SWSNL: Semantic Web Search Using Natural Language

Ivan Habernal[a,*], Miloslav Konopík[b]

[a] *Department of Computer Science and Engineering, Faculty of Applied Sciences, University of West Bohemia, Univerzitní 8, 306 14 Plzeň, Czech Republic*

[b] *NTIS – New Technologies for the Information Society, Faculty of Applied Sciences, University of West Bohemia in Pilsen, Univerzitní 8, 306 14 Plzeň, Czech Republic*

Abstract

As modern search engines are approaching the ability to deal with queries expressed in natural language, full support of natural language interfaces seems to be the next step in the development of future systems. The vision is that of users being able to tell a computer what they would like to find, using any number of sentences and as many details as requested. In this article we describe our effort to move towards this future using currently available technology. The Semantic Web framework was chosen as the best means to achieve this goal. We present our approach to building a complete Semantic Web Search using Natural Language (SWSNL) system. We cover the complete process which includes preprocessing, semantic analysis, semantic interpretation, and executing a SPARQL query to retrieve the results. We perform an end-to-end evaluation on a domain dealing with accommodation options. The domain data come from an existing accommodation portal and we use a corpus of queries obtained by a Facebook campaign. In our paper we work with written texts in the Czech language. In addition to that, the Natural Language Understanding (NLU) module is evaluated on another domain (public transportation) and language (English). We expect that our findings will be valuable for the research community as they are strongly related to issues found in real-world scenarios. We struggled with inconsistencies in the actual Web data, with the performance of the Semantic Web engines on a decently sized knowledge base, and others.

Keywords:
semantic search, natural language understanding, semantic web, statistical semantic analysis

1. Introduction

Keyword-based search engines have proven to be very efficient on collections of unstructured textual content, e.g. web pages [1]. However, if users want to find information in a structured content, e.g. in a database, the basic keyword search might not be expressive enough [2]. A simple, yet sufficient solution on the Web can be provided by a form-based user interface which typically combines keywords with some other restrictions, according to the specific domain structure [3]. Form-based interfaces are user friendly in the sense that they do not require the user's prior knowledge of the underlying data structures. The structure is typically shown as multiple forms or menus that allow further specification of the user request. Nevertheless, it is obvious that from the user's point of view, a simple keyword input is a more straightforward approach than filling forms.

A different approach to retrieving sought information consists in using a domain-specific query language. However, as pointed out by [4], querying structured data using a domain-specific query language, e.g. SQL or SPARQL [5], can be complicated for casual users. Most of the existing query languages use a precise syntax and the users are expected to be familiar with the back-end data structures. Evidently, this allows exact queries to be formulated by domain experts but on the other hand this is a big obstacle for casual users.

One step beyond the above-mentioned traditional approaches are Natural Language Interfaces (NLI). The key feature of such interface is that users can search for the required information by posing their queries using natural language (NL). It is assumed that posing NL queries is more precise than the keyword approach and at the same time more natural than the form-based approach. Recent studies [6, 7] indeed confirmed that NLI offers a better user experience than the form-based search for casual users.

In the article we present a realistic approach to designing and developing a complete Semantic Web Search using Natural Language (SWSNL) system. The system builds on the Semantic Web paradigm and thus uses ontologies as a main vehicle for storing the domain structure and all the data (the Knowledge Base, KB). Furthermore, ontologies, due to their ability to precisely capture the semantics, are also used to describe the meaning of user queries that

*Corresponding author. Tel: +420 377632456
Email addresses: habernal@kiv.zcu.cz (Ivan Habernal), konopik@kiv.zcu.cz (Miloslav Konopík)

are expressed in natural language (NL). The system can be seen as a search engine which accepts NL queries and performs the search in the back-end KB.

Our SWSNL system also has a few aspects that distinguish it from other related work. Firstly, we allow users to formulate their NL queries using more than a single sentence. Secondly, the Natural Language Understanding (NLU) module incorporates a stochastic semantic analysis model and does not require any syntactic parser preprocessing. The NLU module is trained from annotated data and it is language independent. Thirdly, the ontology for describing natural language queries is different from (independent of) the KB ontology. This independence enables switching between various KBs without affecting the NLU module.

The system was thoroughly tested and evaluated on three different domains. Primarily, the end-to-end evaluation was performed on the accommodation options domain using actual data acquired from the Web as well as the corpus of actual NL queries in the Czech language. The other two domains, the Czech public transportation domain and a subset of the English ATIS dataset, were used to verify the portability to different domains and languages.

The initial impulse to build the presented system was an idea to enrich an existing form-based application with a newly developed NLI. The expected benefits were targeted into better user experience and into testing the NLI approach in practice. The selected domain of accommodation options followed the idea as our preliminary survey which revealed a promising potential of such a domain. The selection of the Czech language as the domain language was not a random choice. The decision was meant to verify the technology in a different language than English.

The rest of the article is organized as follows. Section 2 outlines the context of the related work. In section 3 we introduce our system in a schematic overview. Section 4 describes the target domain together with the techniques required to deal with actual Web data sources. A natural language corpus is also presented. Section 5 describes the formalism for capturing natural language query semantics. Section 6 proposes a statistical semantic model for analysing natural language queries and section 7 deals with the semantic interpretation of a semantic annotation in order to perform a search in a particular KB. Section 8 thoroughly describes the evaluation of our SWSNL system. Open issues and future work are then discussed in section 9.

2. Related Work

Before we present the related work, we shortly discuss the terminology of the current research into Natural Language Interfaces. Typically, the term *Natural Language Interfaces* (NLI) is used when a system can be accessed using (written) natural language. Such a system mostly operates on structured information and, given the natural language question, it tries to find the correct answer. The family of NLI systems can be divided into various sub-classes. A *Natural Language Interfaces to Databases* (NLIDB) system holds information in a relational database. The principles of NLIDB have been adapted to the Semantic Web resulting into the *Natural Language Interfaces to Knowledge Bases* (NLIKB). In this case of NLI, the information is stored in the form of ontology, which plays the fundamental role in the Semantic Web.

We define our research as *Semantic Web Search Using Natural Language* (SWSNL). Although NLIKB covers a task similar to our approach, most of the existing NLIKB systems are designed to evaluate rather complex logical queries over the Knowledge Base. On the contrary, SWSNL can be viewed as a special case of a search engine that retrieves a set of results according to a natural language query, operating in the Semantic Web field. Furthermore, our SWSNL system does not belong to the family of question answering systems since these two fields have a very different motivation. Whereas a question answering system tries to answer a user's NL questions, the purpose of our system is to retrieve search results according to a user's NL queries.

2.1. Overview of recent NLIKB systems

A very recent system called *PowerAqua* [8] is an ontology-based NLI system which surpasses traditional systems by managing multiple ontology sources and high scalability. Since its NL processing module remains the same as in the previous *AquaLog* system [9], we will review the *AquaLog* system. *AquaLog* is a portable NLIKB system which handles user queries in a natural language (English) and returns answers inferred from a knowledge base. The system uses GATE[1] libraries (namely the tokenizer, the sentence splitter, the POS tagger, and the VP chunker).

ORAKEL [10] is an ontology-based NLI system. It accepts English factoid questions and translates them into first-order logic forms. This conversion uses full syntax parsing and a compositional semantics approach. *ORAKEL* can be ported into another domain but such porting requires a domain expert to create a domain-dependent lexicon. The lexicon is used for an exact mapping from natural language constructs to ontology entities. A possible drawback of *ORAKEL*'s approach is that the system can neither handle ungrammatical questions nor deal with unknown words.

The *FREyA* system [11] is an NLIKB system that combines syntactic parsing with ontology reasoning. It derives parse trees of English input questions and uses heuristic rules to find a set of potential ontology concepts (for mapping from question terms to ontology concepts) using GATE and the OntoRoot Gazetteer [12]. The primary source for question understanding is the ontology itself. If the system encounters ambiguities, a clarification dialogue

[1] `http://gate.ac.uk`

is offered to the user. The potential ontology concepts retrieved from the question analysis are then transformed into SPARQL. The system was tested on Mooney: Geography dataset [13].

A portable NLIKB system called *PANTO* [14] is based upon the off-the-shelf statistical parser Standford Parser and integrates tools like WordNet and various metrics algorithms to map the NL question terms to an intermediate representation called *QueryTriples*.[2] This semantic description is then mapped onto the *OntoTriples* that are connected to entities from the underlying ontology. This step involves a set of 11 heuristic mapping rules. Finally, OntoTriples are represented as SPARQL queries. The main idea of transforming a NL question into triples in PANTO is based upon the empirical observation that two nominal phrases from a parse tree are expected to be mapped onto a triple in the ontology. The system was evaluated on the Mooney dataset and the output of PANTO was compared to the manually generated SPARQL queries.

The *NLP-Reduce* system [15] does not involve any advanced linguistic and semantic tools and depends on matching the query words to the KB instances. Its core part is a *query generator* which is responsible for creating SPARQL query given the words and the lexicon extracted from the KB. The major strength of the system is its good portability as it does not depend on any complex NLP query processing.

The ontology-based NLI system called *QACID* introduced in [16], covers a cinema/movie domain. Its target language is Spanish. It consists of two main components, the *user query formulation database*, and *textual-entailment engine*. Whereas the former component serves mainly for development and system training purposes, the latter is intended for an unknown query processing. The core of the QACID system is a database of query formulations. The database contains a set of 54 clusters, each cluster represents one type of question and it has a representative query pattern which was derived from a set of training data. Each cluster is also associated with one SPARQL query. As pointed out in the QACID evaluation, the system is not able to answer unknown ontology concepts and therefore it fails if the user poses a query using terms that are not present in the lexicon.

A domain-restricted NLI system called *QUETAL* [17] exploits a robust semantic analysis on a hybrid NLP architecture. The system was developed to answer NL questions on two domains: the Nobel prizes and information about the Language Technology Institute. For these two domains, the ontologies were converted from existing ontologies and merged with other more general ontologies (e.g. SUMO for the Nobel prizes). Question analysis is performed by the HPSG (Head-driven Phrase Structure Grammar [18]) syntactic and semantic parser with the support of the Heart of Gold architecture[3] which provides shallow Named Entity Recognition (NER). The HPSG parser uses grammars for English and German and the output of the parser is a formalism called Minimal Recursion Semantics which is a flat, non-recursive semantic representation format [19]. This question representation is then transformed into the *proto queries* that are suitable for querying a particular KB.

A domain-independ system *QuestIO* [20] extracts the domain lexicalizatin from the KB. The natural language query is transformed into SeRQL and then executed against the KB. The system was tested on 36 questions collected from the GATE user mailing list and several queries over a geographic ontology.

Instead of supporting full NL questions, the *QUICK* (QUery Intent Constructor for Keywords) system guides users in constructing a semantic query from keywords [21]. After typing in the keywords, the system analyses them according to the underlying ontology and allows the user to choose the required semantic query in a clarification dialogue. The performance was tested using two ontologies (a movie database and a song database) and 175 queries consisting of 2-5 keywords. Although QUICK is not a fully-fledged NLI system, it can be viewed as a trade-off between a simple keyword search and the NL question systems. Another keyword-based information extraction and retrieval system based on ontology in the soccer domain is presented in [22]. The system is evaluated only on 10 simple queries. A good overview of other state-of-the art keyword-based semantic search systems is shown in [23].

2.2. Existing Ontologies In Accommodation Domain

The *Travel ontology* [24] is a core part of *Travel Guides*[4]. The goal of this system is to solve the problem of distributed tourist sources and to help users find a vacation package quickly. The ontology thoroughly models the tourist domain but it is not populated with instances of actual accommodation options.

Another travel ontology developed by Knublauch in 2004 [25] was intended as an example how to put Semantic Web techniques into practice. The ontology covers the travel domain, however, it was not designed according to any existing travel information source.

Accessing touristic ontology using NLI is the goal of [26]. The system is evaluated on a set of 20 simple NL queries. The author of [27] deals with a related task, namely with a personalised hotel search using Semantic Web technologies. However, his system is form-based and does not involve NLI.

It is also worth mentioning that ontologies are nowadays used by some companies as a backbone in tourism

[2] A mixed terminology (*triplets* vs. *triples*) appears in the literature. In this paper we use *triples* as proposed in the RDF standard, `http://www.w3.org/TR/rdf-concepts/#section-triples`.

[3] Heart of Gold is an open source middleware for combining shallow and deep NLP components `http://heartofgold.dfki.de/`

[4] `http://sites.google.com/site/ontotravelguides/`

portals and services, such as by Seekda[5] [28, 29] or Mondeca[6].

2.3. Evaluation in Related Work

Whereas in established NLP branches standard test datasets together with evaluation criteria are available, there has been a lack of such dataset in the NLIKB field. Some typical problems related to the evaluation are:

- There is no widely accepted agreement on what is a correct result. This is a very subjective expression as it stands for e.g. a recall of the correct query [10], a number of answered questions [9], or even a number of queries generated as an output [14].
- Each system operates on a different ontology and the ontologies vary in their size and complexity.

As pointed out by [30], more attention should be paid to creating a standard evaluation set. Recently, various evaluation campigns have been initiated as a part of a semantic tool benchmarking project called *Seals*[7] (Semantic Evaluation At Large Scale). In the field of NLIKB, [6] and [7] evaluate different approaches to semantic search for expert and casual users. The Mooney Geoquery was chosen as testing ontology. The dataset contains information about U. S. geography, such as cities, rivers, mountains, countries, etc. Originally, the dataset was in Prolog (consisting of about 1000 Prolog facts), later it was converted into OWL by [31]. The question corpus consists of 880 user questions with relation to U. S. geography and was collected from undergraduate students of the authors of [13]. A deep quantitative analysis of this dataset was performed by [31] and the authors of [10] point out some interesting real-world issues of this dataset.

3. SWSNL System Architecture Overview

Our SWSNL system is a Java-based application consisting of various components. Figure 1 shows the architecture of the system together with the data-flow of an example query.

On the input, the user makes a query in natural language (NL). There are no restrictions on the query in terms of its length, as the user can express the query using keywords, a single sentence or the whole paragraph. We evaluated our system on two languages, Czech and English. The details of the NL queries are described in section 4.1, some examples of actual queries can be found in Appendix A.

The query is analysed by the Natural Language Understanding (NLU) component. This component produces a KB-independent semantic representation of NL query. The NLU component is comprised of three basic blocks, the NL query preprocessing, Named Entity Recognition (NER) and Semantic Analysis. The details of the NLU module are presented in section 6. The module uses a statistical model and thus require an annotated corpus of NL queries.

The semantic interpretation module transforms the KB-independent query annotation into a query language which is then executed against the KB. The KB is created from the existing data, as described in section 4.2. As a result, a list of KB instances with their properties is displayed to the user.

4. Target Domain

The target domain sets constraints for the demonstration of practical and theoretical values of the developed NLI system. Even in cases where the system is domain independent (most of the current NLI systems) a suitable domain will show both the potential and the limitations of the system. As was pointed out in the introduction, we focus on the Czech language in the first place. Due to its high degree of inflection and relatively free word order, it makes the task more challenging. The system can be, however, adapted to English which will be proven later in the evaluation.

We decided not to develop our system using any dataset from the existing related work. Firstly, there is no such dataset in the Czech langauge which meets requirements for performing end-to-end evaluation, namely availability of a KB that contains the desired domain information, a corpus of NL queries, and correct "answers" from the KB for each query. Secondly, the existing English corpora limit the queries only to single clauses or sentences. Thirdly, the widely-used Mooney Geoquery [13] corpus is, from our point of view, a typical corpus for the Question Answering task whereas our system is an replacement/enrichment of and existing form-based UI.

In the rest of this section, the process of collecting an NL query corpus and creating a KB will be depicted.

4.1. Natural Language Query Corpus

Whereas all existing NLI systems presented in the related work section expect queries to be single clauses or single sentences, we decided to go beyond the state of the art by allowing users to formulate their queries using one or more sentences to express their needs more precisely and more naturally.

We decided to use a voluntary-based way and started a Facebook campaign which can target a broad audience. The motivation stems from our previous experience with obtaining data for NLI systems [32] which showed us that obtaining data only from i.e. a group of students can negatively affect the data (in terms of a higher number of e.g. off-topic or silly queries).

The Facebook page contains a clear description of the "virtual" SWSNL system which says: *"There is a search*

[5] `http://www.seekda.com/`
[6] `http://www.mondeca.com/Clients`
[7] `http://www.seals-project.eu/`

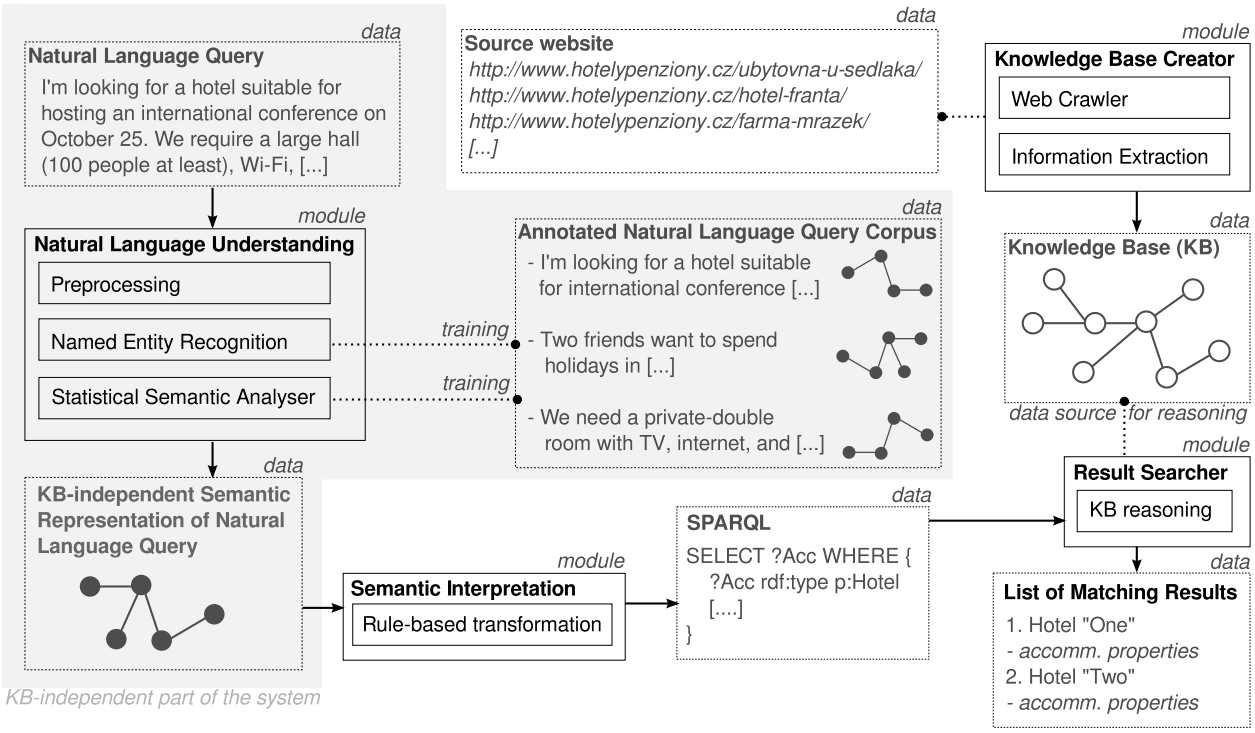

Figure 1: The SWSNL system architecture.

engine that operates on the accommodation domain in the Czech Republic. Try to search for an accommodation option that you really want by describing it using natural language instead of keywords. You are not limited to a single sentence." A few diverse examples followed. At that time, we had no back-end database or knowledge base, thus we did not limit the users to asking queries that can be found in any existing accommodation search portal. On one hand, we did not restrict the user queries to comply with a structure or content of a particular domain model. On the other hand, there was a risk that the KB would not contain the answers to the collected NL queries. We asked each user to pose one to three queries.

We gathered 68 queries in total. This was fewer than expected but the whole campaign was based upon spontaneity and our intention was not to push anyone into participation. Note that the participants were not asked to find answers to their queries. The main goal was to collect just the queries in this step. Examples of the user queries can be found in Appendix A.

Table 1 shows some statistics of the NL query corpus. Although there was no limit to the query length, the distribution of query lengths (in terms of the number of sentences used) illustrates that most of the users used single sentence and two sentence-long queries. However, the average query length shows that the queries are relatively long, which is also important from another point of view—the queries contain a lot of information (see the examples in Appendix A). This makes the corpus quite atypical compared to other existing corpora.

Corpus statistics	
Queries consisting of 1 sentence	37
Queries consisting of 2 sentences	23
Queries consisting of 3 sentences	8
Average query length (words)	24.9
Average number of sentences per a query	1.54
Number of unique participating users	42
Average number of queries per user	1.54

Table 1: Various NL query corpus statistics.

4.2. Knowledge Base and Domain Ontology

In order to evaluate a fully functional SWSNL prototype application, the underlying KB is essential. The KB is a module that stores complete knowledge about the covered domain. In the Semantic Web, ontologies are the main vehicle for domain modeling. Henceforth, we will use KB in terms of an ontology populated with instances.

As a source website for data mining, the portal `hotelypenziony.cz` was chosen. The site provides a large database of various types of accommodation options (from cottages to luxury hotels). Almost all entries have some location information, such as GPS coordinates or an exact address. Furthermore, each accommodation page contains some information about facilities, prices, etc. but these fields are not mandatory. Some entries are enhanced with textual descriptions, like e.g. a site or a restaurant description.

The entire site was downloaded using an in-house crawler written in Groovy. A pattern-based information extrac-

tion tool was developed in order to extract the structured content from the crawled web pages. The extracted data were then transformed into the corresponding ontology instances and properties. The ontology is stored in OWL[8] format.

The domain ontology structure was designed according to the extracted information, see Figure 2. The structure is more or less "flat". It has one important class `Accommodation` with many object and data properties. The reason for this design stems from the structure of the website (the information source), where one accommodation page is equivalent to one instance of the ontology class `Accommodation`.[9]

A transitional relation `isSubpartOf` (see Figure 2) allows capturing a hierarchy of `Location` instances, e.g. `Street` is a sub-part of `City`, etc. However, the property is not *functional*. This means that it is possible that one location is a sub-part of more than one other location. Typically, this is the case of `<c isSubpartOf a> & <c isSubpartOf t>` where `c` is an instance of `City`, `a` is an instance of `District` and `t` is an instance of `TouristRegion`. The task of creating such hierarchy only from the given address and the GPS coordinates can be found in [33].

Some existing ontologies for the accommodation domain were presented in section 2.2. Although some of these ontologies are more complex in terms of number of classes and the relations among them, we decided not to re-use any of these ontologies in our system. The reason was that the structure of the source domain (the website) and the structure of the existing ontologies were so different, that we simply did not see any benefit in trying to fit our data into such different ontologies.

4.2.1. Qualitative Analysis of the Knowledge Base

The total number of all instances (individuals) in the KB is 32,990 and the total number of all triples is 281,686. The total number of instances of the `Accommodation` class is 8641.

An accommodation instance contains two types of properties. We will distinguish them as *text properties* and *structured properties*.

Structured properties. These properties are both datatype and object properties (in OWL terminology) with exactly specified semantics. This kind of property contains structured information such as number values (e.g. various capacities or prices) or facilities (e.g. room facilities, sports activities).

Text properties. Text properties are four datatype properties (in OWL terminology), namely *accommDesc*, *siteDesc*, *conferenceDesc* and *restaurant*. These properties are mostly long text fields extracted from the accommodation website. They contain an unstructured description of the accommodation option, e.g. restaurant information, description of the site, etc. Furthermore, these properties do not have an exactly specified semantic meaning, as opposed to structured properties.

The structured properties are crucial for searching using a query language. On the contrary, if a piece of information is available only in text properties, it can hardly be searched for using a query language. In our KB, about 60% of all accommodation instances have no additional property, except for the obligatory contact information. This means that we cannot say anything specific about these accommodation options. The system only knows that a certain accommodation option exists in a certain place but no additional information is offered. However, almost all the user queries (see the examples in Appendix A) relate to a more specific type of accommodation options, with e.g. certain properties, prices, room types, etc. This means that almost 40% of accommodation options cannot be found when users ask for additional accommodation requirements. Note that this lack of structured properties is caused by the source web data, not by the ontology structure.

4.2.2. Ambiguity in the Knowledge Base

The main advantage of a well-designed ontology is its precise semantics. It allows complex semantic constructs to be used in order to infer new facts and query the KB using a query language.

In reality, systems must often deal with existing data since the amount of work needed to create and populate an ontology from scratch would be tremendous. Also our KB is completely created from existing data. As the original web data were not designed to respect precise domain semantics, a sort of semantic ambiguity or inconsisteny was transferred to the KB. Some notable problems are: (1) The accommodation types are quite ambiguous. There is no clear difference between e.g. *Hut* and *Cottage*, among others. (2) Lots of facilities are duplicated in some sense. For example, the KB contains four similar instances of a parking lot (namely *parkingLot*, *parkingLotWithLights*, *parkingGarage*, and *securedParkingGarage*). Note that we left all the data unchanged because it was not feasible to check and correct all the inconsistencies manually.

4.3. Gold Data Preparation

The *correct* answer (result) to a NL query is a set of accommodation instances that satisfy the user's requirements. Many NL queries contain sorts of subjective expressions, e.g. "cheap", "nice", "close to something", etc. It is necessary to decide in advance how to interpret these expressions consistently. The same interpretation must also be used later by the SWSNL system.

[8] `http://www.w3.org/TR/owl2-overview/`

[9] The inheritance relation between `RoomType` and (`PrivateDouble`, `PrivateSingle`, ...) was an authors' design choice. Another way to capture the same information would be e.g. a property `numberOfBeds`.

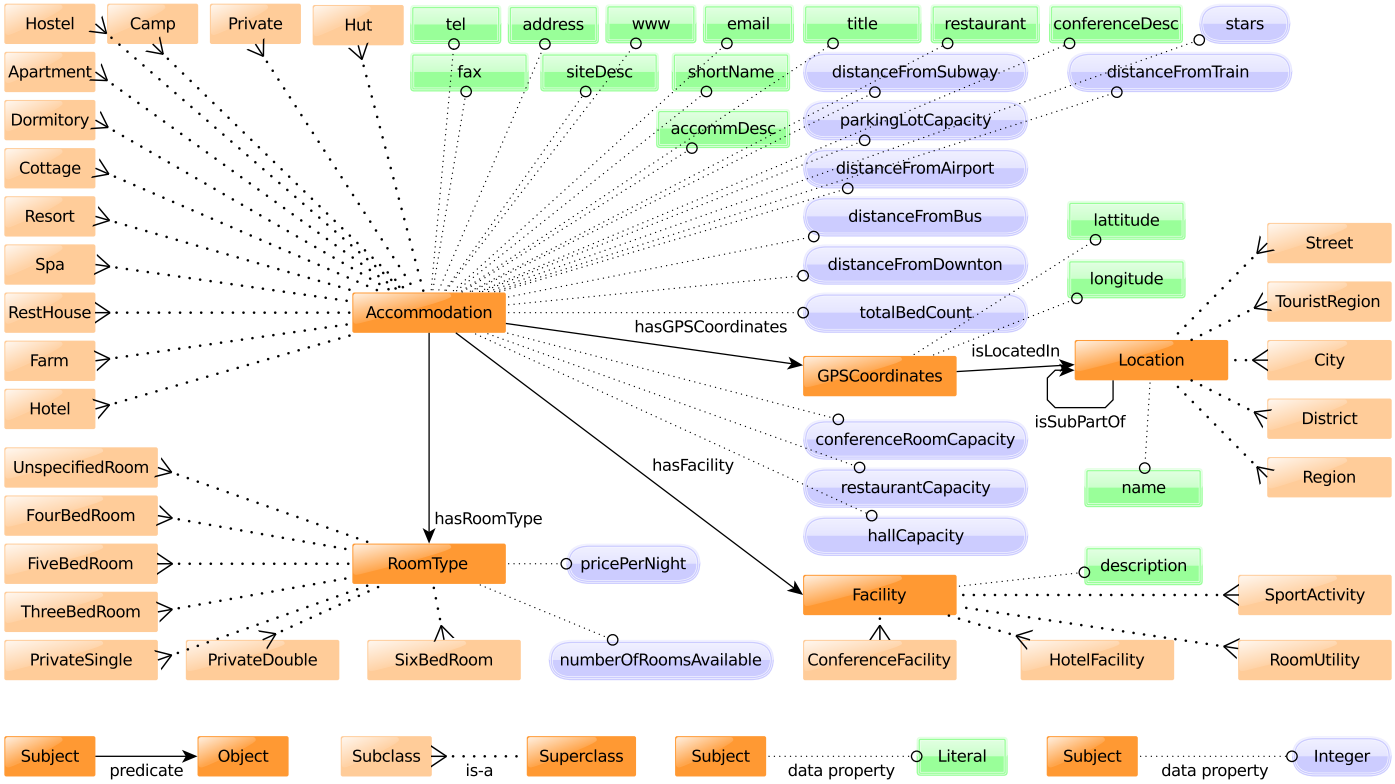

Figure 2: Structure of the *accommodation* ontology.

Expression	Interpretation
near a city	all places in the district in which the city is located
close to (a place)	the distance is less than 1 km
cheap	any room of the accommodation option has a price lower than 1000 CZK
cheapest	returns only one cheapest accommodation option

Table 2: Semantic interpretation of some NL expressions

Table 2 presents some examples of the semantic interpretation. The values (prices and distances) were set after a consensus was reached in our research team, however, this interpretation definitely depends on each user's point of view. We deal only with such expressions that can be quantified or expressed using the KB semantics.

4.3.1. Assigning the Correct Results

For this task we combined manually constructed *structured* and *fulltext* search queries.

Structured search. For each NL query a corresponding SPARQL query was manually constructed. We put as many constraints from the NL query as possible into the SPARQL query. For example, if the user asks for particular facilities (e.g. parking, internet), we add all of them in the SPARQL query. This ensures that only those accommodation instances that satisfy the constraints are selected from the KB. However, adding such strict criteria has the effect. First, the results that *partially* satisfy the requirements are discarded.[10] No ranking of results is involved.

Fulltext search. As presented in section 4.2.1, each accommodation instance can have a few text descriptions (labels). Such a description usually contains information which might be available in the structured properties as well (e.g. description of hotel facilities in the text and then again as structured properties). These text fields can be indexed and searched through using a fulltext search.

Finally, the whole process of assigning the correct results to each NL query was performed as follows:

- If the NL query can be completely expressed using SPARQL, create such a query and add the results to the correct results.
- If the NL query contains other requirements that cannot be expressed using SPARQL, filter the structured search results with a fulltext search and check the returned results manually.
- Moreover, for each NL query create the least restrictive SPARQL query (only with location and accommodation type) and filter the results using a fulltext

[10] This can be solved using OPTIONAL in SPARQL, however, at this stage we focused on results that satisfy all the requirements.

search. The fulltext keywords are manually selected according to the NL queries requirements. The results must be checked manually.

The gold data (in CSV format) as well as the KB (in OWL format) are licensed under CC-BY-SA[11] and are publicly available.[12]

5. Semantic Description of NL Queries

Some traditional approaches how NL query semantics can be expressed are e.g. frames, FOPL (first-order predicate logic), or semantic trees, among others. Related research [32, 34], based upon semantic trees, showed that describing the semantics of NL queries should follow a particular structure, e.g. using a schema for adding constraints to the possible semantic trees. Ontologies offer such a mechanism by allowing precise semantic relations between instances to be defined.

We will now discuss the term *semantic annotation*. In the Semantic Web, this term has multiple meanings, e.g. webpages are annotated according to a particular ontology (taxonomy) or named entities can be annotated with related instances from an ontology for furher named entity disambiguation [35], [36], [37]. Our usage comes from NLU research where sentences (or queries) can be annotated with some additional semantic information. Henceforth, the term *semantic annotation* will be used in the sense of a *semantic description of an NL query*.

On the Semantic Web, ontologies serve as a tool for storing domain information (KB). However, in our proposal *the semantic annotation of NL queries based upon ontologies has no direct relation to the ontologies for storing domain information (KB)*. This means that our semantic annotation is completely independent of a particular KB. Our semantic annotation uses ontologies *exclusively* to describe NL query semantics. The motivation for this separation was to enable e.g. switching between various KBs without affecting the NLU module or adapting the NLI to different language and keep the original KB.

5.1. Ontology for NL Query Semantics

The domain-independent query ontology is shown in Figure 3. Each word is represented as an instance of the *Word* ontology class with some additional morphological properties and its position within the sentence. All *Word* instances are chained using the *hasNextWord* property. An instance of the *Sentence* class covers a whole particular NL query.

The ontology also supports marking named entities. An instance of *NamedEntity* is connected to its content

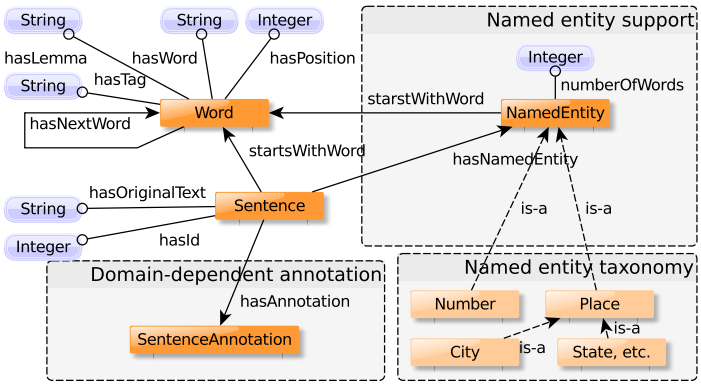

Figure 3: Domain independent NL query ontology (sentence ontology).

words using a property *startsWithWord*, its lenght is stored in the *numberOfWords* property.[13]

Traditionally, named entities are used to label expressions with a special meaning (e.g. cities, persons, countries, etc.). In this work, we extended the usage of a named entity to *all words/word groups that carry important semantic information in the NL query*. For example, we added the following domain named entities: *relative location* (covering phrases such as "near", "in the center", etc.), *relative price* (e.g. "cheap"), or *additional requirements* (e.g. "fitness", "internet", etc.), and others. Thus the named entities in our approach can represent both "traditional" entities (e.g. place, accommodation type, facilities, etc.) as well as certain domain-specific knowledge (e.g. relative location or price) and key phrases. We believe that such arbitrary extension of the meaning of the term *named entity* is still acceptable.

Named entities can also form a taxonomy as shown in Figure 3. This taxonomy can be extended in the ontology for a particular domain. For modeling the taxonomy of named entities, we use the inheritance of classes in OWL.

For each query, instances of the *domain-dependent query ontology* classes (triples) are created. A complete example of an annotated query can be found in Appendix B. The NL query ontologies are stored in the OWL format and were developed using Protégé[14].

5.2. NL Query Corpus Annotation

The whole NL query corpus from section 4.1 was annotated in the above-mentioned fashion. The annotated corpus is required for training the supervised statistical Natural Language Understanding module as will be shown later in section 6.

The corpus was annotated by two independent annotators. For this task we developed an annotation tool

[11] Creative Commons Attribution-ShareAlike 3.0 Unported License, `http://creativecommons.org/licenses/by-sa/3.0/`

[12] `http://liks.fav.zcu.cz/swsnl/`

[13] This marking style was our design choice; we are aware of various different approaches to named entity annotation, such as using offsets in GATE.

[14] Protégé is a free, open source ontology editor and knowledge base framework. `http://protege.stanford.edu/`

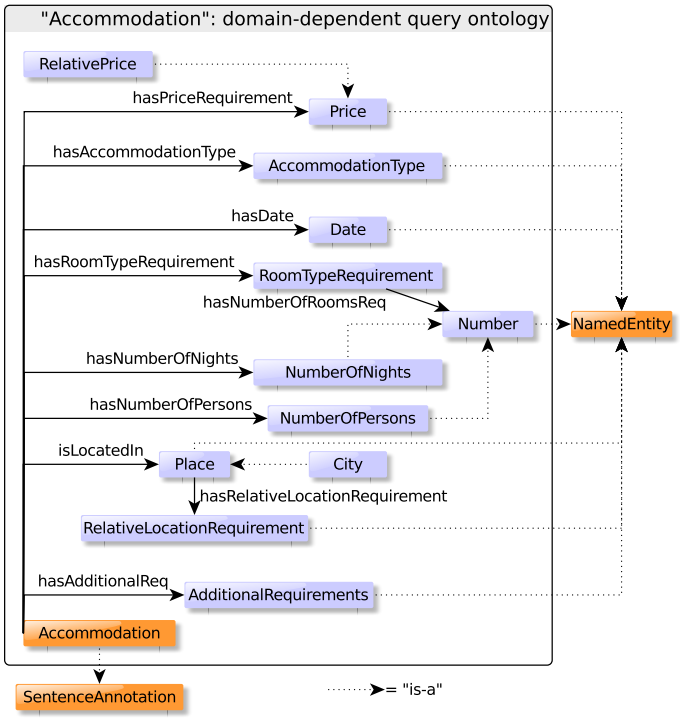

Figure 4: Annotation ontology for queries in the accommodation domain. The dotted lines represent the *is-a* relation.

that provided a balanced trade-off between usability and robustness. The editor hides the complexity of the underlying triples and the sentence ontology.[15] The inter-annotator agreement was measured on the number of matching triples[16] from the NL query annotations. The inter-annotator agreement was found to be $\kappa = 0.68$, 95% CI (0.504, 0.848).

5.3. Formal Definition of Semantic Annotation

Let $S = w_1 \dots w_W$ be a sentence S (a query) consisting of W words $w_{1\dots W}$ and let $T(Subj, Pred, Obj)$ be a triple of a subject $Subj$, a predicate $Pred$ and an object Obj[17]. Then the semantic annotation Sem of a sentence S is an unordered set of M triples

$$Sem(S) = \{T_1, T_2, \dots T_M\}. \quad (1)$$

Within $Sem(S)$, the triples can be connected. Formally, two triples

$$T_x(Subj_1, Pred_1, Obj_1), T_y(Subj_2, Pred_2, Obj_2) \quad (2)$$

can share their subjects or objects, so that $Obj_1 = Subj_2$ or $Subj_1 = Subj_2$, forming de facto a directed graph.

[15] This annotation task could be also achieved using other existing tools such as ontology-based gazetteer in GATE or CA Manager [38].

[16] This is explained in detail later in section 8.1.2

[17] Please note that *Subj*, *Pred* and *Obj* are just names of the first, second and third position in the triple. They may or may not be related to the syntactic role of words in a sentence.

Named Entities. Let N^τ be a Named Entity (NE) instance with type τ. Then

$$N^\tau_{j,k} = N^\tau_{span}(w_j \dots w_k) \quad (3)$$

is NE of type τ which spans the words w_j to w_k, where $1 \leq j < k \leq W$. This means that NE is associated with the words from the sentence. Let C be a *semantic concept*. C is not associated with any words.

Both N^τ and C can be parts of a triple so that each triple has one of the following forms:

$$T(Subj, Pred, Obj) = \quad (4)$$
$$\begin{cases} T(C, Pred, N^\tau_{j,k}) & \text{if subj is } C \text{ and obj is NE} \\ T(N^{\tau_1}_{j,k}, Pred, N^{\tau_2}_{o,p}) & \text{if subj and obj are NE} \\ T(C_a, Pred, C_b) & \text{if subj and obj are } C \end{cases}$$

where j, k, o, p are the NE spans, a, b are used to determine possibly different concepts C and τ_1, τ_2 can be different NE types.

6. Semantic Analysis of Natural Language Queries

The main task of the Natural Language Understanding component is to create a semantic description of previously unseen NL queries. Our NLU component is based upon a statistical model and supervised learning. Furthermore, the component uses both off-the-shelf and newly developed preprocessing tools.

All queries are preprocessed using tokenizer, part-of-speech tagger and lemmatizer from the Prague Dependency Treebank tokenizer and morphological tagger [39].

6.1. Statistical model

Using the formalism from section 5.3, the task of a semantic analysis is to find an appropriate semantic annotation $Sem(S)$ given a sentence S. Using probability, we can formulate the problem as follows:

$$P(Sem(S)|S) = P(T_1, T_2, \dots T_M|S). \quad (5)$$

We approximate the probability by treating the triples as independent, thus

$$P(T_1, T_2, \dots T_M|S) \approx \prod^M P(T_m|S) \quad (6)$$

where

$$P(T_m|S) = P(T_m(Subj, Pred, Obj)|w_1 \dots w_W). \quad (7)$$

Furthermore, we limit our model to the first two triple types from Equation 4. This means that *all* triple objects are NE and *some* triple subjects are NE, formally $T = T(C, Pred, N^\tau)$ and $T = T(N^{\tau_1}, Pred, N^{\tau_2})$. This limitation is based upon our observation that the other two types of triples are not currently present in the corpus.

In our model the triple probabilities are approximated as

$$P(T(C, Pred, N^{\tau})|S) \approx \quad (8)$$
$$P(T(C, Pred, N^{\tau})|N^{\tau}_{j,k}, w_{j-1}) \cdot P(N^{\tau}_{j,k}|S)$$

and

$$P(T(N^{\tau 1}, Pred, N^{\tau 2})|S) \approx \quad (9)$$
$$P(T(N^{\tau 1}_{j,k}, Pred, N^{\tau 2}_{o,p})|N^{\tau 1}_{j,k}, N^{\tau 2}_{o,p}, w_{j-1}, w_{o-1})$$
$$\cdot P(N^{\tau 1}_{j,k}|S) \cdot P(N^{\tau 2}_{o,p}|S),$$

where w is the lemma and $P(N^{\tau}_{j,k}|S)$ is obtained from named entity recognizer.

Given a training corpus $\mathbb{S}$ of sentences and their annotations $\mathbb{S} = \{S_n, Sem(S_n)\}$, the probabilities are estimated using Maximum Likelihood Estimation (MLE).

6.1.1. Named Entity Recognition

Named entity recognition is a crucial part of the query preprocessing. To achieve reasonable performance, we incorporate three various NER approaches.

Maximum Entropy NER. The maximum entropy-based NER (MaxEntNER) introduced in [40] was later extended and trained on a corpus from the Czech News Agency. The parameters of the corpus are very different from our NL query corpus since the Czech News Agency corpus consists of newspaper articles. However, the MaxEntNER is used to identify a few general named entities, such as *Place* and *City*.

LINGVOParser. A shallow semantic parser based upon handwritten grammars, LINGVOParser [41, 42] is used to identify named entities with complex structure, such as *Date*, *Currency* or *Number*.

String Similarity Matching. This NER approach (called *WordSimilarityTrainableNER*) was used as an *ad-hoc* tool to deal with the named entities that are not recognized by the two previous NERs. These are usually the domain-dependent entities. Recall that in our semantic annotation, the set of named entity types is much broader than in general NER because we use the named entities to cover all semantically important word groups (e.g. named entities like *AdditionalRequirements* or *AccommodationType*). The details about string similarity matching will be explained later in section 7.1.

OntologyNER. One of the key features of our system is that the NLU module is independent of the back-end KB. However, some systems from the related work (e.g. [11] or [20]) use the KB as a valuable source for recognizing various domain entities, such as places or various facilities. Therefore we also developed *OntologyNER* which exploits the knowledge stored in the KB. This vocabulary-based NER search for the named entity candidates using instance labels from the KB. The search uses string similarity matching which will be explained later in section 7.1.

7. Semantic Interpretation and Search

After the NL query is automatically annotated with its semantic description, it must be interpreted in order to find the desired results. Since our semantic representation of an NL query (see section 5) is independent of a particular KB, it is transformed into an ontology query language (SPARQL) to perform a search in the KB. As an input, a semantic annotation of an NL query is given. As an output, a SPARQL query for the back-end KB is produced.

The transformation consists of a set of domain-specific heuristic hand-written rules. Each rule processes particular triples from the NL query annotation and outputs a corresponding SPARQL snippet. An example in Figure 5 demonstrates how the semantic annotation of the NL query is transformed into the corresponding SPARQL query.

7.1. Matching Entities to Ontology Instances

Some named entitites recognized in the NL query, such as *City* or *Place*, are related to their corresponding ontology instances. For example, if the NL query search for a certain accommodation option in a particular location (i.e., *City["Praha"]*), the resulting SPARQL query must restrict the search to that entity instance (i.e., `?Accommodation p:isLocatedIn p:cPraha .`). Therefore, named entity disambiguation is necessary to map the recognized named entities to KB instances.

The mapping between named entities and the KB instances is based upon string similarity (or string distance). In order to select the best string metrics we manually created a small subset of the ontology instances paired with the named entities (50 pairs) and measured the accuracy of various types of string metrics. The best accuracy about 96% was achieved with the *Jaro-Winkler* distance [43]. Only a selected subset of the named entity types is matched using this technique, namely all instances of the ontology class *Place* and *Facility*.

Using this string similarity-based named entity disambiguation, three types of errors may occur. First, the named entity is expressed in such a way that the string similarity between the words and the KB instance is very low. Second, the desired named entity has no corresponding instance in the KB. These two types of errors are quite obvious. The third type of error is caused by the ambiguity within the KB. In some cases, two very similar instances have a very similar description. For example, the KB contains more than 30 instances with *Praha (Prague)* in their labels (referring to e.g. *District*, *City* or a city

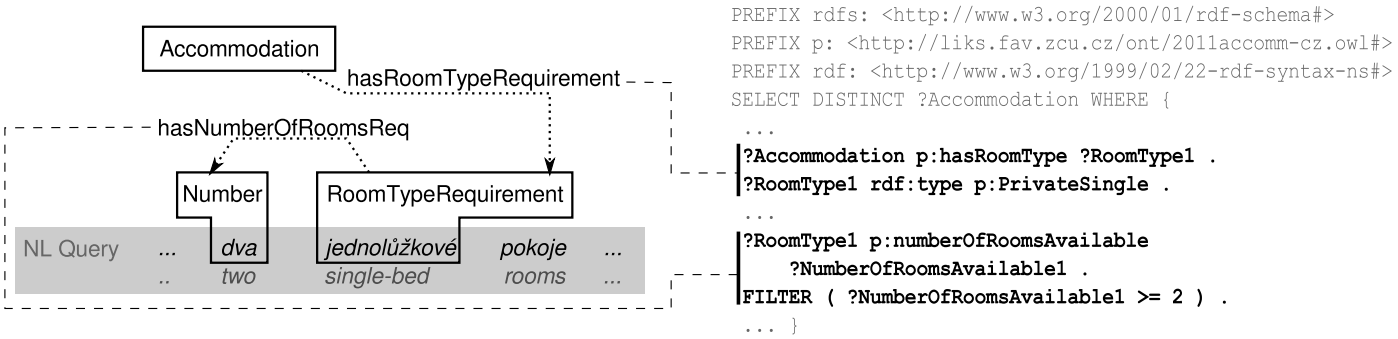

Figure 5: An example of tranforming an annotated NL query into SPARQL. The NL query semantic annotation sample contains two triples in this example, namely *(Accommodation, hasRoomTypeRequirement, RoomTypeRequirement["jednolůžkové"])* and *(RoomTypeRequirement["jednolůžkové"], hasNumberOfRoomsReq, Number["dva"])* where *RoomTypeRequirement* and *Number* represent named entities. These two triples are translated into the two corresponding SPARQL snippets on the right-hand side. Notice that the NL query annotation is created according to the NL query ontology (see Figure 4), whereas the SPARQL query is bound to the particular back-end KB (see Figure 2).

part). Currently, we do not deal with this kind of ambiguity. However, it could be easily solved i.e. by choosing the most likely instance learned from the training data or by showing a clarification dialogue (the case of [11] or [9]).

7.2. Interpretation of Named Entities with Numbers and Dates

The named entitity types *Number* and *Date* are not related to any ontological instances (they do not directly represent any instance from the KB). The LINGVOParser tool used to recognize this type of named entities (see section 6.1.1) also has built-in support for the semantic interpretation of the content of the named entities, namely numbers and dates. The tool utilizes hand-written semantic grammars for translation into a computer representation (e.g. integers or dates). This allows interpretations of these entities to be integrated into the resulting SPARQL query.

7.3. Semantic Reasoning and Result Representation

Given all the mechanisms introduced in the previous sections, it is possible to formulate the SPARQL query. Once the query is created, it can be passed to an ontology reasoner. Currently, various OWL reasoner Java implementations are available. They also vary in their OWL expressivity, which means what subset of the OWL semantics they can handle. To select a suitable reasoner for our system, a simple test was carried out. The most important OWL feature we use is the transitivity among location instances (see section 4.2). Thus, a simple SPARQL query was executed to obtain all instances in a particular location and its sub-locations. Table 3 outlines the results of this test. We tested reasoners from the Jena package as well as the Pellet reasoner. Only reasoners that support transitive properties are listed. Given the measured performance among reasoners, the Pellet reasoner was selected for our system. The performance of the reasoners will be discussed later in section 9.

Reasoner class	**Total time**
PelletReasoner	15m 14s
OWLMicroReasoner	55m 5s
OWLFBRuleReasoner	5h 15m 26s

Table 3: Performance comparison of various OWL reasoners.

The result of the semantic search is a set of accommodation instances. The system displays all the retrieved accommodation instances together with their properties.

8. Evaluation

In this section, we thoroughly evaluate our SWSNL system. The section is structured as follows. First, we focus on the evaluation of the Natural Language Understanding module (introduced in section 6). Various combinations of NER are also tested in this part of the evaluation. Second, we present an end-to-end evaluation of our system. We describe the baseline approach and compare our system results with the baseline results. Third, we evaluate our NLU module on another two corpora and languages to demonstrate its portability capabilities.

Our main corpus (the NL Query corpus on the accommodation domain, see section 4.1) consists of 68 queries. We split the data into training and test set using the *leave-one-out cross-validation* [44]. In this method, each fold of the cross validation has only a single test example and all the rest of the data is used in training. We use this method because the available data is very small.

8.1. Natural Language Understanding Module Evaluation

The NLU module (section 6) works in two stages: the Named Entity Recognition and the ontology-based statistical model for the semantic analysis. Its evaluation will be split into two sections, too. We use standard metrics, namely the *precision (p)*, the *recall (r)* and the *F-measure*

Entity type \ NER	WSim	MaxEnt	Onto	LinPar
AccommodationType	**0.90**	0.0	0.0	0.0
Place	0.37	**0.45**	0.22	0.0
Date	0.0	0.22	0.0	**0.28**
NumberOfPersons	**0.62**	0.0	0.0	0.0
RelativeLocationReq	**0.70**	0.0	0.0	0.0
City	0.33	**0.70**	0.35	0.0
Price	**0.72**	0.13	0.0	0.0
AdditionalReq	**0.57**	0.0	0.04	0.0
Number	0.32	0.10	0.0	**0.37**
RelativePrice	**0.87**	0.0	0.0	0.0
RoomTypeReq	**0.55**	0.0	0.0	0.0

Table 4: F-measure performance of various NER on different entity types. Some NER names are abbreviated; *WSim* is *WordSimilarityTrainableNER* and *LinPar* is *LINGVOParserNER*.

(Fm):

$$p = \frac{tp}{tp + fp}, \quad r = \frac{tp}{tp + fn}, \quad Fm = \frac{2pr}{p + r}, \qquad (10)$$

where *tp* are true positives, *fp* are false positives, and *fn* are false negatives [45].

8.1.1. Evaluation of NER

The system uses four types of NER, namely the *MaxEntNER*, the *WordSimilarityTrainableNER*, the *LINGVOParser*, and the *OntologyNER* (see section 6.1.1). The *correct result* (true positive) is when the named entity from the annotated corpus exactly matches the entity returned by NER, in terms of both the NE type and the word span. The results are shown in Table 4 and can be interpreted as follows.

The second column *(WSim)* displays the results for the *WordSimilarityTrainableNER*. This NER is based on lexical similarity between the training examples and the unknown word(s). This NER performs best mostly for domain-dependent entities, such as *AdditionalReq* or *AccommodationType*, because these entities are expressed using similar words in the data. We assume that this approach is sufficient for domain-dependent entities.

Our assumption that the *MaxEntNER*, even though it is trained on a completely different corpus, would yield acceptable results for general named entities, such as cities or places, is validated by the results in the third column *(MaxEnt)*. This NER outperforms the rest of the NERs in *City* by almost 100% relatively.

The poor results obtained by *OntologyNER* in the fourth column have the following explanation. This NER exploits the lexical similarity between KB instances and the word(s) being recognized. The results for named entities *City* and *Place* are thus similar to the results from the second column *(WordSimilarityTrainableNER)*. However, the performance of the *OntologyNER* is surpassed by the *MaxEntNER*, as it involves many features for entitity recognition, such as the local context, word shape, etc.

Finally, the last column shows that for complex named entities in natural language, such as dates or numbers, the entity-specific hand-crafted NER *LINGVOParser* would be the best choice.

CombinedNER. For each entity type we selected the best performing NER from Table 1 and created the *CombinedNER*. This NER will be used in the overall evaluation and the semantic analysis evaluation in the next sections. However, we expect that with larger training set it would be reasonable to use a more clever NER combination, e.g. a linear interpolation model with combination parameters set by the Expectation-Maximization algorithm.

8.1.2. Semantic Analysis Evaluation

The result of the semantic analysis is a semantic annotation of a NL query (recall section 5.3). However, the annotation of a NL query can also be seen as a set of connected triples. For each NL query, we compute precision, recall and F-measure using the set of the gold triples (the human annotation from the NL query corpus) and the triples returned by the semantic analysis module.[18] A *true positive* result triple is such a triple that matches the gold triple as follows:

- The predicates are the same.
- The objects/subjects have the same ontology class, i.e. <*City, pred, obj*> and <*City, pred, obj*> match while <*City, pred, obj*> and <*Place, pred, obj*> do not.
- Furthermore, if the objects/subjects are named entities, their positions in the sentence and their contents are the same, i.e. <*subj, pred, City(Velké*$_1$, *Popovice*$_2$*)*> and <*subj, pred, City(Popovice*$_2$*)*> do not match.

The results of semantic analysis module (with the *CombinedNER* incorporated) are shown in Table 5. The most important factor that influences the performance is the small size of the NL query corpus. Only 67 training examples[19] are not sufficient to train the semantic model properly. Moreover, the very important part of the *CombinedNER* for recognizing non-general named entities, the *WordSimilarityTrainableNER*, is also completely trained from the data.

	p	**r**	**Fm**
CombinedNER + Semantic Model	0.66	0.34	0.45

Table 5: The results of the semantic analysis with the *CombinedNER*

[18] We also considered some other measures developed for semantic annotation evaluation, but none of them was suitable for our purposes. BDM [46] computes semantic similarity between two annotations of the same token in a document. The tree edit distance [47] expects the semantic annotation to form a tree.

[19] Note that we still use the *leave-one-out cross-validation* which ensures that the training and test data are distinct.

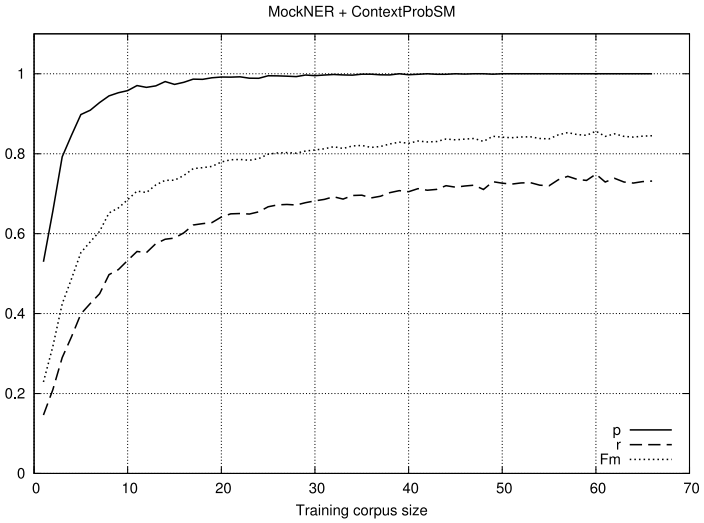

Figure 6: The semantic model performance depending on the training set size.

In the following test, we evaluated the semantic analysis model's dependency on the training data size. Using the same NL query corpus, we simulated 100% accuracy of the NER module (called *MockNER*) and changed the proportions of the training and test sets. Given the results from Figure 6, we can draw the following conclusion. Even with small training data, most of the returned triples are correct ($p \rightarrow 1$). However, to obtain all triples of the semantic annotation of a single NL query, a larger training set is probably required (r is growing more slowly).

8.2. End-to-end Evaluation

Each query from our NL query corpus has its own corresponding set of correct search results (gold results), as described in section 4.3.1. This allows us to evaluate the complete SWSNL system. We performed three experiments. First, the baseline fulltext-based search system was evaluated. Second, we tested our SWSNL system. Third, we performed the same test with a simulation of 100% correct Semantic Analysis module.

For each query the system returns a set of accommodation instances. There is no ranking of the results because the KB querying is purely boolean which means that the single result either matches or not and no further discrimination is given. Thus, for each NL query we compute the precision, the recall, and the F-measure of the returned result set according to the gold set. The final result is then the average of the results for each query (the Macro F-1 measure).

8.2.1. Baseline System

As a baseline for the end-to-end evaluation we used the fulltext search. The crawled documents from the source website were indexed using Apache Lucene.[20] In the preprocessing, the best available Czech stemmer [48] was used, the stop-words were filtered and the content was lowercased. In this experiment, each NL query was treated as a bag of words with stop-words removed.

[20] `http://lucene.apache.org/core/`

		p	**r**	**Fm**
(a)	Baseline fulltext	0.02	0.42	0.03
(b)	**SWSNL**	**0.25**	**0.70**	**0.37**
(d)	Correct semantic annotation	0.73	0.84	0.78

Table 6: The end-to-end results of (a) the baseline fulltext search system, (b) the search using our SWSNL system, and (c) the search using the correct semantic annotation of NL queries.

Other standard measures used in Information Retrieval (IR) for evaluating ranked results are Mean Average Precision, Precision at k (P@k) and R-Precision (R-Prec), see e.g. [45]. These metrics are suitable only for ranked results as retrieved e.g. by a fulltext search. Since we compare the baseline with our SWSNL system which does not produce ranked results, it is not reasonable to measure the baseline using these metrics.

The baseline system results are shown in Table 6, row (a). The results show very clearly that the keyword-based search performs poorly for such complicated tasks. The relatively high recall means that there are many hits in the retrieved set. The low precision is caused by the fact that the search model is not discriminative (i.e. it does not filter the results according to a certain location, etc.). However, this might serve as an approximation of results that would be returned by the current state-of-the-art general-purpose search engines. The parameter N for selecting only the top N retrieved documents was set empirically in order to maximize the F-measure.

8.2.2. SWSNL System Results

Table 6, row (b) shows the overall results of the SWSNL system. The most important conclusion, given these results, is that our semantic web search *significantly outperforms* the fulltext search. Whereas the fulltext-based search simply fails in this complex search task, our SWSNL system is able to yield acceptable results. The obtained F-measure of 36% may seem low but high numbers are not common in challenging IR tasks.

8.2.3. SWSNL System Results With Simulation of Correct Semantic Analysis

We also tested the upper bounds of our SWSNL approach by simulating the 100% correct Semantic Analysis module. The numbers in Table 6, row (c), confirm that some results cannot be found by the purely structured search. This topic was already discussed in section 4.3 where the process of creating the gold results was presented. In more detail, some of the results in the gold data were manually assigned to the queries according to their textual description. Such results are, however, not currently retrieved by our SWSNL system, since the search operates only on structured properties (cf. section 4.2.1). The second reason that affects the performance in this test

is that some errors and ambiguity are encountered during mapping named entities to KB instances (as already discussed in section 7.1).

8.3. Evaluation of NLU Module on Other Domains and Languages

Although we put our main effort into the end-to-end evaluation on the accommodation domain in the Czech language, we also tested our NLU component on two different corpora in order to prove its usability across domains and languages.[21] In the following section, only the NLU module is evaluated as there exist no databases/KBs for these datasets for an end-to-end evaluation.

8.3.1. Czech Public Transportation Queries Corpus

The Czech Public Transportation Queries Corpus *(ConnectionsCZ)* is a NL query corpus in the Czech language. It contains 238 queries in the public transportation domain. Most of the queries ask for a certain connection from one place to another, for train types, travel times, etc. The corpus is a subset of a larger corpus used in [32].

This type of corpus follows the fashion of corpora for Spoken Language Understanding (SLU) or human-computer spoken dialogue, such as ATIS [49], etc. Since our current work deals with search and Semantic Web technologies, we are not convinced that our SWSNL system would be the best approach for this domain and task. Nevertheless, this evaluation serves as proof that our semantic model is domain independent and can perform well if a larger training corpus is available.

In order to port the corpus into our system, we created an ontology for an NL query in this particular domain (see section 5). The domain-dependent query ontology is shown in Figure 7. The queries were then annotated according to this ontology.

We conducted the two following experiments. In the first one, we used the simulation of 100% NER. Figure 8 (a) illustrates the improvement in the semantic analysis as the training set size grows.

In the second experiment, we used the same NER configuration as in the accommodation domain (namely the *Combined NER*, see section 8.1.1). The results are shown in Figure 8 (b). The improvement in the performance is caused by the larger training size for both the NER and the semantic model.

8.3.2. ATIS Corpus Subset

The ATIS (Air Travel Information Service) [49] corpus contains travel information queries in English and it has been widely used in NLU research. It is one of the commonly used corpora for testing of semantic analysis systems, used for evaluation in e.g. [50], [51], [52] and [53].

[21] Both corpora can be found at `http://liks.fav.zcu.cz/swsnl/`.

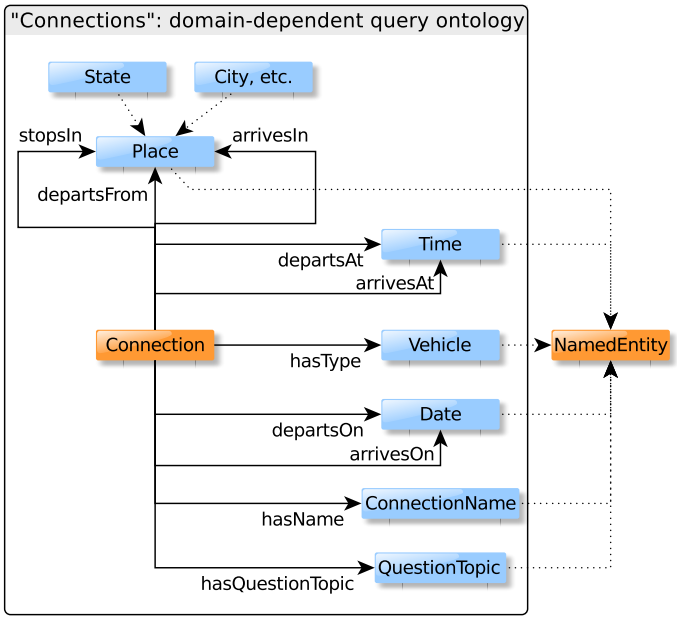

Figure 7: The ontology of the queries in the public transportation domain. The dotted lines represent the *is-a* relation.

We extracted a small subset of the corpus (348 sentences) and annotated them using the same query ontology as for the Czech Public Transportation Queries Corpus, see previous section and Figure 7.

Since our NER components are able to deal with the Czech language only, we performed only one test with the simulation of 100% NER on the ATIS corpus. The test is intended to evaluate the performance of our semantic model on a different language. The results are shown in Figure 8 (c).

We decided to test our system on the ATIS corpus just to get an insight into the possibilities of porting the semantic model to another language. This should not serve as a comparison with other state-of-the-art semantic parsers, for two reasons. Firstly, our semantic annotation is different from the original frame-based ATIS annotation. Secondly, we use a small subset of the whole corpus, which also negatively affects the performance. The complete corpus was not used because a manual annotation according to our semantic representation was required and it was not feasible to annotate the complete corpus.

The obtained results illustrate that our semantic analysis model can be used across different languages. No tweaking is required to adapt the system to English. The achieved performance (about 0.73 F-measure) is satisfactory, given the fact that the system was primarily developed on a different domain and language.

8.3.3. Final Remarks to Evaluation

We will also comment on the fact that our system was not evaluated on the already-mentioned Mooney Geoquery dataset (see section 2.3). Supported by our knowledge of question answering [54] the Mooney Geoquery is a typi-

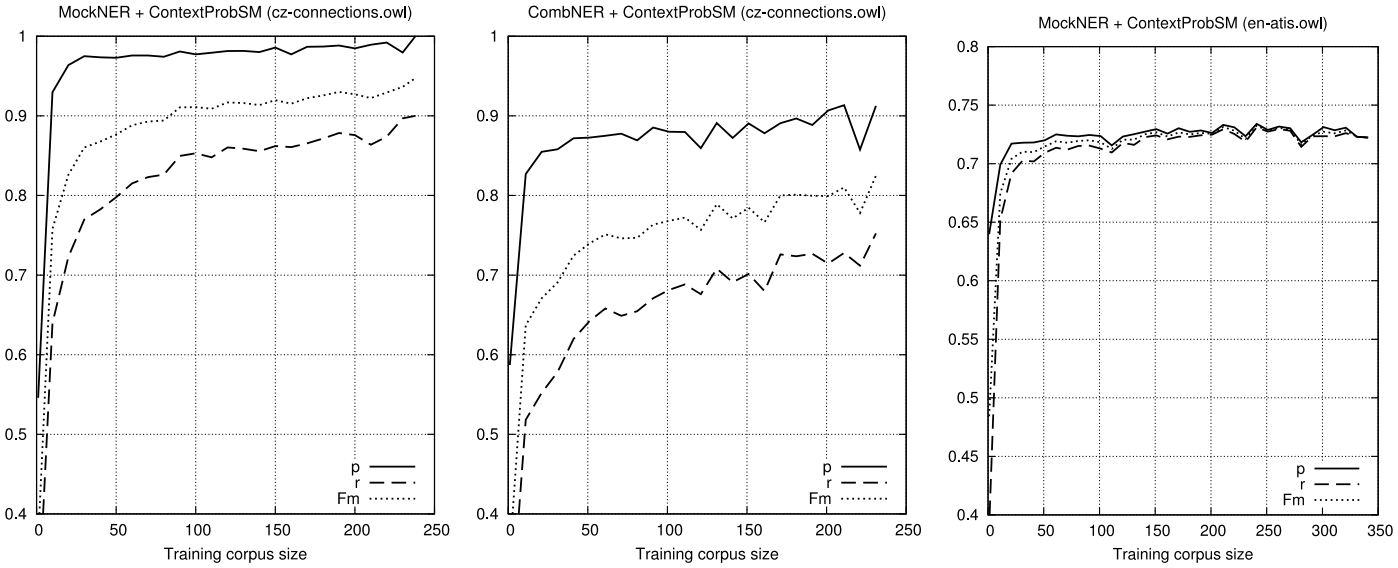

Figure 8: The evaluation of the NLU module on different domains and languages w.r.t. the increasing sizes of the training data. (a) Semantic analysis on the Czech Public Transportation Queries Corpus with a simulation of a 100% correct NER. (b) Semantic analysis on the Czech Public Transportation Queries Corpus with the Combined NER. (c) Semantic analysis on the English ATIS corpus with a simulation of a 100% correct NER.

cal factoid question-answering corpus. The answers may be lists of instances (e.g. *"what states border florida ?"*), numbers (e.g. *"how long is the rio grande river ?"*), or an aggregated result (e.g. *"how many square kilometers in the us ?")*. On contrary, our system is developed as a replacement of a form-based UI for information retrieval. The types of queries in our corpus are significantly different from the Mooney Geoquery queries. In our case, the user wants to find a list of instances that fulfill his or her requirements instead of finding some facts about entities.

9. Conclusion

This final section outlines the major contributions of this article as well as open issues and future work. A Semantic Web Search using Natural Language is, beyond all doubt, a challenging task. In order to develop a real-world system that could be deployed for a public use, many theoretical and practical problems must be resolved. We will examine these problems from various perspectives and also propose solutions to some of them.

9.1. Performance Issues

One of the most critical issues which actually prevents our system from being tested in the real Web environment is the performance of the ontology reasoning. As already shown in Table 3 on page 11, we tested several OWL reasoners and their ability to deal with transitive properties.

The *Pellet* reasoner requires approximately 15 minutes on average to answer a single SPARQL query with transitive relations. We suspect that the reason for this is the size of our KB (281,686 triples, see section 4.2.1). However, to the best of our knowledge, there is no room for improvement without changing the back-end Semantic Web tools completely.

Possible solutions. First, another ontology storage engine can be used, e.g. Sesame[22]. Second, it is possible to do materialisation first and then querying (as is OWLIM[23]).

9.2. Problems Caused by Actual Web Data

Many practical issues related to the KB content were pointed out in section 4.2.1. The most important problem is the missing structured data. In other words, the data from the source Web site are incomplete. However, this will be the case in most of the systems that use public Web sources for populating their KBs. It is not feasible either to check or even to correct the data manually.

Another problem is caused by the inconsistent source data. In our scenario, various KB instances have the same label which makes the mapping from the named entities to ontology instances (NE disambiguation) much harder (see section 7.1).

Possible solutions. Better data post-processing after the information extraction step. Nevertheless, a huge manual effort would be required.

9.3. Future Work

There are four directions that are worth exploring further:

[22] `http://www.openrdf.org/`
[23] `http://www.ontotext.com/owlim`

Larger NL query corpus. The main limitation of our current semantic model performance is the small size of the corpus. Thus, it would be beneficial to obtain more data in e.g. a bootstrapping manner when the prototype is deployed on a testing server and is accessible by the public.

Integrating fulltext search. A combination of structured search and fulltext search is a promising future task, based upon our preliminary research.

Better Named Entity Recognition. The results definitely show that the NER component is crucial in the semantic search. The better the NER is, the better the semantic model performs and the more precise the search results are.

Performance Improvement. Replacing the back-end with more scalable Semantic Web tools or with a relational database.

9.4. Notes to Related Work

We would also like to discuss some of our findings regarding the related work. Since our goal was to design a fully functional real-world system, many details must have been taken into account. However, in some related work these details were ignored od hidden, from our point of view. Firstly, many systems operate either on small ontologies [10, 9] or simple domains [55, 17] therefore it cannot be assumed how would these approches perform on a decent-scale ontologies. Secondly, the natural language queries used for development and testing of some systems are mostly collected by the research teams themselves or by their students [17, 56, 13, 14, 57, 26]. In our opinion, a different way of obtaining queries should be considered in a real-world scenario to avoid any potential skewness in the data. Thirdly, many systems rely on full syntax processing of the queries and a heuristic processing of the semantics in their NLU component [9, 58, 10, 11]. Although syntactic parsing and using rules for NLU have been a widely used approach, however, in some cases this may not a feasible solution, e.g. when a syntactic parser is not available for that particular language. Finally, an exhaustive evaluation and a realistic discussion of the limitations is missing in some systems [59, 14].

9.5. Main Contributions

This article describes an attempt to develop a fully-functional semantic search system with a natural language interface. The following list summarizes the main contributions.

The complete accommodation web portal was converted to the OWL ontology which allows a geospatial search using an automatically-created hierarchy of locations. We also identified many practical problems that are likely to arise when a system must deal with noisy data obtained from the Web.

A corpus of natural language queries in the Czech language was collected using a social media-based approach. We proposed a statistical semantic model for semantic analysis of natural language queries. It is a language-independent model which is trained from labeled data and does not depend on syntactic parsing.

We created an evaluation dataset on which our system was tested. This dataset may be a valuable resource for any related research on natural language interfaces operating on ontologies or as a front-end to information retreival systems. We offer this dataset and two other corpora under a licence which allows both research and commercial purposes.

Acknowledgments

This work was supported by grant no. SGS-2010-028 and by the European Regional Development Fund (ERDF), project "NTIS - New Technologies for Information Society", European Centre of Excellence, CZ.1.05 /1.1.00/02.0090.

Appendix A. Query Examples

Here are a few examples of NL queries in Czech with their English translation from the NL query corpus as introduced in 4.1.

- *Ráda bych se ubytovala v Karlštejně s výhledem na hrad a královské vinice, nejlépe s polopenzí, parkováním a úschovnou kol, pro 2 lidi od pátku do neděle.* — I'd like to have accommodation for two people in Karlštejn with a good view of the castle, including half-board, a parking lot, and a bike to hire from Friday till Sunday.

- *Jsme dva a chceme strávit jarní prázdniny v malebném hotýlku nebo apartmánu v Beskydech, blízko modré sjezdovky, předpokládáme lyžárnu, vyžadujeme polopenzi. Bazén, pingpong, wellness a podobné legrácky potěší, ale nejsou nezbytné.* — Two persons want to spend the spring holidays in a cute hotel or apartment in the Beskydy Mountains. We require a ski-locker room and full board. Swimming pool, a ping-pong table and some wellness will be a plus but that's not necessary.

- *Hledám levné víkendové ubytování pro 3 odoby v Jindřichově Hradci nebo blízkém okolí, v blízkosti půjčovny jízdních kol nebo s možností uskladnění vlastních, pokoj nejlépe s krbem a možností připojení na internet, vlastní sociální zařízení podmínkou.* — I'm looking for cheap weekend accommodation for 3 persons in the area of Jindřichův Hradec, near to a bike-rental store or with a bike locker room. Room including a fireplace, Internet and a private bathroom.

- *Hledám hotel na pořádání mezinárodní konference dne 25.10.. Hotel musí mít sál pro minimálně 100 lidí, wifi pokrytí, možnost uspořádání večeře, a minimálně 10 dvůjlůžkových pokojů. Upřednostňuji hotel s vlastní prezentační technikou* — I'm looking for a hotel suitable for hosting an international conference on October 25. We require a large hall (100 people at least), Wi-Fi, banquet, conference equipment. 10 double-bed rooms are minimum.

Appendix B. Complete NL Query Annotation Example

Please refer to Figure B.9.

References

[1] M. L. Wilson, B. Kules, m. c. schraefel, B. Shneiderman, From keyword search to exploration: Designing future search interfaces for the web, Foundations and Trends in Web Science 2 (1) (2010) 1–97.

[2] E. Demidova, P. Fankhauser, X. Zhou, W. Nejdl, Divq: diversification for keyword search over structured databases, in: Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval, SIGIR '10, ACM, New York, NY, USA, 2010, pp. 331–338.

[3] H. He, W. Meng, C. Yu, Z. Wu, Constructing interface schemas for search interfaces of web databases, in: A. Ngu, M. Kitsuregawa, E. Neuhold, J.-Y. Chung, Q. Sheng (Eds.), Web Information Systems Engineering – WISE 2005, Vol. 3806 of Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2005, pp. 29–42.

[4] E. Kaufmann, A. Bernstein, How useful are natural language interfaces to the semantic web for casual end-users?, in: Proceedings of the 6th international The semantic web and 2nd Asian conference on Asian semantic web conference, ISWC'07/ASWC'07, Springer-Verlag, Berlin, Heidelberg, 2007, pp. 281–294.

[5] E. Prud'hommeaux, A. Seaborne, SPARQL Query Language for RDF, W3C Recommendation (2008).

[6] K. Elbedweihy, S. N. Wrigley, F. Ciravegna, Evaluating semantic search query approaches with expert and casual users, in: 11th International Semantic Web Conference (ISWC2012), Boston, USA, 2012.

[7] K. Elbedweihy, S. N. Wrigley, F. Ciravegna, D. Reinhard, A. Bernstein, Evaluating semantic search systems to identify future directions of research, in: R. García-Castro, N. Lyndon, S. N. Wrigley (Eds.), Second International Workshop on Evaluation of Semantic Technologies, no. 843 in CEUR Workshop Proceedings, 2012, pp. 25–36.

[8] V. Lopez, M. Fernández, E. Motta, N. Stieler, PowerAqua: Supporting users in querying and exploring the Semantic Web, Semantic Web (2011) 1–17.

[9] V. Lopez, V. Uren, E. Motta, M. Pasin, AquaLog: An ontology-driven question answering system for organizational semantic intranets, Web Semantics 5 (2) (2007) 72 – 105.

[10] P. Cimiano, P. Haase, J. Heizmann, M. Mantel, R. Studer, Towards portable natural language interfaces to knowledge bases – the case of the orakel system, Data and Knowledge Engineering 65 (2) (2008) 325 – 354.

[11] D. Damljanovic, M. Agatonovic, H. Cunningham, Natural language interfaces to ontologies: Combining syntactic analysis and ontology-based lookup through the user interaction, in: L. Aroyo, G. Antoniou, E. Hyvönen, A. ten Teije, H. Stuckenschmidt, L. Cabral, T. Tudorache (Eds.), The Semantic Web: Research and Applications, Springer Berlin / Heidelberg, 2010, pp. 106–120.

[12] Cunningham et al., Text Processing with GATE (Version 6), university of Sheffield Department of Computer Science (April 2011).

[13] L. R. Tang, R. J. Mooney, Using multiple clause constructors in inductive logic programming for semantic parsing, in: Proceedings of the 12th European Conference on Machine Learning, Springer-Verlag, London, UK, 2001, pp. 466–477.

[14] C. Wang, M. Xiong, Q. Zhou, Y. Yu, Panto: A portable natural language interface to ontologies, in: Proceedings of the 4th European conference on The Semantic Web: Research and Applications, ESWC '07, Springer-Verlag, Berlin, Heidelberg, 2007, pp. 473–487.

[15] E. Kaufmann, A. Bernstein, L. Fischer, NLP-Reduce: A "naïve" but Domain-independent Natural Language Interface for Querying Ontologies, in: 4th European Semantic Web Conference (ESWC 2007), 2007, pp. 1–2.

[16] Óscar Ferrández, R. Izquierdo, S. Ferrández, J. L. Vicedo, Addressing ontology-based question answering with collections of

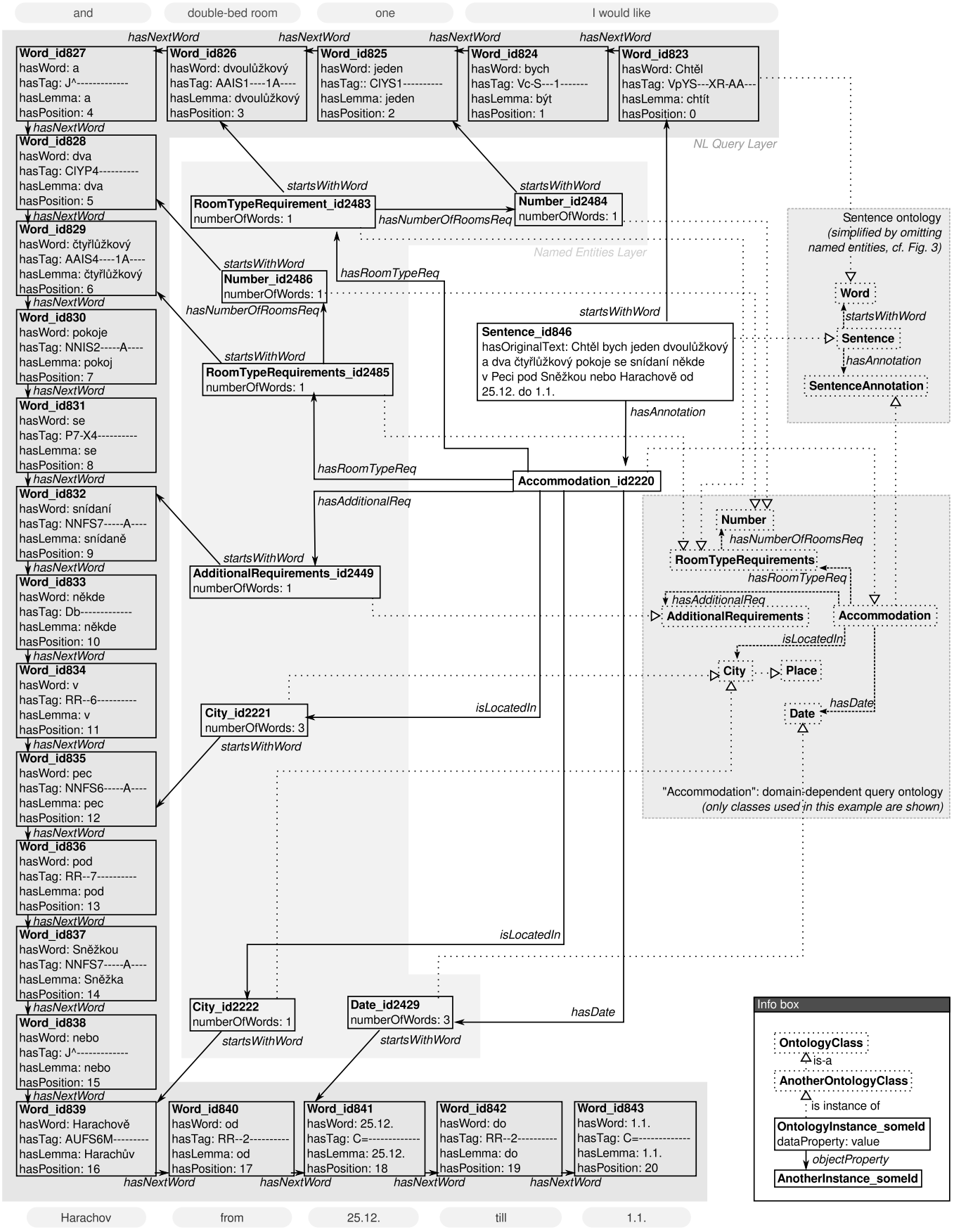

Figure B.9: This example displays a complete semantic annotation of a query from the accommodation corpus. The query can be translated as *"I would like one double-bed room and two four-bed rooms including breakfast somewhere in Pec pod Sněžkou or Harachov from 25. 12. till 1. 1."*. We added the translation to the corresponding tokens for easy understanding.

user queries, Information Processing & Management 45 (2) (2009) 175 – 188.
[17] A. Frank, H.-U. Krieger, F. Xu, H. Uszkoreit, B. Crysmann, B. Jörg, U. Schäfer, Question answering from structured knowledge sources, Journal of Applied Logic 5 (1) (2007) 20 – 48.
[18] C. Pollard, I. A. Sag, Information-based syntax and semantics: Vol. 1: fundamentals, Center for the Study of Language and Information, Stanford, CA, USA, 1988.
[19] A. Copestake, D. Flickinger, C. Pollard, I. Sag, Minimal recursion semantics: An introduction, Research on Language and Computation 3 (2005) 281–332.
[20] D. Damljanovic, V. Tablan, K. Bontcheva, A text-based query interface to owl ontologies, in: N. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odjik, S. Piperidis, D. Tapias (Eds.), Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08), European Language Resources Association (ELRA), Marrakech, Morocco, 2008, pp. 361–375.
[21] G. Zenz, X. Zhou, E. Minack, W. Siberski, W. Nejdl, From keywords to semantic queries–incremental query construction on the semantic web, Web Semantics 7 (3) (2009) 166 – 176.
[22] S. Kara, Özgür Alan, O. Sabuncu, S. Akpınar, N. K. Cicekli, F. N. Alpaslan, An ontology-based retrieval system using semantic indexing, Information Systems 37 (4) (2012) 294 – 305.
[23] B. Fazzinga, T. Lukasiewicz, Semantic search on the web, Semantic Web 1 (1-2) (2010) 89–96.
[24] D. Damljanovic, V. Devedžic, Semantic Web and E-Tourism, in: M. Khosrow-Pour (Ed.), Encyclopedia of Information Science and Technology, Second Edition, IGI Global, 2009, pp. 3426–3432.
[25] H. Knublauch, Ontology-Driven Software Development in the Context of the Semantic Web: An Example Scenario with Protege/OWL, in: D. S. Frankel, E. F. Kendall, D. L. Mcguinness (Eds.), 1st International Workshop on the Model-Driven Semantic Web (MDSW2004), 2004.
[26] J. Ruiz-Martínez, D. Castellanos-Nieves, R. Valencia-García, J. Fernández-Breis, F. García-Sánchez, P. Vivancos-Vicente, J. Castejón-Garrido, J. Camón, R. Martínez-Béjar, Accessing touristic knowledge bases through a natural language interface, in: D. Richards, B.-H. Kang (Eds.), Knowledge Acquisition: Approaches, Algorithms and Applications, Springer Berlin / Heidelberg, 2009, pp. 147–160.
[27] D. Yoo, Hybrid query processing for personalized information retrieval on the semantic web, Knowledge-Based Systems 27 (0) (2012) 211 – 218.
[28] J. Scicluna, N. Steinmetz, Modelling e-tourism services and bundles, in: R. Law, M. Fuchs, F. Ricci (Eds.), Information and Communication Technologies in Tourism 2011, Springer Vienna, 2011, pp. 403–415.
[29] J. Scicluna, N. Steinmetz, M. Zaremba, Service bundling with seekda! dynamic shop, in: U. Gretzel, R. Law, M. Fuchs (Eds.), Information and Communication Technologies in Tourism 2010, Springer Vienna, 2010, pp. 369–380.
[30] D. Damljanović, K. Bontcheva, Towards enhanced usability of natural language interfaces to knowledge bases, in: V. Devedžić, D. Gašević (Eds.), Web 2.0 & Semantic Web, Springer US, 2009, pp. 105–133.
[31] P. Cimiano, M. Minock, Natural language interfaces: What is the problem? – a data-driven quantitative analysis, in: H. Horacek, E. Métais, R. Muñoz, M. Wolska (Eds.), Natural Language Processing and Information Systems, Springer Berlin / Heidelberg, 2010, pp. 192–206.
[32] I. Habernal, M. Konopík, Semantic annotation for the lingvosemantics project, in: Proceedings of the 12th International Conference on Text, Speech and Dialogue, TSD '09, Springer-Verlag, Berlin, Heidelberg, 2009, pp. 299–306.
[33] I. Habernal, Semantic web search using natural language, Ph.D. thesis, University of West Bohemia, Faculty of Applied Sciences (2012).
[34] I. Habernal, M. Konopík, On the way towards standardized semantic corpora for development of semantic analysis systems, in: SEMAPRO 2010: The Fourth International Conference on Advances in Semantic Processing, IARIA, 2010, pp. 96–99.
[35] A. Kiryakov, B. Popov, I. Terziev, D. Manov, D. Ognyanoff, Semantic annotation, indexing, and retrieval, Web Semantics 2 (1) (2004) 49 – 79.
[36] P. Cimiano, S. Handschuh, S. Staab, Towards the self-annotating web, in: Proceedings of the 13th international conference on World Wide Web, WWW '04, ACM, New York, NY, USA, 2004, pp. 462–471.
[37] L. Ding, T. Finin, A. Joshi, R. Pan, R. S. Cost, Y. Peng, P. Reddivari, V. Doshi, J. Sachs, Swoogle: a search and metadata engine for the semantic web, in: Proceedings of the thirteenth ACM international conference on Information and knowledge management, CIKM '04, ACM, New York, NY, USA, 2004, pp. 652–659.
[38] D. Damljanovic, F. Amardeilh, K. Bontcheva, CA manager framework: creating customised workflows for ontology population and semantic annotation., in: Y. Gil, N. F. Noy (Eds.), K-CAP, ACM, 2009, pp. 177–178.
[39] J. Hajič, J. Panevová, E. Hajičová, J. Panevová, P. Sgall, P. Pajas, J. Štěpánek, J. Havelka, M. Mikulová, Prague dependency treebank 2.0, linguistic Data Consortium, Philadelphia (2006).
[40] M. Konkol, M. Konopík, Maximum entropy named entity recognition for czech language, in: Proceedings of the 14th international conference on Text, speech and dialogue, TSD'11, Springer-Verlag, Berlin, Heidelberg, 2011, pp. 203–210.
[41] I. Habernal, M. Konopík, Active tags for semantic analysis, in: Proceedings of the 11th international conference on Text, Speech and Dialogue, TSD '08, Springer-Verlag, Berlin, Heidelberg, 2008, pp. 69–76.
[42] M. Konopík, I. Habernal, Hybrid semantic analysis, in: Proceedings of the 12th International Conference on Text, Speech and Dialogue, TSD '09, Springer-Verlag, Berlin, Heidelberg, 2009, pp. 307–314.
[43] W. E. Winkler, String comparator metrics and enhanced decision rules in the Fellegi-Sunter model of record linkage, in: Proceedings of the Section on Survey Research (American Statistical Association), 1990, pp. 354–359.
[44] B. Liu, Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data, 2nd Edition, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2011.
[45] C. D. Manning, P. Raghavan, H. Schütze, Introduction to Information Retrieval, Cambridge University Press, New York, NY, USA, 2008.
[46] D. Maynard, W. Peters, Y. Li, Evaluating evaluation metrics for ontology-based applications: Infinite reflection., in: LREC, European Language Resources Association, 2008.
[47] K.-C. Tai, The tree-to-tree correction problem, J. ACM 26 (3) (1979) 422–433.
[48] L. Dolamic, J. Savoy, Indexing and stemming approaches for the czech language, Information Processing & Management 45 (6) (2009) 714 – 720.
[49] D. Dahl, M. Bates, M. Brown, W. Fisher, K. Hunicke-Smith, D. Pallett, C. Pao, A. Rudnicky, E. Shriberg, J. Garofolo, J. Fiscus, D. Danielson, E. Bocchieri, B. Buntschuh, B. Schwartz, S. Peters, R. Ingria, R. Weide, Y. Chang, E. Thayer, L. Hirschman, J. Polifroni, B. Lund, G. Kawai, T. Kuhn, L. Norton, ATIS3 Test Data, linguistic Data Consortium, Philadelphia (1995).
[50] Y. He, S. Young, Semantic processing using the hidden vector state model, Computer Speech & Language 19 (1) (2005) 85–106.
[51] E. Iosif, A. Potamianos, A soft-clustering algorithm for automatic induction of semantic classes, in: Interspeech-07, Antwerp, Belgium, 2007, pp. 1609–1612.
[52] M. Jeong, G. G. Lee, Practical use of non-local features for statistical spoken language understanding, Computer Speech and Language 22 (2) (2008) 148–170.
[53] C. Raymond, G. Riccardi, Generative and discriminative algorithms for spoken language understanding, in: Interspeech-07, Antwerp, Belgium, 2007, pp. 1605–1608.

[54] I. Habernal, M. Konopík, O. Rohlík, Question Answering, in: C. Jouis, I. Biskri, J.-G. Ganascia, M. Roux (Eds.), Next Generation Search Engines: Advanced Models for Information Retrieval, IGI Global, 2012, in press.

[55] V. Tablan, D. Damljanovic, K. Bontcheva, A natural language query interface to structured information, in: Proceedings of the 5th European semantic web conference on The semantic web: research and applications, ESWC'08, Springer-Verlag, Berlin, Heidelberg, 2008, pp. 361–375.

[56] J. Ruiz-Martínez, D. Castellanos-Nieves, R. Valencia-García, J. Fernández-Breis, F. García-Sánchez, P. Vivancos-Vicente, J. Castejón-Garrido, J. Camón, R. Martínez-Béjar, Accessing touristic knowledge bases through a natural language interface, in: D. Richards, B.-H. Kang (Eds.), Knowledge Acquisition: Approaches, Algorithms and Applications, Vol. 5465 of Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2009, pp. 147–160.

[57] J. Lim, K.-H. Lee, Constructing composite web services from natural language requests, Web Smantics 8 (1) (2010) 1–13.

[58] N. Stratica, L. Kosseim, B. C. Desai, Using semantic templates for a natural language interface to the CINDI virtual library, Data & Knowledge Engineering 55 (1) (2005) 4 – 19.

[59] M. Gao, J. Liu, N. Zhong, F. Chen, C. Liu, Semantic mapping from natural language questions to OWL queries, Computational Intelligence 27 (2) (2011) 280–314.