Kybernetika

Lukáš Svoboda; Luboš Popelínský
Communication with www in Czech

COMMUNICATION WITH WWW IN CZECH

Lukáš Svoboda and Luboš Popelínský

This paper describes *UIO*, a multi–domain question–answering system for the Czech language that looks for answers on the web. *UIO* exploits two fields, namely natural language interface to databases and question answering. In its current version, *UIO* can be used for asking questions about train and coach timetables, cinema and theatre performances, about currency exchange rates, name–days and on the Diderot Encyclopaedia. Much effort have been made into making addition of a new domain very easy. No limits concerning words or the form of a question need to be set in *UIO*. Users can ask syntactically correct as well as incorrect questions, or use keywords. A Czech morphological analyser and a bottom–up chart parser are employed for analysis of the question. The database of multi–word expressions is automatically updated when a new item has been found on the web. For all domains *UIO* has an accuracy rate about 80 %.

Keywords: question answering, natural language processing

AMS Subject Classification: 68T50, 03B65

1. MOTIVATION

Most of the information frequently needed – public transport timetables, information about train departures, cultural events etc. – can be found on the web. Typically this kind of information consists of two parts, temporal information (e. g. the time of a train departure or the hour of theatre performance) and non–temporal information (e. g. a train destination, a title of a theatre play etc.).

An appropriate system must be able to recognise and correctly analyse temporal and non–temporal information in the query as well as in web pages. As a rule, a user aims at receiving only the most actual information (e. g. which train leaves for Prague this afternoon, when is a particular movie on). General purpose web–based question answering systems [5, 27][1] are not very convenient for such tasks also for another reason. Namely those information pages may be generated dynamically. On the other hand, not all features of those question answering systems need to be employed. Relevant web pages (e. g. a public transport time table) have a well–defined structure which is much easier to analyse than unformatted text. We did not aim at developing a general–purpose question answering system for Czech.

---

[1] A list of question answering systems available on the web can be found on http://www.answerbus.com/systems/index.shtml.

As each particular web server has its own design and a particular user interface it may be difficult for unexperienced user to use them. It was the other reason for development of yet–another question answering system. We also wanted to check how current natural language processing techniques for Czech can help in solving one particular subtask of question answering.

We developed a multi–domain question answering system *UIO* [25] that can answer questions in Czech. Information for the answer is extracted from several frequently used web servers. In its current version, *UIO* can already be used for asking questions about train and coach timetables, cinema and theatre performances, currency exchange rates, name–days and on the Diderot Encyclopaedia. Users can ask syntactically correct as well as incorrect questions, or use keywords. No limits concerning words or the form of a question need to be set in *UIO*.

Main advantage of *UIO* is flexibility. We put much effort to make addition of a new domain into *UIO* very easy. A new module for recognition of multiword expressions (e. g. names of towns, cinemas etc.) has been developed.

This article is organised as follows. Section 2 describes the general architecture of QA systems and Section 3 brings information on relevant QA systems. In Section 4 we give an overview of *UIO* system. Technical details can be found in Section 5. Section 6 contains an example of the query processing. The results obtained with *UIO* are discussed in Section 7. Comparison with the general QA schema can be found in Section 8. We conclude with comparison with other QA systems (Section 9) and with plans for a future work (Section 10).

2. QUESTION ANSWERING SYSTEMS

We briefly describe a general architecture for the question answering task following [11]. The first step is a *question analysis*. If necessary, the user is *requested for clarification* of the question before the analysis. Various systems provide various techniques of analysis, from extracting a set of keywords, expanding this set using synonyms and morphological variants [24] to partial parsing [22] and employing a hierarchy of questions [18]. Although it is not very frequent, a *user model* that may contain e. g. user preferences can also be used.

The QA system is assumed to have an access to a large *document collection* which serves as a knowledge resource for answering questions. It is useful to *pre process this document collection* for faster, and maybe better, answering the question. Most QA systems[2] use only term indexing. SRI Highlight Information Extraction system [17] goes further and employs natural language processing techniques like sentence splitting, tagging, name entity recognition and chunking over the document collection. Different methods of semantic tagging has been also explored [13, 20].

Then *candidate documents* – a subset of the document collection or a sub-part of the knowledge base extracted from the collection of documents – which most likely contains information needed for an answer generation is selected based on the question analysis. Most of the existing QA systems use information retrieval
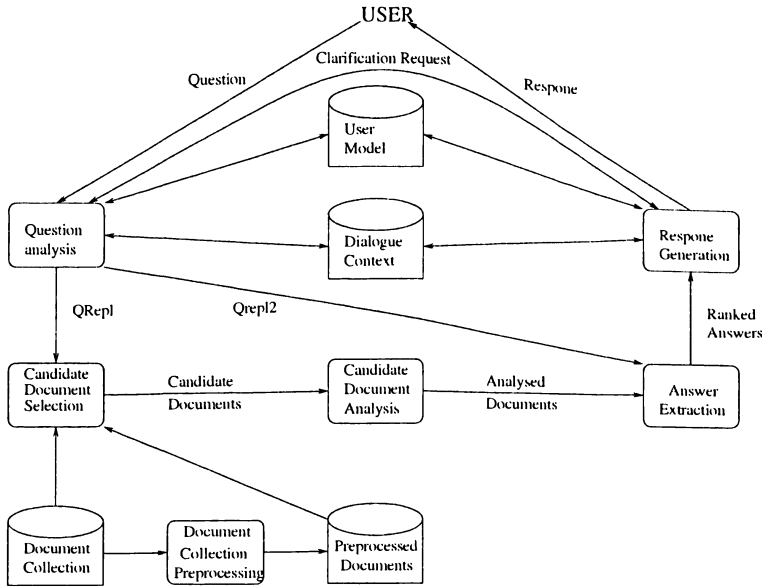
---

[2]http://trec.nist.gov

Fig. 1. A general architecture of question–answering systems.

search engines for this task. Techniques for passage (sub-part) selection [6] are useful here. Ranking techniques can be used for obtaining a sorted list of documents or of sub-parts of documents. The candidate document selection can be followed with *candidate document analysis* if the document pre–processing step was not deep enough. At least classification of multiword strings into named entities[3] should be performed for recognising names of persons, locations, companies, products, addresses etc. Information extraction techniques [1, 16] can be applied.

This subset of documents is then used for *answer generation*. Internal representation of the questions and the documents are matched against each other using various matching techniques. As a result, a set of answer–bearing text units is generated. Additional constraints are then employed to find the best answer. The order of these two operations can be reversed [24]. The internal representation of the best answer is in the step of *response generation* transformed into the form comprehensive to the user [5].

3. RELATED WORK

START[4] (SynTactic Analysis using Reversible Transformation) [13] is one of the best QA systems for English. It uses so called *T-expressions*, (subject object relation) triples and *S-expressions* (correspondence between T-expressions of the same semantic meaning). Huge domain knowledge that is incorporated in *START* as

---

[3]http://www.itl.nist.gov/iaui/894.02/related_projects/muc/
[4]http://www.ai.mit.edu/projects/infolab/

well as advanced indexing techniques result in its good performance. Users can ask question about geography, art and many other domains. *Ask Jeeves*[5] is the first commercial application on the web. It answers about four millions queries a day. Its database consists of URL addresses that have been manually indexed. After syntactic analysis of a question, *Ask Jeeves* looks for the URL that is most similar to the question. *AnswerBus*[6] is one of the new QA systems. It enables the user to ask questions in six languages. *AnswerBus* employs other web search engines to obtain references that are then filtered.

Communication with computer in Czech has a long history. First QA systems for Czech appeared in early eighties and concerned mainly a natural language interface [8, 9, 12, 21]. TIBAQ (Text-and Inference-Based Answering of Question) [21] was able to answer questions on Czech and English scientific texts. It used dependency tree for sentence meaning representation. Information on further development of TIBAQ can be found in [10].

An interesting, and maybe the most extensive, QA project[7] has been recently developed in the same group. It has been intended for answering question about schedule of lectures at MFF Charles University but it can be adapted to other domains. Two modules, *Trans* and *NLQ3*, have been developed. *Trans* performs full analysis – lexical, syntactic and semantic – of the query. The result is then compiled into SQL query. In opposite, *NLQ3* starts with finding keywords with help of an internal dictionary. Rewritten rules are then employed. The results is again expressed in SQL.

Much effort has been also put into development of speech QA systems. The most famous is the multilingual dialog system [2, 19] that has been developed at the University of West Bohemia, Plzeň. It can communicate in four languages, Czech, Slovak, Slovenian and German. This system has been experimentally tested in communication with a train and air-plain timetable information systems and became as a basis for several industrial applications.

4. OVERVIEW OF *UIO*

UIO is a multi-domain question answering system that can communicate with the web in Czech. *UIO* can answer question on following domains:

- train and bus connections
- cultural events (cinema and theatre performances) in main cities in the Czech Republic
- currency exchange rates
- name–days.

If no domain has been chosen with high enough confidence, the Diderot Encyclopaedia[8] is used for finding an answer. Several examples of questions that *UIO* can answer

---

[5]http://www.ask.com
[6]http://www.answerbus.com
[7]M. Brouček et al, http://www.ms.mff.cuni.cz/ mbuz5049/ (February 2000)
[8]http://www.diderot.cz/

are below. The first and the third question are grammatically correct, the second one contains only keywords. The fourth question has partially correct syntax and contains words with typos.

```
Kde hraji Pana prstenu?
/Where is The Lord of the Rings played?/

Pan prstenu
/The Lord of the Rings/

Jak se dostanu rychlikem zitra rano z Prahy do Ostravy?
/How can I get from Prague to Ostrava by express train tomorrow morning?/

rchlikem z Prahy do Ostravy ztra rano
/rpid train from Prague to Ostrava tmmorrow morning/
```
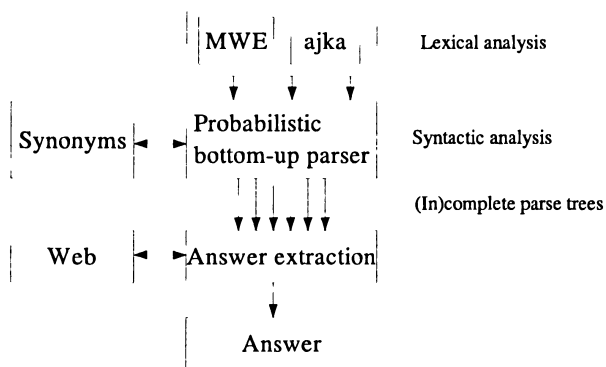

Fig. 2. *UIO:* General scheme.

A question answering (QA) system [11], like *UIO*, to be used in real life must be able to understand syntactically correct questions as well as incomplete queries or queries with typos. The following features must be take in account. There are some domains for which it is more comfortable to use *keywords* instead of a whole sentence. In such cases it is not desirable to let the question analyser parse the question. The second feature concerns *grammatically unrecognised questions*. The used grammar may be not rich enough to recognise a question even if the question is grammatically correct. In that case the parts that have been correctly recognised should also be returned. A question analyser must provide information about the correct reading of possible *typing errors* together with a weight of that reading. *Adding a new domain* to *UIO* (and/or modification of interface to the existing domain) should be straightforward.

The general scheme of *UIO* is in Figure 2. Several NLP tools are used in *UIO*, namely a morphological analyser, a tool for MWE recognition, a bottom–up parser and a dictionary of synonyms. In the the current version of *UIO* these tools are employed only for analysing the question. First, the lexical analysis is performed exploiting the database of multiword expression(MWE). As the second step, a syntactic analysis of the question is performed. This step may result in a set of correct

parse trees. *UIO* enables partial parsing. It happens very often that the question has not been fully recognised. Then the parser returns a set of parse trees for each part of the question. Using this result, *UIO* first recognises the relevant domain. The rest of the result of the analysis serves as an input to the extraction algorithm for this domain.

As soon as the question has been analysed – which means the domain has been recognised, among other things – the answer extraction algorithm is called to extract the right answer from the candidates. The input of this algorithm consists of two parts.

1. names of domains and their weightings. The name of a domain may be either an address of a web server, e. g. in the case of public transport database, or the location of the database, e. g. in the case of Diderot Encyclopaedia.

2. Parameters for extracting the right information from the server or from the database, again with weightings.

UIO is a multi–domain system and a lot of work has been done to make the addition of a new domain very easy. The administrator only need

- to extend the MWE database;
- to extend the grammar;
- to define obligatory and facultative attributes of the new domain;
- to define a source of data (URL or a name of file);
- to define a template for information extraction from the data source;
- to define a filter for building a response from the extractedinformation;
- and for formatting the output.

A part of existing grammar can be used for a new domain. It applies namely for adjectives like "cheap, first, cheapest, nearest" etc. which bear information semantically connected with objects (price in the case of *cheap**, a train or bus stop in the case of *near** etc.). The only task that has to be coded for some of domains in a programming language (Perl) is the procedure for information extraction from the web page found.

In the following section we describe the most important parts of *UIO* in more details.

5. ANALYSIS OF A QUERY IN *UIO*

5.1. Multiword expressions in Czech

Searching Multiword Expressions[9] [3, 4, 7, 13, 15] aims at finding a word or a tuple of words in the input text which form a phrase with its own semantics. For example col-locations (or name entities) like names of towns (e. g. Nové Město na Moravě), names of persons (Jan Hus), or theatres (Divadlo ABC) are MWE.

---

[9]http://lingo.stanford.edu/mwe/

Unfortunately, the methods for MWE recognition used for other languages are not fully applicable. The reason is two–fold, free order of words and the rich morphology of Czech. Moreover we need to check various abbreviation that may appear in query of the QA system. Enormous amount of potential MWE and the morphological ambiguity make the task computationally expensive. The fact that Czech morphological analysers does not know all the words that can appear in MWE makes the task even more difficult.

Search for MWE has not been solved yet for Czech although some attempts have been successful [15]. Because of it we implemented the new module that is convenient for the QA systems. Several features of this module are described below.

5.2. Lexical analysis and finding multiword expressions

The first step analysing a question is finding *multiword expressions* (MWE). Each MWE is a word or words that fully or partially appeared in the question. As words in the question can appear in different word forms (cases for nouns, different forms of verbs), the *ajka* morphological analyser for Czech [23] is employed for the lemmatization and at the same time the MWE database is searched. In the case that the word is unknown to *ajka UIO* tries to extract the stem of this word.

The basic MWE algorithms for Czech must be able to perform diacritic restoration, to find a set of MWE which can match abbreviations in the question, to find incompletely written MWE, e. g.: "vlakem do **Paky**" → "Nová Paka" OR "Stará Paka".

In the current version of *UIO*, the database of MWE consists of the following classes: names of cinemas, theatres, titles of theatre performances and movies, train and bus stations, currencies and name–days. More than one MWE can be assigned to one word (or tuple of words) of the question. Each of the found MWE has its own weighting, i. e. probability that the word (or tuple of words) can be interpreted as MWE.

Before starting the lexical analysis, the MWE database is transformed into a trie structure. Edges correspond with lemma (or stem) of words of MWE and a list corresponds to a MWE. There is a set of additional conditions which have to be checked e. g. word tags, case of the words.

Algorithm determine lemmas (or stem) of all words in the input question. Then it looks for the best list in the trie structure. At the end it checks the condition connected with the node. Detailed example of searching MWE is shown in Section 6.

It may happen that a new item, e. g. a new movie or a new cinema, has been found on a web server when evaluating a query. At that moment the MWE database is automatically updated and the new MWE is added to the particular class of items.

The output of the algorithm contains

- a location of MWE in the input text;
- lemmatized text;
- morphological tags;
- weight of the found MWE;
- information whether diacritic restoration was used;

- information if it is abbreviation;
- information if a case of characters matches that of MWE in the database.

5.3. Parsing

After this lexical analysis, a bottom-up chart parser [14, 26] for probabilistic context–free grammar (CFG) is employed. We extended the traditional parser with a rich repertoire of matching techniques. Namely, each terminal can match:

- regular expression;
- with any word which has a lemma assigned;
- with any word which has assigned the same morphological tags;
- with any word which is a synonym of an word.

These variants can be combined. A terminal can also match a word with a typing error which results in a decrease of the weighting. Non-terminal symbols can be assigned to the following categories:

normal — the usual meaning as in CFG;

semantic — these symbols match individual parts of input sentences which have their own semantic representation, e. g. date, time, mathematical expression;

contextual – they can also carry semantic context from previous queries.

The attribute of a semantic and a context non-terminal keeps semantic information. Non-terminal can be semantic and contextual at the same time. If the input query is correctly analysed by the parser then the result is given to the module for evaluation of an answer.

5.4. S-analysis

In the case of unsuccessful syntactic analysis a partial semantic analysis (called S-analysis) will be started. S-analysis starts with the result obtained with the parser. The result of the parser is a collection of all MWE and all semantic non-terminals with their weightings.

For each domain the S-algorithm creates all queries that can be built from the result of the parser, and that are relevant to this domain. Each of these candidates has a weighting computed from the weightings of the words that appear in the query. If no value of an attribute is known, the default values are used and the weighting of the candidate is decreased. For example, for the domain of train connections we have the attributes **time, date, terminal station** etc. In the case of unknown value of attributes **time** and **date**, they are automatically set to the actual time and date and the weightings are decreased.

To summarise, the weighting of the query is computed from the weightings of expressions which are taken from the lexical and syntactic analysis multiplied by the significance of an attribute for a particular domain. In this way the most probable domain and the most probable query to this domain are selected and evaluated.

6. EXAMPLE

In the example below, all Czech words have been translated into English. The rest remains unchanged.

For the question about train or bus connections that was asked on January 9

```
dotaz:
z Brna do Noveho Mesta na Mor. pozitri v 10 hodin
/from Brno to Nove Mesto na Mor. the day after tomorrow at 10 o'clock/
```

two MWE have been found:

```
0  word_form => 'Brna'                  1  word_form =>   'M\v{e}sta na Mor.'
   start_position => 1                      start_position => 3
   end_position   => 2                      end_position   => 7
   lemma =>     'Brno'                       lemma  =>   'Nove Mesto na Morave'
   class  =>  'NCZVlakZastavky'             class  =>   'NCZVlakZastavky'
   weighting => 0.95                        weighting =>   0.95
```

```
Parsing is running...
424 rules in agenda
33 derivation trees and
907 possible semantic representations, from which
2 valid semantic fragments have been found

All words have been recognized
but the query cannot be infer-ed from the grammar.
```

Although the query was not fully recognized, some parts – semantic fragments(semantic non-terminals) – have been recognized correctly, namely information about train/coach stations and the time period of travel as shown below.

```
0  HASH(0x8c3666c)                      1  HASH(0x8bb4190)
   'ACTION' => 'VLAKY'                     'TIME' => HASH(0x8c29f6c)
   'PLACE' => ARRAY(0x89c00f0)                'HOUR' => 10
      0  HASH(0x8a63e68)                       'MINUTE' => 00
         'LEMMA' => 'Brno'                     'SECOND' => 00
         'prefix' => 'from'                  'DATE' => ARRAY(0x8abb310)
         'priority' => 1                        0  HASH(0x89da968)
      1  HASH(0x8cf1e3c)                           'DAY' => 10
         'LEMMA'=>'Nove Mesto na Morave'          'MONTH' => 01
         'prefix' => 'to'                         'YEAR' => 03
         'priority' => 1                     'priority' => ARRAY(0x8c04e1c)
   'priority' => ARRAY(0x8bba27c)              0  1
      0  1                                     1  0.9801
      1  0.9025                             'start_position' => 7
   'start_position' => 0                    'end_position' => 11
   'end_position' => 7
```

UIO tries to join these two semantic fragments to build a query. As each fragment has been derived from a different set of words, there is no conflict between the fragments and the fragments can be simply joined (see below). If there is a conflict, *UIO* looks for the most probable interpretation of the question.

```
1 => 0.88454025
'ACTION' => 'VLAKY'
'TIME' => HASH(0x8c29f6c)
   'HOUR' => 10
   'MINUTE' => 00
   'SECOND' => 00
'DATE' => ARRAY(0x8abb310)
```

```
0  HASH(0x89da968)
   'DAY' => 10
   'MONTH' => 01
   'YEAR' => 03
'PLACE' => ARRAY(0x8c1e988)
   0  HASH(0x8a63e68)
      'LEMMA' => 'Brno'
      'prefix' => 'from'
      'priority' => 1
   1  HASH(0x8cf1e3c)
      'LEMMA'=>'Nove Mesto na Morave'
      'prefix' => 'to'
      'priority' => 1
```

After having built an interpretation of the question it looks for a domain as described
in Subsection 5.4. The domain of train and bus connections has been chosen.

```
0  HASH(0x8baf468)                        1  HASH(0x8b11428)
   'TIME' => HASH(0x8c56c24)                 'TIME' => HASH(0x8ab14f0)
      'CESTA' => HASH(0x8d4ae84)                'CESTA' => HASH(0x8bf9300)
         'HOUR' => 2                              'HOUR' => 2
         'MINUTE' => 30                           'MINUTE' => 35
      'DEPARTURE' => '10:00'                    'DEPARTURE' => '10:00'
      'ARRIVAL' => '12:30'                      'ARRIVAL' => '12:35'
   'CENA' => 140                             'CENA' => 140
   'DATE' => HASH(0x8a49238)                 'DATE' => HASH(0x8b0a21c)
      'DAY' => 10                              'DAY' => 10
      'MONTH' => 1                             'MONTH' => 1
   'STATION' => ARRAY(0x896b310)             'STATION' => ARRAY(0x8ac4e58)
      0  HASH(0x8b95984)                        0  HASH(0x8bebd1c)
         'TIME' => HASH(0x8bd7944)                'TIME' => HASH(0x86a97f4)
            'END' => '10:00'                         'END' => '10:00'
            'START' => '     '                       'START' => '      '
         'LEMMA' => 'Brno'                        'LEMMA' => 'Brno'
         'TYP' => 'AUTOBUS'                       'TYP' => 'AUTOBUS'
      1  HASH(0x8d4d6b4)                        1  HASH(0x8ce0f04)
         'TIME' => HASH(0x8b4cfec)                'TIME' => HASH(0x8c24370)
            'END' => '      '                        'END' => '     '
            'START' => '12:30'                       'START' => '12:35'
         'LEMMA'=>'Nove Mesto na Morave'         'LEMMA'=>'Nove Mesto na Morave'
      2  HASH(0x8b47cc4)                        2  HASH(0x8c15bfc)
         'LEMMA' => ''                            'LEMMA' => ''
   'STANICE_A' => ARRAY(0x86b2f98)           'STANICE_A' => ARRAY(0x8a32464)
      0  'Brno'                                  0  'Brno'
      1  'Nove Mesto na Morave'                  1  'Nov\'{e} Mesto na Morave'
   'DISTANCE' => 211                         'DISTANCE' => 210
```

User is then provided with the following answer.

```
Searching for train/bus connections http://www.vlak.cz/
[ 80 \% correct domain, 100 \% lematization,
1.09 \% errors and unknown words]

ze stanice: Brno do stanice: Nove Mesto na Morave  pres: ___ v 10:00
/from station: Brno to station: Nove Mesto na Morave  via: ___ at 10:00/
dne: 10.01.03 vlaky: ___ maximalne 5 prestupy
/date: 10.01.03 transport: ___ max 5 changes/
```

Finally, the answer to the question is below. The user is then provided with the
following answer.

```
10.1. 10:00-12:30                                10.1.  10:00-12:35
       -10:00    Brno  AUTOBUS          -10:00   Brno   AUTOBUS
   12:30-        Nove Mesto na Morave 12:35-     Nove Mesto na Morave
     -           211Km    140Kc                  210Km    140Kc
```

7. RESULTS

UIO was tested by different users who asked 3106 questions during a period of three months. No errors were observed when a simple question was asked like "What is on at the Art Cinema?". For complex questions (when the user was inexperienced or tried to beat the system) like "Guy, tell me how to get to Praha, and hurry up!" the system was often able to filter the irrelevant words and to answer to the relevant part of the question. Summary of results is in Table 1.

Table 1. Summary of results.

domain	number of queries (%)	accuracy (%)
train and bus timetable	32.5	89
cultural events	26.2	85
name–day	13.9	79
currency exchange rates	12.5	98
general queries	02.2	12[10]
other question	12.4	—[11]

An answer has been evaluated as correct if it contained either correct information (in the case of train and bus departures, name–days, and exchange rates) or , for the domain of cinema and theater performances, if at least one event has been found.

Information on train and bus departures was asked the most frequently. The main source of errors lay in unrecognised names of stations. As mentioned earlier, the morphological analyser doesn't recognise many of local names. The database of MWE can partially solve this problem. However, some of names remain ambiguous. The second most frequent domain was the domain of cultural events. Most of errors appeared again during multiword expression analysis. The situation here is more complicated as the database of e. g. film titles is automatically updated. When asking questions on name–days, the frequent error appeared when a familiar variant of a name (e. g. William–Bill or Michaela–Míša) was used.

Only few errors appeared in answering queries on currency exchange rates, caused by missing currency.

The average time of response was less than 3 seconds excluding the time of downloading data from a particular web server. However, even difficult queries (i. e. uncovered, corrupt or erroneous) did not take more than 20 seconds to process.

---

[10]General queries: questions like *What can you do?*, *What can you help me with?*, *How do I ask you?*

[11]Other questions addressing usually non-covered domain or just conversational sentences like: *What's date today?*, *Say me something.*

8. COMPARISON WITH THE GENERAL QA SCHEMA

In Section 2 we described the general architecture of QA systems. *Document collection* contains one or more web server addresses for each domain. Actually, each web server consists of many web pages often generated dynamically. Based on the question analysis the right address   the right web server – is chosen. This step is the first step of the *candidate document selection*. All the web pages have been analysed in advance because the structure of these web pages is known in advance. In the second step of the candidate document selection *UIO* fills one or more forms exploiting the knowledge obtained in the process of the question analysis. These forms are then sent to the particular web server. As a result, several web pages are returned. Semantic analysis of these pages is now performed and the result is saved in XML notation. The *answer is extracted* from the XML code that matches with the semantic representation of the question. Then this answer is transformed to the *response* which is in the form understandable to the user. This response is then displayed to the user.

In *UIO* a module for context processing has been implemented. This module consists of two submodules, a module for processing *user model* and and a *module for dialog context*. User model is very simple in the current version of the system and is based on the form filled in the moment of a user registration. This model contains a user preferences and information about e. g. location of the *UIO* user. The dialog context module as well the model for processing user model is used when a question does not contain all the necessary information. In the case of the dialog context it may happen when the user sets a part of information in the previous question(s). *UIO* is able, to some extent, to exploit context analysis to complete the parameter set for the extracting algorithm.

9. COMPARISON WITH OTHER SYSTEMS

We tried to adapt techniques used in *START* system. However, T-expressions are not convenient for inflectional languages like Czech mainly because of the free order of words in Czech sentences. On the other hand, adapting *UIO* to other languages seems to be easy. Adding a new domain to *UIO* is also much easier than to *START*.

Most of approaches for communication in Czech like TIBAQ expects that questions are syntactically correct. This approach is not too convenient for our goal when a user is allowed to ask incomplete questions.

In the recent MFF UK project mentioned in Section 3 two modules for query analysis work separately and they do not exchange information. E. g. if the *Trans* module successfully analyse a subject part of a query but than fails to finish the analysis, *NLQ3* starts from scratch. In *UIO* all results of parsing are exploited and combined with the method of keywords.

We also compared *UIO* with *START, AskJeeves, AnswerBus* and *Google*. All queries for which *UIO* has found a correct answer have been first lemmatized. We checked first 10 pages found by a particular QA system.

None of the systems tested was not able to found an answer on train and bus

connections. It was expected because the information pages are generated dynamically. However, it was surprising that no link to any of information systems was not found. The same situation appeared for question on currency exchange rates. If a query was translated into English *AnswerBus* found a page with link to Czech train timetable.

AskJeeves was only partially successful for the domain of cultural events. It was necessary to introduce some keywords (e.g. a film title, a name of town) in the query. Then *AskJeeves* found links to a page that contained a useful link. *Google* partially succeeded if the query contained a name of cinema. The most frequent question like *Where is Matrix Reloaded on?* was not successfully answered with any system.

10. CONCLUSION AND FUTURE WORK

We described *UIO*, the multi-domain question answering system for Czech that looks for answers on the web. Users can ask syntactically correct or incorrect questions or use only keywords or part of a sentence. A morphological analyser, a database of multiword expressions and a bottom–up parser are employed to analyse the question. The MWE database is automatically updated when a new item has been found on a web server when evaluating a query. *UIO* has an accuracy rate about 80%.

In this version of *UIO* the module for automatically updating the database of MWE allows only new items to be added to the database. We plan to extend it to enable removing out–of–date items, e. g. movies not played any more.

In future we plan to use the shallow parser for Czech [28] that has been developed at NLP Lab FI MU for a better analysis of the question as well as for a more sophisticated analysis of the web pages. Combining the shallow parser with learning [29] is a challenge that awaits us.

REFERENCES

[1] D. E. Appelt and D. J. Israel: Introduction to information extraction technology. In: Proc. 16th Internat. Joint Conference on Artificial Intelligence (IJCAI-99) Tutorial, Stockholm 1999.

[2] M. Aretoulaki, F. Gallwitz, S. Harbeck, I. Ipšič, J. Ivanecký, V. Matoušek, H. Niemann, E. Nöth, and N. Pavešič: Sqel: A multilingual and multifunctional dialogue system. In: Proc. 5th Internat. Conference on Spoken Language Processing (ICSLP '98), Sydney 1998, pp. 2883–2996.

[3] D. Bauer, F. Segond, and A. Zaenen: Enriching an sgml-tagged Dictionary for Machine-aided Comprehension. Technical Report No. MLTT-011, Rank Xerox Research Centre 1994.

[4] E. Bried, F. Segond, and G. Valetto: Formal description of multiword lexemes with the finite-state formalism idarex. In: Proc. 16th Internat. Conference on Computational Linguistic. Morgan Kaufmann, San Francisco, CA 1996.

[5] S. Buchholz and W. Daelemans: Complex answers: a case study using a www question answering system. Natur. Language Engrg. 7 (2001), 4, 301–323.

[6] C. Clarke, G. Cormack, D. Kisman, and T. Lynam: Question answering by passage selection (multitext experiments for trec-9). In: Proc. Ninth Text Retrieval Conference (TREC-9), NIST Special Publication 2000, p. 673.

[7] N. Dufour: A database for computerized multi-word unit recognition. In: Proc. ISP-3, Stuttgart 1998.

[8] J. Hajič: Kodas a simple method of natural language interface to a database. Explizite Beschreibung der Sprache und automatiche Textbearbeitung 6 (1984), Charles University, Prague.

[9] J. Hajič: Nalcom: A multilevel nl-interface. Explizite Beschreibung der Sprache und automatiche Textbearbeitung 15 (1988), Charles University, Prague.

[10] E. Hajičová, J. Borota, J. Hajič, M. Hnáková, V. Kuboň, K. Oliva, and J. Panevová: Text-and-inference based approach to question answering. Theoret. Comput. Linguistic 3 (1995).

[11] L. Hirschman and R. Gaizauskas: Natural language question answering: The view from here. Natur. Language Engrg. 7 (2001), 275–300.

[12] P. Jirků and J. Hajič: Inferencing and search for an answer in tibaq. In: Proc. Ninth Internat. Conference on Computational Linguistics (E. Hajičová, ed.), Charles University, Prague 1982.

[13] B. Katz: From sentence processing to information access on the world wide web. In: Proc. AAAI Spring Symposium on Natural Language, Processing for the World Wide Web, Stanford Univesity, Stanford 1997.

[14] S. Klaas: Parsing Schemata: A Framework for Specification and Analysis of Parsing Algorithm. Springer Verlag, Berlin 1996.

[15] V. Matoušek: Simplified processing of elliptic and anaphoric utterances in a train timetable information retrieval dialogue system. In: Proc. Third Internat. Conference TSD 2000 (P. Sojka, I. Kopeček, and K. Pala, eds., Lecture Notes in Computer Science 1902), Springer–Verlag, Berlin 2001, p. 0399.

[16] D. Maynard, H. Cunningham, K. Bontcheva, and M. Dimitrov: Adapting a robust multi-genre NE system for automatic content extraction. In: 10th Internat. Conference, AIMSA 2002, Varna (Lecture Notes in Artificial Intelligence 2443), Springer–Verlag, Berlin 2002, pp. 264–273.

[17] D. Milward and J. Thomas: From information retrieval to information extraction. In: ACL 2000 Workshop on Recent Advances in Natural Language Processing and Information Retrieval 2000.

[18] D. Moldavan, S. Harabagiu, M. Pasca, R. Mihalcea, R. Goodrum, R. Girji, and V. Rus: Lasso: A tool for surfing the answer net. In: Proc. Eight Text Retrieval Conference (TREC-8), NIST Special Publication 1999.

[19] R. Mouček and K. Taušer: Dialogue system for city for city information centre. In: Proc. 6th World MultConference on Systemics, Cybernetics and Informatics SCI 2002, Orlando 2001, pp. 536–567.

[20] J. Prager and E. Browni: One search engine or two for question-answering. In: Proc. Ninth Text Retrieval Conference (TREC-9), NIST Special Publication 2000, p. 235.

[21] P. Sgall: Natural language understanding and the perspectives of question answeing. In: Proc. Ninth Internat. Conference on Computational Linguistics (E. Hajičová, ed.), Charles University, Prague 1982.

[22] S. Scott and R. Gaizauskas: University of sheffield trec-9 q&a system. In: Proc. Ninth Text Retrieval Conference (TREC-9), NIST Special Publication 2000.

[23] R. Sedláček and P. Smrž: A new Czech morphological analyser ajka. In: Proc. Fourth Internat. Conference TSD 2001 (P. Sojka, I. Kopeček, and K. Pala, eds., Lecture Notes in Computer Science 2166), Springer–Verlag, Berlin 2001, pp. 100–107.

[24] R. Sriharii and W. Li: Information extraction supported question answering. In: Proc. Eight Text Retrieval Conference (TREC-8), NIST Special Publication 1999.

[25] L. Svoboda: UIO, a dialog system for question answering. In: Proc. Znalosti 2003 Workshop (V. Svátek, ed.), 2003.

[26] M. Tomita: Efficient Parsing for Natural Language. Kluwer, Dordrecht 1986.

[27] D. Zhang and W. S. Lee: A web-based question answering system. In: Proc. SMA Annual Symposium 2003, NUS, Singapore 2003.

[28] E. Žáčková: Partial Parsing for Czech. Ph.D. Thesis, Masaryk University, 2002.

[29] E. Žáčková, M. Nepil, and L. Popelínský: Automatic tagging of compound verb groups in Czech corpora. In: Text, Speech and Dialogue: Proc. TSD'2000 Workshop (P. Sojka, I. Kopeček, and K. Pala, eds., Lecture Notes in Computer Science 1902), Springer Verlag, Berlin 2000, p. 0115.

Lukáš Svoboda and Luboš Popelínský, NLP Lab & KD Lab, Faculty of Informatics, Masaryk University in Brno, Botanická 68a, 602 00 Brno. Czech Republic.
e-mails: luks, popel@fi.muni.cz