Corpus-based Pattern Induction for a Knowledge-based Question Answering Approach

Philipp Cimiano
Institute AIFB
University of Karlsruhe
cimiano@aifb.uni-karlsruhe.de

Michael Erdmann
ontoprise GmbH
erdmann@ontoprise.de

Günter Ladwig
Institute AIFB
University of Karlsruhe
gula@aifb.uni-karlsruhe.de

Abstract

In this paper, we present an approach which, given a knowledge base and an appropriate text corpus, automatically induces patterns which can be used to query the knowledge base. We do not only consider simple questions but text problems consisting of several sentences. Answers to complex text problems are determined by an inference process which computes the answer on the basis of the background knowledge in the knowledge base (KB) as well as the textual description of the problem. The question formulated in natural language thus needs to be translated into appropriate KB structures and queries. Our approach to translating the natural language question uses an underlying corpus and the knowledge base to derive meaningful and relevant patterns which can then be used to process the questions and capture their meaning with respect to the underlying knowledge base. We apply our approach to Advanced Placement (AP) questions in three areas: physics, chemistry and biology. We report results of a first evaluation of the translation procedure for the three domains.

1 Introduction

During the lifetime of Aristotle (384 - 322 BC), it was possible for a person to be acquainted with most of the scientific knowledge available at the time, even across multiple disciplines. Aristotle himself was for example adept in physics, poetry, biology, zoology, logic, rhetoric, politics, government and ethics. Nowadays, given the tremendous amount of knowledge available in every scientific discipline, it is typically unfeasible to have deeper knowledge in more than a few areas. Thus, it would be really desirable to have an electronic assistant to answer all our scientific questions. Creating such a *'Digital Aristotle'* is the aim of the HALO project[1]. Its goal is to develop an intelligent information system containing important knowledge of different disciplines and able to answer people's questions about chemistry, physics, biology and other scientific disciplines. Key problems to be solved towards creating such an intelligent system are, on the one hand, to capture all the relevant knowledge, and, on the other, to develop a question answering system capable of exploiting this knowledge to answer users' questions. In this paper we deal with the second problem and present a corpus-based pattern induction approach which generates patterns in order to translate text problems formulated in natural language into appropriate assertions and queries with respect to the knowledge base. The approach is unique in several ways. On the one hand, it is not restricted to processing simple questions consisting of merely one sentence, but is able to process complex textual problems consisting of a short text describing the relevant background - typically consisting of a few sentences, and a specific question. Thus, it is crucial to capture the meaning of the whole textual description as it needs to be taken into account as background knowledge to provide the actual answer to the question. Such textual problems are found in the Advanced Placement (AP) test, which is a standardized US-wide test for high school students to obtain some credit for the introductory courses in college.
On the other hand, the approach does not need any manual customization as the patterns are learned automatically. The approach exploits an underlying knowledge base and a corresponding corpus to generate patterns which can then be used to query the knowledge base. Such an approach has the advantage that it requires no manual effort to customize the system to a specific knowledge base. Further, such an approach is expected to be reasonably robust as it does not need to understand the meaning of the textual question, but only find a relevant pattern for each sentence, which is a much easier task. Of course, such an automatic approach precludes that all relevant patterns are found. One aim of the research presented in this paper has been to determine to which extent the relevant patterns can be found on the basis of an automatic pattern induction approach.

[1] http://www.projecthalo.com/

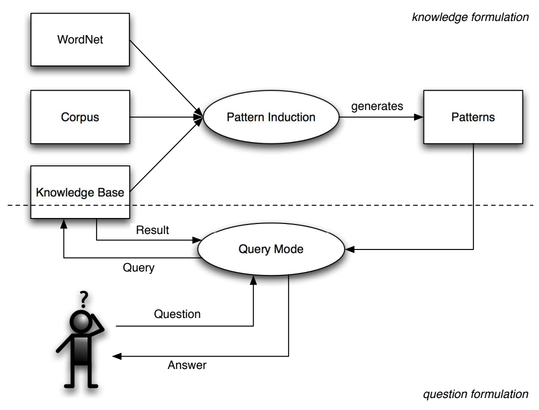

Figure 1. Overview of the approach

We provide an evaluation with respect to AP questions for physics, chemistry and biology. The corresponding knowledge bases have been engineered by domain experts without any background in knowledge representation in a first controlled experiment. These knowledge bases have an intermediary character as those used for the final evaluation of the HALO system will be engineered just in time. It is important to emphasize that such a setting creates especially severe conditions for any knowledge-based question answering system as there is actually no time to customize the system to the final knowledge base used. The approach developed thus has to meet the requirement of being applicable 'on the fly' to any knowledge base without manual support in customization. Our approach has been developed having this target in mind and in fact only requires a corresponding text corpus but no manual tuning.
The paper is structured as follows: in Section 2, we present our approach to pattern induction and query translation as well as the evaluation on AP questions from physics, chemistry and biology in Section 3. Finally, we discuss some related work in Section 4 and conclude in Section 5.

2 Pattern-based approach

The task of our approach is to process the textual description of the problem in question and translate it into corresponding assertions and queries formulated with respect to the knowledge base. The question is then actually answered by the underlying inference engine, the OntoBroker system in our case [6].

The pattern-based approach to query translation consists of two modes: the *pattern induction mode* and the *question answering mode*. In the pattern induction mode, patterns expressing certain relations in the knowledge base are generated on the basis of an underlying corpus. These patterns are then scored and form the input to the question answering component. The question answering component is responsible for, given a certain input sentence of the question, selecting the best matching patterns with highest score, presenting them to a user to choose the right pattern. Figure 1 shows an overview of the workflow. As an example of an AP question for physics, which will be discussed as an illustrating example throughout the remainder of the paper, consider the following:

You drive with a car along a straight road with a constant velocity of 60km/h for 30 minutes and afterwards you walk with 4km/h another 30 minutes in the same direction to the next gasoline station because your car runs out of gasoline. Calculate the average velocity.

Obviously, such a question is quite complex to process. Instead of processing these AP questions in their full complexity, we impose certain restrictions on the questions which will be discussed later in Section 3. In what follows, we describe in more detail the pattern induction and question answering modes.

2.1 Pattern Induction Mode

The patterns that our approach induces are essentially predicate-argument (P/A) structures for verbs and nouns. These patterns also specify the corresponding attributes in the knowledge base and how the arguments of the P/A structure map to these. Basically, these patterns thus realize a syntax-semantics interface. In order to produce appropriate patterns, the following steps are applied:

1. derive *partial frames* from the knowledge base which correspond to information units as used in natural language,
2. parse the corpus (with MiniPar [11])
3. extract predicate-argument (P/A) structures for verbs and nouns,
4. find appropriate P/A structures from the corpus which correspond to partial frames derived above,
5. test and score all the possible mappings between arguments of the P/A structure and the attributes in the partial frame, and
6. select the highest scoring mapping.
7. Finally, store the top 5 scoring patterns for each partial frame for use in the question answering mode.

The aim of the pattern induction algorithm is thus to assign up to 5 relevant patterns to each partial frame such that the corresponding pattern can be spotted in the text problems in question answering mode and thus mapped to the appropriate partial frames in the knowledge base.

Hereby, frames are defined in an object-oriented fashion, i.e. they are a set of relevant attributes for a certain object. Take for example the object *motion*. A motion has typically a *start point* and *end point*, a *duration*, a *speed*, an *acceleration*, etc. as attributes. The corresponding frame would look as follows (in F-Logic [6] syntax):

```
motion[start_point    => location;
       end_point      => location;
       moving_object => object
       start_time     => time;
       end_time       => time;
       speed          => speed;
       duration       => time_interval;
       acceleration   => acceleration
      ].
```

Typically, in natural language sentences, the information packaging unit does not correspond to the complete frames as represented in knowledge bases, i.e. the attributes of a frame are very rarely expressed all at once within one sentence. Therefore, as *partial frames* we consider combinations of up to three attributes for each frame. Thus, if a frame has n attributes and we want to consider all partial frames up to a length of k, we need to consider $\binom{n}{k}$ combinations for each frame. In our approach, we restrict ourselves to partial frames consisting of 1, 2 or 3 attributes, respectively. Hereby, partial frames of size 1 will be mapped to noun phrases, while partial frames of sizes 2 and 3 will be mapped to verb phrases consisting of a transitive verb, verbs with prepositional complements, etc. The noun phrase constructions we consider are the following: NP_c has NP_{attr} of X and NP_{attr} of NP_c is X. Both expressions are used to specify the value (X) of the attribute $attr$ of the entity c. Hereby, the heads of the NP's are expected to match the name of the label of the concept and attribute, respectively. After all the partial frames have been generated for a certain frame, the corpus is consulted to find possible realizations of these. The underlying corpus is parsed using the dependency parser MiniPar [11], developed by Dekang Lin. From the output of the parser, we extract verbs together with their subcategorized arguments as well as the above structures for noun phrases. This is a fairly standard procedure and thus not described here in more detail. Consider the following examples sentence: *The car moves with 60km/h for 30 minutes.* After processing MiniPar's output, the following verb-argument structure is generated (where words are already lemmatized): move(subj: car, with: velocity(60km/h), for: time(30min.)[2]

Now let us consider a partial frame of length 3 for *motion*:

```
motion[ moving_object => object;
        speed         => speed;
        duration      => time_interval
      ].
```

[2]In order to recognize temporal expressions, velocities, etc. we have implemented a set of regular expressions for this purpose.

In what follows, we will refer to the type of the frame as *frame type*, i.e. *motion* for the partial frame above. For a partial frame *pf*, $pf.type$ will thus denote the frame type (*motion* in the above example frame), $pf_{attr}.name$ will denote the name of attribute *attr*, while $pf_{attr}.type$ will denote the type of each attribute, e.g. *object* for *moving_object*, *time_interval* for *duration*, etc.

Assuming that most frame types/labels will be actually nouns, which is actually the case for the knowledge bases we have considered, we resort to WordNet [7] to find the corresponding verbal form, i.e. *move* for *motion* in our example. However, in general we can not expect to always find P/A structures in the corpus exactly for the verbal form derived from the frame label. Therefore, we first of all split the concept label into tokens, removing closed-class words such as articles, determiners or prepositions, and find the corresponding verb form in WordNet for each token, adding it to a set S with score 3. To this set S, the following words are added with the indicated scores:

1. direct synonyms from WordNet for each verb (with the same score as the original word),
2. hyponyms in WordNet for each verb with a score of 1, whereby the score is updated accordingly in case the hyponym is already in the set,
3. hypernyms with score 1 only if they are not already in the set, and
4. so called *DMS-synonyms* which were manually introduced by the domain experts modeling the knowledge bases with score 1, whereby as in the case of the hyponyms the score is updated accordingly if the word is already in the set.

The above scores are then normalized to the interval [0,1]. The above procedure for adding synonyms is clearly biased towards finding specializations rather than generalizations. In fact, when adding hypernyms we adopt a conservative strategy, adding the hypernym only if it was not already there. The idea is to avoid that hypernyms get too high scores.

As an example, consider the frame with the type *'motion of a falling object'*, which is tokenized into *motion*, *falling* and *object*. After looking up the corresponding verbs in WordNet we start with the set S:

{(*move*, 3.0), (*fall*, 3.0), (*objectify*, 3.0)}

Then we add the synonym *travel* of *move*:

{(*move*, 3.0), (*travel*, 3.0), (*fall*, 3.0), (*objectify*, 3.0)}

as well as the hyponyms *fall* and *fly* of *move*:

{(*move*, 3.0), (*travel*, 3.0), (*fall*, 4.0), (*objectify*, 3.0), (*fly*, 1.0)}

Then, *move* and *travel* as well as *modify* are added as hypernyms of *fall* and *objectify*, respectively:

{(*move*, 3.0), (*travel*, 3.0), (*fall*, 4.0), (*objectify*, 3.0), (*fly*, 1.0), (*modify*, 1.0)}

Finally, the DMS-synoym *fall* is added:

$\{$(*move*, 3.0), (*travel*, 3.0), (*fall*, 5.0), (*objectify*, 3.0), (*fly*, 1.0), (*modify*, 1.0)$\}$

After sorting and normalizing, we get the result:

$\{$(*fall*, 1.0), (*move*, 0.6), (*travel*, 0.6), (*objectify*, 0.6), (*fly*, 0.2), (*modify*, 0.2)$\}$

This example clearly illustrates that the above procedure is biased towards specialization. In fact, our aim is to associate very specific verbs such as *fall* with a given partial frame instead of resorting to more general verbs such as *move*. The reason for this is that when the verb *fall* is used in a question, we would like the patterns corresponding to a *motion of a falling object* to score high, while we would like to avoid that such patterns score high for the more general verb *move*.

Further, we would then search for all predicate-argument structures for the verbal form of the frame object name and the synonyms derived with the procedure described above, for instance finding the above predicate-argument structure for *move*.

Now, given a partial frame and a matching predicate-argument structure, we need to determine the highest-scoring mapping between attributes of the partial frame and arguments of the predicate-argument structure. Here, we consider all mapping possibilities. Let pf be a partial frame with n arguments and pa be a matching predicate-argument structure with k arguments. Let the set M contain mappings of positions in pf to positions in pa. The cardinality of M is then $\frac{n!}{(n-k)!}$ and thus contains all possibilities of injectively mapping arguments in pf to arguments in pa.

The procedure in Algorithm 1 then selects the best scoring match. It loops over all pairs of positions mapped to each other, calculating the score for each mapping and returning the mapping $m \in M$ yielding the maximum score. The score is calculated by multiplying the argument scores for each argument pair, whereby the argument scores are computed by calculating the average Levenshtein distance [10] between the name of the attribute ($pf_{arg}.name$) as well as the type of the attribute ($pf_{arg}.type$) on the one hand and each word occurring in the corpus at the corresponding argument position of the P/A structure on the other hand, i.e. for all $w \in pa_{arg}$. Hereby, the Levenshtein similarity $LD_{syn}(a, b)$ is calculated by tokenizing a and determining the maximum of the following formula which takes into account synonyms of the tokens of a as specified by WordNet, i.e. $LD_{syn}(a, b) := max_{s \in firstSynset(t), t \in tokens(a)} 1 - \frac{LD(s,b)}{max(s.length, b.length)}$ where $LD(a, b)$ is the standard Levenshtein distance between a and b. The tokenization (and removal of stopwords) of a is necessary as it can be a complex type such as *time interval*. ”Time”, ”interval” and their synonyms in the first WordNet sense are thus considered separately when calculating the Levenshtein distance, finally returning the maximum of the above formula. The Levenshtein distance is used here simply to ensure robustness with respect to spelling variants or even misspellings in the corpus.

Algorithm 1 Algorithm for selecting the most relevant pattern for a partial frame

```
getBestMapping(Partial Frame pf, P/A Structure pa, Mappings M)
{
  bestMapping := nil;
  bestScore := 0;
  foreach $m$ in M
  {
    score = 1;
    foreach pair $(pf_{attr}, pa_{arg})$
    {
      argScore = 0;
      foreach $w \in pa_{arg}$
      {
        argScore = argScore +
          $\frac{LD_{syn}(pf_{attr}.name,w)+LD_{syn}(pf_{attr}.type,w)}{2}$
      }
      score = score * $\frac{argScore}{|pa_{arg}|}$
    }
    if (score > bestScore)
    {
      bestScore = score;
      bestMapping = $m$;
    }
  }
return (bestMapping,bestScore)
}
```

Overall, the score of a pattern is the product of the value of the verb in the synonym set S as well as the score as calculated by Algorithm 1. For our *motion* example, assuming that the best selected mapping is $\{(moving\_object, subj), (speed, with), (duration, for)\}$, we would have found an appropriate pattern which would map our example sentence into the following KB assertion:

```
motion[ moving_object -> car1;
        speed         -> ``60''(km/h);
        duration      ->  ``20''(min.)
]
```

where *car1* is a constant introduced for an existentially quantified car, i.e. its skolem function. The top 5 best scoring patterns for each partial frame are then stored in an XML representation and can then be loaded into the pattern base of the question answering component, which is described below.

2.2 Question Answering Mode

The question answering component takes as input an AP question and tries to find the most appropriate pattern for each sentence. The input sentence is also parsed using

Algorithm 2 Algorithm calculating the match between a given pattern given a P/A structure found in the input sentence.

```
calculateMatch(P/A Structure input, P/A Structure pattern)
{
score = 0;
forall $arg \in args(input)$
    if ($arg \in args(pattern)$)
        score = score + $LD_{syn}$(pattern.arg.type,input.arg.word)
return $score/|args(input)|$
}
```

MiniPar and corresponding predicate-argument structures are extracted. Then, a pattern which matches the predicate-argument structure is searched in the pattern base using Algorithm 2 to find and score the best matching pattern.

In general, the procedure applied in question answering mode consists of the following steps:

1. The user enters the textual problem, which typically consists of several sentences and a final question.
2. The sentences are parsed with MiniPar (as in step 2 in Section 2.1).
3. The P/A structure in the input sentence is determined.
4. On the basis of the P/A structure, relevant patterns are found.
5. Patterns are presented to the user, who selects an appropriate frame type and specific pattern, thus also specifying the partial frame to which the input is mapped.
6. The attribute slots in the partial frame are filled on the basis of the input sentence, i.e. the partial frame is instantiated.
7. An F-Logic query is computed on the basis of the instantiated frame.
8. The F-Logic query is sent to the OntoBroker system.

Thus, the degree of match needs to be calculated for each P/A input structure and each pattern in the pattern base. The patterns can also be seen essentially as P/A structures with the exception that they also specify the mapping to the knowledge base and thus the argument positions are typed. The corresponding algorithm to determine the degree of match between two P/A structures is given in Alg. 2. The algorithm basically loops over the arguments of the input P/A structure, checking whether the pattern P/A has also the argument in question, calculating the Levenshtein similarity LD_{syn} between the word occurring at the argument position in the input P/A structure (input.arg.word) and the type of the argument in the pattern P/A structure (pattern.arg.type). Synonyms of pattern.arg.type are hereby considered in the calculation of the Levenshtein similarity.

In general, the question answering component will find all matching patterns and rank them according to their matching score as computed by Algorithm 1. The system can then either be applied in automatic fashion, selecting the top scoring pattern automatically or in a semi-automatic mode by presenting the ranked list of patterns to a user, grouped by frame types. This is important as several patterns for different frame types can be found for a given input sentence. For example, considering the sentence *The car moves with 6km/h for 30 minutes.*, *'moves'* can in principle refer to a *trajectory motion*, an *acceleration motion*, a *motion of a falling object* etc. The type of motion can not be determined in many cases just from the input question. In our system, the user is thus first asked to select a relevant frame type from a list. Once he has selected the type of frame, he then gets presented a ranked list of patterns together with the attributes they map to and can select one pattern from a second list. The system has been evaluated in this interactive mode. The evaluation is described in the following section.

3 Evaluation

The evaluation of the system presented was conducted with respect to three different knowledge base modeled in the context of the HALO project. However, we do not report results from the end-to-end evaluation as we are not concerned with the query answering itself, but present a more direct evaluation of our pattern-generation approach. Obviously, due to the large amount of patterns generated, it is not feasible to evaluate each pattern by hand. Therefore, we decided to evaluate the approach in terms of the times that an appropriate pattern was found for an input sentence. In order to evaluate our ranking, we also report the average position of the pattern selected by the user in the ranked list. Overall, the approach described in the section above was developed by using a selected number of sentences from the three domains: physics, chemistry and biology. It was validated with unseen examples entered into the system by our evaluators in a slightly restricted English syntax.

3.1 Experimental Settings

The knowledge bases used in our experiments were modeled for three areas: physics, chemistry and biology on the basis of the text books [9], [3] and [2], respectively. Table 1 gives some statistics for the corpora derived from the textbooks, such as the total number of sentences, the number of sentences for which a parse tree was produced as well as the number of P/A structures extracted for verb phrases (VP) and noun phrases (NP). In particular, we observe that P/A structures were only extracted for between 21.66% - 30% of the sentences. Table 2 gives some figures about the underlying KBs, giving the number of frames, average number of attributes per frame as well as the average number of inverse relations per frame. In fact, though we have not mentioned this in the description of the approach in Section 2 in order

	physics	chemistry	biology
Sentences	14,706	24,650	73,893
Parsed sentences	11,357 (77.23%)	19,588 (79.5%)	49,395 (66.85%)
P-A structures	4,416 (30%)	6,619 (26.86%)	16,004 (21.66%)
VP-structures	4,179	6,270	15,101
NP-structures	237	349	903
Avg. Occ. per P/A	2.37	2.82	3.01

Table 1. Corpus statistics

	physics	chemistry	biology
Frames	29	72	252
Avg. Attributes per Frame	7.72	9.74	25.70
Avg. Inv. Rel. / Frame	1.97	4.46	24.76
Partial Frames	6,540	19,530	1,071,828

Table 2. KB statistics

to simplify the presentation, we also consider inverse relations, that is, attributes of some other frame pointing to the frame in question, as attributes of the latter for the purposes of the pattern induction process. For example, the vehicle involved in a motion is actually expressed via an inverse relation in the physics knowledge base as follows:

```
vehicle[hasMotion -> motion]
```

and does not actually form part of the *motion* frame. Therefore, it is crucial to also consider this inverse relations in the pattern induction process. This is illustrated for example by our running example involving the verb *move*, which takes as subject the vehicle, which is modeled through an inverse relation in the knowledge base.

For the experiments reported in this paper, we simplified the problem by imposing certain restrictions on the language used to describe the textual problem, thus creating certain controlled settings. In particular, we specifically asked the test subjects taking part in the evaluation to use the following sentence types: i) "The X of Y is Z.", ii) "Y has a X of Z." as well as iii) arbitrary grammatical and full sentences consisting of one main verb. The last condition implies in particular that no relative clauses or other subordinated clauses were allowed. Further, the use of parenthesis (e.g. for specifying quantities as in '*A car (5km/h) drives from A to B.*') was disallowed. Finally, we explicitly required the specification of a subject. Obviously, these restrictions simplify the problem substantially. It is important to mention that still under these more controlled settings the problem is far from trivial and highly challenging. Our example at the beginning of Section 2 was for example reformulated by one of the test persons as follows:

A car moves with a constant velocity of 60 km/h. The motion has a duration of 30 minutes. I move with 4 km/h. The motion has a duration of 30 minutes. What is the average velocity of the motion?

	physics	chemistry	biology
Patterns	31,429	30,617	170,129
Avg. Score	0.09	0.04	0.06
Occurrences per Patterns	1.50	1.64	2.34

Table 3. Pattern Statistics

	physics	chemistry	biology
AP text problems	11	15	8
Sentences	48	37	38
Correctly processed sentences	29	10	1
Percentage correctly processed	60.41%	27.02%	2.63%
Average position in raking			
Average pos. of selected pattern	1.24	1.2	1.00
Problems			
P/A not recognized (sentences)	2	2	6
no pattern found (sentences)	0	4	7
no frame type selected by user	7	17	24
no pattern selected by user	10	4	0

Table 4. Results of the evaluation

3.2 Results

Table 3 gives the number of patterns generated for each of the knowledge bases as well as the average score of each pattern. Due to the fact that for the biology domain the number of partial frames was so high, only partial frames with up to 2 attributes were considered.

Note that the average score of the patterns is quite low, which is due to the fact that we multiply scores which are lower than 1, i.e. the score for the mapping as calculated by Algorithm 1 as well as the value of the corresponding synonym. Table 4 shows the results of the evaluation. It shows the number of AP text problems considered as well as the total number of sentences for all the problems. It also shows the total number of correctly processed sentences, i.e. those sentences for which our test persons could find a relevant pattern leading to an appropriate instantiation of the corresponding partial frame. For the physics KB, 60,41% of the sentences are processed correctly, while for the chemistry and biology domain the results are worse with 27,02% and 2,63%, respectively. Further, the table also shows the average position in the ranking list of the frame and pattern selected by the user. In general, the ranking of the patterns seems reasonable for all domains, as the pattern selected was at position 1 on average. The table also shows a more detailed analysis of the involved problems. There are several problems affecting the approach:

- the parser does not produce any output,
- no P/A is recognized in the parser output,

- no corresponding pattern is in the pattern base,
- the pattern is in the library but is not found, or
- the argument mapping for the patterns is not appropriate.

When having a look at the figures in Table 1, we observe that the percentage of parsed sentences (66.85%-77.23%) and P/A structures extracted (21.66%-30%) does not vary substantially enough to explain the high differences in the results of our approach for the different domains. So the reason why our approach performs much worse for the chemistry and biology domains is either that no appropriate patterns are found for the partial frames, the argument mapping is not appropriate or the patterns are in the library but simply not found at query time. In essence, all these cases boil down to lexical mismatches, i.e. cases in which the labels used in the knowledge base and words in the corpus substantially differ. It is certainly also possible (though not very likely) that the language used by our test persons in the questions and the one of the corpus differ substantially for the biology and chemistry domains. This issue will definitely require more investigation. Further, it would also be important to quantitatively analyze how many of the failures to find an appropriate pattern are in fact due to i) parse errors, ii) P/A structure extraction errors, or iii) deficiencies in our pattern induction algorithm.

3.3 Discussion

The results of the physics knowledge base are certainly quite satisfactory, while the results for the chemistry and biology knowledge bases are disappointing. For the chemistry and biology knowledge bases, appropriate patterns were actually not found in many cases. However, there were also problematic cases in the physics domain. Consider for example the following sentences for which the system failed to find an appropriate pattern:

- A metallic ball has a weight of 0.1 kg (in this case there is no attribute *weight* in the KB but only an attribute *mass* which is not found due to the fact that there is no relation between mass and weight in WordNet)
- The initial time of a motion is 8 hours. (in this case no pattern is found because the attribute *initial time* is incorrectly typed as *length*)
- An airplane moves in horizontal direction with a constant velocity of 50km/h. (in this case no pattern is derived as no directions are modeled in the knowledge base)

In the biology and chemistry domains, the main problem was that the verbs and nouns used in the corpus do not correspond to the names of the frames and attributes modeled in the knowledge bases. Even WordNet does not help much here, because it does not contain rich biological or chemical terminology. In the physics domain this was not such a big problem due to the fact that more common terminology is used. We thus conclude that much richer lexical resources are needed for the chemistry and biology domains. In general, our indirect evaluation has also limits as it does not allow to conclude what the main problems are. The sheer size of partial frames in the biology domain is for sure also making the selection of appropriate patterns difficult. Further investigation on other domains is certainly needed to clarify the strengths and limits of our approach.

4 Related Work

There has been certainly a lot of work on natural language interfaces to knowledge or databases in the 70s and 80s. For an overview of these older systems, the interested reader is referred to the overviews in [5] and [1]. Due to space limitations we will in fact only discuss more recent systems in this section.

Thompson et al. [14] present a system in which control strategies for a corresponding parser are automatically derived from training data relying on techniques from inductive logic programming (ILP). The system is evaluated on two domains (jobs and geography), and achieves very decent accuracy levels between 25 - 70% for the geography domain and between 80 - 90% accuracy for the job domain, depending on the amount of training data used.

The PRECISE system [13] has focused on the reliability of NLIs and presented a system which is formally proved to be 100% precise, given an appropriate domain-specific lexicon. In fact, PRECISE requires no additional customization nor domain-specific knowledge other than an appropriate lexicon. The precision of PRECISE of almost 100% on real data is certainly impressive. However, the domains on which PRECISE has been evaluated (job postings, geography) are probably less challenging than the scientific domains we have considered.

The recently presented AquaLog system [12] essentially transforms the natural language question into a triple-representation and then relies on similarity computations to map the triples to appropriate relations defined in the ontology. Some basic support for disambiguation is provided in this way. As our approach, AquaLog does not require any manual tuning of the system to a specific domain. As AquaLog is also based on syntactic similarity metrics, we assume that it would have similar problems as our system in case the KB and user vocabulary differ substantially.

The approach in the QUETAL system [8] implements the mapping from a question to a query via three intermediary stages: i) construction of a Robust Minimal Recursion Semantics (RMRS) representation, ii) mapping to FrameNet types and roles as well as iii) construction of so-called *proto-queries*. The approach implements a hybrid technique to interleave shallow and deep processing. An evaluation of the system shows that it achieves similar precision rates as our system, i.e. 74.1%. Customization is achieved through hand-written rewriting rules transforming FrameNet-like structures to domain-specific structures as provided by the domain ontology.

A similar corpus-based approach to deriving linguistic patterns as presented in this paper was developed in the context of the ORAKEL system (compare [4]). However, the approach was not evaluated with users, but only for its ability to find appropriate linguistic patterns for different common sense knowledge bases about wines, beers, university organization etc.

5 Conclusion

We have presented a pattern-based approach to translating complex textual problems into appropriate KB assertions and queries needed to answer the question as an inference process. The crucial step in our approach is a pattern induction component which, given a knowledge base and a corresponding corpus, induces patterns which are basically predicate-argument structures specifying how the arguments map to frames specified in the knowledge base. In fact, the patterns thus realize the syntax-semantics interface as once an appropriate pattern is found in question answering mode, the appropriate KB structures can be instantiated. We have evaluated our approach with three knowledge bases in three domains: physics, chemistry and biology. While our approach performs well on the physics domain, it yields much worse results on the chemistry and biology domains. According to our conclusions, this is mainly due to lexical mismatches between the way information is modeled in the knowledge base and expressed in the corpus, which prevents our pattern induction component from finding appropriate patterns. In general, our work differs from other related work in that it tackles highly challenging and complex domains such as physics, chemistry and biology. An important problem which we have not yet addressed and thus remains for future work is the treatment of discourse phenomena, in particular the resolution of (anaphoric or bridging) references. The resolution of such references is nevertheless crucial in order to capture the content of the textual problem correctly. Such an approach to reference resolution will indeed need to be implemented before our approach can be evaluated end-to-end, i.e. from the input text problem to the final answer.

Acknowledgments This research has been supported by the HALO project, funded by Vulcan Inc. as well as by the X-Media project, funded by the European Union under grant IST-FP6-026978. We thank Martin Weindel, Kai Kühn and Thomas Schweitzer for participating in the evaluation. Thanks to our colleague Johanna Völker for providing the MiniPar API. The views expressed in this paper solely represent those of the authors and not necessarily those of Vulcan Inc.

References

[1] I. Androutsopoulos, G. Ritchie, and P. Thanisch. Natural language interfaces to databases–an introduction. *Journal of Language Engineering*, 1(1):29–81, 1995.

[2] T. Brown, H. LeMay, B. Bursten, and J. Burdge. *Chemistry: The Central Science*. Prentice Hall, 9th edition, 2002.

[3] N. Campbell and J. Reece. *Biology*. Prentice Hall, 6th edition, 2002.

[4] P. Cimiano. ORAKEL: A Natural Language Interface to an F-Logic Knowledge Base. In *Proceedings of the 9th International Conference on Applications of Natural Language to Information Systems (NLDB)*, pages 401–406, 2004.

[5] A. Copestake and K. S. Jones. Natural language interfaces to databases. *Knowledge Engineering Review*, 1989. Special Issue on the Applications of Natural Language Processing Techniques.

[6] S. Decker, M. Erdmann, D. Fensel, and R. Studer. Ontobroker: Ontology Based Access to Distributed and Semi-Structured Information. In *Database Semantics: Semantic Issues in Multimedia Systems*, pages 351–369. Kluwer, 1999.

[7] C. Fellbaum. *WordNet, an electronic lexical database*. MIT Press, 1998.

[8] A. Frank, H.-U. Krieger, F. Xu, H. Uszkoreit, B. Crysmann, B. Jörg, and U. Schäfer. Question answering from structured knowledge sources. *Journal of Applied Logic, Special Issue on Questions and Answers: Theoretical and Applied Perspectives*, 2006. to appear.

[9] D. Giancoli. *Physics: Principles with Applications*. Prentice Hall, 6th edition, 2004.

[10] V. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966.

[11] D. Lin. Dependency-based evaluation of MINPAR. In *Proceedings of the LREC Workshop on the Evaluation of Parsing Systems*, pages 112–120, 1998.

[12] V. Lopez and E. Motta. Aqualog: An ontology-portable question answering system for the semantic web. In *Proceedings of the International Conference on Natural Language for Information Systems (NLDB)*, pages 89–102, 2004.

[13] A. Popescu, O. Etzioni, and H. Kautz. Towards a theory of natural language interfaces to databases. In *Proceedings of the International Conference on Intelligent User Interfaces (IUI'03)*, pages 149–157, 2003.

[14] C. Thompson, R. Mooney, and L. Tang. Learning to parse natural language database queries into logical form. In *Proceedings of the Workshop on Automata Induction, Grammatical Inference and Language Acquisition*, 1997.