Exploiting Paraphrases in a Question Answering System

Fabio Rinaldi, James Dowdall, Kaarel Kaljurand, Michael Hess
Institute of Computational Linguistics,
University of Zürich
Winterthurerstrasse 190
CH-8057 Zürich, Switzerland
{rinaldi,dowdall,kalju,hess}
@ifi.unizh.ch

Diego Mollá
Centre for Language Technology,
Macquarie University,
Sydney NSW 2109, Australia
{diego}@ics.mq.edu.au

Abstract

We present a Question Answering system for technical domains which makes an intelligent use of paraphrases to increase the likelihood of finding the answer to the user's question. The system implements a simple and efficient logic representation of questions and answers that maps paraphrases to the same underlying semantic representation. Further, paraphrases of technical terminology are dealt with by a separate process that detects surface variants.

1 Introduction

The problem of paraphrases conceals a number of different linguistic problems, which in our opinion need to be treated in separate ways. In fact, paraphrases can happen at various levels in language. Using the examples provided in the call for papers for this workshop, we would like to attempt a simple classification, without any pretense of being exhaustive:

1. Lexical synonymy.

   Example: *article*, *paper*, *publication*

2. Morpho-syntactic variants.

   a) *Oswald killed Kennedy. / Kennedy was killed by Oswald.*

   b) *Edison invented the light bulb. / Edison's invention of the light bulb.*

   while (a) is purely syntactical (active vs passive), (b) involves a nominalisation.

3. PP-attachment.

   a plant in Alabama / the Alabama plant

4. Comparatives vs superlatives.

   be better than anybody else / be the best

5. Subordinate clauses vs separate sentences linked by anaphoric pronouns.

   The tree healed its wounds by growing new bark. / The tree healed its wounds. It grew new bark.

6. Inference.

   The stapler costs $10. / The price of the stapler is $10.

   Where is Thimphu located? / Thimphu is capital of what country?

Of course combinations of the different types are possible, e.g. *Oswald killed Kennedy / Kennedy was assassinated by Oswald* is a combination of (1) and (2).

Different types of knowledge and different linguistic resources are needed to deal with each of the above types. While type (1) can be dealt with using a resource such as WordNet (Fellbaum, 1998), type (2) needs effective parsing and mapping of syntactic structures into a common deeper structure, possibly using a repository of nominalisations like NOMLEX (Meyers et al., 1998). More complex approaches are needed for the other types, up to type (6) where generic world knowledge is required, for instance to know that being a capital of a country implies being located in that country. [1] Such world knowledge could be expressed in the form of axioms, like the following:

`(X costs Y) iff (the price of X is Y)`

In this paper we focus on the role of paraphrases in a Question Answering (QA) system targeted at

[1] Note that the reverse is not true, and therefore this is not a perfect paraphrase.

technical manuals. Technical documentation is characterised by vast amounts of domain-specific terminology, which needs to be exploited for providing intelligent access to the information contained in the manuals (Rinaldi et al., 2002b). The approach taken by QA systems is to allow a user to ask a query (formulated in natural language) and have the system search a background collection of documents in order to locate an answer. The field of Question Answering has flourished in recent years[2], in part, due to the QA track of the TREC competitions (Voorhees and Harman, 2001). These competitions evaluate systems over a common data set allowing developers to benchmark performance in relation to other competitors.

It is a common assumption that technical terminology is subject to strict controls and cannot vary within a given editing process. However this assumption proves all too often to be incorrect. Unless editors are making use of a terminology control system that forces them to use a specific version of a term, they will naturally tend to use various paraphrases to refer to the intended domain concept. Besides in a query a user could use an arbitrary paraphrases of the target term, which might happen to be one of those used in the manual itself or might happen to be a novel one.

We describe some potential solutions to this problem, taking our Question Answering system as an example. We show which benefits our approach based on paraphrases bring to the system. So far two different domains have been targeted by the system. An initial application aims at answering questions about the Unix man pages (Mollá et al., 2000a; Mollá et al., 2000b). A more complex application targets the Aircraft Maintenance Manual (AMM) of the Airbus A320 (Rinaldi et al., 2002b). Recently we have started new work, using the Linux HOWTOs as a new target domain.

In dealing with these domains we have identified two major obstacles for a QA system, which we can summarise as follows:

- The Parsing Problem
- The Paraphrase Problem

The **Parsing Problem** consists in the increased difficulty of parsing text in a technical domain due to domain-specific sublanguage. Various types of multi word expressions characterise these domains, in particular referring to specific concepts like tools, parts or procedures. These multi word expressions might include lexical items which are either unknown to a generic lexicon (e.g. *coax cable*) or have a specific meaning unique to this domain. Abbreviations and acronyms are another common source of inconsistencies. In such cases the parser might either fail to identify the compound as a phrase and consequently fail to parse the sentence including such items. Alternatively the parser might attempt to 'guess' their lexical category (in the set of open class categories), leading to an exponential growth of the number of possible syntactic parses. Not only the internal structure of the compound can be multi-way ambiguous, even the boundaries of the compounds might be difficult to detect and the parsers might try odd combinations of the tokens belonging to the compounds with neighbouring tokens.

The **Paraphrase Problem** resides in the imperfect knowledge of users of the systems, who cannot be expected to be completely familiar with the domain terminology. Even experienced users, who know very well the domain, might not remember the exact wording of a compound and use a paraphrase to refer to the underlying domain concept. Besides even in the manual itself, unless the editors have been forced to use some strict terminology control system, various paraphrases of the same compound will appear, and they need to be identified as co-referent. However, it is not enough to identify all paraphrases within the manual, novel paraphrases might be created by the users each time they query the system.

In the rest of this paper we describe first our Question Answering System (in Section 2) and briefly show how we solved the first of the two problems described above. Then, in Section 3 we show in detail how the system is capable of coping with the Paraphrase Problem. Finally in Section 4 we discuss some related work.

2 A Question Answering System for Technical Domains

Over the past few years our research group has developed an Answer Extraction system (ExtrAns) that works by transforming documents and queries into a semantic representation called Minimal Logical Form (MLF) (Mollá et al., 2000a) and derives the answers by logical proof from the documents. A full linguistic (syntactic and semantic) analysis, complete with lexical alternations (synonyms and hyponyms) is performed. While documents are processed in an off-line stage, the query is processed on-line.

Two real world applications have so far been implemented with the same underlying technology. The original ExtrAns system (Mollá et al., 2000b) is used

[2] Although early work in AI already touched upon the topic, e.g. (Woods, 1977).

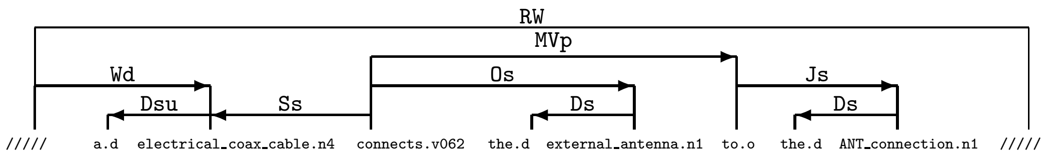

Figure 1: An Example of LG Output

to extract answers to arbitrary user queries over the Unix documentation files ("man pages"). A set of 500+ unedited man pages has been used for this application. An on-line demo of ExtrAns can be found at the project web page.[3]

More recently we tackled a different domain, the Airplane Maintenance Manuals (AMM) of the Airbus A320 (Rinaldi et al., 2002b), which offered the additional challenges of an SGML-based format and a much larger size (120MB).[4] Despite being developed initially for a specific domain, ExtrAns has demonstrated a high level of domain independence.

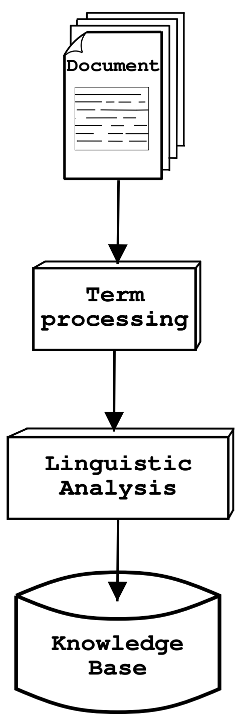

Figure 2: Off-line Processing of Documents

As we work on relatively small volumes of data we can afford to process (in an off-line stage) all the documents in our collection rather than just a few selected paragraphs (see Figure 2). Clearly in some situations (e.g. processing incoming news) such an approach might not be feasible and paragraph indexing techniques would need to be used. Our current approach is particularly targeted to small and medium sized collections.

In an initial phase all multi-word expressions from the domain are collected and structured in an external resource, which we will refer to as the TermBase (Rinaldi et al., 2003; Dowdall et al., 2003). The document sentences (and user queries) are syntactically processed with the Link Grammar (LG) parser (Sleator and Temperley, 1993) which uses a grammar with a wide coverage of English and has a robust treatment of ungrammatical sentences and unknown words. The multi-word terms from the thesaurus are identified and passed to the parser as single tokens. This prevents (futile) analysis of the internal structure of terms (see Figure 1), simplifying parsing by 46%. This solves the first of the problems that we have identified in the introduction ("The Parsing Problem").

In later stages of processing, a corpus-based approach (Brill and Resnik, 1994) is used to deal with ambiguities that cannot be solved with syntactic information only, in particular attachments of prepositional phrases, gerunds and infinitive constructions. ExtrAns adopts an anaphora resolution algorithm (Mollá et al., 2003) that is based on Lappin and Leass' approach (Lappin and Leass, 1994). The original algorithm, which was applied to the syntactic structures generated by McCord's Slot Grammar (McCord et al., 1992), has been ported to the output of Link Grammar. So far the resolution is restricted to sentence-internal pronouns but the same algorithm can be applied to sentence-external pronouns too.

A lexicon of nominalisations based on NOMLEX (Meyers et al., 1998) is used for the most important cases. The main problem here is that the semantic relationship between the base words (mostly, but not exclusively, verbs) and the derived words (mostly, but not exclusively, nouns) is not sufficiently systematic to allow a derivation lexicon to be compiled automatically. Only in relatively rare cases is the relationship as simple as with *to edit <a text>* ↔ *editor of <a text>* / *<text> editor*, as the effort that went into building resources such as NOMLEX also shows.

User queries are processed on-line and converted into MLFs (possibly expanded by synonyms) and proved by refutation over the document knowledge base (see Figure 3). Pointers to the original text attached to the retrieved logical forms allow the system to identify and highlight those words in the retrieved sentence that contribute most to that particular answer. When the user clicks on one of the answers provided, the corresponding document will be displayed with the relevant passages highlighted.

[3] http://www.ifi.unizh.ch/cl/extrans/

[4] Still considerably smaller than the size of the document collections used for TREC

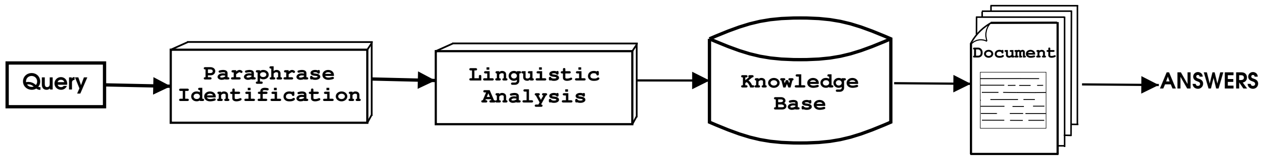

Figure 3: On-line Processing of Queries

The meaning of the documents and of the queries produced by ExtrAns is expressed by means of Minimal Logical Forms (MLFs). The MLFs are designed so that they can be found for *any* sentence (using robust approaches to treat very complex or ungrammatical sentences), and they are optimized for NLP tasks that involve the semantic comparison of sentences, such as Answer Extraction.

The expressivity of the MLFs is minimal in the sense that the main syntactic dependencies between the words are used to express verb-argument relations, and modifier and adjunct relations. However, complex quantification, tense and aspect, temporal relations, plurality, and modality are not expressed. One of the effects of this kind of underspecification is that several natural language queries, although slightly different in meaning, produce the same logical form.

The main feature of the MLFs is the use of reification (the expression of abstract concepts as concrete objects) to achieve flat expressions (Mollá et al., 2000b). The MLFs are expressed as conjunctions of predicates with all the variables existentially bound with wide scope. For example, the MLF of the sentence *"cp will quickly copy the files"* is:

(1)
```
holds(e4), object(cp,o1,[x1]),
object(s_command,o2,[x1]),
evt(s_copy,e4,[x1,x6]),
object(s_file,o3,[x6]),
prop(quickly,p3,[e4]).
```

In other words, there is an entity $x1$ which represents an object of type *cp* and of type *command*, there is an entity $x6$ (a file), there is an entity $e4$, which represents a copying event where the first argument is $x1$ and the second argument is $x6$, there is an entity $p3$ which states that $e4$ is done quickly, and the event $e4$, that is, the copying, holds. The entities $o1$, $o2$, $o3$, $e4$, and $p3$ are the result of reification. The reification of the event, $e4$, has been used to express that the event is done quickly. The other entities are not used in this MLF, but other more complex sentences may need to refer to the reification of properties (adjective-modifying adverbs) or object predicates (non-intersective adjectives such as *the alleged suspect*).

ExtrAns finds the answers to the questions by forming the MLFs of the questions and then running Prolog's default resolution mechanism to find those MLFs that can prove the question. When no direct proof for the user query is found, the system is capable of relaxing the proof criteria in a stepwise manner. First, hyponyms of the query terms will be added as disjunctions in the logical form of the question, thus making it more general but still logically correct. If that fails, the system will attempt approximate matching, in which the sentence (or sentences) with the highest overlap of predicates with the query is retrieved. The (partially) matching sentences are scored and the best fits are returned. In the case that this method finds too many answers because the overlap is too low, the system will attempt keyword matching, in which syntactic criteria are abandoned and only information about word classes is used. This last step corresponds approximately to a traditional passage-retrieval methodology with consideration of the POS tags.

3 Dealing with Paraphrases

The system is capable of dealing with paraphrases at two different levels. On the phrase level, different surface realizations (terms) which refer to the same domain concept will be mapped into a common identifier (synset identifier). On the sentence level, paraphrases which involve a (simple) syntactic transformation will be dealt with by mapping them into the same logical form. In this section we will describe these two approaches and discuss ways to cope with complex types of parapharases.

3.1 Identifying Terminological Paraphrases

During the construction of the MLFs, thesaurus terms are replaced by their synset identifiers. This results in an implicit 'terminological normalization' for the domain. The benefit to the QA process is an assurance that a query and answer need not involve exactly the same surface realization of a term. Utilizing the synsets in the semantic representation means that when the query includes a term, ExtrAns returns sentences that logically answer the query, in-

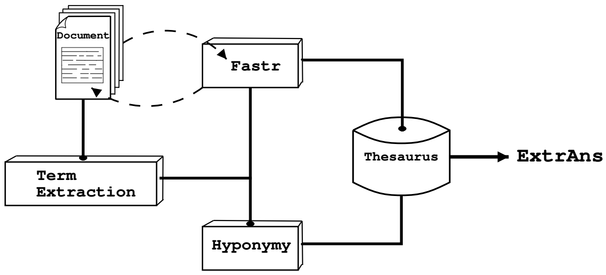

Figure 4: Term Processing

volving any known paraphrase of that term.

For example, the logical form of the query *Where are the stowage compartments installed?* is translated internally into the Horn query (2).

(2)
```
evt(install,A,[B,C]),
object(D,E,[B]),
object(s_stowage_compartment,G,[C])
```

This means that a term (belonging to the same synset as *stowage compartment*) is involved in an install event with an anonymous object. If there is an MLF from the document that can match example (2), then it is selected as a candidate answer and the sentence it originates from is shown to the user.

The process of terminological variation is well investigated (Ibekwe-SanJuan and Dubois, 2002; Daille et al., 1996; Ibekwe-Sanjuan, 1998). The primary focus has been to use linguistically based variation to expand existing term sets through corpus investigation or to produce domain representations. A subset of such variations identifies terms which are strictly synonymous. ExtrAns gathers these morpho-syntactic variations into synsets. The sets are augmented with terms exhibiting three weaker synonymy relations described by Hamon & Nazarenko (2001). These synsets are organized into a hyponymy (isa) hierarchy, a small example of which can be seen in Figure 5. Figure 4 shows a schematic representation of this process.

The first stage is to normalize any terms that contain punctuation by creating a punctuation free version and recording the fact that that the two are strictly synonymous. Further processing is involved in terms containing brackets to determine if the bracketed token is an acronym or simply optional. In the former case an acronym-free term is created and the acronym is stored as a synonym of the remaining tokens which contain it as a regular expression. So *evac* is synonymous with *evacuation* and *ohsc* is synonymous with *overhead stowage compartment*. In cases such as *emergency (hard landings)* the bracketed tokens can not be interpreted as acronyms and so are not removed.

The synonymy relations are identified using the terminology tool Fastr (Jacquemin, 2001). Every token of each term is associated with its part-of-speech, its morphological root, and its synonyms. Phrasal rules represent the manner in which tokens combine to form multi-token terms, and feature-value pairs carry the token specific information. Metarules license the relation between two terms by constraining their phrase structures in conjunction with the morphological and semantic information on the individual tokens.

The metarules can identify simple paraphrases that result from morpho-syntactic variation (*cargo compartment door* $\longrightarrow$ *doors of the cargo compartment*), terms with synonymous heads (*electrical cable* $\longrightarrow$ *electrical line*), terms with synonymous modifiers (*fastener strip* $\longrightarrow$ *attachment strip*) and both (*functional test* $\longrightarrow$ *operational check*). For a description of the frequency and range of types of variation present in the AMM see Rinaldi et al. (2002a).

3.2 Identifying Syntactic Paraphrases

An important effect of using a simplified semantic-based representation such as the Minimal Logical Forms is that various types of syntactic variations are automatically captured by a common representation. This ensures that many potential paraphrases in a user query can map to the same answer into the manual.

For example the question shown in Figure 6 can be answered thanks to the combination of two factors. On the lexical level ExtrAns knows that *APU* is an abbreviation of *Auxiliary Power Unit*, while on the syntactic level the active and passive voices (*supplies* vs *supplied with*) map into the same underlying representation (the same MLF).

Another type of paraphrase which can be detected at this level is the kind that was classified as type (3) in the introduction. For example the question: *Is the sensor connected to the APU ECB?*, can locate the answer *This sensor is connected to the Electronic Control Box (ECB) of the APU.* This has been achieved by introducing meaning postulates that operate at the level of the MLFs (such as "any predicate that affects an object will also affect the *of*-modifiers of that object").

3.3 Weaker Types of Paraphrases

When the thesaurus definition of terminological synonymy fails to locate an answer from the document collection, ExtrAns explores weaker types of paraphrases, where the equivalence between the two terms might not be complete.

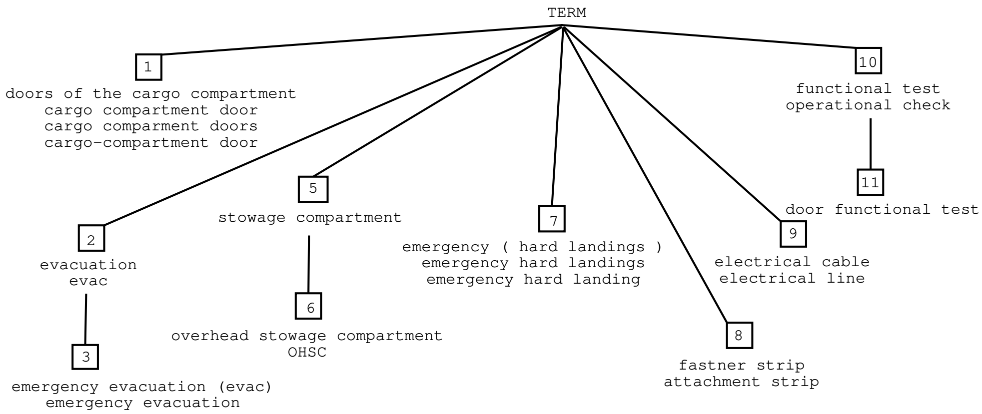

Figure 5: A Sample of the TermBase

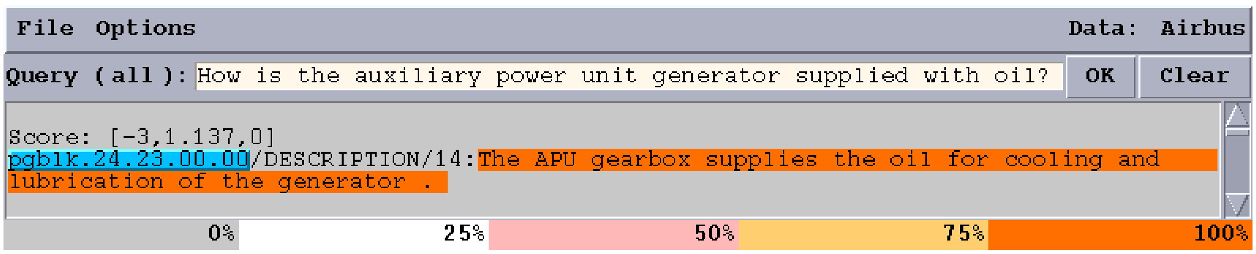

Figure 6: Active vs Passive Voice

First, ExtrAns makes use of the hyponymy relations, which can be considered as sort of unidirectional paraphrases. Instead of looking for synset members, the query is reformulated to included hyponyms and hyperonyms of the terms:

(3)
```
(object(s_stowage_compartment,A,[B]);
object(s_overhead_stowage_compartment,A,[B])),
evt(install,C,[D,B]),
object(E,F,[D|G])
```

Now the alternative objects are in a logical `OR` relation. This query finds the answer in Figure 7 (where *stowage compartment* is a hyperonym of *overhead stowage compartment*).

We have implemented a very simple ad-hoc algorithm to determine lexical hyponymy between terms. Term `A` is a hyponym of term `B` if (*i*) `A` has more tokens than `B`, (*ii*) all the tokens of `B` are present in `A`, and (*iii*) both terms have the same head. There are three provisions. First, ignore terms with dashes and brackets as *cargo compartment* is not a hyponym of *cargo - compartment* and this relation (synonymy) is already known from the normalisation process. Second, compare lemmatised versions of the terms to capture that *stowage compartment* is a hyperonym of *overhead stowage compartments*. Finally, the head of a term is the rightmost non-symbol token (i.e. a word) which can be determined from the part-of-speech tags. This hyponymy relation is comparable to the insertion variations defined by Daille et al. (1996).

The expressivity of the MLF can further be expanded through the use of meaning postulates of the type: "If x **is installed** in y, then x **is** in y". This ensures that the query *Where are the equipment and furnishings?* extracts the answer *The equipment and furnishings are installed in the cockpit.*

4 Related Work

The importance of detecting paraphrasing in Question Answering has been shown dramatically in TREC9 by the Falcon system (Harabagiu et al., 2001), which made use of an ad-hoc module capable of caching answers and detecting question similarity. As in that particular evaluation the organisers deliberately used a set of paraphrases of the same questions, such approach certainly helped in boosting the performance of the system. In an environment where the same question (in different formulations) is likely to be repeated a number of times, a module capable of detecting paraphrases can significantly improve the performance of a Question An-

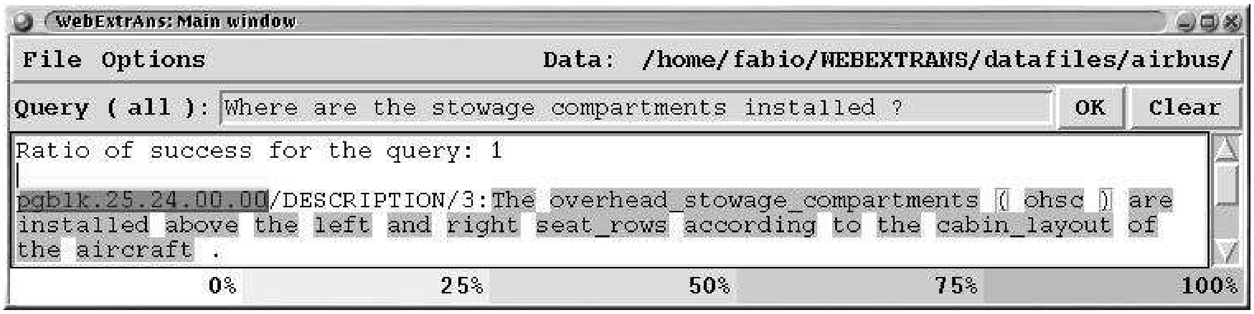

Figure 7: Overhead_stowage_compartment is a Hyponym of Stowage_compartment

swering system.

Another example of application of paraphrases for Question Answering is given in (Murata and Isahara, 2001), which further argues for the importance of paraphrases for other applications such Summarisation, error correction and speech generation.

Our approach for the acquisition of terminological paraphrases might have some points in common with the approach described in (Terada and Tokunaga, 2001). The motivation that they bring forward for the necessity of identifying abbreviations is related to the problem that we have called "the Parsing Problem".

A very different approach to paraphrases is taken in (Takahashi et al., 2001) where they formulate the problem as a special case of Machine Translation, where the source and target language are the same but special rules, based on different parameters, license different types of surface realizations.

Hamon & Nazarenko (2001) explore the terminological needs of consulting systems. This type of IR guides the user in query/keyword expansion or proposes various levels of access into the document base on the original query. A method of generating three types of synonymy relations is investigated using general language and domain specific dictionaries.

5 Conclusion

Automatic recognition of paraphrases is an effective technique to ease the information access burden in a technical domain. We have presented some techniques that we have adopted in a Question Answering system for dealing with paraphrases. These techniques range from the detection of lexical paraphrases and terminology variants, to the use of a simplified logical form that provides the same representation for morpho-syntactic paraphrases, and the use of meaning postulates for paraphrases that require inferences.

References

Eric Brill and Philip Resnik. 1994. A rule-based approach to prepositional phrase attachment disambiguation. In *Proc. COLING '94*, volume 2, pages 998–1004, Kyoto, Japan.

Beatrice Daille, Benot Habert, Christian Jacquemin, and Jean Royauté. 1996. Empirical observation of term variations and principles for their description. *Terminology*, 3(2):197–258.

James Dowdall, Fabio Rinaldi, Fidelia Ibekwe-SanJuan, and Eric SanJuan. 2003. Complex structuring of term variants for Question Answering. In *Proc. ACL-2003 Workshop on Multiword Expressions*, Sapporo, Japan.

Christiane Fellbaum 1998. *WordNet: an electronic lexical database*. MIT Press, Cambridge, MA.

Thierry Hamon and Adeline Nazarenko. 2001. Detection of synonymy links between terms: Experiment and results. In Didier Bourigault, Christian Jacquemin, and Marie-Claude L'Homme, editors, *Recent Advances in Computational Terminology*, pages 185–208. John Benjamins Publishing Company.

Sanda Harabagiu, Dan Moldovan, Marius Paşca, Rada Mihalcea, Mihai Surdeanu, Razvan Bunescu, Roxana Gîrju, Vasile Rus, and Paul Morarescu. 2001. FALCON: Boosting knowledge for answer engines. In Voorhees and Harman (Voorhees and Harman, 2001).

Fidelia Ibekwe-SanJuan and Cyrille Dubois. 2002. Can Syntactic Variations Highlight Semantic Links Between Domain Topics? In *Proceedings of the 6th International Conference on Terminology and Knowledge Engineering (TKE02)*, pages 57–64, Nancy, August.

Fidelia Ibekwe-Sanjuan. 1998. Terminological Variation, a Means of Identifying Research Topics from Texts. In *Proceedings of COLING-ACL*, pages 571–577, Quebec,Canada, August.

Christian Jacquemin. 2001. *Spotting and Discovering Terms through Natural Language Processing*. MIT Press.

Shalom Lappin and Herbert J. Leass. 1994. An algorithm for pronominal anaphora resolution. *Computational Linguistics*, 20(4):535–561.

Michael McCord, Arendse Bernth, Shalom Lappin, and Wlodek Zadrozny. 1992. Natural language processing within a slot grammar framework. *International Journal on Artificial Intelligence Tools*, 1(2):229–277.

Adam Meyers, Catherine Macleod, Roman Yangarber, Ralph Grishman, Leslie Barrett, and Ruth Reeves. 1998. Using NOMLEX to produce nominalization patterns for information extraction. In *Proceedings: the Computational Treatment of Nominals, Montreal, Canada, (Coling-ACL98 workshop)*, August.

Diego Mollá, Gerold Schneider, Rolf Schwitter, and Michael Hess. 2000a. Answer Extraction using a Dependency Grammar in ExtrAns. *Traitement Automatique de Langues (T.A.L.), Special Issue on Dependency Grammar*, 41(1):127–156.

Diego Mollá, Rolf Schwitter, Michael Hess, and Rachel Fournier. 2000b. Extrans, an answer extraction system. *T.A.L. special issue on Information Retrieval oriented Natural Language Processing*.

Diego Mollá, Rolf Schwitter, Fabio Rinaldi, James Dowdall, and Michael Hess. 2003. Anaphora resolution in ExtrAns. In *Proceedings of the International Symposium on Reference Resolution and Its Applications to Question Answering and Summarization*, 23–25 June, Venice, Italy.

Masaki Murata and Hitoshi Isahara. 2001. Universal model for paraphrasing - using transformation based on a defined criteria. In *Proceedings of the NLPRS2001 Workshop on Automatic Paraphrasing: Theories and Applications*.

Fabio Rinaldi, James Dowdall, Michael Hess, Kaarel Kaljurand, Mare Koit, Kadri Vider, and Neeme Kahusk. 2002a. Terminology as Knowledge in Answer Extraction. In *Proceedings of the 6th International Conference on Terminology and Knowledge Engineering (TKE02)*, pages 107–113, Nancy, 28–30 August.

Fabio Rinaldi, James Dowdall, Michael Hess, Diego Mollá, and Rolf Schwitter. 2002b. Towards Answer Extraction: an application to Technical Domains. In *ECAI2002, European Conference on Artificial Intelligence, Lyon*, 21–26 July.

Fabio Rinaldi, James Dowdall, Michael Hess, Kaarel Kaljurand, and Magnus Karlsson. 2003. The Role of Technical Terminology in Question Answering. In *Proceedings of TIA-2003, Terminologie et Intelligence Artificielle*, Strasbourg, April.

Daniel D. Sleator and Davy Temperley. 1993. Parsing English with a link grammar. In *Proc. Third International Workshop on Parsing Technologies*, pages 277–292.

Tetsuro Takahashi, Tomoya Iwakura, Ryu Iida, and Kentaro Inui. 2001. Kura: A revision-based lexico-structural paraphrasing engine. In *Proceedings of the NLPRS2001 Workshop on Automatic Paraphrasing: Theories and Applications*.

Akira Terada and Takenobu Tokunaga. 2001. Automatic disabbreviation by using context information. In *Proceedings of the NLPRS2001 Workshop on Automatic Paraphrasing: Theories and Applications*.

Ellen M. Voorhees and Donna Harman, editors. 2001. *Proceedings of the Ninth Text REtrieval Conference (TREC-9), Gaithersburg, Maryland, November 13-16, 2000*.

W.A. Woods. 1977. Lunar rocks in natural English: Explorations in Natural Language Question Answering. In A. Zampolli, editor, *Linguistic Structures Processing*, volume 5 of *Fundamental Studies in Computer Science*, pages 521–569. North Holland.