Processing Negation in NL Interfaces to Knowledge Bases

Svetla Boytcheva[1], Albena Strupchanska[2], and Galia Angelova[2]

[1] Department of Information Technologies, FMI, Sofia University,
5 J. Bauchier Blvd., 1164 Sofia, Bulgaria,
svetla@fmi.uni-sofia.bg

[2] Bulgarian Academy of Sciences, Linguistic Modelling Lab, CLPP,
25A Acad. G. Bonchev Str., 1113 Sofia, Bulgaria,
{galia,albena}@lml.bas.bg

Abstract. This paper deals with Natural Language (NL) question-answering to knowledge bases (KB). It considers the usual conceptual graphs (CG) approach for NL semantic interpretation by joins of canonical graphs and compares it to the computational linguistics approach for NL question-answering based on logical forms. After these theoretical considerations, the paper presents a system for querying a KB of CG in the domain of finances. It uses controlled English and processes large classes of negative questions. Internally the negation is interpreted as a replacement of the negated type by its siblings from the type hierarchy. The answer is found by KB projection, generalized and presented in NL in a rather summarized form, without a detailed enumeration of types. Thus the paper presents an interface for NL understanding and original techniques for application of CG operations (projection and generalization) as means for obtaining a more "natural" answer to the user's negative questions.

1 Introduction

During the last decades the challenging idea of building NL interfaces for man-machine communication has led to numerous approaches, different research prototypes and industrial systems. The early examples of NL interfaces to databases appeared in the 70's, with `LUNAR` as an English querying system, when important problems like weight of English interrogative pronouns (*each, every*) and scope of question quantifiers were investigated for the first time. The most recent systems are already integrated in industrial applications, like `EnglishQuery` which is embedded in `MicroSoft SQL Server 2000` and advertised as the top NL interface to databases. It deals with plurals and successfully processes large classes of positive questions with relatively complex but syntactically unambiguous sentence structure. However, `EnglishQuery` does not demonstrate particular intelligence in treating quantifiers and behaves inadequately even in simple cases of negative sentences. So we pessimistically conclude that the NL processing field did not progress significantly with the treatment of negation and quantifiers during the last twenty years.

Negation is a rather problematic language phenomenon which might be unclear or ambiguously interpreted by human beings. From computational perspective, processing negative sentences is almost impossible even for such relatively simple systems for NL understanding, like NL interfaces to databases, where the input consists of quite short text (one interrogative sentence). There are at least two reasons why processing negation is so difficult:

- The system has to determine which is the negated sentence phrase (i.e. to decide about the scope); moreover negation scope often overlaps with the scope of the quantifiers and tense operators;
- Negating a sentence phrase means semantic negation of event, object, quality, location, manner, context and so on. The semantic analyzer has to interpret the negation in the proper way, in principle by application of some prover. This interpretation requires:
  (a) detailed knowledge models of the closed world, and
  (b) non-trivial AI proving techniques which are either undecidable or quite ineffective to apply; the implementation of such provers requires too much efforts and they still do not exist practically.

In general, negation is often avoided in AI considerations. Most of the theoretical CG research focuses on positive CG which are well-studied in the recent literature [3]. There are only few works considering the problem of negation in CG. For instance, [10] defines a logical approach for treatment of negated graphs taken as a whole. The paper [7] considers operations with negation of types inside a CG, using lattices over the syntactic structure of graphs. CG are represented in conjunctive normal form (CNF). Some elements of our considerations are similar to the approach in [7] on an abstract level.

This paper presents an original "ontological" treatment of negation in simple NL queries to a KB of CG. Asking a question requires translation of the query to a CG, so in section 2 we briefly overview related research. Section 3 presents the question-answering system that we have implemented. Sections 4, 5 and 6 contain correspondingly an example, current evaluation and the conclusion.

2 Translating NL to Conceptual Graphs

"Understanding" NL sentences by translating them to CG is a popular CG application. CG are a form of logic; they only represent the propositional content of a sentence, syntactic features of the original are lost [20]. For example consider:

```
(PAST)->[SITUATION: [CAT: #]<-(AGNT)<-[EAT]->(PTNT)->[FISH: #]].
```

This graph means: (1) *The cat ate the fish.* (2) *The fish was eaten by the cat.* (3) *The (past) eating of the fish by the cat.* So the general idea is that understanding NL (i.e. extracting CG from text) means to process the syntax, to find the semantics "behind" it and to encode this semantics as a CG.

Most of the systems translating NL to CG work sentence by sentence and translate sentences to isolated graphs. Nearly no attempts were made to resolve some NL references in neighbor input sentences and to translate them to identities and coreference links within the obtained KB of graphs.

2.1 Early implementations in the 80's

The first algorithm [17] proposes semantic parsing of NL sentences by the so-called *compositionality* principle: joins of canonical graphs (describing lexical semantics of encountered words) are performed according to allowed syntactic rules; the results give both the syntax tree and the joined graph as semantic representation. Another source is [19] which discusses in details an earlier implementation [18].

A rather early implementation is presented in [8] where the authors state that *"The join operation plays the same role as the lambda evaluation used in the logical form approach"*. [8] claims that although very similar to the classical logical-form approach on an abstract level, semantic parsing by CG offers natural possibilities to define semantic filters by canonical graphs, which are easy and flexible word-centered descriptions. There is an intuitive parallel between the join operation and the derivation in context-free grammars: since the join is applied to one concept (in two graphs) and the context-free composition works with rules with one variable to the left-side, the join defines some sort of "context-free calculus" over graphs.

`DANTE` [21] is the first serious effort to encode lexical semantics in a systematic way. `DANTE` performs question-answering in Italian from the KB. Its early version works with about 850 extended word-sense definitions. `DANTE` keeps separately morphological, syntactic and semantic knowledge and performs real morphological analysis. Its grammar covers about 80% of the syntactic phenomena in the analyzed corpus. The syntax analysis is performed independently of the semantic interpretation, so the input to the semantic module is a set of syntax trees for the given sentence. Finally, each sentence is translated into CG using a semantic lexicon. `DANTE` semantic analysis is similar to Sowa's proposal [17] on an abstract level.

2.2 Prototypes dealing with controlled languages in the 90's

There are few research prototypes in real domains with practical importance. Two of them deal with medical texts, which due to their telegraphic style are very successfully treated by semantic interpretation using CG, since the semantic structure is more important than the syntactic one for the understanding of the utterance.

`METEXA` [16] analyzes radiological reports and answers questions about their semantic representation. The system lexicon was built using a corpus of 1500 radiological texts containing about 8000 different wordforms with about 120000 occurrences. `METEXA` is the first system for German. It has a fullform lexicon, where the compound German terms are defined, and performs syntactic analysis of the input phrases. The semantic analysis works in parallel with the syntactic one. The implementation is based on resolution similarly to [8].

RECIT [15] analyzes sentences from medical texts in French, English and German and stores the sentence meanings into CG. RECIT works on free-text patient documents in digestive surgery. A system-specific elaboration is the so called "proximity processing", which aims at the decomposition of the sentence into meaningful fragments, given a partial interpretation of the sentence. Thus RECIT analyzer is a modular system, composed of two parts which are necessary to separate the language-independent from the language-specific processing. RECIT analyzer is not based on a formal grammar but on a set of sequential semantically-driven procedures which incrementally build a meaningful structure.

More recent prototypes with certain lexical and structural limitations are BEELINE [13] (which processes limited vocabulary in the world of robots and translates imperative phrases and simple sentences to CG); Knowledge extractor [4, 5](which relies on a knowledge engineer to highlight NL fragments from the input text and translates them to CG); CG Mars Lander [9, 11] (which skips unknown words from input sentences and thus defines a NL sublanguage).

2.3 NL translated to CG via logical form

It is well-known that there are many different ways to combine syntactic and semantic information in a parser [1]. The claim that *"the join operation plays the same role as the lambda evaluation used in the logical form approach"* in [8] was probably correct in 1986 but meanwhile the computational linguistics made a big progress during the last decades. The present theory of logical grammars allows uniform treatment of quantifiers and logical operators. There are well-studied techniques for (partial) resolution of scope and weight. Coherent discourse of several sentences is processed successfully, with relevant resolution of coreferences by bounding variables in the logical forms of neighbor sentences. In contrast, CG-related prototypes made no attempts to process several generalized quantifiers in one sentence. So in this paper we choose the following approach:

(i) to parse the input query using a bottom-up parser for syntactic analysis of controlled English;
(ii) during the parsing process, to build semi-compositionally the logical form of the input; the negation is treated as a logical operator;
(iii) to decompose the logical form to positive and negated disjuncts, and to represent it in a Prenex Conjunctive Normal Form (PCNF) [12];
(iv) to translate the disjuncts to CG and
(v) to process them independently (see section 3).

3 Question-answering

The presented question-answering system deals with controlled English queries in the financial domain. The system can process all main *wh-* types of questions (except *why*-questions) with or without negation. The main question-answering steps are shown in Fig. 1.

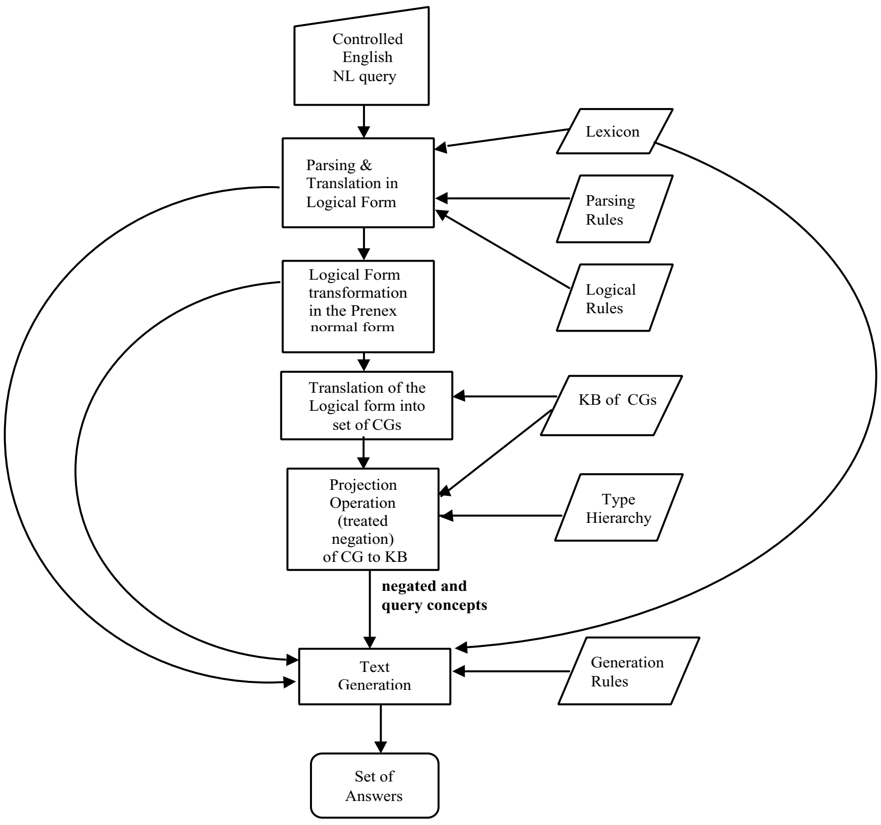
Controlled English NL query
Parsing & Translation in Logical Form
Lexicon
Parsing Rules
Logical Rules
Logical Form transformation in the Prenex normal form
Translation of the Logical form into set of CGs
KB of CGs
Projection Operation (treated negation) of CG to KB
Type Hierarchy
negated and query concepts
Text Generation
Generation Rules
Set of Answers

Fig. 1. Processing NL queries to KB of CG

Some of the queries might have more than one correct answer and sometimes most of them refer to similar information. In order to obtain a more "natural" answer to a user's request, the system generates a generalized answer.

3.1 Recognition and interpretation of negations in user queries

An original bottom-up parser was developed for the purposes of syntactic analysis. The parser uses the following resources: *(i)* a lexicon with common words and financial terms with corresponding morphotactic rules, *(ii)* a set of negative "key-words" like pronouns, particles, etc. which indicate the negation in the input query and *(iii)* a grammar designed to cover more than 80% of the syntactic phenomena from a corpus of queries in controlled English.

During the parsing process the system semicompositionally produces the intermediate logical form of the input utterance. For example, the query

(1) *"Who does not buy bonds?"*

will be translated to the following logical form:

(1') $\neg(\forall(X, bond(X)\&buy(Y)\&\theta(Y, agnt, Univ)\&\theta(Y, obj, X)))$

The object to be extracted is marked by a special variable "Univ". The θ-terms correspond to the thematic roles of the verb. If the question has a negation, as in query (1), negation scope is considered ambiguous at this intermediate processing stage. To solve this problem we first set negation scope to the whole sentence and after that we construct all possible logical forms with localization of the negated phrases. Note that the approach illustrated by examples in this paper is adequate for input NL questions containing no disjunctions and implications. So the present assumption is that the intermediate logical form (1') contains conjunctions only. But obviously there are no theoretical limitations to generalize the considerations and process input logical forms containing for instance disjunctions; for simplicity we focus on input NL queries without logical operators.

Location of the negated sentence phrases: The logical form (1') is transformed to PCNF which is better than the original one, since the negation scope is maximally localized to the phrases, that are presented as a set of conjuncts. Each conjunct is one unambiguous meaning of the sentence and can be treated separately from the remaining conjuncts in the formulae. All conjuncts give all possible meanings of the sentence.
For example, the PCNF of query (1) is the disjunction of the following three logical forms:

(2.1) $\exists(X, \neg bond(X)\&buy(Y)\&\theta(Y, agnt, Univ)\&\theta(Y, obj, X))$
(2.2) $\exists(X, bond(X)\&\neg buy(Y)\&\theta(Y, agnt, Univ)\&\theta(Y, obj, X))$
(2.3) $\exists(X, \neg bond(X)\&\neg buy(Y)\&\theta(Y, agnt, Univ)\&\theta(Y, obj, X))$

Informally these items can be translated in a more "natural" language as:

(2.1a) *Who does buy something different from bonds?*
(2.2a) *Who is doing other actions with bonds except buying them?*
(2.3a) *Who is doing other actions except buying with something different from bonds?*

The PCNF consists of disjunctions of logical forms, containing two major types of literals: concepts and relations between them. Only concepts can be negated in our interpretation. Unfortunately, the number of possible interpretations of a sentence with negation grows factorially with the number of its concepts [14].

Transforming the conjuncts of the query PCNF to a set of CG: Both types of literals in the logical form are translated to CG components as concepts and relations between them respectively (similarly to the techniques proposed in [17]). The concept addressed by the query is translated as a universally quantified instance with morphological and syntactic features derived from the parsing results (i.e. tense of verbs and number of nouns are encoded as referents). So, in the forthcoming projection this concept will be "projected" to all KB concepts that have conforming referents. At this point every negated concept is replaced by its siblings from the type hierarchy. Most generally, every concept corresponding to a verb is replaced by its "antonym or complementary events"; every object is replaced by the so-called *restricted universally quantified* concept (see further details in section 3.2). At the end of these transformations, we obtain a set (Ω) of CG which covers all possible readings of the input NL query.

3.2 Searching the KB

Extraction of CG answers is performed by projection. Each graph from Ω is projected to the KB.

Processing of queries without negation or any modalities: The query PCNF consists of one conjunct only and it is translated to one conceptual graph which has a concept of a "Univ" type. This graph is projected to the KB. All resulting CG are found and a set of concepts (which were projected to the concept of "Univ" type) is retrieved from these CG. These concepts are the most generalized concepts that appear in graphs returned by the projection of the query to the KB. In order to avoid some repetitions and pre-specializations of the answer, the set of concepts is generalized by looking for all the concepts in this set that have common immediate parent. If the result set contains all children of a type than we replace then by this type.

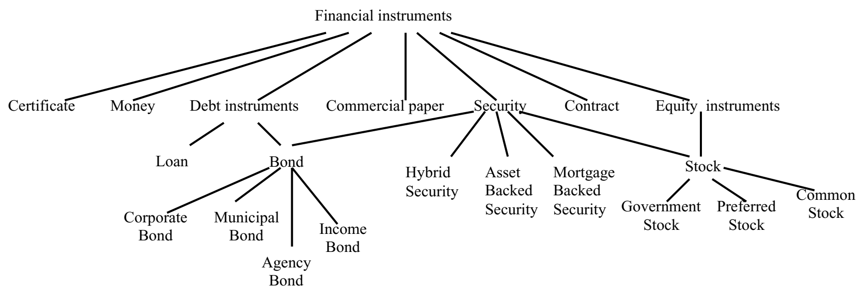

Fig. 2. A part of the type hierarchy of financial instruments

For example the question

(3) *"What is traded on the open market?"*

will be translated to the following logical form:

(3') $\forall(Z, open\_market(Z)\&trade(Y)\&\theta(Y, obj, Univ)\&\theta(Y, loc, Z))$

For the present KB, all concepts retrieved from the projection results are: {*bond, preferred stock, municipal bond, common stock, contract, government stock*}. Then the generalized concepts (according to the part of the type hierarchy represented in Fig.2) are {*bond, contract, stock*}. The generated NL answer will contain these three objects only.

Processing of queries with negation: Translating PCNF with negation to CG depends on:

- Negation of the event in the input query (i.e. negation of the main verb). Events are ordered in the KB hierarchy. So the first step in processing the negation is to find all the siblings of the negated event. Furthermore a new graph for each sibling is produced. All of these graphs have one unknown (universally quantified) concept and they are projected to the knowledge base in order to receive all possible candidates that satisfy this query.
  Example: the query graphs constructed as "negation" of
  (2.2) $\exists(X, bond(X)\&\neg buy(Y)\&\theta(Y, agnt, Univ)\&\theta(Y, obj, X))$
  are:

  ```
  (2.2.1) [BOND: {*}]<-(OBJ)<-[SELL]->(AGNT)->[UNIV: *].
  (2.2.2) [BOND: {*}]<-(OBJ)<-[TRADE]->(AGNT)->[UNIV: *].
  ```

  In the type hierarchy SELL and TRADE are sibling concepts of BUY.
- Negation of some objects and characteristics: The negated concept in this case is presented as a *restricted universally quantified* concept. *Restricted* here means that it can be projected to all concept types belonging to the set S(*nc*), where *nc* is the *negated* concept and:
  $S(nc) = (Sib(nc) \bigcup SonSib(nc)) \setminus Son(nc)$
  $Sib(x) = \{y|sibling(x, y)\}$
  $Son(x) = \{y|parent(x, y)\}$
  $SonSib(x) = \bigcup_{y \in Sib(x)} Son(y)$

  Example: for the concept *Stock* at Fig.2:
  $S(Stock) = (Sib(Stock) \bigcup SonSib(Stock)) \setminus Son(Stock) =$
  {Bond, Hybrid Security, Asset Backed Security, Mortgage Backed Security} $\bigcup${Corporate Bond, Municipal Bond, Agency Bond, Income Bond}$\setminus$
  {Government Stock, Preferred Stock, Common Stock}

Then the query graph constructed as "negation" of
(2.1) $\exists(X, \neg bond(X) \& buy(Y) \& \theta(Y, agnt, Univ) \& \theta(Y, obj, X))$
is:

```
(2.1.1). [Univ: disj{S}]<-(OBJ)<-[BUY]->(AGNT)->[UNIV: *].
```

and the query graphs constructed as "negation" of
(2.3) $\exists(X, \neg bond(X) \& \neg buy(Y) \& \theta(Y, agnt, Univ) \& \theta(Y, obj, X))$
are:

```
(2.3.1). [Univ: disj{S}]<-(OBJ)<-[SELL]->(AGNT)->[UNIV: *].
(2.3.2). [Univ: disj{S}]<-(OBJ)<-[TRADE]->(AGNT)->[UNIV: *].
```

In this way, for query (1) we obtain:
Ω={(2.1.1), (2.2.1), (2.2.2), (2.3.1), (2.3.2)}

Retrieving the answer by KB projection: The projection returns all CG that fulfill the query graph. However, this result may not be convenient for the generation of a NL answer. So we additionally process these CG in order to obtain the corresponding pairs (query concept/KB concept).
For example, projection of the graphs in Ω to the KB returns answers as follows:

- For conjunct (2.1), *Who does buy "non-bonds"?* the answer is:

```
(4.1) [univ\pension_fund, not_bond\government_stock]
```

In other words, "Univ" appears to be "Pension Fund" and "non-bonds" to be "Government Stocks". The answer is generated from the CG in the KB:

```
(5.1) [BUY]-(AGNT)->[PENSION_FUND: #]
           -(OBJ)->[GOVERNMENT_STOCK: {*}]
           -(LOC)->[PRIMARY_MARKET: #].
```

- For conjunct (2.2), *Who "does not buy" bonds?* there are two answers:

```
(4.2.1) [not_buy/sell,univ/demander]
(4.2.2) [not_buy/trade,univ/company,bond/corporate_bond]
```

The answer is generated from the CG in the KB correspondingly:

```
(5.2.1) [SELL]-(AGNT)->[DEMANDER: #]
              -(OBJ)->[BOND: {*}]
              -(LOC)->[PRIMARY_MARKET: #].

(5.2.2) [TRADE]-(AGNT)->[COMPANY: #]
               -(OBJ)->[CORPORATE_BOND: {*}]
               -(CHAR)->[NEWLY_ISSUED: #].
```

- For conjunct (2.3), *Who "does not buy" "non-bonds"?* there are three answers:

```
(4.3.1) [not_buy/sell,univ/broker,not_bond/stock]
(4.3.2) [not_buy/sell,univ/broker,not_bond/hybrid_security]
(4.3.3) [not_buy/trade,univ/stockholder,not_bond/hybrid_security]
```

This answer is generated from the following CG in the KB:

```
(5.3.1) [SELL]-(AGNT)->[BROKER: #]
              -(OBJ)->[STOCK: {*}]->(CHAR)->[MATURITY]-
                                    (ATTR)->[SHORT_TERM].

(5.3.2) [SELL]-(AGNT)->[BROKER: #]
              -(OBJ)->[HYBRID_SECURITY: {*}]
              -(LOC)->[NYSE].

(5.3.3) [TRADE]-(AGNT)->[STOCKHOLDER: #]
               -(OBJ)->[HYBRID_SECURITY: {*}]
               -(LOC)->[STOCK_EXCHANGE].
```

Since the graphs (5.1), (5.2.1), (5.2.2), (5.3.1), (5.3.2) and (5.3.3) are unlikely to constitute coherent discourse, the strategy is to verbalize them as separate sentences.

3.3 Answers Generation

We replaced all negated phrases and questioned concepts in the logical form with positive results of the projection operation of CG to the KB. Now by backward operation we reconstruct the sentence from its logical form. The generation is simpler than the one presented earlier in [2] since we do not approach the discourse problems but rather produce NL answers containing lists of insulated sentences. In the NL generation we also use information from the lexicon and the fact that all answers are universally quantified statements, because of the specific domain. For query (1), the generated set of answers is:

```
Answer to (2.1): ['Pension funds buy government stocks.'],
Answer to (2.2): ['Demanders sell bonds.',
                  'Companies trade corporate bonds.'],
Answer to (2.3): ['Brokers sell stocks and hybrid securities.',
                  'Stockholders trade hybrid securities.']
```

When negating some objects or characteristics it is possible to receive more than one result for the query concept. In these cases the system tries to generalize them, if it is possible, in order to produce more "natural" answer. All of them are shown as answers to the user, since they cannot be further generalized. Note that "Brokers sell stocks" and "Brokers sell hybrid securities" are aggregated as one sentence, but STOCK and HYBRID SECURITY can not be generalized to SECURITY due to the following reasons: *(i)* the negated concept BOND is a child of SECURITY in the type hierarchy (Fig. 2) and *(ii)* the other children of SECURITY are missing.

4 Example with negated location

This section illustrates our question-answering approach for negations of other kinds of sentence phrases. Let us consider the query:

(6) *Who does not buy securities on the primary market?*

Logical form:

(6') $\neg(\forall(X, security(X) \& buy(Y) \& \theta(Y, agnt, Univ) \& \theta(Y, obj, X) \& \theta(Y, loc, Z) \& primary\_market(Z)))$

The PCNF is a disjunction of seven conjuncts:

1. $\forall(X, \neg primary\_market(X) \& \forall(Y, security(Y) \& buy(Z) \& \theta(Z, agnt, Univ) \& \theta(Z, obj, Y) \& \theta(Z, loc, X)))$;
2. $\forall(X, primary\_market(X) \& \exists(Y, \neg security(Y) \& buy(Z) \& \theta(Z, agnt, Univ) \& \theta(Z, obj, Y) \& \theta(Z, loc, X)))$;
3. $\forall(X, primary\_market(X) \& \exists(Y, security(Y) \& \neg buy(Z) \& \theta(Z, agnt, Univ) \& \theta(Z, obj, Y) \& \theta(Z, loc, X)))$;
4. $\forall(X, primary\_market(X) \& \exists(Y, \neg security(Y) \& \neg buy(Z) \& \theta(Z, agnt, Univ) \& \theta(Z, obj, Y) \& \theta(Z, loc, X)))$;
5. $\forall(X, \neg primary\_market(X) \& \exists(Y, \neg security(Y) \& buy(Z) \& \theta(Z, agnt, Univ) \& \theta(Z, obj, Y) \& \theta(Z, loc, X)))$;
6. $\forall(X, \neg primary\_market(X) \& \exists(Y, security(Y) \& \neg buy(Z) \& \theta(Z, agnt, Univ) \& \theta(Z, obj, Y) \& \theta(Z, loc, X)))$;
7. $\forall(X, \neg primary\_market(X) \& \exists(Y, \neg security(Y) \& \neg buy(Z) \& \theta(Z, agnt, Univ) \& \theta(Z, obj, Y) \& \theta(Z, loc, X)))$.

Projection result:

```
1. [],
2. [],
3. [[not_buy/sell,univ/underwriter],
   [not_buy/trade,univ/dealer]],
4. [],
5. [[univ/company,not_security/commercial_paper,
    not_primary_market/open_market]],
6. [[not_buy/sell,univ/corporation,
    not_primary_market/negotiated_market],
   [not_buy/trade,univ/company,security/corporate_bond,
    not_primary_market/open_market]],
7. [].
```

In this example items: 1, 2, 4 and 7 have no projections, because either there is no appropriate information about them in the CG KB or these questions make no sense. In items 3 and 6 there are more than one correct answer due to negation of the main action. The number of such answers depends on the number of complementary verbs of the negated verb - in our case the verb "buy" has two complementary verbs "trade" and "sell".

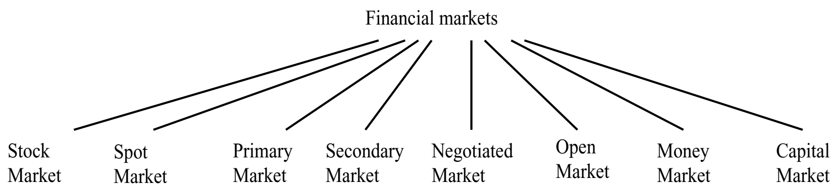

Fig. 3. A fragment of the type hierarchy of financial markets

In item 6 the negated concept "primary market" is projected to its sibling concepts "negotiated market" and "open market" in the type hierarchy (Fig. 3). The generated set of answers to (6) is:

```
1. []
2. []
3. ['Underwriters sell securities on the primary market.',
   'Dealers trade securities on the primary market.']
4. [],
5. ['Companies buy commercial papers on the open market.']
6. ['Corporations sell securities on the negotiated market.',
   'Companies trade corporate bonds on the open market.']
7. []
```

5 Evaluation

In general, question-answering systems are hard to evaluate, as there is no well-defined "correct answer". We cannot give accuracy measures and usually apply task-based evaluation, i.e. we evaluate whether the system helps the user to solve his/her particular problem. In this case, the implemented prototype supports knowledge acquisition process and provides friendly answers to queries about the available KB types and their hierarchical and factual connections. Note that this paper does not solve problems like "whether BUY is the negation or antonym of TRADE and/or SELL"; this question may look complicated for most human beings too. Rather, the paper presents the KB content as it is acquired and labeled by the knowledge engineer. In fact we verbalize the knowledge engineer insights in acquiring types and rely on the assumption that siblings encode different meanings, therefore one of them is always negation of the others in some sense.

The presented system is implemented in Sicstus Prolog and uses the following resources:

- Lexicon of approximately 500 words and grammar of 100 rules;
- Type hierarchy of about 150 concepts in the financial domain;
- KB of about 300 CG.

The system processes most types of *wh*-questions and questions requiring "true/false" answer. We describe in more details the processing of the first type of questions, because they are more difficult and interesting from research point of view. The completeness of the generated NL answers depends only of the completeness of the CG KB. The approach is practically suitable for simple questions because of the factorial complexity of the algorithm for negation interpretation.

Although there are no problems to implement "how many/much" questions, they are not realized in this version of the system. Such questions suppose accumulation of the answers and their processing can be reduced to counting the number of the answers found by the described algorithm, which was considered as relatively useless. The questions *how* and *why* require much more complex KB processing and are not covered in this paper.

6 Conclusion and Further work

The presented system is an example of handling simple questions with or without negation. At the same time it is clear that rather complex questions can be treated only if "restricted" English is turned to "formalized" English by further constraints. Despite the limitations, the system is very useful for verbalization of positive facts in the closed world of a restricted domain and provides rather effective interface for simple question-answering tasks.

At present we plan further development in the following directions: to enlarge the linguistic knowledge of the prototype (lexicon and parsing rules), and to develop a web-based user friendly system interface, to be integrated as a part of `CGWorld` tool [6]. Integration within `CGWorld` looks particularly important since our experience proves that the NL inference is very useful for knowledge engineers while conceptual graphs are acquired.

References

1. Allen, J. Natural Language Understanding, The Benjamin/Cummings Publishing Company, Inc., 1995
2. Angelova, G. and K. Bontcheva. DB-MAT: Knowledge Acquisition, Processing and NL Generation Using Conceptual Graphs. In: P. Eklund, G. Ellis, G. Mann (eds.), Proc. ICCS-1996, LNAI 1115, pp. 115 -129.
3. Chein, M., Mugnier, M. Positive Nested Conceptual Graphs. In: D. Lukose, H. Delugach, M. Keeler, L. Searle, J. Sowa (eds.), Conceptual Structures: Applications, Implementation and Theory, Proc. 5rd ICCS'97, August 1997, LNAI 1257, pp. 95-109.

4. Cyre, W. Knowledge Extractor: A Tool for Extracting Knowledge from Text. In Lukose, Delugach, Keeler, Searle and Sowa (Eds.). Proc. ICCS-97, Seattle, USA, LNAI 1257, pp. 607-610.
5. Cyre, W. Capture, Integration and Analysis of Digital System Requirements with Conceptual Graphs. IEEE Transactions on Knowledge and data Engineering, Vol. 9, No. 1, February 1997.
6. Dobrev, P., Strupchanska, A. and K. Toutanova, CGWorld-2001 - new features and new directions, ICCS 2001 Workshop, July 2001, Stanford University, USA http://www.cs.nmsu.edu/ hdp/CGTools/proceedings/papers/CGWorld.pdf
7. Esch, J., Levinson, R. An Implementation Model for Context and Negation in Conceptual Graphs. In: G. Ellis, R. Levinson, W. Rich, J. Sowa (eds.), Conceptual Structures: Applications, Implementation and Theory, Proc. 3rd ICCS'95, August 1995, LNAI 954, pp.247-262.
8. Fagrues, J., Landau, M.C., Dugourd, A. and L. Catach. Conceptual Graphs for Semantics and Knowledge Processing. In: IBM J. Res. and Develop. Vol. 30 (1), January 1986, pp. 70-79.
9. Fuchs, G. and R. Levinson. The CG Mars Lander. In Lukose, Delugach, Keeler, Searle and Sowa (Eds.). Proc. ICCS-97, Seattle, USA, LNAI 1257, pp. 611-614.
10. Kerdiles, G. Saying It with Pictures: a logical landscape of conceptual graphs, ILLC Dissertation Series DS-2001-09, Institute for Logic, Language and Computation, Universiteit van Amsterdam, 2001, pp. 183.
11. Levinson, R. Symmetry and the Computation of Conceptual Structures. In Ganter and Mineau (Eds.), Proc. ICCS-2000, Darmstadt, Germany, LNAI 1867, pp. 496-509.
12. J.W.Lloyd, Foundations of Logic Programming, Springer-Verlag Berlin Heidelberg, 1984.
13. Mann, G. Control of a Navigating, Rational Agent by Natural Language. PhD Thesis, School of Computer Science and Engineering, University of New South Wales, Sydney, 1996.
14. Poesio, M. Semantic Analysis, In Handbook of Natural Language Processing, Marcel Dekker, Inc., 2000, pp.93-122.
15. Rassinoux, A.-M., Baud, R. H. and J.-R. Scherrer. A Multilingual Analyser of Medical Texts. In: W. Tepfenhart, J. Dick, J. Sowa (eds.), Conceptual Structures: Current Practices. Proc. ICCS'94, LNAI 835, pp. 84-96.
16. Schroeder, M. Knowledge Based Analysis of Radiology Reports using Conceptual Graphs. In: H. Pfeiffer, T. Nagle (eds.), Conceptual Structures: Theory and Implementation, Proc. 7th Annual Workshop, July 1992, LNAI 754.
17. Sowa, J. Conceptual Structures: Information Processing in Mind and Machine. Addison-Wesley, Reading, MA, 1984.
18. Sowa, J. and E. Way. Implementing a Semantic Interpreter using Conceptual Graphs. IBM Journal R&D, Vol. 30 (1), 1986, pp. 57-69.
19. Sowa, J. Using a Lexicon of Canonical Graphs in a Semantic Interpreter. In: Martha Evens (Ed.), Relational Models of the Lexicon, Cambridge University Press, 1988, pp. 113-137.
20. Sowa, J. Towards the Expressive Power of Natural Language. In: J. Sowa (Ed.), Principles of Semantic Networks, Morgan Kaufmann Publishers, 1991, pp. 157-190.
21. Velardi, P., Pazienza, M. and M. De'Giovanetti. Conceptual Graphs for the Analysis and Generation of Sentences. In: IBM J. Res. and Develop. Vol. 32 (2),March 1988, pp. 251-267.