Exploiting Linguistic Indices and Syntactic Structures for Multilingual Question Answering: ITC-irst at CLEF 2005

Hristo Tanev, Milen Kouylekov, Bernardo Magnini, Matteo Negri, and Kiril Simov*
Centro per la Ricerca Scientifica e Technologica ITC-irst
* Bulgarian Academy of Sciences
tanev, kouylekov, magnini, negri@itc.it
* kivs@bultreebank.org

Abstract

This year we participated at 4 Question Answering tasks at CLEF: the Italian monolingual (**I**), Italian-English (**I/E**), Bulgarian monolingual (**B**), and Bulgarian-English (**B/E**) bilingual task. While we did not change the approach in the Italian task (**I**), we experimented with several new approaches based on linguistic structures and statistics in the **B**, **I/E**, and **B/E** tasks.

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; H.3.7 Digital Libraries; H.2.3 [**Database Managment**]: Languages—*Query Languages*

General Terms

Algorithms, Measurement, Experimentation

Keywords

Question answering, Multlinguality, Syntactic index, Syntactic structures, Syntactic network, Tree edit distance, Dependency trees

1 Introduction

This year we participated at 4 QA tracks: the Italian monolingual (**I**), Italian-English (**I/E**), Bulgarian monolingual (**B**), and Bulgarian-English (**B/E**) bilingual task.

Regarding the Italian monolingual task (**I**) we did not modify our system with respect to the previous year.

In the task **B** we participate for the first time, therefore we had to build a new QA system for Bulgarian using some tools and resources from the on-line QA system "Socrates".

This year we augmented the role of the linguistic processing in our QA system DIOGENE. In particular we experimented in the cross-language tasks with two novel approaches: a tree edit distance algorithm for answer extraction and syntactic based Information Retrieval (IR). Although these syntactic based approaches did not bring improvements in terms of performance, we regard these experiments as a step towards strengthening the linguistic framework on which our QA system is based. Moreover, we tested a new model for indexing of syntactic structures.

The rest of the paper is structured as follows: section 2 provides a brief overview of the QA approaches based on syntactic structures, section 3 describes the syntactic tree edit distance algorithm which we used for answer extraction, section 4 introduces our syntactic indexing and Information Retrieval (IR) model, section 5 describes our new system for QA in Bulgarian "Socrates 2", section 6 provides overview of our CLEF results, and section 7 outlines our directions for research in the future.

2 Using the syntax in a Question Answering system. State of the art

Two consecutive years we participated at the CLEF competition with a QA system which mostly relies on a multilingual statistical module which mines the Web to validate the answer (see [Magnini 2002] for details). This year we decided to augment the role of the linguistic knowledge in the process of answer extraction and validation. We carried out two experiments for considering the syntactic structure of the sentence. Our experiments were inspired by the fact that many QA systems consider the syntactic structures of the question and the sentences which may contain the candidate answer. For example, the top performing system of the last years in the TREC *QA* track [Vorhees 2004] - the LCC System [Moldovan et al. 2002] uses deep syntactic parsing and representation in logical form for answer extraction. Deep syntactic analysis is used also by the *Shapaqa* system [Buchholz 2001] and *DIMAP-QA* system [Litkowski 2001].

One of the best performing in the TREC 2004 track is a *QA* group from the university of Singapore [Cui et al. 2005]. They base the answer extraction module of their system on pre-extracted syntactic patterns and approximate dependency relation matching. The authors point that a weakness of the *QA* systems that incorporate parsing is that they rely on exact matching of relations. They claim that although this approach has high precision it has problems with recall. They propose a corpus based evaluation of similarities of the dependency relations using point wise mutual information (PMI) [Church and Hanks 1989]. In this way they calculate the cost of transforming the syntactic tree of the question to a candidate syntactic tree of the document.

In [Punyakanok et al. 2004] the authors build a *QA* system based on a mapping algorithm that is a modification of the *edit distance* algorithm presented in [Zhang and Shasha 1990] for syntactic trees. Their approach is handling the language variability problem by calculating the cost of missing or changed information. The authors propose the following cost operations: insert tree; delete tree; change tree. They make rough approximation of the operations cost. The authors prove, that with this approach they outperform the simple *bag-of-word* approach.

3 Answer extraction using Tree-Edit Distance

We carried out this experiment in the Italian-English cross-language task. First, we used AltaVista to translate the questions from Italian to English (we used also some pre-processing and post-processing translation rules). Second, using sentence-level IR from our syntactic index SyntNet (see section 4.2), we acquired already parsed sentences which contain question keywords. Third, we used a tree edit distance between the affirmative form of the question and the candidate sentences to extract the answer.

3.1 Translation from Italian to English

We used the Altavista interface to Systran (*http://translate.av.com*) to translate the questions from Italian into English. A set of pre-processing and post-processing transformation rules was applied to correct some of the wrong output produced by the automatic translation. Example:

- Pre-processing rule - Dov ′ e ... $->$ Dove é ....
- Post-processing rule - Which thing .... $->$ What ....

We also used a Collins dictionary to translate the words that were not translated by Altavista.

3.2 Edit distance on dependency trees

After we extract candidate sentences which may contain the answer, we used a modification of the *tree edit distance* algorithm presented in [Punyakanok et al. 2004] and [Zhang and Shasha 1990], in order to identify the sentence closest to the question in terms of edit distance and to extract the answer from it. We adapted our algorithm to use syntactic trees already indexed in our syntactic index (we used MiniPar [Lin 1998] to obtain the parse trees in the index).

Since the [Zhang and Shasha 1990] algorithm does not consider labels on edges, while dependency trees provide them, each dependency relation R from a node A to a node B has been re-written as a complex label B-R concatenating the name of the destination node and the name of the relation. All nodes except the root of the tree are relabeled in such way. The algorithm is directional: we aim to find the best (i.e. less costly) sequence of edit operations that transform the dependency tree of the candidate answer sentence into the dependency tree of the question affirmative form. According to the constraints described above, the following transformations are allowed:

- **Insertion**: insert a node from the dependency tree of question into the dependency tree of the candidate answer sentence.
- **Deletion**: delete a node N from the dependency tree of the answer sentence. When N is deleted all its children are attached to the parent of N. It is not required to explicitly delete the children of N as they are going to be either deleted or substituted on a following step.
- **Substitution**: change the label of a node N1 in the answer sentence tree into a label of a node N2 of the question tree. Substitution is allowed only if the two nodes share the same part-of-speech. In case of substitution the relation attached to the substituted node is changed with the relation of the new node.

To adapt the algorithm we addressed the following problems:

1. Transform the dependency tree of question into the dependency tree corresponding to it's affirmative form.
2. Reorder the tree nodes to create an order of the children.
3. Estimate the costs of the delete, insert and replace operations.

The dependency tree of the question is transformed into affirmative form using a set of hand written rules whcih are activated according to the question and answer types. For some answer types a simple hand-crafted pattern that represents the most frequent syntactic relations between the question focus and the answer of the question was used. Questions with such answer types are questions that have a measure as an answer (height, length, etc.)

The *edit distance* algorithm presented in [Punyakanok et al. 2004] and [Zhang and Shasha 1990] requires an ordering on the children of the syntactic tree. We imposed an order on the children of a node in the tree based on the label of the dependency relations and the lexicografic order of the words in the children nodes.

In [Punyakanok et al. 2004] the authors use add-hoc weights of the basic edit operations. In our approach we decided to use cost based on statistic measures. To estimate such cost, we define a weight of each single word representing its relevance through the *inverse document frequency (idf)*, a measure commonly used in *Information Retrieval*. If N is the number of documents in a text collection and N_w is the number of documents of the collection that contain word w then the *idf* of this word is given by the formula:

$$idf(w) = \log \frac{N}{N_w} \tag{1}$$

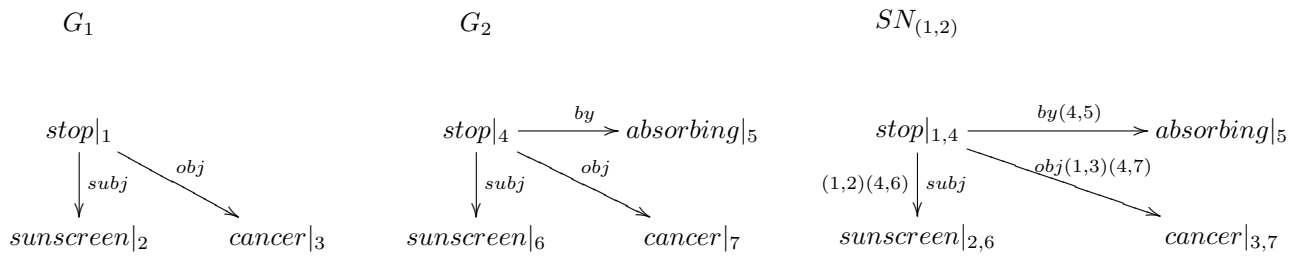

Figure 1: Two parse trees and their Syntactic Network.

The weight of the *insertion* operation is the *idf* of the inserted word. The most frequent words (e.g. stop words) have a zero cost of insertion. In the current version of the system we are still not able to implement a good model that estimates the cost of the deletion operation. In the current experiments we set the cost of deletion to 0. To determine the cost of substitution we used a dependency based thesaurus available at *http://www.cs.ualberta.ca/lindek/downloads.htm*. For each word, the thesaurus lists up to 200 most similar words and their similarities. The cost of a *substitution* is calculated by the following formula:

$$subs(w_1, w_2) = ins(w_2) * (1 - sim(w_1, w_2)) \quad (2)$$

where w_1 is the word from text that is being replaced by the word w_2 from hypothesis and $sim(w_1, w_2)$ is the similarity between w_1 and w_2 in the thesaurus multiplied by the similarity between the corresponding relations. The similarity between relations is stored in a database of relation similarities obtained by comparing dependency relations from a parsed local corpus. The similarities have values from 1 (very similar) to 0 (not similar). If there is no similarity, the cost of substitution is equal to the cost of inserting the word *w2*.

3.3 Answer Extraction

All the sentences retrieved by the IR module of the system are sorted based on the edit distance between their syntactic trees and the affirmative form of the question. As candidate answers we extracted noun phrases part of the syntactic tree of the sentences with the lowest edit distance score.

4 Syntactic Based Information Retrieval

4.1 Syntactic Network

The *Syntactic Network (SyntNet)* is a formalism for representation of a set of dependancy syntactic graphs (input graph set) produced from a dependency parser. Equal sub-structures from the input graph set are merged in one structure in SyntNet. This property facilitates identification of repeating subs-structures, allows for efficient calculation of their frequency, and makes possible efficient mining of structures which span over certain words. This last property was extensively used in our syntactic based IR experiment.

SyntNet models an input graph set, in which each of the graphs represents the syntax of a sentence from a text corpus. In a dependency syntactic graph the vertices are labeled with words or word lemmas and part of speech. In the dependency graphs each two words w and w' are connected with a directed arc if and only if w governs w' . Arcs in the dependency graph are labeled with syntactic relations (see figure 1).

When the SyntNet is built from the input graph set, all vertices labeled with the same word and part of speech are merged in one vertex. Moreover, all equally labeled dependency arcs which connect equally labeled vertices in the same direction are merged in one arc. Therefore, in SyntNet

each vertex and arc usually represents more than one vertex and arc from the input graph set. On figure 1 two dependency graphs G_1 and G_2 are merged in one SyntNet $SN(1,2)$. Each vertex in G_1 and G_2 is labeled with an unique number (e.g. $cancer_{|3}$, $cancer_{|7}$). Arcs are labeled with number pairs - the number of the vertex from which the arc begins and the number of the vertex in which the arc enters (e.g. the arc $stop_{|4} \rightarrow sunscreen_{|6}$ in G_2 is labeled with the pair (4,6)). When the vertices and arcs from G_1 and G_2 are merged in the SyntNet $SN(1,2)$ on figure 1 their numerical labels are also merged (e.g. $cancer_{|3,7}$) in sets of numerical labels. These numerical labels in the SyntNet allow for tracing repeating structures and calculating their frequency. For example on figure 1 we may trace the numerical labels in the sub-structure $sunscreen \leftarrow stop \rightarrow cancer$ and we can see that two possible paths exist following the numerical labels on the arcs on $SN(1,2)$: ($2 \leftarrow 1$, $1 \rightarrow 3$) and ($6 \leftarrow 4$, $4 \rightarrow 7$). Each of these paths corresponds to one occurrence of the sub-structure in the input graph sequence, therefore the above mentioned substructure appears two times.

4.2 Indexing the English CLEF collection

We parsed the English CLEF collection of texts with MiniPar [Lin 1998] and built a SyntNet representation from the parsed sentences. The SyntNet model was implemented as a relational database under the MySQL platform.

4.3 Retrieving and Ranking Sentences

When the question keywords are translated from Italian or Builgarian to English (see [Negri et al. 2003] and [Osenova et al. 2004] for details) we use the syntactic index SyntNet to retrieve the best sentences.

The retrieving process begins with identification of the vertices in SyntNet which represent the question keywords. For example if the question is "What stops cancer?" and the SyntNet is $SN(1,2)$ on figure 1, we will take the vertices *stop* and *cancer*. We call them *keyword vertices*. Each keyword vertex has a weight assigned - derived from its IDF.

Tracing the SyntNet up from a keyword vertex kv we record for each vertex v we encounter on our way what is the distance between v and kv. This distance is calculated from the sum of the IDF of the vertices which stay between kv and v. We will call these vertices *intermediate vertices*.

Keyword vertices which appear as intermediate vertices contribute 0 to the distance. Moreover, if there is some similarity (we measure similarity between words using a syntactic distributional approach similar to [Lin 1998a]) between an intermediate vertex and any of the key vertices, this intermediate vertex contributes to the distance only a part of its IDF.

We will denote thus calculated distance by $|kv\ v|$. Obviously, as many informative vertices stay between kv and v, as bigger their distance is. As less informative these intermediate vertices are, as low the distance is. On the other hand, as similar the intermediate vertices are to the question keywords, as lower the distance is. Actually, $|kv\ v|$ models the quantity of information in the path between kv and v which is not shared with the question.

For each vertex v from the SyntNet which is a root of a tree spanning over a set of question keywords $kwQv_1, kwQv_2, ..., kwQv_n$ we define:

$$syntscore(v) = \frac{\sum_{kwQv_i} IDF(kwQv_i)}{I(Q) + \sum_{kwQv_i} |kwQv_i\ v| + IDF(v)}$$

In this formula I(Q) is the sum of the IDF of all the question keywords. If v is a keyword vertex, $IDF(v)$ in the above formula is considered to be 0. This score we call *syntactic context score* and it evaluates a tree spanning over some the question keywords and rooted in a vertex v. In this formula the distance between the keywords in the syntactic tree as well as the informativeness of the words are taken into consideration.

Finally, the score of a sentence S is calculated as:

$$score(s) = \max_{v \in S} syntscore(v).\frac{I(Q \cap S)}{I(Q)}$$

In this formula $I(Q \cap S)$ is the sum of the IDF of the question keywords which appear in the sentence. This formula combines the highest syntactic context score in the sentence and the relative quantity of the information that the question shares with that sentence.

For the next processing steps only the top ranked sentences are considered. As a last stage DIOGENE system performs answer extarction and Web based answer validation to choose the best answer [Magnini 2002] from the retrieved sentences.

5 A QA system for Bulgarian - "Socrates 2"

In order to participate in the monolingual Bulgarian task we decided to build a QA system for Bulgarian which uses certain templates from the "Socrates" on-line QA system [Tanev 2004], but also incorporates answer extraction techniques for questions for which no patterns exist. We call this system "Socrates 2". Moreover, we decided to build a linguistic index of the Bulgarian CLEF collection in which each word is represented with its lemma and part of speech. In this index the separate sentences were represented rather than whole documents.

5.1 Question processing

"Socrates 2" performs question classification on the basis of simple superficial templates. It classifies the question into one of the following categories: definition questions and questions which require person, location, organization, year, date, manner, reason, or generic name as an answer.

5.2 Building Linguistic Index

Instead of relying on standard search engines, we developed our own sentence retrieval engine and linguistic index of the Bulgarian CLEF collection. The text collection was split into sentences and automatically annotated with part-of-speech tags and word lemmas using the LINGUA system [Tanev and Mitkov 2002]. This linguistic annotation and the IDF of each word were encoded in the linguistic index which backs up our sentence retrieval module.

We think that such an approach is more appropriate for the QA task than the traditional document indexing approaches since it leads to more focused IR. Moreover, a sentence usually provides enough context for answer justification, which makes reasonable to perform IR on this level. In this way, however, some phenomena like intrasententional anaphora are ignored which may potentially lead to the lost of some answers. Another shortcoming of the linguistic indexing model is the probability of errors in the part-of-speech tagging and the sentence splitting processes.

5.3 Sentence retrieval

All the sentences which contain at least one of the question keywords are taken into consideration. Sentences are ranked using the following formula:

$$score(S) = \frac{I(Q \cap S)}{I(Q)}$$

, where $I(Q)$ is the quantity of the information in the question and $I(Q \cap S)$ is the quantity of the information which the question and the sentence share. The formula finds what percent of the question information content is found in the sentence. Information content of the question $I(Q)$ is measured as a sum of the IDF of its keywords. The quantity of the shared information content $I(Q \cap S)$ is the sum of the IDF of the question keywords which appear in the sentence. This can be written in the following way:

$$I(Q) = \sum_{kw_i \in kwQ} IDF(kw_i)$$

$$I(Q \cap S) = \sum_{kw_i \in kwQ \cap wS} IDF(kw_i)$$

, where kwQ are the question keywords and wS are the words from the sentence. A keyword from the question is considered to be equal to a word from the sentence if both words have the same lemma and part of speech. This sentence ranking approach turned out to work rather satisfactory in practice.

5.4 Answer extraction

For definition questions we adapted and used templates and rules already implemented in the "Socrates" on-line demo. Since these templates were already tested and tuned using real on-line users questions submitted to the Socrates Web site (http://tanev.dir.bg/Socrat.htm), we did not make any significant improvements in the system of rules.

We did not develop any specific strategy for the temporal questions, rather they were treated as factoid ones. For identification of factoid answers we created rules for extraction of generic names (without specifying if the name designates location, person, organization, or other entity), dates, and numbers. All other answer candidates (for example noun phrases which are not names or verb phrases) were ignored.

When extacting candidate answers for factoid and temporal questions, "Socrates 2" considers the top 200 ranked sentences whose score is greater than 0.5 ($score(S) > 0.5$). Moreover, only the sentences which have score greater than 0.1 of the score of the top ranked sentence are taken into consideration. In this way, we avoid to extract answers from sentences which does not contain enough question keywords.

Name identification In the general case our system for identification of names classifies as a candidate for a name each sequence of words which begin with capital letters. However, a capitalized word in the beginning of the sentence is considered a part of a name only if it is found in the dictionary of proper names integrated in the LINGUA morphological processor [Krushkov 1997] or it is an unknown word. Usually this strategy recognizes properly the names, however we noticed that often two names appear next to each other, which causes errors in the name recognition. For example, the above mentioned heuristics will extract from the text "poslanikat na Shvecia Sten Ask" ("the ambassador of Sweden Sten Ask") the name candidate "Shvecia Sten Ask" ("Sweden Sten Ask"), while "Shvecia"("Sweden") and "Sten Ask" are two separate names. To overcome this problem, we developed a name splitting strategy which is activated in cases we have a sequence of more than two capitalized names: $N_1N_2N_3...N_n$. In such cases we check if N_1 is a part of the sequence or should be treated as a separate name. Although the sequence can be splitted in each point, we empirically observated that in most cases the sequence should be splitted after N_1. Therefore, in order to simplify the task and augment the processing speed we checked only if the sequence can be splitted after N_1 or no. The test is based on the assumption that if $P(N_1|N_2N_3) < limit$ then a splitting after N_1 will take place, i.e. N_1 and $N_2N_3...N_n$ will be treated as separate names. We set experimentally $limit = 0.8$. After we introduced this name splitting strategy the name recognizer became quite accurate and the number of errors was significally decreased.

Answer scoring and ranking The score of a candidate answer A in a sentence S is calculated considering the distance in tokens between the candidate A and each of the question keywords (kw_i) which appear in the sentence and the IDF of the keywords:

$$score(A, S) = \sum_{kw_i \in kwQ \cap wS} \frac{IDF(kw_i)}{1 + \sqrt{|A\ kw_i|}}$$

This formula gives higher score to the candidate answers which appear close to the most important question keywords. When two candidate answers have equal score, the system prefers the one which appears more often in the top ranked sentences.

6 Evaluation

Task	*Overall accuracy*	*Definition (%)*	*Factoid (%)*	*Temporal (%)*
Italian (run 1)	22.0	38.0	19.2	6.7
Italian (run 2)	19.0	14.2	19.2	6.7
Bulgarian	27.5	40.0	25.0	17.7
Italian/English (run 1)	23.5	38.0	19.8	13.8
Italian/English (run 2)	13.0	38.0	5.8	0
Bulgarian/English	18.5	20.0	17.4	20.7

Table 1: QA Performance at CLEF 2005

We submitted one run at the monolilngual Bulgarian task using our new system "Socrates2". We produced two runs in the Italian monolingual task. In the first run the answer ranking for the factoids and temporal questions for which no answer-extraction patterns exist was delegated exclusively to the Web-based answer validation module. In the second run we considered the keyword density also. As in the previous year, it turned out that considering keyword density deteriorates the result. Regarding the Italian-Englisg task, we run the two experiments described in the previous sections. In the first run we used syntactic IR for factoid and temporal questions and in the second run we used tree edit distance algorithm for factoids. We used syntactic based IR also in the Bulgarian-English cross-language task.

In all the tasks we used the multilingual template-based approach for answering definition questions [Tanev et al. 2004].

With respect to the previous year our overall accuraccy on the monolingual Italian task dropped from 28% to 22%. In particular, the performance on the factoid questions was decreased by about 7% while the accuracy on the definition questions dropped only by 2%. Since the monolingual Italian system is the same as the one used the previous year, the decrease of the performance may be due to increased difficulty of this year questions. For the monolingual Bulgarian task we have 27.5% overall accuracy (25% factoid questions, 40% definition, and 17,65% temporal questions). These results are promissing taking into account the scarce resources we used this year and the non-refined named-entity recognition we performed.

Regarding the Italian-English cross-language task, our experiments with linguistic indices and tree-distance did not bring the expected results in terms of performance. The first run in which syntactic based IR was used for factoids and temporal questions resulted in 19.83% accuracy for factoids and 13.79% for temporal questions. The accuracy on the factoid questions is decreased by 2% from the previous year. The accuracy on the factoids in the second run where we tested tree-edit distance algorithm was only 5.8%. We have further to study how the tree edit distance algorithm will be integrated in the overall QA process.

In the Bulgarian-English cross-language task we have some performance improvement for the factoid questions with respect to the previous year: 17.4% vs. 11.7%. In this task we used syntactic based IR. The translation Bulgarian-English dictionaries were also enriched with person names. We have further to study if the improvement in the performance is due to the new IR approach, the improved translation, or the difficulty of the questions in the **B**/E task is lower with respect to the previous year.

7 Conclusions and future direction

This year we experimented with linguistic indices for Bulgarian and English. The Bulgarian QA system based on the linguistic index achieved promising results considering the simplicity of the QA approach. We tested two novel approaches based on the syntax: one for IR and one for answer extraction. Although the syntactic based approaches did not show high performance, we continue our research in the exploitation of syntactic structures in QA. We believe that the methods

which use linguistic knowledge are potentially more accurate than superficial ones. However, our experience at CLEF 2005 showed that different problems have to be overcome when using syntax in QA: parsing errors, syntactic paraphrases, efficiency issues, etc.

In our future work we intend to study better the use of tree edit distance algorithms. We would like also to study deeper the potential of the SyntNet model: Building on this model, we intend to develop algorithms for pre-selection of the candidate answers. We intend also to use the SyntNet in different lexical acquisition experiments, which will support the development of our QA system.

8 References

Buchholz, S. 2001. Using Grammatical Relations, Answer Frequencies and the World Wide Web for TREC Question Answering. *In Proceeding of the TREC-10 Conference 2001*, pages 496-503.

Church, K. and Hanks, P. 1989. Word Association Norms, Mutual Information, and Lexicography. *In Proceedings of ACL-89*, pages 76-83. Vancouver, Canada.

Cui, H., Li, K., Sun, R., Chua, T., Kan, M 2005. University of Singapore at the TREC-13 Question Answering Main Task *In Proceedings of the TREC-13*

Krushkov, H. 1997. Modelling and Building of Machine Dictionaries and Morphological processors Ph.D. Thesis, University of Plovdiv (in Bulgarian)

Lin, D. 1998. Dependency-based evaluation of MINIPAR. *In Proceedings of the Workshop on Evaluation of Parsing Systems at LREC-98.* Granada, Spain.

Lin, D. 1998a. Automatic Retrieval and Clustering of Similar Words *In Proceedings of COLING-ACL98* Montreal, Canada

Litkowski, K. 2001. CL Research Experiments in TREC-10 Question Answering *In proceeding of the TREC-10 Conference 2001*, pages 251-260.

Magnini, B., Negri, M., Prevete, R., Tanev, H. 2002. Is it the Right Answer? Exploiting Web Redundancy for Answer Validation. *Association for Computational Linguistics 40th Anniversary Meeting (ACL-02), of Pennsylvania, Philadelphia.*

Moldovan, D., Harabagiu, S., Girju, R., Morarescu, P., Lacatsu, F., Novischi, A 2002-2003. LCC Tools for Question Answering. *NIST Special Publication: SP 500-251 The Eleventh Text Retrieval Conference (TREC 2002).*

Negri, M., Tanev, H., Magnini, B. 2003. Bridging Languages for Question Answering: DIOGENE at CLEF 2003 *In Proceedings of CLEF-2003, Trondheim, Norway*

Osenova, P., Simov, A., Simov, K., Tanev, H., Kouylekov, M. 2004. Bulgarian-English Question Answering: Adaptation of Language Resources *In Proceedings of CLEF-2004, Bath, UK*

Punyakanok., V.,Roth, D. and Yih, W., 2004 Mapping Dependencies Trees: An Application to Question Answering *Proceedings of AI & Math 2004*

Tanev, H. 2004. Socrates: A Question Answering Prototype for Bulgarian *Recent Advances in Natural Language Processing III, Selected Papers from RANLP 2003* John Benjamins, Amsterdam/Philadelphia

Tanev, H. an Mikov, R. 2002. Shallow Language Processing Architecture for Bulgarian *In Proceedings of COLING 2002* Taipei, Taiwan

Tanev, H., Kouylekov, M., Negri, M., Coppola, B., Magnini, B. 2004. Multilingual Pattern Libraries for Question Answering : a Case Study for Definition Questions *In Proceedings of LREC 2004* Lisbon, Portugal

Vorhees, E. 2004 Q&A Track Guidelines *In proceeding of TREC-13 2004*

Zhang, K., Shasha, D. 1990 Fast algorithm for the unit cost editing distance between trees. *Journal of algorithms, vol. 11, p. 1245-1262, December 1990.*