AI Communications Ð The European Journal of Artificial Intelligence, 9 (3) 1996, pp. 109-116

Relating Case-Based Problem Solving and Learning Methods to Task and Domain Characteristics: Towards an Analytic Framework

Klaus-Dieter Althoff
Centre for Learning Systems and Applications
Department of Computer Science
University of Kaiserslautern
D-67653 Kaiserslautern, Germany
email: althoff@informatik.uni-kl.de

Agnar Aamodt
Department of Informatics
College of Arts and Science
University of Trondheim
N-7055 Dragvoll, Norway
email: agnar.aamodt@ifi.unit.no

Abstract

A particular strength of case-based reasoning (CBR) over most other methods is its inherent combination of problem solving with sustained learning through problem solving experience. This is therefore a particularly important topic of study, and an issue that has now become mature enough to be addressed in a more systematic way. To enable such an analysis of problem solving and learning, we have initiated work towards the development of an analytic framework for studying CBR methods. It provides an explicit ontology of basic CBR task types, domain characterisations, and types of problem solving and learning methods. Further, it incorporates within this framework a methodology for combining a knowledge-level, top-down analysis with a bottom-up, case-driven one. In this article, we present the underlying view and the basic approach being taken, the main components of the framework and accompanying methodology, examples of studies recently done and how they relate to the framework.

1. Introduction

Over the last few years substantial progress has been made within the case-based reasoning (CBR) field. The problems we are facing have become more clearly identified, research results have led to improved methods for case retrieval as well as improved approaches to the harder problems of adaptation and learning (see, e.g., the collection of recent papers in ICCBR-95: Veloso & Aamodt, 1995). In the course of this development it has also become clear that a particular strength of CBR over most other methods is its inherent combination of problem solving with sustained learning through problem solving experience. This is therefore a particularly important topic of study, and an issue that has now become mature enough to be addressed in a more systematic way. To enable such an analysis of problem solving and learning, a unified framework for describing, analysing, and comparing various types and aspects of CBR methods is needed.

Integration of learning and problem solving may in general start out from different goals, and be viewed from different perspectives. One example is ãconcept formationÒ as a goal, and the formation and utilisation of operationalisation criteria related to the problem solving task, as the perspective. Another example is ãimproved performanceÒ as a goal, and the improvement of total problem solving speed - for computer and human together - as the perspective. A third example is ãsustained learningÒ, i.e. continuous learning through problem solving experience, as a goal, and the impact of the application problem task on the learning method as a perspective. Many more examples may be given, and for each of them a particular area of overlap, an ãintersection spaceÒ between machine learning (ML) and problem solving (PS) methods can be identified. Within this space, dependencies and other relations between specific ML and PS methods may be described and analysed in a systematic way, provided we have a suitable means to structure the space.

We have initiated work towards the development of a framework and a methodology which defines and makes use of such a structure. Since we are studying CBR methods, the natural focus is on the third of the above goals: Sustained and (continuous) learning from each problem solving experience as a natural part of a problem solver's behaviour. Within a broader perspective of integrated learning and problem solving, it is also natural to start a study of learning as close as possible to the source of learning, namely a concrete experience. Our work is related to some earlier suggestions for analytic CBR frameworks, such as the similarity-focused framework by Richter and Wess (Richter, Wess, 1991), Althoff's analysis of systems for technical diagnosis (Althoff, 1992), Aamodt's comparison of knowledge-intensive CBR methods (Aamodt, 1991), Armengol's and Plaza's analysis of CBR system architectures (Armengol & Plaza,1993), and the INRECA framework for systems comparison and evaluation (Auriol et. al, 1995). The framework we are developing extends these previous suggestions in several respects. It provides an explicit ontology of basic CBR task types, domain characterisations, and types of problem solving and learning methods. Further, it incorporates within this framework a methodology for combining a knowledge-level, top-down analysis with a bottom-up, case-driven one. In this article, we present the underlying view and the basic approach being taken, the main components of the framework and accompanying methodology, and examples of studies recently done and how they relate to the framework.

2. Basic approach

2.1. Knowledge-level analysis

A potentially useful way to describe problem solving and learning behaviour is in terms of the *goals* to be achieved, the *tasks* that need to be solved, the *methods* that will accomplish those tasks, and the *knowledge of the application domain* that those methods need. A description of a system along these lines is often referred to as a knowledge level description, and more recent research in knowledge acquisition (e.g. Steels, 1990; Wielinga, Van de Velde et al., 1993) has clearly demonstrated the usability of this approach. The original knowledge-level idea has undergone some modifications over the years, from Newell's highly intentional, purpose-oriented way of describing a system (Newell, 1982), to a more structured and usable type of description. An incorporation of the CBR perspective into knowledge-level modelling is discussed in Aamodt (1995).

Adopting a knowledge-level perspective to the analysis of integrated PS-ML systems enables the description of methods and systems both from a general (intensional) and case-specific (extensional) perspective. A general description relates a method to descriptive terms and relationships within a general model of descriptive concepts - i.e. an ontology of task types, domain characteristics, and method types. Through a case-driven description, methods can be understood by relating them to already known methods within already described/implemented systems (e.g., CBR is combined with rule- and model-based reasoning in the same way as in CREEK (Aamodt, 1994); a decision tree is generated as in INRECA (Manago, Althoff et al., 1993); the similarity measure is adapted as in PATDEX (Wess, 1993); partial determination rules are generated and used like the so called ãshortcut rulesÒ in MOLTKE (Althoff 1992); etc.). After having developed/described a certain number of systems, we will be able to select/instantiate a system description at the knowledge level by a combined use of general and case-specific descriptors. What we are aiming at is an effective combination of top-down and bottom-up analysis and modelling methods, based on an integration of these two perspectives.

From an engineering point of view, this will enable a particular symbol-level architecture to be chosen, and/or a chosen architecture to be instantiated, based on a thorough understanding of the real world application task and its domain characteristics. However, a knowledge-level description in itself will not provide a language detailed enough to describe or analyse system designs, or to arrive at a symbol-level architecture specification. Our approach therefore incorporates a focusing perspective and an analytic ãtoolÒ to help in the more detailed description that guides the architectural specification based on a knowledge-level model.

2.2. Similarity as a focusing mechanism

The focus provided by this mechanism leads to a view of - in principle - all CBR methods as operations related to *similarity*, in one sense or another. That is, problem solving can be described as a process of initial assessment of similarity (case retrieval) followed by a more deliberate assessment of similarity (case adaptation), and learning (case extraction, case indexing, and possibly updates of general knowledge) can be described by relating it to later similarity assessment - i.e. to a pragmatic learning goal. Along with Richter and Wess (1991) we view similarity as an *a posteriori* criterion, and any attempt to assess similarity before a retrieved case has been found useful will only result in a hypothesised similarity assessment. Our retrieval methods should of course try to minimise the difference between the hypothesised similarity measure and the actual similarity determined after the attempt has been made to use the case. A way to describe the role of general domain knowledge, for example, is then as a means to reduce this uncertainty of the initial similarity assessment with respect to the final similarity assessment made after having evaluated the success of the (possibly modified) case in finding a solution to the input problem.

2.3. Tasks and domain characterisations

The types of application domains we address cover a wide spectrum, ranging from strong-theory domains with rather well-defined relationships between domain concepts (e.g., diagnosis of purely technical systems), to weak-theory and open domains with uncertain domain relationships (e.g., medical diagnosis). This is an important feature of the framework, since we particularly want to relate characteristics of the task (the what-to-do) and the knowledge on which performance of the task is based, to the methods (how-to-dos) that enable the problem solver to accomplish the task by use of the knowledge. The starting point is always the real world setting in which the system is to operate. Medical diagnosis, for example, in a real world setting, is far from a pure classification task (Althoff & Bartsch-Sp□rl, 1996). If a system shall cover the major tasks involved in practical diagnosis, it will have to include planning tasks (e.g., setting up and continuously revising an examination protocol), as well as prediction tasks (assessing the consequences of a treatment).

The next section outlines the core components of the framework, with a focus on the knowledge-level description and analysis and the combined top-down and bottom-up oriented methodology. The incorporation of the similarity assessment mechanism is part of ongoing research.

3. Framework and methodological issues

3.1. Basic framework components

At the highest level of generality, a general CBR cycle may be described by four tasks (Aamodt & Plaza, 1994): *Retrieve* the most similar case or cases, *Reuse* the information and knowledge in that case to solve the problem, *Revise* the proposed solution, and *Retain* the parts of this experience likely to be useful for future problem solving. See Figure 1.

Note that the tasks referred to here are internal reasoning tasks, and different from the application problems tasks (e.g. diagnosis, planning, etc.) referred to earlier. The four CBR tasks each involve a number of more specific sub-tasks. An initial description of a problem (top of Fig. 1) defines a new case. In the Retrieve task this new case is used to find a matching case from the collection of previous cases. The retrieved case is combined with the input case - in the Reuse task - into a solved case, i.e. a proposed solution to the initial problem. The Revise task tests this solution for success, e.g. by applying it to the real-life environment or have it evaluated by a teacher, and repaired if failed. This task is important for learning, since the system needs a feedback of how successful its proposed solution actually was. Retain is the main learning task, where useful experience is retained for future reuse, by updating the case-base and possibly also the general domain knowledge. As indicated in the figure, general knowledge usually plays a part in this cycle, by supporting the CBR processes. This support may range from very weak to very strong, depending on the type of CBR method. By general knowledge we here mean general domain-dependent knowledge, as opposed to the specific domain knowledge embodied by cases. For example, in diagnosing a patient by retrieving and reusing the case of a previous patient, a model of anatomy together with causal relationships between pathological and other physiological states may constitute the general knowledge used by a CBR system. A set of rules may have the same role.

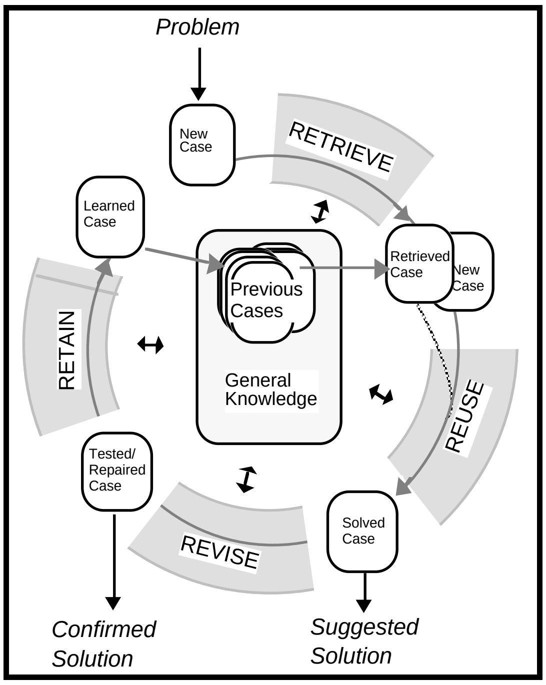

Figure 1 – The CBR Cycle

Knowledge-level analysis, as previously described, is a general approach to systems analysis. It is therefore applicable to the analysis of application tasks and domains - as manifested in the knowledge acquisition methodologies referred to earlier - as well as internal reasoning tasks of a problem solver and learner. In our framework we therefore take a ãtask Ð method Ð domain knowledgeÒ approach both to the analysis of real-world application tasks, and to theanalysis of the CBR reasoning tasks themselves. The mapping between the two is as follows: Methods from the application analysis (e.g. how to solve a technical diagnosis problem, or how to determine the next test to be done) either decomposes an application task into sub-tasks, or it solves the task directly. In both cases these *methods* set up *tasks* at the reasoning level (e.g. problem solving and learning from experience). In the following, we concentrate on the reasoning tasks.

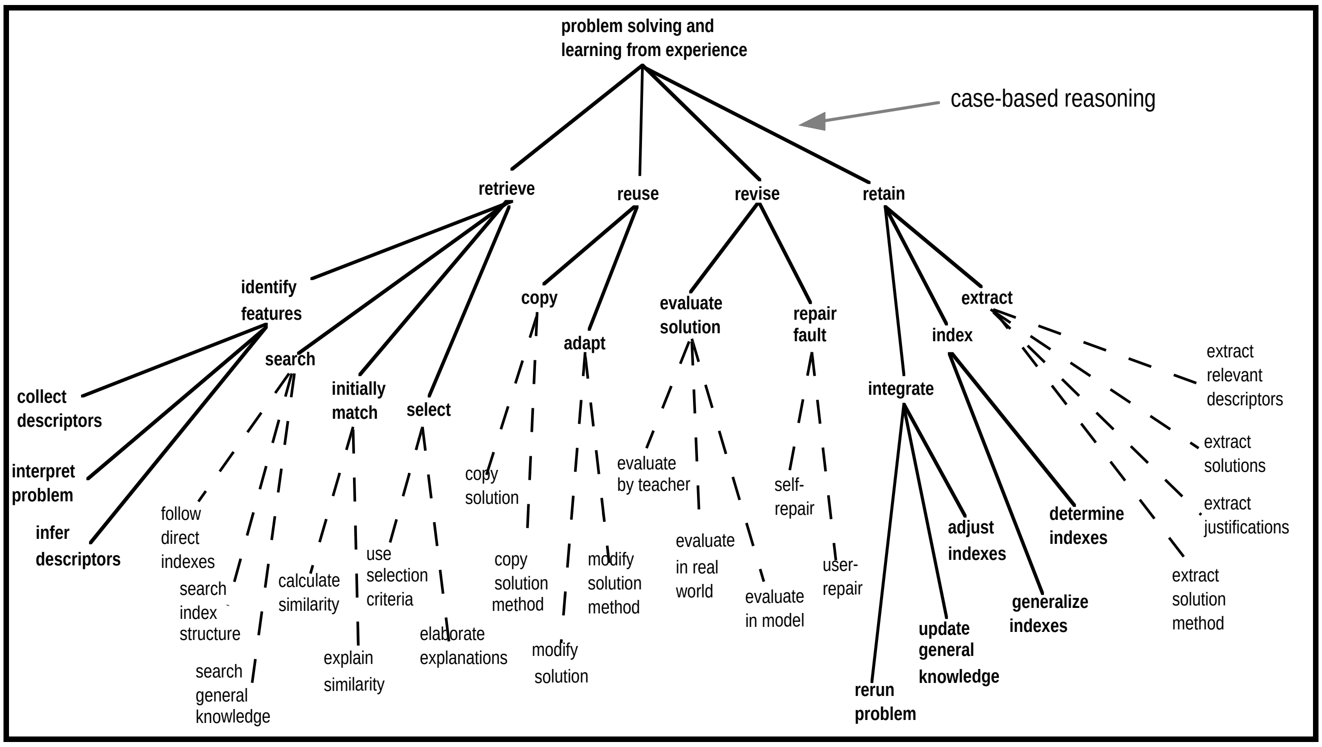

Figure 2 – Task-method decomposition of CBR

The tasks from Figure 1 are further decomposed in Figure 2. The tasks have node names in bold letters, while methods are written in plain text. The links between task nodes (bold lines) are task decompositions, i.e. part-of relations. The links between tasks and methods (stippled lines) identify alternative methods applicable for solving a task. The top-level task is problem solving and learning from experience and the method to accomplish the task is a case-based reasoning method. This splits the top-level task into the four major CBR tasks corresponding to Figure 1. All the four tasks are necessary in order to perform the top-level task. The retrieve task is, in turn, partitioned in the same manner (by a retrieval method) into the tasks identify features, search (to find a set of past cases), initially match (the relevant descriptors to past cases), and select (the most similar case). All task partitions in the figure are considered complete, i.e. the set of sub-tasks of a task are intended to be sufficient to accomplish the task, at this level of description. The figure does not show any control structure over the sub-tasks. The actual control is specified as part of the problem-solving method. The actual retrieval method, for example (not explicitly indicated in the figure), specifies the sequence and loop-backs for the sub-tasks of retrieve. A method specifies the algorithm that identifies and controls the execution of sub-tasks, or solves the task directly, while accessing and utilising the domain knowledge needed to do this. The methods shown in the figure are high level method classes, from which one or more specific methods should be chosen. The method set as shown is incomplete, i.e. one of the methods indicated may be sufficient to solve the task, several methods may be combined, or there may be other methods that have not been mentioned.

The above structure provides the basis for the analytic framework. It needs to be elaborated and described in more detail, characterisations of domain knowledge types need to be added, and dependencies between the various knowledge types need to be identified.

3.2. Methodology

As previously stated, the basic methodological approach is to combine a top-down oriented analysis of application tasks, domain knowledge descriptions, and methods with a bottom-up, case-driven method of studying existing systems. The aim is to arrive at a coherent framework and description language that specialises from the high-level analysis and generalises from the example systems studied.

The baseline of the approach is as follows. We *describe* CBR systems as well as domains and application tasks using two different kinds of criteria, namely criteria characterising the domain and task at hand (domain/task criteria) and criteria describing the abilities and limitations of existing systems and system components (technical/ergonomic criteria) (cf. Althoff & Bartsch-Sp□rl,1996; Althoff, Wess et al., 1995). Examples for domain/task criteria are *size*[1], *theory strength*[2], *openness*[3], *change over time*[4], and *complexity*[5], while *case and knowledge representation*, *similarity assessment*, *validation*, and *data acquisition and maintenance* exemplify important technical/ergonomic criteria. We *analyse* the underlying methods and domain/task characteristics by relating domain/task criteria with technical/ergonomic criteria. Figure 3 gives an example of how CBR systems can be labelled with domain criteria. Combining such a description with a general analysis based on the sub-task structure of Figure 2, was shown to be useful during the final evaluation of the INRECA system (Althoff, Auriol et al., 1995; Althoff, Wess et al., 1995).

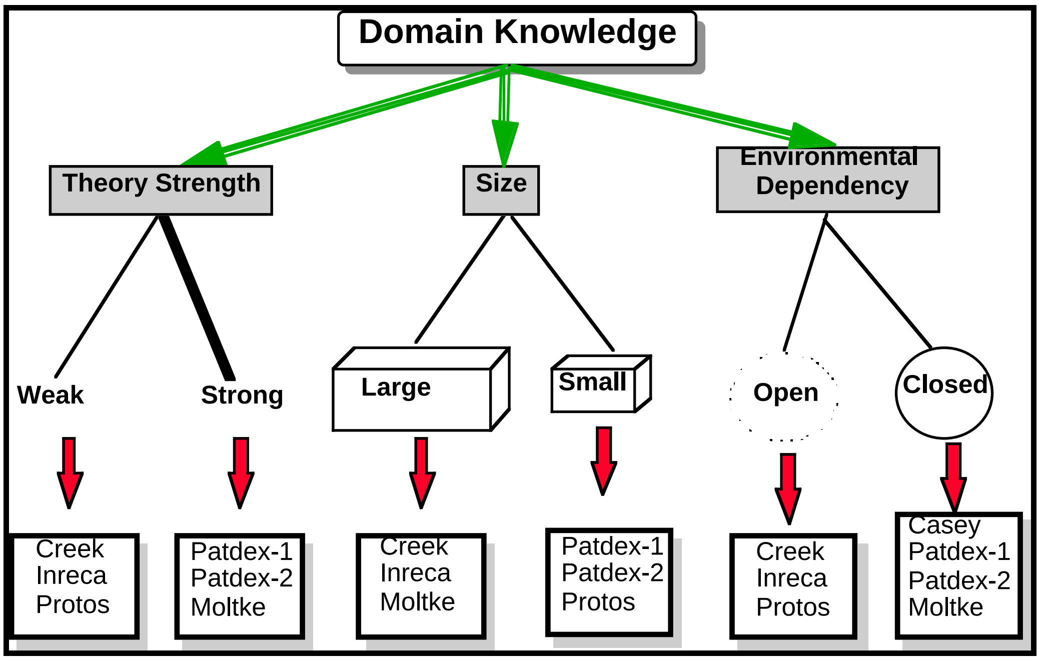

Fig. 3 – An example of domain criteria related to existing systems

From a software or knowledge engineering perspective, we try to arrive at non-functional system properties as a systematic means for (CBR) system development. Since we focus on CBR systems we can define more precise criteria than we could for software systems in general. Additionally, we combine these system-oriented, more technical criteria with an analysis of application domains in

[1] The size of a domain is characterised by the amount of different items representing the explicit knowledge.

[2] The theory strength of a domain depends on the degree of certainty of the involved relationships.

[3] The openness of a domain depends on its environmental dependencies.

[4] The change over time of a domain means that a domain is called static if there are no expected changes and it is called dynamic if it is clear that changes will appear in continuation.

[5] The complexity of a domain means the amount of different taxonomic relations.

which we have experience. On the one hand, feedback from applications can be systematically transformed in an evaluation based on domain and application task criteria. On the other hand, methods extracted from CBR-related systems and tools can be labelled with the results of the application of such criteria. Again, the aim is to close the gap between high level characterisations, on the one side, and concrete systems on the other side. General knowledge level analysis and case-driven analysis are merged in order to come up with application frameworks for particular types of systems, based on a common terminology and a unified model. Here technical/ergonomic criteria are combined with domain/task criteria.

The intended use of this framework both for analysis and development of integrated problem solving and learning systems, can be described as providing answers to the following questions related to the evaluation of AI research (cf. Cohen, 1989):

¥ How is the CBR method to be evaluated an improvement, an alternative, or a complement to other methods? Does it account for more situations, or produce a wider variety of desired behaviour, or is it more efficient in time or space, or does it model human behaviour in a more useful way/detail?

¥ What are the underlying architectural assumptions? Are all design decisions justified? Does the method rely on other integrated methods? Does it subsume other methods?

¥ What is the scope of the method? How extendible is it? Will it scale up?

¥ Why does the method work? Under which conditions would it not work?

¥ What is the relationship between the class of task, of which the current task is an example, and the method used? Can this relationship be stated clearly enough to support a claim that the method is general to the class of task?

4. Example studies

We will point to two small example studies where the framework as specified so far has been used. They both illustrate a bottom-up approach (a case-oriented approach) to the study of CBR methods, undertaken within the general, high level, analytic framework. The first example is a description of the learning method (for the *retain* task), at a suitable level of abstraction, which was derived by abstracting from a set of existing systems. The second example is related to the top-level task of case-based problem solving and learning (see figure 2), and methods for integrating case-based and generalisation-based reasoning.

A set of criteria for analysing and evaluating CBR methods was defined and applied within a study of industrial CBR tools (Althoff, Auriol et al., 1995). These criteria were in turn generalised and extended to CBR research systems (Althoff, Wess et al., 1995). Here the task-method decomposition hierarchy (Fig. 2) was extended with methods that were abstracted from a set of specific CBR algorithms. An abstracted version of these algorithms is shown in figure 4, for the CBR task *retain*.

```
IF no_similar_past_case(current_case)
    THEN construct_new_case;
    ELSE lazy_generalise(old_case);
IF current_case_successful
    THEN integrate_into_successful_cases;
    ELSE integrate_into_total_problem_cases;
DO adaptation UNTIL system_behaves_as_wanted
```

Figure 4 – Abstracted method for the *retain* task. Sources of the algorithm: The CREEK, MOLTKE and PROTOS systems

The THEN and ELSE sentences point to task decomposition methods of the *retain* task, i.e. methods that in turn will lead to a particular selection and control over the *integrate, index, and extract* tasks (figure 2).

The other example starts out from the top level CBR with the aim of studying methods that integrate different reasoning strategies, in this case purely case-based reasoning and reasoning from general domain knowledge. In Table 1 a set of methods, corresponding to some specific research systems, are grouped into levels of integration according to the "tightness" of integration. The different levels of integration are as follows (cf. Auriol, Manago et al., 1995).

	BOLERO	CASEY	CCC+	CREEK	INRECA	MOLTKE	PROTOS	S^3+/INRECA
toolbox level	●	●	●		●	●	●	●
cooperative level	●	●	●	●	●	●		●
workbench level	●			●	●	●		●
seamless level				●	●			●

Table 1. Task: Problem solving and learning from experience, Method: Combining case-specific and general knowledge.

At the *toolbox level* the integration is restricted to the common use of parts of the knowledge-base. A reasoning method is initially chosen, and that method is then used as a single method. Exchange of reasoning results between methods is not done. This is the weakest level of integration. At the *cooperative level* different reasoning methods can exchange intermediate results through a common representation formalism. At the *workbench level* different reasoning methods are aggregated into a single combined method. For instance, a diagnostic system can carry out its classification sub-task using a rule-based module, and the test selection sub-task based on a CBR component. The *seamless level* is the highest level of integration. Different reasoning methods are integrated within a single algorithm and are not viewed as separate methods. Thus, the change of reasoning methods during reasoning is hidden for the user.

While CASEY applies model-based causal reasoning speeded up by CBR (Koton, 1989), BOLERO combines case-based test selection and rule-based diagnostic reasoning (L□pez & Plaza, 1991). In CREEK case-based reasoning is performed within a full-fledged semantic network knowledge model integrating cases as well as general domain knowledge, which enables initial goal-driven context focusing as well as knowledge-intensive, explanation-supported methods for the various CBR tasks (Aamodt, 1994). MOLTKE focuses on rule-based diagnostic reasoning using CBR for exception handling (Althoff, 1992). While INRECA combines CBR with integrated inductive k-d trees for case-filtering, preference learning, and consultation (Althoff, Bergmann et al., 1995; Wess, 1995), S^3+/INRECA integrates INRECA with a service support system basing on general knowledge (Althoff, Wess et al., 1995). CcC+ is a case-based classifier that can cooperate with other sub-systems of the D3 toolbox for diagnostic reasoning (Goos, 1995). However, PROTOS is a stand-alone CBR system for classification, knowledge acquisition, and learning (Bareiss & Porter, 1987).

5. Conclusion and further research

The long term goal of the research reported here is twofold. The first goal is related to AI as an experimental science: To establish aÊdescriptive and analyticframework for improved understanding of the PS-ML integration problem, starting out from a focus on CBR methods. An improved understanding of the ãCBR spaceÒ will be the basis for extensions, suchas into learning methods for more eager generalisation. The resulting framework will serve as a ãlanguageÒand a set of principles for analytic and comparative studies of various systems across different ãtraditionsÒ and research groups. The second goal is related to the needs of improved methods for knowledge-based systems engineering: To specify ML-PS integration as an important task to be understood and handled by the knowledge engineer, and to provide methods for dealing with this problem.

We have reasons to believe that we are on the right track, so far, even if only a high-level and general version of the framework to some extent has been tested. At the core of the framework is the combination of a top-level analysis, based on a task-method decomposition, and a bottom-up analysis of specific systems. It is our intention that the systems descriptions resulting from this analysis can be used by the ongoing efforts of creating an information server for CBR research and development purposes (cf. Althoff & Bartsch-Sp□rl,1995; 1996). An intelligent server should be able to give advice based on a set of *cases* describing CBR systems and applications, embedded in a generic *methodology* for CBR systems decomposition, analysis, and construction. Another use of the framework which we are exploring is its use as a modelling ground for integration of knowledge-based systems into information systems and data base systems in general, based on the different roles these systems (or rather: sub-systems) have in a total decision-support system (Aamodt & Nyg□rd, 1995). One of the major refinements of the frameworkwill be to incorporate relevant parts of the similarity model briefly described in Ch. 2.2.

Acknowledgements

Parts of this research was done within the INRECA project, and benefited from several discussions among project partners (i.e. AcknoSoft, Paris; IMS, Dublin; tecInno, Kaiserslautern; the University of Kaiserslautern). We also thank Brigitte Bartsch-Sp□rl, Enric Plaza, and RalphTraph□nerfor their contributions during several discussions on a number of topics of this article, and Pinar Ozturk for her valuable comments on earlier drafts. Thanks also go to Karl-Heinz Weis jun. for his untiring work on domain and task criteria related to our work.

References

Aamodt, A. (1991). A knowledge-intensive approach to problem solving and sustained learning. Ph.D. Dissertation, University of Trondheim, Norwegian Institute of Technology

Aamodt, A. (1994). Explanation-Driven Case-Based Reasoning. In: S. Wess, K.-D. Althoff & M. M. Richter (eds.), *Topics in Case-Based Reasoning*, Springer Verlag, 274-288

Aamodt, A. (1995). Knowledge Acquisition and learning by experience the role of case-specific knowledge. In: Y. Kodratoff, & G. Tecuci (eds.), *Integration of Knowledge Acquisition and Machine Learning*, Kluwer Academic Publishers, forthcoming, 197-245

Aamodt, A. and Nyg□rd,M. (1995). Different roles and mutual dependencies of data, information, and knowledge - an AI perspective on their integration. *Accepted for publication in Data and Knowledge Engineering*

Aamodt, A. and Plaza, E. (1994). Case-Based Reasoning: Foundational Issues, Methodological Variations and System Approaches. *AI Communications Vol. 7*, *No. 1*, 39-59

Althoff, K.-D. (1992). *Eine fallbasierte Lernkomponente als integrierter Bestandteil der MOLTKE-Werkbank zur Diagnose technischer Systeme.* Doctoral Dissertation, University of Kaiserslautern; also: Sankt Augustin: DISKI 23, infix Verlag

Althoff, K.-D., Auriol, E., Barletta, R. & Manago, M. (1995). *A Review of Industrial Case-Based Reasoning Tools*. AI Intelligence, Oxford, UK

Althoff, K.-D. & Bartsch-Sp□rl, B. (1995). Call forParticipation: Decision Support for Case-Based Applications. *Initiative for systematically collecting information about existing CBR systems* *and applications, BSR Consulting (Munich) and Centre for Learning Systems and Applications (LSA; University of Kaiserslautern)*

Althoff, K.-D. & Bartsch-Sp□rl, B. (1996). Decision Support for Case-Based Applications. *Wirtschaftsinformatik 1/96, special issue on case-based decision support (edited by D. Ehrenberg)*

Althoff, K.-D., Bergmann, R., Globig, C., Wess, S., Auriol, E. & Manago, M. (1995). The Seamless Integration of Induction and CBR in INRECA. *Unpublished manuscript,* University of Kaiserslautern

Althoff, K.-D., Wess, S., Weis, K.-H. jun., Auriol, E., Bergmann, R., Holz, H., Johnston, R., Manago, M., Meissonnier, A., Priebisch, C., Traph□ner, R. & Wilke, W. (1995). *An Evaluation of the Final Integrated System.* INRECA (Esprit project 6322), Deliverable D6

Armengol, E. & Plaza, E. (1994). A Knowledge Level Model of Case-Based Reasoning. In: S. Wess, K.-D. Althoff & M. M. Richter (eds.), *Topics in Case-Based Reasoning*, Springer Verlag, 53-64

Auriol, E., Manago, M., Althoff, K.-D., Wess, S. & Dittrich, S. (1995). Integrating Induction and Case-Based Reasoning: Methodological Approach and First Evaluations. In: J.-P. Haton, M. Keane & M. Manago (eds.), *Advances in Case-Based Reasoning*. Springer Verlag, 18-32

Bareiss, R. & Porter, B. (1987). PROTOS - an exemplar-based learning apprentice. *Proc. 4th International Workshop on ML*, Irvine, 12-23

Cohen, P. R. (1989). Evaluation and Case-Based Reasoning. In: K. Hammond (ed.), *Proc. of the 2nd Darpa Workshop on Case-Based Reasoning*, Morgan Kaufmann, 168-172

David, J.-M., Krivine, J.-P. & Simmons, R. (eds.) (1993). *Second Generation Expert Systems*. Springer Verlag

Goos, K. (1995). *Fallbasiertes Klassifizieren Ð Methoden, Integration und Evaluation*. Doctoral Dissertation, University of W□rzburg

Koton, P. (1989). *Using experience in learning and problem solving.* MIT, Laboratory of Computer Science Ph.D. Dissertation, October 1988), MIT/LCS/TR-441

Newell, A. (1982). The knowledge level. *Artificial Intelligence*, Vol. 18, pp 87-127

L□pez, B. and Plaza, E. (1991). Case-based Learning of StrategicKnowledge. In: Y. Kodratoff (ed.), *Proc. EWSL-91*, Springer Verlag, 398-411

Manago, M., Althoff, K.-D., Auriol, E., Traph□ner, R., Wess, S., Conruyt, N. &Maurer, F. (1993). Induction and Reasoning from Cases. In: M. M. Richter, S. Wess, K.-D. Althoff & F. Maurer (eds.), *Proc. First European Workshop on Case-Based Reasoning (EWCBR-93*), Seki-Report, University of Kaiserslautern

Richter, M. M. & Wess, S. (1991). Similarity, uncertainty and case-based reasoning in Patdex. In: R. S. Boyer (ed.), Automated Reasoning - Essays in Honor of Woody Bledsoe, Kluwer Academic Publishers, 249-266

Steels, L. (1990). Components of Expertise. *AI Magazine*, 11(2), Summer 1990, 29-49

Veloso, M. & Aamodt, A. (eds.) (1995). *Case-Based Reasoning Research and Development*. Springer Verlag

Wess, S. (1993). PATDEX - ein Ansatz zur wissensbasierten und inkrementellen Verbesserung von □hnlichkeitsbewertungen in der fallbasierten Diagnostik. In: F. Puppe and A. G□nter (eds.) (1993). *Expertensysteme 93. Proc. of the 2nd German Conference on Expert Systems.* Springer Verlag, 42-55

Wess, S. (1995). *Fallbasiertes Schlie§en in wissensbasierten Systemen zur Entscheidungsunterst□t-zung und Diagnostik*. Doctoral Dissertation, University of Kaiserslautern

Wielinga, B. J., Van de Velde, W., Schreiber, G. & Akkermans, H. (1993). Towards a unification of knowledge modeling approaches. In: *David, Krivine & Simmons (1993)*, 299-335