Determining Relevant Deep Web Sites by Query Context Identification

Zsolt T. Kardkovács[1] and Domonkos Tikk[1]

Budapest University of Technology and Economics,
Department of Telecommunications and Media Informatics,
{kardkovacs,tikk}@tmit.bme.hu

Abstract. Deep web search requires a transformation between search keywords and semantically described and well-formed data structures. We approached this problem in our "In the Web of Words" (WoW) project by allowing natural language sentence queries and by a context identification method that connects the queries and deep web sites via database information. In this paper we propose a novel SQL based approach that can identify the focus of input questions if the information is represented in a database. We propose a new relational database design technique called normalized natural database (NNDB) to capture the meaning of data structures. We show that a proper NNDB is a context database, and it can serve as the basis of context identification combining the template based techniques and the world model encoded in the database.

Key words: deep web search, context identification

1 Introduction

The online accessible information organized and stored in structured databases is the content of *deep web*. The content of such databases is presented to the user as dynamic web pages being created to answer user's query, and thus standard search engines can hardly index and find them [1, 2]. Therefore, searching on the Internet today can be compared to dragging a net across the surface of the ocean. While a great deal may be caught in the net, there is still a wealth of information that is deep, and missed. Internet content is considerably more diverse and the volume certainly much larger than commonly understood [2, 3, 4].

In the WoW project, our purpose was to create a prototypical search service that could integrate surface web and deep web search, and for the latter case, could allow users to formulate their search expressions in the form of Hungarian natural language questions. The overview of the system is depicted on Figure 1. The core of the system consists of the natural language interface, the context identifier and the deep web search engine (DWSE). These components transforms simple interrogative sentences of a given language into SQL queries in several steps (see [5]). The important issue in this processing chain is the determination of the context (topic) of the question, and databases related to the topic, which

is the topic of this paper. At the end of the search the user is directed the answer pages provided by the deep web sites.

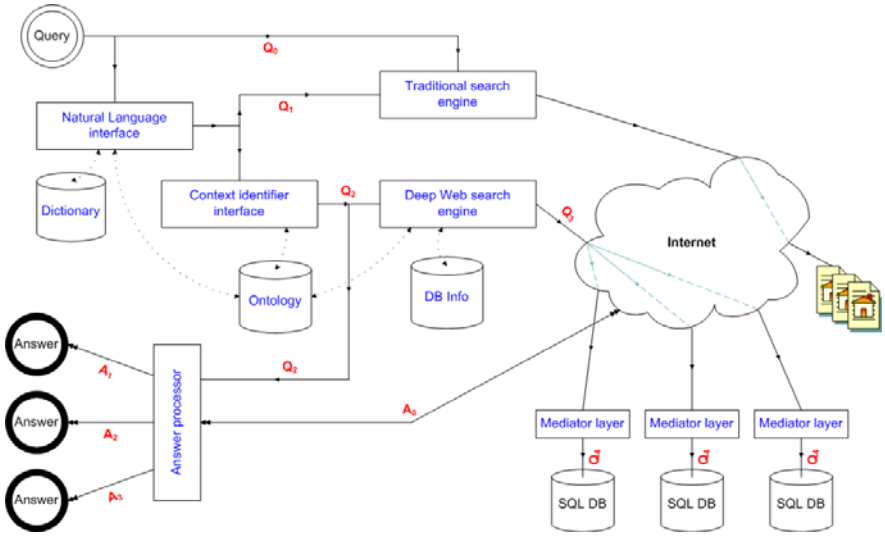

Fig. 1. The complete overview of the WoW system

The deep web search facility of WoW is restricted to the database content of contracted partners available online, i.e. no automatic exploration of deep web sites is performed. Therefore throughout the paper, we assume that schema information of deep web site are available.

The deep web search model of WoW is language independent, however its implementation for a specific application requires natural language processing (NLP) tools. In our experiments we implemented the model of WoW for Hungarian, but it can be easily adopted for other languages having the necessary NLP tools available. We have reported on this issue for Hungarian in our previous works [6, 7].

In this paper, we primarily focus on the context identification and the deep web search engine architecture, and particularly we present some important improvements made to these components of the original model in the WoW system [5]. For context identification, we propose a relational theory based mathematical foundation and we outline its implementation. Using this new approach, we can substitute the formerly used intermediate languages (ER-models, ontologies, or language Q_2 [5]) with standard SQL expressions. This change simplifies the model by omitting the transformation from the intermediate languages to SQL, which was formerly performed by the DWSE component. In order to make this change, we also exploit NLP tools (such as morphological and syntactic parsers, named entity recognizers) and natural language interfaces to databases.

In our proposed framework, the operation of DWSE exclusively relies on the result of context identification. Indeed, if there is available (meta)information on the data that is stored at the deep web site, and the internal representation language is SQL, then the DWSE engine can select the deep web sites relevant in answering the user query by determining the attributes and schemas appearing in the transformed SQL form of the query.

The paper is organized as follows. We briefly review related works on database theory and deep web search in Section 2. Section 3 introduces the most important concept of relational theory used in this article. Section 4 defines the new normalized natural database (NNDB) structure that stores language processing related information. In Section 5, we present a context identification algorithm based on NNDB. Section 6 discussed the role of DWSE in our framework, and Section 7 gives a few illustrative examples on context identification, and briefly presents some test results.

2 Related works

This paper concerns two different though partly connected research areas for which we summarize the literature separately.

Deep web search have been received attention from the late 90's [4, 3], but the first works only intended to characterize the scale of deep web from various aspects. As shown recently in [2], traditional crawl-and-index techniques may not be appropriate to provide a good coverage over the information content of the hidden and dynamic deep web site. In contrast to our cooperative model, another direction in discover the deep web is called the discover-and-forward access model. In this model, search engine having already discovered *query interface* of databases of the web would forward users automatically to the appropriate database to submit or refine the query. There are a few projects, such as MetaQuerier [8] and WISE-Integrator [9] which exploring this particular research direction. We remark that this solution also requires context identification to select the relevant databases for the user query. Our concept provide smaller coverage since it requires active cooperation from the deep web sites but ensures higher quality at answering, which can draw the attention of deep web sites in longer term. Another alterative method for the integration of deep web sites is highly accurate semantic matches between the attributes of the source query interfaces. However, due to the pervasive lack of data present at the query interface the matching accuracy is often questionable. WebIQ [10] solves this problem by discovering instances for interface attributes from the surface and deep web.

Traditional database management systems fail to capture the intentional semantics between data structure elements. In order to overcome this deficiency—which is a motivation of our semantic database approach—Sagiv's [11] and Honeyman's [12] works introduced the notion of intentional database in the early 80's which aimed at capturing the meaning of the represented relations in the database. Their path was followed by Chan end Mendelson [13] on the concept of separable databases. Their works intend to determine conditions on the

separability of schema structure of the database which enables that the modification of a particular relation does not affect other relations, and also eases the introduction of new schemas end relations into the database.

3 Preliminaries

Throughout the paper, we focus on relational databases and relational theory, although our results can be easily adopted to other well-known data models as well. We use the traditional conventions, terms and notations of relational data model theory.

- r denotes a relation; t denotes an element of a relation (aka n-tuple or record) t; R, S, T denote schemas; and A, B, C denote attributes.
- Capitals from the end of the alphabet (like X, Y, Z) denote attribute sets.
- The expression $r(R)$ denotes a relation on schema R.
- The expression $t[X]$ denotes the value of a record over the attribute set X.
- $t \in r$ means the record t is the element of relation r. $A \in R$ means that attribute A is an element of schema R. For simplicity, schemas are considered to be attribute sets.
- $X \rightarrow Y$ denotes a *functional dependency* defined on the schema $X \cup Y \subseteq R$. The functional dependency is true iff all records t_1, t_2 of any relation $r(R)$ over the schema R satisfy: $t_1[X] = t_2[X] \Rightarrow t_1[Y] = t_2[Y]$. If $X \subseteq R$ and $X \rightarrow R$ hold then X is a *superkey*. If additionally there exist no $X' \subset X$ for which $X' \rightarrow R$ is true then X is the *key* of schema R. The set of functional dependencies is denoted by $\mathbf{F}$.
- The closure of an attribute set X over a dependency set $\mathbf{F}$ is defined as the maximal set having the property $X^+(\mathbf{F}) = \{A | \mathbf{F} \models X \rightarrow A\}$.
- The closure of dependency set $\mathbf{F}$ is defined as the maximal set of elements $\mathbf{F}^+ = \{X \rightarrow Y | \mathbf{F} \models X \rightarrow Y\}$. The dependency set $\mathbf{F}$ is a *coverage*, if there exists no such dependency $X \rightarrow Y$ that satisfies
  1. $(\mathbf{F} \setminus \{X \rightarrow Y\})^+ = \mathbf{F}^+$;
  2. $\nexists X' \subset X$, for which $\mathbf{F} \models X' \rightarrow Y$;
  3. Y consists of a single attribute.
- Let $X \dashrightarrow Y$ denote the *inclusion dependency*, which is true iff the domain of X and the domain of Y are identical, and for all relations r_1, r_2 over schemas R_1 and R_2, where $X \subseteq R_1, Y \subseteq R_2$ satisfy: $r_1(X) \subseteq r_2(Y)$. The set of inclusion dependency is denoted with $\mathbf{I}$.
- Let the relational database be denoted with $\mathcal{DB} = <\mathbf{R}, \mathbf{r}, \Sigma>$, where $\mathbf{R}$ is the finite set of schemas in the database, $\mathbf{r}$ is the finite set of relations over $\mathbf{R}$[1], and $\Sigma = \mathbf{F} \cup \mathbf{I}$ is the finite set of (functional and inclusion) dependencies of the schemas of the database.
- We say that a database $\mathcal{DB}$ complies a dependency set Σ, denoted by $\mathcal{DB} \models \Sigma$, if $\forall r \in \mathbf{r}\ r \models \Sigma$ is fulfilled.

[1] Obviously, there is exactly one relation over a given schema.

Relational databases do not specify how to define sensible operations between relations. As a consequence, one can join two relations if data are in appropriate formats along any two attributes. For example, one can join tables about cities and books on attributes population and title, respectively. Therefore database management systems can not determine the intentional semantics between elements of the data structure. In order to capture the notion "self-descriptiveness" we introduce a new concept called semantic database.

Definition 1 (Semantic databases). *Let* $\mathcal{DB} = <\mathbf{R}, \mathbf{r}, \Sigma>$ *be a semantic database if for any attributes and schemas of* $\mathcal{DB}$ *, the following conditions hold regarding a reference function* $\lambda : \mathbf{A} \rightarrow \mathbf{A}$ *(*$\mathbf{A}$ *is the set of all attributes), and a binary (is-a) relation* $\Xi : \mathbf{R} \times \mathbf{R}$*:*

1. *for any* $S \in \mathbf{R}$ *schema with key* A*,* $\lambda(A) = A$*,*
2. *for any attribute* A *there is a schema* $S \in \mathbf{R}$ *and a simple key* $B \rightarrow S$ *such that* $\lambda(A) = B$*,*
3. $\Sigma \models A \dashrightarrow \lambda(A)$ *holds for any attribute* A*,*
4. $\Xi(R, S)$ *is true for any* $R, S \in \mathbf{R}$ *schemas if and only if* $\Sigma \models X \dashrightarrow Y$ *for some appropriate* $X \rightarrow R, Y \rightarrow S$ *attribute sets.*

For example: $\lambda(\mathsf{wife}) = \mathsf{name}$, $\lambda(\mathsf{name}) = \mathsf{name}$, and $\Xi(\mathsf{actor}, \mathsf{person})$. The definition extends the traditional relational database model by the reference function and an is-a relation in order to represent which attributes and schemas are related, and how.

Definition 2 (Valid relationship). *Let* $X \subseteq R_i, Y \subseteq R_j$ *be two attribute sets of the not necessarily distinct schemas* $R_i, R_j \in \mathbf{R}$ *in a database* $\mathcal{DB} = <\mathbf{R}, \mathbf{r}, \Sigma>$*. A valid relationship between* X *and* Y *exists if* $\Sigma \models \{X \dashrightarrow Y, Y \rightarrow R_j\}$*, and is denoted by* $\varepsilon(X, Y)$*.*

Proposition 1. $\varepsilon(A, \lambda(A))$ *for any attribute* A *of the semantic database* $\mathcal{DB} = <\mathbf{R}, \mathbf{r}, \Sigma, \lambda, \Xi>$*.*

Proposition 2. *Let* $\mathcal{DB} = <\mathbf{R}, \mathbf{r}, \Sigma, \lambda, \Xi>$ *be a semantic database. If* X, Y *are keys of schemas* $R, S \in \mathbf{R}$ *respectively that make the relation* $\Xi(R, S)$ *valid, then* $\Xi(R, S) \Rightarrow \varepsilon(X, Y)$*.*

4 Normalized Natural Databases

According to the traditional database design concept, schemas are partitioned until a normal form is reached to keep them clear of redundancy. It must be noted that the model being designed depicts a closed system from a well-defined perspective. In such cases a schema may still contain many attributes, which makes further decomposition necessary if a new view appears. Thus to provide portability the number of attributes must be minimized in order to avoid further decomposition.

Naturally, for an arbitrary decomposition a schema must contain one attribute as something must identify the entities in the relation. It is easy to show that some specific attributes without the key property may have to remain in the schema too. For example, a name used in a natural language to identify an object in the world is inseparable from the object, though it is not a key, because different objects may have the same name. Let us call such an attribute *natural key* introduced by [14, 15]. The natural key of a schema S is denoted by $\kappa(S)$.

In other words, values of natural keys are the (compound) nouns or noun phrases in sentences, which are exact, unambiguous identifiers of an entity of the real world in the given language and context. For example, $\kappa(\mathsf{book}) = \mathsf{title}$, or $\kappa(\mathsf{person}) = \mathsf{family\_name}$, but $\kappa(\mathsf{Book}) \neq \mathsf{author}$, though author is an attribute of the book schema.

Definition 3 ([15]). *A semantic database is called natural, if any of the following conditions are fulfilled:*

1. *If the reference function of the database is λ, then $\lambda(A)$ is a natural key for all attributes A.*
2. *If $\varepsilon(A, B)$ is true for some attributes A, B of the database, then B is a natural key.*
3. *Each key is a natural key.*

Proposition 3. *The three statements in the Definition 3 are equivalent.*

Proof. $1 \Leftrightarrow 3$: According to definition 1, $\lambda(A)$ is always a key. From statement 1 $\lambda(A)$ is a natural key also which implies 3. Conversely, because of statement 3, each key is a natural key, which also holds for $\lambda(A)$, thus implying 1.

$2 \Leftrightarrow 3$: According to the definition of $\varepsilon(A, B)$, B is always a key. Therefore statement 2 implies 3, since B is a natural key. Conversely, the natural key property holds for key B in $\varepsilon(A, B)$ from statement 3, because B is a key in $\varepsilon(A, B)$.

Definition 4 (Normalized natural database). *A natural database $\mathcal{DB} = <\mathbf{R}, \mathbf{r}, \mathbf{F} \cup \mathbf{I}, \lambda, \Xi>$ is normalized (abbreviated by NNDB) if the following statements hold (here $S, R \in \mathbf{R}$):*

1. *Any attribute name appears in a single schema.*
2. *Any R schema with a natural key contains only one element. Such schemas are called primary schemas.*
3. *Any R schema without natural keys contains at least two elements, and for all attributes $A \in R$ there is an $S \neq R$ schema such that $\lambda(A) = \kappa(S)$. Here R is termed a secondary schema.*
4. *For any two different R, S schemas $R \cap S = \emptyset$.*
5. *There are no distinct secondary R, S schemas such that $R^+(\mathbf{F}) \supseteq S$.*
6. *For any R secondary schema there are either no valid non-trivial dependencies on it or there is an embedded dependency $\mathbf{F} \models X \rightarrow A$ defined on it such that $\mathbf{F} \models X \rightarrow R$, and $\nexists X' : X' \subset X$ for which $\mathbf{F} \models X' \rightarrow A$ holds.*
7. *For any inclusion dependency $X \dashrightarrow Y$, Y is a natural key.*

Proposition 4. *Let* $\mathcal{DB} = <\mathbf{R}, \mathbf{r}, \mathbf{F} \cup \mathbf{I}, \lambda, \Xi>$ *be a NNDB. The function* λ *implies an equivalence relation on attributes of* $\mathcal{DB}$ *in the following way: the equivalence class of attribute* A *consists of those attributes* A_j *for which* $\lambda(A_j) = \lambda(A)$*. We denote by* $\|A\|$ *the equivalence class of the attribute* A*.*

Due to the lack of space we omit the proof here (see [16]).

5 Context Identification in NNDBs

All primary schemas have a single attribute: a natural key, and named entities are its instances. For all other attributes, only references must be stored, i.e. references to the attributes they are referring to in a role. A pragmatic property of NNDBs is that they can be easily described by a few metaschemas. That is, NNDBs are extensible and portable to various topic areas. Hence, the context identification procedure is very similar to the idea first proposed in [17]. However, the proposed implementation uses purely relational semantics, it can be decomposed by any database schemas, therefore there is no need for re-engineering techniques. The proposed implementation also solves the identification of the question focus.

The search model of WoW uses NLP tools, namely morphological and syntactic parsers, and named entity recognizer. In our implementation for Hungarian queries we used the morphological parser Hunmorph [18], and our own syntactic parser [6] and named entity recognizer [7].

The context identification algorithm (or the disambiguation) based on a relational NNDB (see Figure 2) is composed of the following steps:

- Determine entities and phrase structures in the interrogative sentence of the natural language. Use a state-of-the-art named entity recognizer to determine with high accuracy the idiomatic expressions, names, labels (i.e. dates, currencies, etc.)—commonly referred to as *entities*—,and a syntactic parser to obtain the beginnings and endings of the phrases.
- Map the identified entities into NNDB elements using the Dictionary and KnowledgeBase. The mapping is essential for handling multilingual issues and synonymity. Naturally, a single entity in the real world may have more than one name, e.g. the Virgin of Orleans, Joan of Arc, Jeanne d'Arc, Joan the Saint refer to the very same person. The NNDB itself does not deal with this kind of ambiguity. For handling multilingual naming, synonymity, and other naming conventions a dictionary layer is necessary on top of the NNDB, which provides the real mapping between entities in questions and elements of the NNDB. Entities are mapped into primary schemas, attributes, and attribute values of natural keys [15].
  However, elements of NNDBs can still be ambiguous in the sense that the different entities may share the same representation forms. For example, the name Charles de Gaulle could stand for a historical personage, a name of street, an airport, a railway station, a national research project, and so on.

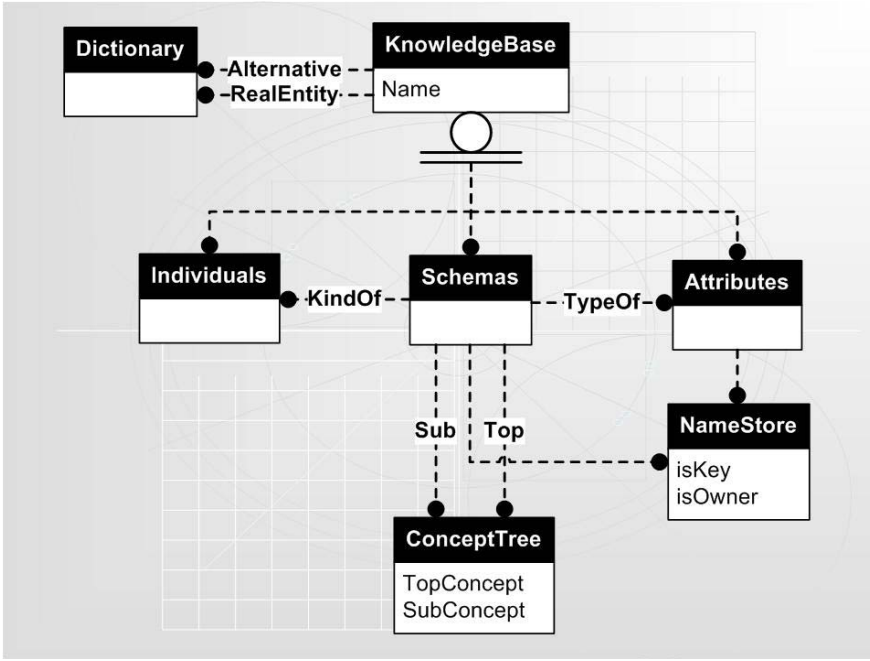

Fig. 2. NNDB example database (IDEF1X notation).

- Identify noun phrases in sentences using a state-of-the-art syntactic parser. Noun phrases describe a possibly non-empty set of entities that the question is all about. In other words, noun phrases contain the key data that identify the context of the sentence.
- Let $t_1, t_2, \ldots t_n$ be the representations of noun phrase heads in a $\mathcal{DB} = <\mathbf{R}, \mathbf{r}, \mathbf{F} \cup \mathbf{I}, \lambda, \Xi>$. If noun phrase heads are all described by $\mathcal{DB}$ then find the common schema which connects all possible meanings of noun phrase heads. In order to do that, one must distinguish among three different cases depending on whether the noun phrase head t_i is a schema, an attribute or an attribute value in $\mathcal{DB}$:

$$\gamma(t_i) = \begin{cases} \|\kappa(t_i)\| & \text{if } t_i \text{ is a schema} \\ \|t_i\| & \text{if } t_i \text{ is an attribute} \\ \{\|t_j\| | t_i \in \mathrm{DOM}(t_j)\} & \text{if } t_i \text{ is a value.} \end{cases}$$

The common schema which connects all noun phrase heads is

$$\Gamma = \bigcap_{i=1}^{n} \gamma(t_i).$$

Note that ambiguities are partially eliminated, therefore meaningless expressions result in an empty set. On the other hand, the disambiguation is not fully

done at this point since alternatives with common schemas are still present after this step. Two typical cases are interrogative sentences that contain a single noun phrase, and sentences that are ambiguously stated by the user. In either case, common schemas are all returned to the user as alternatives for topic restriction.
If noun phrase heads are not represented in databases, the algorithm terminates with an error. Unfortunately, information on purposes, usage, causes, accidental events, experiments, are typically can not be represented and therefore not stored in databases. Moreover, the outlined method is unable to properly resolve cultural idioms (e.g. The Voice as Frank Sinatra), common sense expressions (e.g. Washington as U.S. government), symbolic, associative expressions (e.g. Mecca of movies), etc.

– Noun phrases identify contexts not by naming a common schema but by naming the focus of the question. The information needs of a user executing a search can be represented by a triplet: the question tag, the head of the verb phrase, and by the head of the first noun phrase after the question tag or the verb phrase excluding pronouns if the former is missing. This observation led us to apply a template based focus identification algorithm, i.e., template triplets determine a set of schemas Δ.
– The context of question is $\Gamma \cap \Delta$.

What about ambiguities in NNDBs? Natural language ambiguities can be resolved only by context similar to the way they are resolved in natural conversations. If $\Gamma \cap \Delta$ contains more than a single schema then user is prompted to choose between these options.

6 Deep Web Search Engine in Work

Once the context is identified, one needs to find the databases that may contain information about the input question. It is easy to see that deep web sites might have a correct answer for the input question if and only if all their schemas and attributes can be mapped into the NNDB. Unfortunately, such a mapping is not easy to find algorithmically, hence schema and attribute names, and their semantics might differ from the ones used in NNDBs. This is why WoW requires their partners to declare which schemas and attribute elements are present in their databases, and how they are stored. With these information mediator layers can make way for the proper transformation from NNDB queries to URLs, forms, or deep web database queries.

The role of DWSE is twofold. First, it serves to maintain information on all connected deep web sites by storing their metaschemas and data structures, and second, it determines by simple mathematical relations which deep web sites may be relevant to answer the user's question.

7 An Illustrative Example

One stores information on books, movies, locations, and cultural events. Let $\mathcal{DB} = <\mathbf{R}, \mathbf{r}, \mathbf{F} \cup \mathbf{I}, \lambda, \Xi>$ be defined in the following way:

- $\mathbf{R}$ = {movie, cinema, person, director, actor, location, datetime, shows, acts, directs, lives} where the first 7 schemas are primary schemas, and others are secondary ones.
- For primary schemas: κ(movie) = title, κ(cinema) = cname, κ(person) = pname, κ(actor) = aname, κ(director) = dname, κ(location) = place, κ(datetime) = date.
- For secondary schemas: shows = {movie, datetime, location}, acts = {amovie, aactor}, directs = {dmovie, ddirector}, lives = {person, address}. λ(amovie) = λ(dmovie) = title, λ(datetime) = date, λ(address) = place, λ(aactor) = aname, λ(director) = dname, and λ(person) = pname.
- Ξ(director, person), Ξ(actor, person), Ξ(cinema, location).
- There are no other dependencies.

In the following examples we skip both the named entity recognition and the NNDB mapping step for the sake of simplicity.

For the question "Where did Churchill live?" the context identification procedure finds the single phrase head Churchill. Churchill as a name could stand for either a person, a place or a movie, i.e. $\Gamma = \mathbf{R}$. We have a template for <where, live, ...> where "..." means arbitrary first phrase head. The template maps this triplet into Δ = {location}, therefore the context of this sentence must be $\Delta \cap \Gamma$ = {location}. Based on these information, deep web algorithm transforms user question into the following SQL statement.

```
SELECT place FROM lives WHERE person = 'Winston Churchill'
```

For basic ideas on this transformation see [15, 16]. Due to lack of space, full details will be published in the near future.

The algorithm processes the question "In which movie did Quentin Tarantino play?" in the following way. It first identifies two simple noun phrases: movie and Quentin Tarantino. Since Quentin Tarantino is both a director and an actor $\Gamma = \gamma$(Quentin Tarantino) $\cap\, \gamma$(movie) = {acts, directs}. The focus identification process determines Δ = {acts} using the template <which, play, movie>, that is, the result must be $\Gamma \cap \Delta$ = acts. During the next phase, the next SQL statement is generated:

```
SELECT movie FROM acts WHERE aactor = 'Quentin Tarantino'.
```

When ambiguity has to be resolved, the system prompts the user to clarify the question "Where can I see Pulp Fiction?". The solution produced by the algorithm contains two possibilities: the user either asked about the cinema or the place where the movie will be shown. The algorithm does not prioritize, therefore the user interaction is unavoidable.

In the current implementation our knowledge base contains 18000 terms (∼110 schemas, ∼230 attributes) on 23 topic areas, e.g. movies, books, restaurants, locations, cultural events, and related institutions, groups, people, etc (sources are port.hu, National Széchényi Library, eszemiszom.hu). Our template database consists of cca. 1000 templates. We found that this approach has a 83% precision on a cca. 1000 sentence corpus extracted from Szeged Treebank [19], however, in most cases (cca. 72%) it finds the question to be incomplete. Obviously, this approach also has its limitations: it cannot retrieve information that does not fit into the database model (e.g. questions about reasons, causality, subjectivity). We excluded these types of questions from testing.

8 Concluding Remarks

In this paper, we presented a new relational database design technique called NNDB. We pointed out that a proper NNDB is a context database, and it could serve as the basis of context identification combining the template based techniques and using the world model encoded in the database design. The database structure can be easily extracted from any relational database, and needs no re-engineering technique. Moreover, it has a well-formed mathematical background based on relational theory.

Acknowledgement

Our work was sponsored by Mobile Innovation Center through the Asbóth Oszkár Programme of the Hungarian State.

References

1. Winkler, H.: Suchmaschinen. Metamedien im Internet? In Becker, B., Paetau, M., eds.: Virtualisierung des Sozialen, Frankfurt/NY (1997) 185–202 (In German; English translation: `http://www.uni-paderborn.de/~timwinkler/suchm_e.html`).
2. He, B., Patel, M., Zhang, Z., Chang, K.: Accessing the Deep Web: A Survey. Communications of the ACM **50** (2007) 94–101
3. Bergman, M.K.: The deep web: surfacing hidden value. Journal of Electronic Publishing **7** (2001) `http://www.press.umich.edu/jep/07-01/bergman.html`.
4. Lawrence, S., Giles, C.: Accessibility of information on the web. Nature **400** (1999) 107–109
5. Tikk, D., Kardkovcs, Z.T., Magyar, G.: Searching the deep web: the WOW project. In: New Methods and Practice for the Networked Society. Volume 2 of Advances in Information Systems Development., New York, Springer (2007) 493–504
6. Tikk, D., Kardkovács, Z.T., Andriska, Z., Magyar, G., Babarczy, A., Szakadát, I.: Natural language question processing for hungarian deep web searcher. In Elmenreich, W., Haidinger, W., Machado, T., eds.: 2^{nd} IEEE International Conference on Computational Cybernetics, ICCC 2004, Vienna, Austria (2004) 303–309

7. Tikk, D., Szidarovszky, F.P., Kardkovács, Z.T., Magyar, G.: Entity recognizer in Hungarian question processing. In Bandini, S., Manzoni, S., eds.: AI*IA 2005: Advances in Artificial Intelligence. Number 3673 in Lecture Notes in Artificial Intelligence. Springer, Berlin–Heidelberg–New York (2005) 535–546
8. Chang, K.C.C., He, B., Zhang, Z.: Toward large scale integration: Building a MetaQuerier over databases on the web. In: Proceedings of the 2^{nd} Biennial Conference on Innovative Data Systems Research (CIDR 2005), Asilomar, CA, USA (2005) 44–55
9. He, H., Meng, W., Yu, C., Wu, Z.: Wise-integrator: an automatic integrator of web search interfaces for e-commerce. In: Proceedings of the 29^{th} international conference on Very large data bases. Volume 29., VLDB Endowment (2003) 357–368
10. Wu, W., Doan, A., Yu, C.: WebIQ: learning from the web to match deep-web query interfaces. In: Proc. of the 22^{nd} Int. Conf. on Data Engineering (ICDE'06), IEEE Computer Society (2006) 44–53
11. Sagiv, Y.: A characterization of globally consistent databases and their correct access paths. ACM Transactions on Database Systems **8** (1983) 266–286
12. Honeyman, P.: Testing satisfaction of functional dependencies. Journal of the ACM **29** (1982) 668–677
13. Chan, E.P.F., Mendelzon, A.O.: Independent and separable database schemes. In: PODS'83: Proceeding of the 2^{nd} ACM SIGACT-SIGMOD symposium on Principles of database systems, New York, NY, USA, ACM Press (1983) 288–296
14. Kardkovács, Z.T.: On the transformation of sentences with genitive phrases to SQL statements. In: Proc. of the 10^{th} NLDB. Volume 3513 of Lecture Notes in Computer Science., Alicante, Spain, Springer Verlag (2005) 10–20
15. Kardkovács, Z.T., Tikk, D.: On the transformation of sentences with genitive relations to SQL queries. Data & Knowledge Engineering **61** (2007) 406–416
16. Kardkovács, Z.T.: Querying Heterogeneous Databases Based on Semantic Processing of Well-Formed Language Phrases. PhD thesis, Budapest University of Technology and Economics, Budapest, Hungary (2007) In Hungarian.
17. Meng, X., Wang, S., Wong, K.F., Lum, V.: A chinese query language chiql: Design and evaluation. In: SEEP'98: Proceedings of International Conference on Software Engineering: Education and Practice, New Zealand, IEEE Press (1999) 190–197
18. Trón, V., Németh, L., Halácsy, P., Kornai, A., Gyepesi, G., Varga, D.: Hunmorph: open source word analysis. In: Proc. of ACL. (2005)
19. Csendes, D., Csirik, J., Gyimóthy, T., Kocsor, A.: The Szeged Treebank. In Matousek, V., Mautner, P., Pavelka, T., eds.: Proc. of the 8^{th} TSD. Volume 3658 of Lecture Notes in Computer Science., Karlovy Vary, Czech Republic, Springer Verlag (2005) 123–131