Towards Ontological Question Answering

Rémi Zajac
Computing Research Laboratory, New Mexico State University
zajac@crl.nmsu.edu

Abstract

This paper presents an ontology-based semantic framework to question answering. Both question and source text are parsed into underspecified semantic expressions where names of semantic atoms and predicates are defined in an interlingual ontology. Answer retrieval is done using subsumption and unification, and queries are expanded incrementally using ontological rules. Ranking of answers is achieved by using graded unification.

1 Introduction

1.1 Overview of the approach

This paper presents a lexical semantic framework to question answering, where an interlingual ontology, e.g. (Mahesh and Nirenburg, 1995; Mahesh, 1996; Raskin & Nirenburg), provides the atoms and predicates used in the semantic expression associated to questions and texts. The semantic dictionaries and the ontology are the sources of ontological rules which are used to expand the query (the semantic expression associated to the natural language question). The use of a formal ontology provides a natural path towards knowledge-base and database question answering as well as domain-driven question-answering. This approach provides a common framework for both cross-language question answering and mixed text/database access.

Both question and source text are parsed into a variant of Minimal Recursion Semantics (Copestake et als. 95; Copestake et als. 99). Underspecified semantic expressions (*USE*s) are associated to sentences or sentence fragments by a robust unification-based parser. When the parser does not produce a complete parse for a sentence, the *USE* of the sentence is the conjunction of *USE*s of parsed sentence fragments. For the purpose of QA, *USE*s are conjoined at the paragraph level.

In the semantic dictionary, words are given (partial) semantic descriptions. In these descriptions, concepts, roles and attributes, are defined in an interlingual ontology (Mahesh and Nirenburg, 1995; Mahesh, 1996; Raskin & Nirenburg). Concepts, roles and attributes participate in their own type inheritance hierarchy. Concepts are given definitions (as *USE*s), and roles and attributes participating in the definition are given default types and default values.

Answer retrieval is done using unification on *USE*s and matching answers are ranked by computing the relevance of the answer to the question. If no answer can be found using unification, the query is expanded incrementally until an answer can be found. Query expansion uses ontological rules expressing synonymic, antonymic and meronymic relations between concepts.

1.2 Related work

Question-answering using "flat logical forms" has been used in the ExtrAns system (Molla et als. 00) for Unix man pages, following (Hobbs 86) for logical forms. In this system, concepts are simply words and only a few domain-dependent rules seem to be added to the system to perform inference during the answer extraction process. The rules seem to be limited to hyperonymic relations (for example: "cp is-a command").

In the Falcon system (Harabagiu et als. 00), both questions and answers are translated into logical forms, also following Hobbs. This system uses uses unification to retrieve potential answers. Semantic relations provided by WordNet are used to derive so-called "key-

word alternations" (inventor/invented/invent, killer/assassin, people/population). Hyperonymic relations extracted from WordNet are used to derive axioms for the abductive prover: *prefer* is *like better* (Hobbs et als. 93).

The Qanda system (Breck et als. 99) combines IR and knowledge-representation techniques in a similar way: both texts containing potential answers and questions are parsed into a logical form, and the question's logical form is matched against the text's logical forms. The logical form atoms are concepts, although the KR framework is not described in the paper (the description of the KR-based approach in this paper is rather short, and no reference for this framework is provided).

A somewhat related approach is described in (Keselj 01). The author uses an HPSG grammar to parse questions and potential answer texts into a semantic representation, and answers are extracted by matching the semantic representation of the question and the potential answers. This system does not seem to use any knowledge-based representation.

Also relevant is the work done in the knowledge-base community for accessing knowledge bases using natural language queries. This line of research has a long history, going back perhaps to the Lunar system (Woods 77). More recent relevant work include question answering using Quasi-Logical Forms (Gamback & Ljung 92) where QLFs queries are translated into SQL queries for a relational database. The system described in (Clark et als. 99) assumes that the knowledge is stored in a knowledge base and structured as an ontology of the domain. In this system, the user builds interactively a KB query, the system matches the query against a list of query types which are themselves associated to answer schemata. The KB inference capabilities are used in building an answer from answer schemata (a conceptual graph), from which a natural language text is generated.

2 Underspecified semantic parsing

2.1 Underspecified semantic expressions

The semantic description of a word in the semantic lexicon is a partial underspecified semantic expression (*USE*). For example, the semantics of the verb *run* is simply specified as **run**(e)[1] and almost all associated relations and properties are inherited from the super-type of **run** which is **ground-contact-motion**. The expanded semantic description includes the conjunction of predicates **run**(e) **agent**(e,a) **instrument**(e,i), etc. In the semantic dictionary, the verb *run* and the nouns *run* and *running* share the same core semantic expression (a **run**(e) event). This captures all morphosyntactic variations used to refer to the same event.

Events such as **run**(e) are 1-place predicates where the argument is an index on the referred event.[2] Instead of having a positional argument notation as in (Copestake et als. 99), participants in a relation are described using relational predicates such as **agent**(e,a), **theme**(e,t), etc. where the first argument is linked to the event, and the second argument is the referent of the agent, theme, etc. This allows to represent in a flexible way expressions where not all participants are realized in the linguistic expression.

Objects are unary predicates where the argument indexes the referent[3]. For example, the word *weapon* is defined as **weapon**(x). Objects do not have associated relational predicates as events do, but they may have attributes that describe properties of the object, such as size, shape, etc. These attributes are encoded using 2-place predicates such as **size**(x,y) where the second argument is an atomic value (number, symbol, string). Nouns derived from verbs include an inverse relation to the event. For example, the noun *investor* is an **investor** concept which participates in an investment event described by the following expression:

investor(i) **agent**(e,i) **invest**(e)

This allows for example to capture derivational lexical relations such as invest/investor/investment.

Properties (realized as lexically adjectives and adverbs) are also unary predicates where

[1] For the sake of simplicity, we ignore in this paper scopal information: the handle and scopal arguments of the predicate. See (Copestake et als. 99) for details.

[2] A better representation would use 2 arguments, one for the process of the event (which can be modified by time, tense, aspect properties and adverbial expressions), and the second argument for the concept, which can be modified by adjectival expressions (case of light verbs for example). See e.g. (Riehemann 97).

[3] Ignoring again scopal information.

the argument indexes the referent of the modified event or object.[4]

Parsing natural language expressions as underspecified semantic expressions of this kind gives a fairly a representation of language expression which abstract away from a large variety of idiosyncratic morpho-syntactic forms. At the morpho-lexical level (dictionary), a concept represent a whole family of derived lexical elements. For example, verbs and their nominalizations are mapped to the same concept (*to run/a run/running*), and semantically related adjectives and adverbs are also mapped to the same concept (*rapid/rapidly*). At the morpho-syntactic level (parsing), the semantic representation is invariant across a large set of syntactic forms: passive/active, topicalized/unmarked, and various forms of subordination and coordination. A good overview of the paraphrasing power of this kind of semantic expressions can be found in (Mel'čuk 88; Mel'čuk & Polguere 91) for example. The exact coverage of the morpho-syntactic variants depends therefore on (1) the semantic dictionary and (2) the coverage of the parser.

2.2 Robust semantic parsing

Underspecified semantic expressions are associated to sentences or sentence fragments by a robust unification-based parser. When the parser does not produce a complete parse for a sentence, but only a sequence of fragments (NPs, VPs, clauses, etc.), a post-processor collects fragment *USE*s for each phrase and associate the conjunction of fragment *USE*s to the sentence. Therefore, the semantics associated to a sentence which has not been fully parsed provides partial information with respect to the semantics that would be built for a complete parse. The partial semantic expression will contain all semantic predicates introduced by lexical elements and phrasals in all fragments. However, it will lack some equalities between variables of predicates when these predicates belong to fragments not connected by the parser. Additionally, scope will also be unspecified in these cases.

For the purpose of QA, *USE*s are conjoined at the paragraph level. At this level, a co-reference resolution engine, if available, might introduce additional (co-reference) constraints.

For example, a full parse of *Leonov, the first man to walk in space* produces a semantic expression such as:

name(y,"Leonov") **first**(y) **man**(y)
walk(w) **agent**(w,y) **location**(w,l)
space(l)

If however the parser does not produces a full parse but only a sequence of clausal and phrasal chunks (e.g., *Leonov / the first man / to walk in space*), the partial semantic expression simply looses some of the co-references:

name(x,"Leonov") **first**(y) **man**(y)
walk(w) **agent**(w,z) **location**(w,l)
space(l)

3 Retrieving answers

Answer retrieval is done first using the subsumption test: a potential answer is any fragment/sentence/paragraph which *USE* is subsumed by the query *USE*.[5] If no answer can be found, unification is used instead and results are ranked by computing the relevance of the answer to the question. If no answer can be found using unification, the query is expanded incrementally until an answer can be found. Query expansion uses ontological rules expressing synonymic, antonymic and meronymic relations between concepts[6].

3.1 Subsumption

The subsumption test checks that all predicates in the question *use* subsume some predicates in the potential answer *use*. Predicate subsumption holds if the question predicate is equal or more general (higher in the type hierarchy) than the answer predicate. In addition, variable bindings present in the question should also hold in the answer. An other definition of subsumption is that the unifier of the question and the answer is equivalent to the answer itself.

[4] Representation of adverbial modifiers of adjectives or adverbs use the handle variable, which is not mentioned in this paper.

[5] We can assume that relevant text passages have been retrieved using an IR engine so that we do not need to parse a whole collection.

[6] Hyponymy and hyperonymy are built in the type hierarchy and are therefore used directly by the subsumption and the unification algorithms.

- **Question**: *What is a meerkat?*
- **Answer**: *Photo, The meerkat, a type of mongoose, thrives in its Coachella Valley haven.*

Using the appropriate question/answer type categorization, the parser will produce the following *USE* for the question, where **z** is the distinguished variable:

y / **meerkat**(x) **typeof**(x,y)

The *USE* for the answer will include:

photo(p) **meerkat**(m) **typeof**(m,c) **mongoose**(c) **thrive**(e) **agent**(e,m) **location**(e,l) **possessor**(m,l) **haven**(l) **name**(l,"Coachella Valley")

In this case, the semantics of the question subsumes the semantics of the sentence and a simple subsumption test will retrieve the text containing the answer. Unification will provide the variable bindings $\{y = m,\ z = c\}$, and the extraction of the sub-expression including the distinguished variable provides the semantic fragment corresponding to the answer:

mongoose(m,c)

3.2 Unification

When no answer can be found using subsumption, unification is used instead. The unified *USE* contains predicates of both question and answer. If two predicates have the same name or have a common subtype, they are unified; the unification of two predicates may introduce additional variable bindings. If the two expressions do not have any unifiable predicates, the unification of the two expressions fails.

- **Question**: *Who was the first Russian astronaut to walk in space?*
- **Answer**: *The broad-shouldered but paunchy Leonov, who in 1965 became the first man to walk in space, signed autographs.*

The *USE* of the question contains:

x / **person**(x) **first**(x) **russian**(x) **astronaut**(x) **walk**(w) **agent**(w,x) **location**(w,s) **space**(s)

and the *USE* of the answer contains:

name(y,"Leonov") **location**(e,d) **date**(d,1965) **first**(y) **man**(y) **walk**(w) **agent**(w,y) **location**(w,l) **space**(l) **sign**(s) **agent**(s,y) **theme**(s,z) **autograph**(z) ...

In this case, the question does not subsumes the answer since it contains predicates which are not in the answer (**russian**, **astronaut**). However, the two expressions share the following predicates: **first**, **walk**, **agent**, **location** and **space**. In addition, the question predicate **person** unifies with **man**, which is a subtype of **person** in the semantic type hierarchy. These unifications also bind the variables appearing in the predicates. Therefore, the unification succeeds and the text is a potential answer.

The unified comon sub-expression is:

man(x) **first**(x) **walk**(w) **agent**(w,x) **location**(w,s) **space**(s)

If the text was only partially analyzed, some co-references would be missing. A subsumption test would fails because of missing predicates in the text, but also because of missing co-references. The use of unification however re-introduces the missing co-references, producing the same unified common sub-expression as above.

Valid answers. In the general case, several answers could be found by unification since the presence of one unifiable predicate in the answer and in the text is enough to make the two expressions unifiable. However, this does not guaranty that the text contains the answer. To make sure that the text actually answers the question, we require that the distinguished variable in the question be bound to a variable in the answer. This is indeed the case in this example through for example the unification of **person**(x) and **man**(y). This unification adds the constraint $x = y$ and which therefore enables to recover the information **name**(x,"Leonov"). If the distinguished variable has no binding in the answer, it is not a valid answer (the text simply

shares some semantic elements with the question without addressing the topic of the question itself).

Note that, unlike subsumption, typed unification may succeed when an element in the query is more specific than some matching element in the text. Type unification provides a built-in relaxation method (the use an hyponym of the question element) for matching questions and answers. As we may prefer specific answers to more generic ones, the unification algorithm can keep track of specificity information by computing a distance between the unifier and the initial query.

3.3 Graded Unification

Even if we restrict potential answers to the one containing a binding for distinguished variables, it is possible to retrieve more than one answer. To select an appropriate answer in the set of all answers, we compute the similarity between questions and answers by recording how many predicates get unified. In the example, we have:

- 6 question predicates unify out of a total of 8, and 2 do not unify;
- 5 answer predicates unify out of a total of 12, and 7 do not unify.

We can therefore attribute a score to the unification such as the average percentage of unified predicates in both question and answer, 58% in this case, as suggested in (Molla 01). However, this symmetric scoring does not reflect the fact that, intuitively, the answer is a much better match than the score would indicate. A score that uses a measure of subsumption instead seems more appropriate. We therefore compute the ratio of unified question predicate (number of unified question predicate divided by the total number of question predicates). This number is a measure of relevant information found in the answer. In the example, the retrieved answer answers 75% of the question. To this ratio, we add the number of co-references in the unified sub-expression which also hold in the answer text divided by the number of co-references in the query.

We also compute the ratio of non-unified answer predicates (number of non-unified answer predicates divided by the total number of answer predicates). This number measures the proportion of irrelevant information in the answer (58% in the example). As for answer relevance, we add the number of missing co-references in the answer divided by the number of co-references in the answer.

We then rank answers first by the proportion of relevant information, and for answers containing the same amount of relevant information, by their proportion of irrelevant information (to favor shorter answers).

3.4 Incremental query reformulation

Using unification and a semantic type hierarchy will not solve problems when synonymous expressions are used: synonymous expressions traverse the ontology sideways and not along hyponymy-hyperonymy links. When unification fails, the query needs to be expanded to include synonymous expressions and other types of semantic variations. This is done by rewriting the semantics of the question using ontological inference rules.

An ontological inference rule is used to rewrite part of the question's *use*: the left-hand side of a rule is unified with the the *use* and the match is *replaced* with the right-hand side. The reformulated expression is then used to attempt to retrieve an answer. If no answer is found, the process is repeated iteratively. If the rules are not recursive, the reformulation process terminates and therefore the answer extraction process terminates too.

The incremental process of query reformulation using a breath-first strategy will find an answer which is semantically closest to the question by minimizing the number of semantic transformations of the question, a process which is somewhat analogous to abduction (Hobbs et als. 93).

Synonymy

The ontology represents a language-independent categorization of the word, but in many cases, synonymies do not carry across languages. Therefore, synonymic relations are expressed as lexical relations within the dictionary of the language, and not in the ontology. We generate ontological synonymy rules directly from lexical synonymy relations. Whenever a synonymy relation holds between 2 words in the dictionary, we generate an ontological rule between the semantic expressions

of these two words. If for example *killer* is a synonym of *assassin* in English, the semantic representation might or might not be the same. If they are the same, no rule need to be generated (case of true synonymy). If, however, the semantic expressions are different, we produce a rewrite rule:

$$\text{Sem}(killer) \rightarrow \text{Sem}(assassin)$$

The same method can be adapted for antonymic relations by introducing a negation predicate in the right-hand side.

Meronymy

Part/whole relations (part-of, subset, member of a collection, etc.) are defined in the ontology and can be used to generate ontological rules for some kinds of metonymies. The concept is replaced by the concept pointed by the part-of relation concept (the inverse is also possible). For example, expressions such as:

$$\text{X}(x)\ \textbf{partof}(x,y)\ \text{Y}(y)$$

where **partof** can be any meronymic relation, produce rules like

$$\text{X}(x) \rightarrow \text{Y}(y)$$
$$\text{Y}(y) \rightarrow \text{X}(x)$$

Type relaxation

If rules for relations such as synonymy and meronymy fail to produce an answer, it is possible to relax the question. A relaxed question is generated by substituting the super-type of a predicate for the predicate itself. In such a case, unification can occur with *siblings* of the original predicate, going sideways in the type hierarchy.

Summary

Textual variants of a natural language expression that can be dealt with this approach include (near-)synonymic lexical variations, and a subset of metonymies (part as whole, far-synonymy). The exact coverage of the textual variants that can be handle will depend on the particular ontology that is the source of the rules. An interesting related problem would be the processing of a larger class of metaphoric expressions that share the same core meaning and that are not covered by ontological rules (Martin 90). This would require to add specific metaphor rules to the system.

4 Conclusion

We outlined an interlingual framework to question answering based on underspecified semantics and interlingual ontology. The use of a logico-semantic framework allows to extract semantically relevant answers and to justify these answers. Associated to an ontology, the inference mechanism is extended to use ontological knowledge. For open-text question answering, any language ontology could be used as the source of ontological rules: Sensus (Knight & Luk 94), Mikrokosmos (Mahesh and Nirenburg, 1995; Mahesh, 1996; Raskin & Nirenburg), Wordnet (Fellbaum et als. 91) although the quality of the end results will obviously depend on the coverage and the quality of the ontology.

It could also be possible to use a domain-specific ontology for question answering on restricted technical domains. A domain-specific ontology could then be used to interface to a knowledge-base, opening the way for mixed text and database question answering systems on open-domain as well as on specific domains.

The use of separate semantic lexicon and interlingual ontology enables the development of cross-language question answering systems, as demonstrated for example by the Spanish-English Crest system at CRL.[7]

The use of semantic expressions allows to abstract away from a large range of *morpho-syntactic paraphrases* of the same semantic expression. In addition, the use of ontological rules allows to abstract away from *lexical, textual and stylistic variants* of the same semantic expression. The exact span of theses variants remains to be determined. This paper presents a work in progress, and no evaluation results are available at the time of writing.

References

Breck, Eric, John Burger, David House, Marc Light, Inderjeet Mani. 1999. "Question answering from large document collections". *AAAI Fall Symposium on Question Answering Systems.*

[7] `http://crl.nmsu.edu/research/projects/crest`

Clark, Peter, John Thompson & Bruce Porter. 1999. "A knowledge-based approach to question-answering". *AAAI-99 Fall Symposium on Question Answering Systems.*

Copestake, A, D. Flickinger, R. Malouf, S. Riehemann, I. Sag. 1995. "Translation using Minimal Recursion Semantics". *TMI-95.*

Copestake, A, D. Flickinger, I. Sag. 1999. "Minimal Recursion Semantics: an Introduction". *ms.* (see `http://www-csli.stanford.edu/ aac`)

Gamback, Bjorn & Stefan Ljung. 1992. "Question answering in the Swedish Core Language Engine. *SICS Research Report R92:14*, Swedish Institute of Computer Science, Stockholm, Sweden.

C. Fellbaum, D. Gross, G. Miller. 1991. "Wordnet': a lexical database organized on psychlinguistic principles". *Lexical Acquisition: Exploiting On-Line Resources to Build a Lexicon.* Lawrence Erlbaum Associates.

Harabagiu, Sanda, Dan Moldovan, Marius Pasca, Rada Mihalcea, Mihai Surdeanu, Razvan Bunescu, Roxana Girju, Vasile Rus & Paul Morarescu. "FALCON: boosting knowledge for answer engines". *TREC-9.*

Hobbs, Jerry. 1986. "Overview of the TACITUS project". *Computational Linguistics* 12(3).

Hobbs, Jerry, Mark Stickel, Doug Appelt, Paul Martin. 1986. "Interpretation as Abduction. *Artificial Intelligence* 63, pp69-142.

Keselj, Vlado. 2001. "Question Answering using Unification-based Grammar". *14th Canadian Conference on Artificial Intelligence, AI'2001*, Ottawa, Canada, June 2001. (To appear).

Knight, K. and S. Luk. 1994. "Building a Large-Scale Knowledge Base for Machine Translation". *Proceedings of the American Association of Artificial Intelligence AAAI-94.* Seattle, WA.

Kavi Mahesh. 1996. *Ontology Development for Machine Translation: Ideology and Methodology.* Memorandum in Computer and Cognitive Science, MCCS-96-292, Computing Research Laboratory, New- Mexico State University, Las Cruces, NM.

Mahesh, K. and S. Nirenburg. 1995a. "A Situated Ontology for Practical NLP". *Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing, International Joint Conference on Artificial Intelligence*, Montreal, Canada.

James Martin. 1990. *A computational model of metaphor interpretation.* Academic Press.

Igor Mel'čuk. 1988. *Dependancy Syntax: Theory and Practice.* State University of New York Press, Albany.

Igor Mel'čuk & Alain Polguere. 1991. "Aspects of the Implementation of the Meaning-Text Model for English Generation". In I. Lancashire (ed.) *Research in Humanities Computing 1.* Clarenton press, London. pp204-215.

Molla, Diego, Gerold Schneider, Rold Schwitter & Michael Hess. 2000. "Answer extraction using a dependency grammar in ExtrAns". *TAL* 41(1), pp127-156.

Molla, Diego. 2001. "Towards incremental semantic annotation". *Proc. of the MMA-2001 Conference*, Tokyo.

Raskin, V. and S. Nirenburg. 1996. "Ten Choices for Lexical Semantics". NMSU CRL Technical Report MCCS-96-304.

Riehemann, Susanne. 1997 "Idiomatic construction in HPSG". *ms.*, Stanford University.

Woods, W.A. 1997. "Lunar rock in natural English: explorations in natural language question answering". In A. Zampolli (ed.) *Linguistic Structure Processing*, North Holland.