PUBLICATIONS DE L'INSTITUT MATHÉMATIQUE
Nouvelle série tome 54 (68), 1993, 156–171

INFERENCE IN A RELATIONAL KNOWLEDGE-BASE WITH TWO TYPES OF NULL VALUES

Gordana Pavlović-Lažetić

Abstract. The problem of retrieving a knowledge-from-text base, on a text level, is considered. A solution is given that is based on retrieving a text on a sentence level. It consists of converting a query whose qualification and target attributes are not present in the same sentence, into an equivalent set of subqueries, each of which having all of its attributes present in the same sentence. Logical conditions underlying this decomposition are based on functional and multivalued dependencies between qualification and target attributes of the subqueries. The dependencies are defined in the extended relational model with two types of null values, representing a text. These conditions are proved to provide for answers to the last subquery to coincide with the answers to the original query, i.e., to be correct.

1. Introduction

Lexical data type for storing and manipulating textual databases has been designed and reported elsewhere [**12**]. It expands the relational database concept as to provide for managing text as data. The approach to organizing texts enables simple and precise realization of a set of operations over a text represented as an array of lexical data. Different level operators have been designed and implemented, ranging from low level operators on single lexical units (e.g., find root, part of speech, semantic property of a word), rule-based operators on sets of lexical units (e.g., morphological analysis, resolving ambiguity), up to high level traditional operators over an entire array of lexical data representing a text, such as automatic indexing, determining keywords and phrases, abstracting, retrieval, editing. A specific, high-level semantic operator of intelligent retrieval of a lexical database, has been designed and reported in [**12, 13**]. Knowledge from a text is treated as a disconnected knowledge on a sentence level, so that recall and precision of an answer obtained to a given query depends on ability to map, completely and precisely, a formal query into a search mechanism through a sentence of a text. Since different sentences contain, as a rule, information of different kinds, knowledge-from- text representation scheme is a relational model with two types of null values.

AMS Subject Classification (1980): Primary 68 B 15

This paper addresses the problem of retrieving a knowledge-from-text base, not only on a sentence level, but also on a text level. A mechanism is necessary that converts a query whose qualification is present in a sentence and target attributes are not, into a set of subqueries, each of them having qualifications and target attributes present in the same sentence. An answer to the last subquery has to coincide with an answer to the original query. The corresponding mechanism has its formal ground in the extended relational model with null values, and in query decomposition based on functional and multivalued dependencies in it. The effect is retrieval of a connected text based on retrieving a disconnected one.

2. Intelligent information retrieval – relational knowledge-base model

Intelligent information retrieval (facts extracting from texts) consists in asking a question about a fact from a text (for example, "When a person named X was born?") and finding an answer (for example, "1879"). This activity may be represented by an operator [**13**]

$$\{\text{text}\} \times \{\text{queries}\} \rightarrow \{\text{facts}\}.$$

As opposed to other approaches where the problem of facts extracting is considered as a constituent part of a broader problem of connected text understanding, (e.g., [**1**], [**7**], [**9**], [**15**]), this approach proceeds from the problem itself. The relational model and a relational schema are chosen as a representation scheme of limitted (disconnected, sentence-level) knowledge from a text. A relational schema represents a user's view of a text. For example, from a set of biographies, a user may be interested in birth dates, birth places, degrees, employments, degree institutions and employment institutions, and nothing else. Thus, a relational schema designed as a view to a set of biographies may contain, among the others, the following relation:

DEGREE(*name, institution, degree, degree-date, degree-field*).

The set of biographies may contain the following sentence:

(*a*) *The University of Belgrade gives Ph.D. degree in chemistry.*

The schema à priori defines a set of queries allowed. For example, the following query is a legal one:

(*q*) *What fields the University of Belgrade gives degrees in*?

Mapping of a pair (text, query) into answers to the query (for example, (*biographies,* (*q*)) $\rightarrow$ (*chemistry*)), intensively uses syntactic and semantic properties of words incorporated into lexical data representing those words.

For queries that may be posed over texts the following holds:

A) every query is one variable query, and thus one relation query (many variable query may be decomposed into a set of one variable queries);

B) queries are expressed in a query language (e.g., QUEL [**8**]).

Main components of the sistem for intelligent text retrieval are: a *relational schema* defined for a specific application, *text retrieval operator* (defined by mapping a query over a relation into a search mechanism through a text) and a *search algorithm* for finding an answer.

Definition of a relational schema is a design problem. In order to implement a text retrieval operator, the relational schema is supplied with some language-oriented semantics, e.g.:

$$\left.\begin{array}{ll} f_1 : \{\text{relations-relationships}\} & \rightarrow \\ f_1 : \{\text{relations-entities}\ \} & \rightarrow \\ f_2 : \{\text{domains}\ \} & \rightarrow \\ f_3 : \{\text{attributes}\ \} & \rightarrow \end{array}\right. \begin{cases} \{(\text{lex[position]})\ |\text{root(lex[pos])} = \text{const}_1 \text{ and/or} \\ \qquad \text{type(lex[pos])} = \text{const}_2 \text{ and/or} \\ \qquad \text{prop(lex[pos])} = \text{const}_3 \text{ and/or} \\ \qquad \text{type(lex[pos} - 1]) = \text{prep and/or} \\ \qquad \text{type(lex[pos} + 1]) = \text{prep and/or} \\ \qquad (\ldots)\} \end{cases}$$

$$f_4(=I) : \{\text{constants}\} \rightarrow \{\ \text{constants}\}.$$

One variable query over a relation R of a relational database is of the form

$(q1)$ range of e is R

retrieve $(e.C)$ where $e.A = \alpha$

$(C = \{C_1, C_2, \ldots, C_k\},\quad A = \{A_1, A_2, \ldots, A_m\}$ -sets of attributes of the relation $R,\ \alpha = (a_1, a_2, \ldots a_m)\ - m$ -tuple of constants, $e.C \equiv \{e.C_1, e.C_2, \ldots e.C_k\}$, and $e.A = \alpha$ denotes $e.A_1 = a_1, e.A_2 = a_2, \ldots, e.A_m = a_m)$.

In the example query (q), C ={degree-field}, A ={institution}, α = {University of Belgrade}.

Since the query $(q1)$ is characterized by sets of attributes A, C and a constant α , mapping of the query $(q1)$ into a search mechanism through a text is of the form

$$f = f_1 \times f_2 \times f_3 \times f_4 :\ (q1) \rightarrow (qt1),$$

where $(qt1)$ = (find all the sets of lexical units satisfying $f_3(C)$, $f_2(\text{domain}(C))$ and belonging to sentences that contain $f_1(R)$ and a constant $f_4(\alpha)$ (or $f_3(A_i)$ or $f_2(\text{domain}(A_i))$, for each attribute A_i from A)) .

Let $\mathbf{answ}(q1)$ consist of all the answers to the query $(q1)$ contained in the text (no matter how can they be found). The query $(qt1)(\equiv f(q1))$ has an effectively computable set of answers, $\mathbf{answt}(qt1)$ (implemented by the third component of the system, a search algorithm). In an idealistic case (the mapping f maps ideally elements of a relational schema into language elements and all the answers to the query are contained in the same sentences with the query data), the equality **answ** = $\mathbf{answt}{*}f$ would hold. Something weaker, but still an idealistic case would be

answw' $=$ **answt**$*f$, where **answ'**$(q1)$ is a set of answers to the query $(q1)$ contained in the same sentences with the query data (A, α). It is desirable to provide for, at least, inclusion **answt**$*f \subseteq$ **answ'** , since it means that all the answers found are correct.

The following two measures express levels of satisfaction of the two inclusions, **answt** $*f \subseteq$ **answ'**, **answ'** $\subseteq$ **answt** $*f$:

-*precision* (pr) of the mapping **answt**$*f$: pr(**answt**$*f$) $= p = P($**answt**$*f \subseteq$ **answ'**$)$ (a probability that an answer found from a set of lexical units, representing an image of a query, is correct);

– *recall* (rec) of the mapping **answt**$*f$: rec(**answt**$*f$) $= p' = P($**answ'** $\subseteq$ **answt** $* f)$ (a probability that a set of lexical units, containing an answer, as well as the answer itself, are found).

A probability p is a probability of an intersection of the two independent events ((1) a set of lexical units which is an image of a query, under the mapping f, contains an answer, (2) an answer found from the set of lexical units is correct), (and similarly for the probability p'). Since $f = f_1 \times f_2 \times f_3 \times f_4$, the following holds:

$$p = \mathrm{pr}(f) * \mathrm{pr}(\mathbf{answt}) = \mathrm{pr}(f_1) * \mathrm{pr}(f_2) * \mathrm{pr}(f_3) * \mathrm{pr}(f_4) * \mathrm{pr}(\mathbf{answt})$$
$$p' = \mathrm{rec}(f) * \mathrm{rec}(\mathbf{answt}) = \mathrm{rec}(f_1) * \mathrm{rec}(f_2) * \mathrm{rec}(f_3) * \mathrm{rec}(f_4) * \mathrm{rec}(\mathbf{answt}),$$

where

$$\mathrm{pr}(f_1) = \frac{\text{number of lexical units in } f_1(\text{relations}), \text{semantically related to relations}}{\text{number of lexical units in } f_1(\text{relations})}$$

(and similarly for mappings $f_2 - f_4$, **answt**),

$$\mathrm{rec}(f_1) = \frac{\text{number of lexical units in } f_1(\text{relations}), \text{semantically related to relations}}{\text{number of lexical units semantically related to relations}}$$

(and similarly for mappings $f_2 - f_4$, **answt**).

Intuitivelly, a precison of a mapping is a probability of an image of a database element to be semantically related to that element. A recall of a mapping is a probability of a phrase which is semantically related to a database element, to be in an image of that element under the mapping. If a precision of all the mappings $f_1 - f_4$, **answt** is 100% , then **answt**$*f \subseteq$ **answ'**. If a recall of all the mappings $f_1 - f_4$, **answt** is 100%, then **answ'** $\subseteq$ **answt**$*f$.

The above features of the mappings $f_1 - f_4$, **answt**, are purely empirical and belong both to the field of linguistics and to semantics of a model. Maximizing probabilities p, p' is a primary goal of a design of an intelligent retrieval.

The third component of the system, search algorithm (**answt**) for finding an answer to a query over candidate sets of lexical units obtained by the mapping f, may involve different strategies for determining levels of relevant sentences and phrases [**13**], which are beyond the scope of this paper.

An entity or a relationship represented by a relation in a relational schema corresponds to a sentence from a text. Since a sentence may contain data which are relevant for some (but not for all) attributes of an entity (relationship), relational model used for knowledge from text representation includes null values. For example, a set of biographies, except for the sentence (a), may also contain the following two sentences:

(b) *XX received her M.S. degree in computer science.*

(c) *XX received her M.S. degree from the University of Belgrade.*

Then a virtual contents of the relation *DEGREE* may be

DEGREE	name	institution	degree	degree-date	degree-field
	ρ	Univ. of Belgrade	Ph.D	ρ	chemistry
	XX	ω	M.S.	ω	comp. science
	XX	Univ. of Belgrade	M.S.	ω	ω

where ω denotes yet unknown value, and ρ denotes undefined value [**11**]. Still, by unification of the attributes *institution*, *degree-field* (because of the functional relationship between sets of attributes $\{$*name, degree*$\}$ and $\{$*institution, degree-field*$\}$, another answer, *computer science*, may be obtained. By applying "factual" inference, i.e., operators of loseless projection and join (based on functional and multivalued dependencies), relational model becomes a representation of connected knowledge from text: it provides for linking attribute values not contained in a single sentence.

3. Dependencies in the relational data model with two types of null values

Relational model of data with incomplete information has been considered from all the aspects – structural, manipulative and integrative (e.g., [**3**], [**5**], [**6**], [**10**], [**14**], [**16**]).

In [**11**], a generalizad relational model with undefined (ρ) and unknown (ω)-types of null values was considered, from the point of view of operations of the basic relational algebra [**2**] applicable to relations containing these null values. Undefined value has the meaning "property unapplicable", while unknown value stands for any value (not known yet) from a domain of an attribute, but not for the undefined value. With each operation of the relational algebra (union, intersection, difference, Cartesian product, projection, restriction, join, division), two operations in the relational model with two types of null values were associated: "true" and "maybe" operations. A result of a "true" operation contained tuples that were known to satisfy conditions defining the corresponding operation of the basic relational algebra. A result of a "maybe" operation contained tuples for which it was not known whether they satisfied conditions or not. Properties of and relationships between the operations were proved that are significant for query optimization. It was necessary to define, first, three-valued relations of equality of individual values, equality of tuples, relation membership and inclusion; the truth value set was {T, F, ω } - "true", "false" and "unknown" ("maybe"). For example, a three-valued equality of individual values was defined by the following table:

$\tau(x \doteq y)$	$x \in D$	$x = \omega$	$x = \rho$
$y \in D$	$\tau(x = y)$	ω	F
$y = \omega$	ω	ω	F
$y = \rho$	F	F	T

(τ denotes a truth value, $=$ denotes string equality, $\tau(x = y)$ denotes the truth value of the basic two-valued equality, and D is any domain).

Then an extended equality of tuples r, s ("true", "maybe", "false"), over the same set of attributes X, is defined in the following way:

$$\begin{array}{lll} \tau(r \doteq s) = T(true) & \text{iff} & (\forall A \in X)\ \tau(r[A] \doteq s[A]) = T(true); \\ \tau(r \doteq s) = \omega(\text{“}maybe''), & \text{iff} & (\forall A \in X)\ \tau(r[A] \doteq s[A]) \in \{T, \omega\}, \\ & & \text{and } \tau(r \doteq s) \neq T; \\ \tau(r \doteq s) = F(false) & \text{iff} & (\exists A \in X)\ \tau(r[A] \doteq s[A]) = F. \end{array}$$

There are many possibilities for defining integrity constraints, specifically functional and multivalued dependencies (FD, MVD, respectively, [**4**]), in the above extended model. An "extended" definition of an FD (MVD) has to provide for, at least, "natural extension", which means that if an FD (MVD) holds in the basic relational model (without null values), than it has to hold according to the extended definition, too.

One possible criterion in deciding upon an extended definition of dependencies, is a "degree of certainty" of an assertion that in a relation R of the extended model an FD (MVD) holds. Let a "completion" of R consist of replacing each tuple r that contains ω - values (if there is any) with one or more tuples which are "maybe" equal to r and do not contain ω - values. Then the following alternative definitions of FD (MVD), in increasing order of certainty and restrictiveness, may be introduced:

Definition 1. FD $X \to Y$ (MVD $X \to\to Y$) holds in a relation R with three disjoint sets of attributes X, Y, Z iff

A. for at least one completion of R , FD (MVD) holds according to the definition of FD (MVD) for the basic relational model (ρ-values, if there are any, are treated like any other value from the domain), or

B. for each replacement of ω-values on X, by specific domain values, there exists a completion of R such that FD (MVD) holds in it according to the definition of FD (MVD) for the basic relational model, or

C. for all completions of R , FD (MVD) holds, according to its basic definition.

Although all the three alternatives satisfy "natural extension" principle, the **C**-condition seems to be excessively restrictive. It provides for certainty in a sense that every FD (MVD) which holds in a relation R according to Definition 1C, also holds according to the basic FD (MVD) definition in a relation R' which is obtained from the relation R by substituting arbitrary domain-values for ω-values in R. Still, since every ω value stands for a single value from a domain (not for all the values), this definition does not "catch" some of the FDs (MVDs) that would actually hold after completing incomplete information.

An FD (MVD) defined by either Definition 1A or Definition 1B, may not imply the corresponding basic FD (MVD) in some of the completions of the relation R. Still, since Definition 1B provides for this implication in more cases, and it also "catches" significant amount of FDs (MVDs) that would actually hold after completing incomplete information, the rest of this consideration will relate to Definition 1B.

Definition 1B can be split into two definitions (one for FD, another for MVD), both formalizad and expressed in terms of the relation in question alone.

Definition 2B. Let R be a relation in the extended relational model with two types of null values, with three disjoint subsets of attributes X, Y, Z. An FD $X \to Y$ holds in R iff

$$(\forall r, s \in R)(\tau(r[X] \doteq s[X]) \in \{T, \omega\} \implies \tau(r[Y] \doteq s[Y]) \in \{T, \omega\})$$

For example, in the relation "DEGREE" of the virtual relational schema described in section 2, a functional integrity constraint $X = \{\text{name, degree}\} \to Y = \{\text{institution, degree-field}\}$ holds according to Definition 2B of FD:

$$r = (\text{XX, mr}, \omega, \omega, \text{ computer science}), \quad s = (\text{ XX, mr, UB}, \omega, \omega),$$
$$r[X] = s[X] = (\text{XX, mr}), \quad r[Y] = (\omega, \text{computer science}), \quad s[Y] = (\text{UB}, \omega),$$
$$\tau(r[X] \doteq s[X]) = T, \quad \tau(r[Y] \doteq s[Y]) = \omega \ .$$

It can be noted that Definition 2B is either equivalent to a "strong" FD or lies between "weak" and "strong" FD as defined in [**16**], depending on how one interprets the equality (=) used there.

Definition 3B. Let R be as in the definiton 2B. For $x, x' \in R[X]$ s.t. $\tau(x \doteq x') \in \{T, \omega\}$, and for $z \in R[Z]$ (string-based element), let $Y_{xz}, Y_{(x,x')z}(-R)$ be the sets $\{y \mid (x,y,z) \in R\}$, $\{y \mid y \in R[Y] \& (x,y,z) \notin R \; \& (\exists t \in R)(\tau(t \doteq (x,y,z)) \in \{T,\omega\} \& \tau(t[X] \doteq x') \in \{T,\omega\})\}$, respectively.

Then an MVD $X \to\to Y$ holds in the relation R iff for each pair $x, x' \in R[X]$ such that $\tau(x \doteq x') \in \{T, \omega\}$ and for each $z, z' \in R[Z]$ such that $Y_{xz}, Y_{x'z'}$ are nonempty, the following holds:

$$Y_{xz} \subseteq Y_{x'z'} \cup Y_{(x',x)z'}(-R) \qquad \text{and} \qquad Y_{x'z'} \subseteq Y_{xz} \cup Y_{(x,x')z}(-R).$$

It can be proved that Definition 1B is equivalent to Definitions {2B, 3B}.

For MVD defined by Definition 3B some properties analogous to the properties of MVD in the basic relational model [**4**] may be proved:

$P1$. If X and Y are disjoint, and if the FD $X \to Y$ (defined by Definition 2B) holds in a relation R, then MVD $X \to\to Y$ (defined by Definition 3B) also holds in R.

$P2$. (*characterization theorem*) MVD $X \to\to Y$ (defined by Definition 3B) holds in a relation R iff $\tau(R[X,Y] *_T R[X,Z] \cup R[X,Y] *_\omega R[X,Z] \doteq R) \in \{T,\omega\}$ ($*_T, *_\omega$ are **true** and **maybe** join operators).

The fact that some FDs and MVDs hold in a relation of the relational model with two types of null values provides for decomposition of a query over a relational knowledge base with null values into a set of queries whose target and qualification attributes do not take null values. This decomposition is named *factual inference* and will be elaborated in the next section.

4. Factual inference in a relational knowledge base with null values

In the section 2, an example of two answers to a query over a text has been given. The second answer was present in the text, but the way to come to it was indirect, through other information contained in the text directly. Data we give (constants), information constituting an answer (target values) and indirect information leading to the answer (intermediaries), are semantically related in the text. The semantic relationship has its analogue in a formal scheme, and these are functional and multivalued dependencies in the model with two types of null values. A process of computing this indirection is named *factual inference*.

For a given query over one relation, a sentence containing qualification constants but not values of target attributes, corresponds to a virtual tuple with null values on the query target attributes. An answer to the query cannot be obtained from such a sentence (tuple), but the original query can be decomposed into a set of subqueries whose qualifications and target attributes satisfy an FD or an MVD. An answer to each of the subqueries is then searched for directly in a single sentence (i.e., a tuple having not-null values on target attributes of the subquery). An answer to the last query is then an answer to the original query.

In what follows, an inference algorithm for answer finding will be described, which includes a query decomposition. Then a set of theorems is formulated (and proved) leading to the proof that answers, obtained by the algorithm, are as precise (correct) as a mapping of a formal query into a textual query, and a mapping of the textual query into answers from the text, are precise, and as complete as these mappings are complete.

4.1. Inference algorithm. Let a relational schema R^* be defined by sets of (1) relations, (2) attributes, (3) domains, and (4) functional and multivalued dependencies. Let $R(A, C, REST)$ ($A, C, REST$ - disjoint sets of attributes) be a relation of the schema. Given a query over the relation R

($q1$) range of e is R

retrieve $(e.C)$ where $e.A = \alpha$,

the following algorithm gives answers to the query $(q1)$, contained in texts.

Algorithm A

1. map the query $(q1)$ into lexical categories of texts – into the query $(qt1)$ (mapping f);

2. find answer(s) if it (they) exist(s), to the query $(qt1)$ (mapping **answt**);

3. *if* an answer has been found in the step 2, *then*

3a. *if* an FD $A \to C$ holds in the relation R, *then* stop;

4. let $B_1, B_2, \ldots, B_k$ be a set of pairwise disjoint sets of attributes of the relation R, which do not take ω values, and such that multivalued dependencies $B_1 \to\to B_2, B_2 \to\to B_3, \ldots, B_k \to\to C$ hold in the projection $R[A, B_1, B_2, \ldots, B_k, C]$ of the relation R. Then for the smallest $k > 0$ for which the step 4. has not been done yet, *do* :

4a. substitute the query

$$
\begin{array}{ll}
(q2) & \text{range of } e_1 \text{ is } R \\
& \text{range of } e_2 \text{ is } R \\
& \quad\vdots \\
& \text{range of } e_{k+1} \text{ is } R \\
& \text{retrieve } (e_{k+1}.C) \text{ where } e_{k+1}.B_k = e_k.B_k \text{ and} \\
& \qquad e_k.B_{k-1} = e_{k-1}.B_{k-1} \text{ and } \ldots \text{ and } e_1.A = \alpha
\end{array}
$$

for the query $(q1)$;

4b. decompose the query $(q2)$ into $k+1$ queries $(q2_i)$, that are equivalent to the query $(q2)$:

$$
\begin{array}{rl}
(q2_1) & \text{range of } e_1 \text{ is } R \\
& \text{retrieve } \underbrace{(e_1.B_1)}_{c_1} \text{ where } e_1.A = \alpha \\
(q2_2) & \text{range of } e_2 \text{ is } R \\
& \text{retrieve } \underbrace{(e_2.B_2)}_{c_2} \text{ where } e_2.B_1 = c_1 \\
& \vdots \\
(q2_{k+1}) & \text{range of } e_{k+1} \text{ is } R \\
& \text{retrieve } (e_{k+1}.C) \text{ where } e_{k+1}.B_k = c_k;
\end{array}
$$

4c. map every query from 4b. $(q2_i)$ into a query over a text $(qt2_i$, mapping $f)$, and search for answers to each of the queries $(qt2_i)$; an answer to the query $(qt2_{k+1})$ is an answer to the original query (q_2);

4d. repeat the step 4 *if* possible, *else* stop.

THEOREM A (on the Algorithm A): *Let* pr(f), pr(f_1) – pr(f_4), pr(**answt**), *be precisions of the mappings* f, $f_1 - f_4$, **answt**, *respectively. Then for every answer to the query* ($q1$), *obtained by the algorithm A, the following holds:*

$$P(\text{answer } \in \mathbf{answ}(q1)) = (\text{pr}(f_1) * \text{pr}(f_2) * \text{pr}(f_3) * \text{pr}(f_4) * \text{pr}(\mathbf{answt}))^{k+1}$$

where $k+1$ *is the number of tuple variables of the query* ($q2$) *i.e., answers obtained by the algorithm A are precise up to the precision of the mapping* $\mathbf{answt} * f$ (*and similarly for the recall of answers obtained by the algorithm A*).

Remark 1. If all the values pr(f_1) – pr(f_3) are equal to 1, all the answers obtained by the algorithm A are correct answers to the query ($q1$). A necessary condition for this to hold is that every two distinct attributes of a virtual database map into distinct sets of lexical categories.

Proof of Theorem A is in the Appendix, and it is based on the following two theorems:

THEOREM 1 (on equivalent decomposition of a query over sets of attributes of one relation) *Let* $R(A, C, REST)$ *be a relation in the extended relational model with two types of null values, with disjoint sets of attributes* $A, C, REST$, $D(A)$ - *a Cartesian product of domains of attributes from* A , *and* $\alpha \in D(A)$ - *a constant m-tuple from* $D(A)$. *Let* $B \subseteq REST$ *be a set of not-*ω *attributes* (*attributes not allowed to take* ω-*value*). *For two queries*

($q1$) range of e is R
retrieve ($e.C$) where $e.A = \alpha$

($q2$) range of e is R
range of u is R
retrieve ($u.C$) where $u.B = e.B$ and $e.A = \alpha$

the following holds:

(1) **answer**($q1$) $\subseteq$ **answer**($q2$), *where* **answer**($q1$) , **answer**($q2$) *are results of the queries* ($q1$), ($q2$) , *respectively*;

(2) *a sufficient condition for* $\tau(\mathbf{answer}(q1) \dot{\supseteq} \mathbf{answer}(q2)) \in \{T, \omega\}$ (*i.e.* $\tau(\mathbf{answer}(q1) \doteq \mathbf{answer}(q2)) \in \{T, \omega\}$) *to hold is existence of an MVD* $B \to\to C$ in $R[A, B, C]$;

(3) *if an FD* $A \to C$ *holds in* R, *then an answer to the query* ($q1$) *is unique.*

Proof of the theorem is in the Appendix.

THEOREM 2 (on horizontal generalization of Theorem 1 – generalization on the number of intermediate sets of attributes) *Let* $R(A, B_1, B_2, \ldots, B_k, C, REST)$ *be a relation with disjoint sets of attributes* $A, B_1, B_2, \ldots, B_k, C, REST$, *where* $B_1, \ldots, B_k$ *are not-*ω *attributes. Let* $\alpha \in D(A)$ *be a constant as in Theorem* 1. *For two queries*

($q1$) range of e is R
retrieve ($e.C$) where $e.A = \alpha$

$(q2)$ range of e_1 is R
range of e_2 is R
$\vdots$
range of e_k is R
range of e_{k+1} is R
retrieve $(e_{k+1}.C)$ where $e_{k+1}.B_k = e_k.B_k$ and $e_k.B_{k-1} = e_{k-1}.B_{k-1}$
and $\ldots$ and $e_2.B_1 = e_1.B_1$ and $e_1.A = \alpha$

the following holds:

(1), (3) - *analogously to the corresponding statement of Theorem* 1;

(2) *a sufficient condition for*

$\tau(\mathbf{answer}(q1) \dot{\supseteq} \mathbf{answer}(q2)) \in \{T, \omega\}$, *i.e.,*
$\tau(\mathbf{answer}(q1) \doteq \mathbf{answer}(q2)) \in \{T, \omega\}$

to hold is existence of a set of MVDs:

$$B_1 \to\to B_2,\ B_2 \to\to B_3,\ \ldots,\ B_{k-1} \to\to B_k,\ B_k \to\to C,$$

in the projection $R[A, B_1, B_2, \ldots, B_k, C]$.

Proof of the theorem is in the Appendix.

5. Conclusion

Retrieving a disconnected text (knowledge on a sentence level) was described in [**12**]. The approach is based on defining a relational schema reflecting a user's view of a text. Mappings were defined that map a relational query into a query over the text, and the later one into a set of answers. These mappings essentially use morphological, syntactic and semantic properties of words, incorporated into the lexical data type [**12**] used for representing texts.

Since isolated sentences, corresponding to virtual contents of a relation, do not contain all the informations relevant for attributes of the relation, a model for knowledge-from-text representation is the relational model with null values.

In this paper, an inference algorithm is designed which provides for a query over one relation to be decomposed into a set of subqueries such that an answer to each of the subqueries may be searched for in tuples having not null values on target attributes (i.e., in sentences containing both qualification and target attributes). The inference algorithm defined accomplishes retrieval of a connected text.

Logical conditions providing for equivalence of the original query and decomposed queries is having an MVD held between qualification and target attributes of subqueries. Definitions and properties of functional and multivalued dependencies in the relational model with two types of null values are given.

Two measures of quality of answers to a given query over a text are defined: precision and recall. A proof is given that precision and recall of a set of text-level answers obtained by the inference algorithm directly depends on a precision and recall of a mapping of a relational query into a sentence-level textual query and a mapping of the textual query into answers on a sentence level.

Appendix

Proof of Theorem 1. Using the QUEL semantics [**8**], queries $(q1)$ and $(q2)$ can be interpreted, in the model with two types of null values, as follows:

$(q1)$ (a) restrict the relation$R$ using the condition $\tau(e.A \doteq \alpha) = T$;
(b) project on C;
(c) eliminate duplicates.

$(q2)$ (a') make a Cartesian product$R \times R$;
(b') restrict the result of (a')
using the condition $(u.B = e.B \;\&\; \tau(e.A \doteq \alpha) = T)$;
(c') project on C ;
(d') eliminate duplicates.

In the extended relational algebra [**11**], queries $(q1)$ and $(q2)$ may be expressed as the following expressions:

$(q1)$ $R[A =_T \alpha][C]$;
$(q2)$ $(R[A =_T \alpha][B] *_T R)[C]$.

Since attributes from B do not take ω value,

(1) $R[A, B, C] \subseteq R[A, B] *_T R[B, C]$ implies:

$$R[A, B, C][A =_T \alpha] \subseteq (R[A, B] *_T R[B, C])[A =_T \alpha] = R[A =_T \alpha][A, B] *_T R[B, C]$$

(exchanging the order of operations, [**11**]). Thus,

$$\begin{aligned} R[A, B, C][A =_T \alpha][C] \subseteq & (R[A =_T \alpha][A, B] *_T R[B, C])[C] = \\ & (R[A =_T \alpha][B] *_T R)[C] (\equiv (q_2)). \end{aligned}$$

Since $R[A, B, C][A =_T \alpha][C] = R[A =_T \alpha][C] (\equiv (q_1))$, the inclusion $\mathbf{answer}(q1) \subseteq \mathbf{answer}(q2)$ holds.

(2) If an MVD $B \rightarrow\rightarrow C$ holds in the relation $R[A, B, C]$, then $\tau(R[A, B] *_T R[B, C] \cup R[A, B] *_\omega R[B, C] \doteq R[A, B, C]) \in \{T, \omega\}$ holds, too (characterization theorem). Since attributes from B do not take ω value, $R[A, B] *_\omega R[B, C] = \emptyset$, and the equality $\tau(R[A, B] *_T R[B, C] \doteq R[A, B, C]) \in \{T, \omega\}$ implies the extended equality $\tau(\mathbf{answer}(q1) \doteq \mathbf{answer}(q2')) \in \{T, \omega\}$, instead of the corresponding inclusion in (1);

(3) trivially follows from the definition of FD.

Proof of Theorem 2. (1), (3) Analogously to the proof of the sections (1), (3) of Theorem 1, except for the step (a') of the query's $(q2)$ QUEL semantics, which now consists of Cartesian product of $k+1$ relations R, and the step (b') representing now the whole truth formula from the query $(q2)$;

(2) The proof based on mathematical induction will be applied:

- case $k=1$ is identical to the section (2) of Theorem 1;

- let an inductive hypothesis holds for k , i.e., the statement (2) of Theorem 2 holds for k;

- case $k+1$ will be proved, i.e., for the query

$(q2')$ range of e_1 is R
range of e_2 is R
$\vdots$
range of e_{k+2} is R
retrieve $(e_{k+2}.C)$ where
$e_{k+2}.B_{k+1} = e_{k+1}.B_{k+1}$ and $e_{k+1}.B_k = e_k.B_k$ and $\ldots$ and $e_1.A = \alpha$,

a sufficient condition for the extended equality $\tau(\mathbf{answer}(q1) \doteq \mathbf{answer}(q2')) \in \{T, \omega\}$ to hold is to have the set of MVDs $B_1 \to\to B_2,\ B_2 \to\to B_3,\ \ldots,\ B_k \to\to B_{k+1},\ B_{k+1} \to\to C$, held in the relation $R[A, B_1, B_2, \ldots, B_k, B_{k+1}, C]$.

Semantisc of the query $(q2')$ is the following:

$(q2')$ (a) make a Cartesian product $R \times R \times \cdots \times R$ of $k+2$ relations R;
(b) restrict according to the truth formula $e_{k+2}.B_{k+1} = e_{k+1}.B_{k+1}$ and $e_{k+1}.B_k = e_k.B_k$ and $\ldots$ and $\tau(e_1.A \doteq \alpha) = T$;
(c) project onto $e_{k+2}.C$;
(d) eliminate duplicates,

which is equivalent to the semantics:

(a') make a Cartesian product $R \times R \times \cdots \times R$ of $k+1$ relations R;
(b') restrict according to the truth formula $e_{k+1}.B_k = e_k.B_k$ and $e_k.B_{k-1} = e_{k-1}.B_{k-1}$ and $\ldots$ and $\tau(e_1.A \doteq \alpha) = T$;
(c') project onto $e_{k+1}.B_{k+1}$;
(d') make a Cartesian product with R;
(e') restrict according to the formula $e_{k+2}.B_{k+1} = e_{k+1}.B_{k+1}$;
(f') project onto $e_{k+2}.C$;
(g') eliminate duplicates.

The semantics also corresponds to the following queries:

$(q2'_1)$ (steps (a')–(c')) range of e_1 is R
range of e_2 is R

$\vdots$

range of e_{k+1} is R

retrieve $(e_{k+1}.B_{k+1})$ where $e_{k+1}.B_k = e_k.B_k$ and

$e_k.B_{k-1} = e_{k-1}.B_{k-1}$and ... and $e_1.A = \alpha$,

(whose answer is $\mathbf{answer}(q2'_1) = \{c_1, c_2, \ldots, c_r\}$),

$(q2'_{2i})$ (steps (d')–(g')) range of u is R

$(i = 1, 2, \ldots, r)$ retrieve $(u.C)$ where $u.B_{k+1} = c_i$.

The query $(q2'_1)$ is equivalent to the query

$(q2''_1)$ range of e is R

retrieve $(e.B_{k+1})$ where $e.A = \alpha$

(because of the inductive hypothesis, and because MVDs $B_1 \to\to B_2, \ldots, B_k \to\to B_{k+1}$ hold in $R[A, B_1, B_2, \ldots, B_{k+1}, C]$). Since all the Bs are not-ω attributes, for the query

$(q2'') = (q2''_1) \cup \{q2'_{2i}\}_{i=1,2,\ldots,r}$, the equality $\mathbf{answer}(q2') = \mathbf{answer}(q2'')$ holds.

The query $(q2'')$ semantics transformation gives:

$(q2'') \equiv$ (a'') restrict$R$ according to the truth formula $\tau(e.A \doteq \alpha) = T$;

(b'') project onto $e.B_{k+1}$;

(c'') make a Cartesian product with R;

(d'') restrict according to the truth formula $e.B_{k+1} = u.B_{k+1}$;

(e'') project onto $u.C$;

(f'') eliminate duplicates;

$\equiv$ (a''') make a Cartesian product $R \times R$;

(b''') restrict according to the formula $e.B_{k+1} = u.B_{k+1}$ and $\tau(e.A \doteq \alpha) = T$;

(c''') project onto $u.C$;

(d''') eliminate duplicates

$\equiv$

$(q2''')$ range of e is R

range of u is R

retrieve $(u.C)$ where $u.B_{k+1} = e.B_{k+1}$ and $e.A = \alpha$.

Thus the equality $\mathbf{answer}(q2''') = \mathbf{answer}(q2'')$ holds.

Since the MVD $B_{k+1} \to\to C$ holds in $R[A, B_1, B_2, \ldots, B_{k+1}, C]$, and thus holds in $R[A, B_{k+1}, C]$, the extended equality $\tau(\mathbf{answer}(q1) \doteq \mathbf{answer}(q2''')) \in \{T, \omega\}$ holds (according to the case $k = 1$). Transitivity of equality implies that $\tau(\mathbf{answer}(q2') \doteq \mathbf{answer}(q1)) \in \{T, \omega\}$, which was to be proved.

Proof of Theorem A. The algorithm A contains two steps through which it is possible to find an answer: steps 2 and 4c.

In the case of the step 2, an answer to the query is obtained as **answt** $(f(q1))$. A realistic assumption is that a probability to find directly an answer which is indirectly present in a text, is 0. Thus,

$$P(\mathbf{answt}(f(q1)) \subseteq \mathbf{answ}(q1)) = P(\mathbf{answt}(f(q1)) \subseteq \mathbf{answ'}(q1)) = \\ p \ (= \mathrm{pr}(f_1) * \mathrm{pr}(f_2) * \mathrm{pr}(f_3) * \mathrm{pr}(f_4) * \mathrm{pr}(\mathbf{answt})),$$

and so the theorem is proved for $k = 0$.

In the case of the step 4c, an answer is obtained as an answer to the query $(q2)$

$(q2)$ range of e_1 is R
range of e_2 is R
$\vdots$
range of e_{k+1} is R
retrieve $(e_{k+1}.C)$ where $e_{k+1}.B_k = e_k.B_k$ and
$e_k.B_{k-1} = e_{k-1}.B_{k-1}$ and $\ldots$ and $e_1.A = \alpha$,

which is equivalent (because of the semantics of QUEL, as in the proof of Theorem 2) to the set of $k+1$ queries

$(q2_1)$ range of e_1 is R
retrieve $\underbrace{(e_1.B_1)}_{c_1}$ where $e_1.A = \alpha$

$(q2_2)$ range of e_2 is R
retrieve $\underbrace{(e_2.B_2)}_{c_2}$ where $e_2.B_1 = c_1$

$\vdots$

$(q2_{k+1})$ range of e_{k+1} is R
retrieve $(e_{k+1}.C)$ where $e_{k+1}.B_k = c_k$.

The extended equality $\tau(\mathbf{answer}(q1) \doteq \mathbf{answer}(q2)) \in \{T, \omega\}$ follows from the steps (1), (2) of Theorems 1,2, which means that queries $(q1)$ and $(q2)$ are equivalent. Thus,

$$P(\text{answer} \in \mathbf{answ}(q1)) = P(\mathbf{answt}(f(q2)) \subseteq \mathbf{answ}(q1)) =$$

$$P(\mathbf{answt}(f(q2)) \subseteq \mathbf{answ}\ (q2)) = \prod_{i=\overline{1,k+1}} P(\mathbf{answt}(f(q2_i)) \subseteq \mathbf{answ}(q2_i))$$

(since events whose probabilities are to be calculated are independent)

$$= \prod_{i=\overline{1,k+1}} P(\mathbf{answt}(f(q2_i)) \subseteq \mathbf{answ}'(q2_i)) \text{ (as in the case of the step 2)}$$

$$p^{k+1}(= (\mathrm{pr}(f_1) * \mathrm{pr}(f_2) * \mathrm{pr}(f_3) * \mathrm{pr}(f_4) * \mathrm{pr}(\mathbf{answt}))^{k+1}),$$

which was to be proved.

REFERENCES

[1] L. Birnbaum, *Lexical ambiguity as a touchstone for theories of language analysis*, Proc. Int. Joint Conf. AI **9** (1985), 815–820.

[2] E.F. Codd, *A relational model of data for large shared data banks*, Comm. ACM **13(6)** (1970), 377–387.

[3] E.F. Codd, *Extending the relational model to capture more meaning*, ACM TODS **4(4)** (1979), 397–434.

[4] R. Fagin, *Multivalued dependencies and a new normal form for relational databases*, ACM TODS **2(3)** (1977), 262–278.

[5] B. Goldstein, *Constraints on null values in relational databases*, Proc. IEEE COMPSAC (1981), 101–110.

[6] J. Grant, *Null values in a relational data base*, Information Processing Letters **6(5)** (1977), 156–157.

[7] E. Hajičova et al, *Computer applications of linguistics in Prague*, Informatica **1(1)** (1982), 59–66.

[8] G.D. Held et al, *INGRES - A relational data base system*, Proc. Nat. Comp. Conf. (1975), 409–416.

[9] M. Lebowitz, *Researcher: An experimental intelligent information system*, Proc. Int. Joint Conf. AI **9** (1985), 858–862.

[10] W. Lipski, *On databases with incomplete information*, Journal ACM **28** (1981), 41–70.

[11] G. Pavlović, *On the operations over relations in the relational model of data with two types of null values*, Publ. Inst. Math. (Beograd) **34(48)** (1983), 151–163.

[12] G. Pavlović-Lažetić, E.Wong, *Managing text as data*, Proc. VLDB **12** (1986), 111–116.

[13] G. Pavlović-Lažtić, *Databases and Expert Systems in Managing Text* (in Serbo-Croat.), Ph.D. thesis, Beograd 1988.

[14] L. Siklóssy, *Efficient query evaluation in relational data bases with missing values*, Information Processing Letters **13(4,5)** (1981), 160–163.

[15] R. Simmons, D. Chester, *Relating sentences and semantic networks with procedural logic*, Comm. ACM **25(8)** (1982), 527–547.

[16] Y. Vassiliou, *Functional dependencies and incomplete information*, Proc. VLDB **6** (1980), 260–269.

Matematički fakultet
Studentski trg 16
11001 Beograd, p.p. 550
Jugoslavija

(Received 18 02 1992)