# Pythia: Compositional meaning construction for ontology-based question answering on the Semantic Web

Christina Unger and Philipp Cimiano

CITEC, Bielefeld University, Germany

**Abstract.** In this paper we present the ontology-based question answering system Pythia. It compositionally constructs meaning representations using a vocabulary aligned to the vocabulary of a given ontology. In doing so it relies on a deep linguistic analysis, which allows to construct formal queries even for complex natural language questions (e.g. involving quantification and superlatives).

**Keywords:** ontology-based question answering, compositionality

## 1 Introduction

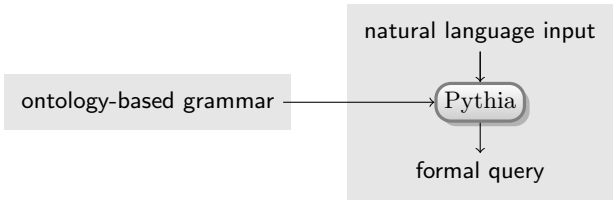
The growing Semantic Web provides a large amount of ontology-based semantic markup that question answering systems can exploit in order to interpret and answer natural language questions. This means that user questions can be interpreted with respect to a particular ontology which provides natural language expressions with a well-defined meaning, thereby allowing to retrieve precise answers.

In this paper we present the ontology-based question answering system Pythia. It is based on the following two main ideas. First, it uses principled linguistic representations in order to compositionally construct general meaning representations that can subsequently be translated into formal queries. Such a deep linguistic analysis allows Pythia to construct formal queries even for complex natural language questions, e.g. involving quantification and superlatives. And second, it relies on a specification of the lexicon-ontology interface that explicates possible linguistic realizations of ontology concepts. This allows to build meaning representations that use a vocabulary aligned to the vocabulary of a given ontology, thereby ensuring a precise and correct mapping of natural language terms to corresponding ontology concepts.

In the following sections we present the system, its architecture, and report on evaluation results with respect to a subset of DBPedia and compare our system with related work.

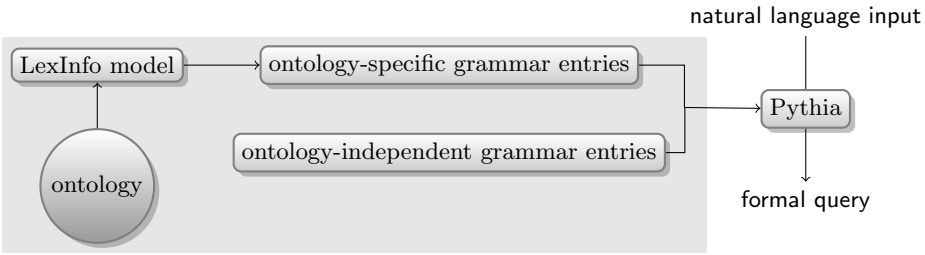
## 2 Approach

The architecture of our question answering system Pythia can be depicted very roughly as follows:




Natural language input is tranformed into a formal query by means of a linguistic analysis that is driven by an ontology-based grammar. Before explicating this transformation, we will briefly describe the grammar and its generation. A more detailed account of grammar generation and of the motivation to use ontology-specific grammars is given in [4].

In Pythia, natural language expressions are parsed and interpreted with respect to a grammar which we assume to be composed of two parts: an ontology-specific part and an ontology-independent part. The ontology-specific part contains lexical entries that refer to individuals, concepts, and properties of the underlying ontology. It is generated automatically from an ontology-lexicon model, as will be described below. The ontology-independent part comprises functional expressions like auxiliary verbs, determiners, wh-words and so on. The overall picture can be sketched as follows:




Both parts of the grammar use principled linguistic representations. More specifically, we assume grammar entries to be pairs of a syntactic and a semantic representation. As syntactic representation we take trees from Lexicalized Tree Adjoining Grammar (LTAG [5]). LTAG is very well-suited for ontology-based grammar generation because it allows for flexible basic units; we can, for example, assume complex grammar entries for examples like population of or has. . . inhabitants. As semantic representations we take DUDEs [2], a kind of Underspecified Discourse Representation Structures (UDRS [6]) augmented with information that allows for a flexible semantic composition.

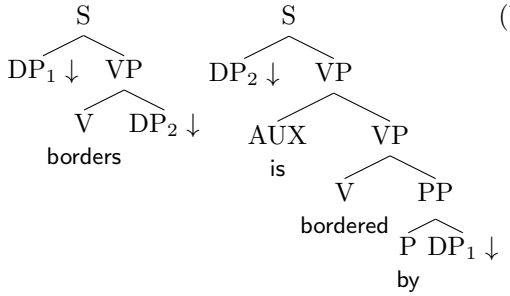
The first step in generating a grammar from a given ontology is to enrich the ontology with information about its verbalization. The framework we use for this is LexInfo[1] [3], which offers a general frame for creating a declarative specification of the lexicon-ontology interface by connecting concepts of the ontology to information about their linguistic realization, i.e. word forms, morphology, subcategoriziation frames and how syntactic and semantic arguments correspond to

[1] `http://lexinfo.net`

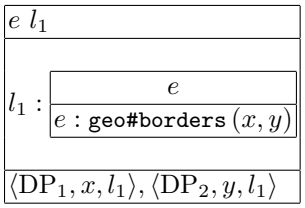
each other. The lexical entries specified by LexInfo are then input to a general mechanism for generating grammar entries, i.e. pairs of syntactic and semantic representations.

For example, the object property `borders` in the ontology is first specified to be verbalized as the transitive verb to border together with the relevant linguistic information (inflection, subcategorization, and so on). The resulting lexical entry is then input to a grammar generation mechanism, which specifies general templates for mapping LexInfo entries to grammar entries. Applied to the entry for to border it gives rise to a family of elementary LTAG trees, two of them – one for active and one for passive use – are given in 1a. They are both paired with the UDRT-like semantic representation in 1b.

1. (a)

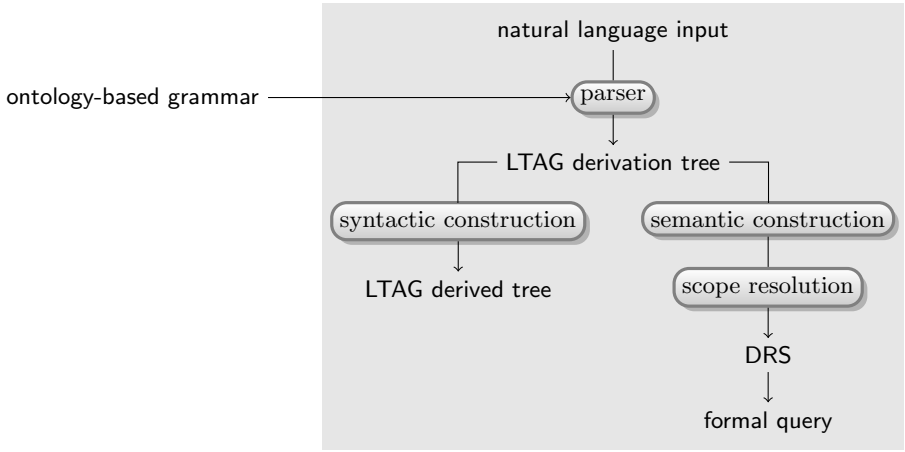



(b)




The syntactic structure encoded in the elementary trees captures the lexical material that is needed for verbalizing the property `borders`. The semantic representation contains a DRS labelded $l_1$, which provides the predicate `geo#borders` corresponding to the intended concept in the ontology (the prefix `geo#` abbreviates the namespace of the ontology), as well as information about the semantic arguments ($x$ and $y$) and about which substitution node in the syntactic structure will provide them.

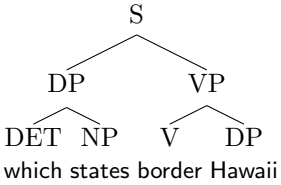
These linguistic representations are then used for parsing and interpreting natural language questions. The process of mapping natural language input to formal queries can be depicted as follows:



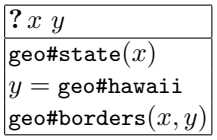
It involves three main steps. First, the input is handed to a parser, which works along the lines of the Earley-type parser devised by Schabes & Joshi [7]. It constructs an LTAG derivation tree, considering only the syntactic part of the grammar entries involved. Next, syntactic and semantic composition rules apply in tandem in order to construct a derived tree together with an according DUDE. The syntactic composition rules are LTAG's standard substitution and adjoin operations, and the semantic composition rules are parallel operations on DUDEs: an argument saturating operation (much like function application) that interprets substitution, and a union operation that interprets adjoin. Once all argument slots are filled, the constructed DUDE corresponds to an equivalent UDRS, which is then subject to scope resolution, resulting in a set of disambiguated Discourse Representation Structures (DRS [8]). Those are subsequently translated into a formal query. In the presented version of the system we use queries formulated in FLogic [9], but any other query language, e.g. SPARQL, could be used as well.

As an example, consider the input question Which states border Hawaii?. The parser produces a derivation tree that yields the derived tree in 2a. Parallel to this, a UDRT-like semantic representation is built which resolves to the DRS in 2b (the question mark serves to point out those variables whose values should be provided as an answer).

2. (a)




(b)




Subsequently, the DRS is translated into the following FLogic query:

```
FORALL X,Y <- X:geo#state AND equal(Y,geo#hawaii) AND
              X[geo#border -> Y]. orderedby X
```

It reads similarly to a first-order formula: for all bindings of `X` and `Y` such that `X` is a state and `Y` equals Hawaii and `X` borders `Y`, return all `X`. The query can then be evaluated with respect to the ontology, e.g. by means of the OntoBroker Engine [10]. Since there are no states bordering Hawaii, the returned result is empty.

## 3 Evaluation

### 3.1 Dataset and grammar generation

There is no established evaluation standard for question answering systems, but an often-used gold standard is Raymond Mooney's ontology comprising geographical information about the U.S. together with a set of 880 annotated user

questions[2]. In order to use these questions for evaluation, we extracted from DBpedia a subset containing all U.S. states, cities, mountains, lakes, rivers and roads. Furthermore, we annotated 865 of the 880 questions with corresponding FLogic query results. (The remaining 15 questions are questions which are out of scope of the ontology, such as Which rivers do not run through USA?).

In order to customize our question answering system to the domain of U.S. geography, we first constructed a LexInfo model for the extracted subset of DBpedia, which specifies how the concepts and relations of this ontology are verbalized. The constructed LexInfo model contains 678 lexical entries, of which 600 correspond to common nouns representing individuals and could be constructed automatically. The remaining 78 entries were built manually, using LexInfo's API. The effort to do so amounted to less than two minutes per entry, leading to a total amount of approximately two and a half hours. Next, those LexInfo entries were input to automatic grammar generation, yielding 2785 grammar entries (pairs of syntactic and semantic representations). Additionally, we manually specified 149 grammar entries for domain-independent elements such as determiners, wh-words, auxiliary words, and so on. The complete set of grammar entries are then used by Pythia for processing user questions. All mentioned resources – the dataset, the questions annotated with FLogic queries, the lexicon model, and the grammar files – are available at `http://www.sc.cit-ec.uni-bielefeld.de/pythia`.

### 3.2 Evaluation results and discussion

Running Pythia on the above mentioned 865 user questions and comparing the results of the constructed queries with the results given by the gold standard queries, we reach a recall of 67 % and precision of 82 %, leading to an F-measure of 73,7%.

Cases in which the system fails to construct an appropriate query can be pinned down to reasons that can roughly be categorized as Pythia-internal and Pythia-external failures. Pythia-external failures mean failures for which Pythia is not to blame. On the one hand side, these comprise questions that are ill-formed:

- *syntactically ill-formed questions:*
  questions that are incomplete or ungrammatical and therefore are not parsed (e.g. What is capital of Iowa?, What are the capital city in Texas?)
- *semantically ill-formed questions:*
  questions that violate sortal restrictions (e.g. Which states border the Missouri river?)

On the other hand side, they comprise questions that fail due to data incompleteness. For example, the ontology concept `highest_point` should be extensionally equivalent to locations with maximal height, which however is not always the case. Thus, if the gold standard uses the concept `highest_point`, while Pythia

[2] Available at `ftp://ftp.cs.utexas.edu/pub/mooney/nl-ilp-data/geosystem/`.

constructs a query asking for the location with maximal height, the results of both sometimes do not match, although the meaning of both queries are intuitively equivalent.

By Pythia-internal failures we mean questions that could in principle be parsed and answered, but for one reason or the other the system fails to do so. There are mainly two reasons for such failure. One reason is incomplete coverage, i.e. there is lexical material (e.g. the verb washed by) or syntactic constructions (e.g. topicalizations and NP disjunctions) missing in our grammar. Examples of questions that cannot be parsed for these reasons are Of the states washed by the Mississippi river, which has the lowest point? and How many states have cities or towns named Springfield?. The other reason for failure is non-compositionality, i.e. cases where the logical form of the whole question is not exactly the composition of the meaning of the parts of the question. This involves components that do not contribute anything to the overall meaning (e.g. american, in the US, give me) and some specific cases of counting with respect to a restriction (e.g. in Which river flows through the most states?, where the number of states has to be counted for each river). However, we skip a deeper discussion of these cases.

The table in Fig. 1 gives an overview of the qualitative coverage of our approach. The categories are taken from Cimiano & Minock [16]). Full treatment is expressed by +, missing treatment by −, (+) denotes partial or ad hoc coverage, and (−) means something is not captured yet but could in principle be incorporated.

| | | |
|---|---|---|
| Question types | wh-questions | + |
| | how ADJ/many | + |
| | requests | (+) |
| | topicalized questions | (−) |
| | nominals | + |
| Ambiguities | lexical, syntactic, scope | + |
| Other phenomena | spatial propositions | + |
| | adjectival modifiers and superlatives | + |
| | aggregation and comparisions | + |
| | negation | + |
| | coordination | (−) |
| | non-compositionality | (+) |
| | variability | (+) |
| | handling out-of-scope questions | − |
| | temporal aspects | − |

**Fig. 1.** Covered question types and phenomena

Finally, let us briefly compare Pythia's results (67% precision and 82% recall) with other systems that were evaluated on the Geobase dataset. C-Phrase[3] by

[3] http://code.google.com/p/c-phrase/

Minock [11] reaches 80-90% precision and recall after 120 minutes of authoring (cf. [12]). Mooney's learned semantic parsers (cf. Mooney [13]), on the other hand, reach a precision between 70 and nearly 100%, and a recall between 60 and 80%. They require no manual effort but rely on a large enough training corpus (queries annotated with semantic representations). PRECISE [14], on the other hand, requires no customization as the needed lexicon is extracted automatically from the input database. Only semantically tractable questions are answered, thereby reaching 100% precision, while for semantically intractable questions the system requests a paraphrase. Of the 880 questions, about 80% of the questions turn out to be tractable.

So, in general Pythia's results are competetive, albeit slightly excelled by other systems. This is because Pythia comes with a trade-off between coverage and the manual effort that is required with respect to the LexInfo model. A majority of the yet uncovered cases could in principle be covered by manually extending the LexInfo model and by including new domain-independent constructions like coordination. The real benefit of Pythia, however, is that it is able to handle linguistically complex queries involving quantification, superlatives and comparisons, aggregation functions, and the like – that is, phenomena that most other (and especially shallow) systems cannot cope with and would be difficult to extend with.

## 4 Conclusion and future work

We presented an ontology-based question answering system that parses user questions with respect to a domain-specific lexicon built automatically from a specification of linguistic realizations of ontology concepts. It compositionally constructs meaning representations that are aligned to the vocabulary of the underlying ontology and can easily be translated into a formal query.

This approach has several advantages. Due to the use of principled linguistic representations, Pythia is able to handle a wide range of linguistically complex queries, involving quantifiers, numerals, comparisons and superlatives, negation, and so on. Furthermore, due to the explicit specification of the lexicon-ontology interface, it is able to correctly map natural language terms to corresponding ontology concepts, even if they are superficially different (e.g. mapping has...inhabitants to the property *population*).

The major challenge for such an approach concerns portability. Adapting Pythia to a new domain requires the creation of a new LexInfo model for that domain, from which domain-specific grammar entries can be generated. This means that in a linguistically rich approach like ours, ontological support comes with a price: scalability. Pythia works very well for a relatively small domain, but requires non-negligible effort for larger domains. Part of our research therefore focuses on replacing the manual construction of LexInfo models by a largely automatic mapping of natural language expression to entities and relations. This will become especially important when applying the approach to larger domains on the Semantic Web, e.g. the whole of DBpedia.

## References


1. Bunt, H.: Semantic Underspecification: Which Technique For What Purpose? In: Computing Meaning, vol. 83, pp. 55–85. Springer Netherlands (2007)
2. Cimiano, P.: Flexible semantic composition with DUDES. In: Proceedings of the 8th International Conference on Computational Semantics (IWCS). Tilburg (2009)
3. Cimiano, P., Buitelaar, P., McCrae, J., Sintek, M.: Lexinfo: A declarative model for the lexicon-ontology interface. Journal of Web Semantics: Science, Services and Agents on the World Wide Web 9(1), pp. 29–51.
4. Unger, C., Hieber, F., Cimiano, P.: Generating LTAG grammars from a lexicon-ontology interface. In: S. Bangalore, R. Frank, and M. Romero (eds.): 10th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+10), Yale University (2010)
5. Schabes, Y.: Mathematical and Computational Aspects of Lexicalized Grammars. Ph. D. thesis, University of Pennsylvania (1990)
6. Reyle, U.: Dealing with ambiguities by underspecification: Construction, representation and deduction. Journal of Semantics 10, 123–179 (1993)
7. Schabes, Y., Joshi, A.K.: An Earley-type parsing algorithm for Tree Adjoining Grammars. In: Proceedings of the 26th annual meeting of ACL, Buffalo, New York, pp. 258–269 (1988)
8. Kamp, H., Reyle, U.: From Discourse to Logic. Kluwer, Dordrecht (1993)
9. Kifer, M., Lausen, G.: F-logic: A higher-order language for reasoning about objects, inheritance, and scheme. Technical report, SIGMOD Record 18(2) (1989)
10. Decker, S., Erdmann, M., Fensel, D., Studer, R.: Ontobroker: Ontology based access to distributed and semi-structured information. In: Database Semantics: Semantic Issues in Multimedia Systems, pp. 351–369. Kluwer (1999)
11. Minock, M.: C-Phrase: A system for building robust natural language interfaces to databases. Data Knowl. Eng. 69(3), 290–302 (2010)
12. Minock, M., Olofsson, P., Näslund, A.: Towards building robust natural language interfaces to databases. In: Proceedings of the International Conference on Applications of Natural Language to Information Systems (NLDB), pp. 187-198 (2008)
13. Mooney, R.: Learning for semantic parsing. In: Gelbukh, A. (ed.) Computational Linguistics and Intelligent Text Processing: Proceedings of the 8th International Conference, CICLing 2007, Mexico City, pp. 311–324. Springer (2007)
14. Popescu, A.-M., Etzioni, O., Kautz, H.: Towards a theory of natural language interfaces to databases. In: IUI'03: Proceedings of the 8th international conference on Intelligent user interfaces, New York, USA, pp. 149–157. ACM (2003)
15. Schiehlen, M.: Semantic Construction from Parse Forests. In: Proceedings of the 16th International Conference on Computational Linguistics, Copenhagen (1996)
16. Cimiano, P., Minock, M.: Natural Language Interfaces: What's the Problem? – A Data-driven Quantitative Analysis. In: Proceedings of the International Conference on Applications of Natural Language to Information Systems (NLDB), pp. 192–206 (2009)