoided with the TOG's C.

Last, we constrain the *structural* parthood relation (3.17), and its proper parthood version in *Definition 3.20.*

$$\forall x, y(s\_part\_of(x,y) \triangleq part\_of(x,y) \land ED(x) \land ED(y)) \tag{3.17}$$

DEFINITION 3.20 (structural proper parthood). *Let ED be endurant as defined in* DOLCE *and $ppart\_of$ mereological proper parthood (3.11), then*

$$\forall x, y(s\_ppart\_of(x,y) \triangleq ppart\_of(x,y) \land ED(x) \land ED(y)) \tag{3.18}$$

We can further constrain (3.17) by defining two subtypes of s_part_of for POBs or to relate two $NPOB$s to ensure POBs and $NPOB$s are not interleaved (3.19, 3.20).

$$\forall x, y(s\_part\_of'(x,y) \triangleq part\_of(x,y) \land POB(x) \land POB(y)) \tag{3.19}$$

$$\forall x, y(s\_part\_of''(x,y) \triangleq part\_of(x,y) \land NPOB(x) \land NPOB(y)) \tag{3.20}$$

In addition, and analogous to the $member\_of'$, we can add a constraint to the part-side of s_part_of to make it a *functional* part (Vieu and Aurnague (2005); Guizzardi (2005)) and label it f_part_of; informally, individual functional dependence captures that for x to function as X then y must function as Y. Functional parthood is, however, motivated by—as opposed to 'due to'—linguistics and has not (yet) been identified explicitly as an important part-whole relation for conceptual data modelling, although modellers surely have dealt with representing functional parthood. For instance, that f_part_of(Car Engine, Car) holds, with the meaning that the entity type Car denotes the set of canonical cars and each car cannot function normally without its canonical, working, engine (that instantiates the entity type Car Engine). It has been included to demonstrate ease of extension of the taxonomy. This concludes the characterisation of the eight leaf types.

Last, observe that by adhering to Ground Mereology in the taxonomy of part-whole relations, we get six other mereological relations 'for free'. These relations (3.21-3.26) are generally useful for domain ontology development and conceptual data modelling, and in particular for Geographical Information Systems and applications for biology and biomedicine; they are as follows (after Varzi (2004a)):

$$\forall x, y(overlap(x, y) \triangleq \exists z(part\_of(z, x) \land part\_of(z, y))) \tag{3.21}$$

$$\forall x, y(underlap(x, y) \triangleq \exists z(part\_of(x, z) \land part\_of(y, z))) \tag{3.22}$$

$$\forall x, y(overcross(x, y) \triangleq overlap(x, y) \land \neg part\_of(x, y)) \tag{3.23}$$

$$\forall x, y(undercross(x, y) \triangleq underlap(x, y) \land \neg part\_of(y, x)) \tag{3.24}$$

$$\forall x, y(proper\_overlap(x, y) \triangleq overcross(x, y) \land overcross(y, x)) \tag{3.25}$$

$$\forall x, y(proper\_underlap(x, y) \triangleq undercross(x, y) \land undercross(y, x)) \tag{3.26}$$

This concludes the preliminaries section. We now return to the characterisation of the TOG relations.

3.5.2 Relating GL to GP and GP to D^f

Recollecting *Figure 1.1* and *3.1*, the rectangles for GL are within the rectangle of GP, which does not reveal how they relate, other than labelling the relation with the mnemonic RE. Here, we look at three options how this relation can be represented, of which the first two are based on the two basic distinct representations of granularity.

1. Set-theoretic: although the GL rectangles in *Figure 1.1* are ordered and Venn-diagram-like within a GP, granular levels are neither an ordered set in GP, because they are not the set extension of GP, nor subsets of a perspective, because GL is not a taxonomic subtype of GP;
2. Mereological: as mentioned before (p.51), GP is not the mereological sum of its levels, although this does not exclude using a proper parthood relation for RE.
3. Other types of relations: $used\_in$, $belongs\_to$, and the bridge rule from contextual reasoning are considered. $used\_in$ is problematic because it has ambiguous semantics in linguistics and does not exclude that a level might as well be used somewhere else, which is not the case. Nor is $belongs\_to$ appropriate, because the meaning of belonging is as of yet still ambiguous. The bridge rule or 'switching' in contextual reasoning (see §5.2.2 for explanation and comparison) require additional semantics to reflect the modified push/pop mechanism, which will make it not interoperable with existing context reasoning systems; for these two reasons this relation can be set aside as well.

Given the problems of points 1 and 3, we take a closer look at the second one. Using any of the parthood relations, it must be at least a *proper* parthood, because there is always more than one granular level in a perspective (*Theorem 3.2*). In addition, taking both the specification of leaf types of part-whole relations (§3.5.1), and recollecting that both GL and GP are concepts, with $CN \subseteq ED$, then we have reduced the options to structural parthood or spatial parthood for either containment or location where the region and object coincide exactly (see also *Figure 3.4*). Both containment and location have a relation for their coordinates, which fits well with the conceptual space or portion of reality of a granular level. However, with the constraints on the relata, it will have to be $ppart\_of$ where only intuitively also "contained in" holds. Therefore, we make RE a subtype of proper parthood $ppart\_of$ and type it with the TOG components as relata. It is not simple $part\_of$ because a level cannot be also part of another perspective (from *Theorem 3.3*) and there is always a remainder, being at least one other level (*Theorem 3.2*).

The same three options to relate GP to D^f are available as for relating GL to GP and the same arguments hold, hence, the notion of parthood is appropriate here as well. Given that parthood is transitive, then if GL is in GP and GP is in D^f, GL is in D^f. Although the deduction is not immediately useful at this stage, it will be for the functions in Chapter 4 and might be for dealing with multiple domains. The transitivity in the other direction, the has_ppart relation, is valid with the restriction that it holds for the default case of one D^f (see §5.5.1 for multiple domains). This brings us to the definition for RE:

DEFINITION 3.21 (RE). $\forall x \exists y$ *The relation* $RE(x, y)$*, and its inverse* RE^-*, hold between two of the three granularity components iff*
- $GL(x) \wedge GP(y)$ *or* $GP(x) \wedge D(y)$ *for* $RE(x, y)$*, and*
- $D(x) \wedge GP(y)$ *or* $GP(x) \wedge GL(y)$ *for* $RE^-(x, y)$*.*

Further, $RE(x, y) \rightarrow ppart\_of(x, y)$ *and* $RE^-(x, y) \rightarrow has\_ppart(x, y)$*.*

With this definition, preceding analysis, and proper parthood, we can demonstrate that RE has the properties of being acyclic and transitive (*Lemma 3.13*). Acyclicity is formally characterised as that an object x does not have a path to itself, *i.e.*, let φ be a variable ranging over one or more relations, then $\forall x \neg\varphi(x, x)$, which can be written in full for RE as

$$\forall x_1 ... x_i ... x_n (\phi(x_1, x_i) \triangleq (RE(x_1, x_2) \wedge ... \wedge RE(x_{n-1}, x_n) \wedge (1 \leq i \leq n) \rightarrow x_1 = x_i)) \tag{3.27}$$

LEMMA 3.13. RE *and* RE^- *are acyclic and transitive.*

<u>*Proof.*</u> We first demonstrate transitivity and then acyclicity. Given
$RE(x, y) \rightarrow ppart\_of(x, y)$ (from *Definition 3.21*)
and from Ground Mereology that $ppart\_of$ is irreflexive, asymmetric, and transitive, therefore RE is transitive as well. Acyclicity, *i.e.*, the negation of (3.27), holds, because of (i) the domain and range restrictions on RE (*Definition 3.21*) that prohibits a $RE(x, y)$ where $D^f(x)$ and $GL(y)$, or, say, $GP(x)$ and $GP(y)$, and (ii) identity of the domain and range such that $\neg(x_1 = x_i)$ can be shown because instances of GL, GP, and D^f are distinct domain elements in any interpretation $\mathcal{I}$ thanks to their definitions in the TOG.
This argument holds also for the inverse relation RE^-, where $RE^-(x, y) \rightarrow has\_ppart(x, y)$, and the typing of domain and range restrictions of RE^-. □

Observe that there is no referral to $ppart\_of$ for acyclicity. This is because acyclicity of $ppart\_of$ can only be proven with identity of x, hence, in Extensional Mereology, but not in Ground Mereology.

Recollecting *Theorem 3.2*, *Proposition 3.3*, and *Corollary 3.2*, we now have characterised them more clearly as being applicable with respect to RE according to the semantics as given in *Definition 3.21* and *Lemma 3.13*.
Summarising the constraints on RE, they are:

U. A GL is contained in a GP, through RE, only once, a GP always contains ≥ 2 GLs that are distinct; a GP is contained always only once in the D, a D always contains ≥ 1 GPs (T.7, A.109, T.8, A.110).

V. *Lemma 3.13*: RE and RE^- are acyclic and transitive (D.5, D.6, D.9, A.38, A.39, A.70, A.71, A.72, A.73, A.74, A.75).

W. Ground Mereology (A.27-A.32, A.34, A.35) and part-whole taxonomy (A.36-A.43).

3.6 Relation RL between two levels in one perspective

Questions. This paragraph contains the analysis of the relation RL between levels contained in one perspective. The main questions that will be addressed are the following. Is, or should be, RL of the same type within a GP? If RL is of the same type within one GP, can it be of a different type in another GP? Are the relations between adjacent levels in all granular perspectives of the same type, such as always $part\_of$, or will they be of different type? If the latter: what different types of relations between levels do, or can, exist?

RL is a binary relation between two adjacent levels in one perspective that are related to each other hierarchically, and is atemporal. Recollecting that GL is categorised as a type of concept CN, then RL is the relation that relates these concepts. It is possible that entities (/types) contained in a level gl_j, where $gl_j \prec gl_i$, are related to entities (/types) contained in gl_i through a different type of relation than the one RL can be mapped onto. Such a relation between entities (/types) is called *granulation relation*, *GR*, which is asymmetric, but not necessarily transitive, because it also depends on the type of granularity; this will addressed in §3.6.2. To analyse the type of relation RL between two adjacent levels, the distinction between granulating according to the same arbitrary scale (RL^s) and non-scale-dependence (RL^n) might be relevant. This will be assessed in the next two sections.

3.6.1 RL^s between scale-dependent levels

To determine the appropriate relation between scale-dependent levels, we have to consider the types of granularity that may indicate the relation between levels within one perspective. For scale-dependency it is important to recognise that it is the *representation* of the entity (/type) that counts. The representation of the real-world entity (/type) is different at different levels of granularity and the representation is the resultant of the *combination* of the entity (/type) and the scale at which it is considered, such as **sgrG**'s Cell as line, lipid bi-layer and 3-dimensional structure and **saoG**'s grid-structure in conjunction with some entity. However, the scale is the decisive factor, thereby reducing the inquiry to *how do two sections of a scale relate to each other?* For instance, m^2 is part of km^2. Alternatively, one can remodel scales as assigning the smallest unit to be the Urelement and build up the coarser measurement units from that Urelement and represent it as set-subset relations. However, the latter is inflexible for two reasons. First, if one decides to add a finer-grained level, definitions of all other levels and their $conv$ functions must be updated and thereby changing the semantics of the levels in that particular granular perspective, whereas conceptually nothing has changed at all. In addition, an infinite regress leads to the infinitesimally small (and infinitely large), but there are few stable boundaries for scales[16] and representing, say, a century in an amount of pico-seconds gives a precision to Century that does not exist in such fine-grained detail (see also §5.3.3). Of course, by representing granularity using the scales, one sets boundaries to avoid such issues. Second, interoperability issues arise when different people take different Urelements and represent the same thing differently, thereby strongly suggesting that the set-theoretic approach is not the optimal mechanism for representation. Linking the levels with a parthood relation and using the intension of GL and the entity (/types) it contains, on the other hand, permits one to add new coarser- and finer-grained levels without disrupting semantics of levels that are declared already, thus fostering stability and reusability of domain granularity frameworks.

Proposition 3.11. *RL^s maps onto a parthood relation.*

The appropriate type of parthood relation then easily falls in place:

[16]This concerns the question of "what is the end of the scale?" and the notion of going off the—human-made—scale, not the existence and ontological nature of boundaries (for the latter, see an introduction by Varzi (2004b)).

LEMMA 3.14. $RL^s(x, y) \rightarrow s\_ppart\_of(x, y)$.

Proof. Given the TOG definition and some basic DOLCE axioms
$\forall x(GL(x) \rightarrow CN(x))$ (*Definition 3.14*)
$\forall x(CN(x) \rightarrow ED(x))$ (from DOLCE)
$\forall x(ED(x) \rightarrow \neg PD(x))$ (from DOLCE)
$\forall x(ED(x) \rightarrow \neg AB(x))$ (from DOLCE)
where AB is an abbreviation for ABstract that is used for scales, and consulting the taxonomy of part-whole relations (§3.5.1), RL^s can be of type s_part_of that has ED as relata, because RL relates GLs (not scales). Given the *strict total* order between levels (*Lemma 3.9*), it has to be *proper* part, thus s_ppart_of. □

Thus, we might define the relation between scale-dependent granular levels, RL^s, as follows.

DEFINITION 3.22 (RL^s, preliminary version). *$\forall x$ and $GL(x)$, relation $RL^s(x_i, x_j)$ between two adjacent scale-dependent levels and $x_i \prec x_j$, then $RL^s(x_i, x_j) = s\_ppart\_of(x_i, x_j)$ and for the inverse $RL^{as-}(x_j, x_i) = has\_s\_ppart(x_j, x_i)$.*

The inverse relation has_s_ppart is the usual inverse relation of the s_ppart_of relation. A particular level always has the lower level except if that level is the lowest one defined; this is implied with the "$x_i \succ x_j$" (x_j cannot be $\perp$ because it is an instance of GL).

3.6.2 RL^n between non-scale-dependent levels

Unlike RL^s, it may seem that there are several candidates for the relation between the levels in the non-scale-dependent granularity, RL^n. This is because one cannot say *a priori* what each of the properties of the criterion are and which relation is used in the hierarchy of the entities (/types), compared to the certainty of using a scale in the scale-dependent granularity. A clear distinction has to be drawn, however, between relating *levels* versus relating its *contents*. The former is covered by RL, whereas the latter is covered by GR and involves a more precise specification of **nrG**.

3.6.2.1 RL between levels

Again, the main point revolves around taxonomic and partonomic relations, as was discussed in Chapter 2 and in the previous section, which has both *ontological* and *representational* motivations. Regarding representational motivations, a set theoretic approach is largely interchangeable with a mereological representation (see Pontow and Schubert (2006) for a comparison), hence of secondary importance to the ontological motivations. Regarding ontology, the following can be observed. With the taxonomic approach, each layer of nodes—level of depth in the tree—resides in a separate level, which might suggest a mapping as $RL^n \equiv is\_a$. However, it does not imply that this also should hold for the granular levels: gl_j is_a gl_i may be valid for GL's values but gl_j does not have more properties—only the contents may have. Looking at mereology, we would have at least $RL^n = part\_of$. Considering the part-whole taxonomy once more, those relations are *candidate*-relations for relating levels of granularity. The mereological part-whole relations in the left-hand branch are *always* transitive, whereas the part-whole relations in the right-hand branch are not necessarily transitive. Given that a granulation hierarchy is transitive (strict total ordering), we need at least the mereological $part\_of$ relation for RL^n as well.

PROPOSITION 3.12. *RL^n maps onto a parthood relation.*

The appropriate type of parthood relation then easily falls in place, like for RL^s, which added in *Lemma 3.15* for completeness.

LEMMA 3.15. $RL^n(x, y) \rightarrow s\_ppart\_of(x, y)$.

Proof. The same argument as *Lemma 3.14*, but then for RL^n. $\square$

A tentative definition of RL^n is analogous to that of RL^s, therefore we can proceed directly to the final definition of RL.

DEFINITION 3.23 (RL). *$\forall x_i, x_j$ relation $RL(x_i, x_j)$ between two adjacent granular levels in a perspective, where $GL(x)$, $(GL(x_i) \leftrightarrow GL(x_j))$, and $x_i \prec x_j$, then*

- *$RL(x_i, x_j) \rightarrow s\_ppart\_of(x_i, x_j)$ relating a fine-grained level to a coarse-grained level, and*
- *the inverse relation, RL^-, maps onto $RL^-(x_j, x_i) \rightarrow has\_s\_ppart(x_j, x_i)$, relating the coarse-grained level to a finer-grained level.*

With this definition resulting from the analysis, it has become trivial to answer the two main questions set out a the beginning of this paragraph:

THEOREM 3.5. *RL is of the same type, s_ppart_of, not only within some particular instance of GP, but it is of the same type between granular levels in* all *granular perspectives.*

In addition, from *Definition 3.23* and previous results on granular levels (§3.4) and relations, two properties can be proven: the 1:1 multiplicity on the levels participating in any RL (or RL^-) in *Theorem 3.6*, and acyclicity of RL (RL^-) in *Lemma 3.16*.

THEOREM 3.6. *The multiplicity (cardinality) of RL and RL^- is 1:1.*

Proof. Let x, y, $z \in GL$, $GP(w)$, and we temporarily introduce RL_d for levels that are *d*irectly related (as opposed through transitivity)
$\forall x, y(RL_d(x, y) \rightarrow \neg\exists z\ (RL(x, z) \wedge RL(z, y)))$
Let us take $RL_d(x, y)$, $RL_d(x, z)$, $RE(x, w)$, $RE(y, w)$, $RE(z, w)$, then $x \neq y$ and $x \neq z$ hold thanks to *Definition 3.23* and proper parthood. Following *Definition 3.14* (GL),
$x \sigma y \rightarrow \neg(x \simeq y)$ (*Lemma 3.6*)
$\forall z, \phi((\phi \rightarrow sG) \wedge GP(z) \wedge has\_granulation(z, \phi) \rightarrow \forall x \exists^{\geq 2} v, w((C(x) \wedge RC(z, x) \wedge GL(w) \wedge$
$RE(w_i, z) \wedge RE(w_j, z) \wedge has\_value(x_i, v_i) \wedge has\_value(x_j, v_j) \wedge$
$R(v) \wedge (RL(w_i, w_j) \wedge RL^-(w_j, w_i)) \rightarrow (v_j > v_i)))$ (*Lemma 3.12*)
then for the levels' values, $\varepsilon_x < \varepsilon_y$ and $\varepsilon_x < \varepsilon_z$. Moreover, by
$\forall x(GP(x) \rightarrow \exists! y, \phi(RC(x, y) \wedge has\_granulation(x, \phi)))$ (*Theorem 3.1*)
we must have $\varepsilon_y = \varepsilon_z$ for they use the same C and TG, hence also $y = z$. However we should have $y \neq z$, because of
$\forall x_1, ..., x_n, y(GL(x) \wedge GP(y) \wedge RE(x, y) \rightarrow \neg(x_1 = x_2) \wedge ... \wedge \neg(x_{n-1} = x_n))$ (*Corollary 3.3*)
thereby leading to a contradiction. Thus, any granular level participating in RL has exactly 1 adjacent coarser-grained level. The analogous argument holds for the inverse RL^-. Therefore, the relation has a 1:1 multiplicity. $\square$

LEMMA 3.16. *RL and RL^- are acyclic.*

Proof. We have that
$\forall x, y(RL(x, y) \rightarrow s\_ppart\_of(x, y))$ (*Definition 3.23*, based on *Lemma 3.14, 3.15*)
$\forall x, y(s\_ppart\_of(x, y) \rightarrow ppart\_of(x, y))$ (*Definition 3.20*)
hold, where the latter adheres to Ground Mereology, thus having the properties irreflexivity (ϕ), asymmetry (ψ), and transitivity (φ):
$\forall x \neg ppart\_of(x, x)$ (ϕ)
$\forall x, y(ppart\_of(x, y) \rightarrow \neg ppart\_of(y, x))$ (ψ)
$\forall x, y, z(ppart\_of(x, y) \wedge ppart\_of(y, z) \rightarrow ppart\_of(x, z))$ (φ)
therefore RL also has at least these properties through inheritance. Let α denote the acyclicity property, then $\alpha \rightarrow \psi$ and $\psi \rightarrow \phi$, and only the stronger notion of 'not through a path' has to be proven for RL. RL relating levels $x_1 ... x_n$ (shorthand notation): ... $\prec x_k \prec x_j \prec x_i \prec x_h \prec$ ..., now add the cycle $x_i \prec x_k$, which results in two coarser-grained levels (x_k and x_h) for x_i, which violates *Theorem 3.6*. The proof for acyclicity of RL^- follows the same argument. $\square$

Having fully characterised RL with a 1:1 cardinality constraint and acyclic subtype of s_ppart_of, there is still the open issue mentioned in the introduction of this section regarding relating the contents across levels. This is addressed in the next subsection.

3.6.2.2 Constraining the relations for the nrG type of granularity

To be more precise about which relations can be used for granularity, the currently permitted granulation relations between the entities (/types) will be added to the taxonomy of types of granularity. This requires an assessment of *which* relations should be included in the TOG *where* and *how* do they affect any other TOG constraints? The answers are shown already in the bottom-half section in *Figure 3.1* and will be explained in the remainder of this section.

The definition for granulation relation is introduced first.

DEFINITION 3.24 (Granulation relation). *A granulation relation GR is a binary relation for granulating entities (/types) in D^s.*

Aside from the subsumption relation, we can pick up the part-whole relations from §3.5.1 where we left it before defining RL and answer the question: which of the part-whole relations in *Figure 3.4* are relevant for granularity? *Example 3.6* illustrates several applicable relations for the topic introduced in *Example 3.5*.

Example 3.6. For a each type of part-whole relation, the domain is indistinguishable at the coarser level where the range resides:

- Parthood relations between endurants, such as functional parthood f-part_of(A subunit, Cholera toxin) as the toxin will not function—cease to be a toxin—without the A subunit, and merely structural parthood s-part_of(β-subunit, G_s protein).
- Sub-process to process relations, related through the $involved\_in$ relation: for instance, involved_in(Covalent binding, Activation) where binding covalently is a lower level sub-process of activation of the protein.
- Spatial containment with contained_in(Phosphodiesterase, Cell), but not a structural part because it moves around in cell plasma.
- Last, one can say that Phosphodiesterase enzyme participates_in the process of Breaking down cAMP; thus, relating an endurant and a perdurant.

The relations illustrated here already capture the semantics in more detail compared to Hobbs' simplification function and other's subsumption and parthood relations, but which now can be defined more accurately. $\diamondsuit$

Analysing the example in conjunction with the taxonomy of part-whole relations, this *suggests* subtypes of GR to be, aside from is_a and $ppart\_of$, also $involved\_in$ and $contained\_in$, and the part-whole relation $participates\_in$. In addition to these examples, **nacG** relates physical or non-physical objects to a non-physical object, which corresponds to using $member\_of$. Comparing this **nrG** list of relations with the part-whole relations in *Figure 3.4*, not all of them are used as GR. We go through each of the excluded relations and justify the decisions. Pending potential further disambiguation, $sub\_quantity\_of$ either requires scales and thus is covered already with RL and the $conv$ function, or can be covered with parthood (cf. the example about alcohol and wine), hence, including the current definition of $sub\_quantity\_of$ is redundant. $located\_in$ could have been included, but the 2D aspects can be covered already with the scales & maps using scale-dependency. The material-object relation $constitutes$ does not granulate anything: one either talks about the same object from a different view at the same level of granularity, or one refers to the amount of matter (part of) the whole is made of. The $member\_of'$ and $f\text{-}part\_of$ enjoy additional constraints on the relation, but do not differentiate with their supertype on the relata.

Some complex relations and examples of 'granularity' in the literature seem to defy a minimal basic set of relations for granularity. Three such relations that may be appealing intuitively as candidate-granulation relations are discussed in the following example, where it also will become clear why they are not added as a type of GR.

Example 3.7. Given the applicability of the parthood and subsumption relations, one can question if this suffices or if further relations should be permitted *a priori* at the domain-independent layer. Several options pass the revue, which can either be represented with one of the GRs after all, are ontologically still too problematic to merit inclusion, or is certainly not a type of GR.
Let us take $causes$ and its inverse $caused\_by$. Granularity in causality, if any, concerns fine-grainedness *between* causal relations but not the relata *of* a particular relation; thus, it cannot be a GR. For instance, 3 levels containing:

1: Ticks $cause$ Lyme disease.
2: Hard ticks infected with a bacterium $cause$ Lyme borreliosis in humans.
3: Infection of *Homo sapiens sapiens* with the bacterium *Borrelia burgdorferi* sensu lato, transmitted by the reservoir host hard tick *Ixodus* spp. (in particular *I. scapularis*) $cause$s the infectious disease Lyme borreliosis in *Homo sapiens sapiens*. (Wang *et al.*, 1999).

and so forth, introducing or abstracting away more detailed causal mechanisms. In this example, (1) is incorrect: *Borrelia burgdorferi* sensu lato is the causative agent of Lyme disease and tick the vector. Setting aside the false abstraction from (2) to (1), statements (2) and (3) are correct, where the *combination* of relation & relata is finer-grained: the domain of (2), 2_D, is preceded by that of the third, 3_D, and likewise for their ranges that $3_R \prec 2_R$. At best, (2) and (3) might be contained in a perspective granulated according to **nfG** to permit folding from (3) to (2). The notion of granularity might aid identification of different levels of detail and disambiguation of causality[17] to limit miscommunication between the various disciplines in biology and medicine.
The possibilities for GR become increasingly unclear where more intuitive or historically established levels are considered. One that currently receives attention in bio-ontology research is the $develops\_from$ relation. Suppose we have the develops from relation as $dF(x, y)$, where x is at a higher level than y, then dF(Embryo, Fertilised egg) and dF(Gastrula, Blastula)—but what level is it? A development *stage* in time is not a granular *level*. Development of the gastrula from the blastula is *part of the process* of development from the fertilised egg into an embryo and they relate to each other as processes, *i.e.*, through $involved\_in$ as in involved_in(Gastrula dF Blastula, Embryo dF FertilisedEgg). A tree structure built up with dF (*Figure 3.6-A*) can have granularity applied by enclosing the innermost relations in a lower level of granularity and the outermost nodes in a higher level of granularity, as depicted in *Figure 3.6-B*: if we have $A\ dF\ B$, $B\ dF\ C$, and $C\ dF\ D$, then $involved\_in(B\ dF\ C,\ A\ dF\ D)$, but also $involved\_in(C\ dF\ D,\ A\ dF\ D)$ and $involved\_in(A\ dF\ B,\ A\ dF\ D)$ (*Figure 3.6-C*). These myriad of possibilities complicate consistent usage of levels, because with each change in the tree, the allocation of entities to a level changes.
Moving further into the informally accepted divisions, the omics spaces, or layers, are: genomic, transcriptomic, proteomic, metabolomic, and phenomic (Toyoda and Wada, 2004), which are, roughly, linked through time displacement, causality, and parthood, and do not correspond to levels of *granularity*, although particular aspects can be granulated, where, *e.g.*, proteins participate in a metabolic pathway.
Tange *et al.* (1998) identified three levels of "granularity sets" by "combining the most encountered granularities of medical history, physical examination, and progress notes, as found in the literature". An example for the three coarse, intermediate, and fine levels is Physical examination $\succ$ Lungs $\succ$ Auscultation, thereby mixing a process, structural part of the human body and "type of observation", hence conflating granulation criteria. It is con-

[17]Causality is subject to much philosophical investigation from both an ontological and biological viewpoint (*e.g.* Johnson, 1990; Lehman *et al.*, 2004; Ellis, 2005). Even if an engineering approach is taken, it is cumbersome and error prone to encode and maintain *e.g.* "if Tick at level x in perspective y, then Tick = Causative agent else Tick = Vector" and so forth.

ceivable to continue searching for types of relations that relate granular levels. Take, for instance, an 'information content' relation for Gene $\prec$ GeneComplex $\prec$ OrganismalGenome $\prec$ PopulationGenomePool where the information encoded on a gene is a smaller piece of information than what is encoded on a gene complex and so forth. This 'information' relation, however, can also be represented as a part-whole relation with corresponding characteristics where the information on the gene is ultimately part of the information encoded on the population genome. Moreover, the relation between the structural components (gene, genome) and the information it 'bears' or 'encodes' has some philosophical issues.
A proliferation of types of relations should be avoided at present, because it still requires substantial investigation into the nature of the relation and permitting arbitrary relations may obfuscate that what is being granulated actually should be subject to closer ontological inspection. In the opposite direction, one could reduce GR to is_a and $part\_of$ instead of taking a careful multiplicative approach, but the major disadvantage is that it over-simplifies granularity. ♢

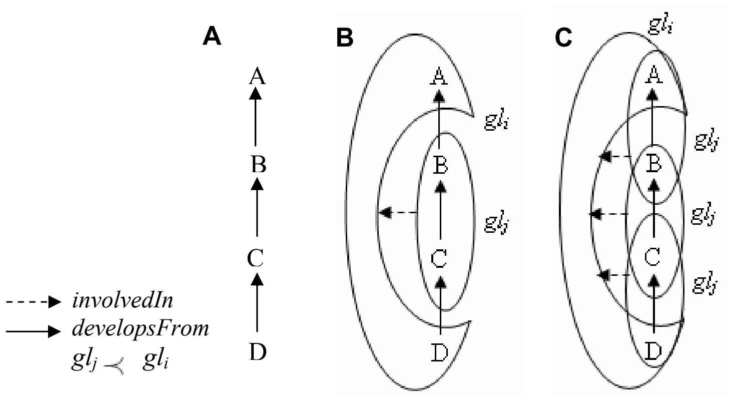


Figure 3.6: Organising the $develops\_from$ relations to allocate them in different levels of granularity through granulation by **nrG**'s $involved\_in$.

There are two places where *GR* will be inserted in the TOG. First, to be able to manage transparently which *GR*s are permitted for use in granulation, we can attach it as a property (attribute) to the **nrG** type with the $has\_permitted$ relation (*Definition 3.25*) and constrain that for each usage of **nrG**, one has to chose only one type of *GR* (*Proposition 3.13*).

DEFINITION 3.25 (has_permitted). *For each* **nrG***, there is is some granulation relation* GR*, which are related through* $has\_permitted$*:*
$\forall \phi, \psi(has\_permitted(\phi, \psi) \rightarrow TG(\phi) \wedge (\phi \rightarrow nrG) \wedge GR(\psi))$.

PROPOSITION 3.13. *Each usage of* **nrG** *is related to exactly one* GR*:*
$\forall \phi(nrG(\phi) \rightarrow \exists! \psi(has\_permitted(\phi, \psi))$.

That *GR* is related to **nrG** does not mean that the relations are not used by other types of granularity: they are categorised under **nrG** for conceptual clarity. One or more of those *GR*s can be used by one or more *TG*s, most notably parthood for scales and subsumption with **nasG**. To provide a means to relate a type of granularity with the granulation relation it uses, the $uses\_GR$ relation is introduced.

DEFINITION 3.26 (uses_GR). *Each type of granularity* TG *may use a granulation relation* GR *through which the contents of different levels are related:* $\forall \phi, \psi(uses\_GR(\phi, \psi) \rightarrow TG(\phi) \wedge GR(\psi))$.

Last, we look into the third aspect: *how GR* affects other TOG constraints and components. Given that TG is related to GP and GL, several additional constraints can be inferred. First, because

$member\_of$ is non-transitive and $participates\_in$ is intransitive, the maximum number of GLs in GPs where GR is either one is 2 (*Lemma 3.17*). Second, propagating from chapter 2, **nfG** uses at least 2 granulation relations (*Proposition 3.14*) and **nacG** is restricted to $member\_of$ (*Proposition 3.15*). Third, we now can state more specifically the **nacG** characterisation from chapter 2, which uses $member\_of$ to relate entities (/types) between the levels and for which the types residing in the levels are distinct (*Proposition 3.16*). In contradistinction, for **nasG** we have that the entities in a lower level are also (sub-) types of the contents in the higher levels, and, hence, use is_a as GR (*Proposition 3.17, Proposition 3.18*).

LEMMA 3.17. *Granular perspectives that have a granulation where the* ***nrG*** *relation is* $member\_of$ *or* $participates\_in$ *contain exactly 2 granular levels.*

Proof. Given the facts that RL is transitive (*Definition 3.23*),
$\forall x(GP(x) \rightarrow \exists^{\geq 2} y(RE^{-}(x,y) \wedge GL(y)))$ (*Theorem 3.2*)
and that $participates\_in$ and $member\_of$ are not transitive (*Definitions 3.17* and *3.16*), therefore we have to limit the amount of levels to ≤ 2 to maintain logically correct inferencing across levels and its contents. Hence, ≥ 2 & ≤ 2 equals exactly 2. □

PROPOSITION 3.14. *For each GP that has type of granulation* ***nfG****, then at least 2 types of granulation relations are used:* $\forall \phi((\phi \rightarrow nfG) \rightarrow \exists^{\geq 2}\psi(uses\_GR(\phi,\psi) \wedge \neg(\psi_i = \psi_j)))$.

PROPOSITION 3.15. *For each GP that has type of granulation* ***nacG****, then the granulation relation is limited to* $member\_of$*:* $uses\_GR(\phi,\psi) \wedge (\phi \rightarrow nacG) \rightarrow (\psi = member\_of)$.

PROPOSITION 3.16. *For each GL that adheres to type of granulation* ***nacG****, the types in the levels have to be distinct:* $(\phi \rightarrow nacG) \wedge adheres\_to(x_i,\phi) \wedge adheres\_to(x_j,\phi) \wedge RL(x_j,x_i) \wedge in\_level(\psi,x_i) \wedge in\_level(\varphi,x_j) \rightarrow (\varphi \rightarrow \neg\psi)$.

PROPOSITION 3.17. *For each GL that adheres to type of granulation* ***nasG****, the entities in the finer levels are also of the type that reside in coarser levels:* $(\phi \rightarrow nasG) \wedge adheres\_to(x_i,\phi) \wedge adheres\_to(x_j,\phi) \wedge RL(x_j,x_i) \wedge in\_level(y,x_j) \wedge in\_level(z,x_i) \rightarrow (PT(y) \rightarrow PT(z))$.

PROPOSITION 3.18. *For each GL that adheres to type of granulation* ***nasG****, GR is* is_a*:* $(\phi \rightarrow nasG) \rightarrow uses\_GR(\phi,\psi) \wedge (\psi = is\_a)$.

Finally, a proposition can be added that *currently*, the types of GR are constrained to six types, and their inverse relations.

PROPOSITION 3.19. *The granulation relation GR subsumes the following relations between entities (/types) in finer- and (adjacent) coarser-grained levels:* is_a, $participates\_in$, $member\_of$, $ppart\_of$, $involved\_in$, *and* $contained\_in$, *as defined in* Definition 3.16-3.19 *and (3.11).*

Consistency in relations used in one perspective. From *Theorem 3.5*, we already know that there is only one type of relation that RL maps onto and that therefore it is obvious that always the same type of relation is used between granular levels. Nonetheless, the question is worth a closer inspection when specified slightly different by taking GR into account: "should the relation between the coarser- and finer-grained *entities (/types)* contained in different levels be of the same type within a GP?". This is also answered in the positive, because levels in a perspective are defined according to its criterion C and the perspective adheres to a particular type of granularity, therefore the relation must be of the same type. If this is not the case during development of a domain granularity framework, then this demands (re-)assessment of the criterion used for granulation, because inconsistency in the relation indicates that, analogous to multiple inheritance in ontologies, different criteria are mixed to construct the levels. In addition, the related questions posed at the start of this paragraph on usage of different relations in different perspectives can be answered in positive as well: since we have six relations specified—-being is_a, $ppart\_of$, $contained\_in$, $involved\_in$, $participates\_in$, and $member\_of$—the cross-granular relations between the entities in the adjacent levels adhere to any one of them.

Summarising the constraints on the relation RL (and its inverse RL^-) between granular levels, we have, in addition to parthood definition and restrictions:

X. *Definition 3.23*, *Theorem 3.5*: RL is a type of structural proper parthood, $s\_ppart\_of(x, y)$ and its inverse relation, $RL^-(y, x)$, a type of has proper part $has\_ppart(y, x)$, where x and y are distinct granular levels within the same granular perspective (T.1, T.2).

Y. *Theorem 3.6*: The multiplicity on the relata of $RL(x, y)$ ($RL^-(x, y)$) is 1:1 (T.3, T.4).

Z. *Lemma 3.16*: RL is acyclic and transitive (A.33, A.76).

Resulting from GR and the **nrG** extension, two constraints have to be added, which are the permitted types of granulation relations, and *Lemma 3.17*.

AA. *Definition 3.24*, *3.25*, *3.26*, *Proposition 3.19*: Permitted cross-granular relations among the entities (/types) populating the levels are is_a, $participates\_in$, $member\_of$, $ppart\_of$, $involved\_in$, and $contained\_in$, and their inverse relations (D.7-D.18, A.38-A.51, A.112, A.114).

AB. *Lemma 3.17*: The amount of granular levels in a granular perspective using $member\_of$ (for perspectives granulated according to **nacG**) or $participates\_in$ (for **nfG**) to relate contents between levels is exactly 2 (A.111, A.77, A.117, A.120).

AC. *Proposition 3.13*, *3.14*, *3.15*, *3.16*, *3.17*, *3.18*: various constraints particular to a type of granularity (A.115, A.116, A.118, A.119, A.121, A.122)

Having addressed the main relations in the TOG, we now turn to relations to relate granular perspectives and link levels across them.

3.7 Linking levels across perspectives

The last TOG components are the relations to link granular perspectives and to link levels across perspectives. Several related functions will be defined in Chapter 4, such as level selection and intersection, whereas here we focus on its prerequisites. There are two main approaches to consider: relating levels directly and relating them through relating their perspectives. We analyse both in this section.

3.7.1 The relation RP between perspectives

The main questions to answer regarding the binary RP relation between perspectives are: what are the characteristics of RP, and what additional knowledge can/does RP provide?

Using notions of contextual reasoning, we could interpret RP as *shifting* contexts. Although this is a useful conceptualisation, it faces similar problems as with RE, i.e., having to extend MCS considerably (see also §5.2.2). The 'more' with RP has to do with the change of criterion C and/or TG instead of looking only at the facts in the context boxes themselves; *i.e.*, RP captures a commonality for a set of bridge rules given two granular perspectives. A simple version of RP can be as follows.

DEFINITION 3.27 (RP). *RP relates two distinct perspectives:*
$\forall x, y(RP(x, y) \rightarrow GP(x) \wedge GP(y) \wedge \neg(x = y))$.

From this definition, it follows immediately that RP is irreflexive and symmetric.

LEMMA 3.18. *RP is irreflexive, $\neg RP(x, x)$, and symmetric, $RP(x, y) \leftrightarrow RP(y, x)$.*

Proof. Irreflexive: the "$\neg(x = y)$" in *Definition 3.27* and one or more (*Proposition 3.3*) unique perspectives (*Corollary 3.2*), therefore the relata can never be the same.
Symmetric: RP's distinct domain and range are both of type GP. □

One might want to refine the definition to also include a 'swapping' of criteria, but from previous results on properties of granular levels, it was shown that it is the combination of criterion and granulation what makes a perspective unique (*Theorem 3.1*), hence, shifting perspective already logically implies changing C or TG. Thus, a relation between perspectives within a domain suffices for the current scope, where the resultant of switching is that different properties of the granulated contents will be highlighted.

RP may not seem useful, but it is necessary when we need to link levels from different granular perspectives, whereby we can retrieve additional targeted information through using RP. This will be elaborated on in the next section.

3.7.2 Options for linking levels from different perspectives

We now have the basic machinery to address linking levels of different perspectives. Two strategies can be identified, which use either overcrossing levels with mereology or chaining levels through RL and RP; this is depicted in *Figure 3.7*.

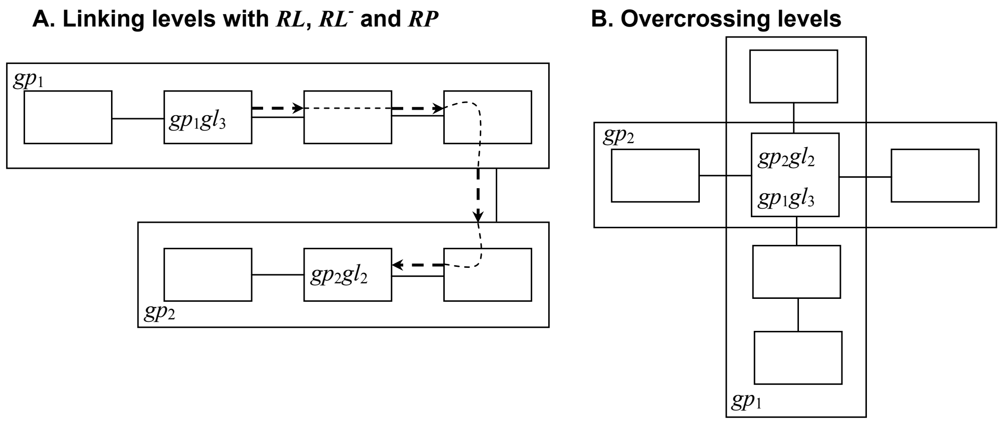


Figure 3.7: Connecting levels and perspectives with RL and RP (A) or overlap and overcross (B).

3.7.2.1 Overcrossing levels

The first option is to *overcross* levels, which means that the two levels are different, but they share at least some of their contents, which thus *overlap* (*Figure 3.7-B*). Overlap and overcross have their usual semantics based on Ground Mereology, as given in (3.23, 3.21), which can be put more precisely for the TOG as follows.

LEMMA 3.19. *Two levels in different perspectives can* overcross*:*
$\forall x, y(overcross(x, y) \land GL(x) \land GL(y) \land \neg(x = y) \rightarrow \exists v, w(RE(x, v) \land RE(y, w) \land \neg(v = w)))$

<u>*Proof.*</u> The proof goes in two steps: first the "$\neg part\_of(x, y)$" of $overcross$ is addressed, subsequently the "$overlap(x, y)$" part of $overcross$, where $overcross$ is defined as

$\forall x, y(overcross(x, y) \triangleq overlap(x, y) \land \neg part\_of(x, y))$ (see 3.23)

1. From typing RE (*Definition 3.21*), $GP(v)$ and $GP(w)$, with $\neg(v = w)$, therefore $\neg(x = y)$, because the combinations of criterion and granulation are distinct for the two levels (*Theorem 3.1*, *Lemma 3.4*); hence, $\neg part\_of(x, y)$ of (3.23) holds.
2. Demonstrate $overlap(x, y)$: this is defined as
   $\forall x, y(overlap(x, y) \triangleq \exists z(part\_of(z, x) \land part\_of(z, y)))$ (see 3.21)
   This applied to the axiom in the lemma implies that $GL(x)$ and $GL(y)$ must have a common part z. This is true if the *content* of a level stands in some part-whole relation to the frame that encloses the entities (/types) of D^s, because then the intersection of the contents of the two levels return the common part, which is z. Let the two sets with the levels'

contents be denoted with X and Y, then $X \cap Y = z$ and $z = \neg\emptyset$. Given that the entities (/types) are not structural parts of granular levels, it will have to be a type of *containment* for overlap to hold (note this is an inverse of the generic in_level (2.8)). The $contained\_in$ relation

$\forall x, y(contained\_in(x, y) \triangleq part\_of(x, y) \wedge V(x) \wedge V(y) \wedge$
$\exists z, w(has\_3D(z, x) \wedge has\_3D(w, y) \wedge ED(z) \wedge ED(w)))$ (see 3.14)

is a subrelation of $part\_of$ and satisfies this idea but not the relata, only $part\_of$ and $ppart\_of$ do. Given that the same entity (/type) can be in different levels and thus shared among ≥ 1 whole—but not in the same hierarchy—it has to be $part\_of$. Then, because $in\_level(x, y) \rightarrow part\_of(x, y)$ and $X \cap Y = z$, therefore $overlap(x, y)$ holds.

Both 1 and 2 return true, and thereby the levels can overcross. □

Given that two levels can overcross, we can extend it to perspectives, shown *Theorem 3.7*; for illustration, the Prover9-*computed* proof of *Theorem 3.7* is included in *Appendix B.3*.

THEOREM 3.7. *If two levels in different perspectives overcross, then their perspectives overcross:* $\forall x_1, x_2, y_1, y_2(overcross(x_1, x_2) \wedge GL(x) \wedge GP(y) \wedge RE(x_1, y_1) \wedge RE(x_2, y_2) \rightarrow overcross(y_1, y_2))$

Proof. Given that

$\forall x, y(RE(x, y) \rightarrow ppart\_of(x, y))$ (*Definition 3.21*)
$\forall x, y(ppart\_of(x, y) \rightarrow part\_of(x, y))$ (see 3.11)

the parthood relations are transitive in Ground Mereology, and so is RE (*Lemma 3.13*), then the overcross from *Lemma 3.19* implies the respective perspectives of the levels overcross. □

Thus, we can have, say, entity type A is granulated with criterion c_1 in gp_1 resulting in A' in some gp_1gl_i and is granulated with c_2 for gp_2, allocated to gp_2gl_j as A''. Overcrossing gp_1gl_i with gp_2gl_j ties A' to A'', providing the intersection where the properties of the entity type combine to represent the property-rich type A; hence, a richer representation of that entity than in their separate perspectives, which conforms to the analysis about properties (§3.3.1). The next example illustrates this for bacteriocins.

Example 3.8. Let d_i be the domain of Bacteriocins, which are non-therapeutical antibiotics used in food science and the food industry to improve food safety and preservation. Tn5301 is the gene encoding for the bacteriocin Nisin. Tn5301 is in level gp_1gl_3 at the Gene-level, and in level gp_2gl_2 of a location perspective, Tn5301 is in the Mobile DNA fragment-level and subsumed by the entity type Transposon. Overcrossing the two levels where the Tn5301s match says that gene Tn5301 is on a transposon. Thus, the overlap provides a richer description of Tn5301, because it combines more properties the entity type has than is represented with only one perspective. ◇

Although *Example 3.8* and *Figure 3.7-B* demonstrate overcross for two perspectives and levels, this can be any amount of relevant levels and perspectives. Without going into details of TOG functions here, overcrossing and linking perspectives will be demonstrated in *Example 3.9* using the already defined granulation of the infectious diseases domain to highlight issues on practically dealing with multiple perspectives. It can be structured better with the TOG components and relations and solved with the TOG functions (Chapter 4) so that one can deal with such situations in a consistent and scalable way.

Example 3.9. The $gran$ function takes only *one* argument—an entity type—and returns *one* value (the level), which was applied to granular levels for human structural anatomy, which served its purpose because there was only one granular perspective[18]. Expanding this scenario to the subject domain of human infectious diseases and multiple granular

[18]Recollect that Kumar *et al.*'s (2005) $gran$ function (see also §1.1.3) does not permit multiple perspectives and Bittner and Smith's (2003) theory granular partitions is ignorant about multiple perspectives within one domain.

perspectives (Keet and Kumar, 2005) reveals limitations of this approach. Of the nine perspectives defined for the infectious diseases domain (see *Appendix A*), let us take the human anatomy and the perspectives for an entity type's mode of action. Kumar *et al.*'s $gran$ function is used with *their notation*. First, assignment (1, 2), retrieval (4-5), and relevant levels (for brevity, in infix notation (8, 9)).

1: Vibrio cholerae → gran(Vibrio cholerae)
2: Cholera toxin → gran(Cholera toxin)
3: gran(Cholera toxin) = Inhibitor
4: gran(Cholera toxin) = Molecule
5: gran(Vibrio cholerae) = Cell
6: gran(Vibrio cholerae) = Organism
7: gran(Vibrio cholerae) = Toxin producer
8: Inhibitor part_of Toxin producer
9: Molecule part_of Organelle part_of Cell part_of Tissue part_of Organ part_of
Body system part_of Organism

Overcrossing levels, (3, 4) returns that Cholera toxin is structurally a type of Molecule and functionally a type of Inhibitor. (5-7) says that *V. cholerae* is a type of Organism that structurally consists of one cell and is also a Toxin producer. However, there are two problems. First, $gran$ returns only one level at a time and as such returns inconsistent or incomplete data upon repeated querying. Second, (5, 6) collapses the hierarchy of levels of the anatomy granular perspective: *V. cholerae* is *both* at the Cell-level *and* the Organism-level, which is due to the ambiguity between the structural viewpoint and the meaning of being an organism. It is not true that all cells are organisms or vice versa, but there is an organism that *consists* of one cell, *i.e.* an unicellular organism. The top level in the anatomy perspective is not organism, but a structural anatomical whole (being an organism has necessary conditions of having a metabolism and that it can self-reproduce).
An advantage of using the perspective a level is in, is that the same entity can be contained in levels in different perspectives: *V. cholerae* can be, correctly, of gp_1gl_i = Cell and gp_2gl_j = Organism, thereby avoiding inconsistencies in a software system and representing the different characteristics of an entity or the different views (perspectives) from where one can look at an entity.
An alternative *ad hoc* option for the $gran$ function is to *rename* it for each different perspective, such as (a, b)

a: gran-moa(Cholera toxin) = Inhibitor
b: gran-anat(Cholera toxin) = Molecule

However, this suggests that $gran\text{-}moa$ (for the mode of action) is a different function from $gran$, which it is not. This liberal naming leads to redundancy, is prone to inconsistency in labelling, and misses the point that the cholera toxin in both cases is the same entity viewed from another perspective. We can address this overcrossing levels and intersecting its contents straightforwardly with a one-off SQL query on a granulated database with a table for each level:

```
SELECT sgp.entity, sgp.glevel, fgp.glevel
       FROM structureGP sgp, functionGP fgp
       WHERE fgp.entity = sgp.entity
              AND sgp.entity = ''CholeraToxin'';
```

We revisit this example in *Example 4.2*, where we have the TOG's functions to solve in a structured and reusable way the type of problems illustrated here. ♢

Both examples 3.8 and 3.9 emphasise the properties of the entity types, and how this can be accommodated for with a simple application of the theory of granularity by overcrossing levels. An alternative methodology for relating levels across perspectives, is to rely primarily on the framework structure with its relations, which is described in the next section.

3.7.2.2 Linking levels

The second method to link two levels contained in different perspectives is shown in *Figure 3.7-A*. This 'long' path traces the connection between entity type A' in gp_1gl_3 through successive steps using rl_1 to the top-most level, subsequently uses rp_a that connects the two perspectives and going via rl_2 to gp_2gl_2 where A'' resides. This approach uses the TOG explicitly so that reasoning is made transparent and straightforward. Then, apart from retrieving combined knowledge about A, one can pick up information along the pathway, thereby retrieving more knowledge than the overcross-method, because it takes advantage of the theory of granularity to a greater extent. In case of the bacteriocin example, this includes, among other things, that the gene/transposon is contained in the bacterium *Lactobacillus lactis* NIZO R5.

Because overlap does not exclude linking and is entailed in the TOG with the predicates of Ground Mereology, both are included in the formalisations in §3.8. The first interpretation with overcrossing levels is ontologically more accurate than the second option, but the second one returns extra information during implementation, which may be useful.

Summarising the constraints on relations between granular perspectives, we have

AD. *Definition 3.27*, *Lemma 3.18*: The relata of RP must be distinct granular perspectives, and RP is irreflexive and symmetric (A.78, A.79, A.80).

AE. *Lemma 3.19*, *Theorem 3.7*: two levels in different perspectives can overcross, and then also their perspectives overcross (A.81, D.28, D.30, T.5).

3.8 Formal characterisation

The FOL preliminaries in §2.3.1 hold here, too, and several abbreviations and conventions are added, which is followed by the formal characterisation of the TOG in §3.8.2.

3.8.1 Conventions and abbreviations

Several components and conventions are taken from the DOLCE+ foundational ontology (Masolo *et al.*, 2004, 2003; Gangemi and Mika, 2003), which are described first. Subsequently, the abbreviations specific to the TOG are given, followed by other adopted conventions.

DOLCE conventions and components

- In certain cases the variables ranging over universals is syntactic sugar for a finite list of first-order axioms (see also Masolo *et al.*, 2003, p26 and Common Logic). In particular:
  - Variables ϕ, ψ,... range over a finite set $\prod$ of explicitly introduced individuals;
  - The subclass of $\prod$ considered for the TOG, called $\prod_X$, is the union of the universals taken from DOLCE and from the TOG, $\prod_X = \prod_{G_{\text{DOLCE}}} \cup \prod_{G_{\text{TOG}}}$, and is identified by means of the predicate $X : X(\phi)$ iff $\phi \in \prod_X$.
    - The finite set of universals used in the formal characterisation which are taken from DOLCE:
      $\prod_{G_{\text{DOLCE}}} = \{PT, AB, R, TR, T, PR, S, AR, Q, TQ, TL, PQ, SL, AQ, ED, PED, M, F, POB, APO, NAPO, NPED, NPOB, MOB, SOB, ASO, SAG, SC, NASO, AS, PD, EV, ACH, ACC, STV, ST, PRO, DF, CN, PT\}$ that can be used for $U(\phi)$ and for U and DOLCE's SD, OD, D, GD, and DJ, which are explained below.
    - The finite set of universals used with variables ranging over universals that used in the formal characterisation and are introduced in with the TOG are:

$\prod_{G_{\text{TOG}}} = \{TG, cG, nG, sG, sgG, saG, sgsG, sgrG, samG, saoG, nrG, nfG, naG, nasG, nacG, GR, is\_a, ppart\_of, participates\_in, member\_of, involved\_in, contained\_in\}$

– Existential quantifiers on universals correspond to $\bigvee_{\psi \in \prod}(\psi(x))$;
– Universal quantifiers on universals correspond to $\bigwedge_{\psi \in \prod}(\psi(x))$;

- $CN(x)$ stands for that *x* is a (social) CoNcept, where $CN(x) \rightarrow NASO(x)$ in DOLCE, $NASO$ is a Non-Agentive Social Object, and CN is an endurant. Further, a CN "(i)is not directly located in space and, in general, has no direct spatial qualities (ii) has no intentionality; (iii) depends on a community of intentional agents" (Masolo *et al.*, 2004);

- $DF(x, y)$ is shorthand for the relation $definedBy(x, y)$, and $DF(x, y) \rightarrow (CN(x) \wedge DS(y))$;

- $PT(x)$ where *x* can be any particular, *i.e.* one that is subsumed by PT in DOLCE;

- $DS(x)$ is a description, which is also a $NASO$ that uses at least one concept, $DS(x) \rightarrow \exists y(US(y, x))$, but does not have to define concepts and may be some complex combination of other concepts (or entities);

- $R(x)$ stand for x is a region, and $Q(x)$ has a quale in the region $R(x)$, then $ql(x, y) \rightarrow (\alpha R(x) \wedge \alpha Q(y))$ where ql is the quale, R the region and Q the quality and the modifier α that it is not just any type of region with any type of quality, but temporal quality TQ with temporal region TQ, PQ and PR for physical, and so forth; more specific axioms related to qualities, quales and regions can be found in (Masolo *et al.*, 2003). In the formalisation in §3.8.2, the $has\_value(x, y)$ and V for DOLCE's R is similar to "ql" and used to denote that a quality *x* has a quale (value) in the region *y* that may me at the type-level.

- DOLCE's participation relation PC (Dd63 in Masolo *et al.*, 2003) for constant participation with T for time interval, ED endurant, PD perdurant, and PRE being present, then
$PC(x, y, t) \rightarrow (ED(x) \wedge PD(y) \wedge T(t))$
$PC_C(x, y) \triangleq \exists t(PRE(y, t)) \wedge \forall t(PRE(y, t) \rightarrow PC(x, y, t))$
The participation relation $participates\_in$, which was introduced in the previous paragraphs and formalised in the next one, corresponds to PC_C, where an entity always participates in a perdurant.

- Dd1-Dd12 in DOLCE have useful definitions for the subsumption relation ((Dd6) corrected w.r.t. Dd6 in Masolo *et al.* (2003)), where ϕ and ψ are variables ranging on universals (but recollect that the first bullet point above still holds):

$$RG(\phi) \triangleq \Box\forall x(\phi(x) \rightarrow \Box\phi(x)) \tag{Dd1}$$

$$NEP(\phi) \triangleq \Box\exists x(\phi(x)) \tag{Dd2}$$

$$DJ(\phi, \psi) \triangleq \Box \neq \exists x(\phi(x) \wedge \psi(x)) \tag{Dd3}$$

$$SB(\phi, \psi) \triangleq \Box\forall x(\psi(x) \rightarrow \phi(x)) \tag{Dd4}$$

$$EQ(\phi, \psi) \triangleq SB(\phi, \psi) \wedge SB(\psi, \phi) \tag{Dd5}$$

$$PSB(\phi, \psi) \triangleq SB(\phi, \psi) \wedge \neg SB(\psi, \phi) \tag{Dd6}$$

$$L(\phi) \triangleq \Box\forall\psi(SB(\phi, \psi) \rightarrow EQ(\phi, \psi)) \tag{Dd7}$$

$$SBL(\phi, \psi) \triangleq SB(\phi, \psi) \wedge L(\psi) \tag{Dd8}$$

$$PSBL(\phi, \psi) \triangleq PSB(\phi, \psi) \wedge L(\psi) \tag{Dd9}$$

$$L_X(\phi) \triangleq X(\phi) \wedge \Box\forall\psi(SB(\phi, \psi) \wedge X(\psi) \rightarrow EQ(\phi, \psi)) \tag{Dd10}$$

$$SBL_X(\phi, \psi) \triangleq SB(\phi, \psi) \wedge L_X \tag{Dd11}$$

$$PSBL_X(\phi, \psi) \triangleq PSB(\phi, \psi) \wedge L_X \tag{Dd12}$$

- One-sided constant dependence with relevant definitions as (D.19-D.23) below.

Other conventions.

- The unique existential quantifier has the usual "!" after the $\exists$, which is shorthand for: $\exists!x\phi \leftrightarrow \exists y \forall x(\phi \leftrightarrow x = y)$;
- Numerical restrictions on quantifiers ranging over variable, are used in abbreviated form. For instance, $\forall^{\geq 2}x$ reads as "for all x where there are at least two particulars", *i.e.* it is the shorthand notation for $\forall x_1, ..., x_n(... \wedge n \geq 2 \wedge ...)$. Such particulars are subsequently indicated with subscripts $_i$, $_j$, $_1$, $_2$ etc., if relevant.
  A 1:*n* relationship between two different entities A and B , *i.e.* for $\forall y \exists^{\geq 1}x$, is:
  $\forall y(A(y) \rightarrow \exists x_1, ..., x_n(B(x_1) \wedge ... \wedge B(x_n)) \wedge (\neg(x_1 = x_2) \wedge ... \wedge \neg(x_1 = x_n) \wedge ... \wedge \neg(x_{n-1} = x_n)) \wedge n \geq 1 \wedge (relation(x_1, y) \wedge ... \wedge (relation(x_n, y))))$
  This can be abbreviated as $A(y) \rightarrow \exists^{\geq n}x(B(x) \wedge relation(x, y) \wedge n \geq 1)$ or with the integer instead of n in the quantification. The general cases for $\exists^{\leq n}x(\phi(x))$ and $\exists^{\geq n}x(\phi(x))$ are:
  - $\exists^{\leq n}x(\phi(x)) \equiv \forall x_1, ..., x_n, x_{n+1}(\phi(x_1) \wedge ... \wedge \phi(x_n) \wedge \phi(x_{n+1}) \rightarrow (x_1 = x_2) \vee ... \vee (x_1 = x_n) \vee (x_1 = x_{n+1}) \vee (x_2 = x_3) \vee ... \vee (x_2 = x_n) \vee (x_2 = x_{n+1}) \vee ... \vee (x_n = x_{n+1}))$
  - $\exists^{\geq n}x(\phi(x)) \equiv \exists x_1, ...x_n(\phi(x_1) \wedge ... \wedge \phi(x_n) \wedge \neg(x_1 = x_2) \wedge ... \wedge \neg(x_1 = x_n) \wedge \neg(x_2 = x_3) \wedge ... \wedge \neg(x_2 = x_n) \wedge ... \wedge \neg(x_{n-1} = x_n))$

  Last, the "=" in $\exists^{=n}x$ is shorthand notation for $\exists^{\geq n}x\exists^{\leq n}x$.
- A vector variable is indicated with $\bar{x}$ and stands for a list of variables: $\bar{x} \equiv x_1, ..., x_n$.
- An *exclusive-or* is denoted with an underlined $\vee$, as in $\underline{\vee}$.

Recollect that although the formalisation quantifies over particulars, one can let the variables range over universals, analogous to the approach taken for the structures of the contents of a granular level (see §2.3). This Second Order Logic formalisation is omitted for reasons of clarity, because the same definitions and constraints apply regardless if one wants to granulate data in a database or the types in an ontology of universals; this is demonstrated for $\bumpeq$ (D.24-D.27).

Abbreviations specific for the theory of granularity

- Universals (unary) in the granularity framework:
  - D is the domain;
  - D^f as the outer framework
  - D^s the subject domain to be granulated;
  - GP is granular perspective;
  - GL is the granular level of a granular perspective;
  - C for criterion used with a granular perspective;
  - TG type of granularity with Chapter 2's taxonomy;
  - GR granulation relation;
  - F as set of conversion functions;
  - $Prop$ for property, which is used for C.
- Binary and ternary relations in the granularity framework:
  - $RL(x, y)$ relates two adjacent levels in a granular perspective;
  - $RC(x, y)$ relates a granular perspective and its criterion;

- $RE(x, y)$ is the relation between two of the three components D, GP, and GL;
- $RP(x, y)$ relates two granular perspectives;
- $\Leftrightarrow$ infix notation for indistinguishability;
- $granulates(x, y)$ relation between D^f and D^s;
- $has\_granulation(x, \phi)$ between GP and TG;
- $adheres\_to(x, \phi)$ between GL and TG;
- $has\_permitted(\phi, \psi)$ granulation relation between **nrG** and GR;
- $uses\_GR(\phi, \psi)$ between TG and GR;
- $conv(x, \phi, \vartheta)$ between GL, an **sG** type and a function ϑ.
- $CP(x, y)$ between C and its $Prop$erties.
- ϑ stands for a function; in the TOG this denotes a function like multiplication.

3.8.2 Formal characterisation of granularity

3.8.2.1 Definitions

Framework components

$$\forall x(D^f(x) \triangleq \exists! y, z(DF(x, y) \wedge (D(x) \rightarrow CN(x)) \wedge granulates(x, z))) \tag{D.1}$$

$$\forall x(D^s(x) \triangleq \exists! y((D(x) \rightarrow (PT(x) \veebar U(x))) \wedge DF(x, y))) \tag{D.2}$$

$$\begin{aligned}\forall x(GP(x) \triangleq &\exists w, y, z, \phi(DF(x, y) \wedge RC(x, z) \wedge C(z) \wedge \\ &RE(x, w) \wedge has\_granulation(x, \phi)))\end{aligned} \tag{D.3}$$

$$\begin{aligned}\forall x(GL(x) \triangleq &\exists! v, w, y, z(DF(x, y) \wedge GP(w) \wedge RE(x, w) \wedge C(z) \wedge \\ &RC(w, z) \wedge R(v) \wedge has\_value(z, v)))\end{aligned} \tag{D.4}$$

Framework relations

$$\forall x, y(RL(x, y) \triangleq s\_ppart\_of(x, y) \wedge GL(x) \wedge GL(y) \wedge \neg(x = y)) \tag{T.1}$$

$$\forall x, y(RL^-(x, y) \triangleq has\_s\_ppart(x, y) \wedge GL(x) \wedge GL(y) \wedge \neg(x = y)) \tag{T.2}$$

$$\forall x, y(RE(x, y) \triangleq ppart\_of(x, y) \wedge ((GL(x) \wedge GP(y)) \veebar (GP(x) \wedge D^f(y)))) \tag{D.5}$$

$$\forall x, y(RE^-(x, y) \triangleq has\_s\_ppart(x, y) \wedge ((GP(x) \wedge GL(y)) \veebar (D^f(x) \wedge GP(y)))) \tag{D.6}$$

Granulation relations

$$\forall x, y(ppart\_of(x, y) \triangleq part\_of(x, y) \wedge \neg part\_of(y, x)) \tag{D.7}$$

$$\forall x, y(involved\_in(x, y) \triangleq ppart\_of(x, y) \wedge PD(x) \wedge PD(y)) \tag{D.8}$$

$$\begin{aligned}\forall x, y(contained\_in(x, y) \triangleq &ppart\_of(x, y) \wedge V(x) \wedge V(y) \wedge \\ &\exists z, w(has\_3D(z, x) \wedge has\_3D(w, y) \wedge ED(z) \wedge ED(w)))\end{aligned} \tag{D.9}$$

$$\forall x, y(participates\_in(x, y) \triangleq mpart\_of(x, y) \wedge ED(x) \wedge PD(y)) \tag{D.10}$$

$$\forall x, y(member\_of(x, y) \triangleq mpart\_of(x, y) \wedge (POB(x) \vee SOB(x)) \wedge SOB(y)) \tag{D.11}$$

$$\forall x, y(s\_ppart\_of(x, y) \triangleq ppart\_of(x, y) \wedge ED(x) \wedge ED(y)) \tag{D.12}$$

$$\forall x, y(has\_ppart(x, y) \triangleq has\_part(x, y) \wedge \neg has\_part(y, x)) \tag{D.13}$$

$$\forall x,y(involves(x,y) \triangleq has\_ppart(x,y) \wedge PD(x) \wedge PD(y)) \tag{D.14}$$

$$\forall x,y(contains(x,y) \triangleq has\_ppart(x,y) \wedge V(x) \wedge V(y) \wedge \exists z,w(has\_3D(z,x) \wedge has\_3D(w,y) \wedge ED(z) \wedge ED(w))) \tag{D.15}$$

$$\forall x,y(has\_participant(x,y) \triangleq has\_mpart(x,y) \wedge ED(y) \wedge PD(x)) \tag{D.16}$$

$$\forall x,y(has\_member(x,y) \triangleq has\_mpart(x,y) \wedge (POB(y) \vee SOB(y)) \wedge SOB(x) \tag{D.17}$$

$$\forall x,y(has\_s\_ppart(x,y) \triangleq has\_ppart(x,y) \wedge ED(x) \wedge ED(y)) \tag{D.18}$$

Other relations

$$\forall x,y(SD(x,y) \triangleq \Box(\exists t(PRE(x,t)) \wedge \forall t(PRE(x,t) \to PRE(y,t))) \tag{D.19}$$

$$\forall \phi,\psi(SD(\phi,\psi) \triangleq DJ(\phi,\psi) \wedge \Box\forall x(\phi(x) \to \exists y(\psi(y) \wedge SD(x,y)))) \tag{D.20}$$

$$\forall \phi,\psi(GD(\phi,\psi) \triangleq DJ(\phi,\psi) \wedge \Box(\forall x(\phi(x) \to \exists t(PRE(x,t)) \wedge \forall x,t((\phi(x) \wedge At(t) \wedge PRE(x,t)) \to \exists y(\psi(y) \wedge PRE(y,t)))))) \tag{D.21}$$

$$\forall \phi,\psi(De(\phi,\psi) \triangleq SD(\phi,\psi) \vee GD(\phi,\psi)) \tag{D.22}$$

$$\forall \phi,\psi(OD(\phi,\psi) \triangleq De(\phi,\psi) \wedge \neg De(\psi,\phi)) \tag{D.23}$$

$$\forall x,y(indistinguishable(x,y) \triangleq \exists z_i,z_j,v_x,v_y,v_w(in\_level(x,z_j) \wedge in\_level(y,z_j) \wedge RL(z_j,z_i) \wedge in\_level(w,z_i) \wedge PT(x) \wedge PT(y) \wedge PT(w) \wedge ((\phi(x,w) \wedge \phi(y,w) \wedge (\phi \to GR)) \vee (V(v) \wedge has\_value(x,v_x) \wedge has\_value(y,v_y) \wedge has\_value(w,v_w) \wedge v_x < v_w \wedge v_y < v_w)))) \tag{D.24}$$

$$\forall \phi,\psi(indistinguishable(\phi,\psi) \triangleq \exists z_i,z_j,v_x,v_y,v_w(in\_level(\phi,z_j) \wedge in\_level(\psi,z_j) \wedge RL(z_j,z_i) \wedge in\_level(\varphi,z_i) \wedge U(\phi) \wedge U(\psi) \wedge U(\varphi) ((\varsigma(\phi,\varphi) \wedge \varsigma(\psi,\varphi) \wedge (\varsigma \to GR)) \vee (V(v) \wedge has\_value(\phi,v_\phi) \wedge has\_value(\psi,v_\psi) \wedge has\_value(\varphi,v_\varphi) \wedge v_\phi < v_\varphi \wedge v_\psi < v_\varphi)))) \tag{D.25}$$

$$\forall x,y(\varphi\text{-}indistinguishable(x,y) \triangleq indistinguishable(x,y) \wedge \varphi(x,z) \wedge \varphi(y,z) \wedge (\varphi \to GR)) \tag{D.26}$$

$$\forall \phi,\psi(\varphi\text{-}indistinguishable(\phi,\psi) \triangleq indistinguishable(\phi,\psi) \wedge \varphi(\phi,\chi) \wedge \varphi(\psi,\chi) \wedge (\varphi \to GR)) \tag{D.27}$$

$$\forall x,y(overlap(x,y) \triangleq \exists z(part\_of(z,x) \wedge part\_of(z,y))) \tag{D.28}$$

$$\forall x,y(p\_overlap(x,y) \triangleq overcross(x,y) \wedge \neg overcross(y,x)) \tag{D.29}$$

$$\forall x,y(overcross(x,y) \triangleq overlap(x,y) \wedge \neg part\_of(x,y)) \tag{D.30}$$

Notes:

- (D.1-D.2): *Definition 3.1.* The description $DS(y)$ can be omitted from D.1 because this is implied by above-mentioned implication from DOLCE: $DF(x,y) \to (C(x) \wedge DS(y))$.
- (D.19-D.23): one-sided dependence from DOLCE (Dd69-Dd-73). DJ disjoint, PRE being present, and De *De*pendence (D in DOLCE, but label changed to avoid confusion with TOG's D); see also *Proposition 3.2.*
- (D.24-D.27): The name is written out here as "indistinguishable(x, y)", alternative notation $x \eqsim y$; recollect that (D.25) and (D.27) can be omitted.
- (D.29): Proper overlap.

3.8.2.2 Taxonomy of types of granularity

$$\forall x(sG(x) \rightarrow cG(x)) \tag{A.1}$$

$$\forall x(nG(x) \rightarrow cG(x)) \tag{A.2}$$

$$\forall x(sgG(x) \rightarrow sG(x)) \tag{A.3}$$

$$\forall x(saG(x) \rightarrow sG(x)) \tag{A.4}$$

$$\forall x(sgsG(x) \rightarrow sgG(x)) \tag{A.5}$$

$$\forall x(sgrG(x) \rightarrow sG(x)) \tag{A.6}$$

$$\forall x(samG(x) \rightarrow saG(x)) \tag{A.7}$$

$$\forall x(saoG(x) \rightarrow saG(x)) \tag{A.8}$$

$$\forall x(nrG(x) \rightarrow nG(x)) \tag{A.9}$$

$$\forall x(nfG(x) \rightarrow nG(x)) \tag{A.10}$$

$$\forall x(naG(x) \rightarrow nG(x)) \tag{A.11}$$

$$\forall x(nacG(x) \rightarrow naG(x)) \tag{A.12}$$

$$\forall x(nasG(x) \rightarrow naG(x)) \tag{A.13}$$

$$\forall x(sG(x) \rightarrow \neg nG(x)) \tag{A.14}$$

$$\forall x(sgG(x) \rightarrow \neg saG(x)) \tag{A.15}$$

$$\forall x(sgsG(x) \rightarrow \neg sgrG(x)) \tag{A.16}$$

$$\forall x(nrG(x) \rightarrow \neg nfG(x)) \tag{A.17}$$

$$\forall x(nrG(x) \rightarrow \neg naG(x)) \tag{A.18}$$

$$\forall x(nfG(x) \rightarrow \neg naG(x)) \tag{A.19}$$

$$\forall x(nacG(x) \rightarrow \neg nasG(x)) \tag{A.20}$$

$$\forall x(cG(x) \rightarrow sG(x) \vee nG(x)) \tag{A.21}$$

$$\forall x(sG(x) \rightarrow sgG(x) \vee saG(x)) \tag{A.22}$$

$$\forall x(sgG(x) \rightarrow sgsG(x) \vee sgrG(x)) \tag{A.23}$$

$$\forall x(saG(x) \rightarrow samG(x) \vee saoG(x)) \tag{A.24}$$

$$\forall x(nG(x) \rightarrow nrG(x) \vee nfG(x) \vee naG(x)) \tag{A.25}$$

$$\forall x(naG(x) \rightarrow nacG(x) \vee nasG(x)) \tag{A.26}$$

Notes:
- (A.1-A.26): taxonomy of types of granularity, disjoint complete subtypes; Chapter 2.

3.8.2.3 Constraints on relations

Parthood and proper parthood relations

$$\forall x\ part\_of(x, x) \tag{A.27}$$

$$\forall x, y, z((part\_of(x, y) \land part\_of(y, z)) \rightarrow part\_of(x, z)) \tag{A.28}$$

$$\forall x, y((part\_of(x, y) \land part\_of(y, x)) \rightarrow x = y) \tag{A.29}$$

$$\forall x\ \neg ppart\_of(x, x) \tag{A.30}$$

$$\forall x, y(ppart\_of(x, y) \rightarrow \neg ppart\_of(y, x)) \tag{A.31}$$

$$\forall x, y, z((ppart\_of(x, y) \land ppart\_of(y, z)) \rightarrow ppart\_of(x, z)) \tag{A.32}$$

$$\forall x(\neg\varphi(x, x) \land (\varphi \rightarrow RL)) \tag{A.33}$$

$$\forall x, y(part\_of(x, y) \leftrightarrow has\_part(y, x)) \tag{A.34}$$

$$\forall x, y(ppart\_of(x, y) \leftrightarrow has\_ppart(y, x)) \tag{A.35}$$

$$\forall x, y(s\_ppart\_of(x, y) \rightarrow ppart\_of(x, y)) \tag{A.36}$$

$$\forall x, y(has\_s\_ppart(x, y) \rightarrow has\_s\_ppart(x, y)) \tag{A.37}$$

Granulation relations

$$\forall x, y(contained\_in(x, y) \rightarrow ppart\_of(x, y)) \tag{A.38}$$

$$\forall x, y(contains(x, y) \leftrightarrow contained\_in(y, x)) \tag{A.39}$$

$$\forall x, y(involved\_in(x, y) \rightarrow ppart\_of(x, y)) \tag{A.40}$$

$$\forall x, y(involved\_in(x, y) \leftrightarrow involves(y, x)) \tag{A.41}$$

$$\forall x, y(participates\_in(x, y) \leftrightarrow has\_participant(y, x)) \tag{A.42}$$

$$\forall x, y(member\_of(x, y) \leftrightarrow has\_member(y, x)) \tag{A.43}$$

$$\forall x, y(ppart\_of(x, y) \rightarrow GR(x, y)) \tag{A.44}$$

$$\forall x, y(member\_of(x, y) \rightarrow GR(x, y)) \tag{A.45}$$

$$\forall x, y(participates\_in(x, y) \rightarrow GR(x, y)) \tag{A.46}$$

$$\forall x, y(is\_a(x, y) \rightarrow GR(x, y)) \tag{A.47}$$

$$\forall x, y(contained\_in(x, y) \rightarrow \neg involved\_in(x, y)) \tag{A.48}$$

$$\forall x, y(ppart\_of(x, y) \rightarrow \neg member\_of(x, y)) \tag{A.49}$$

$$\forall x, y(ppart\_of(x, y) \rightarrow \neg participates\_in(x, y)) \tag{A.50}$$

$$\forall x, y(member\_of(x, y) \rightarrow \neg participates\_in(x, y)) \tag{A.51}$$

Notes:

- (A.33): *Lemma 3.16.* φ is syntactic sugar for a cyclic path, and in the TOG a shorthand for: $\forall x_1 ... x_i ... x_n(\varphi(x_1, x_i) \triangleq (RL(x_1, x_2) \land ... \land RL(x_{n-1}, x_n) \land (1 \leq i \leq n) \rightarrow x_1 = x_i))$.

Other relations

$$\forall x(x\sigma x) \tag{A.52}$$

$$\forall x,y(x\sigma y \rightarrow y\sigma x) \tag{A.53}$$

$$\forall w,x,y,z(in\_level(x,w) \land in\_level(y,w) \land in\_level(z,w) \rightarrow (x\sigma y \land y\sigma z \rightarrow x\sigma z)) \tag{A.54}$$

$$\forall x,y(x \sim y) \tag{A.55}$$

$$\forall x,y(x \sim y \rightarrow y \sim x) \tag{A.56}$$

$$\forall x,y,z(x \sim y \land y \sim z \rightarrow x \sim z) \tag{A.57}$$

$$\forall x,y(x\sigma y \rightarrow \neg(x \Bumpeq y)) \tag{A.58}$$

$$\forall x,y(x \Bumpeq y \rightarrow x \sim y) \tag{A.59}$$

$$\forall x(x \Bumpeq x) \tag{A.60}$$

$$\forall x,y(x \Bumpeq y \leftrightarrow y \Bumpeq x) \tag{A.61}$$

$$\forall x,y,z(x \Bumpeq y \land y \Bumpeq z \rightarrow x \Bumpeq z) \tag{A.62}$$

$$\forall x,y(has\_value(x,y) \rightarrow Prop(x) \land V(y)) \tag{A.63}$$

$$\forall x(Q(x) \rightarrow \exists y(has\_value(x,y) \land V(y))) \tag{A.64}$$

$$\forall x,y(has\_value(x,y) \rightarrow \exists z(has\_value(z,y) \land C(z))) \tag{A.65}$$

$$\forall x,y(granulates(x,y) \rightarrow D^f(x) \land D^s(y)) \tag{A.66}$$

$$\forall x(D^f(x) \rightarrow \exists y\ granulates(x,y)) \tag{A.67}$$

$$\forall x,y,z(granulates(x,y) \land granulates(x,z) \rightarrow y = z) \tag{A.68}$$

$$\forall x,y(in\_level(x,y) \rightarrow (PT(x) \veebar U(x)) \land GL(y)) \tag{A.69}$$

$$\forall x,y(RE(x,y) \rightarrow ppart\_of(x,y)) \tag{A.70}$$

$$\forall x,y(RE^-(x,y) \rightarrow has\_ppart(x,y)) \tag{A.71}$$

$$\forall x,y,z(RE(x,y) \land RE(y,z) \land GL(x) \land GP(y) \land D^f(z) \rightarrow RE(x,z)) \tag{A.72}$$

$$\forall x,y,z(RE^-(x,y) \land RE^-(y,z) \land GL(z) \land GP(y) \land D^f(x) \rightarrow RE(x,z)) \tag{A.73}$$

$$\forall x(\neg\phi(x,x) \land (\phi \rightarrow RE)) \tag{A.74}$$

$$\forall x(\neg\psi(x,x) \land (\psi \rightarrow RE^-)) \tag{A.75}$$

$$\forall x,y,z(RL(x,y) \land RL(y,z) \rightarrow RL(x,z)) \tag{A.76}$$

$$\forall x\exists!y(RL(x,y)) \tag{T.3}$$

$$\forall x\exists!y(RL^-(x,y)) \tag{T.4}$$

$$\forall x,y,z(participates\_in(x,y) \land participates\_in(y,z) \rightarrow \neg participates\_in(x,z)) \tag{A.77}$$

$$\forall x,y(RP(x,y) \rightarrow GP(x) \land GP(y) \land \neg(x = y)) \tag{A.78}$$

$$\forall x\neg RP(x,x) \tag{A.79}$$

$$\forall x_i,x_j(RP(x_i,x_j) \rightarrow RP(x_j,x_i)) \tag{A.80}$$

$$\begin{aligned}&\forall x,y(overcross(x,y) \land GL(x) \land GL(y) \land \neg(x = y) \rightarrow \\ &\qquad \exists v,w(RE(x,v) \land RE(y,w) \land \neg(v = w)))\end{aligned} \tag{A.81}$$

$$\forall x_1, x_2, y_1, y_2(overcross(x_1, x_2) \land GL(x) \land GP(y) \land RE(x_1, y_1) \land RE(x_2, y_2) \rightarrow overcross(y_1, y_2)) \tag{T.5}$$

$$\forall x, \phi(has\_granulation(x, \phi) \rightarrow GP(x) \land TG(\phi)) \tag{A.82}$$

$$\forall x(GP(x) \rightarrow \exists! \phi\ has\_granulation(x, \phi)) \tag{A.83}$$

$$\forall x, \phi(adheres\_to(x, \phi) \rightarrow GL(x) \land TG(\phi)) \tag{A.84}$$

$$\forall x(GL(x) \rightarrow \exists \phi\ adheres\_to(x, \phi)) \tag{A.85}$$

$$\forall x, y(GP(y) \land GL(x) \land RE(x, y) \rightarrow \exists! \phi(has\_granulation(y, \phi) \leftrightarrow adheres\_to(x, \phi))) \tag{A.86}$$

$$\forall x, y(CP(x, y) \rightarrow C(x) \land P(y)) \tag{A.87}$$

$$\forall x(C(x) \rightarrow \exists^{\geq 2} y CP(x, y)) \tag{A.88}$$

Notes:

- (A.63): *Definition 3.5*, where V is DOLCE's Region.
- (A.69): Chapter 2 definition (2.8).
- (A.74-A.75): *Lemma 3.13.* ϕ and ψ are syntactic sugar for a cyclic path (analogous to (A.33)), and shorthand for, respectively:
  $\forall x_1 ... x_i ... x_n(\phi(x_1, x_i) \triangleq (RE(x_1, x_2) \land ... \land RE(x_{n-1}, x_n) \land (1 \leq i \leq n) \rightarrow x_1 = x_i))$ and
  $\forall x_1 ... x_i ... x_n(\psi(x_1, x_i) \triangleq (RE^{-}(x_1, x_2) \land ... \land RE^{-}(x_{n-1}, x_n) \land (1 \leq i \leq n) \rightarrow x_1 = x_i))$

3.8.2.4 Other constraints and characteristics

TOG *components*

$$\forall x(D(x) \rightarrow (D^f(x) \lor D^s(x))) \tag{A.89}$$

$$\forall x(D^f(x) \rightarrow D(x)) \tag{A.90}$$

$$\forall x(D^s(x) \rightarrow D(x)) \tag{A.91}$$

$$\forall x(CN(x) \rightarrow \exists! y DF(x, y)) \tag{A.92}$$

$$\forall x(D^s(x) \rightarrow PT(x) \veebar U(x)) \tag{A.93}$$

$$\forall x(D^f(x) \rightarrow CN(x)) \tag{A.94}$$

$$\forall x(GP(x) \rightarrow CN(x)) \tag{A.95}$$

$$\forall x(GL(x) \rightarrow CN(x)) \tag{A.96}$$

$$\forall x(D^f(x) \rightarrow \exists y\ granulates(x, y)) \tag{A.97}$$

$$\forall x, y((D^s(y) \rightarrow PED(y)) \land granulates(x, y) \rightarrow OD(D^f, D^s)) \tag{A.98}$$

$$\forall x((C(x) \rightarrow \exists^{\geq 2} y(Prop(y) \land \neg Q(y) \land CP(x, y))) \veebar (C(x) \rightarrow \exists y \exists! z(Prop(y) \land Q(z) \land \neg(y = z) \land CP(x, y) \land CP(x, z)))) \tag{A.99}$$

$$\forall x(Q(x) \rightarrow Prop(x)) \tag{A.100}$$

$$\forall x(CN(x) \rightarrow ED(x)) \tag{A.101}$$

$$\forall x(ED(x) \rightarrow PT(x)) \tag{A.102}$$

Notes:

- (A.92): a concept has one definition; taken from DOLCE.
- (A.100-A.102): DOLCE.

Perspectives and their criteria

$$\forall x, y(RC(x,y) \rightarrow GP(x) \wedge C(y)) \tag{A.103}$$

$$\forall x(GP(x) \rightarrow \exists! y(C(y) \wedge RC(x,y))) \tag{A.104}$$

$$\forall x(C(x) \rightarrow \exists y RC(x,y)) \tag{A.105}$$

$$\begin{split}\forall x_1, x_2, \phi_1, \phi_2, y(RC(x_1,y) \wedge RC(x_2,y) \wedge has\_granulation(x_1,\phi_1) \wedge \\ has\_granulation(x_2,\phi_2) \wedge \neg(x_1 = x_2) \rightarrow \neg(\phi_1 = \phi_2))\end{split} \tag{A.106}$$

$$\forall x(GP(x) \rightarrow \exists! y, \phi(RC(x,y) \wedge has\_granulation(x,\phi))) \tag{T.6}$$

$$\begin{split}\forall x \exists z, \phi((C(x) \rightarrow \exists! y(CP(x,y) \wedge Q(y))) \wedge RC(z,x) \wedge \\ has\_granulation(z,\phi) \rightarrow (\phi \rightarrow sG))\end{split} \tag{A.107}$$

$$\forall x, y, z((RC(x,y) \wedge GP(x) \wedge C(y) \wedge RE(z,x) \wedge GL(z)) \rightarrow RC(z,y)) \tag{A.108}$$

Constraints on components contained in another

$$\forall x(GP(x) \rightarrow \exists^{\geq 2} y(RE^{-}(x,y) \wedge GL(y))) \tag{T.7}$$

$$\forall x(GL(x) \rightarrow \exists! y(RE(x,y) \wedge GP(y))) \tag{T.8}$$

$$\forall x(D^f(x) \rightarrow \exists y(RE^{-}(x,y) \wedge GP(y))) \tag{A.109}$$

$$\forall x(GP(x) \rightarrow \exists! y(RE(x,y) \wedge D^f(y))) \tag{A.110}$$

$$\begin{split}\forall x, \psi, \phi(has\_granulation(x,\psi) \wedge GP(x) \wedge nrG(\psi) \wedge has\_permitted(\psi,\phi) \wedge \\ (\phi \equiv participates\_in \vee \phi \equiv member\_of) \rightarrow \exists^{=2} z(GL(z) \wedge RE^{-}(x,z)))\end{split} \tag{A.111}$$

Notes:

- (A.109): *Proposition 3.3*. *Corollary 3.3* ($\forall x_1, ..., x_n, y(GL(x) \wedge GP(y) \wedge RE(x,y) \rightarrow \neg(x_1 = x_2) \wedge ... \wedge \neg(x_{n-1} = x_n))$) and *Proposition 3.9*.
- (A.110): *Proposition 3.3*, which entails *Corollary 3.2*'s $\forall x_1, ..., x_n, y(GP(x) \wedge D^f(y) \wedge RE(x,y) \rightarrow \neg(x_1 = x_2) \wedge ... \wedge \neg(x_{n-1} = x_n))$ (or: $RE(x_i,y) \wedge RE(x_j,y) \wedge D(y) \rightarrow \neg(x_i = x_j))$)).

Further constraints for different types of granularity subsumed by ***cG***

$$\forall \phi, \psi(uses\_GR(\phi,\psi) \rightarrow TG(\phi) \wedge GR(\psi)) \tag{A.112}$$

$$\forall x(cG(x) \rightarrow TG(x)) \tag{A.113}$$

$$\forall \phi, \psi(has\_permitted(\phi,\psi) \rightarrow TG(\phi) \wedge (\phi \rightarrow nrG) \wedge GR(\psi)) \tag{A.114}$$

nrG:

$$\forall \phi((\phi \rightarrow nrG) \rightarrow \exists! \psi\ has\_permitted(\phi,\psi)) \tag{A.115}$$

nfG:

$$\forall \phi((\phi \rightarrow nfG) \rightarrow \exists^{\geq 2} \psi(uses\_GR(\phi,\psi) \wedge \neg(\psi_i = \psi_j))) \tag{A.116}$$

$$\begin{split}\forall \phi, \psi(uses\_GR(\phi,\psi) \wedge (\phi \rightarrow nfG) \wedge (\psi \rightarrow participates\_in)) \rightarrow \\ \forall x \exists^{=2} y(GP(x) \wedge GL(y) \wedge RE(y,x))\end{split} \tag{A.117}$$

nacG:

$$\forall \phi, \psi, \varphi, x_i, x_j((\phi \rightarrow nacG) \wedge adheres\_to(x_i, \phi) \wedge adheres\_to(x_j, \phi) \wedge RL(x_j, x_i) \wedge in\_level(\psi, x_i) \wedge in\_level(\varphi, x_j) \rightarrow (\varphi \rightarrow \neg\psi)) \tag{A.118}$$

$$\forall \phi, \psi(uses\_GR(\phi, \psi) \wedge (\phi \rightarrow nacG) \rightarrow (\psi = member\_of)) \tag{A.119}$$

$$\forall \phi(\phi \rightarrow member\_of) \rightarrow \forall x \exists^{=2} y(GP(x) \wedge GL(y) \wedge RE(y, x)) \tag{A.120}$$

nasG:

$$\forall \phi, \psi((\phi \rightarrow nasG) \rightarrow uses\_GR(\phi, \psi) \wedge (\psi = is\_a)) \tag{A.121}$$

$$\forall \phi, x_i, x_j, y, z((\phi \rightarrow nasG) \wedge adheres\_to(x_i, \phi) \wedge adheres\_to(x_j, \phi) \wedge RL(x_j, x_i) \wedge in\_level(y, x_j) \wedge in\_level(z, x_i) \rightarrow (PT(y) \rightarrow PT(z))) \tag{A.122}$$

sG and its subtypes

$$\forall x, \phi, \vartheta(conv(x, \phi, \vartheta) \rightarrow GL(x) \wedge (\phi \rightarrow sG) \wedge F(\vartheta)) \tag{A.123}$$

$$\forall x, \phi(adheres\_to(x, \phi) \wedge (\phi \rightarrow sG) \rightarrow \exists \vartheta \; conv(x, \phi, \vartheta)) \tag{A.124}$$

$$\forall x(GL(x) \rightarrow \exists \phi \exists^{\leq 2} \vartheta \; conv(x, \phi, \vartheta)) \tag{T.9}$$

$$\forall z, \phi((\phi \rightarrow sG) \wedge GP(z) \wedge has\_granulation(z, \phi) \rightarrow \forall x \exists^{\geq 2} v, w((C(x) \wedge RC(z, x) \wedge GL(w) \wedge RE(w_i, z) \wedge RE(w_j, z) \wedge has\_value(x_i, v_i) \wedge has\_value(x_j, v_j) \wedge R(v) \wedge (RL(w_i, w_j) \wedge RL^{-}(w_j, w_i)) \rightarrow (v_j > v_i))) \tag{A.125}$$

Notes:
- (A.113): For linking the taxonomy of types of granularity to the other parts of the TOG.

3.8.3 Consistency and satisfiability of the TOG

Recollecting the FOL basics and model-theoretic semantics from §2.3.1, the set of sentences Γ for the TOG comprises (D.1-A.125) in the previous section, which may by a consistent theory in the sense of *Definition 2.10* and *2.11* and satisfiable (*Definition 2.13*) if there is an interpretation (model) of the TOG. This can checked manually, which, given the considerable size of the theory, is not a trivial task, and may be done computationally. To verify the formal characterization of the TOG computationally, it has been examined with the Mace4 model searcher, which searches for finite models and counterexamples by transforming the original problem to ground clauses with equality and subsequently uses a special-purpose decision procedure and the least-number heuristic optimization procedure to find a model (Mace4 & Prover9, 2007). For a given finite domain size, checking for the existence of a model is decidable. *Appendix B.1* contains the *satisfiable* input file of the TOG and Mace4 still terminates with a model up to domain size 8, but larger domain sizes run into memory limitation of the test PC (HP desktop, 3GHz Pentium processor, 1GB of RAM) or do not terminate within a reasonable amount of time. The statistics of the output up to domain size 10 is included in *Appendix B.2* and the results of domain sizes against CPU time are shown in *Figure 3.8*, which fit well with an exponential trendline.

On close reading of the input file, one will find that some adjustments had to be made to get a FOL representation of the TOG without abbreviations and conventions, which are summarised here. First, Mace4, and its related theorem prover Prover9, are a FOL model searcher and theorem prover, but we need second order for representing acyclicity of RL, therefore it has been hard-coded for up to three levels that cannot loop back. Also, for any universal $U(x)$, it was brought down to the instance-level and an axiom added to assert disjointness of PT and U, so it

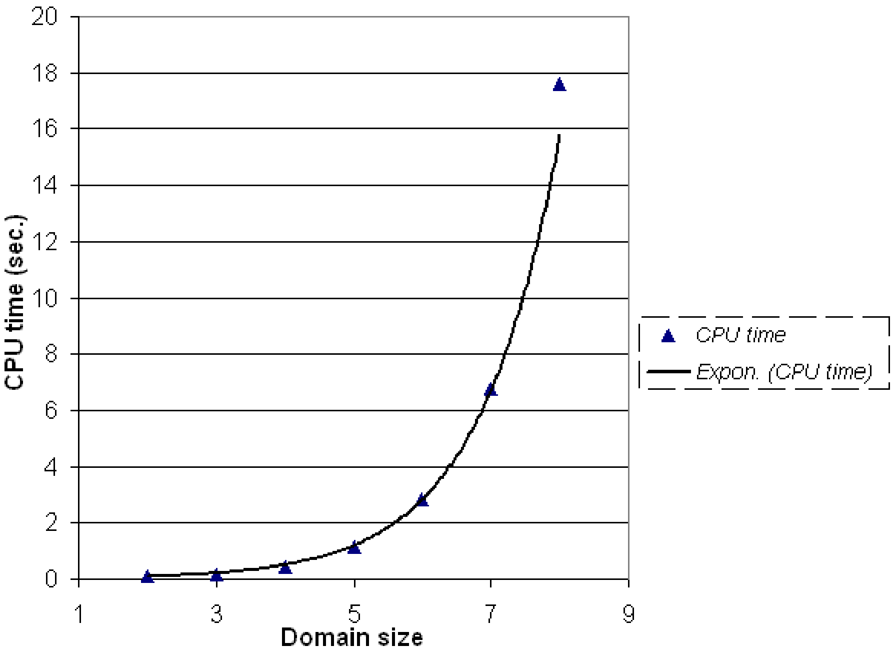


Figure 3.8: Domain size against the CPU time needed by Mace4 to terminate with a model of the TOG, and its trendline of type exponential.

covers instances only. This does not pose a problem, for if there is a model where $\Delta^{\mathcal{I}}$ consists of instances, then so will there be one when one takes the TOG as meta-theory with types/concepts as 'instances' because each real instance can be nominalised into a singleton set[19]. Second, the indistinguishability relation is a primitive relation in the Mace input file so that the reasoner does terminate in a reasonable amount of time, because inclusion of the indistinguishability definition causes the reasoner to take 'a lot' of time even for domain size 2 ($\geq$ 50 hours with 50% CPU load) and the process was killed without terminating with either a model or an inconsistency. Third, the syntactic sugar has been written out, including the reified GR and as binary relation $GR1$ and the counting variables ($\exists^{=2}x$, $\exists!y$ etc.); these differences do not change the TOG.

Developing a *tractable* computational implementation of the TOG is an aspect of future work. Directions for such effective computational implementation will be discussed in sections §5.5.1, which covers the main theoretical issues, such as the complexity of the language required to represent the TOG, and §5.5.2, which highlights several implementation trade-offs.

3.9 Granularity in a theory of granularity

Granularity and the TOG can be defined with its own formalisation in two ways:

1. The taxonomic structure of §2.2 is of the type **nasG**: a granularity in the types of granularity. Granularity is then the domain, granulation by criterion c_1 as structural foundational semantics, with TG **nasG** using as GR the is_a relation. This gp_1 contains four levels, being a top-level with **cG**, the second level with **sG** and **nG**, and so forth, using "$\leftarrow$" as assignment of the meta component to its corresponding domain element and "$\Leftarrow$" to denote

[19]The granulation relations (D.7-D.17) are defined for instances, but can be applied to types by using the "all-some" construction, *e.g.*, $\forall x \exists y\ part\_of(x, y)$ where $U(x)$ and $U(y)$ (see also Artale *et al.*, 1996a; Smith *et al.*, 2005).

contents in that level:

$$d_1 \leftarrow Granularity \tag{3.1}$$
$$gp_1 \leftarrow TG \tag{3.2}$$
$$gp_1gl_1 \Leftarrow \{\mathbf{cG}\} \tag{3.3}$$
$$gp_1gl_2 \Leftarrow \{\mathbf{sG}, \mathbf{nG}\} \tag{3.4}$$
$$gp_1gl_3 \Leftarrow \{\mathbf{sgG}, \mathbf{saG}, \mathbf{nrG}, \mathbf{nfG}, \mathbf{naG}\} \tag{3.5}$$
$$gp_1gl_4 \Leftarrow \{\mathbf{saoG}, \mathbf{samG}, \mathbf{sgrG}, \mathbf{sgpG}, \mathbf{nacG}, \mathbf{nasG}\} \tag{3.6}$$

2. Again taking granularity as domain, we can granulate on the TOG components where the framework components themselves define three levels in a perspective: D^f as top-level, GP as the second level, and GL third—their contents being instances of granularity domain frameworks, perspectives, and levels. This granulation is of the type **nrG**, using $GR = ppart\_of$ as generalisation of RE, and the criterion as TOG framework structural components. More precisely, one can write the subject domain-independent TOG in terms of domain granularity:

$$d_1 \leftarrow Granularity \tag{3.7}$$
$$gp_2 \leftarrow \text{TOG} framework\ structural\ components \tag{3.8}$$
$$gp_2gl_1 \leftarrow D^f \tag{3.9}$$
$$gp_2gl_2 \leftarrow GP \tag{3.10}$$
$$gp_2gl_3 \leftarrow GL \tag{3.11}$$

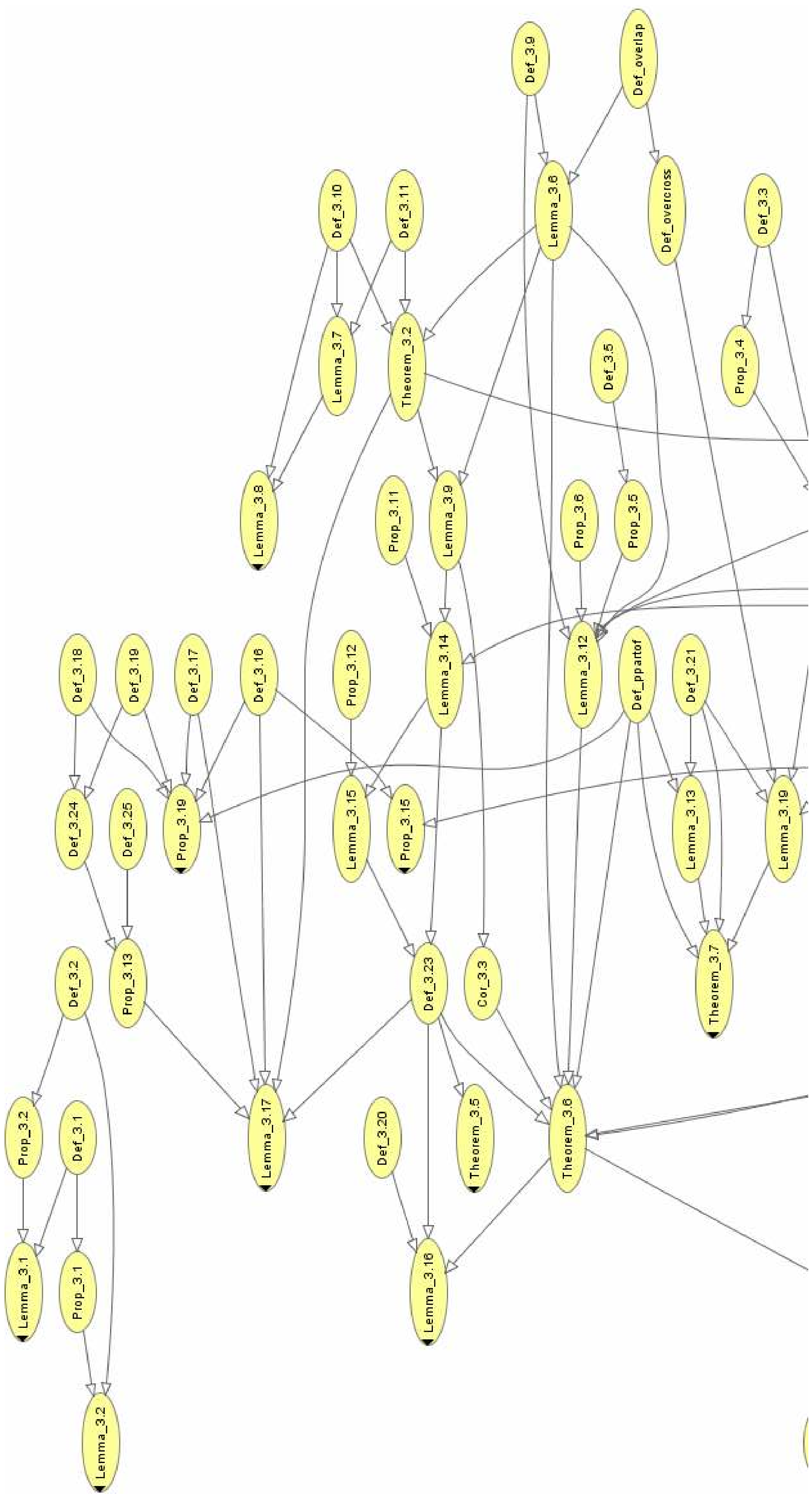
The gp_1 for TG implicitly demonstrates the ease of extension of the taxonomy of types of granularity: either one adds more contents or a new, finer-grained, level. The second perspective, gp_2 is particularly interesting because it indicates feasibility of meta-level specification for, among others, CASE tools, UML stereotypes, and 'punning' with OWL 1.1.

3.10 Chapter summary

The static components of the TOG were analysed from both an ontological and logical viewpoint, formalised, and their constraints proven where possible, resulting in 27 definitions, 19 propositions, 19 lemmas, 7 theorems, and 6 corollaries; their interdependencies are depicted in *Figure 3.9*. We moved from a data-centric treatment of granularity to the conceptual and logical layers, where the components of granularity have become ontologically-motivated modelling constructs. The TOG is formally characterised in FOL, reuses several categories from DOLCE foundational ontology, and with its computational version it has been shown to be a consistent and satisfiable logical theory, thereby ensuring unambiguous semantics and providing a robust and reusable framework for design and implementation of granularity. Theoretical aspects of the theory of granularity were illustrated and demonstrated with several examples taken from the biology subject domain. The TOG meets all key requirements.

3.10.1 Meeting the requirements

The TOG satisfies all requirements set out in §3.1, which is demonstrated here for each individual requirement. Several additional features are listed afterward.


Def_3.9
Def_overlap
Lemma_3.6
Def_overcross
Def_3.3
Def_3.10
Def_3.11
Lemma_3.7
Theorem_3.2
Def_3.5
Prop_3.4
Lemma_3.8
Prop_3.11
Lemma_3.9
Prop_3.6
Prop_3.5
Def_3.18
Def_3.19
Def_3.17
Def_3.16
Prop_3.12
Lemma_3.14
Lemma_3.12
Def_ppartof
Def_3.21
Def_3.24
Def_3.25
Prop_3.19
Lemma_3.15
Prop_3.15
Lemma_3.13
Lemma_3.19
Def_3.2
Prop_3.13
Def_3.23
Cor_3.3
Theorem_3.7
Prop_3.2
Def_3.1
Lemma_3.17
Def_3.20
Theorem_3.5
Theorem_3.6
Lemma_3.1
Prop_3.1
Lemma_3.16
Lemma_3.2

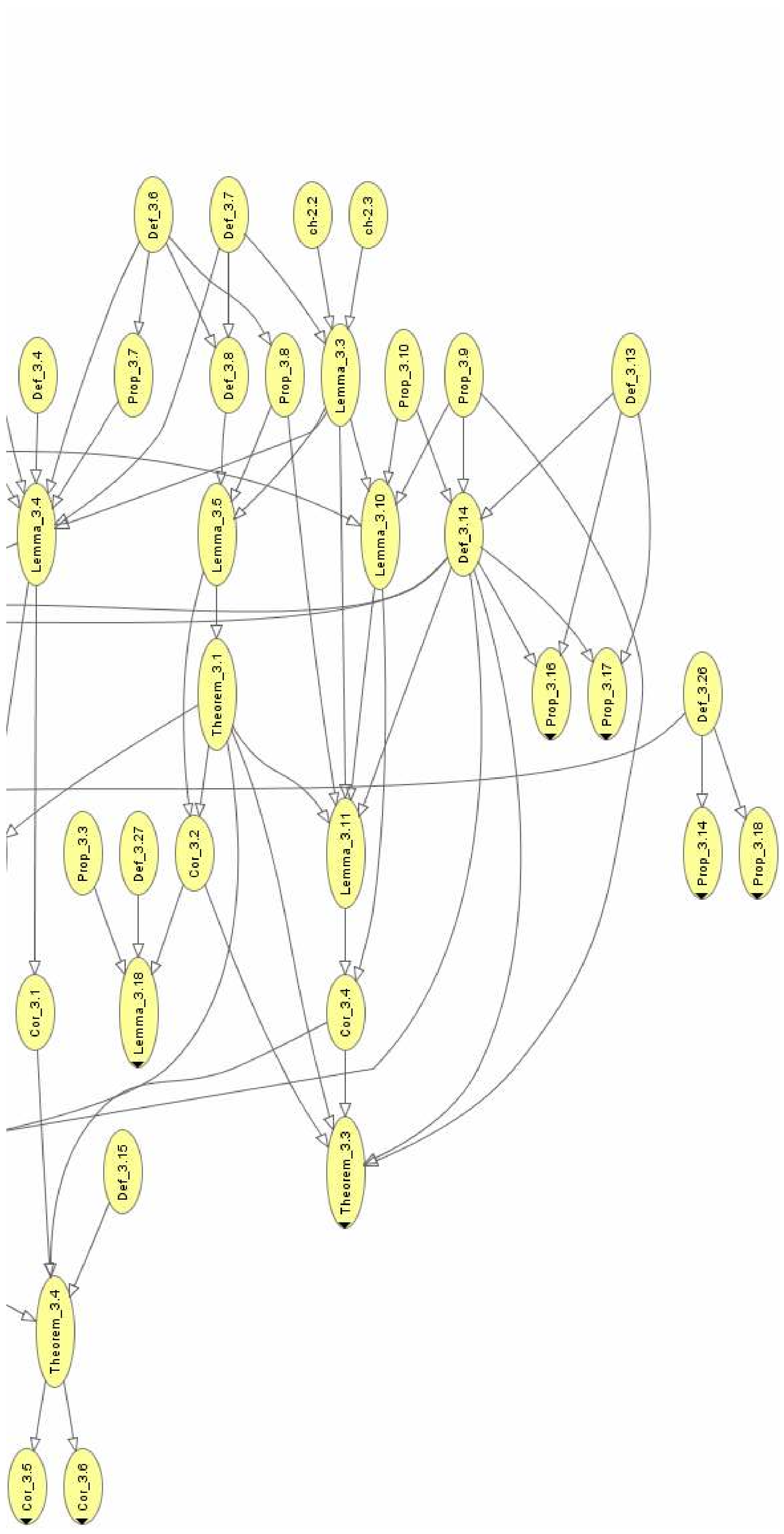


Figure 3.9: Interdependencies between the definitions, propositions, lemmas, theorems, and corollaries of the TOG.

1. *A theory of granularity should be usable eventually in a format for contents at the instance level and in a format for defining a domain granularity framework at the type level.*
This is met by the definition of the domain (*Definition 3.1* and (D.2, A.93)), which states that the contents may be either instances or types. Except for the granulation relations, the axioms are either independent of the actual contents with which the granularity framework can be populated or use PT and U explicitly for particulars and universals. Thus, the TOG is usable for both instance-level and type-level contents, thereby greatly widening the scope of usability and reusability among diverse application scenarios.

2. *A higher level simplifies, makes indistinguishable, the finer-grained details that are indistinguishable at that higher level.*
This is met with indistinguishability (§3.4.1: *Definition 3.11* and (D.24-D.27), and *Lemma 3.6*, *Lemma 3.7*, *Lemma 3.8* and (A.58-A.62)), and is part of the definition of Granular Level (§3.4.2: *Definition 3.14* and (D.4, A.96, A.108)).

3. *Following from the indistinguishability requirement, there have to be at least two levels within a perspective, else there is no granularity.*
This has been proven in *Theorem 3.2* (T.7).

4. *Any theory must be able to accommodate both the quantitative and qualitative aspects of granularity (or: arbitrary scale and non-scale-dependent granularity).*
This distinction was already made in the taxonomy of types of granularity, where **sG** and **nG** are distinguished, yet subsumed by the common **cG**. This distinction is propagated to the TOG with the function ϑ associated with levels that adhere to **sG** types of granularity, composition of the perspective's criterion with or without quality property Q, the permitted relations with **nrG**, and further specifications for **nG** subtypes (*Definition 3.26*, *Definition 3.25*, *Proposition 3.14*, *Lemma 3.17*, *Proposition 3.15*, *Proposition 3.16*, *Proposition 3.18*, *Proposition 3.17*, *Definition 3.15*, *Corollary 3.5*, *Corollary 3.6*, *Lemma 3.12* and (A.112-A.125), *Definition 3.3* and (A.99)).

5. *The logic-based representation has to permit the two main ways for perceiving and representing granularity, being set theoretical and mereological, therefore the relations between the entities (/types) contained in the levels and the relations between granular levels are, at least, either of the type is_a or $part\_of$.*
This is reflected in RL between the granular levels (s_ppart_of, see *Definition 3.23* and (T.1, T.2)) and the permitted granulation relations to link contents of levels, which includes both is_a and $ppart\_of$, as well as subtypes of the part-whole relation $contained\_in$, $involved\_in$, $participates\_in$, and $member\_of$ (D.7-D.17, A.27-A.51).

6. *Given that one granulates according to a certain type of granularity, this also means that there has to be one type of relation between granular levels within a particular granular perspective.*
This has been demonstrated with *Theorem 3.5* (and *Definition 3.23*, *Theorem 3.2*, and *Lemma 3.9* and (T.1, T.2, T.7)).

7. *For ontological correctness and computation, the type of relation between adjacent levels in a perspective has to be transitive for those perspectives that contain >2 levels.*
The relation between granular levels, RL, is a type of mereological proper parthood, hence is transitive, is acyclic as well (*Definition 3.23*, *Lemma 3.16* and (T.1, T.2, A.30-A.33)), and has 1:1 multiplicity (*Theorem 3.6* and (T.3-T.4)).

8. *A type of relation that relates contents in levels of granularity within a particular perspective can have the property of being intransitive, provided there are always exactly 2 levels in any given perspective that contains that type of relation relating the entities (/types);*

Demonstrated in *Lemma 3.17*, which applies in particular to **nfG** and **nacG** (A.117, A.120).

9. *The entities or entity types in a particular granular level have at least one aspect in common, which is a criterion by which to granulate the data, information, or knowledge.*
This is met by *Definition 3.3* and (A.99) and it related constraints, such as that exactly one criterion is related to a granular perspective through the RC relation (*Definition 3.6*), upward distributivity of the values (*Proposition 3.5*, 3.6), that its values are reused for the granular levels, and that a criterion can be reused for different perspectives provided it is used with a different type of granularity, (A.103, A.104, A.105, A.106, T.6, A.107, A.108, A.125)

10. *An entity (/type) never can reside in more than one granular level within the same granular perspective.*
Bearing in mind the clarification about the most specific universal (*Definition 3.12* in §3.4.1) for where GR is is_a ("An entity a or A where $minst(a, A)$ holds never can reside in more than one granular level within the same perspective"), this is entailed in the results of requirement 3.

11. *Given that each level is contained in a granular perspective and granular perspectives contained in a domain granularity framework are disjoint, an entity (/type) in a particular granular level may reside in* ≥ 1 *levels, provided that each level the entity (/type) is contained in a distinct granular perspective.*
With the granular perspective, through the characterisation of the criterion (A.99, A.104, A.105, A.106, A.108, A.65, A.125), one highlights less or equal the amount of properties a given entity (/type) has and allocates the entity (/type) into the appropriate level given the value(s) of the property (§3.3), therefore, it is possible that another property of that entity (/type) is used for allocating that entity (/type) in another granular perspective using a 'remaining' property that matches the criterion of the other perspective.

12. *If there is more than one granular perspective for a subject domain, these perspectives must have some relation among each other.*
This requirement is met with the RP relation (*Definition 3.27*, *Lemma 3.18* and (A.78,A.79, A.80)) and the mereological $overcross$ relation (*Theorem 3.7* and (A.81, D.28, D.30, T.5)).

In addition, there are other features of the TOG, such as its usage of DOLCE foundational ontology and the common Ground Mereology, which is entailed in the mereological theory GEM that DOLCE uses. The TOG also meets desired feature C—provided an entity (/type) is not an orphan in the original data source and the subject domain is covered fully with granular perspectives, it must reside in at least one granular level—in two ways. First, D^f *must* be related to D^s through the $granulates$ relation (*Definition 3.2*, *Proposition 3.2*, *Lemma 3.1*, *Lemma 3.2* and (A.98, A.97, A.67, A.66, D.19-D.23, A.68)). Second, the granular perspective and granular level each have a relation to TG ($has\_granulation$ and $adheres\_to$, in *Definition 3.7*, *Lemma 3.3*, *Definition 3.13*, *Lemma 3.11* and (A.82, A.83 A.84, A.85, A.86)) so that the coverage requirement can be met thanks to the formalised structures of the contents of different types of granularity (see §2.3 and *Figure 3.1*); hence, the entity (/type) is granulated. The remaining desired features will be addressed in the next chapters.

With the TOG, we now can proceed to functions for granular reasoning—querying and abstractions—in order to solve problems regarding dynamic aspects of granularity, and to assess several reasoning scenarios.

Part III

Toward using the Theory of Granularity

Chapter 4

Granular reasoning for applied domain granularity

4.1 Introduction

The TOG that was introduced in Chapter 3 provides a framework that one can use as additional layer for representing the granular aspects of domain knowledge. This can be declared on top of the usual databases, knowledge bases, and ontologies, but does not give any means to take advantage of this framework and to use the added-value to query both the framework components and the granulated data residing in the levels. To enable this kind of cross-granular querying and reasoning, functions will be introduced that cover this dynamic component for the TOG, thereby addressing research question 4—What reasoning tasks can benefit from granularity, and where and how will this affect the following types of tasks?—and its sub-questions (4a) and (4b). This can be formulated into four requirements, a desirable feature carried over from Chapter 3, and four tasks. The requirements are:

1. A set of functions for the TOG with which one can, at least,
   - Query the main TOG components—domain, perspective, and level—directly;
   - Retrieve content of levels;
   - Move between levels from coarser to finer-grained entities (/types) and back.
2. Content retrieval has to make use of the existing structure in the original data source and consider the type of granularity that is used for the granulation.
3. Abstraction functions have to enable moving from finer-grained entities (/types) to a coarser-grained simplification, which may reside in an adjacent coarser granular level or higher up in a level in the same perspective.
4. Expansion functions have to enable moving from coarser-grained entities (/types) to finer-grained detail, which may reside in an adjacent finer granular level or lower down in a level in the same perspective.
5. Desirable feature D: The purposes of the TOG's usage—dynamic aspects with primary distinctions allocating/classifying entities (/types) and information retrieval and reasoning—shall not affect the static structure of a theory of granularity.

To formulate the tasks concisely and demarcate the scope of this chapter, two definitions are introduced first, which are the data source $\mathcal{DS}$ and applied domain granularity $\mathcal{DG}$.

DEFINITION 4.1 ($\mathcal{DS}$). *A data source $\mathcal{DS}$ is a type of D^s that provides the representation and storage of the subject domain in either a database, knowledge base, or ontology.*

DEFINITION 4.2 ($\mathcal{DG}$). *An applied domain granularity framework $\mathcal{DG}$ is the combination of a $\mathcal{DS}$ with an instantiation of a granularity framework that satisfies the constraints of the* TOG.

The four tasks that will be solved in this chapter are:

Task 1. Perform a selection of levels from a particular domain granularity framework d_i^f, *i.e.* $task^1(d_i^f) \rightarrow lss_i$, where $\mathcal{DL}$ is the set of levels within that domain granularity framework and lss_i the selected subset ($lss_i \subseteq \mathcal{DL}$).

Task 2. To retrieve contents of at least one level, we have $task^2(gl_i) \rightarrow \mathcal{E}$, where $\mathcal{E}$ denotes the contents of a granular level, that is, entities or types and, where applicable, any further structure among the entities (/types) other than an unordered set.

Task 3. A formal relationship or transformation rule is required between $\mathcal{DS}$ and $\mathcal{DG}$ to utilise them both for some particular reasoning task. Therefore, $task^3(\mathcal{DS}, \mathcal{DG}) \rightarrow \mathcal{DS}\ related\_to\ \mathcal{DG}$, where the "$related\_to$" has to be specified. Likewise, the relation between $\mathcal{DS}$ and selected subsets ($\mathcal{DG}^1$, ..., $\mathcal{DG}^n$) has to be specified.

Task 4. Granular information retrieval by using a combination of levels of the same or different perspectives on the data source to which a granularity framework has been applied, $\mathcal{DG}$. The result is a subset of $\mathcal{DG}$, $\mathcal{DG}'$; thus, $task^4(\mathcal{DG}, \mathcal{DL}) \rightarrow \mathcal{DG}'$, where $\mathcal{DG}' \subseteq \mathcal{DG}$.

Prior and during investigation of solving the tasks and meeting the requirements, several experiments were carried out. These experiments illustrated granular querying with the GO and FMA (Keet, 2006b) and infectious diseases (Keet, 2006c; Keet and Kumar, 2005). Each experiment required a different, but overlapping, subset of functions and implementation peculiarities (*e.g.*, functions transformed into SQL and STRUQL queries), and they are not readily portable to other databases and ontologies due to its implementation-dependency. Here, I provide a more comprehensive account that unify and extend those functions. They will be defined first at the *conceptual* and then *logical* level in §4.2, thereby ensuring the sought-after portability of their semantics to a wide range of implementation scenarios. They can be mapped into various query languages for, *e.g.*, DB2 databases, OWL-ontologies, or digital libraries, where the formal characterisations ensure unambiguous meaning of the functions. These functions for the TOG are augmented with abstraction and expansion functions in §4.3 and reasoning over relational hierarchies in §4.4 to enlarge the set of management options and reasoning scenarios to more complex granular reasoning, of which a demonstration is included in §4.5. The last section (§4.6) contains a summary of this chapter. An earlier version of §4.3 with the literature review of §5.4 and experiments was published at the *First International Workshop on Object-Role Modeling* (Keet, 2005b), §4.3.1 and §4.3.2 without embedding in the TOG were published at the *10th Congress of the Italian Association for Artificial Intelligence* (Keet, 2007c), and a shorter version of §4.4.2 is published in the *Applied Ontology* journal (Keet and Artale, 2007).

4.2 Retrieval and selection of levels and its contents

The subsections are structured as follows: first, the high-level goal of the function is given, then its formal specification, and afterward the explanation. *Table 4.1* contains an overview of the functions and *Table 4.2* has brief explanations of the function arguments (the relations between the identified sets will be motivated in §4.2.7). The approach taken to define the functions resembles the underlying idea of ConQuer for querying conceptual models (Bloesch and Halpin, 1996, 1997), but where in this case it is not the conceptual model but the granularity framework that is used to structure and simplify the granular queries. They share the notion that the goal of the query can remain the same, but its implementation differs, be it different RDBMS SQL versions as with the ActiveQuery tool for ORM (Bloesch and Halpin, 1997) or any other language peculiarity (OODBMS, DL, and so forth). Given that the data source can be represented and saved in different languages and software, we define the types of granular queries in a purpose-oriented way to ensure portability. The first group deals with querying for perspectives and level, the second with retrieving level's contents, and the third group with conditional cross-granular queries and other auxiliary queries to retrieve additional information.

Table 4.1: Overview of retrieval and selection functions for the TOG.

Function	**Purpose**	**Section**
$getP : D^f \mapsto \mathcal{P}$	Retrieve all perspectives in the domain granularity framework	§4.2.1
$getL : \mathcal{P} \mapsto \mathcal{L}$	Retrieve all granular levels of a particular perspective	§4.2.1
$getC : \mathcal{L} \mapsto \mathcal{E}$	Retrieve contents of a level	§4.2.3
$selectP : \mathcal{P} \mapsto \mathcal{P}$	Select one particular perspective	§4.2.1
$selectL : \mathcal{L} \mapsto \mathcal{L}$	Select one particular level in a perspective	§4.2.1
$selectLs : \mathcal{L} \mapsto \mathcal{L}$	Select more than one level in a perspective	§4.2.2
$selectDL : \mathcal{P} \times \mathcal{L} \mapsto \mathcal{L}$	Select more than one level from more than one perspective	§4.2.2
$selectE : D^s \mapsto \mathcal{E}$	Select a particular entity	§4.2.5
$grain : D^s \mapsto \mathcal{DL}$	Retrieve the level of one entity/instance contained in one perspective	§2.3 §4.2.5
$grains : D^s \mapsto \mathcal{L}$	Retrieve the levels of one entity/instance contained in more perspectives	§4.2.5
$grainMulti : D^s \mapsto \mathcal{L}$	Retrieve the levels of multiple entities/instances contained in one perspective	§4.2.5
$grainsMulti : D^s \mapsto \mathcal{DL}$	Retrieve the levels of multiple entities/instances contained in more perspectives	§4.2.5
$intersect : \mathcal{L} \times \mathcal{L} \mapsto \mathcal{I}$	Intersect the contents of two levels	§4.2.3
$assignGL : D^s \times \mathcal{DL}$	Assign a level to an entity (/type)	§2.3

4.2.1 Selection of one level

Selecting a level within a perspective is a four-step process: retrieve the desired perspective, select a perspective, retrieve the levels in the selected perspective, and then select the desired level (see also *Figure 4.1*). For the selection functions a selection operator $\xi \in \Xi$ is needed so that we have function φ, a binary operator $\{=, \wedge, \vee, >, <\} \in \Xi$, a type X for what can be selected from a set $\mathcal{S}$, and x as instance to be selected: $\varphi_{X\xi x}\mathcal{S}$. This pattern also will be used for other functions specified in the next sections; superscripts and subscripts will be omitted where the meaning is clear from the context.

*F1. **Goal:** retrieve all granular perspectives gp_1, ..., gp_n contained in the domain granularity framework d^f. Input is a particular d^f and output is an unordered set of perspectives of that domain, P. **Specification:***

$$getP : D^f \mapsto \mathcal{P} \tag{F.1}$$

*F2. **Goal:** select a particular granular perspective gp_i from the perspectives retrieved with $getP$. Input is the set of perspectives of the domain, $\mathcal{P}$, output is a set with one perspective $gp_i \in \mathcal{P}$. **Specification:***

$$selectP_{GP=gp_i} : \mathcal{P} \mapsto \mathcal{P} \tag{F.2}$$

Table 4.2: List of function arguments and how they relate to each other.

Abbreviation	Selected section	Relations
$\mathcal{P}$	Set of perspectives in a domain granularity framework D^f, *i.e.*, $gp_1, \ldots, gp_n \in \mathcal{P}$	$\mathcal{P} \subset D^f$ $\mathcal{P} \prec D^f$ $\mathcal{P} \subset \mathcal{DG}$
$\mathcal{DL}$	Set of levels in a domain granularity framework D^f, *i.e.*, $gp_1gl_1, \ldots, gp_ngl_m \in \mathcal{L}$	$\mathcal{DL} \subset D^f$ $\mathcal{DL} \prec D^f$ $\mathcal{DL} \subset \mathcal{DG}$
$\mathcal{L}$	Set of levels $gp_igl_j, \ldots, gp_igl_n$ $(j, n \leq m, i \neq k)$ in a gp_i	$\mathcal{L} \subseteq \mathcal{DL}$
l_i	Selected level, output of the function $selectL$	$l_i \subset \mathcal{L}$
ls_i	Set of selected levels, output of the function $selectLs$	$ls_i \subseteq \mathcal{L}$
lss_i	Set of selected levels, output of the function $selectDL$	$lss_i \subseteq \mathcal{DL}$
$\mathcal{E}$	Collection of universals or particulars residing in a gp_igl_i	$\mathcal{E} \subset \mathcal{DS}$
$\mathcal{I}$	Intersection of the contents of two levels	$\mathcal{I} \subseteq \mathcal{E}_i \cap \mathcal{E}_j$ $\mathcal{I} \subset \mathcal{DS}$

F3. ***Goal:*** *retrieve all granular levels gp_igl_1, ..., gp_igl_n contained in the selected perspective. Input is a $gp_i \in \mathcal{P}$ and output is an ordered set of levels of that perspective, $\mathcal{L}$.* ***Specification:***

$$getL: \mathcal{P} \mapsto \mathcal{L} \qquad \text{(F.3)}$$

F4. ***Goal:*** *select a particular granular level gp_igl_i from the levels retrieved with the $getL$ function. Input is the set of levels of the perspective, $\mathcal{L}$, output is a set, l_i, with a single element from $\mathcal{L}$.* ***Specification:***

$$selectL_{GL=gp_igl_i} : \mathcal{L} \mapsto \mathcal{L} \qquad \text{(F.4)}$$

Observe that there is no need to impose further constraints on the functions. By the characterization of the TOG in Chapter 3, we have, *e.g.*, that it always holds that $\mathcal{P}$ is non-empty (see *Proposition 3.3*) and that $\mathcal{L}$ is an ordered set (*Lemma 3.9*). l_i is a set and result of $selectL_{GL=gp_igl_i}$, where $gp_igl_i \in l_i$ and $l_i \subset \mathcal{L}$.

Functions (F.2, F.4) meet a limited case of $task^1$ where l_i is used for only *one* level within a pre-selected perspective (trivially, $task^1$ implies perspective selection as sub-procedure).

Explanation. Assume we have a particular $DG \in \mathcal{DG}$, then selecting a level means selecting *one* granular level from all declared levels contained in all declared granular perspectives within a single domain. We have d^f as domain granularity framework that contains n granular perspectives and each perspective contains at least two granular levels, amounting to at least $2n$ levels to choose from. Two strategies for level selection are possible:

1. Make level selection a two-step process: select the desired perspective first, then select one of the levels contained in the chosen perspective.
2. Assume there is an unordered set of granular levels, select one, and let the software figure out in which perspective it is contained.

Both approaches have their advantages and disadvantages. Irrespective of practical considerations, (1) takes the ontological position that perspective takes precedence over level and (2) that perspective is secondary to its level. GP has its C and TG by which levels are constrained and only when a certain perspective or view is taken, *then* one looks at ever finer-grained details. In practice, some perspective might be 'obvious' and a user may prefer to go straight to a level, preferring option (2) above (1), but this cannot be assumed. An example of the pro-

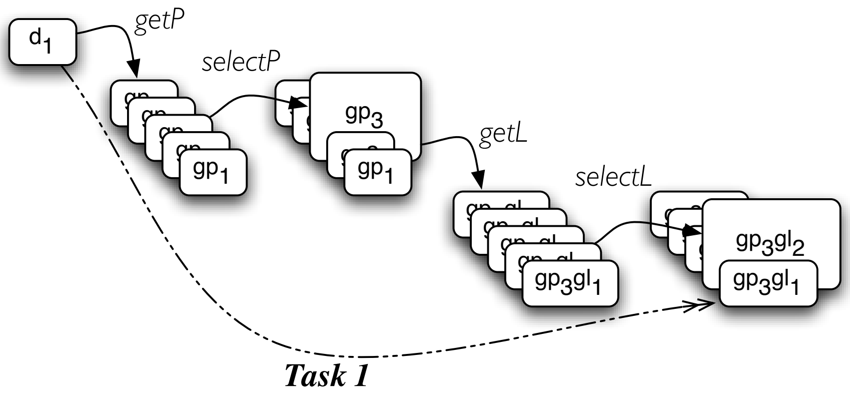


Figure 4.1: Step-wise selection of a granular level gp_3gl_2 for an arbitrary d^f.

cedure at the conceptual level is depicted in *Figure 4.1*: a step-wise selection from the domain through a selected perspective down to the desired level. The corresponding functions to carry out this step-wise selection procedure are $getP(d^f_1) = \{gp_1, ..., gp_5\}$ for the first step in *Figure 4.1*, then selection with $selectP(gp_3) = \{gp_3\}$ and using $getL$ after selecting a perspective, illustrated with $getL(gp_3) = \{gp_3gl_1, ..., gp_3gl_5\}$, so that finally the desired level is selected with $selectL(gp_3gl_2) = \{gp_3gl_2\}$. Although the successive steps may seem cumbersome for selecting just one level, the principle is easily extendable to other selection and retrieval functions, and can be abstracted into one compound operation. Before the function for $task^1$ is specified, we consider selection of more than one level in the next section.

4.2.2 Selection of multiple levels

Two options to select multiple levels are distinguished: 1) selecting several levels to subsequently retrieve and combine its contents for further processing, and 2) conditional selection that considers the contents as well. Option 1 will be addressed in this section and on 2 in the next section. Option 1 can be subdivided into two similar operations: selecting more than one level from one perspective and selecting levels from different perspectives. Both can be accomplished with a sequence of sub-functions. In addition to those introduced in the previous section, we have:

F5. ***Goal:*** *select one or more granular levels contained in one single granular perspective. Input is set of levels $\mathcal{L}$ retrieved with $getL$, output is a set of selected levels ls_i from $\mathcal{L}$ such that $ls_i \subseteq \mathcal{L}$ holds.* ***Specification:***

$$selectLs_{\bigwedge gp_igl_i} : \mathcal{L} \mapsto \mathcal{L} \qquad \text{(F.5)}$$

F6. ***Goal:*** *perform a selection of one or more levels from one or more perspectives. Input is the set of perspectives, $\mathcal{P}$, and for each perspective the set of levels, $\mathcal{L}$, from which one selects, and output is a set of levels contained in d^f, $\mathcal{DL}$, denoted with lss_i, where $lss_i \subseteq \mathcal{DL}$.* ***Specification:***

$$selectDL_{\bigwedge gp_i \bigwedge gp_igp_i} : \mathcal{P} \times \mathcal{L} \mapsto \mathcal{DL} \qquad \text{(F.6)}$$

$task^1$ can be solved now with the $selectDL$ function that returns the set of selected levels lss_i. This task consists of an iteration of three nested functions, which are $selectP$, $getL$, and either $selectL$ or $selectLs$.

Explanation. The $selectL$ function for multi-level selection within one perspective has been extended, where $\mathcal{L}$ is the set of levels in the selected perspective and the binary operator $\wedge$ (with $\wedge \in \Xi$) is used to select multiple levels. This can be written in shorthand notation, $\bigwedge$ such that $\bigwedge gp_i gl_i = gp_i gl_a \wedge ... \wedge gp_i gl_m$ where gp_i contains n levels and $m \leq n$. Thus, we have for multi-level selection (F.5), *e.g.*, $selectLs_{gp_3gl_2 \wedge gp_3gl_4} = \{gp_3gl_2, gp_3gl_4\}$. Note that ls_i is not a proper subset of $\mathcal{L}$ because it is possible that a user wants to select all levels in the chosen perspective. Observe that in that case, ls_i is semantically distinct from the hierarchically ordered levels in its perspective in an instantiation of the TOG: the former is an ordered list that results from a selection whereas the latter captures the structural relatedness of levels with s_ppart_of. The 1-level selection is a special case of the multi-level one, but labelled differently to avoid overloading terms.

For selection of more than one level from more than one perspective, $selectLs$ cannot be extended to selecting levels from multiple perspectives, because one has to take into account selection of at least one other perspective and have some way to distinguish levels belonging to different perspectives. Conceptually, (F.6) is straightforward: one has the granularity framework partially or fully expanded as in the left-hand side of *Figure 4.3* and then one can select the desired levels. A mapping of (F.6) to a formal notation and implementation algorithm can be achieved with a loop in two near-equivalent ways: either to select all desired perspectives and subsequently one or more levels for each selected perspective, or repeat the two-step process of selecting a perspective & retrieve levels and then selecting levels. Level selection as depicted in the left-hand pane of *Figure 4.3* then meets: $selectDL_{gp_1 \wedge gp_1gl_2 \wedge gp_3 \wedge gp_3gl_2 \wedge gp_5 \wedge gp_5gl_4} = \{gp_1gl_2, gp_3gl_2, gp_5gl_4\}$. Observe that $\mathcal{DL}$ is a set containing all levels in the domain, which is never an input parameter of a TOG-function. This is in line with the ontological foundations of the TOG which states that levels do not exist independently of the perspective they are contained in (see §3.4 for justification).

By looking at levels only, one can impose a constraint that such selection of >1 level never contains in its selection a particular level more than once and that in this limited case the order of selection is not relevant. However, as we will see in see §4.5 where advanced reasoning scenarios are introduced, both constraints are too restrictive.

4.2.3 Retrieving the contents of a level

Retrieving the contents of a particular granular level—$task^2$—is conceptually straightforward with a $getC$ function, but this hides many details, in particular the need to use the structure of the contents given the different types of granularity. This is elaborated on in the explanation below.

F7. ***Goal:*** *retrieve the contents, entities (/types), of a selected granular level* $gp_i gl_i$*. Input is the selected level, where* $gp_i gl_i \in \mathcal{L}$ *and output is a set of entities (/types),* $E \in \mathcal{E}$*, that takes into account the structure of the contents in the level.* ***Specification:***

$$getC : \mathcal{L} \mapsto \mathcal{E} \tag{F.7}$$

F8. ***Goal:*** *intersect the retrieved contents of selected granular levels* $gp_i gl_i, gp_j gl_j$ *(with* $RE(gp_i gl_i, gp_i)$*,* $RE(gp_j gl_j, gp_j)$*, and* $i \neq j$*). Input are the contents of the selected levels* E_i *and* E_j *(obtained with* $getC$ *for each level), and output is the intersection, set* $I \in \mathcal{I}$*, where* $\mathcal{I} \subseteq \mathcal{E}$*.* ***Specification:***

$$intersect : \mathcal{L} \times \mathcal{L} \mapsto \mathcal{I} \tag{F.8}$$

Explanation. $getC$ takes a particular granular level as argument and returns the contents of that granular level, *irrespective of how the contents themselves may be structured. Figure 4.3* depicts this graphically for two levels, reflecting that the structure of the levels' contents varies according to the type of granularity they adhere to. We return to this topic after introducing the essential characteristics of $getC$ function. Let gl_1 be a particular granular level such that $gl_1 \in \mathcal{L}$ and $y_1, ..., y_n$ are the entity types or instances residing in that particular level, *i.e.* $y \in U$ or $y \in PT$ with U the set of universals and PT the set of particulars in $\mathcal{DG}$, and $\mathcal{E}$ denotes the collection of universals or particulars that reside in a single granular level—and any further structure within the collection—and, trivially, $\mathcal{E} \subset d^s$ holds; *e.g.*, $getC(gp_3gl_2) = \{y_1, ..., y_n\}$. This characterisation of $getC$, however, does not guarantee preserving the structure of the source data like conveniently depicted in *Figure 4.3*. Although this could be ignored here and deferred to the implementation stage, it can be solved relatively easily by nesting other functions specific for each type of granularity. Before we resolve this, salient problems are illustrated if it were ignored.

Example 4.1. Continuing from *Example 2.7* and *3.4*, and the domain granularity framework for infectious diseases (Keet, 2006c), we have nine perspectives (1) where gp_1 = Mode of transmission, gp_2 = Site of entry, and so forth. Subsequently, we select a perspective (2), retrieve its levels (3), select a level (4), and retrieve its contents (5). The retrieved entity types are abbreviated here for brevity (see*Appendix A* for the full terms like Social environment and Personal hygiene).

```
1: getP(Infectious diseases) = { $gp_1$, $gp_2$, $gp_3$, $gp_4$, $gp_5$, $gp_6$, $gp_7$, $gp_8$, $gp_9$ }
2: selectP($gp_9$) = { $gp_9$ } $\equiv$ Predisposing factors
3: getL(Predisposing factors) = { $gp_9gl_1$, $gp_9gl_2$, $gp_9gl_3$, $gp_9gl_4$ }
4: selectL($gp_9gl_2$) = { $gp_9gl_2$ }
5: getC($gp_9gl_2$) = { SocEnv, PolEnv, EcoEnv, BioEnv, Diet, Stress, Smoking, PersHyg }
6: selectP($gp_8$) = { $gp_8$ } $\equiv$ Pathological process
7: getL(Pathological process) = { $gp_8gl_1$, $gp_8gl_2$, $gp_8gl_3$ }
8: selectL($gp_8gl_2$) = { $gp_8gl_2$ }
9: getC($gp_8gl_2$) = { Congestion, Red hepatization, Grey hepatization, Resolution }
```

(5) is merely an unordered set without further structure among the entity types in the level, but its simple approach ignores that in the data source the environmental factors are in a different branch of the taxonomy than the four types of living habit predisposing factors; that is, there is not one top type in gl_2. Its coarser-grained level gp_9gl_1 contains the entity type Environment that in the $\mathcal{DS}$ subsumes SocEnv, PolEnv, EcoEnv, and BioEnv, whereas Living habits subsumes Diet, Stress, Smoking, and PersHyg. If, on the other hand, we would have selected gp_8 (6), one of its levels (7, 8), and retrieve the contents (9), then there is an internal structure among the entities within the level and not only between the types in adjacent levels, for they are successive sub-processes of the Inflammatory process of pneumococcal pneumonia. However, neither (5) nor (9) reveals that the former is part of a taxonomy and the latter represent successive processes. In addition, one can 'take a turn' and look at types of hepatization. Overlapping the levels, they intersect at Red hepatization and Grey hepatization, as depicted in ***Figure 4.2*** (an abstraction of this shifting is sketched in ***Figure 4.3***), and one can hypothesise if the third subtype, Yellow hepatization, is a synonym of Resolution, related, or if it is a different process. $\diamondsuit$

The problems illustrated in *Example 4.1* revolve around inadequate usage of the original $\mathcal{DS}$ in the $\mathcal{DG}$. To solve this, we first need to look at the types of granularity and their influence on $getC$. Recollecting $has\_granulation$ and $adheres\_to$ from Chapter 3 (*Definition 3.7* and *3.13*) and the structure of the contents of each type of granularity (§2.3), we can obtain both the type of granularity and the content structure upon using $getC$. Therefore, the specifics for retrieving the contents of each type of granularity can be solved automatically and has to be defined only *once* for each leaf type of granularity. To this end, the $adheres\_to$ relation is transformed into a function, called tgL (an abbreviation of *t*ype of *g*ranularity that the *l*evel adheres to), which is defined in *Definition 4.3*.

DEFINITION 4.3 (tgL). tgL : $\mathcal{L} \mapsto TG$ *iff* $adheres\_to(x, \phi)$.

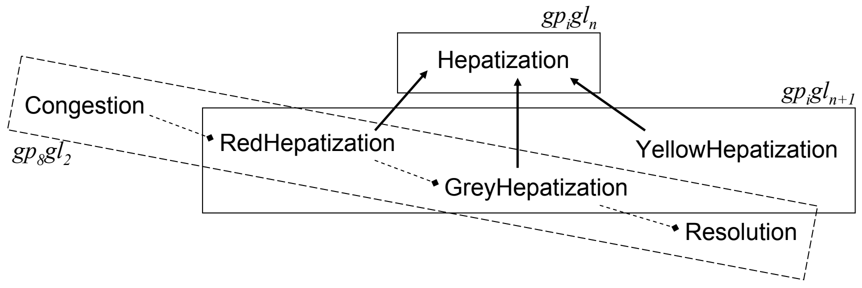


Figure 4.2: Three subtypes of Hepatization, of which at least two are part of the pathological processes of infectious diseases in level gp_8gl_2.

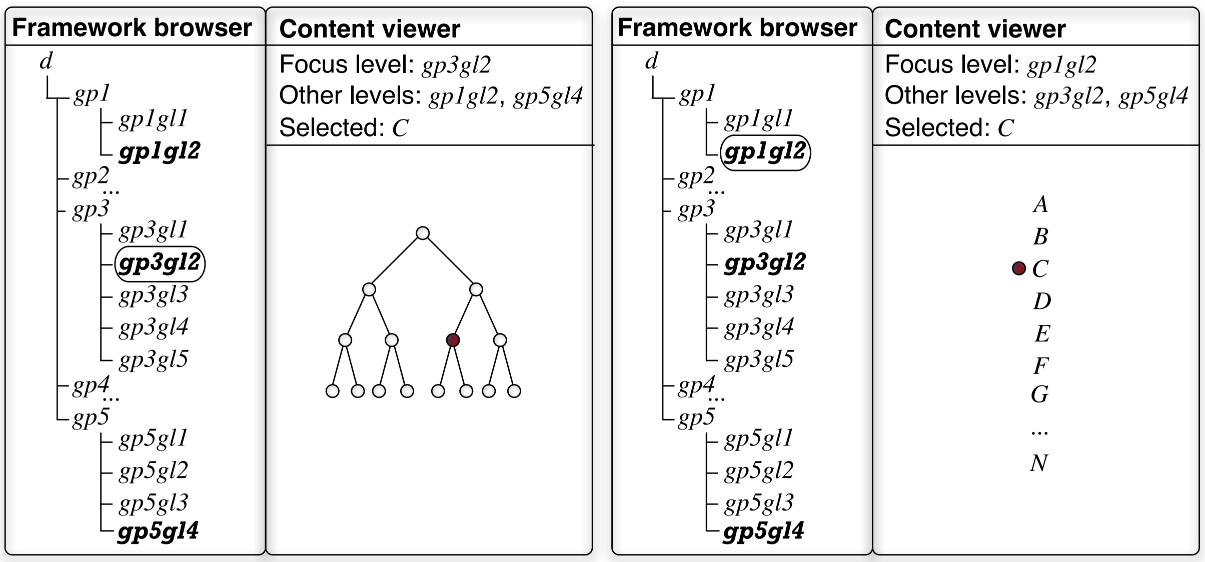


Figure 4.3: Left: Selection of levels, with the contents of gp_3gl_2 depicted; Right: Selection of levels, with the contents of gp_1gl_2 depicted. The black dot indicates that the entity type labelled with C is selected and contained in the hierarchy in gp_3gl_2 and in an ordered list in gp_1gl_2.

A suggested algorithm that demonstrates the nesting of this function in *getC* is included as *Algorithm 1*, where the goal of each **case**-option—*what* to do to retrieve it—is the same but actual operations depend on the software implementation, such as using a recursive query in STRUQL or a method in a C++ program; *i.e.*, its practical realisation depends on how the data, information or knowledge is organised in the $\mathcal{DG}$. Thus, to achieve the purpose of *getC*, one has to *use the type of granularity to which a level adheres* and the finer-grained $\mathcal{DG}$-specific procedures it requires to retrieve the content. Given the types of granularity and their corresponding content structure (§2.3), their impact on nested functions for *getC* are as follows.

- ⋆ For **saoG** we have a grid at each level, hence *getC* typically will retrieve this grid, which is a 2D representation fixed according to its coordinates. Typically, one wants to retrieve the associated representation of the material entity or its cartographic map, too; hence retrieval with *getC* will contain at least two sub-functions to handle this.
- ⋆ **samG** has its instances within a level as an ordered set, and does not need further processing for retrieval.
- ⋆ **sgpG**: entities with additional data about their size, which can be retrieved, *e.g.*, as a two-column table.
- ⋆ **sgrG**: textual representation and corresponding figurines, which can be retrieved as with **sgpG** but with two attributes—label of the entity and figurine—for each object.
- ⋆ **nasG** types have unordered sets and do not need further processing for retrieval.

Algorithm 1 Retrieving a level's contents by taking into account the content structure

```
Require: $x \Leftarrow selectL(x)$
procedure $getC(x)$
1: $\phi \Leftarrow tgL(x)$
2: switch
3:     case $\phi =$ samG : «see text for details»
4:         RESULT $\Leftarrow$ query and sort the set
5:     ⋮
6:     case $\phi =$ nrG : «see text for details»
7:         RESULT $\Leftarrow$ recursive query over relation $GR$
8: end switch
9: return RESULT
```

- ⋆ Depending on the implementation, **nacG** can be an unordered set that is aggregated or have additional 'subgroups' in a level. Members at the lower level can be a) aggregated as an unordered set, b) ordered taxonomically, or c) another representation, like a graph with the positions of sports team players on the field. Consequently, finer-grained behaviour of the sub-procedures of $getC$ depends on the $\mathcal{DS}$.
- ⋆ **nfG**: includes relations among entities within a level, which is useful together with the $getC$ to retrieve all the entities and its relations. (This works only if those entities do not have relations with other entities beyond the level they reside in, else an additional verification is needed that checks that the candidate entity to retrieve is not in another level within the same perspective.) The minimal structure of the representation of the contents are triples with $\langle entity, relation, entity\rangle$, which can be listed as unordered set, or rendered in some graphical representation.
- ⋆ Using $getC$ with a level adhering to **nrG**-type granulated entities: one may want to take into account their respective supertypes at the adjacent coarser-grained level, and then aggregate the subsumees in the branch into a granule in the focal level. An example of this is the query to retrieve the cells from the Cell-level that are part of blood, thereby omitting the other types of cells residing in the Cell-level. Hence, $getC$ uses at least a recursive query to retrieve hierarchically organised content.

Once the content is retrieved, it can be used for further processing, such as intersecting contents of two levels. Intersection follows immediately from *Lemma 3.19* in §3.7.2: if two levels in different perspectives overcross then their contents overlap and then the intersection of the content of the two levels is non-empty. To meet this TOG constraint, the $intersect$ function (F.8) is introduced, which first retrieves the contents of each level with $getC$ and subsequently intersects them, as shown in *Algorithm 2*. Obviously, this can be scaled up to intersection of more than two levels.

4.2.4 Retrieving granulated information

Retrieving granulated information with $task^4$ amounts to re-focussing and extending $task^1$ and reusing $task^2$. $task^4$ goes further by including the level contents to retrieve a subset of $\mathcal{DG}$ with $getC$. *Algorithm 3* demonstrates how the pieces can be put together with a simple brute force approach, which can be useful for reasoning scenarios where minimum interaction with a domain expert is desired.

To allow more elegant information retrieval, we need functions not only for the TOG components, but also to query the individual entities (/types) so that conditional selections can be executed and to provide sufficient machinery to solve the problems illustrated in *Example 4.1*; this will be elaborated on in the next section and *Algorithm 5*. In turn, $task^4$ is a precursor for

Algorithm 2 Intersection of two levels

```
Require: $x_i \Leftarrow selectL(x_i)$
Require: $x_j \Leftarrow selectL(x_j)$
procedure $intersect(x_i, x_j)$
1: if $RE(x_i, y_i) \wedge RE(x_j, y_j) \rightarrow y_i \neq y_j$ then
2:     $E_i \Leftarrow getC(x_i)$ «use Algorithm 1 for content retrieval»
3:     $E_j \Leftarrow getC(x_j)$ «use Algorithm 1 for content retrieval»
4:     RESULT $\Leftarrow E_i \cap E_j$
5:     return RESULT
6: else
7:     print "no. you have to select levels from different perspectives"
8: end if
```

granular reasoning and with experiment ex^2, they fulfill $task^5$. Together, they enable more complex reasoning scenarios with and without user intervention; these scenarios are deferred to §4.5.

4.2.5 Entity (/type) selection to retrieve its levels

We already have the $grain$ function to retrieve the level of a pre-selected entity (/type) in one perspective (2.12). Here, four additional functions are introduced that deal with selecting one or more entities and retrieving one or more levels.

*F9. **Goal:** select an entity (/type) from the subject domain. Input is an entity (/type), (*i.e.*, $U(x)$ or $PT(x)$), $x \in d^s$ and the output of the selection ensures that the selected entity (/type) resides in some level already, hence $x \in \mathcal{E}$. **Specification:***

$$selectE : D^s \mapsto \mathcal{E} \tag{F.9}$$

*F10. **Goal:** given a selected entity (/type), retrieve all the levels (in different perspectives) that that entity (/type) resides in. Input is an entity (/type) x, and $x \in d^s$ (*i.e.*, $U(x)$ or $PT(x)$), and output of the function is a set of levels that is a subset of $\mathcal{DL}$ (the set of all levels in d^f). **Specification:***

$$grains : D^s \mapsto \mathcal{DL} \tag{F.10}$$

*F11. **Goal:** given multiple selected entities (/types), retrieve their levels within one perspective. Input are entities (/types) $x, y, ... \in d^s$, uses $grain$ as nested function, and output of the function is a set of levels within one perspective, $\mathcal{L}$. **Specification:***

$$grainMulti : D^s \mapsto \mathcal{L} \tag{F.11}$$

*F12. **Goal:** given multiple selected entities (/types), retrieve their levels among multiple perspectives. Input are entities (/types) $x, y, ... \in d^s$, uses $grains$ as nested function, and output of the function is a set of levels within $\mathcal{DL}$. **Specification:***

$$grainsMulti : D^s \mapsto \mathcal{DL} \tag{F.12}$$

Explanation. The basic function to retrieve the level an entity resides in, is grain : $D^s \mapsto \mathcal{L}$ (2.12). Its neat simplicity, however, does not suffice, which was illustrated in *Example 3.8* and *3.9*. The main limitations are that it is perspective-unaware and is based on the assumption that

Algorithm 3 Granular information retrieval with $task^4$

```
Require: exists $\mathcal{DG}$
 1: $P \Leftarrow getP(d^f)$ «$task^1$: do level selection of one or more levels»
 2: $x_1 \Leftarrow selectP(x_1)$
 3: $L \Leftarrow getL(x_1)$
 4: if selecting 1 level then
 5:     $y_1 \Leftarrow selectL(y_1)$
Ensure:     $RE(y_1, x_1)$ «and start with $task^2$ for content retrieval of one level»
 6:     $E_1 \Leftarrow getC(y_1)$ «use Algorithm 1 for content retrieval»
 7:     return $y_1$, $x_1$, and $E_1$
 8: else «selecting n levels, n> 1»
 9:     $Y \Leftarrow selectDL(P, L)$
10:     for all $i$ levels, $1 < i \leq n$, in $Y$ do
Ensure:       $RE(y_i, x_i)$
11:     end for
12:     for all $i$ levels, $1 < i \leq n$, in $Y$ do ««start $task^2$ for content retrieval of selected levels»»
13:         $E_i \Leftarrow getC(y_i)$ «use Algorithm 1 for content retrieval»
14:     end for
15:     $X \Leftarrow$ do some post-processing with $E_i$ and $L$ «e.g., with intersection Algorithm 2»
16:     RESULT $\Leftarrow X, P, L, ...$
17:     return RESULT
18: end if
```

an entity (/type) can be categorised in one level only. Although it is possible that a d_i^f contains only one gp_i, it is more realistic that multiple perspectives have been declared and that x is located in more than one granular level across perspectives. To remedy this, several options are at our disposal.

1. Construct some indexing mechanism or modify the name of the $grain$ function according to the perspective (see *Example 3.9* for a discussion). However, what can be addressed adequately at the conceptual and logical level should not be deferred to the implementation stage.
2. Restrict usage to one limited case: if one knows which perspective to search, one can construct a rule alike "if gp_i then do $grain(x) = gp_i gl_i$" or combine the selection operator $selectP_{GP=gp_i}$ with $grain(x)$ to first select a granular perspective gp_i and then to fire the $grain$ function to retrieve the level where entity (/type) x resides. The same approach can be used to decompose the retrieval of multiple levels into sequential steps of the $grain$ function, but this requires additional process management.
3. Define a new function that retrieves *all* levels the selected entity (/type) resides in (F.10). This function is constrained as follows:

$$grains(x) \rightarrow \exists^{\geq 1} y, z(GL(y) \wedge GP(z) \wedge RE^{-}(z, y) \wedge PT(x) \wedge in\_level(x, y)) \tag{4.10}$$

The functions are illustrated in *Example 4.2*, where I also demonstrate that the issues mentioned in the first part of *Example 3.8* and *3.9* in §3.7.2 now can be solved easily.

Example 4.2. Considering bacteriocins once more, we have a $\mathcal{DG}$ with d_a = Bacteriocins, a location perspective gp_1 with $\mathsf{gp}_1\mathsf{gl}_2$ = Mobile DNA fragment, a functional perspective with $\mathsf{gp}_2\mathsf{gl}_3$ = Gene, and assume other levels have been declared to satisfy the TOG constraints. For Tn5301, we can use (F.10) to retrieve its levels (1), or know or guess that Tn5301 must have some location and then query more precisely with (2).

1: grains(Tn5301) = { $\mathsf{gp}_1\mathsf{gl}_2$, $\mathsf{gp}_2\mathsf{gl}_3$ }
2: if gp_1 then do grain(Tn5301) = $\mathsf{gp}_1\mathsf{gl}_2$

Translated into natural language, (1) says "Tn5301 is a gene located on a mobile DNA fragment" and (2) says "querying location, then the level of Tn5301 is mobile DNA fragment".
We also have the domain granularity framework from (Keet, 2006c) and the functions that were introduced in this chapter. Now we retrieve the levels with the $grains$ function for both the *V. cholerae* and the Cholera toxin,

a: grains(V. cholerae) = { gp_2gl_7 , gp_6gl_1 }
b: grains(Cholera toxin) = { gp_2gl_9 , gp_6gl_2 }

where gp_2gl_7 = Cell, gp_2gl_9 = Molecule, gp_6gl_1 = Toxin producer, and gp_6gl_2 = Inhibitor. ♢

Thus, the inconsistencies are solved and proliferation of new $gran$ functions is avoided by adding a single extension to the function's specification and using a formal domain granularity framework to support it. Moving forward to solve the problem with the Predisposing factors described in *Example 4.1*, one more feature is needed in addition to knowing the type of granularity and the $grain$ function: a way to access the content of the adjacent higher level and relate it to the contents of the focal level. For both the **nrG** and **nfG** types, the relation to select the parent type is, or should be, readily provided by the $\mathcal{DS}$ and may be for other data sources that are granulated according to other types of granularity using $uses\_GR$. To specify entity (/type) selection generally, I introduce the $selectE$ function (F.9). With this addition, content retrieval of the previously unordered set ((5) in *Example 4.1*) is solved as follows.

Example 4.3. The predisposing factors are of the granularity type **nrG** and the goal is to answer queries such as "given the predisposing factor Environment at level gp_9gl_1, retrieve the contents at level gp_9gl_2"; *e.g.*, condensed in (1). If the supertype is unknown beforehand, the query needs a preliminary step to retrieve the parent type of the selected entity; *e.g.*, "retrieve the granule of Stress" where Stress is subsumed by Living habit (2).

1: if grain(Environment) = gp_9gl_1 and selectE(x) and grain(x) = gp_9gl_2
then getContent(gp_9gl_2) = { SocEnv, PolEnv, EcoEnv, BioEnv } and is_a(x, Environment)
2: if selectE(Stress) and grain(Stress) = gp_9gl_2 and is_a(Stress, x) and grain(x) = gp_9gl_1 and selectE(y) and grain(y) = gp_9gl_2
then getContent(gp_9gl_2) = { Diet, Stress, Smoking, PersHyg } and subsumes(x,y)

A more detailed example is included in §4.5.1. ♢

In the above example, the levels and entities were retrieved in successive steps. Generalising from the example, several permutations can be assessed to retrieve levels of more than one entity within one or among multiple perspectives, which can be met with repeated use of $grain$ and $grains$, respectively. For clarity, they will be labelled differently to indicate they contain nested functions, hence, their meaning is made explicit.

⋆ 1 entity (/type), one perspective, one level: $grain(x)$;
⋆ 1 entity (/type), multiple perspectives, multiple levels: $grains(x)$;
⋆ multiple entities (/types), one perspective, multiple levels: $grainMulti(x)$;
⋆ multiple entities (/types), multiple perspectives, multiple levels: $grainsMulti(x)$.

These four functions address all permitted options to find the level(s) of entities (/types). Other combinations, such as one entity (/type) in one perspective residing in multiple levels, violate the TOG. For instance, if a user had allocated Nisin both in the Peptide and in the Quaternary protein structures levels within the same perspective, then something is wrong about the knowledge of what Nisin is, the user is confused, the domain granularity framework has been ill-designed, or all of them, because Nisin cannot be both a peptide and complex protein. The constraint to have an entity in no more than one level within the same perspective should have prevented this double allocation, or have returned an error upon checking the $\mathcal{DG}$ for consistency.

4.2.6 Other functions

The main functions to query a domain, its perspectives and levels, and their contents have been defined in the previous sections, but it is possible to define many more functions for a particular

$\mathcal{DG}$, such as granular information retrieval from corpora, databases, queries over a large conceptual data models, and Data WareHouses (DWH). DWH implementations in particular focus on querying the information system with advanced queries. Functions for such queries can be easily added to the TOG: if *Figure 3.1* is directly transformed into a database schema, then each object type can be used as arguments to compose queries. Correspondingly, we can define functions for each combination. For instance, with $\mathcal{P}$ still the set of perspectives, and adding $\mathcal{C}$ as the set of criteria, then (F.13) enables us to retrieve the criterion of a granular perspective; with its inverse, $crit^{-}$, the perspectives that have the selected criteria can be retrieved. Likewise for a level's values, one could define $value$ (F.14) or its associated functions, if any, with $glion$ (F.15), where $\mathcal{V}$ is the set of values and the conversion function $\vartheta \in \Theta$. Moreover, *one can define functions to retrieve each component of the* TOG.

$$\text{crit}: \ \mathcal{P} \mapsto \mathcal{C} \tag{F.13}$$
$$\text{value}: \mathcal{L} \mapsto \mathcal{V} \tag{F.14}$$
$$\text{glion}: \mathcal{L} \mapsto \Theta \tag{F.15}$$
$$\text{grel}: \ \mathcal{L} \times \mathcal{T} \mapsto \mathcal{G} \tag{F.16}$$

To make it more interesting, retrieving the granulation relation between entities (/types) in adjacent levels with (F.16) where $\mathcal{T}$ is the set of types of granularity and $\mathcal{G}$ the set of granulation relations. Analogously, given that we already can retrieve the level an entity (/type) resides in with $selectE$ and link it to $grain$, then $grain$'s output, in turn, can be fed into $value$ or $glion$ and so forth to construct (unions of) conjunctive queries. Put differently, we can use two approaches for proliferation of functions:
(i) convert the TOG relation into functions alike tgL and build more complex functions from these base functions, or
(ii) move in the direction of conjunctive queries at the instance-level and role composition at the type level.
We then have new functions such as tgP for the $has\_granulation$ relation (F.17), grT for $uses\_GR$ (F.18) and as goal for a compound function "retrieve the granulation relation of a perspective" with grP (F.19), or role composition with a newly defined path P_has_GR such that $P\_has\_GR \triangleq has\_granulation \circ uses\_GR$. However, such true role composition makes a language undecidable, idem for its weaker version $R \circ S \sqsubseteq T$ (Schmidt-Schauss, 1989; Wessel, 2001).

$$\text{tgP}: \mathcal{P} \mapsto \mathcal{T} \tag{F.17}$$
$$\text{grT}: \mathcal{T} \mapsto \mathcal{G} \tag{F.18}$$
$$\text{grP}: \mathcal{P} \times \mathcal{T} \mapsto \mathcal{G} \tag{F.19}$$

Note that the main focus of the TOG is on *representing* granularity, whereas any querying specific to an application scenario, such as fact finding in DWHs, comes afterward: the more comprehensive the representation of granularity components, the more versatile and well-founded the queries one can pose over a $\mathcal{DG}$. For instance, for a scenario such as information retrieval from large corpora, additional types of queries may be conceived, such as D-ABS functions to hide information from the diagrammatic display of an ontology (§4.3) and Fagin *et al.*'s (2005) functions. Fagin *et al.* have a "divide" to create levels, "differentiate" to compare documents along dimensions, and "discover" to find finer-grained distinctions among documents that were not yet used to divide the documents along the dimensions in the multi-structural database (MSD). A $divide$ function can serve *prior* to declaring a domain granularity framework if one has insufficient knowledge about either the domain or the $\mathcal{DS}$. The other two functions are useful explore the data in a $\mathcal{DG}$ and can be added easily to the list of TOG functions for document retrieval scenarios (see also Chapter 5).

Combinations of TOG functions enable advanced granular reasoning, which will be demonstrated in §4.5 after addressing $task^3$ in the next section, and abstraction and expansion functions in §4.3.

4.2.7 Relating subsets of $\mathcal{DG}$

In this section we look at $task^3$: how the different subsets of applied domain granularity relate to each other. The results follow trivially from the previous sections and Chapter 3, and are summarised in *Table 4.2*. This will be explained in the remainder of this section.

Domains. First, the types of data sources that will be considered are databases, knowledge bases, and ontologies; hence, instead of any D^s, this is restricted for the current scope with $\mathcal{DS}$ (see *Definition 4.1*) where $\mathcal{DS} \subset D^s$, because the same subject domain can be stored in other types of data sources, such as a digital library.

Looking at $\mathcal{DG}$, which denotes an applied domain granularity framework (*Definition 4.2*), $\mathcal{DS} \subset \mathcal{DG}$, because the latter contains an instance of D^f. More formally:

LEMMA 4.1. *$\mathcal{DS}$ is a proper subset of $\mathcal{DG}$.*

Proof. From *Definition 4.2* we know there is an x such that $D^f(x)$; from *Proposition 3.2* that there must be some y and $D^s(y)$; from *Definition 4.1* that $\mathcal{DS} \subset D^s$; and from *Definition 4.2* we have that $\mathcal{DG}$ is a *combination* of x and a $\mathcal{DS}$. Therefore $\mathcal{DS} \subset \mathcal{DG}$. □

$task^3$ is answered with *Lemma 4.1*.

Perspectives and levels. For the relation between the set of perspectives $\mathcal{P}$ and D^f and $\mathcal{DG}$, we can take a set-based approach or mereological (see *e.g.*, Pontow and Schubert, 2006, for transformations). Using the results obtained in Chapter 3, the following can be derived.

LEMMA 4.2. *$\mathcal{P}$ is a proper part of D^f.*

Proof. We have $RE(x,y) \rightarrow GP(x) \wedge D^f(y)$ and $RE(x,y) \rightarrow ppart\_of(x,y)$ (*Definition 3.21*). Given $\mathcal{P}$ is a set, we use the correspondence between set theory and mereology so that $\prec$ and $\subset$ are comparable, then $\mathcal{P} \prec D^f$, or in its equivalent: $\mathcal{P} \subset D^f$. □

COROLLARY 4.1. *$\mathcal{P} \subset \mathcal{DG}$.*

A similar argumentation holds for the relation between $\mathcal{DL}$ and D^f.

LEMMA 4.3. *$\mathcal{DL}$ is a proper part of D^f.*

Proof. Take *Definition 3.21* with $RE(x,y) \rightarrow GL(x) \wedge GP(y)$, transitivity of RE (*Lemma 3.13*), mandatory GP (*Proposition 3.9*), and *Lemma 4.2*, therefore $\mathcal{DL} \prec D^f$, or in its set-theoretic equivalent: $\mathcal{DL} \subset D^f$. □

COROLLARY 4.2. *$\mathcal{DL} \subset \mathcal{DG}$.*

LEMMA 4.4. *$\mathcal{L} \subseteq \mathcal{DL}$.*

Proof. $\mathcal{L}$ is the set of levels in one perspective (F.3) and $\mathcal{DL}$ the set of levels in all perspectives. By *Proposition 3.3*, there is at least one perspective in a domain, therefore "$\subseteq$". □

Last, the relations between selected levels and $\mathcal{L}$ and $\mathcal{DL}$ follow immediately from *Theorem 3.2* and the definitions of the functions.

LEMMA 4.5. *l is a proper subset of $\mathcal{L}$.*

Proof. By *Theorem 3.2* there are ≥ 2 levels in a perspective; hence, with a selection of *one* level—with function $selectL$ in (F.4)—of all levels in a perspective, we have $l \subset \mathcal{L}$. □

LEMMA 4.6. *ls is a subset of $\mathcal{L}$.*

Proof. By *Theorem 3.2* there are ≥ 2 levels in a perspective; hence, with a selection of *at least one* level but possibly *all* levels—with $selectLs$ in (F.5)—in a perspective, we have $ls \subseteq \mathcal{L}$. □

LEMMA 4.7. *lss is a subset of $\mathcal{DL}$.*

Proof. With $selectDL$ (F.6) one selects of *at least one* level but possibly *all* levels in *all* perspectives in a domain, therefore we have $lss \subseteq \mathcal{DL}$. □

Contents. The two set we consider here are $\mathcal{E}$ and $\mathcal{I}$, which were introduced in §4.2.3. Their relation with each other and $\mathcal{DS}$ follow immediately from the previous lemmas.

LEMMA 4.8. *$\mathcal{E}$ is a proper subset of $\mathcal{DS}$.*

Proof. $\mathcal{E}$ is the output of $selectE$ (F.9), which has $selectL$ (F.4) and $getC$ (F.7) nested. By *Lemma 4.5*, $l \subset \mathcal{L}$, hence there must be a remainder. Assume that the remaining level(s) are not empty, then $\mathcal{E} \subset \mathcal{DS}$. □

Note that in *Lemma 4.8*, one could have assumed that only one level in a $\mathcal{DG}$ actually contains entities (/types), which would change the relation into that of $\mathcal{E} \subseteq \mathcal{DS}$. However, this goes against good sense: there is no point in defining a whole domain granularity framework and then to populate only one level. To enforce preventing this situation, *Lemma 4.8* has $\mathcal{E} \subset \mathcal{DS}$.

Last, we have $\mathcal{I} \subseteq \mathcal{E}$, because by definition of $intersect$ in (F.8), $\mathcal{I}$ must be less or equal to the contents of the levels.

4.3 Abstraction and expansion

Within the scope of granularity, I focus on abstraction with respect to the procedure of ignoring finer-grained details. There can be uninteresting things with respect to a focus or viewpoint, yet if it were included, it would belong to the same level of detail, which has to do with demarcating the subject domain. Excluding this informal use of abstraction, one can distinguish between abstraction and granularity on three points.

⋆ Granularity acknowledges increasing levels of detail and generality, whereas abstraction reduces the scope as much as possible to zoom in on a particular subset of what is available that is a simplification compared to the complete model.
⋆ Granularity is a static structure, but abstraction is the process to go from finer to coarser-grained information.
⋆ Abstraction focuses on contents of levels, whereas with granularity there is a supporting structure.

During a conceptual analysis and modelling stage in software development, abstraction and finer-grained levels of granularity can be competing goals, but once the applied domain granularity framework ($\mathcal{DG}$) has been developed, levels of granularity facilitate the abstraction process computationally, hence resulting in more consistency and higher reliability and repeatability of query processing and answering. The aim of this paragraph is to investigate how abstraction can be integrated with the granularity framework. This requires a focus on the entities (/types) contained in a granular level, but with additional support from the framework components to improve the abstraction operations. Section 4.3.1 contains a summary of existing approaches of abstraction, their deficiencies and proposed solutions to these problems, which are addressed in §4.3.2 where a distinction is made between three main ways of abstraction and their respective 12 basic and complex abstraction functions. The inverse operation of abstraction is expansion, which is elaborated on in §4.3.3.

4.3.1 Overview and key problems of abstraction

A range of approaches to abstractions are described by Campbell *et al.* (1996); Giunchiglia *et al.* (1997); Keet (2005b); Tavana *et al.* (2007). The earlier works on theories of abstraction differ along three dimensions—language, axioms, and rules—and, as summarised by Giunchiglia *et al.* (1997), concern topics such as abstraction for planning, reduction of search, and logical theories; the latter is relevant in the current scope. Campbell *et al.* (1996); Keet (2005b); Tavana *et al.* (2007) provide overviews that focus on abstractions for conceptual data models and ontologies, which are logical theories that have essential graphical 'syntactic sugar' for improved usability from the perspective of the domain expert. The main issues with current approaches will be illustrated now (see §5.4 for details).

Manual abstraction is used for UML modularisation, (E)ER clustering, and 'abstraction hierarchies' (*e.g.*, Jäschke *et al.*, 1993; Lind, 1999; Tavana *et al.*, 2007; Yu *et al.*, 2002; SemTalk) with the drawbacks that it is a laborious, intuitive, not scalable and ad hoc method. *Semi-automatic* abstractions are developed by Campbell *et al.* (1996), which is based on heuristics to simplify large ORM models, but these rules are neither directly applicable to ontologies or other types of conceptual data models (Keet, 2005b), nor enjoys software support. Basic *syntax-focused* abstraction using Local Model Semantics of context reasoning (Ghidini and Giunchiglia, 2003) will be discussed in detail in §5.4. The examples Ghidini and Giunchiglia (2003) provide for abstractions is *taxonomic generalisation*; others also associate abstraction and granularity with the subsumption relation (Degtyarenko and Contrino, 2004; Fonseca *et al.*, 2002; Zhang *et al.*, 2002; Pandurang Nayak and Levy, 1995; Kiriyama and Tomiyama, 1993). This, however, ignores details of abstraction such as collapsing sub-processes into a grander process. Further, the syntax and subsumption of Ghidini and Giunchiglia (2003) are only a component of granularity, overloading their abs function, thereby in itself abstracting away the finer details of the process of abstraction. *Syntax abstraction augmented with semantics* was investigated by Mani (1998); Pandurang Nayak and Levy (1995), where the latter extends (Giunchiglia and Walsh, 1992) and the former augments Hobbs's (1985) contribution. Mani's functions address the important notion of "folding" formally, which may be useful for dealing with entities in non-scale-dependent levels of granularity in particular. For instance in the biology domain there are processes involved in the entity type Second messenger system that abstract into the one entity type, or in systems biology where the cell is composed of (modular) subsystems (*e.g.*, Sontag, 2004). Current abstraction approaches can be grouped according to topic and implementation foci and the nature of the abstraction operations (Keet, 2005b).

⋆ Relation-turned-into-function: this mainly concerns a taxonomy—moving 'up' in the specialisation/generalisation hierarchy through the *is_a* relation abstracting away a distinguishing property—and partonomy (mereology) through the *part_of* relation, or one of its subtypes.
⋆ Conceptual data modelling: ORM heuristics, UML modularisation, and EER abstractions (including clustering).
⋆ Folding operations of different types of entities resulting in other types (perdurant into endurant etc.), focussed on linguistics, the black boxes in biology, ecology models, and Abstraction Hierarchy.
⋆ Syntax-based, such as limited to Local Model Semantics of contextual reasoning.

The solutions for abstractions have three main problems, which can be solved using the TOG; conversely, the TOG benefits from abstractions to augment a wider range of scenarios for complex granular reasoning.

1. Problem: Abstraction focuses only on the contents of a level, thereby lacking a surrounding *framework*.
   Solution: The hitherto liberal and unconstrained 'levels of abstraction' is integrated with TOG components to provide consistency in the applicability and usage of the functions.

2. Problem: A general abstraction function *abs*—with or without minor extensions—does neither reveal *what* it is abstracting nor *how*.
   Solution: First, three conceptually distinct methods of abstraction are identified. Second, their corresponding abstraction functions will be introduced with both finer-grained distinctions and availing of foundational ontological types.

3. Problem: The hitherto proposed solutions are mainly theoretical and not developed for or assessed on its potential for reusability and scalability.
   Solution: As with the TOG functions in §4.2, the conceptualization as well as the corresponding formalisation of the functions will be provided, which simplifies understanding, provides space for extensions with more abstraction functions, and makes them usable and reusable across implementations. There are abstraction functions for both basic and complex folding operations, where the latter also can be built up from the basic ones. Thanks to this approach, abstractions become scalable, are unambiguous to implement, and amenable to automation.

The proposed solutions will be introduced in the next section.

4.3.2 Abstraction operations for the granularity framework

Before we can look into which (type of) abstraction operations are needed, general considerations and assumptions are given.

The various abstractions have different purposes at the conceptual level, regardless implementation issues. What is abstracted away depends on multiple factors, such as i) the subject domain, ii) user's perspective and context, iii) the type of abstraction, iv) the procedure of (consecutive steps of) abstraction, and v) on what type of representation/model the abstraction is performed. The focus here is on points iii-v.

First, there are simplification and generalisation. Simplification entails both (a) abstraction where the finer-grained model as a whole maps into an entity in the coarser-grained level that are different from those in the finer-grained level and (b) abstraction whereby semantically less relevant components are removed in order to maintain the salient entities and relations, which thus may be contained in the fine- *and* coarse-grained theories (but recollect *Definition 3.12* and requirement 10 in Chapter 3). Generalisation indicates there is a hierarchy were the types in the finer-grained level are mapped into their supertype. Because of these differences, some types of abstraction functions can be remodelled as relations between the coarser- and finer-grained entities, but for other types, it involves a coordinated folding or deletion. A deletion can be either permanent or only in the graphical representation whilst maintaining, but hiding, compositionality of the theory. *Figure 4.4* depicts these differences informally, and the three distinct modes of abstraction are defined as follows.

DEFINITION 4.4 (R-abs). *An abstraction is an* R-ABS *iff the more-detailed type ϕ abstracts into its related parent type ψ.*

DEFINITION 4.5 (F-abs). *An abstraction is an* F-ABS *iff a more-detailed theory T_0 (with $\phi_1 ... \phi_n$ entity types, relations, and constraints among them) abstracts into a single related parent type ψ in T_1 that is distinct from any ϕ_i.*

DEFINITION 4.6 (D-abs). *An abstraction is a* D-ABS *iff a more-detailed theory T_0 with $\phi_1 ... \phi_n$ entity types, relations, and constraints abstracts into theory T_1 with $\phi'_1 ... \phi'_m$ entity types, relations, and constraints, and $m < n$ through* deletion *of elements either from T_0 to obtain T_1 or from T_0's user interface.*

Observe that the abstractions are defined over universals but not instances. This is because abstractions are generally used for ontologies and conceptual data models, which range over universals (classes, concepts, relations). However, the same types and abstraction functions can

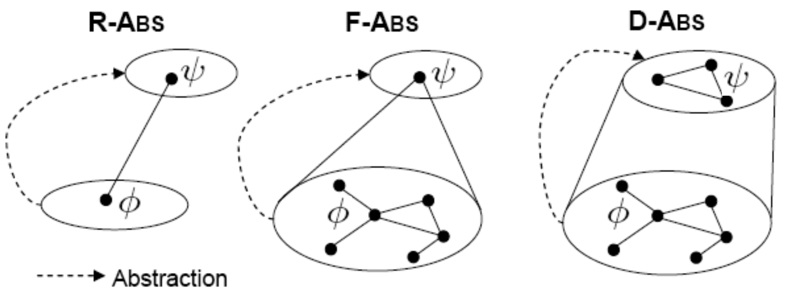


Figure 4.4: Three conceptually distinct kinds of abstraction operations. R-ABS: the relation is remodelled as a function; F-ABS: folding multiple entities and relations into a different type of entity; D-ABS: deleting semantically less relevant entities and relations.

be defined for instances, so that the abs functions resemble ontology-enhanced aggregation and roll-up functions for databases.

Abstraction does not exist in isolation. Proposals in the literature on how to deal with abstraction do allude to levels of granularity, but this is left to the reader to imagine (Ghidini and Giunchiglia, 2003; Giunchiglia *et al.*, 1997; Hobbs, 1985; Keet, 2005b; Pandurang Nayak and Levy, 1995). In contradistinction, here we do have the advantage of the TOG that can provide a supportive framework for abstraction; that is, the well-defined GL is used for the notion of *abstraction level*, and GP for *abstraction hierarchy*. Given the following two definitions for abstraction level and abstraction hierarchy, where instead of the $grain$ function $labs$ is used ($labs : \mathcal{T} \mapsto \Lambda$, where $T \in \mathcal{T}$ is a theory residing in some abstraction level $\lambda_i \in \Lambda$) (Keet, 2007c), we can integrate them with the TOG according to *Proposition 4.1* and *4.2* that impose constraints on usage of any abstraction function abs.

DEFINITION (Abstraction level). *Given a base theory T_0 and a simpler theory T_1 into which T_0 is abstracted, an* abstraction level, *denoted with λ_i, is the surrounding frame that contains T_i, which form a tuple $\langle \Lambda, \mathcal{T} \rangle$, where $\lambda_i \in \Lambda$ and $T_i \in \mathcal{T}$, and $\lambda_0 \preceq \lambda_1$.*

DEFINITION (Abstraction hierarchy). *Let $\mathcal{T}$ be set of theories, $\mathcal{F}$ denote a set of abstraction functions, and Λ the set of levels used with $abs_i \in \mathcal{F}$ on a theory $T_0 \in \mathcal{T}$, then an* abstraction hierarchy $H \in \mathcal{H}$ *is the ordered sequence of levels $\lambda_0, ..., \lambda_n \in \Lambda$, with $n > 1$, obtained from firing $abs_i \geq 1$ times successively on $T_0, T_1, ..., T_{n-1}$ such that $\lambda_0 \preceq \lambda_1 ....\lambda_{n-1} \preceq \lambda_n$ and $labs(T_0) = \lambda_0 ... labs(T_n) = \lambda_n$ hold.*

PROPOSITION 4.1. *Let E_j be contents of a $GL(x_j)$, and contents E_i of $GL(x_i)$ into which E_j is abstracted using abs_i, $E_j, E_i \in \mathcal{E}$, $RL(x_j, x_i)$, then GL is a type of* abstraction level.

PROPOSITION 4.2. *Let $\mathcal{E}$ be set of contents for each level $x_j ... x_1$, where $GL(x)$, $RE(x, y)$, and $GP(y)$, $\mathcal{A}$ denote a set of abstraction functions, and $\mathcal{L}$ the set of levels used with $abs_i \in \mathcal{A}$ on contents $E_j \in \mathcal{E}$, then GP is a type of* abstraction hierarchy $H \in \mathcal{H}$, *where $x_i ... x_1$ are obtained from firing $abs_i \geq 1$ times successively on $E_j, E_i, ..., E_1$ such that $grain(E_j) = x_j ... grain(E_1) = x_1$ hold and entity types in $E_j, E_i, ..., E_1$ relate through a relation φ, where $\varphi \rightarrow GR$.*

To improve abstraction functions as introduced in Ghidini and Giunchiglia (2003); Mani (1998), among others, and make a significant step toward their usability with ontologies, we use several DOLCE categories. In particular, we will use DOLCE's endurant ED for entity types, perdurant PD (continuant and occurent in BFO (2007), respectively, or endurant and occurent in GFO (Herre and Heller, 2006)), and PT as top-type that subsumes any other type (`owl:Thing` in OWL). With these preliminaries, we can proceed to the basic and compound abstraction functions.

4.3.2.1 Basic abstraction functions

The basic types of abstraction of the R-ABS type are the taxonomic and partonomic abstractions and refinements thereof, with the two distinctions compared to earlier works that they have constraints involving TOG components, GP, GL, RL, and GR, and are typed with ontological categories; see *Table 4.3*, first seven entries. They are straightforward granulation relation-turned-into-function abstractions along a hierarchy in the formal ontology or conceptual data model and conform to the main relations in the OBO Relation Ontology for bio-ontologies and the latest results on types of part-whole relations (Keet, 2006d; Keet and Artale, 2007; Smith *et al.*, 2005). Observe that a distinction has to be made between *subsumption* of processes, for which $abs_{isa}(\phi)$

Table 4.3: Overview of basic abstraction functions.

Abstraction	**Constraints; comment**
abs$_{isa}$: $PT \mapsto PT$	abs$_{isa}(\phi) = \psi$, $\phi \subseteq \psi$, $grain(\phi) = x_j$, $grain(\psi) = x_i$, $RL(x_j, x_i)$; Abstract a subtype into its taxonomic supertype
abs$_{ppo}$: $PT \mapsto PT$	as for abs_{isa}, but $ppart\_of(\phi, \psi)$; Abstract a part into its whole
abs$_{in}$: $PD \mapsto PD$	as for abs_{isa}, but $involved\_in(\phi, \psi)$; Abstract a part-process into the whole-process
abs$_{ci}$: $ED \mapsto ED$	as for abs_{isa}, but $contained\_in(\phi, \psi)$; Abstract a smaller contained type into larger type
abs$_{pi}$: $ED \mapsto PD$	as for abs_{isa}, but $participates\_in(\phi, \psi)$; Abstract an endurant into a perdurant
abs$_{mo1}$: $POB \mapsto SOB$	as for abs_{isa}, but $member\_of(\phi, \psi)$; Abstract a physical object into a social object
abs$_{mo2}$: $SOB \mapsto SOB$	as for abs_{isa}, but $member\_of(\phi, \psi)$; Abstract a social object into another social object
abs$_{d1}$: $PT \mapsto \emptyset$	$grain(\phi) = x_j$, $RL(x_j, x_i)$; Abstract a type into 'nothing', deleting it from the theory
abs$_{ph}$: $PT \mapsto \emptyset$	$grain(\phi) = x_j$, $RL(x_j, x_i)$, $\phi_i \Bumpeq \phi_j$ in x_i; Abstract types into indistinguishable, hiding it in the coarser-grained theory

suffices, and mereological sub-processes of the $involved\_in$ type with abs_{in}. For abstraction, it does not matter if the part-processes occur in sequence or in parallel, hence $involved\_in$ covers both possible subtypes (it does make a difference for expansion (§4.3.3)). The abs_{in} and abs_{ci} functions are specialisations of abs_{ppo}, thereby complying with the permitted granulation relations (§3.6.2) and *Proposition 4.2*. The five abstractions are applicable to **nrG** type of granularity; abs_{ppo} also can be used with **saoG**, and **samG**, abs_{mo1} and abs_{mo2} with **nacG**; abs_{isa} also with **nasG**. While it may be tempting to generalise these abstraction functions to an abstraction for any predicate, abs_p, this neither captures the different semantics nor is permitted by the TOG constraints, because then p could be any of the inverse relations too, which corresponds to expansion and not abstraction (*involves*, *has_part* etc.). However, in case one predicate is a subtype of another, then abstraction over predicates is possible by using abs_{isa} where ϕ and ψ denote predicates.

The basic abstractions differ only in the type of entity that is abstracted and relations that are common in domain ontologies, which one can extend trivially and thereby to refine the abs functions. Several such examples are given in *Example 4.4*, which is not meant to be exhaustive but illustrates possibilities.

Example 4.4. An indication of additional abstractions for formal ontologies is as follows. All examples are variants for part-whole abstraction (see also the part-whole relations in §3.5.1) and it might be useful to create subtypes of abs in an application.

- ⋆ Abstract non-agentive physical objects ($NAPO$) or amounts of matter (M) into amounts of matter (M), using (sub-)quantities; *e.g.*, Air and its M-part Oxygen or its $NAPO$-parts the types of molecule such as O_2 and CO_2.
- ⋆ Abstracting social agents (SAGs) like citizens into society (SC), or players and their sports team, locusts into swarm, and so forth for entities denoted with collective nouns and their members.
- ⋆ Mapping a series of processes (PRO) into one event (EV), where events can be accomplishments (ACC) or achievements (ACH), *e.g.*, Running into Marathon.
- ⋆ Also, it is possible to map PRO into states (ST). For instance, with a state Being sitting and lower level processes such as Breathing and Listening.[1] ♢

A commonality between all examples is that the different kinds of entities are not only related through a part-whole relation or a variant thereof, but for each example, belong to the same 'major branch' in DOLCE, as $NAPO$ and M are endurants (ED), so are APO, $NAPO$ and SC; PRO to EV and PRO to ST are all perdurants (PD); thus, $\phi \subset \psi$. Also, all abstract *one* kind of entity or instance into one other kind of entity or instance. These examples are more diverse and more precise than Mani (1998). Mani proposed abstraction operators from process to events, from process to object and from process to state (see also §5.4) without using ontological definitions for what processes and states are, where, on the other hand, well-established DOLCE definitions are used for typing the variables of the abs functions. It is straightforward to create many abstraction functions for ontologically sensible combinations of types of entities. I will not pursue this here; the current definitions are trivially extensible, even if one were to use a different foundational ontology.

The last two basic abstractions, abs_{d1} and abs_{ph}, address two of the three functions for D-ABS. With abs_{d1}, some type is deleted from the coarser-grained theory, which is generally more applicable for conceptual data models than ontologies, unless one wants to trim the ontology for performance reasons (*e.g.*, FMA-Lite, 2007; GOSlim, 2005); thus, $\phi \in E_j$, $\phi \notin E_i$, with $grain(E_j) = x_j$, $grain(E_i) = x_i$, $RL(x_j, x_i)$, and $E_j \subset E_i$. The second function, abs_{ph}, is not a deletion in the strict sense, but to make objects indistinguishable concerning their physical size at different levels of granularity. This is useful in particular for levels adhering to **sgpG**. Its sibling type of granularity **sgrG** can be adequately dealt with using the abs_{isa} or abs_{ci}, because the items on the maps are all of the type region (R), hence one can construct a taxonomy or partonomy of types of regions, like point and polygon, line and multiline, and abstract accordingly with $abs(polygon) = point$ and so forth, so that *e.g.*, a building is represented with a polygon on a more detailed map and as a point on a coarser map with lower resolution, and abstracting from sphere into circular bi-layer into circle into point.

A correct use of the basic abstraction operations partially depend on how the relations between the entities have been defined in the data source. Regarding taxonomies, a problem with the abs_{isa} function can arise when a taxonomy allows multiple inheritance, because unless specified, the software cannot know which subsumer to choose. If one were to allow multiple inheritance, then additional knowledge has to be specified to carry out abstraction correctly. Alternatively, one can expand on the abs function and let it return all subsumers, but this complicates both representation of the result and any successive abstractions and therefore should be avoided. A different complication may arise for abs_{ppo}, depending on the definition of the $ppart\_of$ relation in the data source. If it adheres to standard Ground Mereology, then abs_{ppo} does function as intended, but one also has to observe that at mereology behaves different at the type-level (Artale *et al.*, 1996a,b; Smith *et al.*, 2005, and footnote 19 in Chapter 3).

[1]This is different from the notion of 'state as snapshot', where a process is sliced up in an (in)finite amount of states, thereby making states part of a process. This view is taken by BFO (2007) foundational ontology. Either way, one stands into a *ppart_of* relation to the other, although it might benefit from more detailed specification of the linking between DOLCE and BFO.

With these basic abstraction functions, we have covered the most widely used relations to construct hierarchies in 'simple' formal ontologies. Compound abstractions are required to manage comprehension and visualisation of complex formal ontologies and to enable abstractions for formal conceptual data models.

4.3.2.2 Compound abstraction operations

The compound abstractions address F-ABS, which can be used for **nfG**-granulated perspectives, and the third function for D-ABS, where attributes are hidden; both will be addressed in this section.

Table 4.4: Common compound abstraction functions.

Abstraction	**Constraints; Comment**
abs$_{f1}$: $ED \times ED \mapsto ED$	abs$_{f1}(\phi, \psi) = \varphi$, where $grain(\phi) = x_j$,$grain(\psi) = x_j$,$grain(\varphi) = x_i$, and $RL(x_j, x_i)$; Abstract two endurants into another endurant
abs$_{f2}$: $PD \times PD \mapsto PD$	as for abs_{f1}; Abstract two perdurants into a perdurant
abs$_{f3}$: $ED \times PD \mapsto ED$	as for abs_{f1}, and *Constraint 4.1*; Abstract an endurant and a perdurant into an endurant
abs$_{f4}$: $ED \times PD \mapsto PD$	as for abs_{f1}, and *Constraint 4.1*; Abstract an endurant and a perdurant into a perdurant
abs$_{d2}$: $ED \times Q \mapsto ED$	as for abs_{f1}, and *Constraint 4.2*; Remove an attribute

One uses abs_{f1} to fold two kinds of ED into one; for instance, Blood cell (a $NAPO$) and Plasma (an M) are direct parts of Blood (an M). Although abs_{f1} mentions ED only, this is to be understood as ED *or any of its subtypes*, where the categories of ϕ and ψ must be subsumed by that of φ. This does not hold automatically for perdurants. Recollecting the ontological commitment that processes are part of states (*Example 4.4*) or events or objects (§5.4), this permits folding of a process and a state ($abs'_{f2} : PRO \times ST \mapsto EV$ to abstract, *e.g.*, Running and Being thirsty into Marathon)[2]. This also means that several complex abs functions cannot be recursively applied, because the output type is different from the input types. Another common type of folding is to combine processes and endurants into 'systems', like the Second messenger systems or a nutrient cycle in ecology, for which abs_{f3} can be used (or a refinement where $abs'_{f3} : ED \times PRO \mapsto ED$). This has its equivalent in the EER literature on clustering and abstractions (Jäschke *et al.*, 1993; Tavana *et al.*, 2007), where a relation or entity type is composed of entities and relations. For example "customer orders book", where the ordering process consists of several steps and entities involving, among others, Billing, Paying, Supplier, and Shipment (recollect *Example 2.2*). Both functions require a constraint, being that the input types have to be related to each other, ensuring that not two arbitrary types are folded, but ones that are related so that a connected subset of E_j is folded into a type in E_i, *i.e.*, upon firing abs_{f3} or $abs_{f4} \geq 1$ times for elements in E'_j, where $E'_j \subseteq E_j$, then E'_j is abstracted into φ where $\varphi \in E_i$.

CONSTRAINT 4.1 (folding). *For each ϕ, ψ where $abs_{f3}(\phi, \psi) = \varphi$ or $abs_{f4}(\phi, \psi) = \varphi$, $grain(\varphi) = x_i$, there must be either i) a predicate p such that $p(\phi, \psi) \in E_j$ that is contained in x_j or ii) $\phi = ED$, $\psi = p'$ and $\forall x(ED(x) \rightarrow \exists y(p'(x, y)))$ in E_j.*

Here, as with abs_{d1}, compositionality of the theory is important, which is a desirable feature from a computational viewpoint (Giunchiglia *et al.*, 1997; Pandurang Nayak and Levy, 1995). From the perspective of a domain expert, however, it is debatable, because some details in

[2]Although it may be argued that processes should fold into a state first because they are at finer-grained level than a state. A suggested counter example might be the process Staggering has as parts the process Walking and a state of Being drunk, but this combination is not *part of* but *causes* the staggering, and abstractions of causality is disallowed. It merits further ontological investigation, which is not pursued here.

the logical theory really may be undesirable to develop tractable systems biology simulations and to make ontologies usable for ontology-guided applications (Sontag, 2004; FMA-Lite, 2007; GOSlim, 2005). For F-ABS this can be managed effectively with the current abstraction functions in conjunction with the levels. φ in E_i (in x_i) is not a 'new' entity, but is an element of $\mathcal{DG}$; hence, soundness and completeness can be preserved. Likewise, for any particular software application, one always can remove finer-grained levels and its contents from $\mathcal{DG}$.

Last, we have abs_{d2} for the D-ABS type of abstraction, in addition to the simple deletion of an entity (abs_{d1}). As mentioned, a real deletion can be desired for implementations, but also suppressing details from a user interface that has its main benefit to make large models comprehensible and visually more appealing than cluttered diagrams (Campbell *et al.*, 1996; Keet, 2005b, 2007c; Tzitzikas and Hainaut, 2006). The latter can already be done through toggle features, which lets the user select displaying more or less relations, attributes, and so forth, like with the OntoViz plugin for the Protégé ontology development tool. This can be formally defined with an $abs'_{d2}(\phi, \psi)$, where attribute ψ (a quality Q in DOLCE) folds away. However, because *nothing changes to the underlying theory*, we would have $grain(\varphi) = x_i$, $\varphi = \phi$ and $E_j = E_j$ and $RL(x_j, x_i)$. Semantically, this is quite distinct from the previous abstractions, because in the case of *hiding* the attribute then $\phi = \varphi$ and quality ψ is not folded together with ϕ but simply suppressed from the view. As discussed in Chapter 3, the TOG prohibits that the same entity resides in different levels within the same perspective, therefore abs'_{d2} is invalid for the TOG. This is not to say that suppressing attributes is an irrelevant or incorrect operation in general, but one has to appreciate the distinction between creating an uncluttered graphical layout whilst maintaining the complete detailed model in the background, and performing an abstraction where one moves from a finer-grained level to a coarse-grained level of granularity—the focus is on the latter. Then we have two cases. First, when ψ is a necessary and sufficient condition, ic, of ϕ, then the deletion of an attribute implies $\phi \subseteq \varphi$, hence we could use taxonomic abstraction with abs_{isa}. This implies that the relations between ϕ and φ for abs_{d2} cannot be just any type of relation, but has to be is_a and $\phi \neq \varphi$ hold in the system (cf. hiding, where $\phi = \varphi$). Second, if ψ (a Q) is just some attribute of ϕ (the ED), like Address of the ED Person, then ED and Q in abs_{d2} must be related through a so-called "arbitrary attribute relation" α (for those languages that support attributes), like has address. This can be summarised in the following constraint on abs_{d2}.

CONSTRAINT 4.2 (deletion). *For each ϕ, ψ, φ where $abs_{d2}(\phi, \psi) = \varphi$, $grain(\varphi) = x_i$, either one holds when ψ is deleted:*

- *iff ψ is the necessary and sufficient condition ic of ϕ, then $\phi \subseteq \varphi$ and $abs_{isa}(\phi) = \varphi$ also hold;*
- *iff there is an attribute relation $\alpha(\phi, \psi)$, ψ is not an ic, then $abs_{f3}(\phi, \alpha) = \varphi$ also hold.*

Thus, only the deletion-abstraction fits with granularity and the TOG, but not that of hiding elements in abstractions.

4.3.2.3 Summary and solutions

Three distinct methods of abstraction were identified, which are: R-ABS where a granulation relation GR is converted into a function; F-ABS where multiple entities and relations are folded into a different type of entity; and D-ABS where semantically less relevant entities and relations are deleted. Nine basic and five complex abstraction functions were defined, which have finer-grained distinctions than other abstraction operations reported in the literature, are easily extensible, and make use of foundational ontological types for unambiguous specification. The hitherto informal abstraction levels were precisely defined and integrated with the TOG to provide consistency in the applicability and usage of the functions. This comprises usage of GL, GP for the abstraction hierarchy, GR for correspondence with abstraction functions, and types of granularity TG. Further, by having identified the abstraction functions at the conceptual level and mapped to their corresponding formalisation, it simplifies understanding, provides space for extensions with more abstraction functions, and makes them usable and reusable across implementations. Thanks to this approach, abstraction also has become scalable, is straightforward to implement, and amenable to automation.

4.3.3 Expansion operations

The inverse of abstraction sometimes is called "articulation" (Hobbs, 1985), but its antonym is 'expand' and the process can be described with 'to detail' or 'specify in more detail', therefore I use the term *expand* and *expansion* to denote the opposite direction of abstract and abstraction. Analogous to integration of abstraction level and hierarchy into the TOG, we have for expansion:

PROPOSITION 4.3. *Let* $\psi \in E_j$*,* $in\_level(\psi, x_j)$*, and* E_i *contents of* x_i *into which* ψ *is expanded using* exp_i*, then* GL *is a type of* expansion level.

PROPOSITION 4.4. *Let* $\mathcal{E}$ *be set of contents for each level* $x_1...x_n$*, where* $GL(x)$*,* $RE(x,y)$*, and* $GP(y)$*,* Υ *denote a set of expansion functions, and* $\mathcal{L}$ *the set of levels used with* $exp_i \in \Upsilon$ *on an element* $\psi_1 \in E_1$*,* $\psi_2 \in E_2$ *etc where the contents* $E_1, E_2, ...E_n \in \mathcal{E}$*, then* GP *is a type of* expansion hierarchy $H \in \mathcal{H}$*, where* $x_2...x_n$ *are obtained from firing* $exp_i \geq 1$ *times successively on* $\psi_1, \psi_2, ..., \psi_n$ *such that* $grain(\psi_1) = x_1...grain(\psi_n) = x_n$ *hold and* $\psi_1, \psi_2, ..., \psi_n$ *relate through any relation* φ*, where* $\varphi \rightarrow GR$*.*

The expansion functions are summarised in *Table 4.5* and discussed in the remainder of this section. The intention of an expansion function exp is to capture the process when a user clicks,

Table 4.5: List of pre-defined expansion functions for the TOG.

Expansion	**Constraints; Comment**
$\text{exp}_{isa} : PT \mapsto PT$	$exp_{isa}(\psi) = \phi$, $\phi \subseteq \psi$, $grain(\phi) = x_j$, $grain(\psi) = x_i$, $RL(x_j, x_i)$; Expand a type into its subsumers
$\text{exp}_{po} : PT \mapsto PT$	as for exp_{isa}, but $has\_ppart(\phi, \psi)$; Expand a whole into its parts
$\text{exp}_{in} : PD \mapsto PD$	as for exp_{isa}, but $involved\_in(\phi, \psi)$; Expand a whole process into its part-processes
$\text{exp}_{ci} : ED \mapsto ED$	as for exp_{isa}, but $contained\_in(\phi, \psi)$; Expand from a container into its containing types
$\text{exp}_{pi} : PD \mapsto ED$	as for exp_{isa}, but $participates\_in(\phi, \psi)$; Expanding from the perdurant into its participating endurants
$\text{exp}_{mo} : SOB \mapsto POB \cup SOB$	as for exp_{isa}, but $member\_of(\phi, \psi)$; Expanding a social object to its member (physical/social) objects
$\text{exp}_{d1} : \emptyset \mapsto PT$	*Constraint 4.3*; Reintroduce deleted particular
$\text{exp}_{ph} : \emptyset \mapsto PT$	$grain(\phi) = x_j$, *Constraint 4.4*; Expand from indistinguishable to distinguishable types
$\text{exp}_{f1} : ED \mapsto ED$	$\text{exp}_{f1}(\varphi) = \{\phi_1, ..., \phi_n\}$, where $grain(\phi_i) = x_j$, $grain(\varphi) = x_i$, and $RL(x_j, x_i)$; Expand one endurant into endurants
$\text{exp}_{f2} : PD \mapsto PD$	as for exp_{f1}; Expand one perdurants into perdurants
$\text{exp}_{f3} : ED \mapsto ED \times PD$	as for exp_{f1}, and *Constraint 4.6*; Expand an endurant into a perdurant and an endurant
$\text{exp}_{f4} : PD \mapsto PD \times ED$	as for exp_{f1}, and *Constraint 4.6*; Expand a perdurant into a perdurant and an endurant
$\text{exp}_{d2} : ED \mapsto ED \times Q$	as for exp_{f1}, and *Constraint 4.5*; Add an attribute

e.g., on a "cellular process" type of button labelled with Citric acid cycle, the components of the citric acid cycle are retrieved from the ontology. This is already possible to a limited extent with KEGG and SemTalk that have the finer-grained level hard-coded and pre-computed and thereby correspond to a rigid inverse of F-ABS. When drilling down in a taxonomy, exp retrieves the next level of detail with the subtypes of the selected supertype. Intuitively, one may be inclined to simply define the inverses of all abs functions defined in the previous paragraph. But can we indeed do this and retrieve correct results? No. exp's answer contains not one entity as does abs, but has to contain *all* entities it expands into—has as subtype, as part, it contains, it involves. Put differently, there is a 1:n relation between the input of exp and its output. A further difference is that exp not only has to retrieve its direct descendants, but also somehow re-introduce those finer-grained elements that were abstracted into 'nothing' in case of D-ABS abstractions. To achieve this, the previously introduced $getC$ function can be used as a function nested in an algorithm to use exp. That is, $exp_{isa}(\psi)$ can be defined in terms of selecting ψ at level x_i, retrieving the contents of x_j with $getC(x_j)$ (where $RL(x_j, x_i)$), and to check if there is an is_a relation between ψ and ϕ. An example of this approach was given in *Example 4.3* and a high-level algorithm to achieve the aim of exp_{isa} can proceed as shown in *Algorithm 4*. Steps

Algorithm 4 Expansion with the exp_{isa} function

```
Require: $\psi \Leftarrow someEntityType \in \mathcal{DG}$
procedure $exp_{isa}(\psi)$
 1: $x_i \Leftarrow grain(\psi)$
 2: $j = i + 1$
Ensure: $RL(x_j, x_i)$
 3: $E_j \Leftarrow getC(x_j)$ «use Algorithm 1»
 4: $selectE(\phi)$ and $in\_level(\phi, x_j)$
 5: while $subsumes(\psi, \phi)$ do
 6:     add $\phi$ to RESULT
 7:     if not all contents is checked then
 8:         $\phi \Leftarrow$ select another entity type from $E_j$
 9:     else
10:         $\phi \Leftarrow \top$
11:     end if
12: end while
13: return RESULT
```

2-4 serve for both conceptual clarity and ensure a correct query answer. This is because we do not know what other relations ψ may have, therefore the search space has to be delimited to the set of possible answers in exactly the finer-grained level, which is taken care of by RL and $getC$. In addition, $getC$ also takes into account the internal structure of the finer-grained level as specified in §4.2.3 and *Definition F.7*. Thanks to $getC$, exp will retrieve any additional structure the parts have in x_j, such as horizontal relations how the parts are connected, which for an exp_{in} can be part-processes in sequence, among others. The algorithm can be re-used for exp_{ppo} with a minor change in step 5, which has to iterate over "**while** $has\_ppart(\psi, \phi)$ **do**". The other expansion functions that are the analogues to the abstraction functions for R-ABS also follow the same pattern as provided for exp_{isa}.

The expansions as counterpart for the three D-ABS functions are less straightforward. First, reintroducing a deleted PT can only occur if abs_{d1} is used as well:

CONSTRAINT 4.3. *Let $grain(\phi) = x_j$ and $RL(x_j, x_i)$, then $exp_{d1}()$ must be the inverse of $abs_{d1}(\phi)$.*

This is different for $exp_{ph}()$, which has to rely on the characteristics of the particular finer-grained level and $getC$ to expand:

CONSTRAINT 4.4. *Let $grain(\phi) = x_j$ and $RL(x_j, x_i)$, then given a level x_i, $exp_{ph}()$ must use $selectL(x_j)$ and $getC(x_j)$ to expand into the distinguishable contents of level x_j.*

The third expansion function for reintroducing types does not depend on previously introduced functions, as the only thing that has to be examined is the difference between the relations ϕ has more than φ and take its range ψ that is expanded (see also *Constraint 4.2*):

CONSTRAINT 4.5. *Let $grain(\phi) = x_j$, $grain(\psi) = x_j$, $grain(\varphi) = x_i$, and $RL(x_j, x_i)$, then their difference, $\phi \setminus \varphi$, is ψ (and, implicitly, α).*

Regarding the expansion of the complex abstraction functions, the opposite direction of abs_{f1}, abs_{f2}, abs_{f3}, and abs_{f4} can be defined with an opposite direction of *Constraint 4.1* for exp_{f3} and exp_{f4} with the addition that the relation between ϕ and φ, implicit in *Constraint 4.1*, has to be included explicitly.

CONSTRAINT 4.6. *Let $grain(\phi) = x_j$, $grain(\psi) = x_j$, $grain(\varphi) = x_i$, $RL(x_j, x_i)$, $\delta \rightarrow GR$, then $\delta(\phi, \varphi)$ and either there is a predicate p s.t. $p(\phi, \psi)$ or a $\psi = p'$ and $\forall y(ED(y) \rightarrow z(p'(y, z)))$ to expand ψ in addition to ϕ.*

Correspondingly, in a granular perspective adhering to the **nfG** type, we can unfold—go down to a finer-grained level—by extending the algorithm in step 5 with

```
    while ($subsumes(\psi, \phi)$ or $has\_ppart(\psi, \phi)$ or $has\_participant(\psi, \phi)$ or
        $involves(\psi, \phi)$ or $contains(\psi, \phi)$) do
```

Other variations combining expansion and content retrieval in more elaborate methods or functions are possible too, but not elaborated on further here because they follow the same pattern as provided above.

4.4 Reasoning over a hierarchy of relations

The functions defined in the preceding sections assume a properly defined taxonomy of part-whole relations as defined in §3.5.1 so that logically and ontologically correct deductions are made. With current automated reasoners this is not necessarily guaranteed, however, because they may take into account only the syntax of the relation-subrelation but neither the relata nor the relational properties. The next two subsections deal with the FOL aspects of relational hierarchies and introduces a novel reasoning service for DL reasoners, named *RBox Compatibility service*, respectively.

4.4.1 Relation hierarchies in FOL

The relation-subrelation can easily be visualised as the example in *Figure 4.5*, where the relata are referred to as domain and range for the first and second argument, respectively. Formally, we then have for the depicted relation-subrelation $D_S(x) \rightarrow D_R(x)$, $R_S(x) \rightarrow R_R(x)$, and $S(x, y) \rightarrow R(x, y)$. As a minimum, however, it suffices that either the domain or range, or both, of S is subsumed by those of R. Regarding the TOG, we have, among others, $RL(x, y) \rightarrow s\_ppart\_of(x, y)$, and the taxonomy of part-whole relations that adhere to this notion of relation-subrelation. In addition, we also have that the properties of RL are further constrained compared to s_ppart_of; that is, for subsumed relations, it may be that we also can say something about the properties of the relations. For instance, acyclicity is subsumed by asymmetry, which is in turn subsumed by irreflexivity.

LEMMA 4.9. *If (binary) relation R is asymmetric, then it is also irreflexive.*

See, *e.g.*, page A-6 in (Halpin, 1989) for the proof. Likewise, it can be proven that an acyclic relation is also asymmetric.

LEMMA 4.10. *If (binary) relation R is acyclic, then it is also asymmetric.*

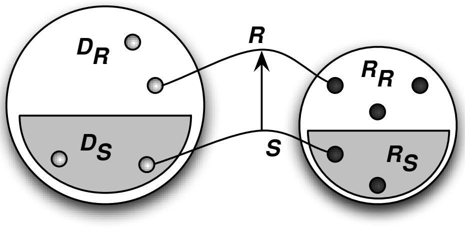


Figure 4.5: Parent (R) and child (S) relations with their relata so that $D_S \subset D_R$ and $R_S \subset R_R$.

Proof. Let a cyclic path be such that $\forall x_1...x_i...x_n(\varphi(x_1, x_i) \triangleq R(x_1, x_2) \wedge ... \wedge R(x_{n-1}, x_n) \wedge x_1 = x_i \wedge 1 \leq i \leq n)$ and acyclicity its negation: $\forall x_1...x_i...x_n \neg\varphi(x_1, x_i)$. Recollecting asymmetry $\forall x, y(R(x, y) \rightarrow \neg R(y, x))$, then the crux is in the path, because with asymmetry this holds for *same* and *adjacent* nodes, whereas with acyclicity this is extended to same, adjacent, *and related but non-adjacent* nodes thanks to the "$i \leq n$" in the path definition that permits $n \geq 1$ where $n \in \mathbb{N}$ (where $\mathbb{N}$ is the set of natural numbers). □

Consequently, we also can construct a hierarchy of relational properties, which has been depicted graphically in (Halpin, 2001) and partially proven in (Halpin, 1989). For instance, let the greek letter denote some relation and the superscript its relational property, then $\varphi^{ac} \rightarrow \psi^{as} \rightarrow \phi^{ir}$; hence, a relation that is acyclic cannot subsume one that is merely irreflexive. However, acyclicity requires a second order logic and not all other relational properties have been implemented in automated reasoners, therefore, we shall focus on the first part in the next section: ensuring that the relata of a sub-relation subsume the relata of its parent relation.

4.4.2 The *RBox Compatibility Service* for DL reasoners

The natural step is to ensure that typing of the relata (domain and range restriction) is done and used correctly, therefore we now focus on automated reasoning and we show that managing a taxonomy of relations requires a new reasoning service to make *ontologically* correct inferences over a logical theory. A decision procedure to guide the modeller was proposed by (Keet, 2006a), for which several implementation options were suggested: a cheat-sheet, drop-down box in a CASE tool to type the relation, or software-support for the decision procedure with questions and examples corresponding to each decision point. However, none of the suggested options can *compute* correctness, or at least logical consistency, and, as we will see in *Example 4.6*, neither can current DL-based automated reasoner.

We choose Description Logics to formulate the desired reasoning service. DLs have been shown useful for reasoning both over conceptual models like EER, ORM, and UML (Artale et al., 2003; Baader *et al.*, 2003; Berardi *et al.*, 2005; Calvanese et al., 1998b; Calvanese *et al.*, 1999; Keet, 2007a) and ontology languages such as the OWL suite (OWL, 2004, 2007) and DL-Lite family (Calvanese *et al.*, 2005, 2007). At present, most properties of the mereological theories can be represented in expressive DLs (Calvanese *et al.*, 2003; Horrocks *et al.*, 2006), except for antisymmetry (see also *Table 5.3* for a general comparison between DL languages).

For the current purpose, we are mainly interested in reasoning over roles given a role hierarchy (represented by the *Role Box*, RBox) and a DL-concept hierarchy (represented by the *Terminological Box*, TBox). For our purposes, it is enough to introduce briefly the simple $\mathcal{ALCI}$ DL language, which has already sufficient expressivity to support the reasoning service we are interested in; see *Appendix C.2* for a general introduction in DL languages. $\mathcal{ALCI}$ is a sub-language of both the proposed OWL 1.1, which is $\mathcal{SROIQ}$ with datatypes, and of the $\mathcal{DLR}$ family of DL languages (Calvanese *et al.* (2003)), which were specifically developed to provide a formal underpinning and unifying paradigm for conceptual modeling languages and permit automated reasoning over conceptual models. With respect to the formal apparatus, we will strictly follow the concept language formalism presented in Baader *et al.* (2003). Basic types of $\mathcal{ALCI}$ are *concepts* and *roles*. A concept—*sensu* DL—is a description gathering the common properties among a collection of individuals; from a logical point of view, it is a unary predicate ranging over the domain of individuals. Inter-relationships between these individuals are represented by means of roles, which are interpreted as binary relations over the domain of individuals. According to the syntax rules of *Figure 4.6*, *$\mathcal{ALCI}$ concepts* (denoted by the letters C and D) are built out of *atomic concepts* (denoted by the letter A) and *atomic roles* (denoted by the letter P). In the following we use $\exists R$ as a shortcut for $\exists R.\top$. As usual, an $\mathcal{ALCI}$ interpretation is a pair, $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $\Delta^{\mathcal{I}}$ is a non-empty set of objects (the *domain* of $\mathcal{I}$) and $\cdot^{\mathcal{I}}$ an *interpretation function* such that, for every concept C, and every role R, we have $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ and $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. Generic concepts and roles are interpreted by $\mathcal{I}$ according to the semantic equations of *Figure 4.6*.

A *knowledge base* in this context is a pair $\Sigma = (\mathcal{T}, \mathcal{R})$ where $\mathcal{T}$ is a set of *terminological axioms* (TBox) of the form $C \sqsubseteq D$ (general concept inclusion axiom), and $\mathcal{R}$ is a set of *role axioms* (RBox) of the form $R \sqsubseteq S$ (subrole axiom) and $R \sqsubseteq C_1 \times C_2$ (Domain & Range axiom). (Domain & Range axioms are a shortcut for the following two axioms: $\exists R \sqsubseteq C_1$, $\exists R^- \sqsubseteq C_2$, with C_1, C_2 generic concepts.) An interpretation $\mathcal{I}$ satisfies $C \sqsubseteq D$ iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, $R \sqsubseteq S$ iff $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$, and $R \sqsubseteq C_1 \times C_2$ iff $R^{\mathcal{I}} \subseteq C_1^{\mathcal{I}} \times C_2^{\mathcal{I}}$. A knowledge base Σ is *satisfiable* if there is an interpretation $\mathcal{I}$ which satisfies every axiom in Σ; in this case $\mathcal{I}$ is called a *model* of Σ. Σ *logically implies* an axiom α (written $\Sigma \models \alpha$) if α is satisfied by every model of Σ. A concept C (role R) is satisfiable, given a knowledge base Σ, if there exists a model $\mathcal{I}$ of Σ such that $C^{\mathcal{I}} \neq \emptyset$ ($R^{\mathcal{I}} \neq \emptyset \times \emptyset$), i.e. $\Sigma \not\models C \sqsubseteq \bot$ ($\Sigma \not\models \exists R \sqsubseteq \bot$), thus, role satisfiability can be reduced to concept satisfiability). We illustrate the DL syntax in the following example.

C, D	$\rightarrow$	$A \mid$	(atomic concept)	$A^{\mathcal{I}} \subseteq$	$\Delta^{\mathcal{I}}$
		$\top \mid$	(top)	$\top^{\mathcal{I}} =$	$\Delta^{\mathcal{I}}$
		$\bot \mid$	(bottom)	$\bot^{\mathcal{I}} =$	$\emptyset$
		$\neg C \mid$	(complement)	$(\neg C)^{\mathcal{I}} =$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
		$C \sqcap D \mid$	(conjunction)	$(C \sqcap D)^{\mathcal{I}} =$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
		$C \sqcup D \mid$	(disjunction)	$(C \sqcup D)^{\mathcal{I}} =$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
		$\forall R.C \mid$	(univ. quantifier)	$(\forall R.C)^{\mathcal{I}} =$	$\{a \in \Delta^{\mathcal{I}} \mid \forall b. R^{\mathcal{I}}(a,b) \Rightarrow C^{\mathcal{I}}(b)\}$
		$\exists R.C \mid$	(exist. quantifier)	$(\exists R.C)^{\mathcal{I}} =$	$\{a \in \Delta^{\mathcal{I}} \mid \exists b. R^{\mathcal{I}}(a,b) \wedge C^{\mathcal{I}}(b)\}$
R, S	$\rightarrow$	$P \mid$	(atomic role)	$P^{\mathcal{I}} \subseteq$	$\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
		R^-	(inverse role)	$R^{-\mathcal{I}} =$	$\{(a,b) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid (b,a) \in R^{\mathcal{I}}\}$

Figure 4.6: Syntax and Semantics for the $\mathcal{ALCI}$ Description Logic language.

Example 4.5. Take some knowledge base, then $part\_of$ is an atomic DL role, ED is endurant, PD perdurant, and PT particular (see ***Figure 3.5***), and omitting the uniqueness and object type reference modes because they are not relevant for the current purpose, then we have the four axioms `CarChassis` $\sqsubseteq$ `ED`, `Car` $\sqsubseteq$ `ED`, `CarChassis` $\sqsubseteq \exists$`PARTOF.Car`, and `Car` $\sqsubseteq \exists$`PARTOF`$^-$`.CarChassis`. Further, we can describe the domain and range restriction of `PARTOF`, one of its subtypes `INVOLVEDIN`, and its domain and range restriction as `PARTOF` $\sqsubseteq$ `PT` $\times$ `PT`, `INVOLVEDIN` $\sqsubseteq$ `PARTOF`, and `INVOLVEDIN` $\sqsubseteq$ `PD` $\times$ `PD`. The DOLCE categories `ED` and `PD` are both subconcepts of `PT` and are disjoint: `ED` $\sqsubseteq$ `PT`, `PD` $\sqsubseteq$ `PT`, and `ED` $\sqsubseteq \neg$`PD`. Given this knowledge base, it is clear that it is logically correct that the car chassis is `PARTOF` of the car and that it can never be `INVOLVEDIN` the car in the current state of the knowledge base. Indeed, both car and car chassis are subconcepts of `ED`, which in turn is a subconcept of `PT`, hence, `PARTOF` can be used. Furthermore, `INVOLVEDIN` is typed with

PD, *i.e.*, it can be used to relate perdurants only, but we have that ED and PD are disjoint, and car chassis and car are both types of ED, hence, declaring INVOLVEDIN for the car and car chassis would lead to a logical inconsistency. ♢

Reasoning over a role taxonomy means checking for role satisfiability and checking the compatibility of domain & range axioms with respect to subrole relations holding in the RBox. Role consistency is an issue when dealing with both RBoxes and TBoxes, since an unsatisfiable concept can be associated to either the domain or the range of a role leading to an inconsistent role. Reasoners, such as Pellet, FaCT, and Racer (Fact++, 2007; Pellet, 2006), can check role satisfiability by reducing it to concept satisfiability—i.e. by checking whether $\Sigma \not\models \exists R \sqsubseteq \bot$. The new reasoning service *RBox compatibility* checks the domain and range restrictions for roles against the RBox and the TBox. This new service aids avoiding unwanted logical consequences of an RBox over a set of concept hierarchies expressed in the TBox. We demonstrate with an example both the relevance of reasoning over an RBox and the problem with extant reasoning services that do not deal adequately—in the sense of deriving *ontologically* correct results—with a role hierarchy. We thus provide a formal definition for the RBox compatibility test and then show how this new reasoning service can help in avoiding ontologically undesired consequences. For the example, the ontology development tool Protégé v3.2 beta is used together with the Racer reasoner (the same was tested with SWOOP and Pellet, resulting in the same issues).

Example 4.6. Continuing from *Example 4.5*, let us take the three top-most categories of DOLCE and add them as classes in Protégé. Then add a role hierarchy with pwrelation at the top that subsumes part-of that in turn subsumes involved-in with their proper domain and range restrictions, as has been defined in §3.5.1. Then, relate Chewing to Eating with the involved-in relation, and Chassis to Car through the part-of relation. This is depicted in *Figure 4.7-A1* and *B* as screenshots from Protégé v3.2 beta. For illustrative purpose, we have created another class hierarchy, too, depicted in *Figure 4.7-A2*, where Chassis and Car are not subconcepts of endurant ED anymore, but of particular PT instead. Further, we have another scenario in *Figure 4.7-C* with an "incompatible" role hierarchy, which is, in this example, a role hierarchy where the domain and range restrictions are *inverted* compared to the correct scenario, since now part-of's subrole involved-in can be used to relate anything to anything whereas part-of itself can only relate perdurants.
Using the reasoning options of Protégé with Racer, we obtain the results as summarized in *Figure 4.8*. Choosing the TBox (A1) with the "correct" RBox (B) shows, as expected, that the ontology is fine. Testing the TBox (A1) with the "incompatible" RBox (C), it says Chassis is inconsistent, whereas using the TBox (A2) with the "incompatible" RBox (C) reclassifies Chassis as a type of perdurant. Although with relation to the scenarios using the RBox (C), these deductions are *logically* correct, the reasoner should have found a compatibility issue in RBox (C), because the domain and range restrictions of a subrole cannot be more general (higher up in the TBox) than those of its parent role. ♢

To define formally the new proposed *RBox compatibility* reasoning service, we start first with assuming that for each role there is provided exactly one user-defined Domain & Range axiom. Without loss of generality, we can assume that the axiom $R \sqsubseteq \top \times \top$ holds when an explicit Domain & Range axiom is lacking.

DEFINITION 4.7 (User-defined Domain and Range Concepts). *Let R be a role and $R \sqsubseteq C_1 \times C_2$ its associated Domain & Range axiom. Then, with the symbol D_R we indicate the* User-defined Domain *of R—i.e., $D_R = C_1$—while with the symbol R_R we indicate the* User-defined Range *of R—i.e., $R_R = C_2$.*

We now define the new reasoning service, *RBox compatibility*, that checks the compatibility of Domain & Range axioms with respect to both the role hierarchy holding in the RBox and the

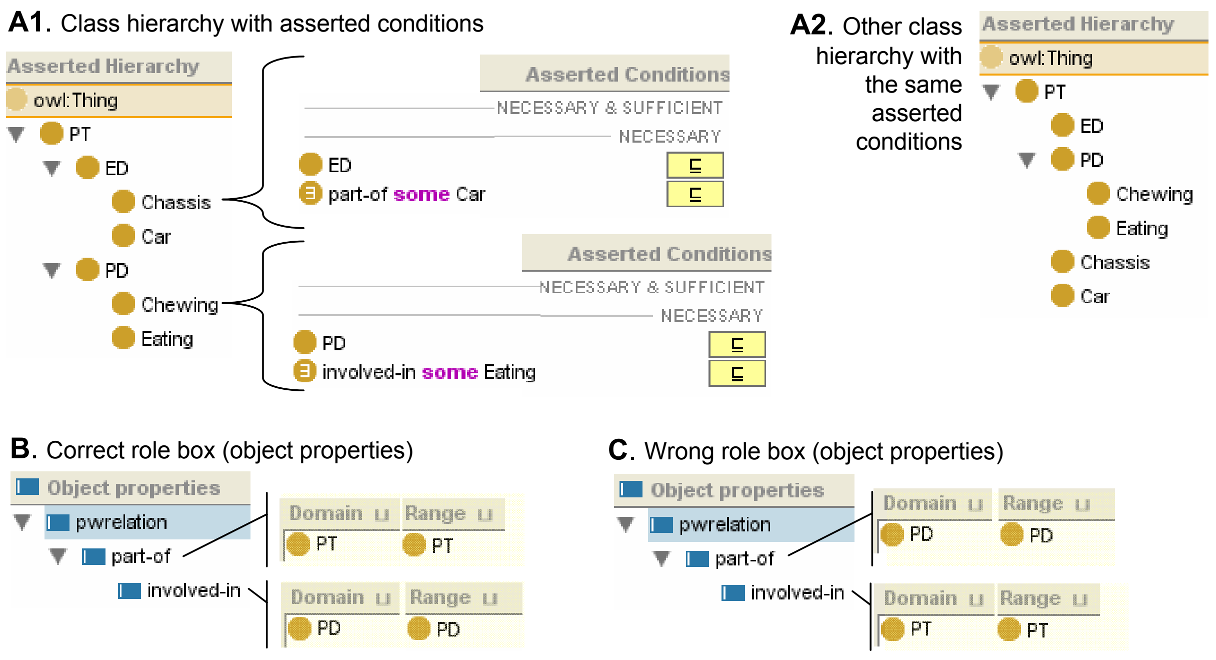
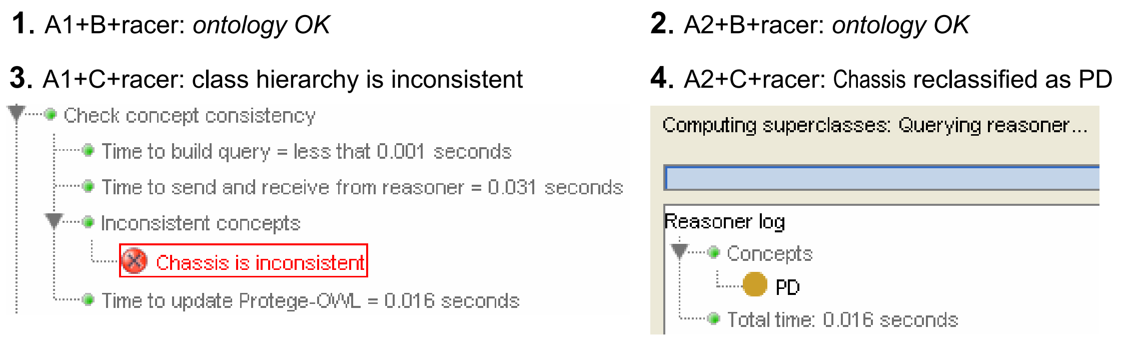


Figure 4.7: Two class hierarchies and two role (object property) hierarchies in Protégé.


1. A1+B+racer: ontology OK
2. A2+B+racer: ontology OK
3. A1+C+racer: class hierarchy is inconsistent
4. A2+C+racer: Chassis reclassified as PD

```
Check concept consistency
    Time to build query = less that 0.001 seconds
    Time to send and receive from reasoner = 0.031 seconds
    Inconsistent concepts
        Chassis is inconsistent
    Time to update Protege-OWL = 0.016 seconds
```

```
Computing superclasses: Querying reasoner...
Reasoner log
    Concepts
        PD
    Total time: 0.016 seconds
```

Figure 4.8: The examined four combinations of automated reasoning over the in *Figure 4.7* shown class hierarchies and object property (Role Box) hierarchies with the current results of the reasoner (Racer) when checking consistency and computing the inferred hierarchy (classifying the taxonomy).

concept hierarchy holding in the TBox. The tests of the RBox compatibility service are not only necessary but also sufficient for finding domain-range problems, because it covers each permutation of domain and range of the parent and child relation in the role hierarchy.

Definition 4.8 (RBox Compatibility). *For each pair of roles, R, S, such that $\langle \mathcal{T}, \mathcal{R} \rangle \models R \sqsubseteq S$, check whether:*
Test 1. $\langle \mathcal{T}, \mathcal{R} \rangle \models D_R \sqsubseteq D_S$ *and* $\langle \mathcal{T}, \mathcal{R} \rangle \models R_R \sqsubseteq R_S$;
Test 2. $\langle \mathcal{T}, \mathcal{R} \rangle \not\models D_S \sqsubseteq D_R$;
Test 3. $\langle \mathcal{T}, \mathcal{R} \rangle \not\models R_S \sqsubseteq R_R$.
An RBox is said to be compatible iff `Test 1` *and* (2 *or* 3) *hold for all pairs of role-subrole in the RBox.*

A formal conceptual model or domain ontology that does not respect the RBox compatibility criterion can be considered as *ontologically flawed*. Checking for RBox compatibility can be done by using the classical (for DL reasoners) subsumption reasoning service. One can define the following actions whenever the above defined tests fail. If `Test 1` does not hold, a warning should be raised that the domain & range restrictions of either R or S are in conflict with the role hierarchy and proposing either

(i) To change the role hierarchy or
(ii) To change domain & range restrictions or
(iii) If the test on the domains fails, then propose a new axiom $R \sqsubseteq D'_R \times R_R$, where $D'_R \equiv D_R \sqcap D_S$[3], which subsequently has to go through the RBox compatibility service (and similarly when `Test 1` fails on range restrictions).

If `Test 2` and `Test 3` fail, a warning should be raised stating that R cannot be a proper subrole of S but that they can be equivalent. In this case, there would be two options: either
(a) Accept the possible equivalence between the two roles or
(b) Change domain & range restrictions.

Outside the current setting of part-whole relations, the above actions should allow a user to leave unchanged both the TBox and the RBox, since both tests do not imply a logical inconsistency with respect to the logical theory. A demonstration of the application of the RBox compatibility service is included in the next example.

Example 4.7. Recollecting *Example 4.5*, observe that both Protégé deductions—*i.e.*, Chassis inconsistent in scenario `A1+C`, and Chassis as a perdurant in scenario `A2+C`—are just logical consequences of disregarding the ontological incorrect (but logically consistent) assumptions contained in RBox (`C`). Put differently: with the RBox compatibility service in place, one obtains these deductions by ignoring all warnings raised and suggestions proposed by the service.

Now consider RBox (`C`) with the RBox compatibility service, then `Test 1` fails because both the domain and range of involved-in, particulars PT, are parent concepts of its superrole (part-of) domain and range restrictions that are set to perdurants PD. Between the possible options (i)-(iii) that an user can choose, the second one is the most appropriate. In particular, the user can assign the correct domain & range restrictions to the roles (with respect to the part-whole taxonomy); thus, obtaining ontologically correct deductions[4].

For illustration, let us assume the user chooses option (i) instead of changing the domain and range restrictions. The result in the RBox is that the roles are inverted such that now part-of $\sqsubseteq$ involved-in, which contradicts the part-whole taxonomy, but is logically correct. `Test 2` and `3` then pass. Subsequent checking of the TBox (`A1`) will still yield an inconsistent Chassis, because the RBox is ontologically flawed. More precisely, Chassis is part-of Car and both are still subconcepts of ED, whereas part-of is typed with PD that is disjoint from ED; hence, Chassis as part of Car can never be instantiated.

Going along with option (iii) can, in this example, still yield an inconsistent theory. Choosing (iii), the following steps occur. A new axiom for involved-in is proposed for the domain restriction: with $D'_R \equiv D_R \sqcap D_S$ we have in the example $D'_R \equiv$ PT $\sqcap$ PD, which, with PD $\sqsubseteq$ PT, results in $D'_R \equiv$ PD, and upon accepting this proposal we get involved-in $\sqsubseteq$ PD $\times$ PT. The same sequence is repeated for the range restrictions, such that we have involved-in $\sqsubseteq$ PD $\times$ PD. Then `Test 2` and `3` can be executed. Recollecting, we have: part-of $\sqsubseteq$ PD $\times$ PD and involved-in $\sqsubseteq$ part-of. Thus, the compatibility reasoning service will suggest options (a) and (b) to the user. By choosing option (b) the user has the possibility to change the domain and range restrictions to those in RBox (`B`). If, on the other hand, the users accepts the option (a) both concepts Chassis and Car will be unsatisfiable.

Looking at the combination (`A2+C`), then upon choosing option (i), both Chassis and Car are re-classified from PT to PD to comply with the domain & range restriction of part-of. Choosing option (iii) together with option (a) will still result into a re-classification

[3]The axiom $C_1 \equiv C_2$ is a shortcut for the axioms: $C_1 \sqsubseteq C_2$ and $C_2 \sqsubseteq C_1$.

[4]Although the precise definition of 'ontologically correct' is a topic of active research efforts, it is thus far generally used to contrast it with obvious modeling errors (such as is_a vs part_of and mixing criteria for subsumption) and corrections made thanks to, *e.g.*, the OntoClean methodology (Guarino and Welty, 2004) or disambiguation of relations through usage of the Relation Ontology (Smith *et al.*, 2005).

of both Chassis and Car as subconcepts of PD. Although the logical theory with the reclassification is satisfiable, it is ontologically flawed both regarding the role hierarchy and the concept hierarchy, and again, selecting option (ii) to correct the domain & range restrictions to that of RBox (B) is the appropriate choice. ♢

Thus, with the currently available reasoners, one may get error messages about inconsistent concepts or undesired equivalences and/or subsumptions between concepts, whereas the error is in the role hierarchy.

Relations/properties/roles are essential components in both conceptual models and ontologies, which receives a more prominent place when role hierarchies and domain & range restrictions are properly declared, as with the part-whole relations taxonomy and the granulation relations. Reasoning over the logical theories requires inclusion of reasoning over both concepts and roles to check if they are consistent. It was demonstrated how ontologically unwanted logical implications in the concept hierarchy can be avoided with the additional RBox compatibility service on the role hierarchy. This holds, of course, also for role hierarchies other than the part-whole taxonomy and can be used to ease conceptual modeling and domain ontology development in general so that it results in a representation that is closer to the real world semantics it intends to represent.

4.5 Granular reasoning: combining tasks and functions

The functions introduced in the previous paragraphs provide the machinery to carry out complex tasks for managing granulated data, information, and knowledge. In particular, to achieve interesting and effective granular reasoning, two topics can be considered: granular reasoning over $\mathcal{DG}$ and granularity in the execution of the reasoning. The former concerns types of elaborate questions a user may want to ask the data source, whereas the latter involves how to decompose the queries into finer-grained sub-tasks, *i.e.*, granular computing for structured automated problem solving. To appreciate the prospects of the latter in §4.5.2, an example of complex granular reasoning with TOG functions will be given in §4.5.1.

4.5.1 Complex reasoning scenarios for the subject domain

One cannot anticipate the full range of complex granular queries a user would want to have answered, but two types of complex conditional selections are likely to occur more frequent: i) Selecting entities (/types) residing in one or more levels and retrieving the contents of another level; *i.e.*, a constrained selection and retrieval such as with the cells in blood (*Example 2.5*); and ii) selecting one entity residing a particular level, retrieve contents from multiple levels; *i.e.*, perform a conditional intersection. The first option will be analysed through an extended example about the liver and Second Messenger System to demonstrate usage of the TOG functions. An algorithm for the second option is presented afterward.

4.4.1.1 Extended granular reasoning example

The aim is to demonstrate use of the functions introduced in the previous paragraphs by answering the query "*which hormone(s) bind(s) to the SMS receptor in the liver?*". This will be done with three methodologies, being step-by-step, brute-force, and by assuming a knowledgeable user. Before doing so, several assumptions are given.

Assumptions and sources. Assume we have a $\mathcal{DG}$ at our disposal with several granular perspectives defined that contain two or more granular levels each. These perspective include, but are not limited to, structural (gp_1) spatial (gp_6) and functional (gp_2) human anatomy, structure (gp_3) and function (gp_4) of molecules, and biological pathways (gp_5). $\mathcal{DG}$ is populated with data sources such as the FMA (2003), GO (GOC) (optionally, with a Second Layer (Myhre *et al.*, 2006) for relations between the three sub-ontologies), and KEGG, which have their entities related in taxonomies, partonomies (structural and containment), and folding of entities and relations such that they do not contradict the TOG constraints. All TOG functions listed in *Table 4.1, 4.3, 4.4, 4.5* are at our disposal in this $\mathcal{DG}$.

Finding the answer. As starting point, it is assumed that the user posing the query is ignorant about the biology domain or is a biologist who does not trust the biological information stored in the software application, hence, s/he also wants and has to be informed about intermediate steps of the reasoning procedure. To decompose the query into intermediate steps, we first retrieve information about the three main entity types in the query—hormones, receptors, and liver—and then combine the information. Subsequently, a 'brute force computation' approach is illustrated that does not need any user intervention, and, last, a 'knowledgeable user' scenario where with input from the domain expert, shortcuts can be taken. The approaches are compared after the three scenarios.

1. *What is liver?* Retrieve level(s) of liver with $grains$(Liver) to retrieve {$\mathsf{gp_1gl}_i$, $\mathsf{gp_2gl}_j$, ...} and successively select each level for further examination using $selectL$:
   - Structurally, it is an organ, because it resides in $\mathsf{gp_1gl}_i$ = Organ contained in $\mathsf{gp_1}$. Alternatively, abs_{isa} can be used several times from Liver up to its root entity Organ.
2. *What are hormones?* Retrieve level(s) of hormones with $grains$(Hormone) to retrieve {$\mathsf{gp_1gl}_m$, $\mathsf{gp_2gl}_n$, $\mathsf{gp_3gl}_c$, ...} and select each level for further examination using $selectL$:
   - In $\mathsf{gp_1}$, the highest common subsumer of Hormone, *i.e.* its root entity reached with abs_{isa}, is Molecule, *i.e.*, it resides in the Molecule-level $\mathsf{gp_1gl}_m$.
   - In gp_2, a hormone is an activator of a pathway; knowing that Hormone resides in the function perspective is useful in point 3.
   - $grains$(Hormone) retrieves more than one level from a perspective $\mathsf{gp_3}$ filled with a chemicals ontology, because a hormone can be a Fatty acid derivative, Amino acid derivative, Peptide, or Steroid.
3. *What is SMS receptor?* Like in points 2 and 3, retrieve level(s) of SMS receptor, then:
   - In the function-perspective gp_4, it is a 'lock'.
   - In $\mathsf{gp_1}$ (the FMA), the highest common subsumer of Receptor is Molecule, *i.e.* it resides in the Molecule-level, but, in more detail,
   - The set of levels return the Protein-level in $\mathsf{gp_3}$, and Protein is a Molecule.
   - To find the whole that SMS receptor is part of, we perform an abs_{ppo}(SMS receptor) in $\mathsf{gp_2}$. This returns Second Messenger System at the immediate coarser-grained level, henceforth abbreviated as SMS.
   - Because SMS receptor is part of the SMS, we need to know the type of granulation of gp_4 and gp_5 to determine if it is of type **nrG** using $ppart\_of$ as GR or of type **nfG**. Depending on the data source, we can query the $has\_granulation(x, \phi)$ relation, use the tgP function, or use expansions because if **nrG** then exp_{ppo}(SMS) returns the structural parts of the system and if **nfG** then exp_{f3}(SMS) returns the entity types with their direct relations. As it appears, $\mathsf{gp_5}$ is granulated with **nfG** and $\mathsf{gp_4}$ with **nrG**.
   - $\mathsf{gp_5}$ being **nfG**-granulated, this implies that all entities that are directly related to SMS receptor in the Molecule-level and in the answer of exp_{f3}(Second messenger system) are also part of SMS. One could have guessed this, because we have observed in point 2 that the types also reside in a function perspective and functions generally have some coordination involving other types and relations.
4. *How do the SMS receptor and liver relate?* Assume that they relate and verify if there is indeed a relation between them by following the path from SMS receptor upwards to Liver. We have to determine through which type of GR.
   - Knowing that the SMS receptor is a protein and in the Molecule-level (point 3) and that Liver is an organ in the Organ-level (point 1), and they are both in $\mathsf{gp_1}$, tgP = **nrG**, and then $grP = ppart\_of$.
   - Then we traverse $\mathsf{gp_1}$ upwards with consecutive abs_{ppo} from Protein to Cell membrane to Cell to Tissue to Organ so that the last abs_{ppo} has Liver in the answer. Note that the last constraint, final $abs_{ppo}(x)$ = Liver, is important because we are not interested in any other

structures that SMS receptor may be part of (such as Intestine or *D. discoideum*, which is a type of organ and type of cellular slime mould, respectively).

5. *What about hormones relating to receptors?* The original query states that a hormone "binds" to a receptor, thus there must be a relation between the two.
   - Given that we now know that SMS receptor is a molecule, that Hormone is a molecule, that SMS receptor occurs in a level of gp_5 that has granulation **nfG**, and that it is in the adjacent finer-grained level of SMS, we retrieve the contents of the level that SMS receptor resides in with a rule 'if gp_5 then $grain$(SMS receptor) = gp_5gl_4' and $getC(gp_5gl_4)$ to verify a relation exists between the two types. Examining the contents, then indeed, Hormone binds to SMS receptor. Alternatively, we can fire a rule that verifies that exp_{ppo}(SMS) $=$ Hormone is true (finer-grained steps in the execution are omitted).
   - From the previous step, we can infer that Hormone is also part of SMS. However, this is a weaker result than obtained with examining the $getC$ result. A cross-check can be done: if exp_{f3}(SMS) $=$ SMS receptor $\wedge$ binds and exp_{f3}(SMS) $=$ Hormone $\wedge$ binds and abs_{ci}(SMS receptor, Hormone) $=$ SMS then SMS receptor binds Hormone.
6. *Which hormones bind to SMS receptor?*
   - Perform an expansion on Hormone by using first $selectE$(Hormone) on the retrieved content and then exp_{isa}(Hormone). This returns the set with the hormones Calcitonin, Chorionic gonadotropin, Corticotropin, Epinephrine, Follicle-stimulating hormone, Glucagon, Luteinizing hormone, Lipotropin, Melanocyte-stimulating hormone, Norepinephrine, Parathyroid hormone, Thyroid-stimulating hormone, Vasopressin (Stryer, 1988). Alternatively, take SMS and do an exp_{f3} to, say, gp_4gl_3 and take the subtypes of Hormone of the Hormone-level in gp_3, thus $intersect$(Hormone, gp_4gl_3), to retrieve the 13 types of hormone that can bind with the SMS receptor.
7. *Which hormones are in the liver and bind to SMS receptor?* The final step requires integration of the result obtained in the previous 6 steps.
   - Because SMS receptor is a structural part of Liver and Hormone is a Molecule that binds to SMS receptor, then also some hormone must be at least located in Liver (but a hormone is not necessarily structural part of the liver). Take the answer set of step 6 with the 13 types of hormone $x_1 ... x_{13}$ and fire for each x_i the abs_{ci} repeatedly up to the Organ-level in gp_6. Where $abs_{ci}(x_i) = y_1 ... abs_{ci}(y_n) =$ Liver, then x_i is in the answer.
8. **The answer**: The hormones binding to the SMS receptor in the liver is the answer where the abs_{ci} iteration in point in 7 evaluates to true, which are Glucagon and Epinephrine.

The second method is brute force without any user intervention by retrieving all hormones, all SMS parts and all entities that are contained in the liver, and to intersect the three sets:

```
1.  $grains(Hormone) = \{x_1, ..., x_n\}$ // Retrieve levels where Hormone resides.
2.  $selectL(x_i)$ // $x_i$ is the level in a functional perspective.
3.  $getC(x_i) = \{y_1, ... y_n\}$ // Retrieve all subtypes of Hormone.
4.  $grains(SMSreceptor) = \{z_1, ..., z_n\}$ // Levels where the SMS receptor resides.
5.  $selectL(z_i)$ // $z_i$ contains the molecule parts of Second messenger system.
6.  $getC(z_i) = \{w_1, ... w_n\}$ // Retrieve the entities & relations of the Second messenger system
7.  $grains(Liver) = \{u_1, ..., u_n\}$ // Set of levels where Liver resides.
8.  $selectL(u_i)$ // $u_i$ is the level in a structural perspective.
9.  $getC(u_i) = \{t_1, ..., t_n\}$ // Get the contents of the Organ-level.
10. $selectE(Liver)$ // Liver is one of the entities in $\{t_1, ..., t_n\}$.
11. $exp_{ci}(Liver) = \{s_1, ..., s_n\}$ // Retrieve all entities that are contained in Liver.
12. $intersect(x_i, z_i) = \{v_1, ..., v_{13}\}$ // Retrieve the overlap in contents of levels $x_i$ and $z_i$
13. $intersect(v, s) = \{glucagon, epinephrine\}$ // Retrieve the overlap between sets $v$ and $s$,
        // which returns the answer set containing Glucagon and Epinephrine.
```

This sequence gives the correct answer, but misses the reasoning *why* this is the correct answer and therefore may not convince the doubting biologist. Merging the biology aspects with the functions, we have a knowledgeable user who knows at least part of the two approaches. This enables the user to obtain the query answer quicker with the available TOG functions. Several strategies are possible, but the one with the least amount of steps is as follows:

```
1.  $grains(SMSreceptor) = \{x_1, ..., x_n\}$ // Set of levels where the SMS receptor resides.
2.  $selectL(x_i)$ Level $x_i$ contains the molecule parts of the Second messenger system.
```

3. $getC(x_i) = \{y_1, ...y_n\}$ // Retrieve entities & relations of the Second messenger system.
4. $selectE(Hormone)$ // An entity from the set $\{y_1, ...y_n\}$.
5. $exp_{isa}(Hormone) = \{z_1, ..., z_{13}\}$ // The 13 types of Hormone that bind to the SMS receptor.
6. $abs_{ci}(z_i) = Liver$ // For each z_i repeat $abs_{ci}(z_i)$ until it is abstracted into Liver.
   // This gives the answer $\{glucagon, epinephrine\}$.

Although the third option contains less steps here, this does not imply it will have a shorter query execution time to retrieve the answer from the $\mathcal{DG}$.

Discussion. Assessing the three methodologies, the first one roughly conforms to a biologist's manual reasoning process to find the answer if one would *not* have the relevant knowledge in a software system, whereas the third one is then a shortcut omitting 'the obvious'. However, there is not a specific body of knowledge each biologist knows, therefore several short-cut strategies alike the third method are possible. Supporting complex reasoning, then, amounts to giving a user high flexibility for querying the information source. The functions defined in §4.2 and 4.3 were sufficient to find the answer for each method.

4.4.1.2 Conditional selections

Regarding the second type of complex conditional selections, there are two options: i) find information about the selected entity, and ii) targeted conditional selections about its coarser/finer-grained entities. The former has been introduced in *Example 3.8* with bacteriocins and an informal example of the latter was given in *Example 2.5* about parts of blood. With the functions at our disposal, the information can be retrieved as follows.

Example 4.8. Assuming a granulated FMA, then one can perform the conditional selection "for the entity Blood, retrieve its parts at the cell level and their functions" to select Blood with $selectE$(Blood), expand with exp_{ppo}(Blood) into adjacent level $gp_i gl_i$ and intersect that with the contents of the Cell-level with $intersect(gp_i gl_i,$ Cell$)$. Then for each blood cell type in the intersection, do $grain(x)$ in gp_2 = Human functional anatomy (Cook *et al.*, 2004; Johansson *et al.*, 2005) to retrieve the blood cell's function. ♢

Abstracting from the examples, these conditional selections have a structure like "given entity x, find its structural parts that are smaller than a" and optional other clauses like "and have a function". This is defined more precisely in *Algorithm 5*. Noteworthy is step 4, where the granulation relation is retrieved from the domain granularity framework to select the right exp or abs in step 7 or 14, respectively.

As the examples demonstrate, one can perform complex querying and reasoning with the TOG functions for both the granularity components, and abstraction and expansion functions for its contents. Aside from implementation issues, one can structure its execution to enhance any implementation, which is described in the next section.

4.5.2 Granularity in execution of granular reasoning

Likewise that there is granularity in the data, information, and knowledge of the subject domain, this also applies to the approach toward execution of granular reasoning, which is also called granular computing for structured problem solving by Yao (2005b, 2007b). This is touched upon in this section.

Approach toward granular reasoning. Coarser- and finer-grained details of the approach toward granular reasoning were mentioned in the previous paragraphs regarding nested functions. Another aspect is that of 'drilling down' into more detail from the conceptual realm to the logical specification and more elaborate encoding in the 'physical' implementation (that is, the

Algorithm 5 Conditional selection and retrieval of finer- or coarser-grained contents

```
Require: $\psi \Leftarrow AnEntityTypeOfInterest \in \mathcal{DG}$
1: $selectE(\psi)$
2: $X \Leftarrow grains(\psi) = \{x_i, ..., x_k\}$
3: $selectL(x_k)$ from $X$
4: $\alpha \Leftarrow grel(x_k)$ «retrieve $GR$ of $x_k$»
5: print "retrieve finer-grained information?  type yes/no:"
6: if yes then
7:     select appropriate $exp$ based on $\alpha$
8:     $\Theta \Leftarrow exp_\alpha(\psi) = \{\theta_1 ... \theta_n\}$
9:     $selectE(\theta_i)$ from $\Theta$
10:    $grain(\theta_i) = x_{k+1}$
11:    $\Upsilon \Leftarrow getC(x_{k+1}) = \{\upsilon_1 ... \upsilon_n\}$
12:    RESULT $\Leftarrow$ intersect $\Theta$ and $\Upsilon$
13: else «no, retrieve coarser-grained information»
14:    select appropriate $abs$ based on $\alpha$
15:    RESULT $\Leftarrow abs_\alpha(\psi) = \varphi$
16: end if
17: return RESULT
```

software program code, queries). Each layer is independent of the next lower level, but not vice versa: a function defined at the conceptual level can be mapped into its logical representation in different formal knowledge representation languages (FOL, DL, etc.), and in turn can be mapped onto various implementation language statements (SQL, STRUQL, Java, C++, etc.) as queries or methods. This approach ensures portability of the intension of the functions and thereby guarantees shareability and reusability; hence, also system integration and linking because they all adhere to the same underlying conceptualisation. The following example demonstrates this for the retrieval of the content of a level.

Example 4.9. At the conceptual level, we have $getC$ to retrieve the contents of a particular level $gp_i gl_i$. This was defined at the logical level in FOL with (F.7) and can be translated into the various programming and query languages. The final code depends not only on the programming language, but also on the type of granularity. Take **nasG** type of granularity where retrieving an unordered set suffices, a table of the perspective gp_i named Perspective_i that has a column for each level it contains, named GLevel_1, ..., GLevel_n, then retrieving the contents of a level corresponds to a projection, written in relational algebra as:

```
project Perspective_i over GLevel_i
```

The equivalent in relational calculus is:

```
Range of P is Perspective_i
Produce P.GLevel_i
```

We have for a relational database table `perspective_i` the following SQL query:

```
SELECT GLevel_i FROM Perspective_i ;
```

If, on the other hand, the contents of the applied domain granularity is stored as an XML file called `gran.xml`, this query is required for XQuery:

```
doc("gran.xml")/Perspective_i/GLevel_i
```

Such translational aspects from TOG to implementation will be discussed §5.5. ♢

Granularity in the execution. $task^5$ consists of a composition of sub-tasks: combining a domain granularity framework d_i^f with its data source $\mathcal{DS}$ into $\mathcal{DG}$ and carrying out a level selection. As such, it is not interesting from a formal perspective, but is useful for developing efficient algorithms to improve performance of a software system with applied domain granularity. This is not the current focus, but may be an interesting avenue for further research. A shortcut for implementing $task^5$ is illustrated in Keet (2006b), but it is neither scalable nor generic enough to ensure consistency across different implementations. The main issue with $task^5$ is that to address it fully, one has to commit to a particular implementation scenario, which is not pursued here. Generically, it is possible to decompose the tasks into several nested functions with as prime advantage the possibility for modular & granular query evaluation by using recurring patterns, which, in turn, can improve efficiency of query evaluation. In addition, this offers reusability because a function like $getC$ can be reused also for other compound functions, hence the code for the function can be reused as well.

4.5.3 Summary

Several scenarios for complex granular reasoning were discussed, which are based on combining basic functions in a variety of sequences. In particular, the functions of the TOG (§4.2 and 4.3) suffice for elaborate granular reasoning in different core scenarios, such as the ignorant user or untrusting biologist as well as quick access for the knowledgeable user. Further, the layering of operations—*i.e.* granularity in the problem solving of granular reasoning—that was introduced in §4.2 was continued with a demonstration how conceptually and logically defined functions can map onto various query languages. This layering ensures portability of the intension of the functions throughout the diverse possible implementations.

4.6 Chapter summary

Fourty-six functions for querying granulated data sources were defined, of which the first set focussed on usage of TOG-components to retrieve information from a domain granularity framework and the second set covers abstractions and expansions using the granulated data. These functions were defined at the conceptual level, *i.e.*, the goal each function aims to achieve, and at the logical level so as to give precise semantics to the functions. With these functions, $task^1$, $task^2$ and $task^4$ we solved, from which the solution to $task^3$ followed as well.

Last, an example was given that not only used most of the functions, but also illustrated the different user- and computation-scenarios toward granular querying. The example informally introduced granularity in the reasoning itself, both with respect to finer-grained implementation details (the 'physical' layer) and in query evaluation strategies where several complex functions can be built up from the simple ones.

4.6.1 Meeting the requirements and tasks

Returning to the requirements and tasks set out in the introduction (§4.1), they have been addressed fully as follows.

1. A set of functions for the TOG *with which one can: (i) Query the* TOG *components—domain, perspective, and level—directly; (ii) Retrieve content of levels; (iii) Move between levels from coarser to finer-grained entities (/types) and back.*
The 19 functions to query the TOG components and to retrieve content of levels are described in §4.2 and summarised in *Table 4.1*. Moving between levels from coarser to finer-grained entity types and back has been addressed with the abstraction and expansion functions (§4.3), which are summarised in *Table 4.3* for the basic and in *Table 4.4* for the complex abstractions, and *Table 4.5* for the expansion functions.

2. Content retrieval has to make use of the existing structure in the original data source and consider the type of granularity that is used for the granulation.
The main function for content retrieval is $getC$ (*Definition F.7*). §4.2.3 elaborates on nested functions that are required for each leaf type in the taxonomy of types of granularity in order to deal with preserving the structure of the level contents. The structure can be accessed through using the tgL function (*Definition 4.3*) that is part of *Algorithm 1*.

3. Abstraction functions have to enable moving from finer-grained entities (/types) to a coarser-grained simplification, which may reside in an adjacent coarser granular level or higher up in a level in the same perspective.
Integration of the abstraction functions with the TOG is ensured with *Proposition 4.1* and *4.2* to relate the abstraction levels to granular levels and abstraction hierarchy to granular perspective.

4. Expansion functions have to enable moving from coarser-grained entities (/types) to finer-grained detail, which may reside in an adjacent finer granular level or lower down in a level in the same perspective.
Integration of the expansions functions with the TOG is ensured with *Proposition 4.3* and *4.4* to relate expansion levels to granular levels and expansion hierarchy to granular perspective. To ensure correct behaviour of the expansion functions, *Constraints 4.3-4.6* were added and a sample algorithm provided (*Algorithm 4*).

D. The purposes of the TOG's usage—dynamic aspects with primary distinctions allocating/classifying entities (/types) and information retrieval & reasoning—shall not affect the static structure of a theory of granularity.
First, all functions introduced in this chapter use TOG components but none changes any of its properties, and, second, no extra 'patchwork' TOG-components are needed for the functions to behave as specified.

$task^1$: *Perform a selection of levels from a particular domain granularity framework* d_i^f, i.e. $task^1(d_i^f) \rightarrow lss_i$, *where* $\mathcal{DL}$ *is the set of levels within that domain granularity framework and* lss_i *the selected subset.*
Whereas functions (F.2, F.4) meet a limited case of $task^1$—l_i is used for only one level within a pre-selected perspective— and $selectLs$ for multiple levels within one perspective (*Definition F.5*), it is addressed fully with the $selectDL$ function (*Definition F.6*) that returns the set of selected levels, lss_i, in the d_i^f. Observe that $task^1$ consists of an iteration of three nested functions, which are $selectP$, $getL$, and either $selectL$ or $selectLs$.

$task^2$: *To retrieve contents of at least one level, we have* $task^2(gl_i) \rightarrow \mathcal{E}$, *where* $\mathcal{E}$ *denotes the contents of a granular level, that is, entities or types and, where applicable, any further structure other than an unordered set.*
This task can be executed with the $getC$ function (*Definition F.7*) and its supporting nested functions to retrieve the type of granularity that a level adheres to, tgL (*Definition 4.3*), and the specific required type of procedures for each leaf type of granularity as outlined in §4.2.3 to ensure contents can be retrieved whilst preserving the structure of the entities (/types) in the level. In addition, usage of the functions is proposed in *Algorithm 1*.

$task^3$: *A formalised relationship or transformation rule is required between* $\mathcal{DS}$ *and* $\mathcal{DG}$ *to utilise them both for some particular reasoning task. Therefore,* $task^3(\mathcal{DS}, \mathcal{DG}) \rightarrow \mathcal{DS}$ $related\_to$ $\mathcal{DG}$, *where the "*$related\_to$*" has to be specified. Likewise, the relation between* $\mathcal{DS}$ *and its selected subsets,* i.e. $\mathcal{DG}^1$, *...,* $\mathcal{DG}^n$, *has to be specified.*
This and other relations between subsets of $\mathcal{DG}$ are summarised in *Table 4.2* and were proven in *Lemmas 4.1-4.8* with additional *Corollaries 4.1-4.4*. Task-specific: $\mathcal{DS}$ is a proper subset of $\mathcal{DG}$

(*Lemma 4.1*).

$task^4$: *Use a combination of levels of the same or different perspectives on the data source to which a granularity framework has been applied, $\mathcal{DG}$. The result is a subset of $\mathcal{DG}$, $\mathcal{DG}'$; thus, the task is* $task^4(\mathcal{DG}, \mathcal{DL}) \rightarrow \mathcal{DG}'$, *where* $\mathcal{DG}' \subseteq \mathcal{DG}$.
This task builds upon $task^1$ and $task^2$, hence, uses the above-mentioned functions, the algorithm for content retrieval (*Algorithm 1*) and, optionally, functions for further processing, such as $intersect$ for intersection of contents of different levels (*Algorithm 2*). The combination of these inputs is presented in *Algorithm 3*. Procedures that are more sophisticated can be composed when also entity (/type) selection, abstraction, and expansion are at one's disposal for conditional selection and retrieval. This was demonstrated in *Algorithm 5* and in §4.5.1 with an extended granular reasoning example about hormones in the liver.

With meeting these requirements and tasks and those ones in the preceding two chapters, the TOG has been specified fully: ontological notions, a logical theory, and functions that enable usage. This combination as presented in Chapters 2, 3, and 4 will be compared to other approaches and solutions in the next chapter.

Part IV

Discussion and Conclusions

Chapter 5

Related research and comparison with the TOG

5.1 Introduction

The aim of this chapter is to provide an overview of research into granularity in the various disciplines, compare it to the TOG, and discuss current limitations of—hence, directions of further research on—the TOG.

Assessment criteria. In §1.1.3, informal assessment criteria for related research were used for identifying problems of the three main types of approaches—formal, engineering, and subject domain—that have been put forward in the literature. Here, the focus is on assessing related theories and engineering solutions with respect to the *guiding principles of the key requirements* that any theory of granularity should meet. The result of this assessment with eight characteristic proposals is summarised in *Table 5.1*.

Chapter outline. Existing approaches to granularity span several disciplines, ranging from philosophical, to logic-based, engineering (software development), and subject domain specific proposals. The aim here is to extract their granularity component and emphasise the core approaches of the most comprehensive solutions of each field. One might think this leads to comparing apples with oranges, but a goal of this research and its resulting TOG is to be more generic than the individual partial solutions so as to provide a common framework that can be used across seemingly distinct areas. §5.2.1 elaborates on ontological aspects of granularity, where mereology and complexity will be considered. The ontology section is followed by formal theories (§5.2.2), ranging from conceptual data modelling for data warehousing, rough- and fuzzy sets, to contextual reasoning. We then proceed to engineering solutions (§5.2.3) and subject domain solutions. The focus of the latter is mainly biology and biomedicine (§5.3.1) and ecology and GIS (§5.3.2), and, briefly, time granularity (§5.3.3). Given the close relation of abstraction with granularity, its related literature will be discussed in §5.4. Last, we look at limitations of the TOG in §5.5, which considers theoretical limitations, such as the complexity of the language required to represent the TOG, and a look ahead to transformations toward implementations.

A longer version of §5.2.1.1 on granularity and mereology has been published in the *Applied Ontology* journal (Keet and Artale, 2007) and *ORM'06* workshop (Keet, 2006d), and a longer version on emergence is available (Keet, 2007e). Shorter versions of §5.4 have been published at the *ORM'05* workshop and *AI*IA 2007* conference (Keet, 2005b, 2007c) and a part of §5.5 has been published in the *AAAI'07 workshop on Semantic e-Science* (Keet and Rodríguez, 2007).

Table 5.1: Comparison of eight other approaches for representing and implementing granularity with respect to the 12 key requirements and four requirements for functions.

Nr.	Requirement	TOG	Other solutions[1,2]							
			TGP	$\mathcal{GMD}$	**MSD**	**PmGC**	**ConG**	**MDER**	**BioGR**[5]	**MADS**
1.	A theory of granularity should be usable in a format for contents at the instance level and at the type level	+	–	+	–	–	–	+	–	–
2.	A higher level makes indistinguishable the finer-grained details that are indistinguishable at that higher level	+	–	–	–	+	+	–	–	–
3.	There have to be at least two levels within a perspective	+	–	–	–	–	+	+	–	p[14]
4.	Accommodate both the quantitative and qualitative aspects of granularity (or: arbitrary scale and non-scale-dependent granularity)	+	–	+	p[9]	+	–	+	–	+
5.	The logic-based representation has to permit the two main ways for perceiving and representing granularity, being set theoretical and mereological	+	–	–	–	–	–	–	–	+
6.	One type of relation between granular levels within a particular granular perspective	+	+	–	+	+	+	+	+	–
7.	The type of relation between adjacent levels in a perspective has to be transitive for those perspectives that contain >2 levels	+	+	–	+	+	+	+	+	–
8.	A type of relation that relates contents in levels of granularity within a particular perspective can have the property of being intransitive, provided there are always exactly 2 levels in any given perspective that contains that type of relation relating the entities (/types)	+	–	–	–	–	–	–	–	–
9.	The entities or entity types in a particular granular level have at least one aspect in common, which is a criterion by which to granulate the data, information, or knowledge	+	–	+	p[9,10]	+	–	+	p[12]	–
10.	An entity (/type) never can reside in more than one granular level within the same granular perspective	+	+	–	–	+	+	+	–	–
11.	An entity (/type) in a particular granular level may reside in ≥ 1 levels, provided that each level the entity (/type) is contained in a distinct granular perspective	+	–	+	+	+	–	+	–	–
12.	If there is more than one granular perspective for a subject domain, these perspectives must have some relation among each other	+	–	p[7]	–	–	–	p[7]	–	+
13.	Functions to query granular components, retrieve content of levels and move between levels from coarser to finer-grained entities (/types) and back	+	–	p	–	–	–	+	–	–
14.	Content retrieval has to make use of the existing structure in the original data source and consider the type of granularity that is used for the granulation	+	–	p	–[10]	–	–	–	–	–
15.	Abstraction functions	+	–	p[3]	–	p[15]	–	p[3]	–	–

Table 5.1: (continued)

Nr.	**Requirement**	TOG	Other solutions[1,2]							
			TGP	$\mathcal{GMD}$	**MSD**	**PmGC**	**ConG**	**MDER**	**BioGR**[5]	**MADS**
16.	Expansion functions	+	–	p[3]	–	p[15]	–	p[3]	–	–
X.	Contains features the TOG does not have	N/A	+[11]	+[4]	+[8]	+[16]	+[5]	–	–	+[12]

[1] TGP by Bittner and Smith (2003), focussed on ontology and mereology; $\mathcal{GMD}$ by Franconi and Kamble (2003, 2004), focussed on data warehousing; MSD by Fagin *et al.* (2005), information/document search and retrieval; PmGC by Yao (2004a), rough sets; ConG by Schmidtke (2005), contextual reasoning; MDER by Malinowski and Zimányi (2006) for data warehousing and extending ER; BioGR by Kumar *et al.* (2005), for biomedical ontologies applications; MADS by Parent *et al.* (2006a,b), for spatio-temporal GIS applications.

[2] + yes; - no or not mentioned; p partial support; N/A not applicable.

[3] Roll-up and drill-down as simplified versions of basic abstraction and expansion functions.

[4] The join, union, and difference operators to combine or compare contents of levels from different perspectives.

[5] It has the notion of being accessible, that eventuality e is accessible in a context c_1 iff the smallest time interval that contains e is part of c_1 and compatible with the level of granularity for which c_1 determines the context size (Schmidtke, 2005). However, this does not seem to add any further advantages or uses.

[6] Kumar *et al.* (2005) refer to Bittner and Smith's TGP, but it is not explicitly included. One might want to assume that the "+" in the TGP column also hold for the BioGR one. This results in an inconsistency, however, for key requirement 1, where the TGP deals with particulars as contents for the levels, whereas BioGR focuses on universals as content.

[7] Through the fact table, but not directly. MultiDimER has a notion of "parallel hierarchies" and "shared levels" in the conceptual representation (not in the relational model), which approximates overlapping levels (§3.7).

[8] Three document information retrieval functions; see §4.2.6.

[9] It can be inferred from the examples and discussion, but is not explicitly represented in the MSD.

[10] Levels are created on the fly with pairwise disjoint collections and then tweaked to equalize the ratio of level:content distribution.

[11] The TGP comprises a formal characterisation (within mereology) of the desirable features A-C listed in §3.1 regarding disjointness and completeness.

[12] "size is a criterion for drawing dividing lines between granular levels. This criterion, however, cannot be applied indiscriminately to all entities on a given level" (Kumar *et al.*, 2005); In their examples, however, the authors do not make use of "smaller" and "larger in size", but use mostly parthood and 'one-off' exceptions twice.

[13] Most notably, temporal aspects, multi-representation structures for semantic flexibility, and an implementation.

[14] Implicit support through adherence to cartographic approach and all spatial data types.

[15] Two operators, for "zooming-in" and "zooming out".

[16] A basic notion of fuzzy measures and a probability function, which are orthogonal to the TOG levels and can be added.

5.2 Ontological and formal theories and engineering solutions for granularity

5.2.1 Ontology and philosophy

Several philosophical investigations into granularity have been carried out, which concerns mainly granularity with respect to mereology, complexity, emergence, and vagueness; the emphasis will be on the first two.

5.2.1.1 Granularity and Mereology

Mereology forms either the basis for a formal theory of granularity, like Bittner and Smith (2003)'s TGP, or informal philosophical investigations about natural kinds and biological systems lead to mereology or at least the basic notion of parts and wholes (*e.g.*, Winther, 2006; Kumar *et al.*, 2005). In order to understand the interplay between mereology and granularity, some background information on mereology is required; in the following, it is assumed that the reader is familiar with the basics of mereology[1] as well as the part-whole taxonomy that was introduced in §3.5.1. However, this taxonomy of part-whole relations does not deal with other facets of parthood relations, such as intra-part relations and if the parts together are *all* parts that make up the whole (for a discussion and various options to address such issues, see *e.g.*, Barbier *et al.* (2003); Guizzardi (2005); Lambrix and Padgham (2000); Motschnig-Pitrik and Kaasbøll (1999); Opdahl *et al.* (2001)). In the context of granularity, a very basic treatment of the inter-part relations is provided in §2.3 regarding the structure of the contents of a granular level, whereas part-exhaustiveness is not explicitly relevant for a *framework*, because it is a requirement on the *contents*, and thus may need to be addressed for design and implementation of granular biological information systems.

From the early efforts in biomedical ontology development in the '90s, part-whole relations were included as core relations, albeit without incorporating a mereological theory; *e.g.*, the FMA (Rosse and Mejino, 2003), GO (GO, 2004) and OpenGALEN. Put differently, from a domain expert's viewpoint, part-whole relations are essential to understanding and representing nature. In the philosophy of biology, this has also been observed and analysed to a greater detail, most recently and comprehensively by Winther (2006), who dubbed this approach *compositional biology* and considers it a methodology of theorizing that commits to "general explanatory, modeling, and part-identification strategies". Furthermore, compositional biology avails of perspectives (structure, process, mechanism, and function) and granulation frames that give existence to the kind of parts one can find in a level within a hierarchy for representing life sciences theories. Winther (2006) analysed several biological textbooks on the organisation of the material, which supported the notion of compositional biology and he concludes that "Partitioning frames, together with the questions of interest and the explanatory resources made available by the perspective, determine the operative explanatory accounts", thus, where the perspective and level of detail in the hierarchy are the inputs and determinants for "legitimate explanation[s]" by biologists[2]. Although Winther's analysis is informal (cf. analytical philosophy) and does not discuss granularity explicitly, important ingredients are present, which put together mereology, perspectives, levels, hierarchy of levels within a given perspective using parts and wholes, and abstracting away coarser- or finer-grained knowledge from the focal level. Thus, it shares the core notions of granularity with the TOG. For a formal approach toward augmenting mereology to make it usable with granularity, we proceed to Bittner and Smith's contribution. Bittner and Smith (2003) developed a comprehensive, "Theory of Granular Partitions" (TGP) that is built from *and on top of* mereological predicates. However, as can be observed in the comparative

[1]For comprehensive introductory overview articles, consult Varzi (2004a); Guizzardi (2005); Keet (2006e).

[2]Besides textbook analyses, this is advocated also by others; see *e.g.*, (Theise, 2005) and references below.

assessment in *Table 5.1*, it meets less that half of the key requirements. Three follow-up articles by either one of the TGP authors assume (i) extension to cover requirement 1 (Kumar *et al.*, 2004, 2005), (ii) a set-based approach (Bittner and Stell, 2003), thereby partially addressing requirement 5, and, most recently, (iii) to divert from mereology by asserting that the contents of grains *do not* stand in a mereological parthood to contents in coarser-grained levels (Rector *et al.*, 2006), which moves a significant step in the direction of requirement 8 by investigating the $member\_of$ relation. Thus, interestingly, these three articles independently converge toward the TOG, albeit that those efforts are as of yet not unified into a single 'TGP+'. As thought experiment, let us assume that combining the contents of these four articles does not lead to an inconsistent theory, then requirements 8, 11, and 12 still would need to be accommodated for in the static aspects of a 'TGP+' so as to deal explicitly with a granulation criterion and multiple granular perspectives, and, if it were to be used, then also requirements 13-16 for the dynamic aspects to query over the granularity framework and its contents. Nevertheless these shortcomings, the TGP is a significant step toward an *ontologically motivated, formal* theory of granularity. In fact, one also could push a mereology-based or -inspired theory of granularity further to not just covering aspects of basic mereology and mereotopology (Varzi, 2006a), but also embedding mereogeometries (Borgo and Masolo, 2007) to cover location and space aspects of granularity. Only the core notion of mereogeometry is included in the TOG with the $contained\_in$ relation, but this possibly could be expanded upon. On the other hand, such extensions will hamper effective usage of a theory of granularity considering the well-known problems of representing part-whole relations in ontologies and conceptual data models due to both the lack of experience by knowledge engineers and limitations in the representation languages[3]. In any case, it may be clear that based on ontological investigations, neither only set theory nor only mereology suffices to capture all facets of granularity. The current solution in the TOG with the types of granularity permits both.

A different issue in the comparison of the TOG with TGP is that the latter focuses on particulars and content of levels and perspectives (partitions in TGP) rather than characteristics of the granularity framework. A consequence is that desired features A-C are fully addressed in TGP. Although ontologically, in the ideal case, any d^s is fully granulated and none of the levels of its corresponding d^f are empty—a requirement of the TGP—this cannot be guaranteed with the TOG because this depends on the data source $\mathcal{DS}$ used to populate the granularity framework. For instance, we may have a $gp_i =$ Human structural anatomy with its granular levels, but the ontology or database that stores the ontology of human anatomy, O_a, may not have a section with organelles. In that case one could try to find an ontology of cell organelles, O_o, and add that to the $\mathcal{DG}$ afterward, but then for at least some time there are empty levels and thereby violating the completeness constraint. The alternative is to link or integrate the ontologies first into one large ontology, O_h (thus, $O_a \cup O_o \subseteq O_h$), and then load the granularity framework with O_h, which may delay a software implementation unduly. The TGP also has the exhaustiveness constraints so that no 'orphans' are permitted—*i.e.*, all objects of the $\mathcal{DS}$ are granulated and reside in a level—which is an ideal case as well, because the $\mathcal{DS}$ may contain other information for which no perspective and levels have been declared. Although both constraints could be accommodated for, it is not obvious what one can gain by doing so, as a $getC$ query would simply return an empty set and it would not lead to an inconsistent $\mathcal{DG}$, compared to the drawback of postponing implementation. Furthermore, one actually can benefit from having such constraints only as *desirable*, because then a granularity framework can serve also as a structure for coordinated ontology linking and integration (Keet, 2008b). Put differently, features A-C could be added as deontic constraints to an implementation TOG (see also §5.5.1). Either way, it may be clear why these constraints are desirables for any theory of granularity as opposed to key requirements.

Separate from the factors that already passed the revue on the interplay between mereology

[3]See (Keet and Artale, 2007) and references therein on the first assertion and (Keet, 2006e) for an overview of the main knowledge representation languages that do (not) support mereology.

and granularity, is the notion of indistinguishability, which is important for granularity, but not at all considered in the discourse about mereology. This is for the straightforward reason that indistinguishability is implicit in mereology: when one considers the wholes, its parts are out of scope, and when one investigates, say, the horizontal relations among the parts, then the whole is beyond the scope. Moreover, by focussing on the part-whole relation then the relation is the focus analogous to considering the indistinguishability relation itself, but by being part (whole) of the whole (part), this immediately provides a rationale as to why the parts are indistinguishable at the coarser-grained level of the whole. Therefore, it is unsurprising that mereology-based or mereology-inspired theories of granularity do not treat is as a separate issue. The requirement to have the notion of indistinguishability explicitly included in any theory of granularity, is because granularity constitutes more than only mereology and is an essential ingredient to granularity for determining why what things goes in which level.

5.2.1.2 Complexity and other notions from philosophy

Within philosophy, and philosophy of biology and theoretical biology in particular, there are two main discourses that are close to granularity, or use it but have it framed in other terminology, which are complexity theory of complex biological systems and emergence with their hierarchical systems in biology and levels of detail and explanation. This is only briefly touched upon here insofar as it has a direct effect on a theory of granularity. Within complex biological systems research, computer simulations are a very important component, "[which] often requires integration of multiple hierarchies of models that are orders of magnitude different in terms of scale and qualitative properties" (Kitano, 2002). To make the mathematical models manageable in software applications (such as E-Cell; PACE (2004); Paton *et al.* (2004)), it is practically useful or even necessary to abstract away smaller details, such as modelling population behaviour with organisms but not their constituent molecules, or only to approximate it, such as the sorting of genetic mutations caused by transpositions (Hartman and Sharan, 2004), and to divide the software in modules. However, then one most likely stumbles upon 'unexpected' behaviour of and results from the software, and the notion of emergence enters the debate. Emergence involves, loosely defined, "[e]mergent entities (properties or substances) [that] 'arise' out of more fundamental entities and yet are 'novel' or 'irreducible' with respect to them." (O'Connor and Wong, 2005)[4]. To sort this out, one avails of levels of detail: the more fundamental properties at a lower level and the 'new' property at the coarser level when the entities are combined into a higher-level entity/system. Summarising (Keet, 2007e), the following can be observed about emergence and its relevance to granularity and biology. Informal usage of emergence in biological discourse and modelling tends towards being of the epistemic type (*e.g.*, Barbier *et al.*, 2003; Opdahl *et al.*, 2001; Rector *et al.*, 2006), but not ontological emergence, primarily due to our lack of knowledge about nature and limitations how to model it. Philosophy adds clarification to better characterise the fuzzy notion of emergence in biology (Delehanty, 2005; Emmeche, 1997; Korn, 2001; Wimsatt, 1995, 2000), but paradoxically it is the methodology of conducting scientific experiments that can give decisive answers. A renewed interest in whole-ism in (molecular) biology and simulations of complex systems does not imply emergent properties exist, but illustrates the realisation that things a more difficult and complex than initially anticipated. Usage of (weak- and epistemological) emergence in bioscience is a shorthand for "we have a gap in our knowledge about the precise relation(s) between the whole and its parts and possibly missing something about the parts themselves as well", which amounts to absence of emergence in the philosophical sense. Relating this to complex biological systems, modelling, and granularity, the following can be observed. Given that the existence of emergent properties is not undisputed, we need better methodologies to investigate such claims. Granularity serves as one of these approaches

[4]The following does *not* deal with a certain usage of 'emergence' in medical literature that actually deal with evolution; see Antonovics *et al.* (2007) for an analysis on this issue.

to investigate postulated emergent properties. Specification of levels of granularity and their contents can provide a methodological modelling framework to enable structured examination of emergence from both a formal ontological modelling approach and the computational angle, and helps elucidating the required level of granularity to explain away emergence. The latter is indirectly also advocated by (Cariani, 1997; Edmonds, 2000; Silberstein and McGreever, 1999, among others), although the discourse uses terminology such as "relative to a model" and at a specific "level of detail". In this approach, Cariani's (1997) operational definition of emergence is useful: emergence relative to a model, only novel at a level of description and "the detection of an emergent event is a joint property of both observer and [measurement-taking] system". When something unexpected, the emergent property/behaviour, pops up during the simulation, one can either 1) change the algorithm, or 2) add more parameters, or 3) make more precise (finer-grained) measurements to resolve the issue (Cariani, 1997). In this setting, granular levels and their contents provide a methodological modelling framework to enable *structured* examination of claims of emergence using both a formal ontological and computational approach. It makes the complex at least less complex, and aids understanding which levels are essential for explanation of some property of observed behaviour, hence that are more, or less, relevant for *e.g.*, developing simulation software.

A side-issue of the investigations in complex biological systems and emergence, but relevant for granular levels in a perspective, is that, unlike (Salthe, 1985, 2001) claims, a minimum of two levels is required, not three. One either needs a focal level and a level 'below' to which it is irreducible, or in the other direction one has the focal level and the emerging property at a higher level that is unpredictable or not derivable. The focal granular level where the assumed emergent property manifests itself exists by virtue of the lower level that supposedly has insufficient explanatory power (the irreducibility argument), or the lower level has sufficient power but emergence occurs at the adjacent level (non-predictability and non-derivability argument). Thus, also from a philosophical argumentation one reaches the validity for the constraint to have at least two levels in a perspective, as was proved earlier in *Theorem 3.2*.

5.2.2 Formal theories

There are several formal theories that cover one or more aspects of granularity into greater or lesser detail. The most widely known formal approach is Hobbs' (1985) early proposal for dealing with granularity, which was discussed in §3.4.1. Although it is not a full-fledged theory, Hobbs did isolate three key requirements that a theory of granularity should have: a notion of level, that it has to do with properties of the entities (/types), and making things (in)distinguishable when moving between levels, which the TOG satisfies. In this section, we look at relatively comprehensive theories that have been developed since Hobbs contribution, which are organised according to three main approaches: formal contributions originating from data warehousing, rough sets and fuzzy logic, and options to reuse contextual reasoning.

5.2.2.1 Conceptual and logical data modelling for Data WareHouses

To some extent, data warehouses (DWH) use granularity in the conceptual and/or logical model for the DWHs, using terminology such as aggregation, roll-up, dimension, and levels of a dimension. We first take a closer look into the Generalised Multi-Dimensional model $\mathcal{GMD}$ (Franconi and Kamble, 2003, 2004; Kamble, 2004), which generalises 16 different types of DWH *logical* modelling approaches (Kamble, 2004), such as the star, snowflake, and Gray's cube, and their logical specifications (among others, Cabibbo and Torlone, 1998). While the $\mathcal{GMD}$ focuses on the logical model, two more recent proposals will be considered that use *conceptual* multidimensional modelling for DWHs extending UML, called YAM^2 (Abelló *et al.*, 2006), and extending ER with MultiDimER (Malinowski and Zimányi, 2006). $\mathcal{GMD}$ has a formal specification only, YAM^2 lacks a full formalisation but has diagrammatic support and some constraints in OCL, whereas MultiDimER has both a diagrammatic and a formal specification. They all originate from DWH-engineering and lack ontological foundations and thereby reusability of the theory outside the DWH setting. For instance, they rely on intuitive notions of dimension, level and so forth, whereas the TOG has these components abstracted to a higher level of generality, unambiguously defined, and substantiated with ontological motivations, so that it can equally be applied to, for instance, DL-based knowledge bases, ORM, DWHs, and ontologies. On the other hand, $\mathcal{GMD}$, MultiDimER, YAM^2, and other proposals for DWHs may have more pre-defined functions to meet specific DWH needs, such as the full cube algebra (Franconi and Kamble, 2004), but the *range of options* for defining functions to query the TOG components is larger, because the more comprehensive and unambiguously defined the language for representing granularity, the more versatile and well-founded the queries one can pose over a $\mathcal{DG}$ (recollect §4.2.6).

In the following two subsections, $\mathcal{GMD}$ will be assessed first, followed by YAM^2 and MultiDimER, where the focus is on the representational component and how the DWH constructs relate to TOG components. The assessment of $\mathcal{GMD}$ and MultiDimER against the key requirements is included in *Table 5.1*.

The Generalised Multidimensional Model $\mathcal{GMD}$. $\mathcal{GMD}$ contains a signature, semantics, and algebra. The algebra is tailored specifically to DWHs, not of a conceptual nature, and incorporates aspects that do not belong to granularity, like the join and difference operations. The $\mathcal{GMD}$ signature is defined as follows:

DEFINITION 5.1 ($\mathcal{GMD}$ signature (Kamble, 2004))**.** *A $\mathcal{GMD}$ signature is a tuple <$\mathcal{F}$, $\mathcal{D}$, $\mathcal{L}$, $\mathcal{M}$, $\mathcal{V}$, $\mathcal{A}$>, where*

- *$\mathcal{F}$ is a finite set of* ***fact names***
- *$\mathcal{D}$ is a finite set of* ***dimension names***
- *$\mathcal{L}$ is a finite set of* ***level names*** *each one associated to a finite set of level elements*
- *$\mathcal{A}$ is a finite set of* ***level attribute names***
- *$\mathcal{M}$ is a finite set of* ***measure names***
- *$\mathcal{V}$ is a finite set of* ***domain names*** *each one associated to a finite set of* ***values***

I address three general aspects first, and then look at the individual components of the signature and how this maps into the TOG. First, $\mathcal{GMD}$ uses set theory and the Urelement for the lowest level in a perspective, which has its limitations and is not the only aspect of granularity (recollect §2.1.4). Second, aggregation, facts, and its dependency on measures, which are heavily used in the $\mathcal{GMD}$, do not cover all types of granularity, but they are useful for **samG**, **nasG**, and possibly **nacG**. Third, the $\mathcal{GMD}$ logical data model is defined in terms of *names* instead of a representation of *entity types*: a database may have a "legal state" (Franconi and Kamble, 2004)—being internally consistent—but this does not guarantee proper grounding in reality; that is, the "finite set of fact names" should stand for a shorthand notation for a "finite set of facts represented with names", and so forth for the other sets in the signature. A semantically inappropriate aggregation into some level for a Sales fact may be a nuisance for a manager, but semantically incorrect reasoning may result in drawing wrong summaries and conclusions in, *e.g.*, a climate model or a medical DWH. On the other hand, logic itself does not deal with subject domain semantics—representing knowledge in ontologies does—and logical database models do not necessarily have to be subject to ontological analysis, because that should occur at the conceptual data modelling stage. In contradistinction, the TOG is subject to these requirements as it also has to suffice for knowledge bases and ontologies in the biology domain, *i.e.*, its intended scope is broader than $\mathcal{GMD}$. Notwithstanding these points, the $\mathcal{GMD}$, may a viable option for a trimmed-down TOG. We now assess where and how the $\mathcal{GMD}$ elements of the signature can be mapped into the TOG.

$\mathcal{GMD}$'s $\mathcal{F}$ and $\mathcal{D}$ imply a prioritisation of two desiderata or criteria. Based on the description of the $\mathcal{GMD}$ and the provided examples, $\mathcal{F}$ can be mapped either into the TOG's D or, together with $\mathcal{D}$, into GP. The *combination* of an $\mathcal{F}$-element and a $\mathcal{D}$-element subsumes GP and given the sets listed in *Table 4.2*, then $\mathcal{P} \subseteq \mathcal{F} \times \mathcal{D}$, because there can be more theoretical combinations of mixing fact names with dimension names than one would have in a particular subject domain. Examples given in (Franconi and Kamble, 2004; Kamble, 2004) are fact names such as Sales and Purchases and dimension names such as Date and Product. This can suffice in a particular DWH, but a categorisation of products should be done *independent* of belonging at times to sales and at times to purchases because these are the roles assigned to the products so that the independence permits reuse of product categories for different purposes. Either way, a Sales-Date combination corresponds to a **sG** type of granularity and a Sales-Product combination of the **nG** type; hence, such examples are covered by the TOG. Because the $\mathcal{GMD}$ does not have this typing of dimensions, it cannot prevent inconsistent level definition, which is solved by the TOG with the taxonomy of types of granularity.

The set of level names $\mathcal{L}$ roughly maps onto the set of levels in a domain granularity framework, $\mathcal{DL}$, but not necessarily. Examples of $\mathcal{GMD}$ level names are Month, Year, Brand, and Beverage category, whereas their respective level elements (dimension values) can have values like January, 2004, Nastro Azzurro, and Beer, respectively (Franconi and Kamble, 2004). Given a domain granularity framework adhering to the TOG, then we would have a gp_1 = Gregorian calendar for a calendar hierarchy, or the Date-dimension of $\mathcal{D}$ in (Franconi and Kamble, 2004; Kamble, 2004), with at least the levels Month and Year and structured level contents $\mathcal{E}$ for January and 2004, respectively. A second perspective for a taxonomy of beverage types can be declared as gp_2 = Beverage category where each depth in the tree can be a granular level, such as Alcoholic beverage and Beer. Brand, on the other hand, is an attribute of an entity (/type) and as such is not granular[5]. Thus, based on the examples, a $\mathcal{GMD}$'s level name can be mapped into a TOG's instance of GL, a particular GP, or not mapped if it is a simple attribute. This mixing can be due to the fact that $\mathcal{GMD}$'s $\mathcal{L}$ and level elements lack characterisation with definition and constraints[6], granulation criteria for unambiguous perspective-awareness of each level, and their relations, *e.g.*, how dimensions names of $\mathcal{D}$ and level names and elements of $\mathcal{L}$ relate, which are included in the TOG.

The set of measure names $\mathcal{M}$ can be used with granularity, because it has functions how one level element relates to another through operations such as averaging and summing, but this does not imply there is a granular relation. Averaging itself is not granular, but making indiscernible the minor variations among a mean value is; hence, it depends on its usage. Likewise, summing Unit sales (Franconi and Kamble, 2004) is of itself not a function to relate to or create a coarser-grained level, but adding daily sales figures for a monthly overview and grouping (summing) the month figures to create yearly sales figure is. This is not because of the summing itself but because *what* is being summed *how*—in this case, it is the calendar that determines which values are summed, not the units that were sold.

The domain names, elements of $\mathcal{V}$ such as string and integer, have nothing to do with granularity, but with modelling data in the data source. The domain values themselves have to have a semantics associated with it, such as numbers denoting temperature measurements, for it to be used in some subtype of **sG** granularity. Last, the attribute names in $\mathcal{A}$ do not have a direct correspondence with the TOG because the attribute is not granular, and therefore outside the scope of the TOG. Attributes may be useful for providing additional information about some entity, but this belongs to data modelling instead, *i.e.* something that a $\mathcal{DS}$ contributes to the $\mathcal{DG}$.

An advantage of $\mathcal{GMD}$ is that it is a readily available theory in line with engineering practice in DWH modelling, whereas for the TOG this transformation step is yet to be carried out.

[5]If one intends to categorise information that products of different brands go together in one branded product, then this has a core the products and its component-parts. Likewise, where a multinational conglomerate of a 'larger' brand, *e.g.*, Procter & Gamble, owns product-specific brands like Colgate toothpaste and Libero diapers, then this denotes a hierarchy of companies in the first place.

[6]$\mathcal{GMD}$ has constraints specified on the interpretation of $\mathcal{F}$, F, for the fact names (Franconi and Kamble, 2003), but not on the levels.

However, $\mathcal{GMD}$ is only applicable to DWHs, whereas the decoupling of the data modelling aspects from the (onto-)logical aspects—as advocated by the TOG approach—facilitates portability of the theory across platforms and implementation scenarios, which in turn eases integration of software systems.

Extensions to UML and ER for modelling multidimensional information. Abelló *et al.* (2006) defines the main components of the YAM^2 UML-based DWH conceptual data modelling language through a combination of textual explanation, UML's Object Constraint Language (OCL), a UML meta-model, and stereotyping. It contains components such as level, descriptor (attribute), dimension, cell, measure, and fact. Levels in a dimension loosely correspond to granular levels and are related through UML's aggregation relation, but despite referral to mereology, formal semantics of the aggregation relation—or at least the authors' precise interpretation—is absent, which thereby makes YAM^2 at least partially ambiguous (see, *e.g.*, Barbier *et al.* (2003); Guizzardi (2005, 2007); Motschnig-Pitrik and Kaasbøll (1999) on the many issues with UML's aggregation relation). YAM^2's meta-model (figure 9 in Abelló *et al.*, 2006) is for a considerable part comparable to the graphical rendering of the TOG in *Figure 3.1*. The two main differences are inclusion of implementation considerations in the YAM^2 meta-model and the limitation to aggregate (summarise) data in a DWH using measures only, whereas the TOG also permits usage of inter-level relations between entities (/types), types of granularity, and the more generic criterion and is formally characterised. Regarding implementation-motivated meta-classes, it makes a distinction between "/SummarizedCell" (and "/SummarizedMeasure") and "FundamentalCell" ("FundamentalMeasure") to indicate types of cells that, according to the OCL, contain data that "must be summarizable" (can be derived) or are "not derived", respectively, but at the conceptual level it should be irrelevant *how* the data is stored and where it originates, because there one should bother with *what* kind of information should be stored. Further, the summarised and fundamental cells and measures in YAM^2 are not disjoint complete, which may result in a particular cell being both fundamental and summarised, contradicting the idea behind the division and it may lead to an inconsistent state of the DWH. Last, YAM^2's pre-defined functions can be adequately addressed within the TOG, which has been discussed and demonstrated in §4.2.6; moreover, the TOG allows for further extension to cover querying for each component in the granularity framework.

MultiDimER meets most key requirements any theory of granularity should have (see *Table 5.1*). There are, however, a few distinctions in addition to those mentioned at the start of this section: (i) the TOG permits more types of relation between the entities (/types) than the set-based MultiDimER (Malinowski and Zimányi mention that it could be a point of further investigation to extend MultiDimER with such constructs) and as a consequence, MultiDimER, like $\mathcal{GMD}$, has underspecified roll-up and drill-down functions compared to TOG's abstraction and expansion functions; (ii) granular querying of the data in a $\mathcal{DG}$ is enhanced to retrieve more information by availing of the taxonomy of types of granularity and structure of the contents of levels, which MultiDimER does not have, although the "metamodel of hierarchy classification" (figure 4 in Malinowski and Zimányi, 2006) provides preliminary characteristics in that direction. Last, an advantage of Malinowski and Zimányi's article that will be useful for software engineers—although it is independent of a formal theory of granularity—is the mapping from the conceptual and logical specification of MultiDimER to the relational model, which has yet to be provided for the TOG.

5.2.2.2 Rough sets, fuzziness, and clustering

The State of the Art of granularity in §1.1.3 mentioned a separate branch in formal approaches to deal with granularity: Granular Computing. This is an umbrella term for efforts that comprise some AI and philosophy, but focus mainly on rough and fuzzy sets, fuzzy logic and their sister disciplines data mining and machine learning. Important ingredients for granular computing, such as similarity, equivalence, indistinguishability, and indiscernibility, have been analysed in detail in §3.4.1 and were summarised in (Keet, 2007d). Characteristic for these approaches is the applied mathematics, data-centric view, and quantitative aspects of data for problem-solving tasks, although the notion of "computing with words" (Zadeh, 1997, 2002; Mendel and Wu, 2007) clearly moves in the direction of subject domain semantics. The notion of a granularity *framework* with formally defined perspectives and levels is absent, but there are notable steps in that direction. Skowron and Peters (2003) have granule g as primitive, but they use it only for attaching lower and upper approximation bound to it. Chen and Yao (2006) use granular perspectives ("multiviews") and a lattice as flexible granulation hierarchy, but do not have level as a structure proper. Qiu *et al.* (2007) make steps from set extension to concept, introduce the MSU implicitly, call GL a "granular world" that denotes a set of "concept granules" (roughly: classes), have a mapping function to go from the finer- to the coarser level, and the union of such levels is a "full granular space", which corresponds to a GP that always must have $GR = is\_a$. This clearly moves in the direction of the TOG, although it is limited to taxonomies only, and, most notably, misses the granulation criterion C, the relation between the levels RL, and quantitative granularity. Yao's (2004a) comprehensive rough set-based partition model will be discussed in some detail in the next paragraph. Afterward, clustering and fuzzyiness will be touched upon.

The partition model. The comparison of Yao's partition model with the TOG is summarised in *Table 5.1*. Before discussing the details of this model, it is of interest to mention that Yao (2004a,b) had specified a list of key requirements as well, which are subsumed by all 16 requirements, except for one. Yao's five main requirements are: a granulation criterion, a granule (level) and granulation structure, the need for granulation methods, to describe and name the levels, and to cater for quantitative characteristics. The TOG meets all these requirements as well. In addition, with the aim of computing and reasoning, Yao lists alos requirements for mappings between different levels, conversion between levels, other operators, and property preservation. The "[o]perators" requirement is not specified other than informal suggestions, such as abstraction, expansion, coarsening, refining, and approximation operators; these are met at least in part by the TOG, which has them formalised and has additional functions for querying the TOG components. "Property preservation" means that a finer and coarser representation preserve their characteristic in that if some coarse-grained representation of the problem does not have a solution, then neither should its finer-grained representation, *i.e.*, maintain the ""false-preserving" property" across levels[7], which is in a similar vein as Knoblock's solution property for planning problems[8], and to some extent the idea of compositionality for abstractions (Giunchiglia *et al.*, 1997; Pandurang Nayak and Levy, 1995). However, there can be good reasons for deviating from this 'ideal' case to permit non-monotonic extensions for finer-grained levels or acceptable imprecision at a coarser-grained level, which was mentioned before in §5.4 and by Keet (2007c) on abstractions and discussed in (Keet, 2007e) in the context of emergence and simulations. Thus, this requirement is not universally applicable and therefore it has not been included in the list of key requirements.

Looking at the partition model itself, it does not fully meet the requirements, because the partitioning using the equivalence relation on attributes results in a lattice of sets, but not levels, and the relation of a finer and its adjacent coarser-grained 'contents of a level' is given only by a pair of mappings between the two "views". More precisely, U/E is the quotient set of more detailed universe U, the equivalence class $[x]_E$ is given by $[x]_E = \{y \in U | xEy\}$, and, following

[7]which Yao summarises from Zhang, B., Zhang, L. (1992). *Theory and applications of problem solving*, North Holland, Amsterdam.

[8]Paraphrased in (Yao, 2004a); full reference: Knoblock, C.A. (1993). *Generating abstraction hierarchies: An automated approach to reducing search in planning.* Kluwer Academic Publishers, Boston.

Dubois and Prade, $[x]_E$ represents a subset of U in the fine-grained universe, and $Name([x]_E)$ denotes an element of coarser-grained U/E. Put differently, at the finer-grained level we have a set of objects that at a coarser-grained level is one whole "named category or concept" (Yao, 2004a). These two views are related through a pair of mappings so that $r : U/E \mapsto U$ and $c : U \mapsto U/E$ and thereby $r(Name([x]_E)) = [x]_E$ and $c(x) = Name([x]_E)$, which for r is simply labelling of an unnamed set, but not granularity, and c amounts to membership assignment of a named set. However, the description of the intention what it should capture is that "As we move from one view to the other, we change our perceptions and representations of the same concept", which corresponds either with Gerstl and Pribbenow's (1995) idea with respect to different perspectives on parts and wholes or the difference between intension of a universal and its set-extension as articulated since the 90s in the context of Ontology & ontologies initiated by Guarino. While distinct, each one can be accommodated for in the TOG.

One advantage of basing the partition model on *rough* set theory is the well-researched aspect of rough inclusions or *approximations*: for lower and upper approximation in a coarse-grained universe U/E where one zooms out on a subset $A \subseteq U$, Yao (2004a) has the approximations as subsets of U/E instead of U:

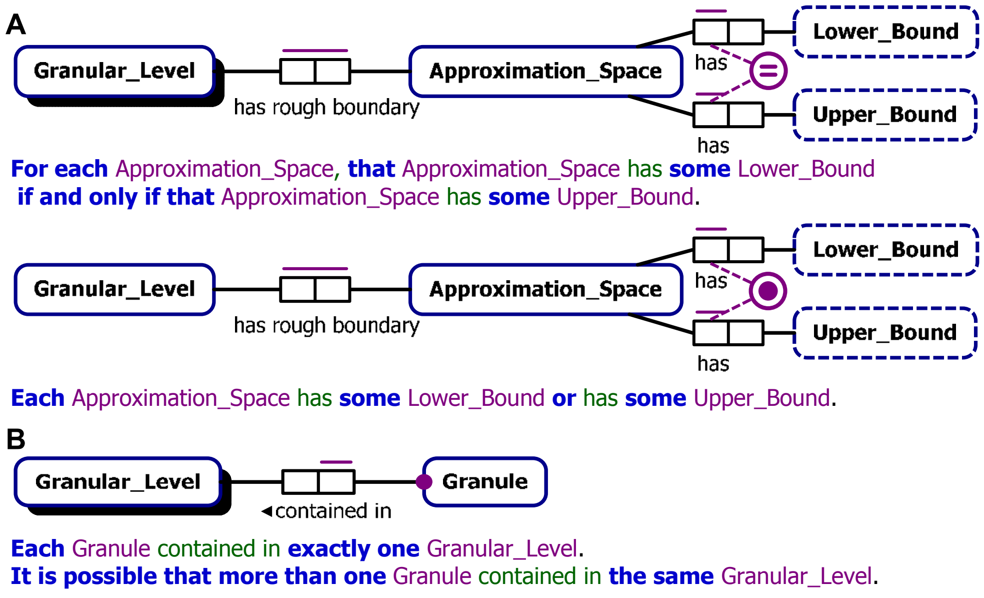
$$\underline{apr}(A) = \{Name([x]_E) | x \in U, [x]_E \subseteq A\}, \tag{5.1}$$
$$\overline{apr}(A) = \{Name([x]_E) | x \in U, [x]_E \cap A \neq \emptyset\} \tag{5.2}$$

with obvious properties, such as $\underline{apr}(A) \subseteq \overline{apr}(A)$ and if $A \subseteq B$ then so are their respective aprs. Relating this to the TOG, recollect that one can identify two axes along which granularity and granule membership can operate (see §3.4.1 on page 69): for *defining* the levels and the *precision* in the definition of a level. The latter has not been included in the TOG in the same amount of detail as in Yao's partition model. Rough sets with degree of membership and approximations is one specific aspect of granularity used interchangeably in rough set theory for either one or both of the axes—level and precision of the level—that is not used (formally or informally) across the spectrum of research on granularity. The TOG is a proper generalization and as such can be extended easily for this specific setting, where there is, in addition to the finer-and coarser levels of granularity, also a measure of precision (degree, approximation, roughness) for each level, such as a level for each kilometre with an approximation space ±10m, *i.e.* a lower bound of 990 metre and upper bound 1010 metre. Practically, this requires an additional fact type in *Figure 3.1* where Granular_Level participates; *e.g.*, Granular_Level has rough boundary Approximation_Space that corresponds to the typed predicate

$$\forall x, y(has\_rough\_boundary(x, y) \rightarrow GL(x) \wedge Approximation\_Space(y)) \tag{5.3}$$

where $Approximation\_Space$ has two related value types associated with it for the lower and upper bound. Two options for extensions are depicted in *Figure 5.1*, which can be elaborated on with constraints such as Lower_Bound < Upper_Bound and that an approximation space of a finer grained level must be smaller than that of a coarser-grained level. It has not been included in the TOG, because investigating how to add and combine rough sets, fuzzy sets, fuzzy logic, and/or rough fuzziness is beyond the current scope. Last, although the partition model, like other rough set approaches, focus on quantitative aspects of granularity with measured values, the formalization includes *arbitrary* attributes (and values), which therefore, implicitly, allows for qualitative granularity as well. However, this is not addressed specifically in (Yao and Liau, 2002; Yao, 2004a), only alluded to by other rough set approaches or its informal discussion (Bittner and Stell, 2003; Chen and Yao, 2006; Yao, 2004b, 2005a) or ignored (Peters *et al.*, 2002; Skowron and Peters, 2003; Skowron and Stepaniuk, 2007).

Other contributions. The principal other formal approaches with a computational scope for granularity are (rough and/or fuzzy) clustering, fuzzy sets, and to a lesser extent the combination of rough sets and fuzzy sets into fuzzy rough sets. The latter considers, among others, a



For each Approximation_Space, that Approximation_Space has some Lower_Bound if and only if that Approximation_Space has some Upper_Bound.

Each Approximation_Space has some Lower_Bound or has some Upper_Bound.

Each Granule contained in exactly one Granular_Level.
It is possible that more than one Granule contained in the same Granular_Level.

Figure 5.1: A: Two options for extending the TOG with approximation values for the levels. The top figure requires a bounded space when one specifies one of the values, the bottom one permits specifying either one or both bounds. Also, one could enforce combining the inclusive-or with the equality constraint. B: an optional additional granule that might facilitate formally characterising some set-theory oriented types of granularity.

fuzzy similarity relation to add another dimension with *degree* of similarity, and attribute reduction of fuzzy attributes using rough sets (refer to Chen *et al.* (2007) for recent results). Within the scope of fuzzy sets, one can distinguish the purely quantitative focus and computing with words (Zadeh, 1997; Mendel and Wu, 2007) with, *e.g.*, linguistic fuzzy models for classification in the context of macroinvertebrate habitat suitability in aquatic ecosystems (Broekhoven *et al.*, 2007), which involves manual adjustment by domain experts of the numerical membership functions associated to the "linguistic values". The named sets in the linguistic fuzzy model, however, do not yet deal with levels of granularity among the sets. Hierarchical and fuzzy clustering (Dawyndt *et al.*, 2006; Kaplan *et al.*, 2004, 2005; Shachar and Linial, 2004; Vernieuwe *et al.*, 2007), among many, have quantitative granularity implicit in the mathematical models. Limitations of such parameter-adjustable generated hierarchies are discussed by Zhou *et al.* (2005). They developed an algorithm to generate just that hierarchy whose data points group together to set-extensions of the universals in the Gene Ontology (see also *Example 3.4*). This, as well as Tsumoto's (2007) mining with a diagnostic taxonomy for headaches, could be interesting in conjunction with classification in logic-based ontologies and bottom-up generation or validation of the granular levels in a granular perspective.

Hata and Mukaidono (1999) explore granulation with fuzzy logic, where three classes of fuzzy information granulation are distinguished. First, their example for fuzzy Kleene classes uses informal granulation through mixing granulation criteria: first a transformation function from a colour gradient to a x-y plot with the usual range [0,1] and then to select parts of the line. Second, their fuzzy probabilistic classes formalise the informal usage of more detailed attributes to calculate an overall probability for an event, which needs further investigation to take a structured approach to 'component-probabilities' of aggregated 'whole probabilities'. The third fuzzy information granulation is based on fuzzy Lukasiewicz classes, which, for the given example, amounts to fuzzy mereology (detecting anatomical parts in images of the whole brain), that, when worked out in greater detail, could be an interesting combination of qualitative with quantitative granularity and traditional bio-ontologies in the Semantic Web with fuzzy OWL-DL (Straccia, 2006).

5.2.2.3 Contextual reasoning

Contextual reasoning with local model semantics considers a a logical theory as a *context* where reasoning primarily occurs within that particular context. A context is demarcated and has parameters and their values, which can be represented as a box: there is inside the box containing expressions, outside the box, which is a set of parameters, and the values of the parameters (Benerecetti *et al.*, 2000, 2001). For example, one might have a granular level Organism represented as a box that contains anything relevant about organisms at that level; any supra-organism matters, *e.g.*, population dynamics, is assumed and specified somewhere else. A parameter could be Metabolism with value *true* (if something does not have a metabolism, it is not a living organism). Formalizations of the box and context switching have been presented elsewhere (Bouquet and Serafini, 2001; Giunchiglia, 1993; Serafini and Roelofsen, 2004; LMS). For the current purpose where levels and perspectives might be candidates for context boxes, it suffices to introduce two of its definitions. First, a multi-context system:

DEFINITION 5.2 (MultiContext System MCS, (Bouquet and Serafini, 2001))**.** *A* Multicontext System (MCS) *for a family of languages $\{L_i\}$, is a pair* MS $= \langle\{C_i = \langle L_i, \Omega_i, \Delta_i\rangle\}, \mathcal{BR}\rangle$*, where each $C_i = \langle L_i, \Omega_i, \Delta_i\rangle$ is a theory (on the language L_i, with axioms Ω_i and natural deduction inference rules Δ_i), and $\mathcal{BR}$ is a set of bridge rules on the set of indices I.*

Note that the languages in $\{L_i\}$ are *propositional*. A FOL version would be preferable with respect to a comparison with TOG and we will see several limitations of using an MCS for granularity due to this difference. Second, a bridge rule *br* (with $br \in \mathcal{BR}$) can be used to link formulas between different contexts, which is defined as:

DEFINITION 5.3 (bridge rule *br*, (Bouquet and Serafini, 2001))**.** *A* bridge rule *on a set of indices I is a schema of the form:*

$$\frac{\langle \phi, i_1 \rangle \ldots \langle \phi, i_n \rangle}{\langle \varphi, i \rangle} br \tag{5.4}$$

where each index $i_1, \ldots,_n, i \in I$ and $\phi_1, \ldots, \phi_n, \psi$ are schematic formulas. A bridge rule can be associated with a restriction*, namely a criterion which states the condition of its applicability.*

The $\langle \phi, i_1 \rangle$ is a well-formed formula that corresponds to an $in\_level(\phi, i_i)$ iff i_1 is not only a context, but also a level of granularity. In addition to these two definitions, there are three mechanisms of context switching—'jumping' from one box to another—and mappings to three different kinds of dimensions along which context representations may vary (Benerecetti *et al.*, 2000, 2001). That is, we have to generalise from *br* between *formulas* to some switching relation, ς, and mechanism, m, between *contexts*. Because a precise definition of context switching was not given by Benerecetti *et al.* (2000, 2001) or others, it is introduced here.

DEFINITION 5.4 (switching ς)**.** *Let $c_1 \ldots c_n \in C$ be contexts in a MultiContext System (MCS), then a* switching relation *ς, where $\varsigma \in \mathcal{S}$, holds such that $\mathcal{S} \subseteq C \times C$. A switching relation ς_i has a switching* mechanism*, m_i associated with it, where $m_i \in \mathcal{M}$ and $\mathcal{M}$ the set of types of mechanisms.*

The current switching mechanisms—the minimal members of $\mathcal{M}$—are summarized first, (based on Benerecetti *et al.* (2000, 2001)). They are expand/contract (also known as localized or parochial reasoning), push/pop, and shifting. Expand/contract involves often the use of some pragmatic rule to divide knowledge with as assumption some situation or subject domain, where c has situations implied in the box with more (expanded, c_1) or less (contracted, c_2) axioms such that for the contents of the contexts, $c_2 \subset c_1$ holds. We denote the inverse of expansion (m_e) with m_c. Push/pop (m_u, and its inverse m_o, respectively) uses a situation implied in c_1 that can be encoded as a parameter of the box—pushed onto a parameter of c_2; thus, some $c_1(x, y, s)$ becomes a $c_2(x, y)$ with an extra parameter, and one pops back with m_o to the more detailed $c_1(x, y, s)$ by reintroducing the assumed situation. Last, shifting, m_s, concerns with changing the values of the parameters: c_1 looks at the same entities and relations from another perspective than c_2. The respective dimensions of the three context switching mechanisms are partiality, approximation, and perspective. Partiality means that only some section is represented, alike representing only some of the properties of a universal instead of all its properties. Approximations indicate different levels of detail on a given state of affairs that also may be partial and might be granular. Last, a perspective "encodes a spatio-temporal, logical, and cognitive point of view on a state of affairs" (Benerecetti *et al.*, 2000).

With this brief introduction to contextual reasoning, we proceed to the assessment on suitability for representing granularity as specified in the TOG. Before taking up transformations and mappings of TOG and MCS elements, it will be useful to look at the five "general desiderata for an adequate logic of context" (Bouquet and Serafini, 2001) and assess its applicability for granularity. Bouquet and Serafini's first, second, fourth and fifth can be applicable, the third not. That is, "(iii) the truth of a common sense fact should be dealt with as dependent on a (possibly infinite) collection of assumptions (which implicitly define its *context*)" is not applicable, for granularity is agnostic about common sense facts and by granulating, one ensures that the definitions of granules, levels, and perspectives will be finite, and they are, ontologically, not defined by its contents. Desideratum iv—each fact must be in a context—is debatable for granularity because it depends on the ontological commitment: it holds if one sees that the world is granular, but cannot be enforced if granularity is deemed to be only a convenient cognitive device used by humans (recollect §3.2.1). Desiderata i, ii, and v are equally applicable to granularity, and subsumed by it: "(i) context should allow a simpler formalization of common sense axioms; (ii) context should allow us to restrict the vocabulary and the facts that are used to solve a problem on a given occasion; ... (v) reasoning across different contexts should be modeled.". Bearing in mind that granularity and contextual reasoning may have different purposes and scope, we assess now if and how the two can be made compatible nevertheless, taking MCS for the exercise[9]. Let C be a context, then one can relate it to the TOG-components as: $GL \rightarrow C$, $GP \rightarrow C$, and $D^f \rightarrow C$. For ease of reference, they will be labelled henceforth as C^l, C^p, and C^d, respectively, although this does not resolve how to add *types* of contexts to a MCS. From this, it is immediately clear that a context by itself is insufficient to capture all aspects of levels, perspectives, and domain. One can, however, encode some of the TOG components in the parameters (P_C) and its values (V_C) of a context to cover a GP's criterion C and type of granularity TG, but the contexts do not capture the cardinality constraints between the TOG-components. Other problems arise with accommodating the TOG relations RC, RE, RL, and RP, and the granulation relations GR, for we have only the contextual reasoning's bridge rule, *br*, and a newly introduced ς at our disposal. Hence, to capture the semantics of the TOG relations, *br* would need to be dressed-up—its semantics changed—in various ways. Let R be the relation that relates C^l (resp. C^p and C^d) to its parameter P_C^l (resp. P_C^p and P_C^d), then $RC \rightarrow R$, and we can define a *simplified* notion of RE, RE^* (for it omits proper parthood and its constraints), with a ς that uses a *push* mechanism (m_u), ς_u:

$$RE(x,y) \wedge GP(y) \wedge GL(x) \rightarrow (\varsigma_u \leftrightarrow RE^*) \wedge C^p(y) \wedge C^l(x) \tag{5.5}$$

This, because one can take the V_C of C^l (through P_C) as the situation s that gets pushed away. Consequently, the pop-version of ς can be used for the inverse RE^-. In analogy, we have RE between GP and D^f with a push/pop mechanism, but with the difference that then *two* situations would have to be pushed, being the criterion and type of granularity. Expand/contract with ς_e and ς_c, respectively, might be used for RL and RL^-, respectively, or, following the push/pop mechanism in (5.5), map RL to ς_u as well, because it relates the boxes that contain finer- and coarser-grained theories (in MCS terminology: varying approximations). Put differently, the se-

[9]LMS/MCS is strictly more general than the main formalisation of context, Propositional Logic of Context (PLC) (Bouquet and Serafini, 2001), therefore PLC is not considered here.

mantics of the switching mechanisms are currently underspecified which hampers establishing a clear mapping for RL. Hence, we would need another more elaborate and formally defined way to dress up ς, whereas the semantics of RE and RL are consistent and straightforward in the TOG thanks to mereology and typing of the domain and range with TOG components. Alternatively, with MCS in FOL, we can add mereology to MCS.

For RP one can easily use the shifting mechanism m_s for the context switching relation, ς_s, and arbitrarily choose the shifting direction, hence, $RP \rightarrow \varsigma_s$. The TOG-information missing from ς_s is that it is an irreflexive symmetric relation (see *Lemma 3.18*) and typing of the relation so that it accepts C^ps only. The problem with this approach, as with the previous ones, is the need to enhance switching, which is already an addition to MCS. Last, one could define a set of bridge rules and add appropriate criteria that would satisfy the constraints on the TOG's GR subtypes. Aside from propositional/predicate logic distinction and properties of the GRs, one still can only consider *contents*, but not contents *and framework*.

Thus, in MCS, the bridge rule is limited to contexts' contents whereas a theory of granularity needs functions/relations for *both* contents *and* framework components, it lacks a mechanism for typing the contexts, and requires further specification of the switching mechanisms and of the switching relation. Nevertheless, representing levels and their perspectives as contexts is an appealing conceptualization because the TOG and contextual reasoning with MCS share underlying ideas, with demarcations (frame/box) and notions such as perspective and shifting. Alternatively, a FOL version of MCS could be interesting, but would need to be enhanced with, at least, more sophisticated bridge rules and a switching relation.

Because this contextual reasoning is insufficiently expressive for formally representing the TOG—hence, for meeting the key requirements for a theory of granularity—one could either encode only a subsection of the TOG or pursue further research into extending the features of MCS. First suggestions for the latter have been pursued by Schmidtke (2003, 2005), who defined a "grain" relation and a "context" relation for **sgpG** type of granularity (time intervals) that are to be added to a context. However, a formal integration of these granularity aspects with MCS has not been published yet. In more recent work (Schmidtke and Woo, 2007), "context" is used again but for mereotopology for qualitative granularity in spatial location, where the "context" corresponds in idea to TOG's granular level, and "grain size" to a value of the criterion. Despite the lack of a clearly defined framework with definitions, Schmidtke and Woo's approach to change **sG** spatial relations into qualitative, **nG**, type of granularity appears useful for additional reasoning scenarios (for robot navigation) and illustrates the interplay between quantitative and qualitative granularity.

5.2.3 Engineering solutions

There are several software implementations specifically for granularity that do not follow explicitly any of the previously mentioned formal approaches; that is, they focus on solving a specific problem in order to get a software system up and running that satisfies the system's requirements. The main problems with such engineering solutions were summarised in §1.1.3. In general, to reuse engineering solutions, *ad hoc* laborious manual translation efforts at the design-level specification or software code are needed to let software components interoperate. Conversely, the TOG can be seen as meta-layer for a global view and thereby achieve design- and implementation-independence and transportability. To some extent, this can be made possible with several prototype CASE tools (Fent *et al.*, 2005; Gandhi *et al.*, 1995; Luján-Mora *et al.*, 2002; Parent *et al.*, 2006a). However, these disparate tools lack the ontological and/or logical foundation and expressiveness of the TOG to represent granularity comprehensively. This can be remedied by adding TOG-constructors to the conceptual data modelling language so that it may percolate automatically to the design and implementation layers. Although this is not realised at the time of writing, the foundations to do so have been laid.

We now look at some of those engineering solutions, which are not immediately reusable, but do give hints as to which implementation aspects may be important and for which application scenarios.

The simplest approach taken is (re)use of taxonomies for enhancing retrieval of more or less granulated data or information from linguistic corpora or databases where the user selects the desired level of detail from the results (Zhang *et al.*, 2002; Doms and Schroeder, 2005; KIM; GoPubMed, 2007). Fagin *et al.* (2005) combines the two with their Multi-Structural Databases (MSD) to enhance the search parameters in the corpus cum database. The comparison of the MSD with the TOG is included in *Table 5.1*. Although MSD does not have several of the key requirements, it has three additional document information retrieval functions (recollect §4.2.6), which also can serve to find new levels to facilitate bottom-up development of a domain granularity framework. One of the main drawbacks of MSD is that levels are created on the fly with pairwise disjoint collections and tweaked to balance the level:content distribution ratio. This practice can be structured in conjunction with the use of ontologies so as to generate meaningful levels and to ease integration in a granulation framework so that the system then can avail of granular queries. For instance, it may be interesting to combine Fagin *et al.*'s MSD-approach with KIM or GoPubMed (2007) to enhance the search parameters in the news clippings corpus or PubMed, yet also have the (bio-)ontologies and user-friendly query interface that is well-known to domain experts. A drawback to such a combination may be performance. Both Zukerman *et al.* (2000) and Tange *et al.* (1998) had observed better performance with more but shallow granular perspectives. A GoPubMed+MSD might have an acceptable performance if it uses only the GO, which is, at present, a simple taxonomic tree with primitive concepts, but doing the same with the full FMA will result in performance degradation, given that querying the FMA itself is already slow (Keet, 2006b) and the OWL version of FMA is too large to load in ontology editors, let alone querying the ontology (Zhang *et al.*, 2006). Hence, performance-optimizing algorithms may be necessary, but such algorithms should not be conflated with the framework.

Natural language processing. Three main implementations for granularity with corpora passed the revue above: MSD (Fagin *et al.*, 2005) for newspaper clippings, GoPubMed (Doms and Schroeder, 2005) for scientific literature in biomedicine, and mining medical narratives or electronic health records at different levels of detail (Tange *et al.*, 1998). These solutions deal with structuring relatively unstructured data for the purpose of granular information retrieval. Distinct from this facet of granularity and linguistic corpora is McCalla *et al.*'s (1992) automated study advisor, which takes an already existing document structure—a textbook with chapters, sections, paragraphs etc.—and aims to retrieve the coarser- or finer-grained index numbers of the text-part(s) that is (are) related to a particular exam question. That is, usage of NLP and a table of contents through linking these indices to exam questions. With advances in semi-structured data and XML-enabled documents over the past 15 years, and the TOG, this will be even easier to implement. More recently, Sonamthiang *et al.* (2007) added more analysis dimensions of student behaviour other than their exam errors, such as combining student's actions with time spent on all or some exercises in conjunction with time granularity. Also, instead of passively taking a textbook structure, one can take into account its granularity from the moment of writing (Yao, 2007c), which, in turn, if done well, could facilitate NLP significantly.

Visualisation. This topic was already mentioned in §1.1.2, where the focus was on (i) visualisation of conceptual data models, ontologies, and, to a lesser extent, similar views like that of gene networks and metabolic pathways (Hettne and Boyer, 2005; Jarke *et al.*, 2005; Khatri and Draghici, 2005; Krivov and Villa, 2005; Krivov, 2005; Schmitt and Haaland, 2004; Tzitzikas and Hainaut, 2006), and (ii) the data retrieval that is necessary prior to the actual visualisation of the diagram. The granularity and abstraction functions that were introduced in Chapter 4 can be useful to ameliorate the problems of how to un-clutter such views, for they can rely on the levels and perspective(s) for query- and navigation capabilities, including diagrammatic query formulation. Although how this can be done was presented in the setting of abstraction (Keet, 2005b, 2007c), the proposed abstraction hierarchy neatly corresponds to a granular perspective and abstraction level to granular level; see §4.3.2 for details. In addition, with the TOG we also have the framework functions (§4.2) and the types of granularity so that transparent implementation of the functions for retrieval of the display-objects from the data source is facilitated even better than with abstraction alone.

5.3 Granularity for a particular subject domain

Most contributions in the area of granularity in a particular subject domain are of an informal nature, with the exception of time and geographic information systems. Ontology and formal theories are necessary, but need to be applicable to these domains and, vice versa, potentially interesting facets of the subject domain, even if informal, may have to be addressed by the theory. Because a main driving force in this research from the life sciences, this topic will receive ample attention in §5.3.1 (biology and biomedicine) and §5.3.2 (ecology and GIS). The last section (§5.3.3) touches upon time granularity.

5.3.1 Biology and biomedicine

The least-informal contributions in biomedicine are (Kumar *et al.*, 2005, 2004; Keet and Kumar, 2005), which have received attention in preceding chapters already. The main problems were illustrated, discussed, and subsequently resolved with the TOG—recollect §1.1.3 and *Example 2.6, 2.7, 3.9, 4.2*, and Keet (2006c). Aside from these issues, it is worth assessing both Kumar *et al.*'s list of seven "basic principles, which we believe can serve as axioms of a full-fledged theory of granularity" and their proposal for biomedical granularity against the list of key requirements put forward in Chapter 3. The latter is summarised in *Table 5.1*. Regarding the former, "we shall see different ways in which they need to be modified to deal with certain special cases" (Kumar *et al.*, 2005), which are principles 2 and 5. Principle 1 asserts that levels can contain only one kind of thing, such as types of cells only, hence restricting a theory of granularity ranging over universals only, even though granularity can be equally applied to instances, and excluding **nfG** type of granularity despite its wide usage. Principle 2 asserts that the types in a lower level must be part of those in a higher level, hence a mereology-only theory of granularity, which leads to a strange third principle: "Every level of granularity is such that summing all the grains together yields the entire human body" (Kumar *et al.*, 2005). Although taking the mereological sum to constitute the whole is common in mereological theories, it certainly is not the case that the whole always must be the "entire human body". Even if this were a list not of basic principles for *any* theory of granularity, but for human medicine *only*, then we would still run into problems: *e.g.*, is the mereological sum of all process in the human body the same "entire human body", what about body cavities, and does it include the microflora? This principle might be useful for just human structural anatomy in the context of medicine, but certainly not a key requirement for a generic theory of granularity. Principles 4 and 5 deal with "size", where the former says that the types (mentioned in principle 1) "do not need to be all of the same size" and the latter that the types "in a given level must be smaller in size than those entities on the next higher level of which they are parts.". As discussed before in §2.4.2 and *Example 2.6*, using physical size only causes problems, which Kumar *et al.* realised as well and they acknowledge violating their own principles 2 and 5, but instead of resolving it, they accepted "exceptions" for cardinal body parts and organ systems. Principle 6 inserts both practically and philosophically thorny issues: "With each level of granularity there is associated some *specific type of causal understanding* and thus some specific *family of causal laws*; when one moves up a level, then the grains on the lower levels become causally irrelevant." (emphasis added). The interested reader is referred to §5.2.1, references therein, *Example 3.7* regarding ticks and Lyme disease, and in particular Ellis (2005);

Johnson (1990); Lehman *et al.* (2004) for examples and research on representation of and reasoning with causation. Principle 7 is interesting, which requires that "[s]ome entities can change through time in such a way that one and the same entity (an embryo, a tumor, an organism) can occupy a sequence of different levels of granularity in succession". This, however, is not dealt with by Kumar *et al.* (2005), as their proposal, as well as the TGP that it builds upon, are atemporal; likewise, the TOG does not address granularity for temporal $\mathcal{DS}$s. Although principle 7 might appear to contradict key requirement 10, it does not. If key requirement 10 should be amended with '...at time t' or if a new requirement should be added to emphasise the atemporal and temporal characteristics is a point of further investigation. Concerning the OBO Foundry relations as laid down in the Relation Ontology (Smith *et al.*, 2005), it would only involve entity types related through the $transformation\_of$ relation, for then the participating types keep their identity. In general, it may require an extension with temporal logics and temporal conceptual data modelling (*e.g.*, Artale *et al.*, 2002, 2006), which is beyond the current scope.

Other contributions in the area of biomedical applications present preliminary models or human structural anatomy—canonical and extended to pathological structures—and little axiomatization for inter-level relations and set elements residing in a level (Elmasri *et al.*, 2007; Rector *et al.*, 2006; Ribba *et al.*, 2006). There are multiple informal text-based and/or figure-based descriptions of examples of granular levels in the biology domain (Grizzi and Chiriva-Internati, 2005; Hunter and Borg, 2003; Salthe, 1985; Toyoda and Wada, 2004; Winther, 2006, among others). For instance, we have (5.6) by Grizzi and Chiriva-Internati (2005) to organise human anatomy, which is shared by Elmasri *et al.* (2007) without the Organism-level, but also the more detailed (5.7) by Keet (2006c) and (5.8, 5.9) by Kumar *et al.* (2005, 2006), where (5.8) is intended to be generally applicable and (5.9) only in the context of colorectal carcinoma. To add more variation, one can place these granular perspectives against another single-purpose model with just three levels along two axes—structural anatomy (5.10) plotted against time (5.11)—which apparently suffices to model cancer growth of colorectal cancer (Ribba *et al.*, 2006). Keet and Kumar (2005) developed nine perspectives up to a depth of 3 levels in the subject domain of infections diseases, some of which were refined up to 6 levels for Phylogeny and 4 for Predisposing factors in (Keet, 2006c).

$$\begin{split} &Molecule \prec Sub\text{-}cellular\ Entity \prec Cell \prec Tissue \prec Organ \prec Apparatus \prec \\ &\quad Organism \end{split} \tag{5.6}$$

$$\begin{split} &Molecule \prec Cell\ part \prec Cell \prec Tissue\ part \prec Tissue \prec Organ\ part \prec Organ \prec \\ &\quad Organ\ System,\ Subdivision\ of\ principle\ body\ part,\ Principal \\ &\quad body\ part \prec Body \end{split} \tag{5.7}$$

$$\begin{split} &Biological\ macromolecule \prec Subcellular\ organelle \prec Collection\ of \\ &\quad subcellular\ organelles \prec Cell \prec Collection\ of\ cells \prec Tissue \\ &\quad subdivision \prec Tissue \prec Organ\ part \prec Organ \prec Cardinal\ body \\ &\quad part \prec Organ\ system \prec Organism \end{split} \tag{5.8}$$

$$Subcellular \prec Cell \prec Tissue \prec Organ\ part \prec Organ \prec Organ\ system \tag{5.9}$$

$$Molecule(Gene) \prec Cell \prec Tissue \tag{5.10}$$

$$Negligible \prec Hour \prec Days \tag{5.11}$$

Aside from the labelling issues, it may be clear that what exactly constitutes the "full" granular perspective for human structural anatomy is not yet settled, and that in at least some, but possibly all, practical models only a subset of the granular levels seem to be directly relevant. Instead of putting up with the plethora of incompatible hierarchies where each modeller devises his own version, the ideal case would be to have one inclusive perspective, where modellers can select the levels they need for their single-purpose model; hence, user preference at the front-end, yet interoperability through a shared perspective and levels among all the models at the back-end. This will be analysed in §5.5.1 in the context of dealing with more than one domain granularity framework.

Whereas the focus was mainly or exclusively on humans and medicine, above, and therefore macro-structures, cell physiologists and biochemists are developing the more detailed levels to cover anything cellular and sub-cellular. An example of such finer-grained characterization is ChEBI with 13078 entity types ranging from ions and radicals to complexes (ChEBI, 2007) and Degtyarenko and Contrino's (2004) ontology of bioinorganic proteins COMe that covers molecules and *parts* of molecules such as residues, whereas the modelling of cell signalling pathways starts with small molecules and moves up to multi-component complexes alike the Biological macromolecule-level in (5.8) and from elementary biochemical reactions to metabolic pathways, such as Energy production, at a higher level of granularity (Aldridge *et al.*, 2006). Molecular Dynamics goes even smaller down to the atoms, which is at present only informally linked to coarser-grained levels (*e.g.*, Ji *et al.*, 2007). With the TOG, it is easy to add such extensions in a *structured* and *consistent* manner so that the comprehensive granular perspective is *reusable* across life science information systems. Further, the TOG functions enable smooth switching between levels.

In a different direction, there are structured controlled vocabularies of anatomy of other types of organisms, too[10], which could be harmonised on granularity. As opposed to a list of about 100 "common terms" (SAEL, 2006) or ontology alignment in CARO[11], one could add the orthogonal granularity dimension to deal with anatomical structures that some, but not all, species have and let the latter skip a few levels to achieve better overall alignment than would be possible on the strict taxonomies and partonomies alone.

Current efforts in granularity in biomedicine, bio-ontologies, and medical informatics focus on *qualitative* knowledge. It is expected that this will extend to *quantitative* data in a structured approach, such as granularity in medical imaging, granularity in molecule concentrations in different parts of the cell for software simulations (such as PACE, 2004; E-Cell), and epidemiological studies. The latter encroaches on GIS with, *e.g.*, monitoring of the spread of contagious infections such as foot-and-mouth disease and SARS, or longitudinal studies of regional cancer incidence[12]. Granularity in GISs is the topic of the next section.

5.3.2 Ecology and geographic information systems

Ecology and geography deal with management of much larger (data) structures than occur in the type of information systems assessed in the previous section, and, more importantly, have a longer history in information systems development. Within the current scope, the spatial databases of GIS are of interest, which deal in various ways with granularity. Most of the proposals on representation and usage of spatial granularity have a data-centric focus (Bittner and Smith, 2003; Camossi *et al.*, 2003; Rigaux and Scholl, 1995; Stell and Worboys, 1998; Zhou and Jones, 2003; Zhou *et al.*, 2004), except for minor "adornments" for granular spatial and/or temporal entity types in conceptual data modelling languages, such as the Oracle Cartridge, Granular GeoGraph (Fent *et al.*, 2005), MADS (Parent *et al.*, 2006a), and DISTIL (Ram *et al.*, 2001). That is, they all lack any kind of *framework* for dealing with granularity and therefore use elaborate functions and queries to fill this gap and they have to find a way to deal with a variety of oftentimes inconsistent hierarchies that are incompatible across implementations. The TOG effectively

[10]A.o., zebrafish, mouse, flatworm, and fruitfly (respectively, *Brachydanio rerio*, *Mus musculus*, *Caenorhabditis elegans*, *Drosophila melanogaster*). http://www.obofoundry.org/ (date accessed: 9-8-2007).

[11]At the time of writing (August 2007), there is only preliminary information available at http://www.bioontology.org/wiki/index.php/CARO:Main_Page.

[12]See, *e.g.*, the International Journal of Health Geographics at http://www.ij-healthgeographics.com for case studies and implementations.

lifts the data-centrism up to the conceptual level and thereby makes it possible to structure the perspectives and levels consistently across implementations and, consequently, offer a new, simpler, way of querying the data in the granular levels. In addition, it leaves open the option to system developers to integrate the framework either in the database or in application software; hence, the TOG serves *both* multi-resolution *and* multi-representation spatial databases, and the pre-computed versus dynamically calculated population of granular levels[13]. In the remainder of this section, we go through several proposals within GIS and ecology research, respectively, with respect to their treatment of granularity[14]. Observe that regarding representation at the conceptual level, querying, and implementation, MADS (Parent *et al.*, 2006a)—refined in the MurMur project (MurMur; Parent *et al.*, 2006b)—is most comprehensive and therefore included in the comparison table *Table 5.1* and will be discussed below as well.

5.3.2.1 Modelling granularity for GIS

There are several sub-topics within GIS modelling and software development, most of which have a relatively long history (see Rigaux and Scholl, 1995, and references therein). Core notions that GIS systems deal with are resolution, (cartographic) generalisation and simplification, and multiple perspectives. Resolution refers to a minimum geometric measure that the focal object must have to be relevant and be included in the map. Definitions of generalisation and simplification are not consistent across the literature (Zhou and Jones, 2003; Zhou *et al.*, 2004; Stell and Worboys, 1998), but a distinction is made between plain reduction in resolution and hiding attributes or whole objects; that is, going from a detailed spatial shape to a simpler spatial shape on a map to represent some object (from Polygon to Point) versus going up in a taxonomy (from Wheat to Cereal)[15]. Recollecting the taxonomy of types of granularity (§2.2), and to be more precise than the myriad of examples in GIS literature, GIS applications deal with the types **nasG**, **sgrG**, **sgpG**, **samG**, and **saoG**, which is a subset of all types. Thus, the emphasis is on scale-dependent granularity, although the distinction with non-scale-dependent granularity—in GIS regularly referred to as *semantic granularity*—has been well noted (Camossi *et al.*, 2003; Fonseca *et al.*, 2002; Parent *et al.*, 2006a; Rigaux and Scholl, 1995; Stell and Worboys, 1998). In particular, Bittner and Smith (2003) and Rigaux and Scholl (1995) attempt to tackle the latter through the notion of partitions using mereology and set theory, respectively, and Stell and Worboys (1998) with a "granularity lattice" as a set of levels of detail. This granularity lattice is made up of pairs $\langle\sigma, \tau\rangle$, where each pair is denoted with a granularity g_i, σ denotes the spatial level of detail, and τ a given depth in a taxonomy, to which an order is applied ($\langle\sigma_1, \tau_1\rangle \leq \langle\sigma_2, \tau_2\rangle$ iff $\sigma_1 \leq \sigma_2$ and $\tau_1 \leq \tau_2$; hence, then also $g_1 \leq g_2$) and a set of maps is associated to each pair $\langle\sigma_i, \tau_i\rangle$. The only use of g_i is to go from a particular map at g_i to its adjacent coarser map g_j or vice versa using the "Lift" or "Gen" functions, respectively. However, it does not go further than this rudimentary notion of granular levels in a lattice at the logical level. The object-centred formal approaches of Bittner and Smith (2003) and Rigaux and Scholl (1995), on the other hand, do not deal with levels of granularity, but focus on constraints on the objects, such as pairwise disjointness of the subsets (partitions or granules), covering constraints, and behaviour of geometric attributes. A different approach to non-scale-dependent granularity is the introduction of ontologies with Fonseca

[13]"For a spatial database system to support applications that require variable level of resolution, one approach, called the multi-representation approach, is to pregenerate and store spatial data at different resolution levels. The other approach, called the multiresolution approach, is to store only the data at the highest level of resolution and simplify and generalize data dynamically." (Zhou *et al.*, 2004), which is also referred to as "database resolution" (storage) versus user-centric "presentation resolution" (Zhou and Jones, 2003). For an assessment on their performance trade-offs, the reader is referred to Zhou *et al.* (2004).

[14]GISs offer additional functionality, such as overlay maps, approximations and vagueness, performance optimizations, and time, which are not dealt with in this section; refer to §5.3.3 for time granularity.

[15]So, one has either a particular object x, say, the Louvre in Paris, that is represented as polygon at resolution $r1$ and as point at resolution $r2$, with $r1$ being finer-grained than $r2$, or a land parcel that has plants of type Y (say, wheat) at $r1$, and Z (cereal) at $r2$, with $r1$ finer-grained than $r2$ and $Y \subset Z$; see also Chapter 2.

et al.'s (2002) Ontology-Driven Geographic Information System ODGIS. In addition to the taxonomic "vertical" granularity (Lake is finer-grained than Body of Water), Fonseca *et al.* add a "horizontal" component at same level of detail where another property of the type is highlighted, *e.g.*, Lake in the role of Protected Area. Likewise, Camossi *et al.* (2003) and Zhou and Jones (2003) add the notion of criterion to emphasise a particular property within a hierarchy, which is in contrast to the unrestricted levels in DISTIL that are defined on the fly by an end user modeller (Ram *et al.*, 2001). These works provide a basic treatment of highlighting one or more properties of an entity (/type), which has been investigated in §3.3: the ontologically-motivated analysis has resulted in the unambiguous granularity components *criterion* and *granular perspective* in the TOG in order to enable consistent representation throughout and between applications. One usage of the informal hierarchies, however, merits further investigation to characterize it formally: conditional granular levels across perspectives. The intuition is illustrated by Camossi *et al.* (2003), which we limit here for two granular perspectives, being "administrative boundaries map" with four levels and a "hydrographic map" that classifies rivers on their water flow, also with four levels; see *Table 5.2*. Irrespective if the levels make sense ontologically, if the given values in Hydro-gp are indeed the values used for map-making with respect to the administrative boundaries, and that Camossi *et al.* (2003) awkwardly calls the "$\Leftrightarrow$" an "`equivalent-to`" relation, the intention is to represent somehow the constraint that *if one makes a map with granularity at the Province-level then only rivers with a flow $\geq$ 10 000 litres/min should be included in the map.* Generalising this constraint, then one has a conditional selection and retrieval:

if $selectL(gp_a gl_i)$ *and* $getC(gp_a gl_i)$*, then* $selectL(gp_b gl_j)$ *and* $getC(gp_b gl_j)$ *as well*

Although it may seem a peculiarity for GIS, we could apply such a constraint to (5.10) and (5.11) for cancer growth of colorectal cancer as well, alike *"if the medical doctor needs a day-by-day view of the growth of the cancer in patient1, then deliver the tissue samples"* as opposed to delivering cell cultures or microarrays. This type of constraint is neither dealt with in the static components (Chapter 3) nor added as compound function in the dynamic components (Chapter 4) of the TOG, because such constraints can be declared for the domain granularity framework only and, as shown with the generic constraint, can be dealt with adequately with the currently defined TOG functions. It, however, merits further investigation if and how this type of constraint could be added to the TOG and where and how it adds value to a granularity-enhanced information system in general.

Table 5.2: Two sample perspectives for some GIS application (based on Camossi *et al.*, 2003).

Admin-gp		**Hydro-gp**
Country	$\Leftrightarrow$	(river with flow $\geq$) 100 000 litres/min
$\uparrow$		$\uparrow$
Province	$\Leftrightarrow$	(river with flow $\geq$) 10 000 litres/min
$\uparrow$		$\uparrow$
Region	$\Leftrightarrow$	(river with flow $\geq$) 2500 litres/min
$\uparrow$		$\uparrow$
Municipality	$\Leftrightarrow$	(river with flow $\geq$) 1000 litres/min

Returning to the notion of 'horizontal' navigation and granular perspectives, the feature of multi-representation for different types of users—or: different (granular) perspectives—is most comprehensively addressed by the MADS conceptual data modelling language and tools (Parent *et al.*, 2006a,b). Although MADS does not support explicitly granular perspectives and levels as in the TOG, it deals with these notions for individual entity types and relations in a conceptual data model (which has formal foundation, see appendix A in Parent *et al.* (2006a)). Each entity type and relation in a MADS conceptual data model can have extra tabs for each perspective on the type so that upon clicking it, one sees attributes relevant to the chosen perspective only, such as for the engineers or managers perspective. One main distinction with the TOG, however, is that although there is one rectangle with tabs that suggests a unified approach to the entity types, in the formal representation underneath, a multi-viewpoint entity type has as much different entity types, such as AvalancheEventE and AvalancheEventM for Engineers or Managers (see Figure 1 in Parent *et al.* (2006b)). In contradistinction, with the TOG it is assumed there is *one* entity type and for each perspective a subset of the attributes (properties) is considered. However, the two approaches probably could be made compatible with little programming effort and a few additional axioms in the MADS formalisation. Furthermore, the main focus of MADS is conceptual data modelling for spatio-temporal databases and GIS applications in particular (hence, aspects such as indistinguishability and relations between entity types at different levels of granularity do not receive attention). To this end, it already supports the standard spatial data types[16] for spatial entities at different levels of granularity, whereas this is not pre-defined for the TOG because its aim is more generic so as to have a constructor for granular perspective (GP) where then the ISO categorisation in an instance of it; the same holds for the temporal aspects[17]. One can, of course, use such spatial (temporal) entities for defining the levels of a perspective in a domain granularity framework and subsequently integrate it with a spatial (temporal) database that adheres to these ISO standards. In addition, MADS has a comprehensive data manipulation language with algebra and conceptual query builder tool, which, similar to the data warehousing functions, is compatible with the TOG. Refer to *Table 5.1* for the features that the TOG has but that have not been addressed in MADS. One of the possible avenues of further research is to investigate how one best can augment a conceptual data modelling language with the TOG components, primarily with spatio/temporal languages such as MADS for immediate benefits in GIS, as well as to the generic conceptual data modelling languages like UML and ORM.

5.3.2.2 Extending the GIS domain to ecology

Information systems for ecology (and climatology) complicate GISs. They require the geographic information, temporally indexed, and then add other pieces of information and dimensions. One relatively straight-forward extension are the ecosystem hierarchies such as the "National hierarchical framework of ecological units" used by the US Department of Agriculture's Forest Service (Bailey, 1995), which as been analysed in detail by Sorokine *et al.* (2004, 2006). Their rigour to distinguish within that classification between a partonomy of individuals for particular regions to be part of a larger region and a taxonomy of classes for types of ecological units helps disambiguation, but does not solve the core problem with the levels and their contents. For instance, one has hierarchies such as California coastal steppe, Mixed forest, and Redwood forest *Province* $\prec$ Mediterranean *Division* $\prec$ Humid temperate *Domain*. To see what is principally wrong with such a hierarchy, we start with Appendix 1 of (Bailey, 1995), which contains the mapping from the USDA ecological units to the widely used Köppen climate classification equivalent, which are for the top two: Subtropical dry summer (Cs) $\prec$ C–Subtropical Climates. That is, Köppen's system (and similar ones) is based on a *combination of properties independent of a particular area and time* and each region that satisfies those properties is classified accordingly[18]. In contrast, the USDA system orders along the line of 'the kind of weather we see in California', which changes over time and does not indicate any particular characteristic. The Köppen classification meets the constraints of a TOG granular perspective, the informal USDA ecological units does not. Moreover, if we indeed look at *ecological* units and not just the physical geography

[16]ISO TC 211, Geographic Information—Spatial Schema, ISO 19107:2003.

[17]ISO TC 211, Geographic Information—Temporal Schema, ISO 19108:2002.

[18]"It is based on the concept that native vegetation is the best expression of climate, thus climate zone boundaries have been selected with vegetation distribution in mind. It combines average annual and monthly temperatures and precipitation, and the seasonality of precipitation" Wikipedia [http://en.wikipedia.org/wiki/Koppen_climate_classification], with source attribution: McKnight, T.L., Hess, D. (2000). Climate Zones and Types: The Köppen System. In: *Physical Geography: A Landscape Appreciation*. Upper Saddle River, NJ: Prentice Hall, pp200-201.

(temperature and humidity) for climate, then Steppe, Mixed forest, and Redwood forest are rather distinct units for they have different vegetation—and it is the latter that ecology is interested in. This ranges from forest indicators such age structure/diameter distribution, deadwood, and tree species composition among 35 properties and a tentative refinement of the typology of forest types (Barbati *et al.*, 2007), to niche-overlap of co-existing species at coarse-grained land plots at larger time intervals to effective spatio-temporal niche differentiation when observed at finer-grained values[19]. Even demarcation of a particular ecosystem is scale-dependent: Earth $\succ$ Continent $\succ$ Mountain ridge $\succ$ River in the mountains $\succ$ River sediment all can denote an ecosystem at its own level of granularity. There is a wide range of mostly quantitative properties—often called parameters or indices in ecology—that are taken into the equation in ecology (see, *e.g.*, Ricklefs and Miller (2000) for comprehensive treatment and examples), where choosing the wrong scale can lead to false conclusions. In addition, sub-disciplines such as molecular ecology and, as discussed in Chapter 1, metagenomics, add additional levels of granularity to ecology. Structuring this kind of information, both ontologically in the sense of developing even simple taxonomies (Keet, 2005a) and a structured approach to granularity other than informal granularity hierarchies (Salthe, 1985) requires considerable effort to achieve. The TOG may facilitate this endeavour.

5.3.3 Time

Although time granularity is an active field of research, it hardly receives any attention in biological modelling, where even time without granularity is only one of the components. Moreover, at the time of writing, there are very limited options in the commonly used knowledge representation languages to deal with time; therefore, time and time granularity will receive relatively little attention in this chapter, but is an area of current and further investigation. After a brief general introduction of topics in time granularity, I will introduce some of the characteristic issues in time granularity through examples with calendar hierarchies and how the TOG could be of use. Subsequently, Euzenat and Montanari's (2005) list of key components and dimensions is summarised and discussed where the TOG meets them.

Introduction. For good introductory overviews on time granularity, the reader is referred to Bettini *et al.* (1998) for a glossary, Euzenat and Montanari (2005) for main theories and fundamentals of time granularity, and (Hobbs and Pan, 2004; Hobbs and Pustejovsky, 2003, §2-3) for implementable OWL Time axioms. Recurring themes are representing and reasoning over granularity in time, granularity in change of time, and calendar hierarchies. For a diversity of approaches, sub-topics, and usages, consult, *e.g.*, Bettini and Ruffini (2003); Böhlen *et al.* (2006); Dal Lago and Montanari (2001); Mota *et al.* (1995); Schmidtke (2005); Stell (2003b) who investigate, respectively, granularity conversions within time granularity, aggregation of time intervals in temporal databases, finite representation of infinite time granularities with automata, seasonal cycles, time in context, and granularity in the changes over time. On the theoretical side, the three main existing formalism are set-theoretic, logical (on quantitative time granularity), and the relational algebra granularity conversion operators for qualitative time granularity (Euzenat and Montanari, 2005).

Before we look at calendar hierarchies, two basic components of time have to be introduced, which are time *interval* and *duration* that both use time *instant* (point in time), with *begin* and *end*, and *inside* a time interval. Then, an interval is a temporal entity having an instant demarcating its start and another instant demarcating the end of the interval. Duration involves the arithmetic according to some scale whereby one temporal unit is composed of smaller temporal

[19]For instance, grassland ants that have dawn, afternoon or nocturnal foraging habits—hence, also at slightly different temperatures—and whose routes are apart if observed in the, say, 5 m^2 plots as opposed to 100 m^2 (Albrecht and Gotelli, 2001).

units. For example, Day has a *duration* of 24 hours and an *interval* from 00:00 to 23:59 if we were to take the level containing minutes as direct finer-grained or lowest level, thereby ignoring finer-grained seconds and other granular perspectives and levels. In calendar hierarchies and calendar software, interval and duration are often mixed, and periodicity added; *e.g.*, a recurring calendar appointment for an hour long group meeting starting at 10am two days before the deadline for submitting agenda points for the faculty council that is held every first Thursday of the month except if it falls on a public holiday. The TOG does not provide the full machinery to deal with such complex time granularity, because it also requires non-granular time operators and inter-level constraints on entities that are not covered by the theory; for definitions and implementation suggestions see (Bettini *et al.*, 2007; Dal Lago and Montanari, 2001; Euzenat and Montanari, 2005; Ning *et al.*, 2002).

Calendars and ontological commitment. We take a closer look at several fundamental characteristics of calendar entities and relate them to TOG components, which enforces clear modelling so that implicit commitments can be made explicit. Issues with calendar hierarchies are primarily due to insufficient ontological analysis, and only secondary due to granularity issues. For instance, within the scope of aggregation in data warehousing in enterprises the hierarchy goes along the lines of Hour $\prec$ Day $\prec$ Week $\prec$ Month $\prec$ Quarter $\prec$ Year, where Day is, implicitly, Working day, which may be Weekday *sensu* ISO standard ISO-8601, but could also entail a shift-workday in the 'weekend'. But recollecting the cancer growth of colorectal cancer in (5.11), then each hour and each day—including lunch breaks and weekends—counts in disease progress, which therefore always requires standard conversion functions such as Hour * 24 $\equiv$ Day and Day * 7 $\equiv$ Week—be it as mereological sum or a set. In contrast, one uses *ad hoc* aggregations of working hours or days and sums for calculating, say, a monthly employee salary based on the hours worked. Thus, while superficially the same, the hierarchies are ontologically distinct; hence, following the TOG's granulation criterion and the types of granularity, there are *two* granular perspectives to capture this distinction.

A second ontological issue with the calendar hierarchies is the commitment to an Urelement (also called bottom element, greatest lower bound, or chronon) and its set-theoretic representation versus mereology. The former determines the granularity of each level by virtue of the arbitrary choice of the Urelement, whereas with mereology that can be adjusted for each level to make interoperable the different extensions through the same concept. For instance, when we take Second as Urelement, then Year is not 365 or 366 days—which it would have been if we had chosen Day as Urelement—or 365.24 mean solar days, but precisely 31 569 260 seconds. Conversely, with mereology, we can specify each calendar unit in the hierarchy either by its next finer-grained type it has as part or by its *intension*, *i.e.*, properties, of Year. The latter approach also enables interoperability among calendar hierarchies based on different calendars, such as the Islamic calendar, where the meaning of Year may be similar enough but the extension different. Expanding on this notion, then with converter software for days, one can calculate not only that Thursday 21st of April 2005 (Gregorian) coincides with (Hijri) yawm al-khamis the 12th of Rabi-al-Awwal in 1426 AH, but also have specification of the intension of particular days and find out that that day was Miladun-Nabi (birth of the Prophet) that is equivalent to the meaning of Christmas Day. On the other hand, there are distinct advantages to the Urelement and set-theoretic approach for certain applications. In particular, it makes operators for granularity conversions and interval aggregations easier (Böhlen *et al.*, 2006; Bettini *et al.*, 2007; Ning *et al.*, 2002). For instance, the calculations between weeks and months is far from trivial, but can be implemented with two straightforward conversion functions and an Urelement like Day: one for month down to days, and one from days aggregated into week (see Bettini *et al.*, 2007; Ning *et al.*, 2002, for details). The TOG can accommodate *both* approaches through assigning the appropriate type of granularity to each perspective and levels—**samG** with the Urelement and **nrG** for the mereological representation. It is outside the scope of the thesis to subject calendar hierarchies to a rigorous ontological analysis, to describe a full domain granularity framework, and to cover all possible calendar operations to have a common framework for all calendar details. As has been demonstrated, the TOG can aid making implicit assumptions explicit and contains the components to represent such distinctions, which are neither at one's disposal with star schemas and other data warehouse conceptual- and logical modelling languages (Abelló *et al.*, 2006; Franconi and Kamble, 2003; Kamble, 2004; Malinowski and Zimányi, 2006) nor with more comprehensive formal approaches for calendar hierarchies (Bettini *et al.*, 2007). Thus, it is feasible to define proper hierarchies of calendar units that adhere to different granulation criteria and type. The TOG functions for querying levels are trivially extensible with Bettini *et al.* (2007) and Ning *et al.* (2002)'s operations on labelled sets for granularity conversion.

Requirements for and components of theories of time granularity. The list of possible components and characteristics of theories for time granularity (Euzenat and Montanari, 2005) will be summarised and the TOG positioned with respect to the plethora of time granularity formalisms. Thereby it will give an idea of which facets of time granularity can be captured with the TOG. No time granularity theory meets all the properties, because several properties are conflicting, such as having a partial order or a total order and using propositional, first, or second-order logic. Also, each one has its trade-offs; the interested reader is referred to (Euzenat and Montanari, 2005) and references therein for details.

The first decision concerns the language to represent granularity and the choice to focus on qualitative or quantitative characteristics of the language on which a temporal representation language is grounded; *e.g.*, set-theory, topology, metric spaces, and vector spaces. In addition, one has to consider expressive power of the chosen language, which can be exact and conjunctive, propositional, first-order, or second order logic (the latter for quantifying over levels of granularity). Second, one has to deal with representing "layers" or levels and their properties. Important decisions for this concern the structure of time—continuous, dense, or discrete—and global organization of layers—partial order, (semi-)lattice, or total order. In addition to this ordering of levels, it is possible to go into more detail regarding pairwise organization of layers, such as assessing properties of homogeneity and alignment—and within these, the four cases year-month, year-week, month-day, working week-day—and how the objects in the levels behave, that is, if they persist, change category, or vanish. Third, there is a wide range of types of operators and their properties. Two basic ingredients are to be able to select a level ("contextualization") and to move across levels ("projections"), which allows one to make granularity conversions from coarser- to finer-grained detail and vice versa, comparing granularities (which one is coarser/finer-grained), and to check if two representations represent the same thing at different level of granularity. One set of properties for operators to change in granularity include reflexivity, symmetry, order-preservation, transitivity, oriented transitivity, and downward/upward transitivity, and the other set of properties concern properties of levels, being contiguity, total covering, convexity, synchronization, and homogeneity. Fourth, and last, there is a notion of internal versus external layers to determine "if an existing formalism will be extended with these new [time granularity] operators or if these operators will be defined and applied from the outside to representations using existing formalisms".

In these terms of formal theories of time granularity, the TOG is agnostic about the structure of time and how objects behave, but permits both homogeneity (**naG**) and alignment (**nfG**, **nacG**) on the organization of layers, commits to a strict total order between levels (RL), is mainly in first order with some second-order logic as syntactic sugar for the static component and uses the second order for a subset of the functions, meets several of the properties on the operators, such as transitivity (up to acyclicity), and has the two essential types of operators for level selection and moving across levels. The TOG's options for linking granular perspectives provides a rudimentary way for (time) granularity conversions, which leaves room for extensions with more granularity conversion operators. Further, the TOG allows for both quantitative and qualitative granularity and the Urelement. Moreover, which has not been covered by Euzenat and Montanari's list, one has to be explicit in the *criterion* by which one granulates, as discussed in the preceding paragraph.

5.4 Abstraction

This section contains a literature overview on abstraction, which extends §4.3.1 that summarises the problems with the here reviewed contributions. How the abstraction functions with the TOG solve the issues is described on page 136 and a comparison of the abstraction solutions from the literature with the TOG will be provided in §5.4.2.

5.4.1 Manual, automatic, and complex abstractions

Manual abstractions. Manual efforts of abstraction has been, and is being, carried out informally or in somewhat structured fashion with UML modularisation (SemTalk; OMG UMLSpec, 2005) during software development, (E)ER clustering (Jäschke *et al.*, 1993) and the Abstraction Hierarchy (AH) (Lind, 1999; Yu *et al.*, 2002), which are laborious and intuitive *ad hoc* methods. AH does not have a supporting modelling paradigm like UML and (E)ER, and is based on stepwise decompositions from high-level Purpose via abstract-, general- and, physical functions, down to Physical Form, alike an informal top-down evaluation strategy. The relative freedom in applying this loose structure has its counterpart in the life sciences with the use of "black boxes" (*e.g.*, Hunter and Borg, 2003, among many), like the MAPK Signalling and Second Messenger System of *Example 2.2* and Cellular Process box in tools such as PathwayAssist. Abstractions in the software-supported modelling of ecology such as with such as STELLA (ISEE) are also manual. Comparing STELLA models, then differences between levels of detail involve more entities and relations and/or additional attributes (Keet, 2005a; Tett and Wilson, 2000). This latter aspect also emerges in EER abstractions, where the same entity can recur in different levels of granularity, called "major types" (Jäschke *et al.*, 1993), which is inconsistent with an ontological approach toward granularity and abstraction, unless one is more precise in the graphical representation and MSU. Jäschke *et al.* (1993) focus on generalisation and dominance grouping for entity abstraction and define several rules to restrict relationship clustering. SemTalk allows the modeller to expand and abstract as he pleases, but, given that there is a mapping from the graphics to OWL and that one can avail of the Cerebra reasoner (Fillies *et al.*, 2005), it is within reach to implement automated abstraction and expansion. However, at present, neither the mappings are disclosed nor how the parent-children relations are dealt with formally. A different manual abstraction approach that does not suppress the details, but abstracts away that what is deemed less important because it is *non-functional* as *e.g.* (Hanahan, 2000, fig. 3 p60) shows for the "reductionist view" versus "heterotypic cell biology", is not elaborated on further here because prioritization of information at the same level of granularity is distinct from leaving out finer-grained details.

(Semi-)automatic abstractions. A Java implementation of automatic abstractions through clustering on ER diagrams has recently been developed by Tavana *et al.* (2007) who reused techniques from machine-part cellular manufacturing, but it does not iterate for several abstractions in sequence. Zukerman *et al.* (2000) used abstraction with dynamic Bayesian networks to model and test predictive accuracy of computer game player's interest, where the abstraction function is tailored specifically to the gaming setting: locations/actions performed more often by the gamer make it into a higher abstraction layer that represents them as action classes or groups of locations. Automatic abstractions based on heuristics have been developed by Campbell *et al.* (1996), who aim to simplify large ORM models by suppressing roles and objects that, based on the encoded semantics, are deemed less relevant; *e.g.*, a role for identification is more important than roles not used as key. Summarising these rules, we have:

Rule 1: mandatory roles, 10 points
Rule 2: unary roles, 10 points
Rule 3: non-leaf object types, 9 points (thus, delete leaves)
Rule 4: smallest maximum frequency, 8 for uniqueness constraint, else lower
Rule 5: non-value types, 7 points (thus, value types have lower importance)
Rule 6: anchor points, 6 points
Rule 7: single-role set constraint, 5 points
Rule 8: multi-role set constraints, 4 points
Rule 9: set constraints and anchor points, 3 points
Rule 10: joining roles of set constraints, 2 points
Rule 11: first role of set constraint, 1 point
Rule 12: first role of internal uniqueness constraint 1 point

A salient aspect of the rules is the emphasis on treatment of role-set constraints, which translate to projections over tables, If absent—be it in an ORM model or a logical theory in another language—then 12 rules reduce to only 7. Each successive higher level is reached by firing all rules and removing from the graphical view of the model those object types and their roles with lowest weight values. Campbell *et al.*'s rules focus on semantic importance *assumed* from syntax, as if that what is not meant in the rules must be unimportant. However, for instance, a mandatory role does not *imply* semantic importance in the UoD: mandatory fact types can be important to its immediate neighbours, but in the overall semantics in the whole model may play only a secondary role, or a subsection of a model can be relatively isolated, or be just 'nice to have' but not essential according to the domain expert. No model captures this and relying on encoded semantics will not address it either. On the contrary, if one knows the model, abstraction heuristics are influenced by such background information. Campbell *et al.*'s case study ORM model contains several role set-related rules, which are not necessarily a salient feature in ORM models in general and certainly not in other conceptual data modelling methods. If other models or modelling languages had been used as first case study, rules (2, 7-11) would not have been included as such, although most aspects of the remaining rules are, heuristically, useful. This illustrates limitations of a bottom-up case study based approach: generalising from a bottom-up approach may require tweaking a methodology to make it generic. Extended rules were tested on more models (Keet, 2005b), which demonstrated that those theoretical improvements of abstractions were useful to improve heuristics and make them applicable to a wider range of models and modelling languages. Improved heuristics that are tested with more (ORM) models may be useful for identifying (contents of) granular levels in a specific subject domain.

Ghidini and Giunchiglia (2003) formalised abstraction by exploiting Local Model Semantics of context reasoning (§5.2.2.3) and formalising abstraction "as a pair of (formal) languages plus a mapping between them", where the mapping is a syntactic abstraction with the abs function. Given the ground language L_0 and the abstract language L_1, symbol abstraction operates on constant symbols (5.12), and likewise for function symbols (with $abs(f_i) = f$) and predicate symbols (as $abs(p_i) = p$). Arity abstractions on functions and predicates lower the arity by one; where the arity in L_0 is one, the function maps into a constant, and a predicate into a proposition.

$$c_1, ..., c_n \in L_0, c \in L_1 \text{ and } abs(c_i) = c, \text{ for all } i \in [1, n] \tag{5.12}$$

The examples Ghidini and Giunchiglia (2003) provide for function and predicate abstractions involve taxonomic generalisation, as in collapsing $walkfrom(loc)$, $drivefrom(loc)$ and $flyfrom(loc)$ into $gofrom(loc)$ "abstracting away the details of how one moves from one location to another". Thus, what their abs function actually does, is, given an entity A, to return its taxonomic parent entity B. However, this ignores collapsing sequential or parallel sub-processes. For example, a statement of the type "John goes from Rome by train" abstracts away that John walked from the apartment to the bus stop, took the bus to the train station and then took the train whilst reading a book and listening to music, which could, or maybe should, be catered for as well. The syntax and subsumption can be a component of granularity, but their approach overloads abs, thereby in itself abstracting away the finer details of the process of abstraction.

Complex abstraction rules. In contrast to the manual and informal abstractions, bottom-up developed heuristics, and the pure syntax-based formalism, Mani (1998) introduced four types of abstraction, in addition to "type shifting operators" for accomplishing grain size shifts. Simple type shifting goes from coarse to finer-grained and back with event to processes ($abs_{\chi}(x) = y$ where $Process(x) \wedge Event(y)$), process to objects ($abs_{\theta}$), process to states ($abs_{\omega}$), and three abstraction operations to fold process and objects, processes and states, or to fold events and propositions, respectively μ, ϑ, and ξ where $abs_{\mu}(x, y) = z$ s.t. $(abs_{\chi} \cup abs_{\theta})$, $abs_{\vartheta}(x, y) = z$ s.t. $(abs_{\chi} \cup abs_{\omega})$, and $abs_{\xi}(x, y) = z$ s.t. $(abs_{p1} \cup abs_{p2})$. Both "meronymic abstractions" and "sortal" (set-based) abstractions are introduced to cope with natural language semantics. Although Mani's family of abstraction functions is developed for dealing with polysemy and underspecification, it is a relatively comprehensive treatment that captures the varying semantics of abstraction better—like the abs_i functions in §4.3 do, too—than other approaches. These functions can be useful in particular with entities in non-scale-dependent levels of granularity. For a biology domain, one can think of abstracting biological complex entities like Second Messenger System or MAPK Signalling (*Example 2.2*), thereby working toward a computational abstraction instead of the present manual approach. Mani's approach has the same intention as the property of compositionality of the MI abstraction of Pandurang Nayak and Levy (1995) (extending Giunchiglia and Walsh (1992)), who claim that theory compositionality is intrinsic to abstractions. Pandurang Nayak and Levy use a two-step process 1) abstraction of the intended domain model and 2) define set of formulas that formalises the abstracted domain to make the simplifying assumptions explicit in the base (more detailed) model. This implicitly assumes one already has the two levels, albeit not in the structured and consistent manner as with the TOG. Mani used the same underlying idea for asserting indistinguishability under a certain abstraction, alike the φ-indistinguishability, and "relative grain size" to have two level of granularity, but did not develop these ideas further.

5.4.2 Comparison with abstraction for the TOG

Summarising the three main problems of the solutions for abstractions (§4.3.1): abstraction focuses only on the contents of a level, thereby lacking a surrounding *framework*; a general abstraction function abs does neither reveal *what* it is abstracting nor *how*; and existing proposed solutions are mainly theoretical and not developed for or assessed on its potential for reusability and scalability. A framework (the TOG), three types of abstraction, and 14 abstraction functions for both basic abstractions—including taxonomic and mereological abstractions—and complex folding were introduced in §4.3.2, so that abstractions are scalable, unambiguous to implement, and amenable to automation. This can be achieved already without availing of a full-fledged theory of granularity, as has been demonstrated in (Keet, 2007c). Integrating abstractions with the TOG comes at a cost of subjecting it to more constraints, but also entails additional advantages (see also §4.3.2 on p.136). Taking into account the types of granularity, an additional abstraction function (abs_{ph} and one more constraint (*Constraint 4.2*) could be defined. Regarding restrictions on abs in the TOG, R-ABS only applies to the *GR*s that are converted into a function and the suppression of information from a visual representation of D-ABS is not covered. The latter can be done with some engineering rework by assigning the visual result of each iteration of suppressing information from the diagram as a granularity/abstraction level, and denote each element (object type and relation) in the visual result as a parent type of their corresponding elements in the finer-grained visual representation, or do this through an indexing system alike with multi-representation databases. Alternatively, or in addition, and recollecting the permitted abs_{d2} for deletion and disallowed abs'_{d2} for the hiding of quality properties, more functions can be defined for other to-be-hidden elements. Such simple hiding, however, breaks down with theories of over about 100 entity types (see §1.1.2, §5.2.3, and references therein), and may be augmented with a generalisation of Campbell *et al.*'s rules and their weights for ontologies and several conceptual data modelling languages to find out what the *important* elements are in the large diagram (that is the interface to a logical theory). Campbell *et al.*'s rules are written in the format of "if ϕ then keep it" instead of "if ϕ then abstract it away", which can be adjusted easily; *e.g.*,

Rule 3: *[keep] non-leaf object types*

can be rewritten for, *e.g.*, parthood relations, where $\neg(ppart\_of = R)$, as

Rule 3′: *if* $part\_of(\phi, \psi) \wedge \neg R(\phi, \varphi)$, *then* $abs_{po}(\phi) = \psi$.

Experiments carried out with a section of the FMA and the full Bacteriocins conceptual data model using such enhanced rules yielded promising results (Keet, 2005b). Observe also that, like discussed in the GIS section (§5.3.2.1), with the granularity framework one can specify rules between levels, which is a feature not readily possible in other theories of granularity and abstraction. Even without such a possible extension to enhance the abstraction functions with rules, the TOG has not only a foundational framework, but also several extra features. The notions of abstraction level and abstraction hierarchy are useful addition within the context of abstractions (Keet, 2007c), but the granularity framework offers, besides (i) consistency in the applicability and usage of the functions, also (ii) the option to transform the functions to make them applicable at the instance level whilst keeping the foundations and, more importantly, (iii) have the functions relate to a particular type of granularity (through the relation with the level), which, in turn, (iv) greatly facilitates abstraction's 'inverse' relation expansion. Expansions functions can be complex without the full TOG, but within the granularity framework they keep the intuitive elegance of (near-)inverse of the corresponding abstraction functions. Any complexities of the expansion functions are built-up from—or, broken down into—smaller, reusable, sub-functions that are nested in the execution procedure of the main function. Last, although one can define functions for abstraction levels and hierarchies, they are already available with the TOG and directly usable to facilitate management of abstractions.

5.5 Limitations of and opportunities for the TOG

Although the TOG compares favourably with other theories of and solutions for granularity, there may be points for additions to the theory and some trade-offs when transferring the TOG to the design and implementation stages. We first look at theoretical considerations (§5.5.1) and then at implementation possibilities (§5.5.2).

5.5.1 Theoretical issues

Limitations originate from the language used—and required—to formalise the TOG and from the theory-data interface, which will be discussed in the next two sections. In addition, the case of more than one domain and domain granularity framework will be touched upon.

5.5.1.1 Complexity and language

From the Mace4 model checking (§3.8.3) it already has become clear that (i) inclusion of the definition of indistinguishability causes the model searches to go on 'forever', whereas keeping it as a primitive relation results in a model for the TOG, *i.e.*, then *by computation*, it is *known* that the TOG is satisfiable; (ii) the satisfiable input is not scalable because it does not terminate within a reasonable amount of time from domain size 9 onward, and (iii) the obtained statistics for domain size against CPU time fits very well with an exponential trendline. Ideally, one would want to have the full TOG known to be satisfiable, that automated reasoning will terminate with certainty, and scalability to more realistic domain sizes, *i.e.*, to have the TOG represented in at least a decidable language and, for performance reasons, in a language of the lowest complexity possible. These conflicting goals for the TOG will be investigated in this subsection.

To assess decidability of the theory and complexity of the language required to represent the TOG, we first can make four observations with respect to the language constructors used in the formal characterisation of the TOG and assess it against complexity of the well-investigated DL languages.

1. The FOL that is used to characterise the TOG, is undecidable, *i.e.*, given a decision problem, there is no algorithm that always terminates with 'yes' or 'no' (refer to, *e.g.*, Hedman (2004) for a comprehensive explanation of Gödel's two Incompleteness Theorems).
2. RL is acyclic. Results obtained from complexity of DL languages has demonstrated that acyclicity is costly and the only DL language that has a constructor for it, $\mathcal{DLR}_\mu$ (Calvanese *et al.*, 1999), is ExpTime-complete for logical implication and satisfiability, like its base language $\mathcal{DLR}$ and sister languages $\mathcal{DLR}_{ifd}$ and $\mathcal{DLR}_{reg}$ (Calvanese *et al.*, 1998a, 2003).
3. GP is uniquely identified by the combination of its C and TG, which means it is, in ER terminology, a weak entity type, for which one needs the **id** constructor, which is available in $\mathcal{DLR}_{ifd}$ (Calvanese *et al.*, 2001a).
4. Other interesting constructors in the TOG are qualified number restrictions (RE^-, CP, and $conv$), a ternary relation ($conv$), transitivity of relations (RL, RE, and mereological parthood relations), and antisymmetry (Ground Mereology).

Most DL languages remain within the decidable fragment of FOL, but none is expressive enough to represent the full TOG. To see this, a comparison of the potentially most suitable DL languages (Keet and Rodríguez, 2007) is included in *Table 5.3*, which is based on their respective sources Berardi *et al.* (2005); Calvanese *et al.* (1998a, 1999, 2001a, 2003); Calvanese *et al.* (2005, 2006a,b); Cuenca Grau *et al.* (2006); Horrocks *et al.* (2006); Keet (2007a,b); McGuinness and van Harmelen (2004); OWL (2007). The languages are (i) OWL-Lite, OWL-DL, and OWL 1.1 ontology languages for the Semantic Web, (ii) $\mathcal{DLR}_{ifd}$, $\mathcal{DLR}_{reg}$, and $\mathcal{DLR}_\mu$ that are expressive DL languages roughly corresponding to conceptual data modelling languages such as ER, UML, and ORM, and (iii) three lean DLs of the $DL\text{-}Lite$ family that are tightly coupled to relational databases. From this comparison and items 2-4 above, we can observe the following:

- ⋆ There is no *known* guaranteed to be decidable DL fragment of FOL that is expressive enough to represent the TOG.
- ⋆ None of the DL languages has a constructor for antisymmetry. Antisymmetry is needed for $part\_of$, which is used in the definitions of $overlap$ and $overcross$ that, in turn, are used for relating levels across perspectives; hence, making $ppart\_of$ as primitive in the TOG instead does not avoid the problem (nor does extending the mereological theory included in the TOG to GEM where $overlap$ can be taken as primitive Varzi (2004a)).
- ⋆ A "$\mathcal{DLR}_{\mu ifd}$" suffices to represent most of the TOG, except for antisymmetry and RP's symmetry.
- ⋆ If one chooses OWL 1.1, then acyclicity, external uniqueness, and antisymmetry have to be removed from the TOG and the ternary relation reified.

Let us elaborate on a possible "$\mathcal{DLR}_{\mu ifd}$". Acyclicity, thus also transitivity and irreflexivity, is accomplished in $\mathcal{DLR}_\mu$ by inclusion in the language of the least fixpoint constructor (Calvanese *et al.*, 1999), whereas **id** and **fd** constraints involve the use of a generalised ABox and Skolem constants (Calvanese *et al.*, 2001a); $\mathcal{DLR}_{ifd}$ and $\mathcal{DLR}_\mu$ syntax and semantics are summarised in *Appendix C*. Considering the constructors and semantics of the two languages, one directly can observe the following: $\mathcal{DLR}_\mu$ requires an extended interpretation function with valuation ρ on $\mathcal{I}$, hence, adding the least fixpoint to $\mathcal{DLR}_{ifd}$ would require the same modification of the interpretation function. Conversely, adding **id** and **fd** to any $\mathcal{DLR}$ does not affect the interpretation function itself. Moreover, the interpretation for **id** and **fd** are just constraints and, unlike the least fixpoint, *they do not add any concept or role constructors to the syntax*. Therefore, it looks promising

Table 5.3: DL-based ontology and conceptual data modelling languages with their differentia; terms in braces are regularly considered as synonyms; indirect or implied support ($\pm$).

Language $\Rightarrow$	OWL			$DL\text{-}Lite$			$\mathcal{DLR}$		
Feature $\Downarrow$	**Lite**	**DL**	**v1.1**	$\mathcal{F}$	$\mathcal{R}$	$\mathcal{A}$	*ifd*	μ	*reg*
Role hierarchy (taxonomy of relations)	+	+	+	-	+	+	+	+	+
N-ary roles (where $n \geq 2$, ternary, quaternary relation etc.)	-	-	-	$\pm$	$\pm$	$\pm$	+	+	+
Role concatenation (limited role composition)	-	-	+	-	-	-	-	-	+
Role acyclicity (least fixpoint construct)	-	-	-	-	-	-	-	+	-
Symmetry	+	+	+	-	+	+	-	-	-
Role values (role attribute values, like strings and integers)	-	-	-	-	-	+	-	-	-
Qualified number restrictions	-	-	+	-	-	-	+	+	+
One-of, enumerated classes	-	+	+	-	-	-	-	-	-
Functional dependency (or UML method)	+	+	+	+	-	+	+	-	+
Covering constraint over concepts (total/complete covering)	-	+	+	-	-	-	+	+	+
Complement of concepts (disjointness of classes)	-	+	+	+	+	+	+	+	+
Complement of roles (disjointness of roles)	-	-	+	+	+	+	+	+	+
Concept identification (primary key with > attribute)	-	-	-	-	-	-	+	-	-
Range typing (define concept of the 2nd participant in role)	-	+	+	-	+	+	+	+	+
Reflexivity	-	-	+	-	-	-	-	+	+
Antisymmetry	-	-	-	-	-	-	-	-	-
Transitivity	+	+	+	-	-	-	-	+	+
Asymmetry	+	+	+	-	+	+	-	$\pm$	-
Irreflexivity	-	-	+	-	-	-	-	+	-

to add them to $\mathcal{DLR}_\mu$. However, considering the reasoning techniques, the following can be observed. For reasoning with $\mathcal{DLR}_\mu$, the language is first encoded into $\mu\mathcal{ALCQI}$ and then $\mu\mathcal{ALCI}_f$ to use techniques based on two-way alternating automata on infinite trees for concept satisfiability. $\mu\mathcal{ALCQI}$ has binary relations only (transformation from an n-ary where $n > 2$ through the usual reification), but acyclicity for n-aries where $n > 2$ is rare anyway[20]. Reasoning in the presence of a non-empty ABox remains an open problem for $\mathcal{DLR}_\mu$ (Calvanese *et al.*, 1999). Reasoning on $\mathcal{DLR}_{\mathit{ifd}}$ is for TBoxes the same as for $\mathcal{DLR}$ TBoxes; put differently, the identification and functional dependency assertions are dealt with in the ABox. More precisely, $\mathcal{DLR}_{\mathit{ifd}}$ uses a generalised ABox (Calvanese *et al.*, 2001a): take an alphabet of new symbols (called Skolem constants[21]) and the interpretation function extended for individuals (see *Appendix C*). Thus, we have sk-constants x, y, $x_1, ..., x_n$ and R an n-ary relation, ABox assertions of the type $C(x)$, $R(x_1, ..., x_n)$, $x \neq y$, and $x = y$, and an extended interpretation function to assign to each sk-constant x an individual $x^\mathcal{I} \in \Delta^\mathcal{I}$, then an interpretation $\mathcal{I}$ satisfies $C(x)$ if $x^\mathcal{I} \in C^\mathcal{I}$, $R(x_1, ..., x_n)$ if $(x_1^\mathcal{I}, ..., x_n^\mathcal{I}) \in R^\mathcal{I}$, $x \neq y$ if $x^\mathcal{I} \neq y^\mathcal{I}$, and $x = y$ if $x^\mathcal{I} = y^\mathcal{I}$. From the semantics of **id** and **fd** in the previous section, it is clear that logical implication in $\mathcal{DLR}_{\mathit{ifd}}$ can be reduced to knowledge base satisfiability (theorem 2 in Calvanese *et al.*, 2001a). Of these results, three points have to be highlighted. First, we prove the following theorem.

[20]ORM allows ring constraints (DL role properties) on two of the n ORM-roles, see *Figure 5.2-B*, which is an issue for an ORM to $\mathcal{DLR}_{\mu\mathit{ifd}}$ mapping (*Lemma 5.1*), but not for mapping the TOG into $\mathcal{DLR}_{\mu\mathit{ifd}}$.

[21]a sk-constant denotes an individual in an interpretation so that different sk-constants may denote the same individual.

THEOREM 5.1 (Type-level combination $\mathcal{DLR}_{\mu ifd}$). *Provided the theory remains at the type-level (TBox Σ), $\mathcal{DLR}_{ifd}$ and $\mathcal{DLR}_{\mu}$ can be combined into $\mathcal{DLR}_{\mu ifd}$ and remain ExpTime-complete for logical implication and concept satisfiability.*

Proof. First, by theorem 2 in (Calvanese *et al.*, 2001a), we have that logical implication (w.r.t. the **id** and **fd** assertions) for $\mathcal{DLR}_{ifd}$ can be reduced to knowledge base satisfiability. Then, given a set $\mathcal{F}$ of identification and functional dependency assertions and a $\mathcal{DLR}_{ifd}$ TBox $\mathcal{T}$, then $\mathcal{T} \cup \mathcal{F}$ is satisfiable iff $\mathcal{T}$ is, and a $\mathcal{DLR}_{ifd}$ TBox is the same as a $\mathcal{DLR}$ TBox (Calvanese *et al.*, 2001a). Thus, for a conceptual data model or ontology sensu Σ in *Definition C.1*, we have $\Sigma_{\mathcal{DLR}_{ifd}} \equiv \Sigma_{\mathcal{DLR}}$ and logical implication of inclusion assertions can be verified without considering identification and functional dependency. We also have $\Sigma_{\mathcal{DLR}} \subseteq \Sigma_{\mathcal{DLR}_{\mu}}$, therefore the complexity results of $\mathcal{DLR}_{\mu}$ still hold. □

Second, if we also have an ABox, *i.e.*, a knowledge base sensu $\mathcal{K}$ in *Definition C.2*, then a saturation of $\mathcal{K}$ as an ABox A_s is needed, where we have to use the sk-constants. To summarise, the resultant theorem is reproduced here, where the notation of the definition of knowledge base is adjusted to reflect the notation in *Definition C.2*; see Calvanese *et al.* (2001a) for details.

THEOREM 5.2 (theorem 3 in (Calvanese *et al.*, 2001a)). *A $\mathcal{DLR}_{ifd}$ knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{R}, \mathcal{A}, \mathcal{F})$ is satisfiable if and only if there exists a saturation $\mathcal{A}_s$ of $\mathcal{K}$ that does not violate the identification and functional dependency assertions in $\mathcal{K}$ and such that the knowledge base $(\mathcal{T}, \mathcal{R}, \mathcal{A}, \mathcal{A}_s)$ is satisfiable.*

Third, unary functional dependencies in $\mathcal{DLR}_{ifd}$ is undecidable, which was proven with the tiling problem (Calvanese *et al.*, 2001a). This leaves to solve

(i) How does $\mathcal{DLR}_{\mu}$ behave in the presence of a generalised ABox?
(ii) Does this interfere with **id** and **fd**?

If either one has as solution an undecidable language or conflict, this means for the TOG rendered in a DL language and in the presence of an ABox that, at least, *either GP* will not be identified by its C and TG but will have to be identified through a simple reference scheme *or RL* cannot be acyclic and RE and RL cannot be transitive. It will be shown that *under two additional constraints* in the presence of an ABox, $\mathcal{DLR}_{\mu ifd}$ remains decidable. We assess **fd**s and acyclic roles first (*Lemma 5.1*) and **id**s and acyclic roles afterward (*Lemma 5.2*).

LEMMA 5.1. *Least (greatest) fixpoint $\mu X.C$ ($\nu X.C$) together with n-ary roles $R_a \in \mathcal{R}$ where $n > 2$ leads to undecidability in $\mathcal{DLR}_{\mu ifd}$.*

Proof. Acyclic roles are inherently binary, and the reasoning procedure and complexity results in (Calvanese *et al.*, 1999) is only for binary relations with least/greatest fixpoint, therefore a role $R_b \in \mathcal{R}$ that is used with $\mu X.C$ ($\nu X.C$) should remain binary, and binary roles do not have unary **fd**s. However, n-aries (where $n > 2$) with acyclicity are possible in some knowledge representation languages; *e.g.*, an ORM model with acyclic ring constraint in a ternary relation (see *Figure 5.2-A*). In this way, we easily can reintroduce unary **fd**s that are have been shown to be undecidable in $\mathcal{DLR}_{ifd}$ (Calvanese *et al.*, 2001a). Let R_{a3} be a ternary relation with acyclicity over role elements r_1 and r_2 to store a *single*-inheritance hierarchy (tree) H_s over instances in $\mathcal{A}$ where objects for r_2, are R_{a3}-successors of objects ($C(c_i)$ with $1 \leq i \leq m$ and m a non-negative integer) for r_1, and each concept associated with r_1 agrees on r_3. For instance, each node is identified by a number and also has a name (a string); a sample population is included in *Table 5.4*. Thus, we have (**fd** R_{a3} $r_1 \rightarrow r_3$). We also achieve undecidability with a ternary R_{a3} and a *multiple*-inheritance hierarchy H_m (*Figure 5.2-B*): assume (**fd** R_{a4} $r_1 \rightarrow r_3$) does not hold, i.e., two tuples of R_{a3} do not agree on r_1 and r_3, we still can construct a unary **fd** because values in r_2 are not unique and then (**fd** R_{a3} $r_2 \rightarrow r_3$) is possible. □

Table 5.4: R_a in H_s.

r1	r2	r3
1	2	aa
1	3	aa
2	4	bb
2	5	bb
3	6	cc
3	7	cc
...	...	...
m	n	zz
m	n+1	zz

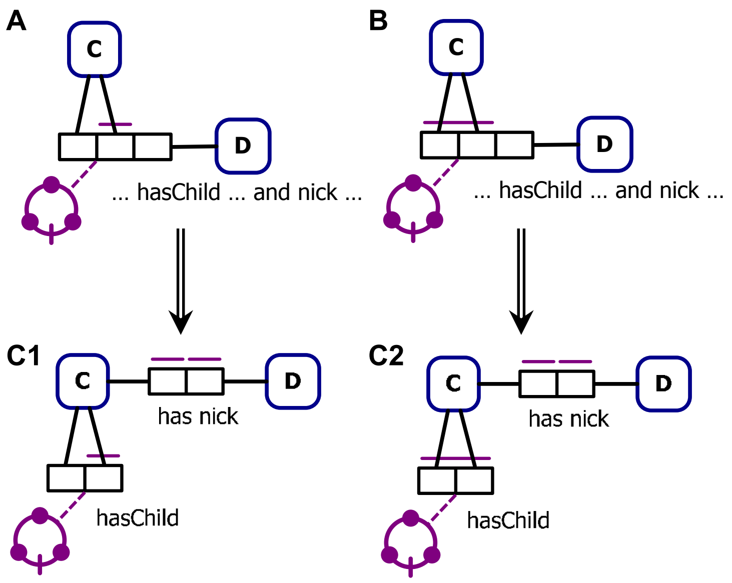


Figure 5.2: ORM2 examples with acyclic relations; numbering of the ORM-roles/DL-role elements from left to right. A: single-inheritance tree with an attribute (invalid in ORM but valid in a UML class diagram); B: multiple-inheritance tree with an attribute; C1 & C2: the satisfiable and valid versions of A and B, respectively.

We can regain decidability with H_s if and only if C has at least 2 attributes and a non-unary **fd** is asserted, and with H_m when the child has at least 2 attributes too. Although possible, these additional constraints result in excessive duplication and such conceptual data models and its logical version should be normalised (Abiteboul *et al.*, 1995; Halpin, 2001) by separating the acyclicity among the two DL-role elements into a binary R_b, such as in *Figure 5.2-C1* and *C2*. This constraints is particularly relevant for so-called triple stores where a hierarchy is represented as tuples $\langle term, relation, term\rangle$, *e.g.*, $\langle mitochondrion, ppart\_of, cell\rangle$ and so forth. This corresponds to a ternary relation with an unary **fd** in the conceptual data model regardless if one has a H_s or H_m. This can be prevented by using a conceptual model of the ontology language (Tziviskou and Keet, 2007) as opposed to a one-off conceptual data model for ontologies-stored-in-a-database (such as the GO).

The next lemma and proof deals with effects of the interplay between the fixpoint and **id**.

Lemma 5.2. *A $\mathcal{DLR}_{\mu ifd}$ concept C is satisfiable if a binary acyclic role $R_b \in \mathcal{R}$ in $\mathcal{DLR}_{\mu ifd}$ is* not *used in an identification assertion* **id**.

<u>*Proof.*</u> (*Sketch*) Given that **id** checking relies on the use of sk-constants to construct a saturation, one can construct cycles where two different sk-constants denote the same individual, violating

acyclicity and leading to a contradiction in a least/greatest fixpoint assertion in the presence of an ABox. □

Given the two interferences between least/greatest fixpoint and identification and functional dependencies in an ABox, we come to the following result for satisfiability and logical implication for a $\mathcal{DLR}_{\mu ifd}$ knowledge base $\mathcal{K}$.

THEOREM 5.3. *Given a knowledge base* $\mathcal{K} = (\mathcal{T}, \mathcal{R}, \mathcal{A}, \mathcal{F})$ *of* $\mathcal{DLR}_{\mu ifd}$*, where* $\mathcal{DLR}_{\mu ifd} = (\mathcal{DLR}_{ifd}, \mathcal{DLR}_{\mu})$*, satisfiability and logical implication* $\mathcal{DLR}_{\mu ifd}$ *is ExpTime-complete, provided the following conditions are met:*

- *Least (greatest) fixpoint* $\mu X.C$ $(\nu X.C)$ *is used only with binary roles* $R_b \in \mathcal{R}$*;*
- R_b *does not occur in any identification assertion,* i.e.*, for (***id** $C[i_1]R_1, ..., [i_h]R_h$*) then* $R_b \neq R_1, ..., R_b \neq R_h$*.*

Proof. In absence of $\mathcal{A}$, we keep complexity results (*Theorem 5.1*). In the presence of a non-empty $\mathcal{A}$, undecidability for n-ary roles, where $n > 2$, together with least (greatest) fixpoint was proven in *Lemma 5.1*; the first condition prevents this case. Also in the presence of a non-empty $\mathcal{A}$, participation of R_b in an **id** assertion can lead to unsatisfiability (*Lemma 5.2*); the second condition prevents this case. Thus, provided the two conditions hold, we keep the complexity results of $\mathcal{DLR}_{ifd}$ and $\mathcal{DLR}_{\mu}$ Calvanese *et al.* (1999, 2001a) when combined into $\mathcal{DLR}_{\mu ifd}$, which is ExpTime-complete. □

Observe that, given that the **id** and **fd** assertions do not change the semantic rules, the semantic rules for $\mathcal{DLR}_{\mu ifd}$ are those of $\mathcal{DLR}_{\mu}$ as provided in *Table C.3* and concept and roles can be build according to the $\mathcal{DLR}_{\mu}$ syntax (which is a superset of $\mathcal{DLR}_{ifd}$).

Assessing the TOG where the fixpoint construct is needed, the following can be observed.

⋆ RL is an acyclic binary relation and not involved in an **id** assertion;
⋆ RE is a transitive binary relation and not used in an **id** assertion;
⋆ RP is irreflexive (and symmetric, which cannot be done in $\mathcal{DLR}_{\mu ifd}$) binary relation and does not participate in an **id** assertion.

Thus, we can map the TOG into a $\mathcal{DLR}_{\mu ifd}$ representation, except for RP's symmetry and $part\_of$'s antisymmetry. Further, we could take $ppart\_of$ as primitive and remove overcrossing levels and perspectives because we have RP anyway, and encode RP as two relations to evade its symmetry.

Thus, the *lower bound* complexity of the language required to represent the TOG is in ExpTime, which is also corroborated by the experimental data in *Figure 3.8* and *Appendix B.2*. These mildly encouraging results do not mean that the full TOG is not implementable computationally, just that one will observe slow performance that may be unacceptable for the user. An alternative route is to simplify the TOG for different implementation purposes, such as for OWL 1.1 (Horrocks *et al.*, 2006; OWL, 2007) and use the Pellet automated reasoner, but then we have the $\mathcal{SROIQ}$ DL language that is NExpTime-hard in concept satisfiability. Further simplification steps can be taken to implement it in relational or deductive databases that support recursive queries[22]. Alternatively, a simpler version of the TOG could be added to or integrated with WSML (De Bruijn and Heymans, 2007; Tziviskou and Keet, 2007). There are more such transformations—simplifications—possible for the TOG to take it toward implementation, which is left for future work.

[22]SQL:1999 and SQL:2003 both support recursive queries, which are more or less implemented in currently available RDBMSs. Implementation feasibility of granularity with respect to recursive queries in relational databases are assessed and discussed in (Keet, 2006b), concluding that there is both a gap in theory and software tools and their usages to achieve a smooth realisation for both the tested relational databases systems—GO and FMA—as well as alternative possible implementations such as XML files with XQuery and RDF with SPARQL.

5.5.1.2 The data-framework interface

Unlike most other contributions on granularity, the TOG puts constraints on the data with respect to the structure of the contents but not on two desirable features listed in §3.1. These features are (A) the entities (/types) are disjoint and (B) the entities (/types) are not necessarily exhaustive. The first feature does not affect the static components of a granular framework and is from that viewpoint uninteresting. Where it can have a negative impact is the successive firing of abstraction functions, which require single-inheritance trees for best performance. Therefore, it is *highly recommendable* to satisfy desirable feature A, but not mandatory; that is, if one were to integrate the TOG with, *e.g.*, ORM2, feature A has to be included as a deontic constraint. Feature B is trivially met, because the theory adheres to the open world assumption, and so do two of the three focal application areas, knowledge bases and ontologies. On the other hand, one could enforce the exhaustiveness constraint when the TOG will be implemented in a database. For these differences, one cannot add the constraint on (non-) exhaustiveness explicitly without limiting applicability scenarios.

Feature C—provided an entity (/type) is not an orphan in the original data source and the subject domain is covered fully with granular perspectives, it must reside in at least one granular level—was addressed in Chapter 3 (see §3.10.1), which is less restrictive than optional axiom (2.12) discussed in §2.3.1) that makes the ontological commitment that the world is granular. However, one may want to loosen this weaker constraint C even further *under controlled implementation scenarios* where one would like to discover the 'violating' entities (/types). Put differently, *not* having constraint C and (2.12) *a priori* can give interesting avenues for further investigation, as described by Keet *et al.* (2007) in the survey of requirements for automated reasoning services for bio-ontologies.

5.5.1.3 More than one domain granularity framework

The TOG does not deal with the case where there is more than one domain granularity framework. This situation may arise during re-use of multiple legacy perspectives and levels and, among others, GIS, model-organism, and, spatio- and temporal- database integration. A full investigation is beyond the scope of the current research, and here we will only assess if, and if yes where, it may require an extension of the TOG.

Main scenarios. Alike scenarios for linking or integrating multiple data sources of the same, similar, or complementary domains (Keet, 2004a), this can occur with multiple domain granularity frameworks as well. More precisely, let d^{f_1} and d^{f_2} be two particular domain granularity frameworks and, by transitivity, the perspectives and levels they contain, then there are four main scenarios: (i) $d^{f_1} \equiv d^{f_2}$, (ii) $d^{f_1} \cap d^{f_2} = \emptyset$, (iii) $d^{f_1} \subset d^{f_2}$, or (iv) d^{f_1} partially overlaps with d^{f_2}. The main sub-problem is having the same perspective(s) in different domains but they differ in the levels—*i.e.*, $RE^{-}(d^{f_1}, gp^1_1)$, $RE^{-}(d^{f_2}, gp^2_1)$, $gp^1_1 = gp^2_1$ because their TG and C are the same, but $\mathcal{L}^1 \neq \mathcal{L}^2$ for gp_1. This sub-problem will be discussed in the next paragraph.

Delegating compatibility issues of levels in a perspective to a sub-problem, then scenario iii ($d^{f_1} \subset d^{f_2}$) reduces to difference in the amount of perspectives, meaning that we have $gp^1_i = gp^2_i$ with $i \geq 1$, but also that $\mathcal{P}^1 \subset \mathcal{P}^2$ and thus gp^2_j, with $j > 1$ and $i \neq j$. Hence, we only have to combine the perspectives, which, given that D^f is typed as a CN, then results in a new d^{f_3} for the combined d^{f_1} and d^{f_2}, and $\mathcal{P}^2 \equiv \mathcal{P}^3$. It is similar for the partial overlap of d^{f_1} with d^{f_2} (scenario iv), but then we have $\mathcal{P}^1 \subset \mathcal{P}^3$ and $\mathcal{P}^2 \subset \mathcal{P}^3$. This is again similar for $d^{f_1} \cap d^{f_2} = \emptyset$, but with one additional option: it may be that disjoint perspectives can be concatenated to one larger perspective, with or without adding a connecting level; hence, either $gp^1_1 \cup gp^2_1 \equiv gp^3_1$ or $gp^1_1 \cup gp^2_1$ + ≥ 1 level $\equiv gp^3_1$. Concerning the sets of perspectives, then $\mathcal{P}^1 \cup \mathcal{P}^2 \subseteq \mathcal{P}^3$. Hence, none of these scenarios pose any problem for the TOG.

Differences in amount of levels in perspectives. Regarding $RE^{-}(d^{f_1}, gp_1^1)$, $RE^{-}(d^{f_2}, gp_1^2)$, $gp_1^1 = gp_1^2$, and $\mathcal{L}^1 \neq \mathcal{L}^2$ for gp_1, one can recollect as example the perspectives and their levels for human structural anatomy in §5.3.1. We have to assess options to deal adequately with such differences in levels of the same perspectives from different domains. Before outlining the options, there are two preliminaries. First, the outcome depends on the requirements for the granulated information system(s), analogous to schema integration solutions: if it has to become one system or if the original sources should remain unaltered. The current TOG not only suffices, but can be part of the methodology for system integration, for it can provide the single comprehensive granular perspective that contains all levels of the relevant perspective from the other domain granularity frameworks; by analogy, it gives a global view as with a LAV integration methodology[23]. Second, to be more precise, let d^{f_1} and d^{f_2} be the source domains and d^{f_3} the comprehensive one, $gp_1^1 = gp_1^2 = gp_1^3$, *i.e.*, they agree on C and TG, and for the set of levels in each of these perspectives gp_1, $\mathcal{L}^1 \subseteq \mathcal{L}^3$, $\mathcal{L}^2 \subseteq \mathcal{L}^3$, and $\mathcal{L}^1 \neq \mathcal{L}^2$ hold, where $\mathcal{L}^1 \cap \mathcal{L}^2$ may be empty (this can be scaled up to an arbitrary amount of domains and perspectives, omitted for brevity). The three principal options are then as follows.

i. Adding both gp_1^1 and gp_1^2 to d^{f_3} violates the current TOG constraints on unique perspectives that are identified by their TG and C (*Corollary 3.2* and *Lemma 3.5*). Alternatively, one could identify each GP by its TG, C, *and* amount of levels, but then one cannot distinguish between the case where gp_1^1 and gp_1^2 have the same amount of levels but covering different granularities. Thus, it would require an additional mechanism for specification of "sameness" of levels in those perspectives that agree on their TG and C values and of "semantic disjointess" that the hierarchy of levels in gp_1^1 and gp_1^2 are distinct.

ii. Take the comprehensive gp_1^3, propagate all its levels that back into gp_1^1 and gp_1^2, and leave empty the levels that are not used by the applications that use d^{f_1} and d^{f_2}. This requires a resolve for either the granulation relations—because there is a difference between entities being directly connected to its parent (child) in the coarser- (finer-) grained level and related by transitivity, and for $member\_of$ and $participates\_in$ for their perspectives have a maximum of two levels—and the abstraction functions, because they cover abstraction of contents one level at a time only.

iii. Implement query rewriting alike a LAV for schema integration but then for the granularity framework. This has no effect on the TOG, because it is an implementation solution to map a comprehensive gp_1^3 to one or more perspectives with less levels (gp_1^1 and gp_1^2).

At this point of the investigation of the theoretical aspects of granularity and implementation experiments, it is unclear whether differences in amount of levels in a perspective due to reuse of legacy systems and data- information- or knowledge integration of such systems aided by a granularity framework will be an important enough usage to merit amending the TOG along the lines of option i or ii. If it is, then adding abstraction functions that skip levels is least disruptive compared to the other changes. In any case, option iii is always feasible.

5.5.2 Implementation considerations

In this section, we look ahead toward implementing the TOG and therefore can be read for a large part as explorative with suggestions for future work. Questions that have to be answered are, among others: what is the optimal strategy for adding or integrating granularity? How can

[23]Database integration theory uses either the local-as-view (LAV) or global-as-view (GAV) principle, where LAV means that each source table is defined as a view of the global model and GAV has the global schema expressed in terms of the data sources. LAV is favourable when a shared global view, such as an ontology, is available, and they tend to be more flexible when extending systems than GAV, but querying in GAV architectures has been found to be easier than LAV systems (Lenzerini, 2002). Three existing types of architectures for database integration are centralised DWHs, Federated Databases (FD), and decentralised Piazza peer Data Management System (PDMS). Some experiments and comparisons between FD and DWH are discussed by Shoop *et al.* (2001); Poggi and Ruzzi (2004); Kerschberg (2001), but no conclusive answer on the best architecture for the methodology could be established, also because it depends on characteristics of the state of the data sources.

verification of compliance of a d^f with the TOG constraints be done? Are there limitations of the data source that prevent a full implementation of the TOG? If so, what is the 'minimum' set of TOG features that must be implemented, does the data source support this, and if not even that, what new technology is required to meet the TOG features? What, if any, is the trade-off between support for the static features of representing an applied domain granularity framework and, or versus, support for the functions to query a granularity framework?

The main issue that will receive attention in this section is how to ensure that a particular domain granularity framework does not violate the TOG constraints. Theoretically, this is straightforward by adhering to a model-theoretic semantics or using a second-order meta-layer for the TOG components (sometimes called 'punning') and then their instances as types in a (first order or less expressive) logical theory. Practically, this is much less straightforward, depending on the implementation scenario. This will be discussed in the next subsection. Afterward, we look briefly at two alternative scenarios, which are adding granularity constructs to a conceptual data modelling language and using a domain granularity framework for ontology linking analogous to using a LAV global schema for database integration.

5.5.2.1 Using the TOG to develop an applied domain granularity framework $\mathcal{DG}$

This concerns experiments 1 and 2 described in Chapter 1—*i.e.*, $ex^1(D) \rightarrow d_i^f$ and $ex^2(d_i^s \cup d_i^f) \rightarrow \mathcal{DG}$—to use the TOG to constrain d_i with its perspectives and levels. ex^1 comprises two components: a procedure to develop a d^f and one where the d^f is checked against the TOG that it does not violate any constraint. A methodology for d_i^f- and $\mathcal{DG}$-development consists of the following protocol:

1. Demarcate subject domain—which aspects to include and which not.
2. Define signature, containing the use of the components of the granularity framework and the operations one can use to manipulate the data.
3. Identify granular perspectives for the chosen subject domain.
4. Identify granular levels and assign the levels to their appropriate perspective.
5. Load this domain granularity framework with data (or: assign a level to the entities).

This methodology was tested successfully with the domain of infectious diseases (Keet, 2006c). The framework was examined *manually* on compliance with the TOG and did not violate any TOG constraint. However, it will be more reliable to develop *computational* support for automated consistency checking. To realise this, multiple implementation decisions have to be taken, such as: which logic (see also §5.5.1), reasoner, options for graphical support, maintenance strategy, and so forth. Depending on these decisions, there are several ways to check that a domain granularity framework indeed satisfies the TOG constrains.

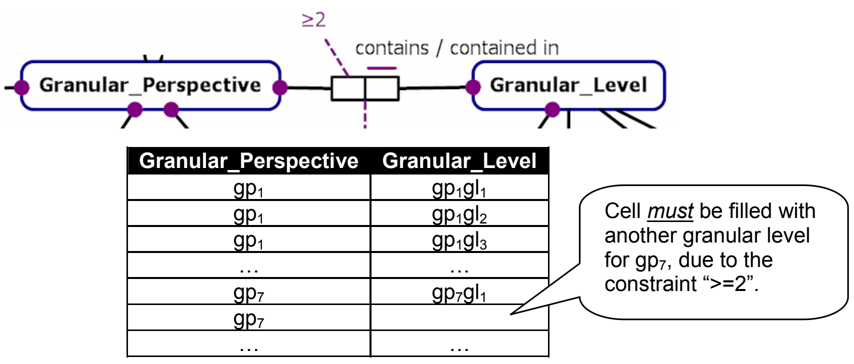
⋆ Database implementation, including ontologies stored in a database: the usual integrity constraint checking can be used (see also *Example 5.1*) because a d^f then amounts to a model of the TOG as logical theory in the formal conceptual data model.

⋆ Ontology or conceptual data model (represented and stored at the type-level): TOG as a theory in second order logic, 'punning', UML meta-constructs and stereotypes or a similar approach for other knowledge representation languages to have the TOG components instantiating d^f at the type-level of the ontology or conceptual data model.

⋆ Knowledge base: given that a knowledge base can contain both type-level information (TBox in a DL knowledge base) and instance-level data (ABox), one can choose to either use one of the two previous options or use an additional transformation step whereby the instances in a d^f are nominalised into a singleton set and then used with the rest of the type-level $\mathcal{DS}$ so that they are treated as part of the logical theory at the type-level that then can be fed to an automated reasoner.

For the two type-level implementation scenarios, one may want to use the consistency checking alike when combining an automated reasoner with an conceptual data modelling or ontology development tool (Fillottrani *et al.*, 2006; Franconi *et al.*, 2000; iCOM, 2002) to check consistency of the logical theory and to apply OntoClean (Guarino and Welty, 2004) constraints for a well-formed ontology, but then to apply the TOG for a well-formed granularity framework. After illustrating in *Example 5.1* the general idea of TOG consistency checking, we look at the issues to resolve to get it working for ontologies.

Example 5.1. Consistency checking of a d_i^f is based on testing the domain granularity framework against the definitions and constraints provided in the TOG. For instance, let d_1 = Infectious diseases and gp_7 = Pathological structure (see *Appendix A*). If only one level were contained in gp_7, then constraint (T.7) would have been violated, because each perspective must contain at least 2 levels; hence this d_1 does not satisfy the TOG and is an invalid domain granularity framework. *Figure 5.3* shows this if the TOG would have been implemented in a database.
Let us take gp_2 = Site of entry and gp_3 = Site of effect, which use the same data source; however, because their user-semantics is different this does not violate constraint (A.110). What would violate (A.110) is to include, *e.g.*, two species taxonomies or two infectious diseases classifications: if the perspectives are the same—*i.e.*, same C and TG—then one is redundant. Alternatively, they might be competing perspectives, but then the data is inconsistent and any reasoner will detect the conflicts, return inconsistent query answers, or needs permanent manual intervention. For instance, it may return two or more types of diseases or several types of causative organisms where there is only one in reality. If multiple causative organisms are included in the query answer, then at least one inconsistency in the theory appears to exist, therefore requires further investigation to ascertain either the real causative organism or why there is more than one. However, this should not remain in the system. ♢


≥2
contains / contained in
Granular_Perspective
Granular_Level
Cell *must* be filled with another granular level for gp7, due to the constraint ">=2".

Granular_Perspective	Granular_Level
gp1	gp1gl1
gp1	gp1gl2
gp1	gp1gl3
…	…
gp7	gp7gl1
gp7	
…	…

Figure 5.3: Section of *Figure 3.1*, which now also serves as a conceptual data model for database development and management. The table corresponding to the relation must have at least two tuples where the value of the `Granular_Perspective` column equals $\texttt{gp}_7$, and each granular level must be unique, preventing a user to include $\texttt{gp}_7\texttt{gl}_1$ twice.

From a user point of view, the inconsistencies mentioned in *Example 5.1* may be welcome not only for a database scenario but also for knowledge bases and ontologies, because it may indicate a new research topic to elucidate the correct answer. Therefore, it could be a nice feature to toggle on/off certain TOG constraints in an implementation of the automated consistency checking, so that it aids the user in understanding both the software system's features and the granulated information stored in the system.

Recollecting the complexity section above (§5.5.1.1), it is important *how* the TOG can be—and has to be—mapped onto, and possibly be incorporated in, a (decidable) knowledge representation or conceptual data modelling language. The TOG basics have been mapped already through the graphical rendering with ORM2 in *Figure 3.1*[24], but now we have moved to type-level representation and reasoning. Before conceiving new reasoning services for granularity to ensure a domain granularity framework complies with the TOG, the extant automated reasoners, such as Pellet, FaCT++ and Racer, will have to implement the *RBox reasoning service* to reason over DL-role hierarchies. The RBox reasoning service (*Definition 4.8*) is required to check the consistency, *i.e.*, correct implementation, of the granulation relations and that the theory distinguishes between RE and RL on their relata and not only syntactic subsumption fo the DL-roles. Given that several hurdles for successful implementation of the representation and automated consistency checking remain, it may be clear why ex^1 and ex^2 have been carried out mostly manually. To summarise, this is partly due to the modelling constructs used in the TOG and partly due to a chosen scenario—off-setting databases (including ontologies stored in databases) versus type-level ontologies—but also due to practical engineering limitations where the theory exist, but the solution has not yet been implemented—be it due to novelty, as with the RBox Compatibility service, or sub-optimal software development (see Keet, 2006b, for a discussion). Therefore, $task^5$, which combines $task^4$ and ex^2 into one, is satisfied only to a limited extent.

Combining the TOG-compliant d^f with the $\mathcal{DS}$ may be done through may paths, some of which will be highlighted from the list of options and illustrated in *Example 5.2* afterward.

1. Databases: (i) Database views where the view corresponds to a level gl_i and the tuples the contents E, (ii) Database star-like schema where the 'fact table' has one column for the entities and one column for each perspective, with ii-a) granularity added to an existing database or ii-b) as separate database and data loaded into this, and (iii) Conceptual data models (see next subsection);
2. Knowledge bases: (i) Add to TBox and use option ii or iii in item 3, (ii) Add to ABox, (iii) Add as rules;
3. Ontologies: (i) Add to ontology-stored-in-a-database and use one of the three options in item 1, (ii) Integrate with a formal ontology, and (iii) Use the framework to link multiple ontologies (see next subsection).

The following example demonstrates d^f to $\mathcal{DS}$ linking for databases and the TBox and ABox of a knowledge base, but for simplified versions of a d^f only.

Example 5.2. Let us first introduce a basic algorithm (*Algorithm 6*) to load a section of a taxonomy into its appropriate level, such as FMA cell types into the Cell-level, with a modified $assignGL$ function that has a nested while-loop that uses a recursive query, *i.e.*, $assignGLM : d^s \mapsto \mathcal{L}$. Looking at relational databases, one can avail of a recursive query to implement the loop, which is defined in SQL:1999 by using common table expressions as temporary views:

```
WITH [ RECURSIVE ] <query_alias_name> [ ( <column_list> ) ]
   AS ( <select_query> )
   <query_using_query_alias_name>
```

Alternatively, languages such as STRUQL could be used for recursive query support or deductive databases (Abiteboul *et al.*, 1995; Borgida *et al.*, 2002). The level, x in *Algorithm 6*, could stand for a (materialised) view, even though it is an impoverished version of a full instantiation of GL.

Another, much less efficient, option is not to integrate granularity with the database, but to add it as an extension. For instance, one can add a 'star-like' structure, as depicted in *Figure 5.4*. It has a *loose* mapping between TOG components:

$D^f \rightsquigarrow$ *star-like model*

$\mathcal{DG} \rightsquigarrow$ *large central table* (*the* '*subject area*')

[24]ORM has a FOL underpinning (Halpin, 1989; Hofstede and Proper, 1998), transformations to ER and UML (Halpin, 2001) and relational database schemas in, *e.g.*, Microsoft Visio and NORMA. Another option is to *add* constructors to a conceptual data modelling language for representing granularity (see §5.5.2.2).

$gp_i \rightsquigarrow$ *unary dimension table*
$gp_i gl_i \rightsquigarrow$ *tuple in dimension table*

The mapping is loosely, because not all constraints of the TOG can be mapped. Although the empty cells in the table are a waste, the overall system is clearer than the views-based definitions and entity allocations.

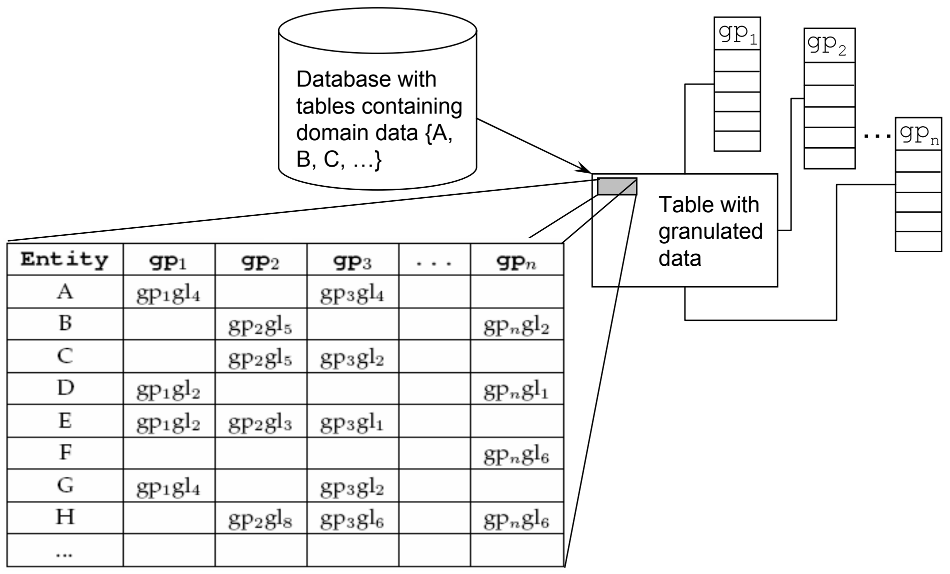
Last, we briefly look at DL knowledge bases (see also *Appendix C* and Baader *et al.* (2003)). For the current purpose, any language that has qualified number restrictions will do, such as $\mathcal{SROIQ}$ and the $\mathcal{DRL}$-family. Linking domain granularity can be applied to both the TBox and the ABox, which has been illustrated before (Keet, 2006c). Briefly, one adds TOG constrains to the TBox, like for RE^- between GP and GL: $\texttt{GP} \sqsubseteq \geq 2\, \texttt{RE}^- . \texttt{GL}$. This enables us to represent a domain granularity framework in ABox statements with at least two levels ($\texttt{gp}_1\texttt{gl}_1 : \texttt{GL}$ and $\texttt{gp}_1\texttt{gl}_2 : \texttt{GL}$) and to ensure they are contained in the right perspective with $\langle \texttt{gp}_1, \texttt{gp}_1\texttt{gl}_1 \rangle : \texttt{RE}^-$ and $\langle \texttt{gp}_1, \texttt{gp}_1\texttt{gl}_2 \rangle : \texttt{RE}^-$. Second, loading data into the levels or applying levels to the data can be done manually with, *e.g.*, $\texttt{AirBorne} \sqsubseteq \exists\, \texttt{in\_level.gp}_1\texttt{gl}_1$ and $\texttt{PersonToPerson} \sqsubseteq \exists\, \texttt{in\_level.gp}_1\texttt{gl}_2$. Keet (2006c) does not demonstrate how concepts can be added *automatically* to their level(s), thus inferred from the added knowledge. *Algorithm 7* suggests how this may be done for a new concept NewC that is going to be added to the $\mathcal{DG}$ together with $\texttt{NewC} \sqcap \exists\, \texttt{involved\_in.D}$, with D already in the knowledge base and assigned to a particular level. (The algorithm can be amended with automatically testing for other relations than *involved_in* and *is_a*.) Alternatively, procedural extension as in CLASSIC may be able to do this; with **K** as operator (Baader and Nutt, 2003), then using $\mathbf{K}\texttt{NewC} \sqsubseteq \exists\, \texttt{in\_level.gp}_2\texttt{gl}_1$, which states that "those individuals that are known to be NewCs are of granularity level $\texttt{gp}_2\texttt{gl}_1$". Similar types of rules exist in other knowledge bases, but not specifically as part of any of the DLs. Other extensions may be worthwhile to examine on usability for automation of level assignment, which are object-oriented constructs (Möller, 1996), other applications, or with procedural extensions, or DL Programs (Grosof, 2003). In any case, considerable further engineering work will have to be carried out before realising a prototype (DL-)knowledge base with domain granularity. ♢

Algorithm 6 Loading a taxonomy into a level with *assignGLM*

```
1: $x \Leftarrow selectL(x)$
2: $y \Leftarrow selectE(y)$ «y is the 'root entity' for the section of the taxonomy that will be loaded into x»
procedure $assignGL(y, x)$
Require: $in\_level(y, x)$
3: $a \Leftarrow y$
4: $b \Leftarrow$ select a child entity of $a$
5: while $subsumes(a, b)$ do
6:    $assignGL(b, x)$
7:    $a \Leftarrow b$
8:    $b \Leftarrow$ select next entity using recursive query
9: end while
```

5.5.2.2 Alternative scenarios

In addition to the essential consistency checking that a particular d^f does not violate the TOG, there are many possibly usages for granular information systems that each have their own set of issues to resolve. In this section, we look at two such topics, which are prospects for enhancing conceptual data modelling languages to represent granularity and ontology linking.


Database with tables containing domain data {A, B, C, ...}
Table with granulated data
gp_1
gp_2
gp_n
...

Entity	gp_1	gp_2	gp_3	...	gp_n
A	gp_1gl_4		gp_3gl_4		
B		gp_2gl_5			gp_ngl_2
C		gp_2gl_5	gp_3gl_2		
D	gp_1gl_2				gp_ngl_1
E	gp_1gl_2	gp_2gl_3	gp_3gl_1		
F					gp_ngl_6
G	gp_1gl_4		gp_3gl_2		
H		gp_2gl_8	gp_3gl_6		gp_ngl_6
...					

Figure 5.4: Star-like schema for granularity added to a database with domain data.

Granularity-enhanced conceptual data modelling. A major advantage of adding granularity constructs to a conceptual data modelling language is that then the granularity defined at the conceptual layer can propagate automatically to granulation of the data in the database without having to specify it at the physical database layer. This topic received some attention in the related research sections on data warehousing (§5.2.2) with MultiDimER and YAM^2 and conceptual data modelling for GIS applications (§5.3.2) with Granular GeoGraph and MADS. The former two are hierarchy-focussed whereas the latter two are entity type-focussed and, if combined, they still cover TOG components only partially. Ideally, from a conceptual data modeller point of view, one should have (i) 'simple' constructs (with one icon) to add a TOG component to a conceptual data model, *e.g.*, for granular perspective such that all constraints pertaining to GP in the TOG automatically follow and are enforced, and (ii) a toggle feature to switch between a plain conceptual data model and the orthogonally positioned granular version, which are linked through level assignments of the entity types in the conceptual data model. Thus, one could extend those extended conceptual data modelling languages to cater for these two requirements, or start from scratch with ORM, ER, or UML. In contrast to ER and ORM, with full UML one could build upon ontology-enhanced UML (Guizzardi *et al.*, 2004a,b; Guizzardi, 2005) and/or re-use and augment UML packages with stereotypes and OCL constraints to function as basic containers for levels and perspectives. This avenue has been pursued by Luján-Mora *et al.* (2002, 2006) who have integrated it with RationalRose for DWH modelling. The main problem, as with YAM^2, is the informal nature of UML: even though a formal characterisation of UML is available (Berardi *et al.*, 2005), this leaves open what packages are (they are intended for modularisation) and what their formal relation is to UML class diagram constructs and models. An alternative avenue in the same direction that is being explored is to use one common DL language for all the main conceptual data modeling languages and simplify the TOG to fit the chosen DL; see *Appendix C.1* and Keet (2008a,c). Thus, there are ample opportunities for investigating transformation of the TOG into usable primitives and constructs with icons and other user-friendly modelling solutions in conceptual data modelling languages and their CASE tools.

Ontology linking. Although the main focus is computational support for granularity, one example will be given where it is, in fact, the *modelling exercise* that is of greatest benefit, which does not require software support other than for organisational purposes—at this stage. Non-granular ontology linking and integration is an active area of research over the past decade. It

Algorithm 7 Allocate new entity NewC to the right granular level

```
Require: $gran(\texttt{D}) = x$ «where x is e.g. $gp_1gl_2$»
Require: $RE(x, z)$ «where z would then be $gp_1$»
Require: $\varphi(\texttt{NewC, D})$ «where $\varphi$ is a granulation relation $GR$, such as $involved\_in$»
 1: if exists $y$ and $RL(y, x)$ and $involved\_in(\texttt{NewC, D})$ then «thus y would then be $gp_1gl_3$, and we
    demonstrate it for $\varphi = involved\_in$»
 2:   $assignGL(\texttt{NewC}, y)$
 3: else
 4:   $y \Leftarrow instantiate(GL)$
 5:   $assignP(y, z)$ «or: add axiom $RE(y, z)$ to the knowledge base»
Require:   $RE(y, z)$
 6:   $assignGL(\texttt{NewC}, y)$
 7: end if
 8: if $is\_a(\texttt{NewC, X})$ and $grain(\texttt{X}) = w$ and $tgL(w) = \phi$ and $grel(w) = \psi$ then «to check for possible
    assignments to other levels, where w is e.g. $gp_2gl_i$»
 9:   switch
10:     case $\psi = ppart\_of$ and $\phi = nrG$
11:           $assignGL(\texttt{NewC}, w)$
12:         ⋮
13:     case $\psi = is\_a$ and $\phi = nasG$
14:     if      exists $v$ and $RL(v, w)$ and $RE(w, u)$ and $RE(v, u)$ and $u \neq z$ then
15:               $assignGL(\texttt{NewC}, v)$
16:     else
17:               $v \Leftarrow instantiate(GL)$
18:               $assignP(v, u)$ «or: add axiom $RE(v, u)$ to the knowledge base»
Require:          $RE(v, u)$
19:               $assignGL(\texttt{NewC}, u)$
20:     end if
21:   end switch
22: end if
23: if $participates\_in(\texttt{Z}, \texttt{NewC})$ and $grain(\texttt{Z}) = t$ then
24:   do similar processing to allocate NewC in $s$ where $RL(t, s)$ holds
25: end if
```

can be carried out with a variety of approaches, such as mapping ontologies with rules (Heflin and Hendler, 2000) or approximate ontology translation with an inserted O_{map} between the other ontologies (Akahani *et al.*, 2002) or loose mappings (Calvanese *et al.*, 2001b), and other approaches with their respective issues (*e.g.*, Bouquet *et al.*, 2004; Klein, 2001; Noy and Musen, 2002, among many); see Keet (2004a) for a literature review. It will be demonstrated here that with a domain granularity framework, the connectivity is not just any mapping, but semantically enriched with *why* and *how* the mapping has been performed. The connected ontologies remain separable, yet are integrated into one coherent system within this domain granularity framework. One of the generic scenarios is illustrated in *Example 5.3*, which is illustrated for OBO Foundry ontologies in (Keet, 2008b) and touched upon in *Figure 1.2*.

Example 5.3. Let us take three ontologies $\mathcal{O}_1$, $\mathcal{O}_2$, and $\mathcal{O}_3$, each with their own subject domain (*e.g.*, Molecules, Biological processes, and Signalling pathways respectively), $\mathcal{O}_1 \cap \mathcal{O}_2 = \emptyset$, $X \in \mathcal{O}_1$, $Y \in \mathcal{O}_2$, and $Z \in \mathcal{O}_3$. Each ontology has its own domain granularity framework, d_1^f, d_2^f, and d_3^f, and the linked ontologies framework is d_c^f. Further, in shorthand notations, we have $RE(gp_i, d_3^f)$ with $has\_granulation(gp_i, nrG)$ and for this instance $uses\_GR(nrG, is\_a)$, and we have $RE(gp_j, d_3^f)$ with $has\_granulation(gp_i, nrG)$

and $uses\_GR(nrG, ppart\_of)$—a taxonomy and a partonomy of signalling pathways—and likewise (either one or both) for the perspectives in d^f_1 and d^f_2. From the domain knowledge, we know that $abs_{f3}(X, Y) = Z$, hence, also that, because of abs, (i) X and Y have to be related to each other (ii) X, Y are in a finer-grained level than Z is, and because of the "$_{f3}$", that that granulation must be of type **nfG**. From the known perspectives in d^f_1, d^f_2, and d^f_3, we know that no such perspective exists that has granulation **nfG**, hence to link $\mathcal{O}_1$, $\mathcal{O}_2$, and $\mathcal{O}_3$, we need a *new* perspective, gp_n, where $RE(gp_n, d^f_c)$ and $has\_granulation(gp_n, nfG)$, to relate the ontologies such that $\mathcal{O}_3$ resides in $gp_n gl_1$ and $\mathcal{O}_1$ and $\mathcal{O}_2$ in $gp_n gl_2$. ◇

As the example illustrated, a domain granularity framework can be utilised for *meaningful* ontology linking and integration. A next step is cross-ontology reasoning over such ontologies linked by granularity, which is already a requirement from bio-ontologists (OBO Foundry, 2006; Keet *et al.*, 2007; Smith *et al.*, 2007). For instance, there are OWL-DL versions of DOLCE and BFO which aid smooth integration with bio-ontologies and other ontologies in the Semantic Web. However, to realise this, the TOG will have to be transformed into a DL representation first.

5.6 Summary

In this chapter we have taken a closer look at various proposals for representing and implementing granularity, compared eight main contributions against the TOG key requirements, and assessed also other related works from diverse fields, such as ontology, data warehousing, rough sets, context, biomedicine GIS, and time. No proposed theory or engineering solution meets all the key requirements, although MultiDimER (Malinowski and Zimányi, 2006) comes closest. Hence, regarding the representation of a theory of granularity, the TOG compares favourably with all current other theories and engineering solutions, is more generic without being too encompassing and thereby has a wider applicability than the extant (partial) theories.

Considering the TOG functions, they are more diverse and conceptually cleaner than most of the theories and engineering solutions. The reason for this is that with the comprehensive framework, one can query each component and combinations thereof, which other theories and engineering solutions do not have at their disposal or only to a limited extent for they have a restricted application scenario and less components in the theory to query. On the other hand, a few highly specialised engineering and domain solutions do have implementation-specific functions, such as the level discovery (Fagin *et al.*, 2005) and granularity conversions for time granularity (Bettini *et al.*, 2007). Such implementation-scenario functions, however, not only can be added easily to a TOG-based implementation but also simplify the original functions; *e.g.*, Ning *et al.* (2002) already tried "labelled set" as predecessor to granular level to ease function definition, which has become straightforward with the TOG components.

Assessment of granularity as used in the subject domains of biology, biomedicine, GIS, and ecology revealed that modelling of biological granularity remains primarily at the stage of informal representations. They thus can benefit greatly from an ontologically motivated, formal theory of granularity, thanks to the structured approach toward modelling that the TOG can offer, the interoperability among existing theories at different levels of granularity, and the query functionality to utilise the representation and retrieve granular information.

Last, we looked as limitations of the TOG and some implementation considerations to link the TOG to a domain granularity framework d^f and the data source $\mathcal{DS}$ to generate a $\mathcal{DG}$. The main theoretical hurdle for computational implementations is the complexity of the language to formally characterise the TOG and the simplifications for diverse implementation scenarios. Using the full TOG can of course be used for manual efforts on database integration and ontology linking, and for augmenting conceptual data modelling languages with an orthogonal granularity dimension.

Chapter 6

Conclusions and further research

This chapter presents the conclusions about the research presented in this thesis and contains suggestions for future work. Each of the preceding chapters contain summaries of the respective sub-topics that will be related and integrated here in order to answer the research questions and tasks set out in Chapter 1.

6.1 Conclusions

The fundamental contribution is that we moved from a data-centric treatment of granularity to the conceptual and logical layers, where informally defined components of granularity have become ontologically-motivated modelling constructs proper. This was achieved through the analysis and formalisation of three elements:

I. The *foundational semantics of granularity* are disambiguated and structured in a taxonomy of types of granularity. This taxonomy makes explicit both the ways of granulation and representation, and how entities are organised within a level of granularity.
II. The static *components of granularity* were subjected to an ontological analysis and formalised in a consistent and satisfiable logical theory, the Theory Of Granularity (TOG), to ensure unambiguous semantics;
III. An extensible set of domain- and implementation-independent *functions for granular querying and inferencing* were defined for the TOG components to enable granular reasoning over the theory that can be used across multiple implementation scenarios.

Effectively, granularity is both lifted up to a higher layer of abstraction alike conceptual modelling does for software design and database schemas and precisiated in a formal theory with model-theoretic semantics, thereby having made the representation of granularity domain- and implementation independent. Hence, reusability across implementations is ensured, which in turn facilitates interoperability among information systems, such as granulation of ontologies, knowledge bases, databases, data warehouses, and biological and geographical information systems.

6.1.1 Research questions revisited

We now revisit the research questions and tasks from Chapter 1. The main research question was *Why, how, and where will usage of granularity improve knowledge representation and knowledge management?* This question was broken down into four main research questions and five tasks and, for each sub-topic, further questions, tasks, and requirements. After answering those questions and solving the tasks, we return to the main research question.

1. Can a subject domain-independent reusable theory of granularity be defined and formalised?
Yes. A consistent and satisfiable Theory Of Granularity (TOG) was defined and formalised in first order predicate logic regarding both the *static* components that can serve as modelling constructs in ontologies and conceptual data models and *dynamic* components with functions to use the static part effectively for granular querying and reasoning. To substantiate this short answer, the four sub-questions will be answered now.

1a. *What are the characteristics of different 'types' of granularity?*
Based on the investigation in Chapter 2, one can identify 4 principal dimensions where types of granularity and their representation differ, which are: (i) Arbitrary scale (quantitative) versus non-scale-dependent (qualitative) granularity; (ii) How levels, and its contents, in a perspective relate to each other; (iii) Difference in emphasis for granulation, being entity-, relation-, or criterion-focused; and (iv) The (mathematical) representation, such as based on set theory versus mereology. These dimensions of the mechanisms of granulation motivated development of a taxonomy of types of granularity, which is depicted in *Figure 2.3* and has been explained in §2.2. The distinguishing characteristics of each type are summarised in *Table 6.1* and from which follows how a level's contents are structured when adhering to a leaf type (§2.3).

Table 6.1: Distinguishing characteristics for branching in the taxonomy of types of granularity.

Branching point	**Distinguishing feature**
sG – nG	scale – non-scale (or, roughly: quantitative – qualitative)
sgG – saG	grain size – aggregation (or: scale *on* entity – scale *of* entity)
sgrG – sgpG	resolution – size of the entity
saoG – samG	overlay aggregated – entities aggregated according to scale
naG – nrG – nfG	semantic aggregation – one type of relation between entities in different levels – different type of relation between entities in levels and relations among entities in level
nacG – nasG	parent-child not taxonomic and relative independence of contents of higher/lower level – parent-child with taxonomic inheritance

1b. *What are the key requirements for and components of granularity?*
Key requirements that any theory of granularity should meet followed from the assessment of the state of the art and investigation into the foundational semantics of granularity. These 12 requirements were formulated in §3.1 and are satisfied by the TOG (§3.10.1). They are:

- ⋆ A theory of granularity should be usable eventually in a format for contents at the instance level and in a format for defining a domain granularity framework at the type level;
- ⋆ A higher level simplifies, makes indistinguishable, the finer-grained details that are indistinguishable at that higher level;
- ⋆ Following from the indistinguishability requirement, there have to be at least two levels within a perspective, else there is no granularity;
- ⋆ Any theory must be able to accommodate both the quantitative and qualitative aspects of granularity (or: arbitrary scale and non-scale-dependent granularity);
- ⋆ The logic-based representation has to permit the two main ways for perceiving and representing granularity, being set theoretical and mereological, therefore the relations between the entities (/types) contained in the levels and the relations between granular levels are, at least, either of the type is_a or $part\_of$;
- ⋆ Given that one granulates according to a certain type of granularity, this also means that there has to be one type of relation between granular levels within a particular granular perspective;
- ⋆ For ontological correctness and computation, the type of relation between adjacent levels

in a perspective has to be transitive for those perspectives that contain >2 levels;

- ⋆ A type of relation that relates contents in levels of granularity within a particular perspective can have the property of being intransitive, provided there are always exactly 2 levels in any given perspective that contains that type of relation relating the entities (/types);
- ⋆ The entities or entity types in a particular granular level have at least one aspect in common, which is a criterion by which to granulate the data, information, or knowledge;
- ⋆ An entity (/type) never can reside in more than one granular level within the same granular perspective;
- ⋆ Given that each level is contained in a granular perspective and granular perspectives contained in a domain granularity framework are disjoint, an entity (/type) in a particular granular level may reside in ≥ 1 levels, provided that each level the entity (/type) is contained in a distinct granular perspective;
- ⋆ If there is more than one granular perspective for a subject domain, these perspectives must have some relation among each other.

In addition, the TOG uses both part of the formal foundational ontology DOLCE to ensure unambiguous typing and the well-known Ground Mereology. Moreover, it followed from the comparative assessment in Chapter 6 that, to the best of my knowledge, there is no other theoretical or engineering solution that meets all these requirements.

With an eye on implementations, four *desirable features* were formulated, being (A) The entities (/types) are disjoint; (B) The entities (/types) are not necessarily exhaustive, because this may not be possible due to our gaps in knowledge of the natural world (within a closed world assumption, they are (disjoint) exhaustive); (C) Provided an entity (/type) is not an orphan in the original data source and the subject domain is covered fully with granular perspectives, it must reside in at least one granular level; (D) The purposes of its usage—dynamic aspects with primary distinctions allocating/classifying entities (/types) and information retrieval & reasoning—shall not affect the static structure of a theory of granularity. Feature D is fully satisfied: all functions defined in Chapter 4 use TOG components but none changes any of its properties and no additional 'patchwork' components are needed for the functions to behave as specified. Feature B is trivially met and feature C is met as well, although C may benefit from controlled flexibility for experimentation with bio-ontologies. Feature A does not affect the static components of a granular framework and is from that viewpoint uninteresting, but, depending on the actual data source, might have an effect on successive firing of abstraction functions, therefore it is highly recommendable to satisfy A but not mandatory (§5.5.1).

The key structural components of granularity are depicted in the top-half of *Figure 3.1*. They are a *domain granularity framework* demarcating the boundary, *granular perspective* with its *criterion* and type of granularity by which one granulates the data, *granular level*, and the relations between them, being RE between domain, perspective, and level and RL between the levels in a granulation hierarchy within a perspective, where each one has their relata constrained to the appropriate TOG-components. There are further details and constraints among these components (§3.8), which have been formally characterised and proven in Chapter 3, with 19 lemmas and 7 theorems.

1c. *What are the characteristics of those components of granularity, such as granular level and perspective? How are they related to each other? How are levels of granularity related, what are the (primitive) relations?*

Characteristics of the TOG components received detailed attention in Chapter 3 and are summarised here. The *granulation criterion* is composed of properties—qualitative or quantitative—according to which one granulates the data; that is, one focuses on a subset of properties of all the properties of entities (/types), which is called the *granular perspective* on the subject domain. A domain granularity framework can contain multiple granular perspectives. Granular perspectives can be uniquely identified by the combination of the criterion and the type of granulation used. Together with the similarity, equivalence, and indistinguishability relations, one identifies *granular levels* that adhere to the type of granularity of the perspective they are contained in, and by which one assesses which entity (/type) belongs to which level in its appropriate perspective. Based on these ingredients, *Theorem 3.2* was proven, which says that *a granular perspective must contain at least two granular levels*, else there is no granularity. Refer to §3.2-3.4 for details.

The two main relations in the granularity framework are the relation between the three components (domain, perspective, and level), RE, and between levels within one perspective, RL (§3.5-3.6). To be precise about the ontological aspects and semantics of these relations, meronymic and mereological part-whole relations were structured in a taxonomy based on two principal features—being transitive or not and the category(/ies) of the relata—and its leaf types were formally characterised (§3.5.1). The mereological proper parthood relation is used for RE, and structural proper parthood for RL. The latter is formulated in *Theorem 3.5*, which says that *RL is of the same type, s_ppart_of, between granular levels in* all *granular perspectives* and is not only transitive, asymmetric, and irreflexive, but also acyclic. In addition, there is a *single* path from the finest-grained level up to the coarsest one, and vice versa, within one perspective—thus with a *1:1 multiplicity on RL and its inverse, RL^-*—that is proved in *Theorem 3.6*. Mereology is used for relating components of the *framework*, which does not exclude using set theory-based granularity among the *entities (/types)* that fill the levels. In fact, it is not either-or, but *both-and*, because of consistent distinction between characteristics of granularity at the conceptual and ontological layers and of the granular data at the logical and physical layers, and separating the granularity components in the TOG from the types of granularity. Thus, the TOG can be used for both set-based granularity and mereology-based granularity for they are identified as different types of granularity and augmented with recording the *granulation relation* used in the data source. Currently, six basic granulation relations have been defined—$ppart\_of$, $contained\_in$, $involved\_in$, is_a, $member\_of$, and $participates\_in$—where the latter two have the constraint that there are exactly two levels in that perspective when they are used between the entities (/types) in the levels.

1d. *Based on the types of granularity, where and how does this influence a reusable theory of granularity?*
There are three principal points where types of granularity (TG) influence reusability; refer to *Figure 3.1* for a cursory graphical rendering. First, TG is related to both granular perspective and granular level and is one of the two essential components for the identification of granular perspectives (§3.3, *Theorem 3.1*). Thus, one has to specify the TG and content structure for the leaf types only once and can reuse it for multiple perspectives. Second, having TG related to granular level greatly simplifies content retrieval: for each TG, one specifies only *once* the procedures or functions how contents adhering to that type of granularity have to be retrieved. Subsequently, any level that has as attribute that TG can immediately use that function without having to re-code the same function for each level in the information system (§4.2.3). Third, it is an effective vehicle to represent the granulation relations used in the data source between entities (/types) of different level of detail and, moreover, through the TG, the TOG accommodates *both* set theory-based granularity *and* mereology-based granularity.

1—*On domain independence and reusability*
The representation of granularity is domain- and implementation independent, because (i) granularity is lifted up to a higher layer of abstraction and its components, such as level, perspective, and the relations, have become separate constructs; (ii) these TOG components do not depend on a particular subject domain and can be equally used for qualitative and quantitative granularity; and (iii) the formal characterisation is in FOL as opposed to limited to one particular programming or conceptual data modelling language. Point iii covers reusability because the FOL theory can be transformed to various implementation languages, as has been illustrated in *Example 4.9* and §5.5.2. Thus, reusability across implementations is ensured, which in turn

facilitates interoperability among information systems that deal with different levels of granularity. Reusability has been taken into account also within the TOG, most notably with the types of granularity, which was described in answer 1d above.

2. How does a meta-level theory of granularity constrain domain specific granularity?
Figure 1.1 depicts informally that the TOG constrains a domain granularity framework, d^f, but does not explain how this can materialise. This has been assessed in §5.5.2, experimented with for infectious diseases, the FMA, and GO (Keet, 2006b,c), and illustrated in *Example 5.1*. Checking compliance of a d^f with the TOG depends on the implementation scenario. For databases, including ontologies stored in a database, one can avail of the usual integrity constraint checking. For ontologies or conceptual data models, one needs second order logic and/or means such as UML stereotypes to have the TOG components instantiating d^f at the type-level. For knowledge bases, one can choose either one of the former or use an additional transformation step whereby the instances in a d^f are nominalised into singleton sets and then used with the rest of the type-level subject domain information.

3. How to relate domain data to a domain granularity framework and apply a domain granularity framework to data?
From a theoretical perspective, the identified subsets and their relations between TOG, d^f, and the data source $\mathcal{DS}$ were proven in §4.2.7 and summarised in *Table 4.2*. To answer the second part of the question, the following two sub-questions will be answered first.

3a. *What are the kind of design decisions, if any, to make a theory of granularity implementable?*
There are three types of design decisions. First, there are theoretical limitations (§5.5.1) to the current formalization in FOL, which also have been demonstrated experimentally with the Mace4 model searcher (§3.8.3). Therefore, choices have to be made as to which constraint(s) of the TOG will be left out in order to represent granularity with a decidable expressive yet scalable Description Logic (DL) language; see also *Table 5.3*. The DL language that has most of the required constructors is $\mathcal{DLR}_{\mu ifd}$ and then $\mathcal{SROIQ}$, which are ExpTime-complete (§5.5.1) and NExpTime-hard (Horrocks *et al.*, 2006) for concept satisfiablity, respectively. If the TOG is used for an off-line paper exercise for modelling (see *Example 5.3* on ontology linking), then one can use the full TOG, of course. Second, assuming a computational implementation, the application scenario requires design decisions, which involves (i) the requirement for computational support for automated consistency checking (see also the answer to question 2 and §5.5.2) and (ii) if it is to be implemented in a database, knowledge base, or ontology (see 3b). Third, the latter induces another set of design decisions, which are technological (which language (§5.5.1), automated reasoner, options for graphical support, maintenance strategy, and so forth) and domain expert's requirements (information systems granulation, integration, performance optimizations, and so forth).

3b. *Where does it make a difference in proposed solution using a database versus knowledge base or ontology as type of data source?*
The differences follow from the distinction between instances versus universals (§2.3.1). Logically, databases contain instances, knowledge bases can contain both instances and universals (*e.g.*, DL-concepts), and, ontologically, an ontology has either instances or universals but not both. In addition, abstractions and expansions are usually performed on type-level knowledge (§4.3 and §5.4) whereas aggregation of instances is prevalent in databases. The functions to query the granularity framework (§4.2) use granularity framework instances, but the functions can be amended easily by changing the definitions' arguments so that the sets be sets with universals as members instead of instances of TOG framework components. Practically, however, automated constraint checking of a d^f's compliance with the TOG is not only easier for database implementation than for type-level ontologies (see answer to question 2), but also other auto-

mated reasoning services over instances is computationally less costly.

Returning now to the second part of question 3—applying a domain granularity framework to data—there are several implementation issues concerning computational support, which follow from the observations in 3a and 3b above. Simple applications to automate granulation of an information system, such as discussed in *Example 2.5* and *Figure 2.5*, is easy to implement for databases (Keet, 2006b), and so is computational implementation of *Algorithm 6* to load a taxonomy into a level through a recursive query. Other possibilities were discussed in §5.5.2 and illustrated in *Example 5.2*, including automation of level assignment to speed up granulation by inferring the appropriate level of a new entity (/type), which is proposed in *Algorithm 7*.

4. What reasoning tasks require, or can benefit from, granularity? Where and how will this affect the following types of tasks?
This question and its two sub-questions have been refined first into four requirements, desirable feature D (which has been addressed above), and four tasks (see §6.1.2 below), hence the question will be answered after the following summary of the requirements.

A set of functions for the TOG *with which one can: (i) Query the* TOG *components directly; (ii) Retrieve content of levels; (iii) Move between levels from coarser to finer-grained entities (/types) and back. Content retrieval has to make use of the existing structure in the original data source and consider the type of granularity that is used for the granulation.*
The 19 functions to query the TOG components and to retrieve content of levels are described in §4.2 and summarised in *Table 4.1*. They cover level assignment; selection and retrieval of the domain, perspective, and level; retrieval of contents of a level and intersection of contents from different levels; and selection of one or more entities (/types) and subsequent retrieval of the one or more levels it (/they) reside in. Moreover, one can define functions to retrieve each component of the TOG, such as retrieving the granulation relation between entities (/types) in adjacent levels, retrieve the criterion of a granular perspective, and a level's values.

The main function for content retrieval is $getC$ (*Definition F.7*). §4.2.3 elaborates on nested functions that are required for each leaf type in the taxonomy of types of granularity in order to preserve the structure of the level contents upon retrieval. The structure can be accessed through using the tgL function (*Definition 4.3*) that is based on the $adheres\_to$ relation between a level and its type of granularity with the structure of the contents for each leaf type (§2.3). *Algorithm 1* demonstrates $getC$'s usage. Thus, one has to define one retrieval function for each leaf type only once instead of repeating this for each perspective and level in a granulated information system.

Moving between levels from coarser to finer-grained entity types and back has been addressed with the abstraction and expansion functions (§4.3), which are summarised in *Table 4.3* for the basic and in *Table 4.4* for the complex abstractions, and *Table 4.5* for the expansion functions that are, at the conceptual level, the inverse of abstraction. These functions cover the three principally distinct ways of making abstractions. Because abstraction and expansion are more common for type-level logical theories (ontologies, conceptual data models), they have been defined for types only, although one could apply them equally to instance-level data as well. Their usefulness was demonstrated with an extended example in §4.5.1.

Abstraction (expansion) functions have to enable moving from finer-grained entity types to a coarser-grained simplification (vive versa), which may reside in an adjacent coarser granular level or higher up in a level in the same perspective (vice versa).
Integration of the abstraction (expansion) functions with the TOG is ensured with *Proposition 4.1* and *4.2* (*Proposition 4.3* and *4.4*) that relate abstraction (expansion) level to granular level and abstraction (expansion) hierarchy to granular perspective. To ensure correct behaviour of the expansion functions, *Constraints 4.3-4.6* were added and a sample algorithm provided (*Algorithm*

4) that has nested functions to retrieve the contents with $getC$, hence, taking into account the type of granularity and any structure that the contents may have.

Returning to the original research questions, then their respective answers are as follows.

4a. *For usage of the structure:* e.g., *level selection, fact finding, ontology browsing.*
Effective usage of the TOG components is achieved with the 19 functions defined, discussed, and illustrated in §4.2 and summarised above. In addition, there are 27 entity (/type)-focussed functions with the abstractions and expansions. These functions can be used for level selection, content retrieval, ontology browsing, and so forth. The abstraction and expansion functions (§4.3), which rely on several TOG component functions, are particularly useful for ontology and conceptual data model browsing and management. Fact finding in data warehouses has not been addressed explicitly, but can be done easily with a few of the 46 pre-defined functions: with $grain$ one get the levels an entity resides in, $crit$ retrieves the granulation criterion ($crit^{-}$ the perspectives using the selected criteria), $value$ returns the value of a level, and $grel$ returns the granulation relation; hence, one can search for entities (/types) that satisfy certain criteria and select values from the $value$ answer to narrow down the query answer, or given an entity, retrieve the levels it (they) reside in. In addition, by having the TOG components as easily accessible queryable components, conditional selections and rules between level can be specified as well, such as the GIS example in §5.3.2. Further, with a simple extension to granular level as depicted in *Figure 5.1*, one also can query a selected level's approximation space, if defined. Last, thanks to the types of granularity and the RP relation between perspectives, additional (time) granularity conversions (§5.3.3) are possible as well.

4b. *What are the ways for moving between levels (abstraction and expansion) and which functions do they involve?*
Three conceptually distinct ways of abstraction were identified, consisting of remodelling a relation between finer- and coarser-grained entities as a function (R-ABS), folding multiple entities and relations into a different type of entity (F-ABS), and hiding or deleting less relevant entities and relations (D-ABS). The 14 abstraction functions (§4.3) fall in to one of these three categories. The corresponding 13 expansion functions rely on $getC$ together with the type of granularity of the perspective where the expansion is performed.

6.1.2 Solving the tasks

The solutions to the five tasks set out in Chapter 1 have been addressed in Chapter 4 and 5, which are summarised below in a self-contained manner. To meet the tasks, aforementioned 46 functions for querying granulated data sources have been defined, which enable reasoning over the granular levels and its contents. These functions were defined at the conceptual level with the goal each function aims to achieve, as well as the logical level to give precise semantics to the functions to ensure reusability of the functions and interoperability among the granulated information systems.

Task 1. *Perform a selection of levels from a particular domain granularity framework* d_i^f, i.e, $task^1(d_i^f) \rightarrow lss_i$, *where* $\mathcal{DL}$ *is the set of levels within that domain granularity framework and* lss_i *the selected subset.*
The simple level-selection functions (*Definition F.2*, *F.4*) meet a limited case of $task^1$, because there one can select only one level within a pre-selected perspective; likewise, $selectLs$ for selecting multiple levels within one perspective (*Definition F.5*). It is addressed fully with the $selectDL$ function (*Definition F.6*) that returns the set of selected levels, lss_i, in the domain granularity framework (d_i^f). Overall, $task^1$ consists of an iteration of three nested functions, which are

$selectP$ for selecting a granular perspective, $getL$ to retrieve the granular level, and then either $selectL$ to select one level or $selectLs$ to select multiple levels (depicted in *Figure 4.1*). This highly modular approach ensures flexibility for its implementation so that for any application the *goal* is the same but particular program code can vary to, for instance, tune performance of the information system.

Task 2. *To retrieve contents of at least one level, we have* $task^2(gl_i) \rightarrow \mathcal{E}$*, where* $\mathcal{E}$ *denotes the contents of a granular level, that is, entities or types and, where applicable, any further structure other than an unordered set.*
This task can be executed with the $getC$ function to get the contents of a level (*Definition F.7*) and its supporting nested functions to, first, retrieve the type of granularity that a level adheres to with tgL (*Definition 4.3*) and, second, the required procedures specific for the leaf type of granularity to ensure contents can be retrieved whilst preserving the structure of the entities (/types) in the level (§4.2.3). To support seamless execution of $getC$, appropriate usage of the functions is proposed in *Algorithm 1*.

Task 3. *A formalised relationship or transformation rule is required between a data source,* $\mathcal{DS}$*, and domain granularity framework applied to the data source,* $\mathcal{DG}$*, to utilise them both for some particular reasoning task. Therefore,* $task^3(\mathcal{DS}, \mathcal{DG}) \rightarrow \mathcal{DS}\ related\_to\ \mathcal{DG}$*, where the "*$related\_to$*" has to be specified. Likewise, the relation between* $\mathcal{DS}$ *and its selected granulated subsets,* i.e. $\mathcal{DG}^1$, ..., $\mathcal{DG}^n$*, has to be specified.*
Task-specific, a data source—database, knowledge base, or ontology—with its data, $\mathcal{DS}$, is a proper subset of the applied domain granularity framework (granulated data source), $\mathcal{DG}$, (*Lemma 4.1*). In addition to this relation, several other useful subsets and parts of $\mathcal{DG}$ were identified, which simplify granular query formulation and answering, such as the set of levels in a domain granularity framework ($\mathcal{DL}$), the collection of universals or particulars residing in a particular granular level ($\mathcal{E}$), and the intersection of two levels ($\mathcal{I}$). These subsets, or parts, are summarised in *Table 4.2* and their relations were proven in *Lemmas 4.1-4.8* with additional *Corollaries 4.1-4.4*.

Task 4. *Use a combination of levels of the same or different perspectives on the data source to which a granularity framework has been applied,* $\mathcal{DG}$*. The result is a subset of* $\mathcal{DG}$*,* $\mathcal{DG}'$*; thus, the task is* $task^4(\mathcal{DG}, \mathcal{DL}) \rightarrow \mathcal{DG}'$*, where* $\mathcal{DG}' \subseteq \mathcal{DG}$*.*
This task builds upon $task^1$ and $task^2$, hence, uses the above-mentioned functions, the algorithm for content retrieval (*Algorithm 1*) and, optionally, functions for further processing, such as $intersect$ for intersection of contents of different levels (*Algorithm 2*). The combination of these inputs is presented in *Algorithm 3*, thereby solving $task^4$. More sophisticated procedures can be composed when also entity (/type) selection with $selectE$ and abstraction and expansion (§4.3) are at one's disposal for conditional selection and retrieval. This was demonstrated in *Algorithm 5* and in §4.5.1 with an extended granular reasoning example about hormones in the liver.

Task 5. $task^4$ *has* ex^2 *as sub-task or can be performed vice versa, therefore they can be combined into one more complex operation:* $task^5(d_i^s, d_i^f, \mathcal{DL}) \rightarrow \mathcal{DG}'$*.*
$task^5$, which combines $task^4$ and experiment ex^2 into one, is satisfied only to a limited extent because of its dependency on ex^2, which, in turn, is dependent on ex^1. Experiments ex^1 and ex^2 have been carried out successfully, but mostly manually (Keet, 2006c), because there remain several hurdles for successful implementation and automation of granular querying. The complicating factors are (i) the constructs used in the TOG (§5.5.1) and limits of type-level query languages and reasoners (*e.g.*, the RBox Compatibility service (*Definition 4.8*)), (ii) the chosen implementation scenario for databases, knowledge bases, and ontologies (§5.5.2), and (iii) suboptimal engineering practices of extant information systems (Keet, 2006b).

Why, how, and where will usage of granularity improve knowledge representation and knowledge management?

Why Granularity is a novel way of structured knowledge representation that is orthogonal to extant modelling methods. It enables a *structured and consistent way for carving up the subject domain* through granularity type selection and criterion-selection for a particular granular perspective and at different levels of detail so that a user can avoid unnecessary detail or too coarse generalisations, yet have the full domain at one's disposal for reasoning across levels of granularity, and also be deployed for users with different foci.

How The formal, hence unambiguous, Teory Of Granularity, TOG, can be added to ontologies, knowledge bases, database, and conceptual data models with a formal foundation. That is, the hitherto data-centric solutions for dealing with granularity are lifted to the ontological and logical layers and the informal subject domain and philosophical contributions are precisiated in a formal theory, thereby *providing easily reusable modelling constructs for granularity components* such as level, perspective, type of granularity, and granulation criterion. These static components are augmented with *clear, extensible functions for granular querying and reasoning* that are reusable across implementation scenarios.

Where The structured and consistent representation of granularity enabled by the TOG contributes to *good modelling methodology and practice, hence, to the overall quality of the data source*. The genericity of the TOG can guarantee *interoperability between granulated data sources*, such as GISs and bio-ontologies, because its FOL representation is implementation- and scenario-independent. Further, it offers an additional method for *linking data sources that contain contents at different levels of granularity*, and, subsequently, *reasoning across those levels of granularity*.

6.2 Further research

Several topics merit further investigation regarding both the theory and transformation of the TOG to representation- and query languages that are used commonly in information systems. The more interesting ones have been touched upon in §5.5, which will be summarised here and augmented with a few practical suggestions for bio-ontologies and bioinformatics.

Theory. Areas of further investigation can be to specify in finer detail the interrelation between mereology and granularity—in particular the spatial component to refine containment (Borgo and Masolo, 2007) and aggregates—, rough/fuzzy extensions for level specification, and if adding rules for conditional level retrieval may be sufficiently useful to incorporate into a generic theory. In addition to the 46 functions that have been pre-defined for the TOG, one could look at other extensions, such as granularity conversion functions between levels in distinct perspectives that are granulated with the same or different type of granularity. Another direction could be to transform the TOG constraints into a Description Logic language and how it propagates to conceptual data modelling languages. This involves addressing satisfactorily a computational implementation of antisymmetry, (ir)reflexivity and acyclicity, be it in a $\mathcal{DLR}_{\mu ifd}$ or another DL language. Alternatively, one may want to pursue direct translation from the TOG into a conceptual data modelling language and automatically propagate it to data in a database. Also, and with a look ahead toward implementation, it may be of interest to work out in detail the database integration or ontology linking with a domain granularity framework in addition to the usual LAV approach. This would also involve an assessment of query languages and their trade-offs for supporting different reasoning services and scenarios.

Engineering. Aside from possible engineering follow-ups of results from the theory paragraph, above, one could develop graphical support to model the granularity framework components to improve usability, develop automated abstraction and expansion based on the TOG functions and experimentally evaluate the proposed 7 algorithms. Also, integration of the TOG with MADS (Parent *et al.*, 2006a,b) may give tangible results in a short time frame. A different avenue could be to use granularity in the query evaluation strategies & reasoning by availing of the modularity of the functions where several complex functions can be built up from the simple ones.

Subject domain: biology. Analogous to addressing the Ontology Comprehension Problem with granularity and abstraction, one may want to apply it to visualisation of, *e.g.*, metabolic pathways. Further, one could to add granularity to information retrieval from large corpora such as enabling finer-grained PubMed searches by enhancing tools such as GoPubMed (GoPubMed, 2007) with indexing the search results not only by the GO and MeSH taxonomies, but also at different levels of granular detail. Regarding the theory and engineering of ontology linking and cross-granular querying, one could experiment with the OBO Foundry that already has these aims on paper (Keet, 2008b; OBO Foundry, 2006; Smith *et al.*, 2007). Useful first steps may be carried out with the FMA by giving computational support for granularity that is currently possible only as one-off queries with the OQAFMA query agent (Mork *et al.*, 2003; OQAFMA). That is, to position a domain granularity framework orthogonally to an existing data source so that reasoning over applied domain granularity can be made possible more comprehensively. Further, the TOG can be used to develop comprehensive GIS and anatomy granular perspectives so that legacy application hierarchies can be either improved or linked up for interoperability, or both.

Appendix A

Granulating the infectious diseases subject domain

A.1 Introduction

To give a flavour of diverse perspectives, levels and their contents, the subject domain of infectious diseases was granulated. This, *informal* granulation is shown in *Table A.1*, which is an extended version of the one published as (Keet and Kumar, 2005), yet shorter than the detailed granulation in (Keet, 2006c). Various examples in Chapters 2-4 analyse and discuss this granulation in more detail.

The following data sources were used to populate the granularity framework: (i) the FMA (2003) for the perspectives Site of entry and Site of effect and for other inferencing on anatomical structures; (ii) Snomed CT and ICD 10 for DiseaseClassification; (iii) the species taxonomy of the Tree of Life (Maddison and Schulz, 2004) for the Phylogeny of infectious organisms; (iv) the remainder of granular perspectives, levels and data, such as the Mode of transmission and Predisposing factors, are based on a manual compilation from various data sources: scientific literature (Nature Editorial, 2005; Rodal *et al.*, 2005; Wang *et al.*, 1999; Rossetto, 2000; Widell *et al.*, 1996; Hollinger and Kleinman, 2003; Weiss and McMichael, 2004), academic textbooks (Schlegel, 1995; Stryer, 1988), and others (NCID; Melhorn, 2004; Pathologie).

A.2 Informal granulation with sample contents for infectious diseases

Let us take subject domain d_1 = Infectious diseases and take a perspective on the domain that focuses on the pathological processes upon infection, gp_1 = Pathological process. With this gp_1, we can identify and specify granular levels containing processes and part-processes, such as gp_1gl_1 = System process and gp_1gl_3 = Congestive process, so that one can add, preferably automatically, the domain data taken from a knowledge base or ontology (such as PathInfo; Melhorn, 2004; Snomed CT; ICD 10; GO, 2004). For instance, the contents—*i.e.*, entity types or their instances from the chosen subject domain— at the gp_1gl_1 level will contain Inflammatory process and Proliferative process, and congestive processes in inflammation has entity types contained in the gp_1gl_3 level {Serous exudation, Vascular engorgement, Rapid bacterial proliferation}. Both the type-level specification of d_1, gp_1 and its levels are constrained by the domain-independent TOG so that one cannot specify perspectives and levels arbitrarily, and which, in turn, has constraint coming from the foundational semantics of granularity so that the perspectives and levels adhere to one type of granularity.

Table notes:
* Names in braces do not belong to the content of that level but are added for illustrative purpose to indicate the parent type in the coarser level of the entity types in that finer-grained level.
** Condensed version of the Site of entry (and Site of Effect) that has 9 levels in (Keet, 2006c).
*** Condensed version of the Phylogeny that has 6 levels in (Keet, 2006c).
gp'_4: redundant or competitive perspective w.r.t. gp_4.
gp'_9: refinement of gp_9 along a partonomic granulation instead of subsumption.

Table A.1: Condensed version of the informal granulation with 9 perspectives and up to 3 levels each with sample contents from the infectious diseases UoD.

GP	Criteria for perspectives		Sample contents for $\mathbf{gl}_1$	Sample contents for $\mathbf{gl}_2$	Sample contents for $\mathbf{gl}_3$
gp_1	Source	Mode of Transmission	Air-borne, Food-borne, Water-borne, Direct contact	(Direct contact:)* Person-to-person, Animal-to-person; (Food-borne:) Production, Preservation, Preparation	(Person-to-person:) Sexual intercourse, Skin, Blood
gp_2	Site	Site of entry**	Respiratory system, Digestive system	(Digestive system:) Stomach, Duodenum, Colon	
gp_3	Site	Site of effect	Respiratory system, Digestive system	(Digestive system:) Stomach, Duodenum, Colon	
gp_4	Infectious organism	Common name	Worms and flukes, Arthropods, Protozoa, Fungi and moulds, Bacteria	(Worms and flukes:) Roundworms, Hookworms, Tapeworms, Threadworms; (Bacteria:) Gram–, Gram+, Cocci, Rod, Flagellate	(Cocci:) Mono, Di, Strepto, Staphylo
gp'_4	Infectious organism	Phylogeny***	(Prokaryote:) Eubacteria; (Metazoa:) *Trypanosomatidae*, *Ancylostomatidae*	(Eubacteria:) *Salmonella* spp., *Aeromonas* spp.; (Trypanosomatidae:) *Leishmania* spp., *Tripanosoma* spp.	(Salmonella spp.:) *S. enteritidis*, *S. typhi*; (Leishmania spp.:) *L. braziliensis*, *L. donovani*, *L. tropica*
gp_5	Disease classification	Infectious disease	(Infectious disease:) Dysentery, Pneumonia, Meningitis	(Pneumonia:) Lobar pneumonia, Segmental or lobular pneumonia, Bronchopneumonia, Interstitial pneumonia	
gp_6	Pathology	Mode of action	Toxin-producer, Genetic interference	(Toxin-producer:) Stimulator, Inhibitor	(Inhibitor:) Covalent binding of the small subunit of the cholera toxin to G-protein, Covalent modification by pertussis toxin of inhibitory G_i protein
gp_7	Pathology	Pathological structure	Marbled parenchyma, White cicatricial tissue	(White cicatricial tissue:) Dense collagen connective tissue with reduced cell density	
gp_8	Pathology	Pathological process	Inflammatory process, Proliferative process	(Inflammatory process:) Congestion, Red hepatisation, Grey hepatisation, Resolution	(Congestion:) Serous exudation, Vascular engorgement, Rapid bacterial proliferation
gp_9	Predisposing factors		Living habits, Hereditary, Environment, Age	(Living habits:) Diet, Smoking, Stress, Personal hygiene; (Environment:) SocialEnvironment, EconomicEnvironment, PoliticalEnvironment, BiologicalEnvironment	
gp'_9	Predisposing factors		BiologicalEnvironment, EconomicEnvironment	(BiologicalEnv:) MixedFarmingFarm, ColonizationZone, EcologicalSite; (EconomicEnv:) PersistentPoverty, PublicHealthSystem, DrugRelatedTRIPS	(EcologicalSite:) ClimateChangeAffectedZone, HabitatDestructionZone

Appendix B

Input for Mace4 model searcher and Prover9 theorem prover

Version: Mace 4 (http://www.cs.unm.edu/∼mccune/prover9/), June-2007 (LADR0607-win).

B.1 Satisfiable input

```
formulas(sos).

% framework elements defs.
all x (Df(x) <-> exists y exists z (DF(x,y) & (D(x) -> CN(x)) & granulates(x,z))).
all x (Ds(x) <-> exists y ((D(x) -> (PT(x) | U(x))) & DF(x,y))).
all x (GP(x) <-> exists w exists y exists z exists p (DF(x,y) & RC(x,z) & C(z) &
     RE(x,w) & hasgranulation(x,p))).
all x (GL(x) <-> exists v exists w exists y exists z (DF(x,y) & GP(w) & RE(x,w) &
     C(z) & RC(w,z) & R(v) & hasvalue(z,v))).
all x (( C(x) -> exists y exists z (Prop(y) & Prop(z) & -(y = z) & -Q(y) & -Q(z) &
     CP(x,y) & CP(x,z))) | ( C(x) -> exists y exists z (Prop(y) & Q(z) & -(y = z) &
     CP(x,y) & CP(x,z)))).
all x (Q(x) -> Prop(x)).

% TOG relations defs
all x all y (RL(x,y) <-> sppartof(x,y) & GL(x) & GL(y) & -(x=y)).
all x all y (RLinv(x,y) <-> hassppart(x,y) & GL(x) & GL(y) & -(x=y)).
all x all y (RE(x,y) <-> ppartof(x,y) & ((GL(x) & GP(y)) | (GP(x) & Df(y)))).
all x all y (REinv(x,y) <-> hasppart(x,y) & ((GP(x) & GL(y)) | (Df(x) & GP(y)))).

% Granulation relations
all x all y (ppartof(x,y) <-> partof(x,y) & -partof(y,x)).
all x all y (involvedin(x,y) <-> ppartof(x, y) & PD(x) & PD(y)).
all x all y exists z exists w (containedin(x,y) <-> ppartof(x,y) & V(x) & V(y) &
     has3D(z,x) & has3D(w,y) & ED(z) & ED(w)).
all x all y (participatesin(x, y) <-> mpartof(x,y) & ED(x) & PD(y)).
all x all y (memberof(x, y) <-> mpartof(x, y) & (POB(x) | SOB(x)) & SOB(y)).
all x all y (sppartof(x,y) <-> ppartof(x,y) & ED(x) & ED(y)).
all x all y (hasppart(x, y) <-> haspart(x, y) & -(haspart(y, x))).
all x all y (involves(x, y) <-> hasppart(x, y) & PD(x) & PD(y)).
all x all y exists z exists w (contains(x, y) <-> hasppart(x, y) & V(x) & V(y) &
     has3D(z, x) & has3D(w, y) & ED(z) & ED(w)).
all x all y (hasparticipant(x, y) <-> hasmpart(x, y) & ED(y) & PD(x)).
all x all y (hasmember(x, y) <-> hasmpart(x, y) & (POB(y) | SOB(y)) & SOB(x)).
all x all y (hassppart(x,y) <-> hasppart(x,y) & ED(x) & ED(y)).
% all x all y all w exists i exists j (indist(x,y) <-> isoflevel(x,j) &
     isoflevel(y,j) & RL(j,i) & isoflevel(w,i) & PT(x) & PT(y) & PT(w) &
     (((participatesin(x,w) & participatesin(y,w)) | (memberof(x,w) &
     memberof(y,w)) | (ppartof(x,w) & ppartof(y,w)) |  (involvedin(x,w) &
     involvedin(y,w)) | (containedin(x,w) & containedin(y,w)) | ((PT(x) -> PT(w))
     & (PT(y) -> PT(w)))) | exists a exists b exists c ((V(a) & V(b) & V(c)
     & hasvalue(x, a) & hasvalue(y,b) & hasvalue(w,c) & <(a,c) & <(b,c))))).
% GR is the reified version of the binary GR1
all x all y (varphiindist(x, y) <-> indist(x,y) & varphi(x,z) & varphi(y, z) &
     (varphi(x,y) -> GR1(x,y))).
all x all y (overlap(x,y) <-> exists z (partof(z,x) & partof(z,y))).
all x all y (poverlap(x,y) <-> overcross(x,y) & -overcross(y,x)).
all x all y (overcross(x,y) <-> overlap(x,y) & -partof(x,y)).

% Taxonomy of types of granularity
all x (sG(x) -> cG(x)).
all x (nG(x) -> cG(x)).
all x (sgG(x) -> sG(x)).
all x (saG(x) -> sG(x)).
all x (sgsG(x) -> sgG(x)).
all x (sgrG(x) -> sG(x)).
all x (samG(x) -> saG(x)).
all x (saoG(x) -> saG(x)).
all x (nrG(x) -> nG(x)).
all x (nfG(x) -> nG(x)).
all x (naG(x) -> nG(x)).
all x (nacG(x) -> naG(x)).
all x (nasG(x) -> naG(x)).
all x (sG(x) -> -nG(x)).
all x (sgG(x) -> -saG(x)).
all x (sgsG(x) -> -sgrG(x)).
all x (nrG(x) -> -nfG(x)).
all x (nrG(x) -> -naG(x)).
all x (nfG(x) -> -naG(x)).
all x (nacG(x) -> -nasG(x)).
all x (cG(x) -> sG(x) | nG(x)).
all x (sG(x) -> sgG(x) | saG(x)).
all x (sgG(x) -> sgsG(x) | sgrG(x)).
all x (saG(x) -> samG(x) | saoG(x)).
all x (nG(x) -> nrG(x) | nfG(x) | naG(x)).
all x (naG(x) -> nacG(x) | nasG(x)).

% Constraints on relations, parthood and proper parthood.
all x partof(x,x).
all x all y all z (partof(x,y) & partof(y,z) -> partof(x,z)).
all x all y (partof(x,y) & partof(y,x) -> x=y).
all x -ppartof(x,x).
all x all y (ppartof(x,y) -> -ppartof(y,x)).
all x all y all z (ppartof(x,y) & ppartof(y,z) -> ppartof(x,z)).
all x all y (partof(x,y) <-> haspart(y,x)).
all x all y (ppartof(x,y) <-> hasppart(y,x)).
all x all y (sppartof(x,y) -> ppartof(x,y)).
all x all y (hassppart(x,y) -> hasppart(x,y)).
all x -RL(x,x).
all x -RLinv(x,x).

% Granulation relations
all x all y (containedin(x,y) -> ppartof(x,y)).
all x all y (contains(x, y) <-> containedin(y, x)).
all x all y (involvedin(x, y) -> ppartof(x, y)).
all x all y (involvedin(x, y) <-> involves(y, x)).
all x all y (participatesin(x, y) <-> hasparticipant(y, x)).
all x all y (memberof(x, y) <-> hasmember(y, x)).
all x all y (ppartof(x, y) -> GR1(x, y)).
all x all y (memberof(x, y) -> GR1(x, y)).
all x all y (participatesin(x, y) -> GR1(x, y)).
all x all y (isa(x, y) -> GR1(x, y)).
all x all y (containedin(x, y) -> -involvedin(x, y)).
all x all y (ppartof(x, y) -> -memberof(x, y)).
all x all y (ppartof(x, y) -> -participatesin(x, y)).
all x all y (memberof(x, y) -> -participatesin(x, y)).

% Other relations
all x all y (sim(x,y) -> -varphiindist(x,y)).
all x all y (indist(x,y) -> equiv(x,y)).
all x indist(x,x).
all x all y (indist(x,y) -> indist(y,x)).
all x all y all z (indist(x,y) & indist(y,z) -> indist(x,z)).
all x sim(x,x).
all x all y (sim(x,y) -> sim(y,x)).
all x all y all z all w (isoflevel(x,w) & isoflevel(y,w) & isoflevel(z,w) ->
     (sim(x,y) & sim(y,z) -> sim(x,z))).
all x equiv(x,x).
all x all y (equiv(x,y) -> equiv(y,x)).
all x all y all z (equiv(x,y) & equiv(y,z) -> equiv(x,z)).
all x all y (hasvalue(x, y) -> P(x) & V(y)).
all x (Q(x) -> exists y (hasvalue(x, y) & V(y))).
all x all y (hasvalue(x, y) -> exists z (hasvalue(z, y) & C(z))).
all x all y (granulates(x, y) -> Df(x) & Ds(y)).
all x (Df(x) -> exists y granulates(x, y)).
all x all y all z (granulates(x,y) & granulates(x, z) -> y=z).
all x all y (isoflevel(x,y) -> (PT(x) | U(x)) & GL(y)).
all x all y (CP(x,y) -> C(x) & Prop(y)).
all x (C(x) -> exists y exists z (CP(x,y) & CP(x,z) & -(y=z))).

% TOG relations
all x all y (RE(x,y) -> ppartof(x,y)).
all x all y (REinv(x,y) -> hasppart(x,y)).
all x all y all z (RE(x, y) & RE(y, z) & GL(x) & GP(y) & Df(z) -> RE(x, z)).
all x all y all z (REinv(x,y) & REinv(y,z) & GL(z) & GP(y) & Df(x) -> REinv(x,z)).
all x all y all z (RL(x, y) &  RL(y, z) ->  RL(x,z)).
% adding acyclicity for a hierarchy with 3 levels
all x all y all z (RL(x, y) &  RL(y, z) -> -(x = y) & -(x = z) & -(y = z)).
all x (GL(x) -> exists y (RL(x, y))).
all x all y all z (RL(x,y) & RL(x, z) -> y=z).
all x all y all z (RLinv(x, y) &  RLinv(y, z) ->  RLinv(x, z)).
all x (GL(x) -> exists y (RLinv(x,y))).
all x all y all z (RLinv(x,y) & RLinv(x, z) -> y=z).
all x all y all z (participatesin(x, y) & participatesin(y,z) ->
     -participatesin(x, z)).
all x all y (RP(x, y) -> GP(x) & GP(y) & -(x=y)).
all x -RP(x,x).
all x all y (RP(x, y) -> RP(y,x)).
all x all y (overcross(x,y) & GL(x) & GL(y) & -(x=y) ->
     exists v exists w (RE(x, v) & RE(y, w) & -(v=w))).
all w all x all y all z (overcross(w,x) & GL(w) & GL(x) & GP(y) &
     GP(z) & RE(w,y) & RE(x,z) -> overcross(y,z)).
all x all p (hasgranulation(x, p) -> GP(x) & TG(p)).
all x (GP(x) -> exists p hasgranulation(x,p)).
all x all y all z ( hasgranulation(x,y) & hasgranulation(x,z) -> y=z).
all x (adheresto(x,p) -> GL(x) & TG(p)).
all x (GL(x) -> exists p adheresto(x,p)).
all x all y (GP(y) & GL(x) & RE(x,y) -> exists p exists q (
     (hasgranulation(x,p) <-> adheresto(y,p)) -> p=q )).

% Constraints and characteristics, TOG elements
all x (D(x) -> (Df(x) | Ds(x))).
all x (Df(x) -> D(x) & CN(x)).
all x (Ds(x) -> D(x)).
all x (CN(x) -> exists y DF(x, y)).
all x (Ds(x) -> PT(x) | U(x)).
all x (GP(x) -> CN(x)).
all x (GL(x) -> CN(x)).
all x (Df(x) -> exists y granulates(x, y)).
all x (Df(x) & (Ds(y) -> PED(y)) & granulates(x,y) -> OD(x,y)).
all x all y all z (DF(x,y) & DF(x,z) -> y=z).
% because one cannot do U(\phi), the disjointness is added:
all x (PT(x) -> -U(x)).
% a bit more from dolce.
all x (CN(x) -> ED(x)).
all x (ED(x) -> PT(x)).
all x (PD(x) -> PT(x)).
all x (ED(x) -> -PD(x)).
all x (ED(x) -> -AB(x)).
all x (PD(x) -> -AB(x)).
all x (V(x) -> AB(x)).

% Perspectives and their criteria
all x all y (RC(x,y) -> GP(x) & C(y)).
all x (GP(x) -> exists y RC(x,y)).
all x all y all z (RC(x, y) & RC(x, z) -> y=z).
all x (C(x) -> exists y RC(x,y)).
all x all y all z all p all q (RC(x,y) & RC(z, y) & hasgranulation(x,p) &
     hasgranulation(z,q) & -(x=z) -> -(p=q)).
all x (GP(x) -> exists y exists p (RC(x,y) & hasgranulation(x,p))).
all x all y exists z exists p ((C(x) -> Q(y)) & RC(z, x) &
     hasgranulation(z,p) -> sG(p)).
all x all y all z ((RC(x, y) & GP(x) & C(y) & RE(z, x) & GL(z)) -> RC1(z, y)).
all x all y (RC1(x,y) -> GL(x) & C(y)).

% Constraints on elements contained in another
all x (GP(x) ->  exists y exists z (REinv(x,y) & GL(y) & GL(z) & -(y=z) &
     REinv(x,z))).
all x (Df(x) ->  exists y (REinv(x,y) & GP(y))).
all x (GL(x) -> exists y (RE(x,y) & GP(y))).
all x all y all z (RE(x,y) & GL(x) & RE(x,z) -> y=z).
all x (GP(x) -> exists y (RE(x, y) &  Df(y))).
all x all y all z (RE(x,y) & GP(x) & RE(x,z) -> y=z).
all x all p all q (hasgranulation(x,p) & GP(x) & nrG(p) & haspermitted(p,q) &
     GR(q) & exists a exists b ((GR1(a,b) <-> participatesin(a,b)) | (GR1(a,b) <->
     memberof(a,b))) -> exists z exists w exists v (GL(z) & GL(w) & GL(v) &
     -(z=w) & REinv(x,z) & REinv(x,w) & REinv(x,v) -> (z=v) | (w=v))).

% Further constraints for different types of granularity subsumed by cG
all x all y (usesGR(x,y) -> TG(x) & GR(y)).
all x (cG(x) -> TG(x)).
all p all q (haspermitted(p,q) -> nrG(p) & GR(q)).
all p (nrG(p) -> exists q (haspermitted(p,q))).
all p exists q exists r (haspermitted(p,q) & haspermitted(p,r) -> q=r).
all p (nfG(p) -> exists q exists r (usesGR(p,q) & usesGR(p,r) & -(q=r))).
all p all q (usesGR(p,q) & nfG(p) & exists a exists b ((GR1(a,b) <->
     participatesin(a,b)) -> all z exists w exists u (GP(z) & GL(w) & GL(u) &
     -(w=u) & RE(w,z) & RE(u,z) & (exists v (GL(v) & RE(v,z) -> (w=v) | (u=v)))))).
all p all q all r all x all y (nacG(p) & adheresto(x,p) & adheresto(y,p) & RL(y,x)
     & isoflevel(q,x) & isoflevel(r,y) -> (U(r) -> -U(q)) ).
all p all q (usesGR(p,q) & nacG(p) -> exists a exists b exists x exists y (
     (GR1(a,b) <-> memberof(a,b)) & isoflevel(a,x) & isoflevel(b,y) & RL(a,b))).
all a all b exists x exists y ((GR1(a,b) <-> memberof(a,b)) & isoflevel(a,x) &
     isoflevel(b,y) & RL(x,y) -> all z exists w exists u (GP(z) & GL(w) & GL(u) &
     -(w=u) & RE(w,z) & RE(u,z) & (exists v (GL(v) & RE(v,z) -> (w=v) | (u=v)) ))).
all p (nasG(p) -> all q exists a exists b (usesGR(p,q) & (GR1(a,b) <-> isa(a,b)))).
all p all q all r all x all y (nasG(p) & adheresto(x,p) & adheresto(y,p) & RL(y,x)
     & isoflevel(q,x) & isoflevel(r,y) -> (PT(r) -> PT(q))).
```

```
all x all p all t (conv(x,p,t) -> GL(x) & sG(p) & F(t)).
all x all p (adheresto(x,p) & sG(p) -> exists t conv(x,p,t)).
all x (GL(x) -> (all t all u all v ( conv(x,p,t) & conv(x,p,u) & conv(x,p,v) ->
     (t=u) | (t=v) | (u=v)))).
all z all p (sG(p) & GP(z) & hasgranulation(z,p) -> (all x all y exists a exists b
     exists c exists d ((C(x) & RC(z,x) & GL(c) & GL(d) & -(c=d) &
     RE(c,z) & RE(d,z) & hasvalue(x,a) & hasvalue(y,b) & R(a) & R(b) &
     -(a=b) & RL(c,d) & RLinv(d,c) ) -> >(b,a)))).

end_of_list.
formulas(goals).
end_of_list.
```

The final statistics are as follows:

```
============================== Mace4 ====================
Mace4 (32) version June-2007, June 2007.
Process 3240 was started by mkeet on ubz6169xp,
Sun Oct 14 16:57:55 2007
The command was "bin/mace4 -f tog08.in".
============================== end of head ================

============================== STATISTICS ===============

For domain size 2.

Current CPU time: 0.00 seconds (total CPU time: 0.12 seconds).
Ground clauses: seen=1776, kept=1379.
Selections=363, assignments=363, propagations=206, current_models=1.
Rewrite_terms=3040, rewrite_bools=1979, indexes=144.
Rules_from_neg_clauses=2, cross_offs=2.

============================== end of statistics =====================

User_CPU=0.12, System_CPU=0.01, Wall_clock=0.

Exiting with 1 model.
```

B.2 Statistics domain size and computational limits

Statistics for domain size 3, 200MB memory

```
============================== STATISTICS ============================
For domain size 3.
Current CPU time: 0.00 seconds (total CPU time: 0.19 seconds).
Ground clauses: seen=6039, kept=4509.
Selections=1006, assignments=1009, propagations=429, current_models=1.
Rewrite_terms=15813, rewrite_bools=6746, indexes=194.
Rules_from_neg_clauses=0, cross_offs=3.
============================== end of statistics =====================
```

Statistics for domain size 4, 200MB memory

```
============================== STATISTICS ============================
For domain size 4.
Current CPU time: 0.00 seconds (total CPU time: 0.41 seconds).
Ground clauses: seen=17272, kept=12498.
Selections=2313, assignments=2317, propagations=744, current_models=1.
Rewrite_terms=57764, rewrite_bools=19378, indexes=254.
Rules_from_neg_clauses=0, cross_offs=4.
============================== end of statistics =====================
```

Statistics for domain size 5, 200MB memory

```
============================== STATISTICS ============================
For domain size 5.
Current CPU time: 0.00 seconds (total CPU time: 1.16 seconds).
Ground clauses: seen=42915, kept=30380.
Selections=4676, assignments=4681, propagations=1155, current_models=1.
Rewrite_terms=165045, rewrite_bools=48404, indexes=324.
Rules_from_neg_clauses=0, cross_offs=5.
============================== end of statistics =====================
```

Statistics for domain size 6, 200MB memory

```
============================== STATISTICS ============================
For domain size 6.
Current CPU time: 0.00 seconds (total CPU time: 2.83 seconds).
Ground clauses: seen=94824, kept=66141.
Selections=8581, assignments=8587, propagations=1668, current_models=1.
Rewrite_terms=396630, rewrite_bools=107483, indexes=404.
Rules_from_neg_clauses=0, cross_offs=6.
============================== end of statistics =====================
```

Statistics for domain size 7, 200MB memory

```
============================== STATISTICS ============================
For domain size 7.
Current CPU time: 0.00 seconds (total CPU time: 6.78 seconds).
Ground clauses: seen=190351, kept=131439.
Selections=14610, assignments=14617, propagations=2289, current_models=1.
Rewrite_terms=840077, rewrite_bools=216676, indexes=494.
Rules_from_neg_clauses=0, cross_offs=7.
============================== end of statistics =====================
```

Statistics for domain size 8, 200MB memory with memory error

```
============================== STATISTICS ============================
For domain size 8.
Current CPU time: 0.00 seconds (total CPU time: 16.42 seconds).
Ground clauses: seen=353424, kept=242324.
Selections=21426, assignments=21434, propagations=2904, current_models=0.
Rewrite_terms=1617288, rewrite_bools=402803, indexes=594.
Rules_from_neg_clauses=0, cross_offs=8.
============================== end of statistics =====================
```

Statistics for domain size 9, 200MB memory with memory error

```
============================== STATISTICS ============================
For domain size 9.
Current CPU time: 0.00 seconds (total CPU time: 12.52 seconds).
Ground clauses: seen=615627, kept=419958.
Selections=0, assignments=0, propagations=1251, current_models=0.
Rewrite_terms=0, rewrite_bools=476640, indexes=0.
Rules_from_neg_clauses=0, cross_offs=0.
============================== end of statistics =====================
```

Statistics for domain size 8, 500MB memory

```
============================== STATISTICS ============================
For domain size 8.
Current CPU time: 0.00 seconds (total CPU time: 17.16 seconds).
Ground clauses: seen=353424, kept=242324.
Selections=23441, assignments=23449, propagations=3024, current_models=1.
Rewrite_terms=1617288, rewrite_bools=403716, indexes=594.
Rules_from_neg_clauses=0, cross_offs=8.
============================== end of statistics =====================
```

Statistics for domain size 9, 500MB memory (used: about 390MB). Process killed after 2h 31min.

Statistics for domain size 10, 500MB memory with memory error.

```
============================== STATISTICS ============================
For domain size 10.
Current CPU time: 0.00 seconds (total CPU time: 35.97 seconds).
Ground clauses: seen=1017280, kept=691335.
Selections=22634, assignments=22642, propagations=4238, current_models=0.
Rewrite_terms=2517614, rewrite_bools=1149879, indexes=626.
Rules_from_neg_clauses=0, cross_offs=8.
============================== end of statistics =====================
```

Statistics for domain size 10, 700MB memory (used: about 570MB). Process killed after 40mins.

B.3 Excerpt computed proofs

Computed proof of *Theorem 3.4* with Prover9.

```
============================== PROOF =================================

% Proof 1 at 2.11 (+ 0.03) seconds.
% Length of proof is 24.
% Level of proof is 6.
% Maximum clause weight is 4.
% Given clauses 0.

29 (all x (sgG(x) -> sG(x))) # label(non_clause).  [assumption].
30 (all x (saG(x) -> sG(x))) # label(non_clause).  [assumption].
36 (all x (nfG(x) -> nG(x))) # label(non_clause).  [assumption].
40 (all x (sG(x) -> -nG(x))) # label(non_clause).  [assumption].
48 (all x (sG(x) -> sgG(x) | saG(x))) # label(non_clause).  [assumption].
162 (all p all q (usesGR(p,q) & nfG(p) & (exists a exists b ((GR1(a,b) <->
      participatesin(a,b)) -> (all z exists w exists u (GP(z) & GL(w) & GL(u) &
      w != u & RE(w,z) & RE(u,z) & (exists v (GL(v) & RE(v,z) -> w = v |
      u = v)))))))) # label(non_clause).  [assumption].
168 (all x all p all t (conv(x,p,t) -> GL(x) & sG(p) & F(t)))
      # label(non_clause).  [assumption].
171 (all x (GL(x) -> (all t all u all v (conv(x,p,t) & conv(x,p,u) &
      conv(x,p,v) -> t = u | t = v | u = v)))) # label(non_clause)
      # label(goal).  [goal].
371 -sgG(x) | sG(x).  [clausify(29)].
373 -saG(x) | sG(x).  [clausify(30)].
375 -sG(x) | -nG(x).  [clausify(40)].
377 -sG(x) | sgG(x) | saG(x).  [clausify(48)].
379 -conv(x,y,z) | sG(y).  [clausify(168)].
415 nfG(x).  [clausify(162)].
416 -nfG(x) | nG(x).  [clausify(36)].
430 -nG(x) | -sgG(x).  [resolve(375,a,371,b)].
432 -conv(x,y,z) | sgG(y) | saG(y).  [resolve(379,b,377,a)].
552 -nG(x) | -saG(x).  [resolve(375,a,373,b)].
562 -conv(x,y,z) | saG(y) | -nG(y).  [resolve(432,b,430,b)].
721 conv(c1,p,c4).  [deny(171)].
1058 nG(x).  [resolve(415,a,416,a)].
1102 -conv(x,y,z) | -nG(y) | -nG(y).  [resolve(562,b,552,b)].
1103 -conv(x,y,z).  [copy(1102),merge(c),unit_del(b,1058)].
1104 $F.  [resolve(1103,a,721,a)].

============================== end of proof ==========================
```

Computed proof of *Theorem 3.7* with Prover9.

```
============================== PROOF =================================

% Proof 1 at 2.59 (+ 0.11) seconds.
% Length of proof is 23.
% Level of proof is 6.
% Maximum clause weight is 17.
% Given clauses 689.

9 (all x all y (RE(x,y) <-> ppartof(x,y) & (GL(x) & GP(y) | GP(x) & Df(y))))
      # label(non_clause).  [assumption].
56 (all x -ppartof(x,x)) # label(non_clause).  [assumption].
101 (all x all y all z (RE(x,y) & RE(y,z) & GL(x) & GP(y) & Df(z) -> RE(x,z)))
      # label(non_clause).  [assumption].
151 (all x all y all z (RE(x,y) & GL(x) & RE(x,z) -> y = z))
      # label(non_clause).  [assumption].
152 (all x (GP(x) -> (exists y (RE(x,y) & Df(y))))) # label(non_clause).  [assumption].
171 (all w all x all y all z (overcross(w,x) & GL(w) & GL(x) & GP(y) & GP(z) &
      RE(w,y) & RE(x,z) -> overcross(y,z))) # label(non_clause) # label(goal).  [goal].
185 -RE(x,y) | -RE(y,z) | -GL(x) | -GP(y) | -Df(z) | RE(x,z).  [clausify(101)].
195 -GP(x) | Df(f46(x)).  [clausify(152)].
585 -RE(x,y) | ppartof(x,y).  [clausify(9)].
607 -ppartof(x,x).  [clausify(56)].
680 -RE(x,y) | -GL(x) | -RE(x,z) | z = y.  [clausify(151)].
681 -GP(x) | RE(x,f46(x)).  [clausify(152)].
719 GL(c1).  [deny(171)].
721 GP(c3).  [deny(171)].
723 RE(c1,c3).  [deny(171)].
794 -GP(x) | -RE(y,z) | -RE(z,f46(x)) | -GL(y) | -GP(z) | RE(y,f46(x)).
      [resolve(195,b,185,e)].
1115 -GP(x) | -RE(y,x) | -RE(x,f46(x)) | -GL(y) | RE(y,f46(x)).  [factor(794,a,e)].
1176 RE(c3,f46(c3)).  [resolve(721,a,681,a)].
1205 -RE(c1,x) | c3 = x.  [resolve(723,a,680,c),unit_del(b,719)].
1376 RE(c1,f46(c3)).  [resolve(1115,b,723,a),unit_del(a,721),
      unit_del(b,1176),unit_del(c,719)].
2239 ppartof(c3,f46(c3)).  [resolve(1176,a,585,a)].
2692 f46(c3) = c3.  [resolve(1376,a,1205,a),flip(a)].
2748 $F.  [back_rewrite(2239),rewrite([2692(3)]),unit_del(a,607)].

============================== end of proof ==========================
```

Appendix C

Introduction to Description Logics, $\mathcal{DLR}_{ifd}$, and $\mathcal{DLR}_{\mu}$

It was shown in §5.5.1 that a combination of $\mathcal{DLR}_{ifd}$ and $\mathcal{DLR}_{\mu}$ into a, hypothetical, "$\mathcal{DLR}_{\mu ifd}$" will have most of the constructors required to transform the TOG into a DL version. In this appendix it will be shown that in the presence of an ABox, two restrictions have to be put on $\mathcal{DLR}_{\mu ifd}$ in order to remain within ExpTime completeness for satisfiability and logical implication.

The remainder of the appendix is organised as follows. First, it will be argued in §C.1 that transforming the TOG into a DL language is the most efficient strategy toward computational implementations. Second, a brief introduction into Description Logics, $\mathcal{DLR}_{ifd}$, and $\mathcal{DLR}_{\mu}$ will be given (§C.2), which functions as background information for the assessment of combining $\mathcal{DLR}_{ifd}$ and $\mathcal{DLR}_{\mu}$ in §5.5.1.

C.1 Introduction

Several options to transform the TOG toward usage in information systems were examined in §5.5. To implement most of the TOG with widest applicability—hence, maintaining the largest possible common foundational framework for granularity across implementations—we will have to resort to a Description Logic. There are several reasons for this choice:

- ⋆ Transforming the TOG into a DL representation allows seamless propagation to both ontology languages, such as OWL, and to conceptual modelling languages, such as EER, UML, and ORM. The DL $\mathcal{DLR}$ and its variants were developed for automated reasoning over formal conceptual data models and there are several mappings to EER, UML class diagrams, and most of ORM2 already[1], which are summarised in *Figure C.1* and in Keet (2008a).
- ⋆ Given the previous point, then by virtue of adding granularity to a *formal* ontology or conceptual modelling language, one may achieve automated propagation of granularity to information systems that are built based on a formal conceptual data model. Likewise, it will then make the TOG usable with formal approaches to data integration and semantic connectivity of ontologies across levels of granularity in addition to the current methodologies.
- ⋆ DL serves as higher level of abstraction for Datalog and SQL-based information systems and as knowledge-based system it combines type-level (TBox) as well as instance-level (ABox) data, which is a highly desirably combination from the perspective of domain experts.
- ⋆ Transforming the TOG in FOL to a DL puts the language to represent the theory within the decidable FOL-fragment, which will result in better performance of automated reasoning over the granularity framework, such as checking consistency of a particular domain granularity framework on compliance with the TOG constraints.
- ⋆ Adoption of the TOG by domain experts, conceptual data modellers, and ontology developers will be greatly facilitated when it is offered in a user-friendly graphical software tool. There are several DL-based conceptual modelling and ontology editors that would require only a minor extension for modelling granularity compared to the efforts of building a tool from scratch. Such an extension

[1]The (partial) mapping from ORM/ORM2 to $\mathcal{DLR}_{ifd}$ has been phased out from the appendix due to space limitations; it can be found in (Keet, 2007a,b).

to a DL-based CASE or ODE tool ensures it is within a setting already known to domain experts and modellers, yet permitting that the TOG keeps its foundation of an unambiguous semantics. Thus, by mapping it to a DL, the most comprehensive range of implementation options can be covered with, relatively, the least amount of effort. The assessment in §5.5.1 made clear that, regarding the language constructors, the most suitable DL is $\mathcal{DLR}_{\mu ifd}$. Observe that this choice still permits a further transformation or simplification to more widely used DL languages, such as $\mathcal{SROIQ}$ (Horrocks *et al.*, 2006), which the basis for the new OWL v1.1 (OWL, 2007) and a $DL\text{-}Lite$ (Calvanese *et al.*, 2007).

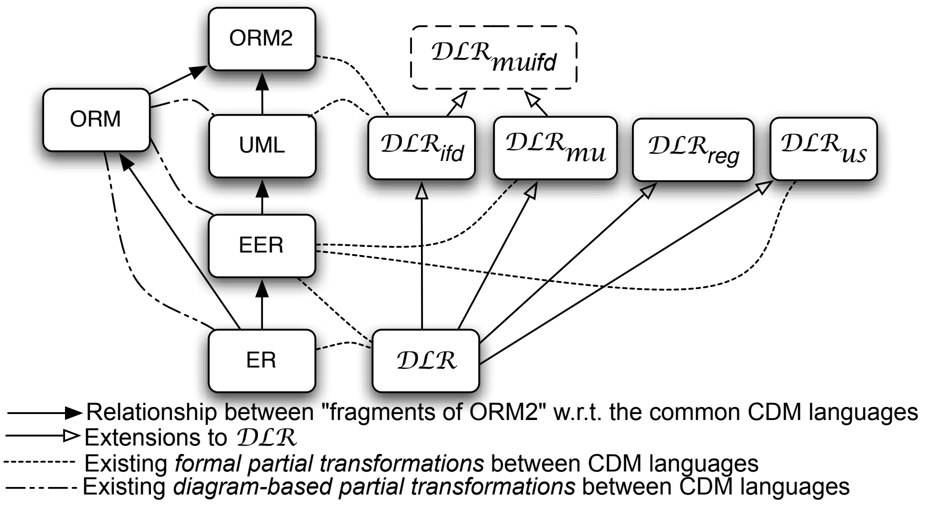

ORM2
ORM
UML
EER
ER
$\mathcal{DLR}_{muifd}$
$\mathcal{DLR}_{ifd}$
$\mathcal{DLR}_{mu}$
$\mathcal{DLR}_{reg}$
$\mathcal{DLR}_{us}$
$\mathcal{DLR}$
Relationship between "fragments of ORM2" w.r.t. the common CDM languages
Extensions to $\mathcal{DLR}$
Existing *formal partial transformations* between CDM languages
Existing *diagram-based partial transformations* between CDM languages

Figure C.1: Relations between several formal conceptual modelling languages.

C.2 Description logic languages

C.2.1 The basics

Description Logic languages are decidable fragments of first order logic and are used for logic-based knowledge representation, such as formal conceptual modelling and ontology development; e.g., the Web Ontology Language OWL 1.1, OWL-DL, and OWL-Lite are based on DL languages. The appropriate DL language to represent the information of the Universe of Discourse depends on requirements what the user wants to represent and what she wants to do with the knowledge base system. The basic ingredients of all DL languages are *concepts* and *roles*, where a DL-role is an n-ary predicate and $n \geq 2$, although in most DL languages $n = 2$. In addition, each DL language has several constructors, which varies among the DL languages to give greater or lesser expressivity and efficiency of automated reasoning over a logical theory. *Table C.1* lists some examples; consult the appendix of (Baader *et al.*, 2003) for a comprehensive list. More introductory information about DL can be found in Baader and Nutt (2003), and usages and extension in Baader *et al.* (2003).

DL knowledge bases are composed of the *Terminological Box* (TBox, $\mathcal{T}$), which contains axioms at the concept-level, and the *Assertional Box* (ABox, $\mathcal{A}$) that contains assertions about instances. We use a refinement of the notion of TBox to distinguish between concept hierarchies and role hierarchies (this distinction will be useful later) and define a DL-based ontology or conceptual data model as follows.

DEFINITION C.1 (DL-based ontology or conceptual data model). *A DL-based ontology or conceptual data model is a pair $\Sigma = (\mathcal{T}, \mathcal{R})$ where $\mathcal{T}$ is a set of terminological axioms of the form $C \sqsubseteq D$ (general concept inclusion axiom), and $\mathcal{R}$ is a set of role axioms of the form $R \sqsubseteq S$ (subrole axiom) and $R \sqsubseteq C_1 \times C_2$ (Domain & Range axiom).*

The Domain & Range axioms in *Definition C.1* are a shortcut for $\exists R \sqsubseteq C_1$, and $\exists R^- \sqsubseteq C_2$, with C_1, C_2 concepts. The set of role axioms $\mathcal{R}$ is also called the *Role box* (RBox). The formal semantics of each DL language follows the usual notion of interpretation, $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where the interpretation function $\cdot^{\mathcal{I}}$ assigns to each concept C a subset $C^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$ and to each relation R of arity n a subset $R^{\mathcal{I}}$ of $(\Delta^{\mathcal{I}})^n$. Thus, an interpretation $\mathcal{I}$ satisfies $C \sqsubseteq D$ iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, $R \sqsubseteq S$ iff $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$, and $R \sqsubseteq C_1 \times C_2$ iff $R^{\mathcal{I}} \subseteq C_1^{\mathcal{I}} \times C_2^{\mathcal{I}}$.

Σ is satisfiable if there is an interpretation $\mathcal{I}$ which satisfies every axiom in Σ; in this case $\mathcal{I}$ is called a model of Σ. Σ logically implies an axiom α, written as $\Sigma \models \alpha$, if α is satisfied by every model of Σ. Given a Σ, a concept C (role R) is satisfiable if there exists a model $\mathcal{I}$ of Σ such that $C^{\mathcal{I}} \neq \emptyset$ ($R^{\mathcal{I}} \neq \emptyset \times \emptyset$), i.e. $\Sigma \not\models C \sqsubseteq \bot$ ($\Sigma \not\models \exists R \sqsubseteq \bot$). For a DL ABox $\mathcal{A}$, an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ is extended to also map individual names, *i.e.*, each individual a is mapped to an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ and, assuming the unique name assumption, if a, b are distinct names then $a^{\mathcal{I}} \neq b^{\mathcal{I}}$. Further, $\mathcal{I}$ satisfies $C(a)$ if $a^{\mathcal{I}} \in C^{\mathcal{I}}$ and satisfies $R(a, b)$ if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$; common alternative notations are $a : C$ and $a, b : R$, respectively. We now can define a DL-knowledge base $\mathcal{K}$ as follows.

DEFINITION C.2 (DL knowledge base). *A DL knowledge base is a triple $\mathcal{K} = (\mathcal{T}, \mathcal{R}, \mathcal{A})$, where $\mathcal{T}$ is a set of terminological axioms of the form $C \sqsubseteq D$ (general concept inclusion axiom), $\mathcal{R}$ is a set of role axioms of the form $R \sqsubseteq S$ (subrole axiom) and $R \sqsubseteq C_1 \times C_2$ (Domain & Range axiom), and $\mathcal{A}$ is a set of assertional axioms of the form $C(a)$ and $R(a, b)$ (instantiation of a concept and role, respectively).*

$\mathcal{K}$ is satisfiable if there is an interpretation $\mathcal{I}$ that satisfies every axiom in $\mathcal{K}$; in this case $\mathcal{I}$ is called a model of $\mathcal{K}$. $\mathcal{K}$ logically implies an axiom α, written as $\mathcal{K} \models \alpha$, if α is satisfied by every model of $\mathcal{K}$.

There are two points to take into account with definitions C.1 and C.2. First, the distinguishing characteristic between an ontology and knowledge base is the absence/presence of the ABox. Second, multiple extensions to the 'simple' $\mathcal{K}$ and Σ and are possible, as we will see later in this appendix.

Table C.1: Non-exhaustive list of concept and role constructors and terminological and assertional axioms, respectively, DL syntax, and their semantics; C is a concept and R a role.

Name	DL syntax	Semantics
Top concept	$\top$	$\Delta^{\mathcal{I}}$
Bottom concept	$\bot$	$\emptyset$
Concept	C	$C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
Concept disjunction	$C_1 \sqcap C_2$	$C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$
Concept conjunction	$C_1 \sqcup C_2$	$C_1^{\mathcal{I}} \cup C_2^{\mathcal{I}}$
Concept negation	$\neg C$	$\Delta^{\mathcal{I}} \backslash C^{\mathcal{I}}$
Universal restriction	$\forall R.C$	$\{a \in \Delta^{\mathcal{I}} \vert \forall b.(a, b) \in R^{\mathcal{I}} \rightarrow b \in C^{\mathcal{I}}\}$
Existential restriction	$\exists R.C$	$\{a \in \Delta^{\mathcal{I}} \vert \exists b.(a, b) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
Role complement	$\neg R$	$\Delta \times \Delta \backslash R^{\mathcal{I}}$
Reflexive transitive closure	R^*	$\bigcup_{n \geq 0} (R^{\mathcal{I}})^n$
Concept inclusion	$C_1 \sqsubseteq C_2$	$C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$
Role inclusion	$R_1 \sqsubseteq R_2$	$R_1^{\mathcal{I}} \subseteq R_2^{\mathcal{I}}$
Equality of concepts	$C_1 \equiv C_2$	$C_1^{\mathcal{I}} = C_2^{\mathcal{I}}$
Equality of roles	$R_1 \equiv R_2$	$R_1^{\mathcal{I}} = R_2^{\mathcal{I}}$
Concept assertion	$C(a)$	$a^{\mathcal{I}} \in C^{\mathcal{I}}$
Role assertion	$R(a, b)$	$(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$

C.2.2 $\mathcal{DLR}_{ifd}$ and $\mathcal{DLR}_{\mu}$

The starting point: $\mathcal{DLR}$. The base language of the expressive DL family for conceptual data modeling, $\mathcal{DLR}$, is introduced first (Calvanese and De Giacomo, 2003), and subsequently the "$_{ifd}$" and "$_{\mu}$" extensions (Berardi *et al.*, 2005; Calvanese *et al.*, 1999, 2001a). Take atomic relations (P) and atomic concepts A as the basic elements of $\mathcal{DLR}$. We then can construct arbitrary relations with arity ≥ 2 and arbitrary concepts according to the following syntax:

$R \longrightarrow \top_n \mid P \mid (\$i/n : C) \mid \neg R \mid R_1 \sqcap R_2$

$C \longrightarrow \top_1 \mid A \mid \neg C \mid C_1 \sqcap C_2 \mid \exists[\$i]R \mid \leq k[\$i]R$

i denotes a component of a relation; if components are not named, then integer numbers between 1 and n_{max} are used, where n is the arity of the relation. k is a nonnegative integer for multiplicity (cardinality).

Only relations of the same arity can be combined to form expressions of type $R_1 \sqcap R_2$, and $i \leq n$, *i.e.*, the concepts and relations must be well-typed. The semantics of $\mathcal{DLR}$ is specified through the usual notion of interpretation, where $\mathcal{I}= (\Delta^\mathcal{I}, \cdot^\mathcal{I})$, and the interpretation function $\cdot^\mathcal{I}$ assigns to each concept C a subset $C^\mathcal{I}$ of $\Delta^\mathcal{I}$ and to each n-ary R a subset $R^\mathcal{I}$ of $(\Delta^\mathcal{I})^n$, s.t. the conditions are satisfied following *Table C.2*. $\top_1$ denotes the interpretation domain, $\top_n$ for $n \geq 1$ denotes a subset of the n-cartesian product of the domain, which covers all introduced n-ary relations; hence "$\neg$" on relations means difference rather than the complement. The $(\$i/n : C)$ denotes all tuples in $\top_n$ that have an instance of C as their i-th component. $\mathcal{DLR}$ is a proper generalization of $\mathcal{ALCQI}$, where the usual DL constructs can be re-expressed in $\mathcal{DLR}$ as (Calvanese and De Giacomo, 2003):

$\exists P.C$ as $\exists[\$1](P \sqcap (\$2/2 : C))$
$\exists P^-.C$ as $\exists[\$2](P \sqcap (\$1/2 : C))$
$\forall P.C$ as $\neg\exists[\$1](P \sqcap (\$2/2 : \neg C))$
$\forall P^-.C$ as $\neg\exists[\$2](P \sqcap (\$1/2 : \neg C))$
$\leq kP.C$ as $\leq k[\$1](P \sqcap (\$2/2 : C))$
$\leq kP^-.C$ as $\leq k\exists[\$2](P \sqcap (\$1/2 : C))$

The following abbreviations can be used:

- $C_1 \sqcup C_2$ for $\neg(\neg C_1 \sqcap \neg C_2)$
- $C_1 \Rightarrow C_2$ for $\neg C_1 \sqcup C_2$
- $(\geq k[i]R)$ for $\neg(\leq k-1[i]R)$
- $\exists[i]R$ for $(\geq 1[i]R)$
- $\forall[i]R$ for $\neg\exists[i]\neg R$
- $R_1 \sqcup R_2$ for $\neg(\neg R_1 \sqcap \neg R_2)$
- $(i/n : C)$ is abbreviated to $(i : C)$ where n is clear from the context

$\mathcal{DLR}$ with identification and functional dependency: $\mathcal{DLR}_{\mathit{ifd}}$. $\mathcal{DLR}_{\mathit{ifd}}$ is an extension of $\mathcal{DLR}$ with identification assertions on a concept C, which has the form

(**id** $C[i_1]R_1, ..., [i_h]R_h$)

where each R_j is a relation and each i_j denotes one component of R_j. Then, if a is an instance of C that is the i_j-th component of a tuple t_j of R_j, for $j \in \{1, ..., h\}$, and b is an instance of C that is the i_j-th component of a tuple s_j of R_j, for $j \in \{1, ..., h\}$, and for each j, t_j agrees with s_j in all components different from i_j, then a and b are the same object. In addition to the usual interpretation function given above, we have for the **id**:

- An interpretation $\mathcal{I}$ satisfies the assertion (**id** $C[i_1]R_1, ..., [i_h]R_h$) if for all $a, b \in C^\mathcal{I}$ and for all $t_1, s_1 \in R_1^\mathcal{I}$, ..., $t_h, s_h \in R_h^\mathcal{I}$ we have that:

$$\left.\begin{array}{r} a = t_1[i_1] = ... = t_h[i_h] \\ b = s_1[i_1] = ... = s_h[i_h] \\ t_j[i] = s_j[i], \text{ for } j \in \{1, ..., h\}, \text{ and for } i \neq j \end{array}\right\} \text{ implies } a = b$$

$\mathcal{DLR}_{\mathit{ifd}}$ also supports functional dependency assertions on a relation R to deal with operations, which has the form

(**fd** $R\, i_1, ..., i_h \rightarrow j$)

where $h \geq 2$, and $i_1, ..., i_h, j$ denote components of R, and semantics:

- An interpretation $\mathcal{I}$ satisfies the assertion (**fd** $R\, i_1, ..., i_h \rightarrow j$) if for all $t, s \in R^\mathcal{I}$, we have that

$$t[i_1] = s[i_1], ..., t[i_h] = s[i_h] \text{ implies } t_j = s_j$$

Last, there are notational variants

- Set difference for R, where the "$\dot\neg$" can be used to distinguish it from normal negation.
- dropping the "\$" before the i
- "$t[i]$" for the i-th component of tuple t, s.t. one can rewrite $(\$i/n : C)^\mathcal{I} = \{(d_1, ..., d_n) \in \top_n^\mathcal{I} | d_i \in C^\mathcal{I}\}$ with the previous point into $(i/n : C)^\mathcal{I} = \{t \in \top_n^\mathcal{I} | t[i] \in C^\mathcal{I}\}$
- Use $\sharp S$ to denote the cardinality of the set S, s.t. one can rewrite $(\leq k[\$i]R)^\mathcal{I} = \{d \in \Delta^\mathcal{I} || \{(d_1, ..., d_n) \in R_1^\mathcal{I} | d_i = d|\} \leq k\}$ with the second and third point into $(\leq k[i]R)^\mathcal{I} = \{d \in \Delta^\mathcal{I} | \sharp\{t \in R_1^\mathcal{I} | t[i] = d\} \leq k\}$

$\mathcal{DLR}$ with fixpoints: $\mathcal{DLR}_\mu$. Proceeding now to $\mathcal{DLR}_\mu$, it is different compared to $\mathcal{DLR}$ and $\mathcal{DLR}_{\mathit{ifd}}$ in that it has constructor for least (and greatest) fixpoint for recursive structures over single-inheritance

Table C.2: Semantic rules for $\mathcal{DLR}$ and $\mathcal{DLR}_{ifd}$.

$\top_n^{\mathcal{I}} \subseteq (\Delta^{\mathcal{I}})^n$	$(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
$P^{\mathcal{I}} \subseteq \top_n^{\mathcal{I}}$	$(C_1 \sqcap C_2)^{\mathcal{I}} = C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$
$(\neg R)^{\mathcal{I}} = \top_n^{\mathcal{I}} \setminus R^{\mathcal{I}}$	$(\$i/n : C)^{\mathcal{I}} = \{(d_1, ..., d_n) \in \top_n^{\mathcal{I}} \| d_i \in C^{\mathcal{I}}\}$
$(R_1 \sqcap R_2)^{\mathcal{I}} = R_1^{\mathcal{I}} \cap R_2^{\mathcal{I}}$	$(\exists[\$i]R)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \| \exists(d_1, ..., d_n) \in R^{\mathcal{I}}.d_i = d\}$
$\top_1^{\mathcal{I}} = \Delta^{\mathcal{I}}$	$(\leq k[\$i]R)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \|\| \{(d_1, ..., d_n) \in R_1^{\mathcal{I}} \| d_i = d\} \leq k\}$
$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$	

Table C.3: Semantic rules for $\mathcal{DLR}_\mu$.

$(\top_n)_\rho^{\mathcal{I}} = \top_n^{\mathcal{I}} \subseteq (\Delta^{\mathcal{I}})^n$	$(\neg C)_\rho^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C_\rho^{\mathcal{I}}$
$P_\rho^{\mathcal{I}} = P^{\mathcal{I}} \subseteq \top_n^{\mathcal{I}}$	$(C_1 \sqcap C_2)_\rho^{\mathcal{I}} = (C_1)_\rho^{\mathcal{I}} \cap (C_2)_\rho^{\mathcal{I}}$
$(\neg R)_\rho^{\mathcal{I}} = \top_n^{\mathcal{I}} \setminus R_\rho^{\mathcal{I}}$	$(\$i/n : C)_\rho^{\mathcal{I}} = \{(d_1, ..., d_n) \in \top_n^{\mathcal{I}} \| d_i \in C_\rho^{\mathcal{I}}\}$
$(R_1 \sqcap R_2)_\rho^{\mathcal{I}} = (R_1)_\rho^{\mathcal{I}} \cap (R_2)_\rho^{\mathcal{I}}$	$(\exists[\$i]R)_\rho^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \| \exists(d_1, ..., d_n) \in R_\rho^{\mathcal{I}}.d_i = d\}$
$(\top_1)_\rho^{\mathcal{I}} = \Delta^{\mathcal{I}}$	$(\leq k[\$i]R)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \|\| \#\{(d_1, ..., d_n) \in R_\rho^{\mathcal{I}} \| d_i = d\} \leq k\}$
$A_\rho^{\mathcal{I}} = A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$	$(\mu X.C)_\rho^{\mathcal{I}} = \bigcap\{\mathcal{E} \subseteq \Delta^{\mathcal{I}} \| C_{\rho[X/\mathcal{E}]}^{\mathcal{I}} \subseteq \mathcal{E}\}$

trees of a role (Calvanese *et al.*, 1999), which allows a modeller to represent acyclicity, transitivity, asymmetry, and (ir)reflexivity. Using standard first-order notions of scope, bound and free occurrences of variables etc., treating μ and ν as quantifiers, then relations, with arity between 2 and n_{max}, and concepts can be built according to the following syntax:

$R \longrightarrow \top_n \mid P \mid (\$i/n : C) \mid \neg R \mid R_1 \sqcap R_2$

$C \longrightarrow \top_1 \mid A \mid X \mid \neg C \mid C_1 \sqcap C_2 \mid \exists[\$i]R \mid \ \leq k[\$i]R \mid \mu X.C$

Denotations are as for $\mathcal{DLR}$ above, with the additions that X denotes a concept variable and every free occurrence of X in $\mu X.C$ is in the scope of an even number of negations (where $(\leq k[\$i]R)$ counts as one negation). Due to the presence of free variables, the interpretation function $\cdot^{\mathcal{I}}$ cannot be extended directly to every concept of the logic, therefore, valuation ρ is introduced on an interpretation so that ρ is a mapping from variables to subsets of $\Delta^{\mathcal{I}}$. Given a valuation ρ, $\rho[X/\mathcal{E}]$ denotes the valuation identical to ρ except that $\rho[X/\mathcal{E}] = \mathcal{E}$. Then, by associating to $\mathcal{I}$ and ρ an extension function $\cdot_\rho^{\mathcal{I}}$, we get the semantic rules for $\mathcal{DLR}_\mu$ as shown in *Table* C.3. $\nu X.C$ is used as abbreviation for $\neg\mu X.\neg C[X/\neg X]$, where $C[X/\neg X]$ denotes the concept obtained by replacing all free occurrences of X by $\neg X$; hence, its semantics follows as:

$(\nu X.C)_\rho^{\mathcal{I}} = \bigcup\{\mathcal{E} \subseteq \Delta^{\mathcal{I}} | \mathcal{E} \subseteq C_{\rho[X/\mathcal{E}]}^{\mathcal{I}}\}$

With $C_{\rho[X/\mathcal{E}]}^{\mathcal{I}}$ as operator from $\mathcal{E}$ of $\Delta^{\mathcal{I}}$ to subsets of $\Delta^{\mathcal{I}}$, then $\mu X.C$ denotes the least fixpoint of the operator and $\nu X.C$ the greatest fixpoint. With R^* denoting reflexive transitive closure (see *Table C.1*), the we have the equivalences

$\exists R^*.C = \mu X(C \sqcup \exists R.X)$

$\forall R^*.C = \nu X.(C \sqcap \forall R.X)$

To see how this can be used, one first defines a new concept Tree, and subsequently define a particular tree, such as an acyclic hierarchy of granular levels, with the Tree concept, the node (*e.g.*, granular level GL) and edge (role, *e.g.*, RL between levels). More precisely (based on Calvanese *et al.*, 1999; Calvanese and De Giacomo, 2003), Tree is inductively defined as:

DEFINITION C.3 (Tree). Tree *is the concept with the smallest extension among those satisfying*

(i) An individual that is an EmptyTree *is a* Tree*; and*

(ii) If an individual is a Node*, has at most one parent, has some children, and all children are* Trees*, then such an individual is a* Tree*.*

In $\mathcal{DLR}_\mu$ notation, where `Node` is a concept in the tree and `Edge` the relation between nodes, we have

`Tree[Node,Edge]` $\equiv \mu X.$`(EmptyTree` $\sqcup$ `(Node` $\sqcap$ `(`≤ 1`[$2]Edge)` $\sqcap$
$\exists$`[$1]Edge` $\sqcap$ $\neg\exists$`[$1](Edge` $\sqcap$ `($ 2/2:`$\neg X$`))))`

For a hierarchy of granular levels in some gp_1, `Hgp1`, we can then state `Hgp1` $\equiv$ `Tree[GL1,RL]`, provided that we also have `GL1` $\sqsubseteq$ `GL` and the levels in gp_1 instantiating `GL1`.

This concludes the DL introduction and summaries of $\mathcal{DLR}_{ifd}$ and $\mathcal{DLR}_\mu$.

Bibliography

Abelló, A., Samos, J., Saltor, F. (2006). YAM^2: a multidimensional conceptual model extending UML. *Information Systems*, 2006, **31**(6): 541-567.

Abiteboul, S., Hull, R., Vianu, V. (1995). *Foundations of databases*. Addison Wesley, USA.

Akahani, J., Hiramatsu, K., Kogure, K. (2002). Coordinating Heterogeneous Information Services based On Approximate Ontology Translation. *First International Joint Conference on Autonomous Agents & Multiagent Systems (AA MAS 2002)*, Bologna, Italy. 2002.

Albrecht, M., Gotelli, N.J. (2001). Spatial and temporal niche partitioning in grassland ants. *Oecologia*, 2001, **126**: 134-141.

Aldridge, B.B., Burke, J.M., Lauffenburger, D.A., Sorger, P.K. (2006). Physicochemical modelling of cell signalling pathways. *Nature Cell Biology*, 2006, **8**: 1195-1203.

Antezana, E., Tsiporkova, E., Mironov, V., et al. (2006). A cell-cycle knowledge integration framework. *data integration in the Life Sciences (DILS'06)*, Springer, Lecture Notes in Computer Science **4075**, 19-34.

Antonovics, J., Abbate, J.L., Baker, C.H., Daley, D., Hood, M.E., et al. (2007). Evolution by Any Other Name: Antibiotic Resistance and Avoidance of the E-Word. *PLoS Biology*, 2007, **5**(2): e30.

Artale, A., Franconi, E., Guarino, N., Pazzi, L. (1996a). Part-Whole Relations in Object-Centered Systems: an Overview. *Data & Knowledge Engineering*, 1996, **20**(3): 347-383.

Artale, A., Franconi, E., Guarino, N. (1996b). Open Problems for Part-Whole Relations. In: *Proceedings of 1996 International Workshop on Description Logics (DL'96)*. AAAI Press, Cambridge, MA, 1996, 70-73.

Artale, A., Franconi, E., Mandreoli, F. (2003). Description logics for modeling dynamic information. In: Chomicki, J., van der Meyden, R., Saake, G. (Eds), *Logics for Emerging Applications of Databases*. Lecture Notes in Computer Science, Springer-Verlag, Berlin.

Artale, A., Franconi, E., Wolter, F., Zakharyaschev, M. (2002). A temporal description logic for reasoning about conceptual schemas and queries. In: *Proceedings of the 8th Joint European Conference on Logics in Artificial Intelligence (JELIA-02)*, S. Flesca, S. Greco, N. Leone, G. Ianni (eds.), Springer Verlag, 2002, Lecture Notes in Artificial Intelligence **2424**, 98-110.

Artale, A., Parent, C., Spaccapietra, S. (2006). Modeling the evolution of objects in temporal information systems. In: *4th International Symposium on Foundations of Information and Knowledge Systems (FoIKS-06)*, Springer-Verlag, 2006, Lecture Notes in Computer Science **3861**, 22-42.

Atauri, P. de, Orrell, P.D., Ramsey, S., Bolouri, H. (2004). Evolution of 'design' principles in biochemical networks. *Systems Biology*, 2004, **1**(1).

Baader, F., Nutt, W. (2003). Basic Description Logics. In: *Description Logics Handbook*, Baader, F. Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds). Cambridge University Press, 2003. pp47-100.

Baader, F. Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds). (2003). *Description Logics Handbook*, Cambridge University Press, 2003.

Bailey, R.G. (2001). *Description of the ecoregions of the United States*. Technical Report, USDA Forest Service, March 1995, http://www.fs.fed.us/land/ecosysmgmt/ecoreg1_home.html, Date last modified: 8-17-2001 (Date accessed 20-8-2007).

Barbati, A., Corona, P., Marchetti, M. (2007). A forest typology for monitoring sustainable forest management: The case of European Forest Types [abstract]. *Plant Biosystems*, 2007, 141(1): 93-103.

Barbier, F., Henderson-Sellers, B., Le Parc-Lacayrelle, A., Bruel, J.-M. (2003). Formalization of the whole-part relationship in the Unified Modelling Language. *IEEE Transactions on Software Engineering*, 2003, **29**(5): 459-470.

Bargiela, A. Pedrycz, W. (2006). The roots of granular computing. *IEEE International Conference on Granular Computing 2006 (GrC06)*, 10-12 May 2006, Atlanta, USA. IEEE Xplore (ISBN 1-4244-0134-8), **1**, 806-809.

Bender, A. Glen, R.C. (2004). Molecular similarity: a key technique in molecular informatics. *Org. Biomol. Chem.*, 2004, **2**: 3204–3218.

Benerecetti, M., Bouquet, P., Ghidini, C. (2000). Contextual reasoning distilled. *Journal of Theoretical and Experimental Artificial Intelligence*, 2000, **12**(3):279-305.

Benerecetti, M., Bouquet, P., Ghidini, C. (2001). On the dimensions of context dependence: partiality, approximation, and perspective. *Third International and Interdisciplinary Conference on Modelling and Using Context (Context 2001)*, Dundee, Scotland, August 2001.

Berardi, D., Calvanese, D., De Giacomo, G. (2005). Reasoning on UML class diagrams. *Artificial Intelligence*, 2005, **168**(1-2): 70-118.

Bernal, A., Ear, U., Kyrpides, N. (2001). Genomes OnLine Database (GOLD): a monitor of genome projects world-wide. *Nucleic Acids Research*, 2001, **29**(1): 126-127.

Bettini, C., Dyreson, C.E., Evans, W.S., Snodgrass, R.T., Wang, X.S. (1998). A glossary of time granularity concepts. In: *Temporal Databases: Research and Practice*, O. Etzion, S. Jajodia, S. Sripada (Eds.), Lecture Notes in Computer Science State-of-the-art Survey 1399, Springer, 1998. pp406-413.

Bettini, C., Mascetti, S., Wang, X.S. (2007). Supporting temporal reasoning by mapping calendar expressions to minimal periodic sets. *Journal of Artificial Intelligence Research*, **28**, 299-348.

Bettini, C., Ruffini, S. (2003). Direct granularity conversions among temporal constraints. *Journal of Universal Computer Science*, 2003, **9**(9): 1123-1136.

Bittner, T., Donnelly, M. (2005). Computational ontologies of parthood, componenthood, and containment, In: *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence 2005 (IJCAI05)*. Kaelbling, L. (ed.), 382-387.

Bittner, T., Smith, B. (2003). A Theory of Granular Partitions. In: *Foundations of Geographic Information Science*, Duckham, M, Goodchild, MF, Worboys, MF (eds.), London: Taylor & Francis Books, 2003, pp117-151.

Bittner, T., Stell, J. (2003). Stratified rough sets and vagueness, In: *Spatial Information Theory. Cognitive and Computational Foundations of Geographic Information Science. International Conference (COSIT'03)*, Kuhn, W., Worboys, M., Timpf, S. (eds.). 2003. pp286-303.

Bloesch, A.C., Halpin, T.A. (1996). ConQuer: a conceptual query language. *Proceedings of ER'96: 15th International Conference onconceptual modeling*. Springer Lecture Notes in Computer Science **1157**, 121-133.

Bloesch, A.C., Halpin, T.A. (1997). Conceptual Queries using ConQuer-II. *Proceedings of ER'97: 16th International Conference on Conceptual Modeling*. Springer Lecture Notes in Computer Science **1331**, 113-126.

Böhlen, M., Gamper, J., Jensen, C.S. (2006). Multi-dimensional aggregation for temporal data. In: *Proceedings of the 10th International Conference on Extending Database Technologies (EDBT'06)*, Ioannidis, Y., et al. (Eds.), Springer Berlin, Lecture Notes in Computer Science **3896**, 257-275.

Borgida, A., Lenzerini, M., Rosati, R. (2002). Description Logics for Data Bases. In: *Description Logics Handbook*, Baader, F. Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds). Cambridge University Press, 2002. pp472-494.

Borgo, S., Masolo, C. (2007). Full mereogeometries. *Journal of Philosophical Logic*, 2007. (to appear).

Bouquet, P., Serafini, L. (2001). Two formalizations of context: a comparison. *Third International and Interdisciplinary Conference on Modelling and Using Context, Context 2001*, Dundee, Scotland, August 2001.

Bouquet, P., Euzenat, J., Franconi, E., Serafini, L., Stamou, G., Tessaris, S. (2004). *Specification of a common framework for characterizing alignment*. KnowledgeWeb Deliverable D2.2.1, v1.2, 3-8-2004.

Broekhoven, E. van, Adriaenssens, V., De Baets, B. (2007). Interpretability-preserving genetic optimization of linguistic terms in fuzzy models for fuzzy ordered classification: an ecological case study. *International Journal of Approximate Reasoning*, 2007, **44**: 65-90.

Bruijn, J. De. *WSML Abstract syntax*. WSML Working Draft D16.3v0.3, 20-9-2007. http://www.wsmo.org/TR/d16/d16.3/v0.3/20070920. Date accessed: 21-9-2007.

Bruijn, J. De, Heymans, S. (2007). A semantic framework for language layering in WSML. In *Proceedings of RR2007*, Springer Lecture Notes in Computer Science **4524**, 103-117.

Cabibbo, L., Torlone, R. (1998). A logical approach to multidimensional databases. In *Proceedings of EDBT'98*, 1998, 183-197.

Calvanese, D., Damaggio, E., De Giacomo, G., Lenzerini, M., Rosati, R. (2003). Semantic data integration in P2P systems. *International Workshop on Databases, Information Systems and Peer-to-Peer Computing*, 2003.

Calvanese, D., De Giacomo, G. (2003). Expressive description logics. In: *The Description Logic Handbook:*

Theory, Implementation and Applications, Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. (Eds). Cambridge University Press, 2003. pp178-218.

Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R. (2005). DL-Lite: Tractable description logics for ontologies. In: *Proceedings of the 20th National Conference on Artificial Intelligence (AAAI 2005)*, pp602-607.

Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Poggi, A., Rosati, R. (2006a). Linking data to ontologies: The description logic $DL\text{-}Lite_{\mathcal{A}}$. In *Proceedings of the Second Workshop OWL Experiences and Directions (OWLED 2006)*.

Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R. (2006b). Data complexity of query answering in description logics. *Proceedings of the 10th International Conference of Knowledge Representation and Reasoning (KR-2006)*, Lake District, UK, 2006. pp260-270.

Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R. (2007). Tractable reasoning and efficient query answering in description logics: The DL-Lite family. *Journal of Automated Reasoning*, 2007, **39**(3): 385-429.

Calvanese, C., De Giacomo, G., Lenzerini, M. (1998a). On the decidability of query containment under constraints. In: *Proceedings of the 17th ACM SIGACT SIGMOD SIGART Sym. on Principles of Database Systems (PODS'98)*, 149-158.

Calvanese, D., De Giacomo, G., Lenzerini, M. (1999). Reasoning in expressive description logics with fixpoints based on automata on infinite trees. In: *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI'99)*, 84-89.

Calvanese, D., De Giacomo, G., Lenzerini, M. (2001a). Identification constraints and functional dependencies in Description Logics. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI 2001)*, 2001, 155-160.

Calvanese, D., De Giacomo, G., Lenzerini, M. (2001b). Ontology of integration and integration of ontologies. In *Proceedings of the International Workshop on Description Logics 2001 (DL'01)*.

Calvanese, D., Lenzerini, M., Nardi, D. (1998b). Description logics for conceptual data modeling. In J. Chomicki and G. Saake, editors, *Logics for Databases and Information Systems*. Kluwer, Amsterdam.

Camossi, E., Bertolotto, M., Bertino, E., Guerrini, G. (2003). Issues on Modelling Spatial Granularity. *Workshop on fundamental issues in spatial and geographic ontologies*, 23 Sept. 2003. Ittingen, Switzerland.

Campbell, L.J., Halpin, T.A. and Proper, H.A. (1996). Conceptual Schemas with Abstractions: Making flat conceptual schemas more comprehensible. *Data & Knowledge Engineering*, 1996, **20**(1): 39-85.

Cariani, P. (1997). Emergence of new signal-primitives in neural systems. *Intellectica*, 1997, 25:95-143.

Chen, Y.H., Yao, Y.Y. (2006). Multiview intelligent data analysis based on granular computing. *IEEE International Conference on Granular Computing (GrC'06)*. IEEE Computer Society, 2006, 281-286.

Chen, D., Wang, X., Zhao, S. (2007). Attribute Reduction Based on Fuzzy Rough Sets. In: *Proceedigns of the International Conference Rough Sets and Intelligent Systems Paradigms (RSEISP 2007)*, Kryszkiewicz, M., Peters, J.F., Rybinski, H., Skowron, A. (Eds.). Springer, Berlin, Lecture Notes in Artificial Intelligence **4585**, 381-390.

Cook, D.L., Mejino, J.L.V., Rosse, C. (2004). Evolution of a Foundational Model of Physiology: Symbolic Representation for Functional Bioinformatics. In: *Proceedings of MEDINFO 2004*, M. Fieschi *et al.* (eds.). Amsterdam: IOS Press, 2004. pp336-340.

Cuenca Grau, B., Parsia, B., Sirin, E., Kalyanpur, A. (2006). Modularity and Web Ontologies. In *Proceedings of the 10th International Conference on Principles of Knowledge Representation and Reasoning (KR2006)*. Lake District, UK.

Cuenca Grau, B., Horrocks, I., Parsia, B., Patel-Schneider, P., Sattler, U. (2006). Next steps for OWL. In *Proceedings of the Second Workshop OWL Experiences and Directions (OWLED-2006)*.

Cuenca Grau, B., Horrocks, I., Kazakov, Y., Sattler, U. (2007). Just the Right Amount: Extracting Modules from Ontologies. *Proceedings of the 16th International World Wide Web Conference (WWW-2007)*. Canada.

Dal Lago, U., Montanari, A. (2001). Calendars, Time Granularities, and Automata. *Proceedings of the 7th International Symposium on Advances in Spatial and Temporal Databases (SSTD 2001)*, C.S. Jensen et al. (Eds.), Springer-Verlag Berlin Heidelberg Lecture Notes in Computer Science **2121**, 279-298.

Daraselia, N., Egorov, S., Yazhuk, A., Novichkova, S., Yuryev, A., Mazo, I. (2004). Extracting Protein Function Information from MEDLINE Using a Full-Sentence Parser. *Proceedings of the Second European Workshop on Data Mining and Text Mining in Bioinformatics*. 2004. pp11-18.

Dawyndt, P., De Meyer, H., De Baets, B. (2006). UPGMA clustering revisited: A weight-driven approach to transitive approximation. *International Journal of Approximate Reasoning*, 2006, **42**(3): 174-191.

Degtyarenko, K., Contrino, S. (2004). COMe: the ontology of bioinorganic proteins. *BMC Structural Biology*, 2004, **4**: 3.

Delehanty, M. (2005). Emergent properties and the context objection to reduction. *Biology and Philosophy*, 2005, **20**(4): 715-734.

DeLong, E.F. (2005). Microbial community genomics in the ocean. *Nature Reviews Microbiology*, 2005, **3**: 459-469.

Doms, A., Schroeder, M. (2005). GoPubMed: Exploring PubMed with the GeneOntology. *Nucleic Acids Research*, 2005, **33**: W783-W786.

Earl, D. (2005). The classical theory of concepts. *Internet Encyclopedia of Philosophy*, 2005. http://www.iep.utm.edu/c/concepts.htm.

Editorial. (2005). Lessons from Listeria. *Nature Structural & Molecular Biology*, 2005, **12**: 1.

Edmonds, B. (2000). Complexity and Scientific Modelling. *Foundations of Science*, 2000, **5**(3): 379-390.

Ellis, G.F.R. (2005). Physics, complexity and causality. *Nature*, 2005, **435**: 743.

Elmasri, R., Fu, J., Ji, F. (2007). Multi-level conceptual modeling for biomedical data and ontologies integration. *20th IEEE International Symposium on Computer-Based Medical Systems (CBMS'07)*. June 20-22, 2007, Maribor, Slovenia. pp589-594.

Emmeche, C., Køppe, S., Stjernfelt, F. (1997). Explaining emergence: Towards an ontology of levels. *Journal for General Philosophy of Science*, 1997, **28**: 83-119.

Eppley, J.M., Tyson, G.W., Getz, W.M., Banfield, J.F. (2007). Strainer: Software for analysis of population variation in community genomic datasets [abstract]. *BMC Bioinformatics*, 2007, **8**:398. Published 17-10-2007, date accessed: 7-11-2007.

Euzenat, J. (1995). An algebraic approach to granularity in qualitative time and space representation. In: *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'95)*, 894-900.

Euzenat, J., Montanari, A. (2005). Time granularity. In: *Handbook of temporal reasoning in artificial intelligence*, Fisher, M., Gabbay, D., Vila, L. (Eds.). Amsterdam: Elsevier 2005. pp59-118.

Fagin, R., Guha, R., Kumar, R., Novak, J., Sivakumar, D., Tomkins, A. (2005). Multi-Structural Databases. *PODS 2005*. Baltimore, Maryland, June 13-16, 2005.

Fangerau, H. (2004). Finding European bioethical literature: an evaluation of the leading abstracting and indexing services. *Journal of Medical Ethics*, 2004, **30**(3): 299-303.

Fent, I. de, Gubiani, D., Montanari, A. (2005). Granular GeoGraph: a multi-granular conceptual model for spatial data. In: *Proceedings of the 13th Italian Symposium on Advanced Databases (SEBD'05)*. Calí, A., Calvanese, D., Franconi, E., Lenzerini, M., Tanca, L. (eds). Roma: Aracne editrice, 2005, pp368-379.

Fillies, C., Weichhardt, F., Smith, B. (2005). Semantically correct Visio Drawings. *2nd European Semantic Web Conference (ESWC'05)*, Heraklion, Crete, May 29-June 1, 2005.

Fillottrani, P., Franconi, E., Tessaris, S. (2006) The new ICOM ontology editor. In: *19th International Workshop on Description Logics (DL 2006)*, Lake District, UK. May 2006.

Fonseca, F., Egenhofer, M., Davis, C., Camara, G. (2002). Semantic Granularity in Ontology-Driven Geographic Information Systems. *Annals of Mathematics and Artificial Intelligence*, 2002, **36**(1-2): 121-151.

Franconi, E., Kamble, A. (2003). The GMD Data Model for Multidimensional Information: a brief introduction. *5th International Conference on Data Warehousing and Knowledge Discovery (DaWaK'03)*, Prague, Czech Republic, 2003.

Franconi, E., Kamble, A. (2004). The GMD Data Model and Algebra for Multidimensional Information. *Proceedings of the 16th International Conference on Advanced Information Systems Engineering (CAiSE-04)*, Riga, Latvia, 2004.

Franconi, E., Ng, G. (2000). The iCom tool for intelligent conceptual modelling. *7th Intl. Workshop on Knowledge Representation meets Databases (KRDB'00)*, Berlin, Germany. 2000.

Galperin, M.Y. (2005). The Molecular Biology Database Collection: 2005 update. *Nucleic Acids Research*, 2005, **33**: D5-D24.

Gandhi, M., Robertson, E., Van Gucht, D. (1995). Leveled Entity-Relationship Model, an extension of ER notions to provide for a hierarchical model, using O-O-like encapsulation. *Entity-Relationship '95*.

Gangemi, A., Mika, P. (2003). Understanding the Semantic Web through Descriptions and Situations. *International Conference on Ontologies, Databases and Applications of SEmantics (ODBASE 2003)*, Catania, (Italy), November 3-7, 2003.

Gärdenfors, P. (2000). *Conceptual Spaces: The Geometry of Thought*. Bradford Books: MIT Press, 2000.

Gene Ontology Consortium. The Gene Ontology (GO) database and informatics resource. *Nucleic Acids Research*, 2004, **32**: D258-D261.

Gerstl, P., Pribbenow, S. (1995). Midwinters, end games, and body parts: a classification of part-whole relations. *International Journal of Human-Computer Studies*, 1995, 43:865-889.

Gevers, D., Cohan, F.M., Lawrence, J.G., Spratt, B.G., Coenye, T., Feil, E.J., Stackebrandt, Peer, Y. van der, Vandamme, P., Thompson, F.L., Swings, J. (2005). Re-evaluating prokaryotic species. *Nature Reviews*

Microbiology, 2005, **3**(9): 733-739.

Ghidini, C., Giunchiglia, F. (2003). *A semantics for abstraction*. Technical Report DIT-03-082, University of Trento, Italy. 2003. 15p.

Giunchiglia, F. (1993). Contextual reasoning. *Epistemologia - Special Issue on I Linguaggi e le Macchine*, 1993, **XVI**: 345-364.

Giunchiglia, F., Villafiorita, A., Walsh, T. (1997). Theories of abstraction. *AI Communications*, 1997, **10**(3-4): 167-176.

Giunchiglia, F., Walsh, T. (1992). A theory of abstraction. *Artificial Intelligence*, 1992, **57**(2-3): 323-389.

Gogarten, J.P., Townsend, J.P. (2005). Horizontal gene transfer, genome innovation and evolution. *Nature Reviews Microbiology*, 2005, **3**(9): 679-687.

Graham, M.H., Dayton, P.K. (2002). On the evolution of ecological ideas: paradigms and scientific progress. *Ecology*, 2002, **83**(6): 1481-1489.

Grizzi, F., Chiriva-Internati, M. (2005). The complexity of anatomical systems. *Theoretical Biology and Medical Modelling*, 2005, **2**: 26.

Grosof, B.N., Horrocks, I., Volz, R., Decker, S. (2003). Description Logic Programs: Combining Logic Programs with Description Logic. *The Twelfth International World Wide Web Conference*, 20-24 May 2003, Budapest, Hungary. ACM, 2003. pp48-57.

Gross, L. (2007). Untapped Bounty: Sampling the Seas to Survey Microbial Biodiversity. *PLoS Biology*, 2007, *5*(3): e85.

Guarino, N. (1998). Formal Ontology and Information Systems. *Proceedings of Formal Ontology in Information Systems (FOIS'98)*. Amsterdam: IOS Press, 1998.

Guarino, N., Welty, C. (2000a). A Formal Ontology of Properties. *Proceedings of 12th Int. Conf. on Knowledge Engineering and Knowledge Management (EKAW'00)*, Dieng, R (ed.), Springer Verlag, Lecture Notes in Computer Science, 2000.

Guarino, N., Welty, C. (2000b). Identity, Unity, and individuality: towards a formal toolkit for ontological analysis. *Proceedings of ECAI-2000*. Horn, W. (ed.). IOS Press, Amsterdam. 2000.

Guarino, N., Welty, C.A. (2004). An overview of OntoClean. In: *Handbook on ontologies*. Stab, S., Studer, R. (eds.). Springer Verlag, 2004. pp151-159.

Guizzardi, G. (2007). Modal Aspects of Object Types and Part-Whole Relations and the de re/de dicto distinction. *19th International Conference on Advances in Information Systems Engineering (CAiSE)*, Trondheim, Norway, 2007. Springer-Verlag, Berlin, Lecture Notes in Computer Science **4495**.

Guizzardi, G. (2005). *Ontological foundations for structural conceptual models*. PhD Thesis, Telematica Institute, Twente University, Enschede, the Netherlands.

Guizzardi, G., Wagner, G., Guarino, N., van Sinderen, M. (2004a). An Ontologically Well-Founded Profile for UML Conceptual Models. *16th International Conference on Advances in Information Systems Engineering (CAiSE)*, Latvia, 2004. Springer-Verlag , Berlin, Lecture Notes in Artificial Intelligence.

Guizzardi, G., Wagner, G., Herre, H. (2004b). On the Foundations of UML as an Ontology Representation Language. In: *Proceedings of EKAW 2004*. Motta, E, *et al.* (eds.), Springer-Verlag Berlin, Lecture Notes in Artificial Intelligence **3257**, 47-62.

Halpin, T.A. (1989). *A logical analysis of information systems: static aspects of the data-oriented perspective*. PhD Thesis, University of Queensland, Australia. 1989.

Halpin, T. (2001). *Information Modeling and Relational Databases*. San Francisco: Morgan Kaufmann Publishers, 2001.

Halpin, T. A. (2004). Comparing Metamodels for ER, ORM and UML Data Models. *Advanced Topics in Database Research*, Siau, K (Ed.), 2004, Idea Publishing Group, Hershey PA, USA. **3**: 23-44.

Hanahan, D., Weinberg, R.A. (2000). The Hallmarks of Cancer. *Cell*, **100**: 57-70, 2000.

Hartman, T., Sharan, R. (2004). A 1.5-Approximation Algorithm for Sorting by Transpositions and Transreversals. *Proceedings of the 4th Workshop on Algorithms in Bioinformatics*, Bergen, Norway, September 14-17, 2004, pp50-61.

Hata, Y., Mukaidono, M. (1999). On Some Classes of Fuzzy Information Granularity and Their Representations. In: *Proceedings of the Twenty Ninth IEEE International Symposium on Multiple-Valued Logic* May 20 - 22, 1999, Freiburg im Breisgau, Germany. pp288-293.

Hawley, K. (2004). Temporal Parts. *The Stanford Encyclopedia of Philosophy (Winter 2004 Edition)*. Zalta, E.N. (ed.). http://plato.stanford.edu/archives/win2004/entries/temporal-parts/.

Hedman, S. (2004). *A first course in logic—an introduction to model theory, proof theory, computability, and complexity*. Oxford: Oxford University Press.

Hefflin, J., Hendler, J. (2000). Dynamic ontologies on the Web. *Proceedings of 17th National Conference on Artificial Intelligence (AAAI 2000)*. 2000.

Herre, H., Heller, B. (2006). Semantic foundations of medical information systems based on top-level ontologies. *Knowledge-Based Systems*, 2006, **19**: 107-115.

Hettne, K., Boyer, S. (2005). *The Nuclear Receptor Pathway Database (NRpath)*. Technical Report Deliverable Activity A.1 and A.3, WP6.1 Pharmainformatics, InfoBiomed Project, http://www.infobiomed.org. 2005.

Hey, J. (2001). The mind of the species problem. *Trends in Ecology and Evolution*, 2001, **16**: 326-329.

Hobbs, J.R. (1985). Granularity. *International Joint Conference on Artificial Intelligence (IJCAI85)*, 1985, 432-435.

Hobbs, J.R., Pustejovsky, J. (2003). Annotating and Reasoning about Time and Events. In: *Proceedings of the AAAI Spring Symposium on Logical Formulization of Commonsense Reasoning*, Stanford University, CA, 2003.

Hobbs, J.R., Pan, F. (2004). An Ontology of Time for the Semantic Web. *ACM Transactions on Asian Language Processing (TALIP): Special issue on Temporal Information Processing*, 2004, **3**(1): 66-85.

Hofstede, A.H.M. ter, Proper, H.A. (1998). How to Formalize It? Formalization Principles for Information Systems Development Methods. *Information & Software Technology*, 1998, **40**(10): 519-540.

Hollinger, F.B., Kleinman, S. (2003). Transfusion transmission of West Nile virus: a merging of historical and contemporary perspectives. *Transfusion*, 2003, **43**(8): 992-997.

Horrocks, I., Kutz, O., Sattler, U. (2006). The Even More Irresistible $\mathcal{SROIQ}$. In: *Proceedings of the 10th International Conference of Knowledge Representation and Reasoning (KR-2006)*, Lake District, UK, 2006.

Hunter, P.J., Borg, T. (2003). Integration from Proteins to Organs: The Physiome Project. *Nature*, 2003, **4**(3): 237-243.

Jäschke, P., Oberweis, A., Stucky, W. (1993). Extending ER Model Clustering by relationship clustering. *Proceedings of the 12th International Conferences on Entity Relationship Approach*, 15-17 Dec, Arlington, Texas, 1993.

Jaiswal, P., Ware, D., Ni, J., Chang, K., Zhao, W., Schmidt, S., Pan, X., Clark, K., Teytelman, L., Cartinhour, S., Stein, L., McCouch, S. (2002). Gramene: development and integration of trait and gene ontologies for rice. *Comparative and Functional Genomics*, 2002, **3**: 132-136.

Jarke, M., Quix, C., Becks, A., Franconi, E. (2005). Showcase featuring a meaningful subset of the SEWASIE prototypes. Deliverable D7.7, SEWASIE Project, 7-5-2005. http://www.sewasie.org/documents/D-7-7-showcase.pdf.

Ji, L., Nymeyer, H., Yu, Y. (2007). Data-driven time-parallelization on the AMF simulation of proteins. *Sixth IEEE International Workshop on High Performance Computational Biology (HiCOMB'07)*, California, USA, 26 March 2007.

Johansson, I. (2004a). On the transitivity of the parthood relation. In: *Relations and predicates*, Hochberg, H. and Mulligan, K. (eds.). Ontos Verlag: Frankfurt, 2004. pp161-181.

Johansson, I. (2004b). The Ontology of temperature. *Philosophical Communications*, 2004, **32**: 115-124.

Johansson, I. (2005). Bioinformatics and Biological Reality. *Journal of Biomedical Informatics*, 2006, **39**: 274-287.

Johansson, I. (2006). Mereology and Ordinary Language - Reply to Varzi. *Applied Ontology*, 2006, **1**(2): 157-161.

Johansson, I., Smith, B., Munn, K., Tsikolia, N., Elsner, K., Ernst, D., Siebert, D. (2005). Functional Anatomy: A Taxonomic Proposal. *Acta Biotheoretica*, 2005, **53**(3): 153-166.

Johnson, D.M. (1990). Can Abstractions be Causes? *Biology and Philosophy*, 1990, **5**: 63-77.

Kamble, A.S. (2004). *A Data Warehouse Conceptual Data Model for Multidimensional Information*. PhD thesis, University of Manchester, UK. 2004.

Kaplan, N., Sasson, O., Inbar, U., Friedlich, M., Fromer, M., Fleischer, H., Portugaly, E., Linial, N., Linial, M. (2005). ProtoNet 4.0: A hierarchical classification of one million protein sequences. *Nucleic Acids Research*, 2005, **33**: D216-D218.

Kaplan, N. Friedlich, M. Fromer, Linial, M. (2004). A functional hierarchical organization of the protein sequence space. *BMC Bioinformatics*, 2004, **5**: 196.

Keet, C.M. (2004a). *Aspects of ontology integration*. Technical Report, School of Computing, Napier University, UK. 2004.

Keet, C.M. (2004b). *Ontology development and integration for the biosciences*. Technical Report, School of Computing, Napier University, UK. 2004.

Keet, C.M. (2004c). Ontology integration for applied bioscience. *Summer School Semantic Mining in Biomedicine*, research student poster session, Balatonfüred, Hungary, 2-11 July 2004.

Keet, C.M. (2005a). Factors affecting ontology development in ecology. *Data Integration in the Life Sciences (DILS2005)*. Ludäscher, B., Raschid, L. (eds.). San Diego, USA, 20-22 July 2005. Springer Verlag, Lecture

Notes in Bioinformatics **3615**, 46-62.

Keet, C.M. (2005b). Using abstractions to facilitate management of large ORM models and ontologies. *International Workshop on Object-Role Modeling (ORM'05)*. In: OTM Workshops 2005. Halpin, T., Meersman, R. (eds.), Berlin: Springer-Verlag, Lecture Notes in Computer Science **3762**, 603-612.

Keet, C.M. (2005c). Current characteristics and historical perspective of Computer Science and IT with/for biology. In: *CSBio reader: extended abstracts of the 'CS&IT with/for biology' Seminar Series*, Free University of Bozen-Bolzano. Keet, C.M., Franconi E. (eds.). 2005. pp1-8.

Keet, C.M. (2006a). A taxonomy of types of granularity. *IEEE Conference in Granular Computing (GrC2006)*, 10-12 May 2006, Atlanta, USA. IEEE Computer Society (ISBN 1-4244-0134-8) / IEEE Xplore, **1**, 106-111.

Keet, C.M. (2006b). *Granular information retrieval from the Gene Ontology and from the Foundational Model of Anatomy with OQAFMA*. Technical Report KRDB06-1, Faculty of Computer Science, Free University of Bozen-Bolzano, 2006. http://www.inf.unibz.it/krdb/pub/. Extended abstract: Keet, C.M. Using bio-ontologies in RDBMSs for querying granular information. *Bioinformatics, Ontologies and databases (SBIOLBD 2006)*, 13-2-2006, EPFL, Lausanne, Switzerland. (http://www.meteck.org/files/SBIOLBD06_extabs.pdf).

Keet, C.M. (2006c). *A formal approach for using granularity with the infectious diseases domain*. Technical Report KRDB06-2, Faculty of Computer Science, Free University of Bozen-Bolzano, 2006. http://www.inf.unibz.it/krdb/pub/.

Keet, C.M. (2006d). Part-whole relations in Object-Role Models. *2nd International Workshop on Object-Role Modelling (ORM'06)*, Montpellier, France, Nov 2-3, 2006. In: OTM Workshops 2006. Meersman, R., Tari, Z., Herrero., P. *et al.* (Eds.), Berlin: Springer-Verlag, Lecture Notes in Computer Science **4278**, 1118-1127.

Keet, C.M. (2006e). *Introduction to part-whole relations: mereology, conceptual modelling and mathematical aspects*. KRDB Research Centre Technical Report KRDB06-3, Faculty of Computer Science, Free University of Bozen-Bolzano, Italy. 2 October 2006. http://www.inf.unibz.it/krdb/pub/.

Keet, C.M. (2006f). Enhancing biological information systems with granularity. *KnowledgeWeb PhD Symposium (KWEPSY06)*, Budva, Montenegro, 17 June 2006.

Keet, C.M. (2007a). *Mapping the Object-Role Modeling language ORM2 into Description Logic language $\mathcal{DLR}_{ifd}$*. KRDB Research Centre Technical Report KRDB07-2, Faculty of Computer Science, Free University of Bozen-Bolzano, Italy. 15 February 2007. http://www.inf.unibz.it/krdb/pub/. arXiv:cs.LO/0702089v1.

Keet, C.M. (2007b). Prospects for and issues with mapping the Object-Role Modeling language into $\mathcal{DLR}_{ifd}$. *20th International Workshop on Description Logics (DL'07)*, 8-10 June 2007, Bressanone, Italy. CEUR-WS, **Vol-250**, 331-338.

Keet, C.M. (2007c). Enhancing comprehension of ontologies and conceptual models through abstractions. *10th Congress of the Italian Association for Artificial Intelligence (AI*IA 2007)*, Rome, September 10-13, 2007. Basili, R., Pazienza, M.T. (Eds.), Springer-Verlag, Lecture Notes in Artificial Intelligence **4733**, 814-822.

Keet, C.M. (2007d). Granulation with indistinguishability, equivalence or similarity. *IEEE International Conference on Granular Computing (GrC2007)*, San Francisco, November 2-4, 2007. IEEE Computer Society, 11-16.

Keet, C.M. (2007e). *Granularity as a modelling approach to investigate hypothesized emergence in biology*. Unpublished manuscript v1.0, 3-8-2007. 22 p. http://www.meteck.org/files/Granemergev1.pdf.

Keet, C.M. (2008a). A formal comparison of conceptual data modeling languages. *13th International Workshop on Exploring Modeling Methods in Systems Analysis and Design (EMMSAD'08)*. Montpellier, France, 16-17 June 2008. CEUR-WS **Vol-xx** (in print).

Keet, C.M. (2008b). Toward cross-granular querying over modularized ontologies. *International Workshop on Ontologies: Reasoning and Modularity (WORM'08)*. Tenerife, Spain, 1 June 2008. submitted.

Keet, C.M. (2008c). Unifying industry-grade class-based conceptual data modeling languages with $\mathcal{CM}_{com}$. *21st International Workshop on Description Logics (DL'08)*, 13-16 May 2008, Dresden, Germany. CEUR-WS, **Vol-xx**(accepted).

Keet, C.M., Artale, A. (2007). Representing and Reasoning over a Taxonomy of Part-Whole Relations. *Applied Ontology – Special Issue on Ontological Foundations for Conceptual Models*, 2007, **2**(4): *in print*.

Keet, C.M., Franconi, E. (Eds.). (2005). *CSBio reader: extended abstracts of the 'CS&IT with/for biology' Seminar Series*, Free University of Bozen-Bolzano, Italy. 57p. http://www.inf.unibz.it/krdb/biology/.

Keet, C.M., Kumar, A. (2005). Applying partitions to infectious diseases. *XIX International Congress of the European Federation for Medical Informatics (MIE2005)*, 28-31 August 2005, Geneva, Switzerland. 2005. In: Connecting Medical Informatics and bio-informatics, Engelbrecht, R., Geissbuhler, A., Lovis, C. Mihalas, G. (eds.). Amsterdam: IOS Press. pp1236-1241.

Keet, C.M., Rodríguez, M. (2007). Toward using biomedical ontologies: trade-offs between ontology languages. *AAAI 2007 Workshop on Semantic e-Science (SeS07)*, 23 July 2007, Vancouver, Canada. AAAI 2007 Technical Report, **WS-07-11**, 65-68.

Keet, C.M., Roos, M., Marshall, M.S. (2007). A survey of requirements for automated reasoning services for bio-ontologies in OWL. *Third International Workshop OWL: Experiences and Directions (OWLED 2007)*, 6-7 June 2007, Innsbruck, Austria. CEUR-WS, **Vol-258**. 10p.

Kerschberg, L. (2001). Knowledge Management in Heterogeneous Data Warehouse Environments. In: *International Conference on Data Warehousing and Knowledge Discovery*, Springer Verlag, Munich, Lecture Notes in Computer Science **2114**, 1-10.

Khatri, P. Draghici, S. (2005). Ontological analysis of gene expression data: current tools, limitations, and open problems. *Bioinformatics*, 2005, **21**(18): 3587-3595.

Kiriyama, T., Tomiyama, T. (1993). Reasoning about Models across Multiple Ontologies. *International Qualitative Reasoning Workshop* Washington, May 16-20, 1993.

Kitano, H. (2002). Computational systems biology. *Nature*, 2002, **420**: 206-210.

Kitano, H., Funahashi, A., Matsuoka, Y., Oda, K. (2005). Using process diagrams for the graphical representation of biological networks. *Nature Biotechnology*, 2005, **23**(8): 961-966.

Klawonn, F., Kruse, R. (2004). The Inherent Indistinguishability in Fuzzy Systems. In: *Logic versus Approximation: Essays Dedicated to Michael M. Richter on the Occasion of his 65th Birthday*. Lenski, W. (ed.). Springer: Berlin / Heidelberg, Lecture Notes in Computer Science **3075**, 6-17.

Klein, M. (2001). Combining and Relating Ontologies: Problems and Solutions. *IJCAI Workshop on Ontologies*, Seattle, 2001.

Kolenbrander, P.E., Egland, P.G., Diaz, P.I., Palmer, R.J. (2005). Genome-genome interactions: bacterial communities in initial dental plaque. *Trends in Microbiology*, 2005, **13**(1): 11-15.

Korn, R.W. (2005). The Emergence Principle in Biological Hierarchies. *Biology and Philosophy*, 2005, **20**(1): 137-151.

Krivov, S., Villa F. (2005).Towards an Ontology based visual query system. *International Workshop on Data Integration in the Life Sciences (DILS2005)*, San Diego, USA, 20-22 July 2005. Springer Verlag, Lecture Notes in Bioinformatics **3615**, 313-316.

Krivov, S., Williams, R., Villa, F. (2005). *A Visualization Model for the Languages of Semantic Web*. Technical Report, University of Vermont. 9-6-2005. http://www.uvm.edu/∼skrivov/growl.pdf. (Date accessed: 3-10-2005)

Kumar, A., Yip, L., Smith, B., Grenon P. (2004). Bridging the Gap between Medical and Bioinformatics Using Formal Ontological Principles. *Third International Workshop on Computational Terminology*, Geneva, Switzerland. 2004.

Kumar, A., Smith, B., Novotny, D.D. (2005). Biomedical Informatics and Granularity. *Comparative and Functional Genomics*, 2005, **5**(6-7): 501-508.

Kumar, A., Yip, Y.L., Smith, B., Grenon, P. (2006). Bridging the Gap between Medical and Bioinformatics: an ontological case study in colon carcinoma. *Computers in Biology and Medicine*, 2006, **36**(7): 694-711.

Laboratory of Applied Ontology. (2005). *Formal Ontology for Knowledge Representation and Natural Language Processing*. Doctorate Course, ICT School, University of Trento, Italy. http://www.loa-cnr.it/PhD.html. Date accessed: 30-6-2005.

Lambrix, P., Padgham, L. (2000). Conceptual modeling in a document management environment using part-of reasoning in description logics. *Data & Knowledge Engineering*, 2000, **32**(1):51-86.

Lazebnik, Y. (2002). Can a Biologist Fix a Radio?—or, What I Learned while Studying Apoptosis. *Cancer Cell*, 2002, **2**: 179-182.

Lehmann, J., Borgo, S., Masolo, C., Gangemi, A. (2004). Causality and Causation in DOLCE. In: *Formal Ontology in Information Systems (FOIS 2004)*. Varzi, A.C., Vieu L. (eds.). IOS Press Amsterdam, 2004, pp273-284.

Lenzerini, M. (2002). Data Integration: A Theoretical Perspective. *21st Symposium on Principles of Database Systems*, 2002. pp233-246.

Lin, T.Y. (2006). Toward a Theory of Granular Computing. *IEEE International Conference on Granular Computing (GrC06)*. 10-12 May 2006, Atlanta, USA. IEEE Computer Society.

Lind, M. (1999). Making sense of the abstraction hierarchy. *Cognitive Science Approaches to Process Control (CSAPC99)*, Villeneuve d'Ascq, France 21-24 September, 1999.

Lorenz, P., Eck, J. (2005). Metagenomics and industrial applications. *Nature Reviews Microbiology*, 2005, **3**: 510-516.

Luján-Mora, S., Trujillo, J., Song, I. (2002). Multidimensional Modeling with UML Package Diagrams. *Proceedings of the 21st International Conference on Conceptual Modeling*. London: Springer-Verlag, Lecture Notes in Computer Science **2503**, 199-213.

Luján-Mora, S., Trujillo, J., Song, I. (2006). A UML profile for multidimensional modeling in data warehouses. *Data & Knowledge Engineering*, 2006, **59**(3): 725-769 .

MacLeod, M.C., Rubenstein, E.M. (2005). Universals. *The Internet Encyclopedia of Philosophy*. 2005. http://www.iep.utm.edu/u/universa.htm.

Maddison, D.R., Schulz, K.-S. (ed.). (2004). The Tree of Life Web Project. 2004. http://tolweb.org.

Malinowski, E., Zimányi, E. (2006). Hierarchies in a multidimensional model: From conceptual modeling to logical representation. *Data & Knowledge Engineering*, 2006, **59**(2): 348-377.

Mani, I. (1998). A theory of granularity and its application to problems of polysemy and underspecification of meaning. In: *Principles of Knowledge Representation and Reasoning*: Proceedings of the Sixth International Conference (KR98), A.G. Cohn, L.K. Schubert, and S.C. Shapiro (eds.). San Mateo: Morgan Kaufmann. 1998. pp245-255.

Masolo, C., Borgo, S., Gangemi, A., Guarino, N., Oltramari, A. (2003). *Ontology Library*. WonderWeb Deliverable D18 (ver. 1.0, 31-12-2003). http://wonderweb.semanticweb.org. 2003.

Masolo, C., Vieu, L., Bottazzi, E., Catenacci, C., Ferrario, R., Gangemi, A., Guarino, N. (2004). Social Roles and their Descriptions. *Ninth International Conference on the Principles of Knowledge Representation and Reasoning (KR2004)*. 2004.

McCalla, G.I., Greer, J.E., Barrie, B., Pospisil, P. (1992). Granularity Hierarchies. *Computers and Mathematics with Applications: Special Issue on Semantic Networks*, 1992, **23**: 363-376.

McGuinness, D. L., van Harmelen, F. (2004). OWL Web Ontology Language Overview. W3C Recommendation. http://www.w3.org/TR/owl-features/.

Melhorn, H. (ed.). (2004). *Encyclopedic reference of parasitology*. 2nd ed. Springer: Heidelberg, 2004.

Mencar, C., Castellanoa, G., Fanellia, A.M. (2007). Distinguishability quantification of fuzzy sets. *Information Sciences*, 2007, **177**(1): 130-149.

Mendel, J., Dongrui Wu, D. (2007). Perceptual Reasoning: A New Computing With Words Engine. *IEEE International Conference on Granular Computing (GrC2007)*, San Francisco, November 2-4, 2007. IEEE Computer Society, 446-451.

Möller, R. (1996). A functional layer for Description Logics: knowledge representation meets object-oriented programming. *Proceedings of Object-Oriented Programming, Systems, Languages and Applications (OOPSLA'96)*.

Montecucco, C., Schiavo, G. (1995). Structure and function of tetanus and botulinum neurotoxins. *Quarterly Review of Biophysics*, 1995, **4**: 423-72.

Mork, P., Brinkley, J.F., Rosse, C. (2003). OQAFMA Query Agent for the Foundational Model of Anatomy: a prototype for providing flexible and efficient access to large semantic networks. *Journal of Biomedical Informatics*, 2003, **36**: 501-517.

Mota, E., Haggith, M. Smaill, A., Robertson, D. (1995). Time granularity in simulation models of ecological systems. *Proceedings of the IJCAI Workshop on Executable Temporal Logics*, Montreal, Canada.

Motschnig-Pitrik, R., Kaasbøll, J. (1999). Part-Whole Relationship Categories and Their Application in Object-Oriented Analysis. *IEEE Transactions on Knowledge and Data Engineering*, 1999, **11**(5): 779-797.

Myhre, S., Tveit, H., Mollestad, T., Laegreid, A. (2006). Additional Gene Ontology structure for improved biological reasoning. *Bioinformatics*, 2006, **22**(16): 2020-2027.

Nicholson, J.K., Holmes, E., Lindon, J.C., Wilson, I.D. (2004). The challenges of modelling mammalian biocomplexity. *Nature Biotechnology*, 2004, **22**(10): 1268-1274.

Nicholson, J.K. (2006). Global systems biology, personalized medicine and molecular epidemiology. *Molecular Systems Biology*, 2006, **3**: 1-6.

Ning, P., Wang, X.S., Jajodia, S. (2002). An Algebraic Representation of Calendars. *Annals of Mathematics and Artificial Intelligence*, 2002, **63**(1-2): 5-38.

Noy, N.F., Musen, M.A. (2002). Evaluating Ontology-Mapping Tools: Requirements and Experience. *Workshop on Evaluation of Ontology Tools at EKAW'02 (EON2002)*.

Object Management Group. (2005). *Unified Modeling Language: Superstructure*. v2.0. formal/05-07-04. http://www.omg.org/cgi-bin/doc?formal/05-07-04.

O'Connor, T., Wong, H.Y. (2005). Emergent Properties. *The Stanford Encyclopedia of Philosophy* (Summer 2005 Edition), Zalta, E.N. (ed.). http://plato.stanford.edu/archives/sum2005/entries/properties-emergent/.

Odell, J.J. (1998). *Advanced Object-Oriented Analysis & Design using UML*. Cambridge: Cambridge University Press. 1998.

Opdahl, A., Henderson-Sellers, B., Barbier F. (2001). Ontological Analysis of whole-part relationships in OO-models. *Information and Software Technology*, 2001, **43**, 387-399.

Pandurang Nayak, P., Levy, A.Y. (1995). A semantic theory of abstractions. In: *Proceedings of the Interna-*

tional Joint Conference on Artificial Intelligence, Mellish, C. (ed.). San Mateo: Morgan Kaufmann, 1995. pp196-203.

Parent, C., Spaccapietra, S., Zimányi, E. (2006a). Conceptual modeling for traditional and spatio-temporal applications—the MADS approach. Berlin Heidelberg: Springer Verlag. 2006.

Parent, C., Spaccapietra, S., Zimányi, E. (2006b). The MurMur project: modeling and querying multi-representation spatio-temporal databases. *Information Systems*, **31**(8): 733 - 769.

Passel, M.W.J. van. (2006). *Anomalous DNA in prokaryotic genomes*. PhD Thesis, Department of Medical Micorbiology, AMC, University of Amsterdam, the Netherlands. 2006.

Paton, R., Gregory, R., Vlachos, C., Saunders, J., Wu, H. (2004). Evolvable social agents for bacterial systems modeling. *IEEE Transactions on Nanobioscience*, 2004, **3**(3): 208-216.

Patterson, D.J. (2000). The other protists. *The Tree of Life Web Project*. 2000. http://tolweb.org/tree?group=The_other_protists&contgroup=Eukaryotes.

Peters, J.F., Skowron, A., Ramanna, S., Synak, P. (2002). Rough sets and information granulation. In: T.B. Bilgic, D. Baets, and O. Kaynak (eds.), *Proceedings of 10th International Fuzzy Systems Association World Congress*, Springer-Verlag, Lecture Notes in Artificial Intelligence **2715**, 370-377.

Plant Ontology Consortium. (2002). The Plant Ontology Consortium and Plant Ontologies. *Comparative and Functional Genomics*, 2002, **3**: 137-142.

Poggi, A., Ruzzi, M. (2004). Filling the gap between data federation and data integration. In: di Pula, S.M. (ed.): *Proceedings of the 12th Italian Symposium on Advanced Database Systems (SEBD'04)*, Cagliari, Italy. 2004. pp270-281.

Pontow, C., Schubert, R. (2006). A mathematical analysis of theories of parthood. *Data & Knowledge Engineering*, 2006, **59**: 107-138.

Popper, K.R. (1996). *The myth of the framework—in defence of science and rationality*. London: Routledge. 1996. 229p.

Prinz, J. (1998). Vagueness, Language, and Ontology. *Electronic Journal of Analytical Philosophy*, 1998, 6 (Spring). http://ejap.louisiana.edu/EJAP/1998/prinz98.html.

Qiu, T., Chen, X., Liu, Q., Huang, H. (2007). A Granular Space Model For Ontology Learning. *IEEE International Conference on Granular Computing (GrC2007)*, San Francisco, November 2-4, 2007. IEEE Computer Society, 61-65.

Ram, S., Snodgrass, R.T., Khatri, V., Hwang, Y. (2001). DISTIL: a design support environment for conceptual modeling of spatio-temporal requirements. In: *Proceedings of the 20th International Conference on Conceptual Modeling (ER'01)*. Springer, Berlin: Lecture Notes in Computer Science **2224**, 70-83.

Rector, A. (2003). Modularisation of domain ontologies implemented in description logics and related formalisms including OWL. *Proceedings of Knowledge Capture 2003 (K-CAP03)*, Genari J (ed.), 2003. pp121-129.

Rector, A., Rogers, J., Bittner, T. (2006). Granularity, scale and collectivity: When size does and does not matter. *Journal of Biomedical Informatics*, 2006, **39**(3): 333-349.

Reformat, M., Pedrycz, W., Pizzi, N. (2004). Building a software experience factory using granular-based models. *Fuzzy Sets and Systems*, 2004, **145**: 111-139.

Ribba, B., Colin, T., Schnell, S. (2006). A multiscale mathematical model of cancer, and its use in analyzing irradiation therapies. *Theoretical Biology and Medical Modelling*, 2006, **3**: 7.

Richmond, B. (1991). *Systems thinking - four key questions*. High Performance Systems. 1991.

Ricklefs, R.E., Miller, G.L. (2000). *Ecology*. W.H. Freeman, 4th ed., New York.

Rigaux, P., Scholl, M. (1995). Multi-scale partitions: applications to spatial and statistical databases. In: *Proceedings of the 4th International Symposium on Advances in Spatial Databases (SSD'95)*. Springer, Berlin: Lecture Notes in Computer Science **951**, 170-183.

Rodal, A.A., Sokolova, O., Robins, D.B., Daugherty, K.M., Hippenmeyer, S., Riezman, H., Grigorieff, N., Goode, B.L. (2005). Conformational changes in the Arp2/3 complex leading to actin nucleation. *Nature Structural & Molecular Biology*, 2005, **12**: 26-31.

Rosse, C. (2005). The Foundational Role of Anatomy for Biomedical Ontologies. *Ontology and Biomedical Informatics*, 28 April - 3 May 2005, Rome.

Rosse, C., Mejino, J.L.V. (2003). A reference ontology for biomedical informatics: the foundational model of anatomy. *Journal of Biomedical Informatics*, 2003, **36**: 478-500.

Rossetto, O., de Bernard, M., Pellizzari, R., Vitale, G., Caccin, P., Schiavo, G., Montecucco, C. (2000). Bacterial toxins with intracellular protease activity. *Clinica Chimica Acta*, 2000, **291**(2): 189-199.

Rutishauser, R., Moline, P. (2005). Evo-devo and the search for homology ("sameness") in biological systems. *Theory in Biosciences*, 2005, **124**: 213-241.

Salthe, S.N. (2001). Summary of the Principles of Hierarchy Theory. November 2001.

http://www.nbi.dk/∼natphil/salthe/hierarchy_th.html (accessed: 10-10-2005).

Salthe, S.N. (1985). *Evolving hierarchical systems—their structure and representation*. New York: Columbia University Press. 1985. 343p.

Schlegel, H.G. *General Microbiology*. 7th ed. Cambridge: Cambridge University Press, 1995.

Sattler, U. (1995). A concept language for an engineering application with part-whole relations. In: *Proceedings of the international workshop on description logics*, Borgida, A., Lenzerini, M., Nardi, D., Nebel, B. (Eds.), 1995. pp119-123.

Schleper, C., Jurgens, G., Jonuscheit, M. (2005). Genomic studies of uncultivated Archae. *Nature Reviews Microbiology*, 2005, **3**: 479-488.

Schmidt-Schauss, M. Subsumption in KL-ONE is undecidable. In: *Proceedings of 1st Conference on Knowledge Representation and Reasoning (KR'89)*, 1989. pp421-431.

Schmidtke, H.R. (2003). Restricting scalability with granularity. *Workshop on Fundamental Issues in Spatial and Geographical Ontology (COSIT'03)*. 23-9-2003, Ittingen, Switzerland.

Schmidtke, H.R. (2005). Granularity as a Parameter of Context. In: M*odeling and Using Context: 5th International and Interdisciplinary Conference CONTEXT 2005*, Paris, France, July 5-8, 2005. Dey A, Kokinov B, Leake D, Turner R (eds.). Lecture Notes in Computer Science **3554**, 450-463.

Schmidtke, H.R., Woo, W. (2007). A size-based qualitative approach to the representation of spatial granularity. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI'07)*, 563-568.

Schmitt, C.P., Haaland, P. (2004). Issues in pathway bioinformatics. *2004 IEEE North Carolina Symposium on Biotechnology and Bioinformatics (NCSBB)*, October 12-15, 2004.

Schneider, L. (2003). Designing Foundational Ontologies. The Object-Centered High-level Reference Ontology OCHRE as a Case Study. In: Song, I.-Y.; Liddle, S.W.; Ling, T.W.; Scheuermann, P. (eds.), *Conceptual Modeling - ER 2003, 22nd International Conference on Conceptual Modeling*. Lecture Notes in Computer Science 2813. Heidelberg: Springer, 91-104.

Schulz, S., Kumar, A., Bittner, T. (2006). Biomedical ontologies: What part-of is and isn't. *Journal of Biomedical Informatics*, 2006, **39**(3): 350-361.

Schulz, S., Hahn, U., Romacker, M. (2000). Modeling Anatomical Spatial Relations with Description Logics. *Proceedings of the AMIA Symposium 2000*. Overhage, J.M. (ed.), 779-83.

Serafini, L., Roelofsen, F. (2004). Complexity of contextual reasoning. In: *Proceeding the 19th National Conference on Artificial Intelligence (AAAI-04)*. Morgan Kaufmann. 2004.

Seshadri, R., Kravitz, S.A., Smarr, L., Gilna, P., Frazier, M. (2007). CAMERA: A Community Resource for Metagenomics. *PLoS Biology*, 2007, **5**(3): e75.

Shachar, O., Linial, M. (2004). A robust method to detect structural and functional remote homologues. *Proteins: Structure, Function, and Bioinformatics*, 2004, **57**: 532-538.

Shanks, G., Tansley, E., Weber, R. (2004). Representing composites in conceptual modeling. *Communications of the ACM*, 2004, **47**(7): 77-80.

Shoop, E., Silverstein, K.A.T., Johnson, J.E., Retzel, E.F. (2001). MetaFam: a unified classification of protein families. *Bioinformatics*, 2001, **17**(3): 262-271.

Silberstein, M., McGreever, J. (1999). The search for ontological emergence. *The Philosophical Quarterly*, 1999, **49**(195): 182-200.

Skowron, A., Peters, J.F. (2003). Rough sets: Trends and challenges - plenary paper. In: Wang, G., Liu, Q., Yao, Y., Skowron, A. (eds.), *Proceedings of RSFDGrC 2003: Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing*. Springer-Verlag, Lecture Notes in Artificial Intelligence **2639**, 25-34.

Skowron , A., Stepaniuk, J. (2007). Modeling of High Quality Granules. In: *Proceedigns of the International Conference Rough Sets and Intelligent Systems Paradigms (RSEISP 2007)*, Kryszkiewicz, M., Peters, J.F., Rybinski, H., Skowron, A. (Eds.). Springer, Berlin, Lecture Notes in Artificial Intelligence **4585**, 300-309.

Smith, B. (2004). Beyond Concepts, or: Ontology as Reality Representation. Varzi,A., Vieu, L. (eds.), *Formal Ontology and Information Systems. Proceedings of the Third International Conference (FOIS 2004)*, Amsterdam: IOS Press, 2004, 73-84.

Smith, B., Ashburner, M., Rosse, C., Bard, J., Bug, W., Ceusters, W., Goldberg, L.J., Eilbeck, K., Ireland, A., Mungall, C.J., The OBI Consortium, Leontis, N., Rocca-Serra, A.B., Ruttenberg, A., Sansone, S-A., Shah, M., Whetzel, P.L., Lewis, S. (2007). The OBO Foundry: Coordinated Evolution of Ontologies to Support Biomedical Data Integration. *Nature Biotechnology*, **25**(11): 1251-1255.

Smith, B., Brogaard, B. (2002). Quantum Mereotopology. *Annals of Mathematics and Artificial Intelligience*, **35**(1-2): 153-175.

Smith, B., Ceusters, W., Klagges, B., Köhler, J., Kumar, A., Lomax, J., Mungall, C., Neuhaus, F., Rector, A.L., Rosse, C. (2005). Relations in biomedical ontologies. *Genome Biology*, 2005, **6**:R46.

Smith, B., Kusnierczyk, W., Schober, D., Ceusters, W. (2006). Towards a reference terminology for ontology research and development in the biomedical domain. *Proceedings of KR-MED 2006*, Baltimore, USA, 2006.

Sonamthiang, S., Cercone, N., Naruedomkul, K. (2007). Discovering Hierarchical Pattern of Students' Learning Behavior in Intelligent Tutoring Systems. *IEEE International Conference on Granular Computing (GrC2007)*, IEEE Computer Society, 2007, 485-489.

Sontag, E.D. (2004). Some new directions in control theory inspired by systems biology. *Systems Biology*, 2004, **1**(1): 9-18.

Sorensen, R. (2003). Vagueness. *The Stanford Encyclopedia of Philosophy*. Fall 2003 Edition, Zalta, E.N. (ed.). http://plato.stanford.edu/archives/fall2003/entries/vagueness/.

Sorokine, A., Bittner, T., Renschler, C. (2004). Ontological investigation of ecosystem hierarchies and formal theory for multiscale ecosystems classifications. *Proceedings of GIScience04*, Lecture Notes in Computer Science **3234**. 2004.

Sorokine, A., Bittner, T., Renscher, C. (2006). Ontological investigation of ecosystem hierarchies and formal theory for multiscale ecosystem classifications. *Geoinformatica*, 2006, 10(3): 313-335.

Sowa, J.F. (2000). *Knowledge representation: logical, philosophical, and computational foundations*. China Machine Press. 2000. 594p.

Stell, J.G. (2003a). Qualitative Extents for Spatio-Temporal Granularity. *Spatial Cognition and Computation*, 2003, **3**(2-3): 119-136.

Stell, J.G. (2003b). Granularity in change over time. In: *Foundations of Geographic Information Science*, Duckham, M, Goodchild, MF, Worboys, MF (eds.), London: Taylor & Francis Books, 2003, 95-115.

Stell, J., Worboys, M. (1998). Stratified map spaces: a formal basis for multi-resolution spatial databases. In: *Proceedings of the 8th International Symposium on Spatial Data Handling (SDH'98)*, International Geographical Union, 180-189.

Straccia, U. (2006). A Fuzzy Description Logic for the Semantic Web. In: *Capturing Intelligence: Fuzzy Logic and the Semantic Web*, Sanchez, E., ed., Elsevier, 2006.

Stryer, L. (1988). *Biochemistry*. New York: WH Freeman and Co, 3rd ed. 1089p.

Swoyer, C. (2000). Properties. *The Stanford Encyclopedia of Philosophy*, (Winter 2000 Edition), Zalta, E.N. (ed.), http://plato.stanford.edu/archives/win2000/entries/properties/.

Tan, H.B.K, Hao, L., Yang, Y. (2003). On formalizations of the whole-part relationship in the Unified Modeling Language. *IEEE Transactions on Software Engineering*, **29**(11): 1054-1055.

Tange, H.J., Schouten, H.C, Kester, A.D.M., Hasman, A. (1998). The Granularity of Medical Narratives and Its Effect on the Speed and Completeness of Information Retrieval. *Journal of the American Medical Informatics Association*, 1998, **5**(6): 571-582.

Tavana, M., Joglekar, P., Redmond, M.A. (2007). An automated entity-relationship clustering algorithm for conceptual database design. *Information Systems*, 2007, **32**(5): 773-792.

Tett, P., Wilson, H. (2000). From biogeochemical to ecological models of marine microplankton. *Journal of Marine Systems*, 2000, **25**: 431-446.

Theise, N.D. (2005). Now you see it, now you don't. *Nature*, **435**: 1165.

Toyoda, T., Wada, A. (2004). Omic space: coordinate-based integration and analysis of genomic phenomic interactions. *Bioinformatics*, 2004, **20**(11): 1759-65.

Tsumoto, S. (2007). Mining Diagnostic Taxonomy and Diagnostic Rules for Multi-Stage Medical Diagnosis from Hospital Clinical Data. *IEEE International Conference on Granular Computing (GrC2007)*, IEEE Computer Society, 2007, 611-616.

Tzitzikas, Y., Hainaut, J-L. (2006). On the visualization of large-sized ontologies. In: *Proceedings of the working conference on Advanced Visual Interfaces 2006 (AVI2006)*, Venice, Italy. 99-102.

Tziviskou, C., Keet, C.M. (2007). A Meta-Model for Ontologies with ORM2. *Third International Workshop on Object-Role Modelling (ORM 2007)*, Algarve, Portugal, Nov 28-30, 2007. In: *OTM Workshops 2007*. Meersman, R., Tari, Z, Herrero, P. et al (Eds.), Berlin: Springer-Verlag, Lecture Notes in Computer Science **4805**, 624-633.

Varzi, A.C. (2004a). Mereology. *The Stanford Encyclopedia of Philosophy*. (Fall 2004 Edition), Zalta, E.N. (ed.). http://plato.stanford.edu/archives/fall2004/entries/mereology/.

Varzi, A.C. (2004b). Boundary. *The Stanford Encyclopedia of Philosophy* (Spring 2004 Edition), Zalta, E.N. (ed.). http://plato.stanford.edu/archives/spr2004/entries/boundary/.

Varzi, A.C. (2006a). The formal ontology of space: parts, wholes, and locations. In: *The Logic of Space*, Aiello, M., Pratt-Hartmann I., van Benthem J. (eds.). Dordrecht: Kluwer Academic Publishers. Chapter 1, 104p. draft http://www.columbia.edu/~av72/papers/Space_2006.pdf, date accessed: 16-5-2006.

Varzi, A.C. (2006b). A Note on the Transitivity of Parthood. *Applied Ontology*, 2006, **1**: 141-146. http://www.columbia.edu/∼av72/papers/AO_2006.pdf date accessed: 17-5-2006.

Vernieuwe, H., Verhoest, N.E.C., De Baets, B., Hoeben, R., De Troch, F.P. (2007). Cluster-based fuzzy models of groundwater transport. *Advances in Water Resources*, 2007, **30**(4): 701-714.

Vieu, L., Aurnague, M. (2005). Part-of Relations, Functionality and Dependence. In: M. Aurnague, M. Hickmann, L. Vieu (eds.), *Categorization of Spatial Entities in Language and Cognition*. Amsterdam: John Benjamins. 2005.

Wang, G., Van Dam, A.P., Schwartz, I., Dankert, J. (1999). Molecular Typing of Borrelia burgdorferi Sensu Lato: Taxonomic, Epidemiological, and Clinical Implications. *Clinical Microbiology Reviews*, 1999, **12**(4): 633-653.

Weiss, R.A., McMichael, A.J. Social and environmental risk factors in the emergence of infectious diseases. *Nature Medicine*, 2004, **10**: S70-S76.

Wessel, M. (2001). Obstacles on the way to qualitative spatial reasoning with description logics: some undecidability results. In: *Proceedings of the International Workshop in Description Logics*, 2001.

Westerhoff, H.V., Palsson, B.O. (2004). The evolution of molecular biology into systems biology. *Nature Biotechnology*, 2004, **22**(10): 1249-1252.

Widell, A., Elmud, H., Persson, M.H., Jonsson, M. (1996). Transmission of hepatitis C via both erythrocyte and platelet transfusions from a single donor in serological window-phase of hepatitis C. *Vox Sang*, 1996, **71**(1): 55-7.

Wimsatt, W.C. (1995). The ontology of complex systems: Levels of Organization, Perspectives, and Causal Thickets. *Canadian Journal of Philosophy*, 1994, **20**: 207-274.

Wimsatt, W.C. (2000). Emergence as Non-Aggregativity and the Biases of Reductionisms. *Foundations of Science*, 2000, **5**(3):269-297.

Winther, R.S. (2006). Parts and Theories in Compositional Biology. *Biology and Philosophy*, 2006, **21**(4): 471-499.

Winston, M.E., Chaffin, R., Herrmann, D. (1987). A taxonomy of partwhole relations. *Cognitive Science*, 1987, **11**(4): 417-444.

Wolstencroft, K., Stevens, R., Haarslev, V. (2007). Applying OWL reasoning to genomic data. In Baker, C., and Cheung, H., eds., *Semantic Web: revolutionizing knowledge discovery in the life sciences*. Springer: New York, 2007. pp225-248.

Yao, J.T. (2005). Information granulation and granular relationships. *IEEE International Conference on Granular Computing (GrC2005)*, 2005, **1**: 326-329.

Yao, J.T. (2007a). A ten-year review of granular computing. *IEEE International Conference on Granular Computing (GrC2007)*, IEEE Computer Society, 2007, 734-739.

Yao, Y.Y. (2004a). A partition model of granular computing. *Lecture Notes in Computer Science Transactions on Rough Sets*, 2004, **1**: 232-253.

Yao, Y.Y. (2004b). Granular Computing. *Computer Science (Ji Suan Ji Ke Xue)*, Proceedings of The 4th Chinese National Conference on Rough Sets and Soft Computing, 2004, **31**: 1-5.

Yao, Y.Y. (2005a). Perspectives of Granular Computing. *IEEE Conference on Granular Computing (GrC2005)*, **1**: 85-90.

Yao, Y.Y. (2006). Three perspectives of granular computing. *Proceedings of International Forum on Theory of GrC from Rough Set Perspective, Journal of Nanchang Institute of Technology*, 2006, **25**(2): 16-21.

Yao, Y.Y. (2007b). The art of granular computing. In: *Proceedings of the International Conference on Rough Sets and Emerging Intelligent Systems Paradigms*, 2007.

Yao, Y.Y. (2007c). Structured Writing with Granular Computing Strategies. *IEEE International Conference on Granular Computing (GrC2007)*, IEEE Computer Society, 2007, 72-77.

Yao, Y.Y., Liau, C.-J. (2002). A generalized decision logic language for granular computing. *2002 IEEE World Congress on Computational Intelligence (FUZZ-IEEE'02)*, pp. 1092-1097, 2002.

Yu, X., Lau, E., Vicente, K.J., Carter, M.W. (2002). Toward theory-driven, quantitative performance measurement in ergonomics science: the abstraction hierarchy as a framework for data analysis. *Theoretical Issues in Ergonomics Science*, 2002, **3**(2): 124-142.

Zadeh, L.A. (1997). Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic. *Fuzzy Sets and Systems*, 1997, **90**(2): 111-127.

Zadeh, L.A. (2002). From computing with numbers to computing with words—from manipulations of measurements to manipulation of perceptions. *International Journal of applied mathematics and computer science*, 2002, **12**(3): 307-324.

Zhang, J., Silvescu, A., Honavar, V. (2002). *Ontology-Driven Induction of Decision Trees at Multiple Levels of Abstraction*. Technical Report ISU-CS-TR 02-13, Computer Science, Iowa State University. 2002.

http://archives.cs.iastate.edu/documents/disk0/00/00/02/91/.

Zhang, S., Bodenreider, O., Golbreich, C. (2006). Experience in reasoning with the Foundational Model of Anatomy in OWL DL. In: Altman RB, Dunker AK, Hunter L, Murray TA, Klein TE, (Eds.). *Pacific Symposium on Biocomputing (PSB'06)*, World Scientific, 2006, 200-211.

Zhou, S., Jones, C.B. (2003). A multi-representation spatial data model. *Proceedings of the 8th International Symposium on Advances in Spatial and Temporal Databases (SSTD 2003)*, Springer, Berlin, Lecture Notes in Computer Science **2750**, 394-411.

Zhou, X., Prasher, S., Sun, S., Xu, K. (2004). Multiresolution spatial databases: making web-based spatial applications faster. In: *Proceedings of the 6th Asia-Pacific Web Conference (APWeb 2004)*, Springer, Berlin, Lecture Notes in Computer Science **3007**, 36-47.

Zhou, Y., Young, J.A., Santrosyan, A., Chen, K., Yan, S.F., Winzeler, E.A. (2005). In silico gene function prediction using ontology-based pattern identification. *Bioinformatics*, 2005, **21**(7): 1237-1245.

Zukerman, I., Albrecht, D., Nicholson, A., Doktor, K. (2000). Trading off granularity against complexity in predictive models for complex domains. In: *Proceedings of the 6th International Pacific Rim Conference on Artificial Intelligence (PRCAI 2000).*

Internet resources

Bad Bug Book. http://www.cfsan.fda.gov/∼mow/intro.html. Last accessed on: 24-10-2007.

BFO: Basic Formal Ontology. http://www.ifomis.uni-saarland.de/bfo. Last accessed on: 27-6-2007.

BioPAX: Biological Pathways Exchange. http://www.biopax.org/. Last accessed on: 24-10-2007.

LMS: Contextual reasoning. http://dit.unitn.it/∼context. Last accessed on: 25-10-2007.

ChEBI: Chemical Entities of Biological Interest. http://www.ebi.ac.uk/chebi/ (release 2038, 31-10-2007). Last accessed on: 8-11-2007.

E-cell project. http://www.e-cell.org/ecell/. Last accessed on: 25-10-2007.

Fact++ DL reasoner. http://owl.man.ac.uk/factplusplus/. Last accessed on: 25-10-2007.

FMA-lite. http://obo.sourceforge.net/cgi-bin/detail.cgi?fma_lite. Date accessed: 14-2-2007.

FMA: Foundational Model of Anatomy. 2003. http://fme.biostr.washington.edu:8089/FME/index.html. Last accessed on: 25-10-2007.

GenBank. http://www.psc.edu/general/software/packages/genbank/genbank.html. Last accessed on: 24-10-2007.

GOC: Gene Ontology Consortium. http://www.geneontology.org. Last accessed on: 24-10-2007.

GoPubMed. http://www.gopubmed.org. Last accessed on: 22-10-2007.

GO-slim. http://www.geneontology.org/GO.slims.shtml. Last accessed on: 27-6-2007.

GrOWL: Graphical OWL. http://esd.uvm.edu/dmaps/growl/. Last accessed: 13-11-2007.

iCOM. http://www.inf.unibz.it/∼franconi/icom. Last accessed on: 24-10-2007.

ICD 10: International Classification of Diseases. 2003. http://www.who.int/classifications/icd/en/. Last accessed on: 25-10-2007.

ISEE Systems. http://www.iseesystems.com. Last accessed on: 25-10-2007.

KEGG: Kyoto Encyclopedia of Genes and Genomes. http://www.genome.jp/kegg/. Last accessed on: 25-10-2007.

KIM browser plugin: http://www.ontotext.com/kim/plugin/index.html and user guide: http://www.ontotext.com/kim/plugin/guide.html. Last accessed on: 7-11-2007.

Mace4 and Prover9. http://www.cs.unm.edu/ mccune/prover9/. Last accessed on: 17-10-2007.

MurMur Project. http://lbdwww.epfl.ch/e/MurMur/. Last accessed on: 25-10-2007.

National Center for Infectious Diseases. http://www.cdc.gov/ncidod/.

NIST: National Institute of Standards and Technology (NIST) Reference on Constants, Units, and Uncertainty. http://physics.nist.gov/cuu/Units/second.html. Last accessed on: 25-10-2007.

OBO Foundry. http://obofoundry.org. Last accessed on: 24-10-2007.

ORM: Object-Role Modeling. http://www.orm.net. Last accessed on: 18-10-2007.

OntoViz. http://protege.cim3.net/cgi-bin/wiki.pl?OntoViz. Last accessed on: 25-10-2007.

OpenGALEN. http://www.opengalen.org/. Last accessed on: 25-10-2007.

OQAFMA: Ontology Query Agent for the Foundational Model of Anatomy. http://sig.biostr.washington.edu/projects/oqafma/. Last accessed on: 24-10-2007.

OWL: Web Ontology Language. W3C. http://www.w3.org/TR/owl-features/. Last accessed on: 25-10-2007.

OWL 1.1: Web Ontology Language. http://webont.org/owl/1.1/ (Editor's draft of 6-4-2007). Last accessed on: 25-10-2007.

OWLViz. http://www.co-ode.org/downloads/owlviz/. Last accessed on: 13-11-2007.

PACE: Programmable Artificial Cell Evolution. http://134.147.93.66/bmcmyp/Data/PACE/Public. Accessed in 2006, link defunct on 25-10-2007.

PathInfo. http://www.vbi.vt.edu/pathport/pathinfo. Last accessed on: 24-10-2007.

PathwayAssist, Ingenuity. http://www.ingenuity.com/products/pathways_analysis.html. Last accessed on: 24-10-2007.

Pathologie Online. http://www.pathologie-online.de/.

Pellet OWL-DL reasoner. http://pellet.owldl.com/. Last accessed on: 25-10-2007.

RDF: Resource Description Framework. http://www.w3.org/RDF/. Last accessed on: 24-10-2007.

SAEL: SOFG Anatomy Entry List. http://www.sofg.org/sael/. Last accessed on: 25-10-2007.

SemTalk, Semtation. http://www.semtalk.com. Last accessed on: 25-10-2007.

Snomed CT. http://www.ihtsdo.org/our-standards/snomed-ct/. Last accessed on: 25-10-2007.

UniProt. http://beta.uniprot.org/. Last accessed on: 24-10-2007.

W3C HCLS IG: World Wide Web Consortium's Semantic Web Health Care and Life Sciences Interest Group. http://www.w3.org/2001/sw/hcls/. Last accessed on: 25-10-2007.

WSDL: Web Service Definition Language v 1.1. http://www.w3.org/TR/wsdl. Last accessed: 12-12-2007.

WSML: Web Service Modeling Language v0.21. http://www.wsmo.org/TR/d16/d16.1/v0.21/. Last accessed: 12-12-2007.

WSMO: Web Service Modeling Ontology D2v1.3. http://www.wsmo.org/TR/d2/v1.3/. Last accessed: 12-12-2007.

WSMX: Web Service eXecution environment. http://www.wsmx.org/. Last accessed: 12-12-2007.

Index

Abbreviations

BFO Basic Formal Ontology
CASE Computer-Aided Software Engineering
DL Description Logics
DOLCE Descriptive Ontology for Linguistic and Cognitive Engineering
DWH Data WareHouse
ER Entity-Relationship modeling
FMA Foundational Model of Anatomy
FOL First Order (predicate) Logic
GAV Global As View
GBA Guilt By Association
GEM General Extensional Mereology
GIS Geographic Information System
$\mathcal{GMD}$ Generalized MultiDimensional model
GO Gene Ontology
KEGG Kyoto Encyclopedia of Genes and Genomes
LAV Local As View
MADS Modeling of Application Data with Spatio-temporal features
MCS MultiContext System
MSD Multi-Structural Databases
MSU Most Specific Universal
NORMA Neumont ORM Architect
OBO Open Biomedical Ontologies
OCHRE Object-Centred High-level Reference Ontology
OCL Object Constraint Language
OODBMS Object-Oriented DataBase Management System
OQAFMA Ontology Query Agent for the Foundational Model of Anatomy
ORM Object-Role Modelling
OWL Web Ontology Language
RDBMS Relational DataBase Management System
RDF Resource Description Framework
SMS Second Messenger System
SPARQL RDF Query Language
SQL Structured Query Language
TGP Theory of Granular Partitions
TOG Theory of Granularity
UoD Universe of Discourse
UML Unified Modeling Language
USDA United States Department of Agriculture
WSML Web Service Modeling Language
WSMO Web Service Modeling Ontology
XML eXtensible Markup Language

Index

Curriculum Vitae and Publications

Maria (Marijke) Keet was born on December 8th 1973, in Heeze, the Netherlands. She obtained her VWO diploma [highest level secondary education, scientific specialisation] from Strabrecht College, Geldrop, the Netherlands, in 1992. Afterward, she studied at the Landbouwuniversiteit Wageningen [Wageningen Agricultural University], the Netherlands, and graduated with a Master of Science in Food Science free specialisation in January 1998 after completing theses in Food Microbiology, Molecular Ecology (Microbiology), and Applied Philosophy, and a 6-months internship at the Centro Internacionál de la Papa [International Potato Centre] in Lima, Peru.

Subsequently, she completed an MCSE and commenced employment in industry as an IT systems engineer for Brunel IT in the Netherlands and from 1999 for Compaq and Eurologic Systems in Dublin, Ireland, during which she obtained several industry IT-certificates, such as ITIL and ASE, and started a Bachelor of Science in IT & Computing at the Open University UK in 2001. She graduated with a BSc(Honours), 1st Class Honours, in March 2004 after completing a thesis in conceptual modelling and database development for biology at the Department of Mathematics & Computing. In addition, in the university-year 2002-2003 she studied full-time for a Master of Arts in Peace & Development Studies at the University *of* Limerick, Ireland, and graduated with a MA 1st Class Honours in December 2003 after completion of a thesis on terrorism and game theory at the Department of Government & Society.

After one year as research student at Napier University, Edinburgh, UK, and an internship at the Laboratorio di Ontologia Applicata CNR [Laboratory for Applied Ontology] in Trento, Italy, in 2004, she enrolled in the PhD Programme at the Knowledge Representation meets DataBases Research Centre at the Faculty of Computer Science, Libera Università di Bolzano / Freie Universität Bozen / Free University of Bozen-Bolzano, Italy, in January 2005. Since February 2008 she is employed by the KRDB Research Center in the position of ricercatore a tempo determinato (RTD) [Assistant Professor].

Publications

Peer-reviewed journal, conference, and workshop papers that were written during the PhD programme and are at the time of print published or in publication:

1. Keet, C.M. (2008). Unifying industry-grade class-based conceptual data modeling languages with $\mathcal{CM}_{com}$. *21st International Workshop on Description Logics (DL'08)*, 13-16 May 2008, Dresden, Germany. CEUR-WS (ISSN 1613-0073), **Vol-xx**(accepted). 11p.

2. Artale, A., Keet, C.M. (2008). Essential and mandatory part-whole relations in conceptual data models. *21st International Workshop on Description Logics (DL'08)*. 13-16 May 2008, Dresden, Germany. CEUR-WS (ISSN 1613-0073), **Vol-xx**(accepted). 11p.

3. Keet, C.M., Artale, A. (2008). Representing and Reasoning over a Taxonomy of Part-Whole Relations. *Applied Ontology – Special Issue on Ontological Foundations for Conceptual Models* (ISSN 1570-5838), 2008, **2**(4): (in print, 17p.).

4. Keet, C.M. (2008). A formal comparison of conceptual data modeling languages. *13th International Workshop on Exploring Modeling Methods in Systems Analysis and Design (EMMSAD'08)*. 16-17 June 2008, Montpellier, France. CEUR-WS (ISSN 1613-0073), **Vol-xx**(accepted). 15p.

5. Keet, C.M. (2007). Granulation with indistinguishability, equivalence or similarity. *IEEE International Conference on Granular Computing (GrC2007)*, San Francisco, November 2-4, 2007. IEEE Computer Society (ISBN 0-7695-3032-X), 11-16.

6. Keet, C.M. (2007). Enhancing comprehension of ontologies and conceptual models through abstractions. *10th Congress of the Italian Association for Artificial Intelligence (AI*IA 2007)*, Rome, September 10-13, 2007. Basili, R., Pazienza, M.T. (Eds.), Springer-Verlag Lecture Notes in Artificial Intelligence (ISSN 0302-9743) **LNAI 4733**, 814-822.

7. Tziviskou, C., Keet, C.M. (2007). A Meta-Model for Ontologies with ORM2. *Third International Workshop on Object-Role Modelling (ORM 2007)*, Algarve, Portugal, Nov 18-20, 2007. In: *OTM Workshops 2007*. Meersman, R., Tari, Z., Herrero, P. et al (Eds.), Berlin: Springer-Verlag, Lecture Notes in Computer Science (ISSN 0302-9743) **LNCS 4805**, 624-633.

8. Keet, C.M., Rodríguez, M. (2007). Toward using biomedical ontologies: trade-offs between ontology languages. *AAAI 2007 Workshop on Semantic e-Science (SeS07)*, 23 July 2007, Vancouver, Canada. *AAAI 2007 Technical Report* (ISBN 978-1-57735-338-6), **WS-07-11**, 65-68.

9. Keet, C.M. (2007). Prospects for and issues with mapping the Object-Role Modeling language into $\mathcal{DLR}_{ifd}$. *20th International Workshop on Description Logics (DL'07)*, 8-10 June 2007, Bressanone, Italy. CEUR-WS (ISSN 1613-0073), **Vol-250** / (ISBN 978-88-6046-008-5), 331-338.

10. Keet, C.M., Roos, M., Marshall, M.S. (2007). A survey of requirements for automated reasoning services for bio-ontologies in OWL. *Third International Workshop OWL: Experiences and Directions (OWLED'07)*, 6-7 June 2007, Innsbruck, Austria. CEUR-WS (ISSN 1613-0073), **Vol-258**. 10p.

11. Keet, C.M. (2006). A taxonomy of types of granularity. *IEEE International Conference on Granular Computing (GrC2006)*, Atlanta, USA, May 10-12 2006. IEEE Computer Society (ISBN 1-4244-0134-8), 106-111.

12. Keet, C.M. (2006). Part-whole relations in Object-Role Models. *Second International Workshop on Object-Role Modelling (ORM 2006)*, Montpellier, France, Nov 2-3, 2006. In: *OTM Workshops 2006*. Meersman, R., Tari, Z., Herrero., P. et al. (Eds.), Berlin: Springer-Verlag, Lecture Notes in Computer Science (ISSN 0302-9743) **LNCS 4278**, 1118-1127.

13. Keet, C.M. (2006). Representations of the ecological niche. *Third International Workshop on Philosophy and Informatics (WSPI 2006)*, Saarbrücken, Germany. 3-4 May 2006. Johansson, I., Klein, B., Roth-Berghofer, T. (Eds.), *IFOMIS Reports* (ISSN 1611-4019), **14**, 75-88.

14. Keet, C.M. (2006). Enhancing biological information systems with granularity. *KnowledgeWeb PhD Symposium (KWEPSY06)*, Budva, Montenegro, 17 June 2006. 6p.

15. Keet, C.M. (2005). Using abstractions to facilitate management of large ORM models and ontologies. *International Workshop on Object-Role Modeling (ORM 2005)*. Cyprus, 3-4 November 2005. In: *OTM Workshops 2005*. Halpin, T., Meersman, R. (eds.), Berlin: Springer-Verlag, Lecture Notes in Computer Science (ISSN 0302-9743) **LNCS 3762**, 603-612.

16. Keet, C.M. (2005). Factors affecting ontology development in ecology. *Data Integration in the Life Sciences 2005 (DILS2005)*, Ludäscher, B, Raschid, L. (eds.). San Diego, USA, 20-22 July 2005. Berlin: Springer-Verlag, Lecture Notes in Bioinformatics (ISSN 0302-9743) **LNBI 3615**, 46-62.

17. Keet, C.M., Kumar, A. (2005). Applying partitions to infectious diseases. *XIX International Congress of the European Federation for Medical Informatics (MIE2005)*. Geneva, Switzerland, 28-31 August, 2005. In: *Connecting Medical Informatics and bioinformatics*, Engelbrecht, R., Geissbuhler, A., Lovis, C. Mihalas, G. (eds.). Amsterdam: IOS Press ENMI (ISSN 1861-3179), 2005, **1**(1), 1236-1241.

Papers under review or manuscript in preparation for submission:

1. Artale, A., Keet, C.M. Essential, mandatory, and shared parts in conceptual data models. In: *Innovations in Information Systems modeling: Methods and Best Practices*. IGI Global, Advances in Database Research Series, Halpin, T.A., Proper, H.A., Krogstie, J. (Eds.). (2008, under review, invited book chapter).

2. Keet, C.M. A formal theory to represent granularity. (2008, under review at an international conference, with ISBN publication).
3. Keet, C.M. Toward cross-granular querying over modularized ontologies. (2008, under review at an international workshop, with ISSN publication).
4. Keet, C.M. A formalization of the ecological niche. (2006, under review at an international journal).
5. Keet, C.M. Representing transformation in biomedical ontologies. (manuscript in preparation for submission to an international conference, with ISBN publication).

Symposia & seminar papers that were written and published online during the PhD programme:

1. Keet, C.M. (2006). *Introduction to part-whole relations: mereology, conceptual modelling and mathematical aspects*. KRDB Research Centre Technical Report KRDB06-3, Faculty of Computer Science, Free University of Bozen-Bolzano, Italy. 2 October 2006. / tutorial reader.
2. Keet, C.M. (2006). *Granular information retrieval from the Gene Ontology and from the Foundational Model of Anatomy with OQAFMA*. KRDB Research Centre Technical Report KRDB06-1, Free University of Bozen-Bolzano, 6 April 2006. / SBIOLBD 2006 extended abstract.
3. Keet, C.M. (2005). Current characteristics and historical perspective of Computer Science and IT with/for biology. *CSBio Reader: extended abstracts of the 'CS&IT with/for biology' Seminar Series*, Free University of Bozen-Bolzano, **1**, 1-8.

Technical Reports that were written and published online during the PhD programme:

1. Keet, C.M., Rodríguez, M. (2007). *Comprehensiveness versus Scalability: guidelines for choosing an appropriate knowledge representation language for bio-ontologies*. KRDB Research Centre Technical Report KRDB07-5, Faculty of Computer Science, Free University of Bozen-Bolzano, Italy. 9 May 2007. 8p.
2. Keet, C.M. (2007). *Mapping the Object-Role Modeling language ORM2 into Description Logic language $\mathcal{DLR}_{ifd}$*. KRDB Research Centre Technical Report KRDB07-2, Faculty of Computer Science, Free University of Bozen-Bolzano, Italy. 15 February 2007. / arXiv:cs.LO/0702089v1.
3. Keet, C.M. (2006). *A formal approach for using granularity in the subject domain of infectious diseases*. KRDB Research Centre Technical Report KRDB06-2, Free University of Bozen-Bolzano, 26 April 2006. 24p.
4. Jarrar, M., Keet, C.M., Dongilli, P. (2006). *Multilingual verbalization of ORM conceptual models and axiomatized ontologies*. STARLab Technical Report, Vrije Universiteit Brussel, February 2006.

Editing:

1. Keet, C.M., Franconi, E. (eds.). (2005). *CSBio reader: extended abstracts of the 'CS&IT with/for biology' Seminar Series*. Free University of Bozen-Bolzano, 2005.

Older peer-reviewed papers:

1. Keet, C.M. (2004). Conceptual Modelling and Ontologies for Biology: experiences with the bacteriocin database. *Agropecuaria Conference (Interjoven '04)*. 3-6 June 2004, San José de las Lajas, Cuba.
2. Keet, C.M. (2003). Biological Data and Conceptual Modelling Methods. *Journal of Conceptual Modeling*, **29**, October 2003.
3. Keet, C.M. (2003). Democracy in the European Union. *UL Perspectives, Journal of Political Studies*, 2003, **2**, 71-80.
4. Keet, C.M. (2003). *Towards a resolution of terrorism using game theory - coalitions, negotiations and audience costs*. Dept of Politics and Public Admin Working Papers, University of Limerick. Number 1, December 2003.

A Formal Theory of Granularity

Toward enhancing biological and applied life sciences information systems with granularity

Computationally managing different levels of detail in biological data, information, and knowledge—biological granularity—is indispensable for both dealing advantageously with the huge amounts of data that are being generated by scientists and for structuring the knowledge to analyse and vertically integrate biological data and information across levels of granularity. Managing such databases, knowledge bases, and ontologies effectively and efficiently requires new foundational methodologies to push implementations to the next phase of *in silico* biology.

To address these issues, we move from a data-centric and underspecified treatment of granularity to the conceptual and logical layers, where informally defined elements of granularity have become ontologically motivated modelling constructs proper. This is achieved as follows. First, foundational semantics of granularity are disambiguated and structured in a taxonomy of types of granularity. This taxonomy makes explicit both the ways of granulation and representation, and how entities are organised within a level of granularity. Second, the static components of granularity—such as levels, indistinguishability, and granulation criteria, and how levels and entities relate—were subjected to an ontological analysis and formalised in a satisfiable logical theory, the Theory Of Granularity (TOG), so that an unambiguous meaning is ensured, interesting properties proven, and satisfiability computationally demonstrated. Third, an extensible set of domain- and implementation-independent functions are defined for both the TOG elements to enable granular querying and reasoning over the theory, and for moving between entities residing in different levels through abstraction and expansion functions.

Effectively, granularity is lifted up to a higher layer of abstraction alike conceptual modelling languages do for software design and physical database schemas, thereby having made the representation of granularity domain- and implementation independent. Hence, reusability across implementations can be ensured, which in turn facilitates interoperability among information systems, such as granulation of ontologies, knowledge bases, databases, data warehouses, and biological and geographical information systems.

Printed and bound by DigiPrint, Bolzano