ECAI 2008
M. Ghallab et al. (Eds.)
IOS Press, 2008

doi:10.3233/978-1-58603-891-5-313

Semantic Decomposition for Question Answering

Sven Hartrumpf[1]

Abstract. In this paper, we develop and evaluate methods for decomposing complex questions for a question answering system to less complex questions. This aims at increasing the number of correct answers, especially in (deep) semantic question answering systems. For example, an event that occurs as a temporal restriction of a question can be queried for its date and the resulting answer can be substituted in the original question leading to a simpler, revised question. We present six decomposition classes, which are employed for annotating the 996 different German QA@CLEF questions from 2004 till 2008 and trigger different decomposition methods. Most methods work on the level of semantic representations, thereby avoiding natural language generation, a second parsing step, and possible errors in these two steps. The decomposition classes are not equally distributed, but three of them occur frequently in the questions. In the evaluation, the precision and recall for automatically classifying questions with respect to the decomposition classes are investigated. Then, the impact on a deep question answering system is determined. On the QA@CLEF questions, which by construction prefer questions that can be answered from single sentences, the performance gain in number of correct answers is not large, but significant. This encourages us to develop and test further decomposition classes and methods as future work.

1 INTRODUCTION

Real-world questions aimed at document collections that are nontrivial in size and content are often complex questions. These go beyond what most questions in question answering (QA) tracks in evaluation campaigns like CLEF or TREC ask for. (Over the years however, there is a trend to slightly increase the difficulty of questions.) In this paper, we investigate the case of **decomposable questions**, i.e. a question that can be made solvable (or more easily solvable or more probably solvable) by decomposing it into several questions where later questions are built on top of the answers to earlier questions. We will concentrate on the decomposition into sequences of two questions.

Several decomposition methods can be developed based on the semantic representation of questions. A prerequisite of this approach is a parser that produces adequate semantic representations for questions. Since the decomposition works on the semantic level, the most natural choice of a QA system for testing and applying decomposition is a deep (or semantic) QA system that works on the same (or similar) semantic representations of questions and documents.

Most answer candidates produced by **question decomposition** cannot be delivered by shallow QA systems because the subquestion and the revised question derived from the original question are typically answered in different passages of a document or—more likely—in different documents.

2 DECOMPOSITION METHODS

One can define several **decomposition classes**. Each class is described in a separate subsection below and can be linked to a decomposition method that tries to exploit the semantics of the class for improving QA results.

2.1 Temporal decomposition

A class that has been investigated before (see Section 3) is **temporal decomposition** where a situation that is used as a temporal specification can be replaced by its date of occurrence. Figure 1, Figure 2, and Figure 3 illustrate the effect of this decomposition on the level of semantic representations for the **original question**, the **subquestion** (subordinate question) and its answer(s), and the **revised question**, respectively. The original question is a simplified version of CLEF_07_090[2]. The simplifications are as follows: a coreference between questions is resolved and a support verb construction is replaced by a nearly synonymous verb in order to concentrate on the decomposition effect itself.[3]

As the parser applied in the QA system used for the evaluation (see Section 4) produces semantic networks, the semantic representations in these figures are semantic networks. The semantic networks follow the MultiNet (multilayered extended semantic network, [6]) formalism. Each arc is labeled by a relation from a predefined inventory of around 140 relations. The relations occurring in this paper are briefly described in Table 1. Each node represents a concept like *enden* (*'to end'*; more specifically *enden.1.1*, but the numerical extension of concept names is omitted for brevity) and belongs to an ontological sort (written as a subscript of the concept name). The question focus (or top-level node for non-questions) is marked by a question mark followed by the sentence type. Nodes are characterized by layer features, which describe different intensional and extensional aspects like facticity and cardinality or cardinal value. Note that for improving the layout, instantiating relations (SUB, SUBR, SUBS) and some other relations are folded below the concept node, where the relation starts.

Other examples of temporal decomposition are questions asking for the age of a person at an event, e.g. *Wie alt war Konrad Adenauer, als er starb?* (*'How old was Konrad Adenauer when he died?'*).[4] If the coreference between the personal pronoun and the named entity

[1] Intelligent Information and Communication Systems (IICS), University of Hagen (FernUniversität in Hagen), 58084 Hagen, Germany, email: sven.hartrumpf@fernuni-hagen.de

[2] Question 90 from the German test set of QA@CLEF 2007 [2].

[3] The QA system used for evaluation can handle the simplified and the original question equally well because the system employs a coreference resolver and a normalization module for support verb constructions.

[4] The original sentences are shown in italics; the translated English ones are single-quoted in addition.

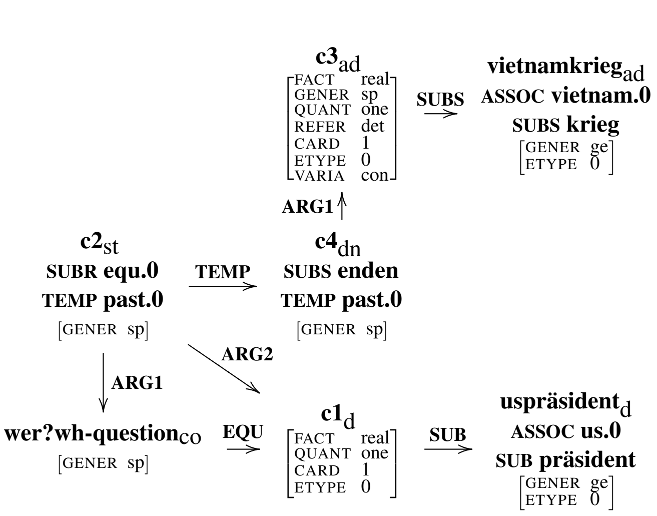

Figure 1. Example of a temporal decomposition: semantic network of the original question *'Who was US president when the Vietnam war ended?'* (*Wer war US-Präsident, als der Vietnamkrieg endete?*)

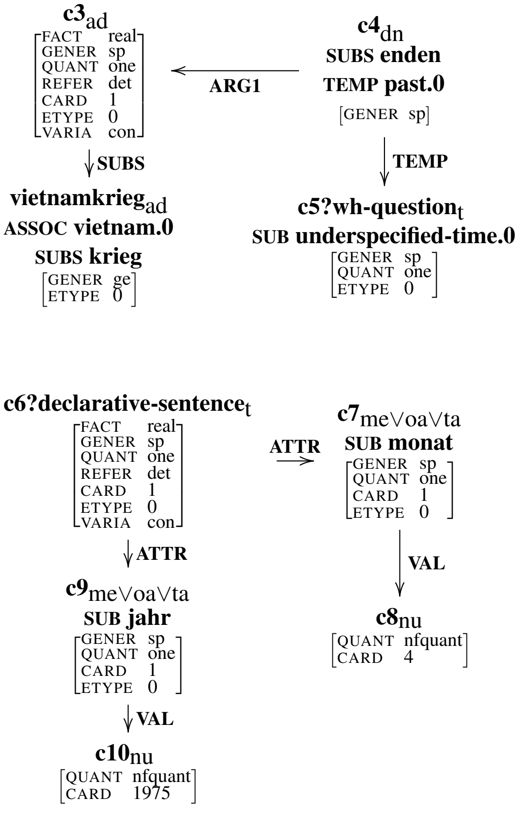

Figure 2. Example of a temporal decomposition: semantic network for the subquestion *'When did the Vietnam war end?'* (*Wann endete der Vietnamkrieg?)* and the subanswer *April 1975*

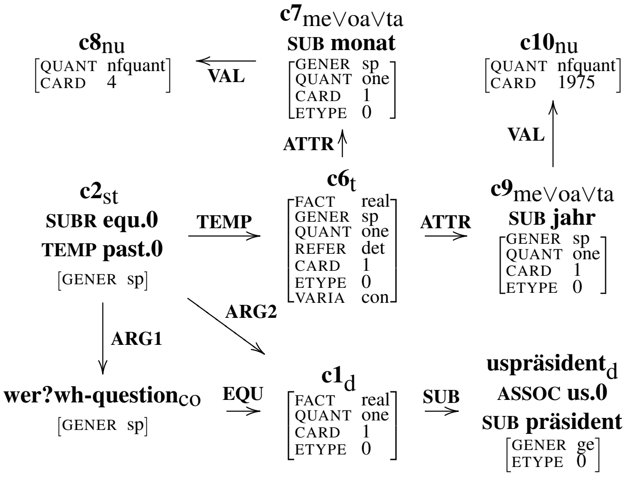

Figure 3. Example of a temporal decomposition: semantic network for the revised question *'Who was US president in April 1975?'* (*Wer war US-Präsident im April 1975?*)

is correctly resolved, a subquestion could be *Wann starb Konrad Adenauer?* (*'When did Konrad Adenauer die?'*) and the revised question, given the correct **subanswer** *1967*, is *Wie alt war Konrad Adenauer 1967?* (*'How old was Konrad Adenauer in 1967?'*).

Although subquestions, subanswers, and revised questions are shown in natural language (NL) in this paper, the handling in the implementation is different. There, most decompositions are working on the level of semantic representations. One advantage of a deep approach (besides the accurate spotting of places and classes of decomposition) is that NL generation is not needed for neither the subquestion nor the revised questions; instead one can directly work on semantic representations derived from the original question. To yield the revised question, the semantic network for the subanswer is substituted into the semantic network of the original question. As a consequence the subquestion and the revised question need not be parsed because their semantic representations are already available. This saves time and avoids noise from several sources of errors. As some of the decomposition examples indicate, an answer could sometimes be found for the original question without decomposition, too. Therefore, decompositions are only seen as alternatives to the original question.

Table 1. Short description of relations used in this paper

Relation	Description
ARG1, ARG2	Specification of arguments (metalevel)
ASSOC	Relation of association
ATTR	Specification of an attribute
EQU	Equality/Equivalence relation
SUB	Relation of conceptual subordination (for objects)
SUBR	Relation of conceptual subordination (metalevel)
SUBS	Relation of conceptual subordination (for situations)
TEMP	Relation for the temporal embedding of a situation
VAL	Relation between an attribute and its value

2.2 Local decomposition

Exploiting the popular analogy between space and time, there is also a class for **local decomposition**, where a local (or spatial) restriction can be replaced by the answer(s) to a subquestion about the exact location(s) fulfilling the restriction. For example in a question like *'Which parties reign in countries that are in Northern Europe?'*, the subquestion would be *'Which countries are located in Northern Europe?'* and a revised question (using a list-valued answer) could be *'Which parties reign in Sweden, Norway, and Finland?'*. Similarly, questions aiming at the local origin or the local direction can be decomposed. Often such questions cannot be expressed in a straightforward way; for example in German, they have to include a so-called correlate like *dorther* und *dorthin* in *Was kommt dorther, wo ewiges Eis existiert?* (*'What comes from where perpetual ice exists?'*) and *Wer reist dorthin, wo es nie friert?* (*'Who travels to places where it never freezes?'*), respectively.

2.3 Coordinated situation decomposition

A question can contain several propositions also in the form of a conjunction of situations (**coordinated situation decomposition**), often involving ellipsis, e.g. *'Which football players played for Borussia Mönchengladbach and Real Madrid?'* (*Welche Fußballspieler spielten für Borussia Mönchengladbach und Real Madrid?*). The first three decomposition classes can be seen as subclasses of the class **multisituation decomposition** because they typically involve two situations that are linked by a relation in the semantic network for the question.

2.4 Meronymy decomposition

Meronymy decomposition tries to generate and exploit geographical knowledge on the fly (here: meronymy knowledge for geographical objects). This decomposition class has already been exploited for GIR (geographic information retrieval), with some positive effects (see [7] for a description of *query decomposition*). For example, a question like *When did a hurricane hit Northern Germany?* could lead via subquestions like *Which regions are in Northern Germany?* or *Which cities are in Northern Germany?* to more specific revised questions like *When did a hurricane hit Niedersachsen?* or *When did a hurricane hit Hamburg?*, respectively. In contrast to meronymy decomposition, local decomposition is restricted to cases where the location of a situation is described by a second situation (like the one expressed by the relative clause *'that are in Northern Europe'*) and not by a named entity.

2.5 Description decomposition

Descriptions in the question focus like *Which football players* in the example for the class *coordinated situations* can be used to first query for such objects with a subquestion. For a question like *Which Italian cities have an Olympic stadium?*, the subquestion derived by this decomposition class (**description decomposition**) is: *Name Italian cities.* The original question and its subquestion typically lead to many different revised questions, each formed by plugging in a subanswer for the subquestion above: *'Does Rome have an Olympic stadium?'*, *'Does Pisa have an Olympic stadium?'*, etc.

2.6 Operational decomposition

Operational questions form a separate decomposition class (**operational decomposition**), e.g. *How many countries belonged to the EU in 1994?* Some of such questions can be answered directly because the information is explicit in a document from the document collection of the QA system, e.g. for the question above, a phrase like *The 12 EU member states ...* might be found in a document. But for more specific questions with fewer relevant documents, it becomes more likely that one really has to calculate an operation on the different answers of a subquestion. In the above case, the subquestion would be *Name countries that belong to the EU.* Some heuristics are important for this class to avoid subquestions that are unlikely to lead to the correct answer to the original question. For example, if the question indicates that there are more than several dozen answers to the potential subquestion. This can be assumed reliably if the question focus contains units like *thousand* or *million*, e.g. *Wie viele Millionen Menschen vereint der Europäische Wirtschaftsraum?* (CLEF_05_084) (*'How many million people does the European Economic Area unite?'*). Operations other than cardinality are maximum, minimum, mean average, etc.

2.7 Other decomposition classes and corpus statistics

One could define something like multi-property decomposition, where a concept is modified by two or more properties. But this can lead to incorrect answers. For example, consider the question *Name a German communist politician.* It can easily happen that the QA system finds a person named *Müller* who is a politician in Germany but not a communist and another politician named *Müller* who is a communist but in Switzerland. Therefore, splitting or decomposing such questions seems too dangerous without additional measures for preserving precision of the QA system. Note that a solution for intertextual named entity identification and tracking will provide a clean solution for such questions, making multi-property decomposition redundant because all properties of a given named entity will (ideally) be represented at one unique concept node.

Table 2 contains the frequencies of the above decomposition classes for the QA@CLEF test sets from 2004 till 2008.[5] A question can belong to zero or more decomposition classes, e.g. *How many aristocrats were archbishops in Italy before the Western Schism?* should be annotated with four decomposition classes: *operational* (*How many*), *description* (*aristocrats*), *meronymy* (*in Italy*), *temporal* (*before the Western Schism*). Note that the last percentage in Table 2 is smaller than the sum of its column because some questions belong to two classes. Operational decomposition in the QA@CLEF test sets involves only cardinality. A class is annotated if—given the general knowledge of the document collection—it is likely that the decomposition leads to an answer. For example, counting the 15 federal states of Germany given the German news corpora is likely to succeed by *operational decomposition*, whereas correctly counting the UN member states is unlikely. A perfect annotation, i.e. one where all the documents are checked whether a decomposition can lead to a correct answer, is far too expensive and would in the end need to assume some concrete QA system or at least some class of QA approaches to classify a question. A similar problem exists to decide whether a decomposition is needed or not in the sense that an answer cannot be found without exploiting the decomposition class at hand.

[5] The test set from 2003 was omitted because most decomposable questions had no answer in the German document collection (so-called NIL questions).

Again this would require expensive manual annotation and would need to assume some knowledge of the relevant QA system. Therefore, the annotation is not a perfect one, but an annotation that is aimed to measure the potential of decomposition.[6] In a sense, this annotation shows an upper bound for the effect of decomposition.

Table 2. Frequency of decomposition classes in QA@CLEF 2004–2008

Decomposition class	Absolute frequency	Affected questions (%)
description	131	13.1
operational	24	2.4
multisituation		
temporal	16	1.6
local, other	7	0.7
other	6	0.6
decomposition (total)	184	15.9

The fact that around 16% of the QA@CLEF questions could profit from decomposition is significant but might seem not overwhelming. But as indicated above, most QA@CLEF questions are by design of the test set development process oriented towards the surface structure of a single document sentence. Many real-world questions show more diverse and difficult patterns.

A good complementary impression will be provided by the evaluation in Section 4. There, a QA system will be run twice: without decomposition and with decomposition, making the impact of decomposition directly measurable.

A question can often be decomposed in several ways when it contains several propositions. For example, the question *Which planet orbits the sun once in every 12 years?* can be decomposed into: a) *What orbits the sun once in every 12 years?* b) *Is ⟨subanswer⟩ a planet?* Here, the outer predicate comes first, and the inner predicate second. But also the opposite order is possible: a) *Name planets!* b) *Does ⟨subanswer⟩ orbit the sun once in every 12 years?* Now, the inner predicate is contained in the subquestion, the outer predicate in the revised question. Note that the revised question is always just a decision question (yes/no question). The two different decompositions can be drawn as a **decomposition rectangle** which connects the original question and its answer by two different two-edge paths forming a rectangle.

The decomposition rectangle is in general applicable to all classes, therefore also to temporal decomposition. But the two paths through the rectangle are not equally likely to succeed. For example, consider the example given by [8]: *Who was the German Chancellor when the Berlin Wall was opened?* The preferred decomposition as it is realized in our QA system is the subquestion *When was the Berlin Wall opened?* and the revised question *Who was the German Chancellor in ⟨subanswer⟩?* The alternative decomposition would be *Who was German Chancellor?* and as revised question *Was ⟨subanswer⟩ German Chancellor when the Berlin Wall was opened?* The reason behind the different chances of success is that human beings use temporal specifications like years much more frequently than events described in separate clauses like the temporal *when*-clause above. This is at least true for texts about history and politics.

With an ideal knowledge base, both decompositions will lead to the same answer(s), but probably with different run times. However, for a realistic knowledge base like one derived from a news article corpus and a QA system composed of NLP modules which still make some errors, the set of answers obtained by the two decompositions can differ. One refinement of the current system could be to try both subquestions first and then to decide which path to follow. This decision could prefer the subquestion which returns fewer answers in order to reduce the run time for the revised question(s).

The decomposition of questions could also be handled by a machine learning approach. For example in a supervised manner, the correct decomposition could be learned from a corpus of annotated questions. The correct decomposition consists of the following bits of information:

- where to split
- how to adjust sentence types (here: question types)
- how to integrate the subanswer(s) into the original question to form the revised question(s)

But there are not many frequent decomposition classes on the deep semantic level, so that an explicit manual listing is the most efficient classifier implementation. For the same reason, no rule-based approach for implementing the decomposition methods was selected; instead, the decomposition methods were implemented using an API for semantics networks of the MultiNet formalism.

The decomposition methods described above are applicable with minimal changes to languages other than German as long as a MultiNet producing parser (or similar) for the language is available because it works on the semantic representation only and not on surface strings. The methods check for semantic network relations (and only rarely for language-specific concepts) so that a transfer from German to other languages will be fast and straightforward.

3 RELATED WORK

Some of the decomposition classes have been investigated in one or the other form. For example, decomposition has been tried in the context of temporally restricted questions. [8] showed the example *Who was the German Chancellor when the Berlin Wall was opened?* Their method works not on the semantic level like ours, but mainly on the syntactic level. There has not been any large-scale evaluation of this decomposition class or several decomposition classes.

[3] uses the term *decomposition* in a wider sense in the context of domain-specific QA. For questions like essay questions or biographical questions (e.g. *Describe the life of Andrea Palladio.*), decomposition means coming up with several questions so that the summary of their answers will provide an informative answer to the user's original question.

A general objection to investigating decomposition could come from logic-oriented approaches: If you had one large knowledge base and an adequate calculus, there would be no need for decomposition; a standard theorem prover will do all the magic automatically and find answers over as many intermediate steps as needed. But a logic-oriented approach to advanced questions whose answers must be constructed from several passages or documents needs to devise a method to load the formulas for the relevant sentences, passages, or documents into main memory at the same time. Note that the corpus used since QA@CLEF 2007 comprises around 1700 million relations. This collection is still medium-sized, and our QA method (mainly: sentence-oriented semantic network comparison plus relation or concept indexes, coreference resolution, and question decomposition, [4, 5]) is also applicable to collections that are one or two order of magnitudes larger. The number of 1700 million relations comes from counting edges in the simplified form of semantic networks of the QA system InSicht used for evaluation in this paper. Standard theorem provers are known to be not well suited for such large collections of relations (atomic formulas) and large set of rules, see for instance [1].

[6] The annotation of the QA@CLEF test sets will be made freely available.

4 EVALUATION

The first evaluation is aimed to determine how well our system can classify a question into one of the decomposition classes or into the rest class *atomic* (or non-decomposable). Table 3 shows that precision and recall are promising, but not perfect. Imperfect precision is normally no problem: As false positive (or spurious) decompositions rarely lead to highly scored answer candidates (if any), the effect of false positive decompositions is mainly an increased run time, but no precision loss for the QA system as a whole (as also witnessed by the second row of Table 4). On the other hand, imperfect recall is not too problematic because the annotation describes an upper bound. So, many missed decompositions would never have led to a correct answer for other reasons. But as Table 3 indicates there is room for improving the classifier, for example to detect reliably questions where operational decomposition is unlikely to succeed (beyond the technique from Section 2.6).

The obvious choice for evaluating the benefit of question decomposition is to count how many additional questions are correctly answered with decomposition added to our deep QA system InSicht.[7] The results for the decomposable questions from QA@CLEF 2004–2008 are shown in Table 4 (16 correct NIL answers are among the correct results with and without decomposition). For simplicity, the QA system tries only one decomposition per question. Note that the number of decomposable questions given in Table 2 is only an upper bound; therefore the overall performance gain of 5.3% achieved by question decomposition looks promising. This increase is already statistically significant at the level of p=0.1.

The run time penalty imposed by question decomposition is small. A simple cache containing subquestions and their answers can help to amortize the additional run time when using question decomposition: popular subquestions will appear again and again (like *city in Italy* and *beach in California* in the domain of tourism) so that the subquestions will not incur any overhead. To improve performance further, one can integrate this subquestion cache with a general cache for original questions. In this way, the number of cache hits increases because answers to previous original questions can be used as instant answers to subquestions from question decomposition.

Table 3. Results of automatic decomposition classification

Class	Precision (%)	Recall (%)
decomposition	65.3	74.0
atomic	96.7	91.7
all	90.0	88.5

Table 4. Effect of decomposition on the decomposable questions from QA@CLEF 2004–2008 for our QA system

Class	Correct answers		
	without decomposition	with decomposition	relative change
decomposition	25	43	+72.0%
atomic	314	314	±0%
all	339	357	+5.3%

5 CONCLUSION

We have described six decomposition methods. Their application to semantic representations of questions posed in a QA system has shown significant improvements, even on question test sets that typically contain easier questions than the ones that can profit from decomposition most. In the future, we would like to develop further decomposition methods, by analyzing manually or automatically larger question sets.

Some of the decomposition methods described above (especially local decomposition) have a positive impact on GIR; but also for general IR queries some of the investigated decomposition methods can be exploited to generate promising variants of NL queries during query expansion. We would like to apply a QA system with question decomposition to a GIR task in a more general way. We have some initial positive evaluation results from GeoCLEF (the GIR task at CLEF) showing that additional relevant documents can be found for topics or queries with descriptions of locations that are typically not found in gazetteers or similar resources, e.g. geographic descriptions like *cities belonging to Southeastern Europe* or *islands in the southern part of the Baltic Sea*.

REFERENCES

[1] Johan Bos, ‘Towards wide-coverage semantic interpretation’, in *Proceedings of the 6th International Workshop on Computational Semantics (IWCS 6)*, pp. 42–53, (2005).

[2] Danilo Giampiccolo, Pamela Forner, Anselmo Peñas, Christelle Ayache, Dan Cristea, Valentin Jijkoun, Petya Osenova, Paulo Rocha, Bogdan Sacaleanu, and Richard Sutcliffe, ‘Overview of the CLEF 2007 multilingual question answering track’, in *Results of the CLEF 2007 Cross-Language System Evaluation Campaign, Working Notes for the CLEF 2007 Workshop*, Budapest, Hungary, (2007).

[3] Sanda Harabagiu, ‘Questions and intentions’, in *Advances in Open Domain Question Answering*, eds., Tomek Strzalkowski and Sanda Harabagiu, volume 32 of *Text, Speech and Language Technology*, 99–147, Springer, Dordrecht, (2006).

[4] Sven Hartrumpf, ‘Question answering using sentence parsing and semantic network matching’, in *Multilingual Information Access for Text, Speech and Images: 5th Workshop of the Cross-Language Evaluation Forum, CLEF 2004*, eds., C. Peters, P. Clough, J. Gonzalo, G. J. F. Jones, M. Kluck, and B. Magnini, volume 3491 of *Lecture Notes in Computer Science*, 512–521, Springer, Berlin, (2005).

[5] Sven Hartrumpf, ‘Extending knowledge and deepening linguistic processing for the question answering system InSicht’, in *Accessing Multilingual Information Repositories: 6th Workshop of the Cross-Language Evaluation Forum, CLEF 2005, Vienna, Austria, Revised Selected Papers*, eds., Carol Peters, Fredric C. Gey, Julio Gonzalo, Gareth J. F. Jones, Michael Kluck, Bernardo Magnini, Henning Müller, and Maarten de Rijke, volume 4022 of *Lecture Notes in Computer Science*, 361–369, Springer, Berlin, (2006).

[6] Hermann Helbig, *Knowledge Representation and the Semantics of Natural Language*, Springer, Berlin, 2006.

[7] Johannes Leveling and Sven Hartrumpf, ‘University of Hagen at GeoCLEF 2007: Exploring location indicators for geographic information retrieval’, in *Results of the CLEF 2007 Cross-Language System Evaluation Campaign, Working Notes for the CLEF 2007 Workshop*, Budapest, Hungary, (2007).

[8] Günter Neumann and Bogdan Sacaleanu, ‘Experiments on cross-linguality and question-type driven strategy selection for open-domain QA’, in *Accessing Multilingual Information Repositories: 6th Workshop of the Cross-Language Evaluation Forum, CLEF 2005, Vienna, Austria, Revised Selected Papers*, eds., Carol Peters, Fredric C. Gey, Julio Gonzalo, Gareth J. F. Jones, Michael Kluck, Bernardo Magnini, Henning Müller, and Maarten de Rijke, volume 4022 of *Lecture Notes in Computer Science*, 429–438, Springer, Berlin, (2006).

[7] One could also investigate how many additional pairs of answer candidate and support are found by decomposition. Such additional pairs help the answer selection module if there are more ways to investigate the validity of an answer candidate.