A logic programming based approach to the QA@CLEF05 track

Paulo Quaresma and Irene Rodrigues
Departamento de Informátiva, Universidade de Évora, Portugal
{pq,ipr}@di.uevora.pt

Abstract

In this paper the methodology followed to build a question-answering system for the Portuguese language is described. The system has two modules: preliminary analysis of documents (information extraction) and query processing (information retrieval). The proposed approach is based on computational linguistic theories: syntactical analysis (constraint grammars); followed by semantic analysis using the discourse representation theory; and, finally, a semantic/pragmatic interpretation using ontologies and logical inference.

The system was evaluated under the CLEF'05 question and answering track with a collection of documents from two newspapers: Público (Portuguese) and Folha de São Paulo (Brazilian).

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: H.3.1 Content Analysis and Indexing; H.3.3 Information Search and Retrieval; H.3.4 Systems and Software

General Terms

Measurement, Performance, Experimentation, Algorithms

Keywords

Question answering, NLP, Prolog

1 Introduction

This paper describes an ongoing project at the Informatics Department of the University of Évora, Portugal, aiming to develop a question answering system for the Portuguese language.

The system has two modules: preliminary analysis of documents (information extraction) and query processing (information retrieval).

The analysis of the document collection and queries is done using models from computational linguistic theories. This processing includes: syntactical analysis of sentences using the constraint grammar Palavras [3]; semantical analysis using discourse representation theory [5]; and, finally, semantic/pragmatic interpretation using ontologies and logical inference.

Knowledge representation and ontologies are handled through the use of an extension to PROLOG, ISCO[1, 2], which allows to integrate logic programming and external databases. In this way it is possible to solve scalability problems like the need to represent more than 10 millions of discourse entities.

The QA system, in order to satisfy CLEF requirements, should be able to answer queries in natural language, based on information conveyed by a collection of documents. The answer to a specific question is: a set of words and the identification of the document and sentence, which was used as the source of information. For instance, for the following question:

Who is John Lennon's widow?

Our system answers:

Yoko Ono - document: publico/publico95/950807/001 - sentence: 2

At the moment, the system is able to answer questions about:

- Places:

  Where is Régua?

- Dates:

  When was Mr. X arrested?

- Definitions:

  What is ONU?

- Specific:

  How many times was Mr. X accused of being a drug dealer?

  What crimes as Mr. Y committed?

However, the system can also be used to improve other information retrieval tasks:

- Improve the performance of information retrieval systems, helping to obtain the set of relevant documents.

  Given a query, the system can be used to retrieve all documents having a sentence which verifies the query.

  For the question "Who is John Lennon's widow" our system is able to retrieve all documents having a sentence which verifies the following term:

$$\exists X, Y : widow(X, Y), name(X,' John\_Lennon')$$

- Help to structure documents in a semi-automatic way.

  The system is able to extract facts from the documents and to give them a structure (accordingly with an ontology).

The proposed system is an evolution of a previous system evaluated at CLEF 2004 [7]. Some of the existing problems were solved, namely, the need to use a pre-processing information retrieval engine to decrease the complexity of the problem. In this CLEF edition, we were able to solve this major scalability problem via the use of ISCO and its power to connect PROLOG and relational databases.

However, the pre-processing of the collection of documents took more time than we expected and we were not able to answer questions to the Folha de S. Paulo newspaper. As we will point out in the evaluation section this was our major problem and it is the reason why our results didn't improve from CLEF04 to CLEF05.

In the next section the architecture of the system is described. In sections 3 and 4 the syntactical and the semantical modules are described in detail. Section 5 presents the knowledge representation approach. Section 6 describes the semantic-pragmatic interpretation of the documents, based on the previous analysis and on the ontology. Section 7 shows the processing of a query and the generation of the correspondent answer. In section 8 the evaluation results are presented. Finally, in section 9 some conclusions and future work are discussed.

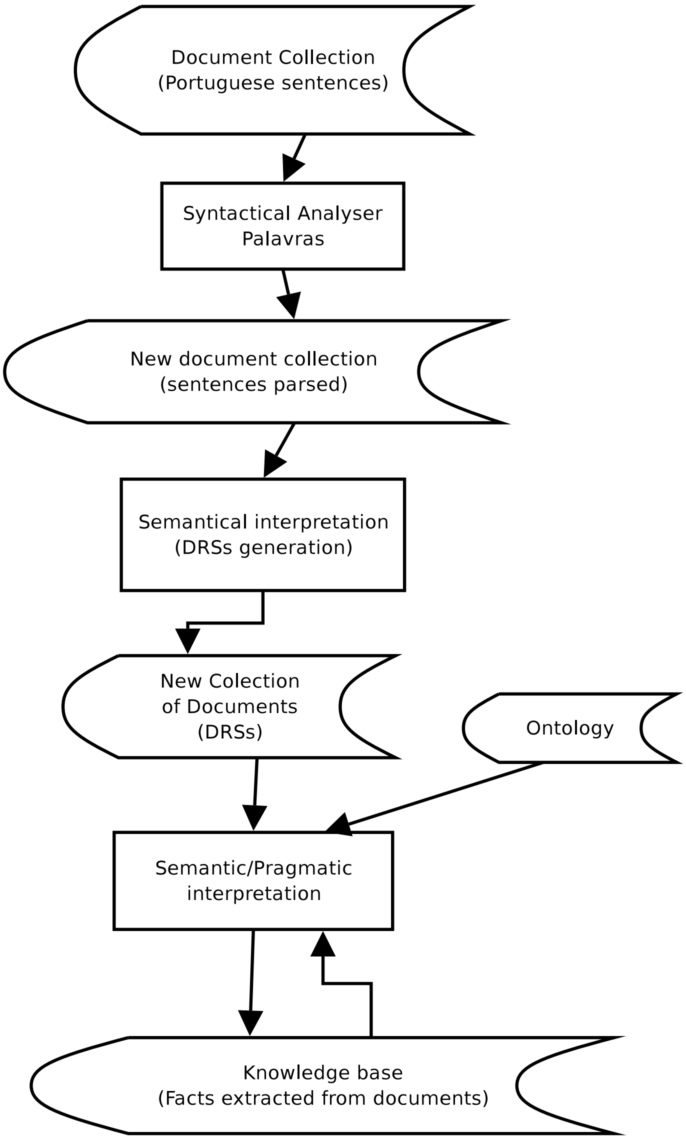

Figure 1: Document Processing

2 Architecture

The QA system has two main modules:

- Information extraction;

  This module extracts information from the documents and it creates a knowledge base.

  The module, see figure 1, is composed by several sub-modules:

  - Syntactical analysis: sentences are processed with the Palavras[3] parser.

    After this phase, a new collection of documents (with the parsing result) is obtained.
  - Semantic analysis: the new collection of sentences in rewritten [5] creating another collection, where each document has a DRS (structure for the discourse representation), a list of discourse referents and a set of conditions.

- Semantic and pragmatic interpretation: in this phase the previous collection of documents is processed, taking into account an ontology and, as a result, a knowledge base is built. This knowledge base contains instances of the ontology.

- Information retrieval

  This module processes the query and it generates the answer: a set of words and the identification of the document and sentence where the answer was found.

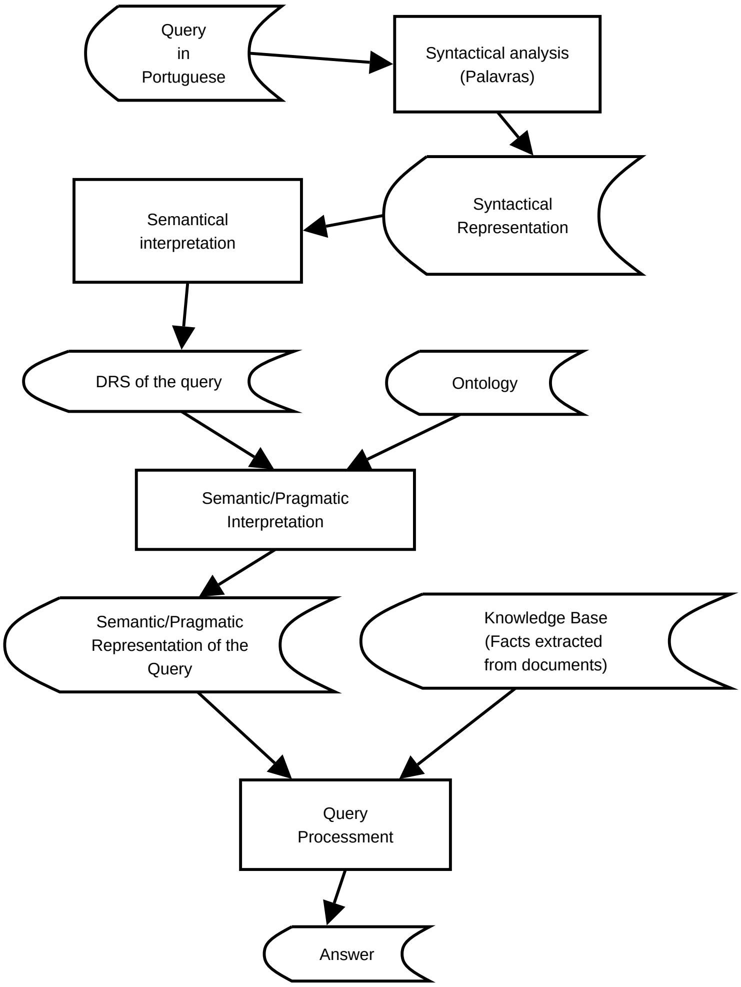

Figure 2: Query Processing

Figure 2 shows the diagram of this module. It is composed by the following phases:

- Syntactical analysis: using the parser Palavras[3].
- Semantic analysis: from the parser output, a discourse structure is built, a DRS[5] with the correspondent referents.
- Semantic/Pragmatic interpretation: in this phase, some conditions are rewritten, taking into account the ontology, and generating a new DRS.
- Query Processing: the final query representation is interpreted in the knowledge base through the unification of the discourse entities of the query with documents discourse entities (see section 7).

In the next sections these sub-modules are described in more detail.

3 Syntactical analysis

Syntactical analysis is obtained through the use of the PALAVRAS parser from Eckhard Bick[3], developed in the context of the VISL[1] project at the *University of Southern Denmark*. This parser gives good morpho-syntactical information and it has a god coverage of the Portuguese language. For instance, in our system the verb lemma (infinitive form) is used as the name of the predicates in the semantic analysis.

As an example, consider the following sentence (3.1):

> Um patologista defendeu que Jimi Hendrix morreu de asfixia após ter ingerido álcool e uma dose excessiva de barbitúricos. (3.1)
>
> ”A pathologist argued that Jimi Hendrix died of asphyxia after drinking alcoholic beverages and an excessive dose of barbiturics”.

The syntactical structure of this sentence is the following:

```
STA:fcl
=SUBJ:np
==>N:art( um  M S <arti>) Um
==H:n( patologista  M S <Hprof>)
  patologista
=P:v-fin( defender  PS 3S IND)
  defendeu
=ACC:fcl
==SUB:conj-s( que ) que
==SUBJ:prop( Jimi_Hendrix  M/F S)
  Jimi_Hendrix
==P:v-fin( morrer  PS 3S IND) morreu
==PIV:pp
===H:prp( de ) de
===P<:np
====H:n( asfixia  F S <sick>) asfixia
====N<:pp
=====H:prp( após ) após
=====P<:icl
======P:vp
=======AUX:v-inf( ter ) ter
=======MV:v-pcp( ingerir ) ingerido
======ACC:n( álcool  M S <cm-liq>)
   álcool
======CO:conj-c( e ) e
======ACC:np
=======>N:art( um  F S <arti>) uma
=======H:n( dose  F S) dose
=======N<:adj( excessivo  F S)
   excessiva
=======N<:pp
========H:prp( de ) de
========P<:n( barbitúrico  M P)
   barbitúricos
```

This structure is transformed by our system in a equivalent Prolog representation, which will be used as the input of the semantic analyser.

[1] Visual Interactive Syntax Learning

```
sta(fcl,
    subj(np,
        n(art('um',
              'M', 'S', <arti> ), 'Um'),
        h(n('patologista',
             'M', 'S', <Hprof> ),
          'patologista')),
    p(v_fin('defender', 'PS', '3S', 'IND'),
       'defendeu'),
    acc(fcl,
        sub(
            conj_s('que'), 'que')),
    subj(prop('Jimi_Hendrix', 'M/F', 'S'),
         'Jimi_Hendrix'),
    p(v_fin('morrer', 'PS', '3S', 'IND'),
      'morreu'),
    piv(pp,
        h(prp('de'),'de'),
        p(np,
          h(n('asfixia','F','S', <sick> ), 'asfixia'),
          n(pp,
            h(prp('após'), 'após'),
            p(icl,
              p(vp,
                aux(v_inf('ter'),'ter'),
                mv(v_pcp('ingerir'), 'ingerido')),
              acc(n(´álcool',
                     'M', 'S', <cm-liq> ), ´álcool'),
              co(conj_c('e'), 'e'),
              acc(np,
                  n(art('um',
                        'F','S',<arti>), 'uma'),
                  h(n('dose','F','S'), 'dose'),
                  n(adj('excessivo',
                        'F','S'), 'excessiva'),
                  n(pp,
                    h(prp('de'),'de'),
                    p(n('barbitúrico',
                         'M','P'), 'barbitúricos' , '.' )
                  ) ) ) ) ) ) ).
```

4 Semantic analysis

The semantic analysis rewrites the syntactical structure in to a discourse representation structure [5], DRS. At present, we only deal with factual sentences, i.e., sentences with existential quantification over the discourse entities. So, our discourse structures are sets of referents, existentially quantified variables, and sets of conditions, first order predicates.

Each syntactical tree, represented in Prolog, is rewritten accordingly with a set of rules and integrated in a DRS.

In order to allow alternative interpretations, the link between prepositional phrases in done using the relation *rel* with 3 arguments, the preposition and two discourse entities. This predicate *rel* allows the semantic/pragmatic interpretation to infer the adequate connection between the referents. For instance, the sentence 'O dono do cão'/'The owner of the dog', is represented by the following DRS:

```
drs(
    entities:[ A:(def, male, sing),
               B:(def, male, sing)],
    conditions:[owner(A),
               dog(B),
               rel(of,A,B)]
)
```

As it can be seen in the next section, this representation allows the semantic/pragmatic interpretation to rewrite the DRS, obtaining the following structure:

```
drs(
    entities:[ A:(def, male, sing),
               B:(def, male, sing)],
    conditions:[belongs(A,B),
               person(A),
               dog(B)]
)
```

In order to show an example of a syntactical tree transformation into a DRS, we show sentence (3.1) rewritten :

```
drs (entities:[ A: (indef, male, sing),
                B: (def, male/fem, sing),
                C: (def, fem, sing),
                D: (def, male, sing),
                E: (indef, fem, sing) ],
     condições:[ pathologist (A),
                 argue(A,B),
                 name(B, 'Jimmy Hendrix'),
                 died(B),
                 rel (of, B, C),
                 asphyxia(C),
                 rel (after, C, D),
                 drinking(D),
                 alcohol(D),
                 dose(D),
                 excessive(D),
                 rel(of, D, E),
                 barbiturics(E)])
```

User queries are also interpreted and rewritten into DRS. For instance, the question:

Como morreu Jimi Hendrix?/How did Jimi Hendrix died? (4.1)

is transformed into the following discourse structure:

```
drs(
    entities:[F:(def, male/fem, sing),
              G: interrog(que), male, sing]
    conditions:[died(F),
               name(F,'Jimmy Hendrix'),
               rel(of,F,G)]
)
```

This representation is obtained because “Como/How” is interpreted as “de que/of what”. In the semantic-pragmatic interpretation and in the query processing phase, the structure (4.1) might unify with sentence (3.1) and we may obtain the following answer: “Jimi Hendrix died of asphyxia”.

5 Ontology and knowledge representation

In order to represent the ontology and the extracted facts, we use an extension to logic programming, ISCO[1, 2], which allows Prolog to access databases. This technology is fundamental to our system because we have a very large database of referents: more than 9 millions only for the Público newspaper.

Databases are defined in ISCO from ontologies. Our system uses two different ontologies:

- one ontology built by us aiming to model common knowledge, such as, geography (mainly places), and dates;

  This kind of knowledge is important to correctly extract facts from the documents and to be able to answer questions about places. The ontology defines places (cities, countries, . . .) and relations between places.

- one ontology generated automatically from the document collection [9, 8];

  This ontology, although being very simple, allows the representation of the domain knowledge.

The ontology can be defined directly in ISCO or in OWL (Ontology Web Language) and transformed in ISCO [8].

The knowledge extraction module identifies facts (instances of an ontology classes) and inserts them as rows in a database table.

For instance, sentence (3.1), with semantic representation in page 7, would generate several tuples in the database. First order logical expressions are *skolemized*, i.e., each variable existentially quantified is replaced by a different identifier:

- `(123,''Jimmy Hendrix'')` is added to table *name*
- `(123)` is added to table *die*
- `(124)` is added to table *asphyxia*
- `rel(de,123,124)` is added to table *rel*

In the document processing phase, our system uses the first sentence interpretation (note that for each sentence there might exist several distinct interpretations). This is caused by temporal and spacial complexity problems but it does not seem to decrease much the performance of the system. Nevertheless, additional measures should be done in order to fully evaluate the impact of this option.

Additionally, we also add information in the database linking referents with the documents and sentences were they appeared. For instance the tuple `(123,'publico/publico95/950605/005',4)` would be added to table *referred_in*.

6 Semantic/Pragmatic Interpretation

Semantic/pragmatic interpretation tries to reinterpret semantical information, taking into account the considered ontology.

This process receives as input a discourse representation structure, DRS, and it interprets it using rules obtained from the knowledge ontology and the information in the database.

In order to obtain a good interpretation, our strategy is to search for the best explanation that supports the sentence logical form. This strategy for pragmatic interpretation was initially proposed by [4].

The knowledge base for the pragmatic interpretation is built from the ontology description in ISCO. The inference in the knowledge base uses abduction and finite domain constraint solvers.

Suppose the following sentence:

> "A. conduz com uma taxa de alcoolemia de 2.15."
>
> "A. drives with an alcoholic rate of 2.15.".

which, by the semantic analysis, is transformed into the following structure: one DRS, four discourse referents, and a set of conditions:

```
drs(
    entities:[A:(def,male,sing),
              B:(indef,fem,sing),
              C:(indef,fem,sing),
              D:(def,male,sing)]
    conditions:[name(A, 'A.'),
               drive(A),
               rel(with,A,B),
               rate(B),
               rel(of,B,C),
               alcohol(C),
               rel(of,C,D),
               number(D,2.15)]
)
```

The semantic/pragmatic interpretation process, using information from the ontology, will rewrite the DRS into the following one:

```
drs(
    entities:[A:(def,male,sing),
              B:(def,male,sing)]
    conditions:[name(A, 'A.'),
               person(A),
               drive(A,_,_,_,B),
               alcohol\_rate(B,2,15)]
)
```

The interpretation of *rel(with,A,B)* as *drive(A,_,_,_,B)* is possible because the ontology has a class *drive*, which relates persons driving in a time interval with a alcoholic rate in blood.

One of the major problems of this phase is to correctly identify the distinct referents in the documents. It is important to use the same skolem constant to identify the same referent but, in any situation, different individuals should have the same identifier (skolem constant).

7 Answer generation

The generation of the answer is done in two steps:

1. identification of the database referent that unifies with the referent of the interrogative pronoun in the question.

2. retrieval of the referent properties and generation of the answer.

In order to illustrate this process, suppose the following question:

> “Quem cometeu um homicídio por conduzir alcoolizado?”
>
> “Who committed an homicide because he/she was driving drunk?”

This question is represent by the following DRS, after syntactical and semantical analysis:

```
drs(
    entities:[A:(who,male/fem,sing),
              B:(indef,male,sing),
              C:(indef,male,sing)],
    conditions:[committed(A,B),
               homicide(B),
               rel(because,A,C),
               drunk(C),
               drive(C)]
)
```

The semantic/pragmatic interpretation of this question is done using the ontology of concepts and it allows to obtain the following DRS:

```
drs(
    entities:[A:(who,male/fem,sing),
              B:(indef,male/fem,sing/plu),
              C:(def,fem,sing),
    conditions:[homicide(A,B),
               person(A),
               person(B),
               drive(A,_,_,_,C),
               alcohol\_rate(C),C>0.5]
)
```

- In order to perform the first step of the answer generation:

  We keep the referent variables of the question and we try to prove the conditions of the DRS in the knowledge base. If the conditions can be satisfied in the knowledge base, the discourse referents are unified with the identifiers (skolem constants) of the individuals.

- The next step is to retrieve the words that constitute the answer:

  In this phase we should retrieve the conditions about the identified referent *A* and choose which ones better characterize the entity. Our first option is to choose a condition with the predicate *name* (name(A,Name)).

  However, it is not always simple to find the adequate answer to a question. See, for instance, the following questions:

  - What crimes committed X?
  - How many habitants has Kalininegrado?
  - What is the nationality of Miss Universe?
  - Who is Flavio Briatore?

  In order to choose the best answer to a question our systems has an algorithm which takes into account the syntactical category of the words that may appear in the answer and it tries to avoid answers with words that appear in the question.

  Questions about places or dates have a special treatment involving the access to a database of places or dates.

Note that several answers may exist for a specific question. In CLEF05 we decided to calculate all possible answers and to choose the most frequent one.

CLEF05 also imposes that the answer is supported by a single sentence in a specific document. Our system is able to obtain answers with conditions in several documents but we constrained the system to obtain only answers with referents introduced in the same sentence (predicate *referred_in* allows to obtain that information –see page 8).

8 Evaluation

The evaluation of our system was performed in the context of CLEF – Cross Language Evaluation Forum – 2005. In this forum a set (200) of questions is elaborated by a jury and given to the system. The system's answers are, then, evaluated by the same jury.

Our system had the following results:

- 25% of correct answers (50 answers).
- 1.5% correct but unsupported answers (3 answers).
- 11% inexact answers – too many (or too few) words (22 answers).
- 62.5% wrong answers (125 answers).

The system had 125 wrong answers, but it is important to point out that 105 of these wrong answers were NIL answers, i.e., situations were the system was not able to find any answer to the questions. So, only in 10% of the situations (20 answers) our system gave a really wrong answer.

The major problem with the remaining 105 no-answers is the fact that, due to time constraints we were not able to process the collection of documents from the Folha de S. Paulo newspaper. At present, we do not know how many of these no-answers would be answered by this collection, but we expect our results to improve significatively.

A preliminary analysis of the other incorrect answers showed that the main cause of problems in our system is related with lack of knowledge: wrong syntactical analysis; lack of synonyms; and, mostly, an incomplete ontology. In fact, most problems are related with incorrect pragmatic analysis due to an incomplete ontology.

However, and taking into account that our answers were computed using only one collection of documents (the Público newspaper), and comparing with the CLEF2004 results, we believe our system produced good and promising results. In fact, it showed to have a quite good precision on the non NIL answers: only 10% of these answers were wrong.

9 Conclusions and Future Work

We propose an architecture for a question answering system for the Portuguese language.

Our system uses natural language processing techniques to create a knowledge base with the information conveyed by documents. Queries are analysed by the same tools and logical inferences over the knowledge base are performed, trying to find an adequated answer. The inference process is performed using a logic programming framework and the Prolog inference engine.

The system main problems are related with errors in the several NLP tools and with the lack of coverage of the ontology.

At present, we are re-evaluating our system using a knowledge base with the two newspaper collection. We expect our results to improve significatively.

As future work, we intend to explore the problem of automatically build ontologies. The improvement of the used NLP tools is another area needing much work. We also intend to handle anaphoric relations in the documents, allowing the reduction of the number of distinct referents.

References

[1] Salvador Abreu. Isco: A practical language for heterogeneous information system construction. In *Proceedings of INAP'01*, Tokyo, Japan, October 2001. INAP.

[2] Salvador Abreu, Paulo Quaresma, Luis Quintano, and Irene Rodrigues. A dialogue manager for accessing databases. In *13th European-Japanese Conference on Information Modelling and Knowledge Bases*, pages 213–224, Kitakyushu, Japan, June 2003. Kyushu Institute of Technology. To be published by IOS Press.

[3] Eckhard Bick. *The Parsing System "Palavras". Automatic Grammatical Analysis of Portuguese in a Constraint Grammar Framework.* Aarhus University Press, 2000.

[4] Jerry Hobbs, Mark Stickel, Douglas Appelt, and Paul Martin. Interpretation as abduction. Technical Report SRI Technical Note 499, 333 Ravenswood Ave., Menlo Park, CA 94025, 1990.

[5] Hans Kamp and Uwe Reyle. *From Discourse to Logic:An Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory.* Dordrecht: D. Reidel, 1993.

[6] M-F Moens. Interrogating legal documents: The future of legal information systems? In *Proceedings of the JURIX 2003 Workshop on Question Answering for Interrogating Legal Documents*, pages 19–30, Utrecht, Netherlands, 2003. Utrecht University.

[7] Paulo Quaresma and Irene Rodrigues. Using dialogues to access semantic knowledge in a web legal IR system. In Marie-Francine Moens, editor, *Procs. of the Workshop on Question Answering for Interrogating Legal Documents of JURIX'03 – The 16th Annual Conference on Legal Knowledge and Information Systems*, Utrecht, Netherlands, December 2003. Utrecht University.

[8] José Saias. Uma metodologia para a construção automática de ontologias e a sua aplicação em sistemas de recuperação de informação – a methodology for the automatic creation of ontologies and its application in information retrieval systems. Master's thesis, University of Évora, Portugal, 2003. In Portuguese.

[9] José Saias and Paulo Quaresma. Using nlp techniques to create legal ontologies in a logic programming based web information retrieval system. In *Workshop on Legal Ontologies and Web based legal information management of the 9th International Conference on Artificial Intelligence and Law*, Edinburgh, Scotland, June 2003.