Title
ABBREVIATED QUERY INTERPRETATION IN ENTITY-RELATIONSHIP ORIENTED UNIVERSAL--RELATION DATABASES

Permalink
https://escholarship.org/uc/item/67p1h252

Authors
Markowitz, V.M.
Shoshani, A.

Publication Date
1988-02-01
Information and Computing
Sciences Division

Abbreviated Query Interpretation in
Entity-Relationship Oriented
Universal-Relation Databases

V.M. Markowitz and A. Shoshani

February 1988

TWO-WEEK LOAN COPY
This is a Library Circulating Copy
which may be borrowed for two weeks.

Prepared for the U.S. Department of Energy under Contract Number DE-AC03-76SF00098.
DISCLAIMER

This document was prepared as an account of work sponsored by the United States Government. While this document is believed to contain correct information, neither the United States Government nor any agency thereof, nor the Regents of the University of California, nor any of their employees, makes any warranty, express or implied, or assumes any legal responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by its trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof, or the Regents of the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof or the Regents of the University of California.
ABBREVIATED QUERY INTERPRETATION
IN ENTITY-RELATIONSHIP ORIENTED
UNIVERSAL-RELATION DATABASES

Victor M. Markowitz and Arie Shoshani

Information & Computing Sciences Division
Computing Sciences Research & Development
Lawrence Berkeley Laboratory
1 Cyclotron Road
Berkeley, California 94720

February 1988

This work was supported by the Director, Office of Energy Research, Applied Mathematical Sciences Research Program, of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098.
ABBREVIATED QUERY INTERPRETATION IN
ENTITY-RELATIONSHIP ORIENTED UNIVERSAL-RELATION DATABASES

Victor M. Markowitz and Arie Shoshani
Computer Science Research Department
Lawrence Berkeley Laboratory, Berkeley, CA 94720

Abstract. Entity-Relationship (ER) oriented interfaces have gained in popularity both as design
and query interfaces. ER queries are based on specifying navigation paths over ER diagrams.
Guided (automatic) navigation in ER-diagrams could enable the expression of concise queries by
users with partial or no knowledge of the diagram. We explore in this paper the methodologies
for supporting guided navigation, by using and extending results of the Universal Relation (UR)
theory. UR interfaces have been developed in order to free the user from logical navigation, that
is, from knowing what is the logical structure of the database. The main result of this effort is
the UR methodologies for the interpretation of abbreviated queries in relational databases. We
propose a new criterion for interpreting abbreviated queries, namely the existence of a side effect
free update mapping for the view implied by the respective query. Based on this criterion we
propose a new UR methodology for abbreviated query interpretation. Subsequently, we use the
underlying UR formalism in developing an ER-oriented graph procedure for determining the
most plausible navigation paths in ER diagrams.

1. Introduction.

1.1 Motivation.

There has been a great deal of interest and work on providing simple, easy to use data models,
query languages and user interfaces for database management systems. The relational model was
an important step in achieving these goals by using an easy to understand table structure (the
relation), and providing a mathematical basis for manipulating these tables. However, the rela-
tional approach leaves the burden of designing the correct (normalized) database on the user, and
a large body of theory was developed to cope with the issue of normalization. Even if the prob-
lem of database design is overlooked or delegated to experts, such as database administrators,
there is an additional burden placed on users in writing queries: they need to give the explicit
association of relations by using join expressions.

As database systems grow in complexity, it becomes increasingly difficult for users to understand and remember the database schema, and to formulate queries. The capability of avoiding explicit joins, and in general the ability to write concise queries is a desirable goal. Consider the following example. Suppose that divisions consists of departments, departments have projects, and projects are assigned to employees. A user who does not know, or remember, the details of such a structure (or just being lazy) may want to ask the query "how many employees over 40 work in the Computer Science division". One would like an intelligent system to generate an equivalent query that includes the intermediate elements projects and departments as well as the join expressions connecting them. We refer to such concise queries as abbreviated queries.

There are two basic approaches that have been taken by researchers in order to deal with the above problems of database design and query expressions. The first is to use data models that have more semantic intuition. Several proposals have been made for models that capture more semantics, from the enrichment of the relational model with more meaning [Codd], to the development of new semantic models (cf. [BG]). One model that is gaining in popularity is the Entity-Relationship (ER) model [Chen], and variations of it (e.g. [TYF]). It is generally agreed that the concepts of entities, relationships, and their properties, are more natural in designing databases for users who are not data management experts. Furthermore, since in ER schemas the associations between entities are represented explicitly by relationships, queries based on the ER model can avoid the explicit expression of joins [MR]. In general, ER schemas seem a good candidate for supporting abbreviated queries. The problem is that the ER model lacks a formal foundation, and there is no theory that gives guidance on how to construct the full query from an abbreviated query.

The second approach is the Universal Relation (UR) model [Ull83]. In contrast to the first approach, the UR model has a rich theoretical foundation, but is devoid of explicit semantics, thus providing user independence from the logical database structure. UR schemas consist of unnamed sets of attributes, and the user can express queries by specifying the attributes of interest only, leaving the interpretation of the query, to the UR system, that is, the task of finding the necessary joins connecting the attributes. The UR approach generated many works that suggest ways of interpreting such abbreviated queries (a survey is provided by [MRW]). The UR approach lead to interesting techniques of query interpretation, but UR databases proved to be very difficult to design, mainly because the database semantics are now implied by the attribute names. Consequently, the attribute naming becomes a critical problem, and the
requirement to know and understand the semantic structure is replaced, for the user, by the requirement to manage a names structure.

1.2 Summary Of Results.

The purpose of this paper is two-fold: first, to propose a new, intuitively appealing, methodology for query interpretation in UR databases; second, to apply this methodology to the ER model. Thus, we combine the benefit of a theoretical basis in the UR model with the attractiveness of the ER model as a user interface. We discuss each of these topics below.

Most UR methodologies for query interpretation consist of choosing interpretations that involve only lossless joins [MRW]. In fact, these methodologies attempt to generate the view that corresponds to the user's perception of the database. While losslessness has attractive mathematical properties, the intuition for using it in generating views is not obvious. The major problem with views is the difficulty of mapping updates expressed over the view, into updates of base relations. This problem has been studied extensively (e.g. [DB], [Kel]). Side effect freeness, meaning that the underlying updates must produce no effects beyond the intended update on the view, is generally accepted as ensuring the correctness of the interpretation given to updates over views. Indeed, users would expect their views to behave under updates as any basic object in the database, that is, without side effects. We investigate how the view update problem is reflected in UR databases, and propose the existence of a side effect free view-update mapping as the criterion to guide query interpretation. Interestingly, it turns out that the losslessness condition is equivalent to the existence of a side effect free delete mapping. However, the existence of side effect free update (insert and delete) mappings is a stronger condition and has clearer intuition than the losslessness of joins.

Our second purpose is to apply the above extended UR query interpretation methodology to the ER model. Consider ER schemas as graphs (or diagrams). As has been observed by many authors (e.g. [WK]), an ER query can be seen as selecting a subgraph of the ER diagram, and associating predicate conditions with the diagram vertices. Such a subgraph corresponds to interpreting an ER query since it represents all the joins necessary to evaluate the query in the underlying ER-consistent UR database. We use the term query-subgraph to denote a connected subgraph of the ER diagram that represents an ER query. An abbreviated ER query can be represented by disconnected, rather than connected, subgraphs. This can be viewed as a shortcut to the specification of a full query, which obviously simplifies the user task of expressing queries. Thus, in the example mentioned above, namely "how many employees over 40 work in the
Computer Science division", one could take advantage of the information in the ER diagram and generate the ER query-subgraph that includes the intermediate entities projects and departments as well as the relationships connecting them.

In general, every abbreviated ER query may be associated with several query-subgraphs, depending on the complexity of the ER diagram. Naturally, we would like to select the most plausible query-subgraph. As mentioned above, we want to take advantage of the extensive work done on query interpretation in the context of the UR model. We base our work on previously developed techniques [Mar] for associating UR databases with ER schemas. We call UR databases that are associated with ER schemas ER-consistent databases. In ER-consistent databases a view implied by some query is represented by a query-subgraph. By associating UR databases with ER schemas, the UR methodologies for query interpretation can be extended to ER queries.

We propose a methodology for the interpretation of abbreviated ER queries. Based on an underlying UR formalism, we develop an ER-oriented graph procedure for determining plausible query-subgraphs in ER diagrams. We grade the plausibility of the query-subgraphs as follows: the query-subgraph is (0) least plausible if it corresponds to a view that has no side effect free update mapping; (1) plausible if it corresponds to a view that has a side effect free delete mapping but no side effect free insert mapping (or equivalently, involves only lossless joins); and (2) most plausible if it corresponds to a view that has a side effect free update (delete and insert) mapping. By using several levels of plausibility, where the highest level corresponds to a condition stronger than join losslessness, our procedure is more selective than analogous UR methodologies. Conversely, unlike analogous UR methodologies based only on join losslessness, the weakest plausibility guarantees at least one query-subgraph for any query.

Another work on the application of the UR methodologies for query interpretation to the ER model, that is worth mentioning here, is [ZM]. It basically uses the UR methodology of [MU], that is based on the same criteria of join losslessness, to select navigation paths in ER diagrams. Actually, [ZM] takes the UR concepts and applies them directly to the ER schema in an informal way. However, [ZM] does not provide the precise association of UR concepts, such as object and lossless join, with ER concepts. Unlike [ZM], we use a precise correspondence of UR and ER concepts. In addition, the UR methodology used by us is an extended methodology, as we have mentioned above.

The rest of the paper is organized as follows. We briefly review the necessary relational and UR concepts, and the ER diagram, in sections 2 and 3 respectively. ER-consistency for UR
schemas is discussed in section 4. In section 5, we investigate the update behavior of ER-consistent UR databases. In section 6, we propose a new criterion for interpreting UR abbreviated queries. Based on this extended UR formalism, we develop in section 7 a methodology for interpreting abbreviated ER queries. We close the paper by summarizing the results and drawing some conclusions.

2. Preliminary Definitions.

We use in our paper basic graph theoretical concepts. Any textbook on graph theory, such as [Even], can provide the necessary reference. We mention below only the concepts of root and induced subgraph of a digraph (directed graph). Let $G = (V, E)$ denote a digraph with $V$ its set of vertices, and $E$ its set of edges;

(i) $X \in V$ is a root for $G$ iff for every $Y \in V$, there is a dipath (directed path, meaning that all the edges on the path have the same direction) from $X$ to $Y$ in $G$;

(ii) The subgraph induced by any $V' \subseteq V$, denoted $G(V')$, is defined as follows: $G(V') = (V', E')$, where $E' = \{(X_i, X_j) | X_i, X_j \in V' \text{ and } (X_i, X_j) \in E\}$.

Now, we briefly review the concepts of the relational and UR models referred to in the paper. For details and an extensive bibliography the reader is referred to [Ull82] for both models, and to [MRW] and [Ull83] for the UR model.

A Universal Relation schema, $(O, K)$, is a finite set $(O)$ of objects, together with a set of key dependencies $(K)$. An object $O_i \in O$ is an unnamed set of attributes. All the objects are assumed to be subsets of a universal set of attributes $U$. Objects are either basic or non-basic. Intuitively basic objects represent nondecomposable facts, while non-basic objects represent meaningful, but decomposable, facts, spanning over several basic objects. On the semantic level, every attribute is assigned a domain. An UR database state, $u$, consists of a set of base relations, $<r_1...r_k>$, one for every basic object, $O_i \in O$, of the UR schema $(O, K)$.

Relations are manipulated by relational algebra operators (cf. [Ull82]). Let $O_i$ and $O_j$ be two objects associated with relations $r_i$ and $r_j$, respectively. We denote by $t$ a tuple; and by $t[W]$ the sub-tuple of $t$ corresponding to the attribute set $W$. The following table summarizes the definitions of the algebraic operations we shall use in the paper.
<table>
<thead>
<tr>
<th>RA Operation</th>
<th>Syntax</th>
<th>Prerequisite</th>
<th>Derived Object</th>
<th>Derived Relation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Projection</td>
<td>$\prod_W(r_i)$</td>
<td>$W \subseteq O_i$</td>
<td>$O_r := W$</td>
<td>{{t[W] \mid t \in r_i} }</td>
</tr>
<tr>
<td>Union</td>
<td>$r_i \cup r_j$</td>
<td>$O_i = O_j$</td>
<td>$O_r := O_i$</td>
<td>{{t \mid t \in r_i \text{ or } t \in r_j} }</td>
</tr>
<tr>
<td>Natural Join</td>
<td>$r_i \bowtie r_j$</td>
<td>$O_i \cap O_j \neq \emptyset$</td>
<td>$O_r := O_i \cup O_j$</td>
<td>{{t \mid [O_r] \in r_i \text{ and } t[O_r] \in r_j} }</td>
</tr>
</tbody>
</table>

Given an object set $O$ and an object $O_i \in O$,

(i) the key dependency over $O_i$ is a statement of the form $K_i \rightarrow O_i$, where $K_i \subseteq O_i$; $K_i \rightarrow O_i$ is valid in a UR state $u$ if and only if for any two tuples of $r_i$, $t$ and $t'$, $t \neq t'$ implies $t[K_i] \neq t'[K_i]$. $K_i$ is called key;

(ii) the superobject of object $O_i$ is any object $O_j \in O$ such that $O_i \subseteq O_j$; a superobject of $O_i$, $O_j \in O$, is said to be a minimal superobject of $O_i$ iff $\not\exists O_k \in O$ such that $O_k \subset O_i \subset O_j$; the set of superobjects of $O_i$ is denoted $\text{Sup}(O_i)$, and the set of minimal superobjects of $O_i$ is denoted $\text{Sup}(O_i)$;

\quad e.g. in figure 1, $\text{Sup}(C) = \{(SC), (TC), (TCS)\}$, and $\text{Sup}(C) = \{(SC), (TC)\}$;

(iii) the subobject of object $O_i$ is any object $O_j \in O$ such that $O_j \subset O_i$; a subobject of $O_i$, $O_j \in O$, is said to be a minimal subobject of $O_i$ iff $\not\exists O_k \in O$ such that $O_k \subset O_i \subset O_j$; the set of subobjects of $O_i$ is denoted $\text{Sub}(O_i)$, and the set of minimal subobjects of $O_i$ is denoted $\text{Sub}(O_i)$;

\quad e.g. in figure 1, $\text{Sub}(TCS) = \{(S), (C), (SC), (TC)\}$, and $\text{Sub}(TCS) = \{(SC), (TC)\}$.

\begin{figure}
\centering
\includegraphics[width=0.5\textwidth]{fig1.png}
\caption{Object Graph Example.}
\end{figure}
We propose the following graph representation for a set of objects.

**Definition 2.1 — Object Graph** (example in figure 1).

Let $O$ be an object set. The *object graph* associated with $O$, is the digraph $G_O = (V, E)$, where $V = O$ and $O_i \rightarrow O_j$ is an edge of $E$ iff $O_i$ is a minimal superobject of $O_j$, that is, $O_i \in \text{Sup}(O_j)$.

Legal states have to satisfy the *containment condition* [MRW]: let $O_i$ and $O_j$ be two objects and $r_i$ and $r_j$ be their associated relations, respectively; if $O_j$ is a superobject of $O_i$ then $\Pi_{O_i}(r_j) \subseteq r_i$.

**Definition 2.2 — Non-Basic Object.**

A *non-basic object*, $O_i$, is defined as the union of basic objects, such that the subgraph of $G_O$ induced by $\text{Sub}(O_i)$ is connected. A non basic-object $O_i$ is associated with a relation, $r_i$, that is defined as the *natural join* of all the relations associated with the basic subobjects of $O_i$:

$$r_i = \bigwedge_{O_j \in \text{Sub}(O_i)} r_j.$$

3. **Entity-Relationship Diagram.**

Entity-Relationship concepts [Chen] reflect a natural, although limited, view of the world: entities are qualified by their attributes and interactions between entities are expressed by relationships. An *entity-set* groups entities of the same type, where the entity-type is characterized by the sharing of the same set of attributes. A relationship represents the interaction of several entities, and relationships of the same type are grouped in a *relationship-set*. A subset of the attributes associated with an entity-set may be specified as the *entity-identifier*, which is used to distinguish among the occurrences of an entity-set. *Association-cardinalities* are restrictions on the minimum and maximum number of entities from a given entity-set, that can be related, in the context of some relationship-set, to a specific combination of entities from all the other entity-sets involved in that relationship-set.

ER-schemas are expressible in a diagrammatic form called *ER-diagram* (ERD) which we define as a directed graph (see example in figure 2). Entity-sets, relationship-sets, and attributes of entity-sets or relationship-sets, are represented by entity, relationship and attribute vertices, respectively. Entity, relationship and attribute vertices, are denoted as e-vertices, r-vertices, and a-vertices, respectively, and are represented graphically by rectangles, diamonds, and ellipses,
respectively. ERD vertices are connected by directed edges. Every vertex is labeled by the name of the associated entity-set, relationship-set, or attribute; e-vertices and r-vertices are uniquely identified by their labels globally, while a-vertices are uniquely identified by their labels only locally, within the set of a-vertices connected to some e-vertex or r-vertex. The attribute-reduced ERD is an ERD with the a-vertices, and all their incident edges, removed.

We use an extended ERD representation that includes generalization hierarchies. For the sake of simplicity we do not allow the association of entities from the same entity-set (a formal definition of this restriction is given later, in constraint ER3 of definition 3.3), and leave out the representation of existence dependencies between relationship-sets that has been considered in [Mar]. Also, in order to simplify the presentation, we do not include in our model relationship-sets that have their own attributes, but our results are just as valid if this restriction is removed. We shall refer only to maximum association cardinalities of one and greater than one, called many. We assume that every relationship-set involves at least one entity-set with a many association-cardinality.

Basic notations:

- $G_{ER}$ and $\overline{G}_{ER}$ denote an ER diagram and the corresponding attribute-reduced ER diagram;
- $A_i, E_i, R_i$ denote an a-vertex, e-vertex, and r-vertex, respectively;
- $X_i \rightarrow X_j$ denotes an edge between vertices $X_i$ and $X_j$;
- $X_i \rightarrow\rightarrow X_j$ denotes a dipath between vertices $X_i$ and $X_j$;
- $R_i \rightarrow E_j$ and $R_i \rightarrow\rightarrow E_j$ denote the association-cardinalities of many and one respectively, of

![Entity-Relationship Diagram Example](image)

**Fig.2** Entity-Relationship Diagram Example (*identifiers are underlined*).
entity-set $E_j$ in relationship-set $R_i$.

ERD edges specify existence constraints, depending on the type of the connected vertices:

$(A_i \rightarrow E_j)$ an attribute does not exist independently, but only related to some entity-set $E_j$;

$\text{ISA}$

$(E_i \rightarrow E_j)$ the ISA relationship expresses a generalization relationship between two entity-sets; $E_i$ is said to be a specialization of $E_j$, and $E_j$ is said to be a generic entity-set (generalization) of $E_i$;

$\text{ID}$

$(E_i \rightarrow E_j)$ the ID relationship expresses an identification relationship between an entity-set, $E_i$, called weak entity-set, which cannot be identified by its own attributes, but has to be identified by its relationship with another entity-set, $E_j$; $E_i$ is said to be a dependent of $E_j$;

$(R_i \rightarrow E_j)$ relationship-set $R_i$ involves entity-set $E_j$, therefore a relationship in $R_i$ exists provided the related entity from $E_j$, also exists.

Additional (derived) notations:

- $E_i \overset{\text{ISA}}{\rightarrow} E_j$ denotes a dipath of ISA-edges;

- $\text{Attr}(E_i) \overset{\text{ISA}}{=} \{ A_j \mid A_j \rightarrow E_i \in G_{ER} \}$, denotes the set of a-vertices connected to an e-vertex $E_i$;

- $\text{Id}(E_i) \subseteq \text{Attr}(E_i)$ denotes the entity-identifier specified for e-vertex $E_i$;

- $\text{GEN}(E_i) \overset{\text{ISA}}{=} \{ E_k \mid E_i \rightarrow E_k \in G_{ER} \}$, denotes the set of generalizations of entity-set $E_i$;

- $\text{ENT}(E_i) \overset{\text{ID}}{=} \{ E_k \mid E_i \rightarrow E_k \in G_{ER} \}$, denotes the set of entity-sets on which entity-set $E_i$ is ID-dependent;

- $\text{ENT}(R_i) \overset{\text{ID}}{=} \{ E_k \mid R_i \rightarrow E_k \in G_{ER} \}$, denotes the set of entity-sets associated by relationship-set $R_i$;

- $\text{ENT} \rightarrow \text{ENT}^*$ denotes the existence of an 1-1 correspondence, $C$, between the e-vertices of two sets of e-vertices, ENT and ENT*, belonging to an ERD, $G_{ER}$:

$$C = \{ (E_i, E_j) \mid E_i \in \text{ENT} \land E_j \in \text{ENT}^* \text{ and ( either } \overset{\text{ISA}}{E_i \rightarrow E_j \in G_{ER}} \text{ or } E_i = E_j \} \}.$$ 

Definition 3.1 — Specialization Cluster.

Let $G_{ER}$ be an ERD, and $E_i \in G_{ER}$, an e-vertex; the specialization cluster rooted in $E_i$, $\text{SPEC}^*(E_i)$, is the set of all the e-vertices representing specializations of the entity-set represented by $E_i$: $\text{SPEC}^*(E_i) = E_i \cup \{ E_j \mid \overset{\text{ISA}}{E_j \rightarrow E_i \in G_{ER}} \}$. 
If $E_i$ has no generalization, that is, $GEN(E_i) = \emptyset$, then the specialization cluster is said to be maximal.

**Definition 3.2 — Uplink.**

Let $G_{ER}$ be an ERD, and $E_i$ an e-vertex of $G_{ER}$. $E_i$ is said to be an **upper link** (uplink) of the e-vertex set $\Lambda = \{E_j \mid E_j \in G_{ER}\}$, iff $\forall E_j \in \Lambda : E_j \rightarrow E_i \in G_{ER}$ (possibly of length 0), and there is no $E_k$ ($k \neq i$), such that $E_k \rightarrow E_i \in G_{ER}$ and $\forall E_j \in \Lambda : E_j \rightarrow E_k \in G_{ER}$. The set of all uplinks of a set of e-vertices $\Lambda$, is denoted $uplink(\Lambda)$.

In figure 2, for instance, $uplink(SCIENTIST, EMPLOYEE)$ is $\{EMPLOYEE\}$.

**Definition 3.3 — Entity—Relationship Diagram.**

An *Entity—Relationship Diagram (ERD)* is a finite labeled digraph $G_{ER} = (V, H)$, where $V$ is the disjoint union of three subsets of vertices: $E$ (e-vertices), $R$ (r-vertices), and $A$ (a-vertices); $H$ is the set of directed edges, where an edge can be of one of the following forms: $A_i \rightarrow E_j$, $E_i \rightarrow E_j$, and $R_i \rightarrow E_j$. $G_{ER}$ obeys the following constraints:

(ER1) $G_{ER}$ is an acyclic digraph without parallel edges;

(ER2) $\forall E_i \in G_{ER} : outdegree(E_i) = 1$;

(ER3) for any e-vertex/r-vertex $X_i$ holds:

$$\forall (E_j, E_k) \in ENT(X_i) \times ENT(X_i) : uplink(E_j, E_k) = \emptyset;$$

(ER4) $\forall E_i \in G_{ER}$: if $GEN(E_i) \neq \emptyset$ then $Id(E_i) = \emptyset$; $ENT(E_i) = \emptyset$; and $E_i$ belongs to a **unique maximal specialization cluster**;

otherwise $Id(E_i) \neq \emptyset$;

(ER5) $\forall R_i \in G_{ER} : ENT(R_i) \geq 2$.

Constraint (ER1) above guarantees that directed cycles do not exist so that, for instance, an entity-set will neither be defined as depending on identification on itself, nor be defined as a

---

![Diagram](image-url)

**Fig.3** Proper Subgraph of Entity—Relationship Diagram of Figure 2.
proper subset of itself. Constraint (ER2) insures that an attribute characterizes a single entity-set. Constraint (ER3) states the restriction of not representing associations between entities from a same entity-set. The rules of identifier specification are given by constraint (ER4); (ER4) also states that every generalization hierarchy is a rooted tree.

We need the following definition, of proper ERD subgraphs, in the next section.

**Definition 3.4 — Proper ER—Diagram Subgraph.**

Let \( G_{ER} = (V, H) \) be a ERD. \( G'_{ER} = (V', H') \) is said to be a proper ERD subgraph of \( G_{ER} \) iff \( V' \subseteq V \), \( H' \subseteq H \), and for any vertex \( X_i \in V' \):

\[
ENT'(X_i) \equiv ENT(X_i) \text{ and } GEN'(X_i) \equiv GEN(X_i).
\]

A proper ERD subgraph of some ERD is a well-defined ERD on its own, which also preserves the structural definition of all its entity-sets and relationship-sets. Thus, in an ERD subgraph a relationship-set involves the same entity-sets, a weak entity-set remains dependent on the same entity-sets, and a specialization entity-set is generalized by the same generic entity-sets, as in the original ERD. An example of a proper ERD subgraph is given in figure 3.

### 4. Entity—Relationship Consistent Universal—Relation Schemas.

We have proposed the ERD as a higher-level schema on top of both relational and UR databases [Mar]. A UR (or relational) schema which is the translation of an ERD, is said to be (trivially) *ER-consistent*. Accordingly, the state associated with an ERD is the state of its UR (or relational) translation. An UR (or relational) database whose schema is ER-consistent, is said to be an *ER-consistent database*.

The main restriction of UR schemas is their inability to represent directly multiple relationships between attributes: the *Unique Role Assumption* [MRW] requires that among any set of attributes there exists at most one relationship. In order to partially overcome this restriction we first map the ERD into an intermediary ERD, in which a relationship-set is uniquely defined by the involved entity-sets so that the relationship-set name need not be used. A *unique-role ERD (UERD)* is an ERD in which there is no pair of relationship-sets, \( (R_i, R_j) \), such that there exists a subset of \( ENT(R_i) \), \( ENT' \), with \( ENT' \rightarrow \rightarrow ENT(R_j) \). We shall assume that every entity-set has at least one attribute. The mapping of an ERD into an UERD is straightforward: for a pair of relationship-sets, \( (R_i, R_j) \), where \( \exists ENT' \subseteq ENT(R_i) \) such that \( ENT' \rightarrow \rightarrow ENT(R_j) \), break the correspondence by introducing for anyone of the corresponding entity-sets, two specialization entity-sets having single—valued attributes designating their roles.
(role−attribute). An example is given in figure 4. Note that the transformations must be done only until the ERD becomes an UERD, and not for all the involved entity-sets (e.g. DEPARTMENT in figure 4).

The mapping of an UERD into an UR schema, $T_u$, is presented, in figure 5, and exemplified in figure 6. The mapping $T_u$ is total, that is, any ERD is translatable to an UR schema of the form $(O, K)$, where $O$ is a set of objects and $K$ is a set of key dependencies. We have shown in [Mar] that $T_u$ insures that the keys and the containment condition represent the ER structure as expected, and that the key structure is sufficient to characterize entity-sets and relationship-sets, and to express ISA hierarchies.

**T. : Mapping ER-Diagram Into Universal Relational Schema**

*Input*: $G_{ER} = (V, H)$, an ERD;  
*Output*: the UR schema $(O, K)$ interpreting $G_{ER}$;

1. prefix the labels of all $e$-vertices by the label of corresponding $e$-vertex/$r$-vertex, (for global uniqueness);
2. map ERD $G_{ER}$ into a unique role ERD, $G^u_{ER}$ (mapping $M_{UERD}$);
3. for every $e$-vertex/$r$-vertex $X, \in G^u_{ER}$ define recursively the following set of $a$-vertices:
   
   $e$-vertex/$r$-vertex:
   
   $\text{Attr}^*(X_i) = \text{Attr}(X_i) \bigcup_{X_i \rightarrow X_j \in \sigma_{ER}} \text{Attr}^*(X_j)$;

   $e$-vertex:
   
   $\text{Key}(X_i) = \text{Id}(X_i) \bigcup_{X_i \rightarrow X_j \in \sigma_{ER}} \text{Key}(X_j)$;

   $r$-vertex:
   
   $\text{Key}(X_i) = \bigcup_{X_i \rightarrow X_j \in G_{ER}} \text{Key}(X_j)$;

4. for every $e$-vertex/$r$-vertex $X, \in G^u_{ER}$: define object $O, := \text{Attr}^*(X_i)$; $K, := \text{Key}(X_i)$;

   \[O := O \cup O_i, \quad K := K \cup \{K_i \rightarrow O_i\} \]

*Fig.5* Mapping Entity-Relationship Diagram Into Universal-Relation Schema.
Proposition 4.1. Let \((O, K)\) be the UR translation of UERD \(G_{ER}^{u}\), and \(G_O\) its associated object graph. Then

(i) \(G_O\) is isomorphic to the attribute-reduced UERD \(\overline{G}_{ER}^{u}\) (Proposition 4.5 [Mar]).

(ii) Let \(O_i\) be a non-basic object; the object subgraph induced by \(\text{Sub}(O_i), G_O(\text{Sub}(O_i))\), is isomorphic to a connected proper subgraph of \(\overline{G}_{ER}^{u}\). Conversely, for any proper connected subgraph of \(\overline{G}_{ER}^{u}\), either it has a root, and then the root corresponds to a basic object, or the subgraph corresponds to a non-basic object.

There is no established design methodology for UR schemas. UR schemas are difficult to design, mainly because most of the explicit structure of the schema is implied by the attribute

\[ UR \ Schema \ (keys \ are \ underlined) : \{ O, OP, P, PD, D \} \]

Fig. 7 Universal-Relation Schema and Associated Entity-Relationship Diagram.
names. Although the selection of the objects seems to be so critical to capturing the accurate intuition, the meaning of objects is inherently obscure. In fact, it is not clear how a designer could decide on the structure of objects. Consider, for example, products \((P)\), owners \((O)\) who own products \((P)\), and dealers \((D)\) who stock products \((P)\); [MRS] proposes to represent this example by the following set of objects: \((OP, PD, P)\). We contend that no matter which are the associated keys, this UR schema has no reasonable ERD correspondent (see [Mar] for details and a reverse mapping, from UR schemas into ER diagrams). This difficulty can be overcome by associating UR schemas with ER diagrams. In ER-consistent UR schemas every basic object represents either an entity-set or a relationship-set, and the basic object structure results directly from mapping the ER diagram into the UR schema, while every non-basic object represents a proper ERD subgraph. Thus, returning to the example above, the UR schema of figure 7 is the result of directly mapping the ER diagram of figure 7.

5. Update Behavior In Universal—Relation Databases.

Our purpose in this section is to investigate under what conditions updates over non-basic objects have side-effect free mappings into updates over the underlying basic objects. We begin by first discussing the update behavior of the basic objects in an UR database.

An elementary update in an UR (or relational) database consist of: (i) modifying an attribute value in a tuple; (ii) deleting a tuple from a relation; and (iii) inserting a tuple into a relation. Note that in ER-consistent databases, every relation corresponds to an entity-set or relationship-set, and every tuple represents an entity or relationship respectively. Updates in ER-consistent databases refer to information, rather than data, structures; thus, an elementary update refers to an entity/relationship, or an attribute of an entity/relationship.

Let \( r \) be a database state associated with UR schema \((O, K)\), and \( r_i \) the relation associated with the basic object \( O_i \in O \). We shall denote the deletion/insertion of a tuple, \( t \), from/into \( r_i \) as \( \text{update}(t, r_i) \). \( \text{update}(t, r_i) \) maps \( r_i \) and \( r \), into \( r_i' \) and \( r' \) respectively. In general the consistency of \( r' \) is not assured by this mapping alone, so that additional updates must be performed, namely \( \text{delete}(t, r_i) \) propagates to the relations associated with all the superobjects of \( O_i \), and \( \text{insert}(t, r_i) \) propagates to the relations associated with all the subobjects of \( O_i \).
Definition 5.1 — Update Propagation.

Let $r$ be a database state associated with UR schema $(O, K)$, and $r_i$ the relation associated with the basic object $O_i \in O$. $update(t, r_i)$ propagates as follows:

- $delete: \forall O_j ( O_j \in Sup(O_i) )$: delete from $r_j$ the set of tuples $\{ t' | t' \in r_j \text{ and } t'[O_i] = t \}$;
- $insert: \forall O_j ( O_j \in Sub(O_i) )$: insert into $r_j$ the tuple $t_i[O_j]$.

The propagation of $update(t, r_i)$ spans all the relations associated with the objects in $Sup(O_i)$ for deletion, and $Sub(O_i)$ for insertion, respectively, and is characterized by the object subgraphs induced by $Sup(O_i)$ and $Sub(O_i)$, respectively (e.g. see figure 8). Let the set of updates triggered by $update(t, r_i)$, including $update(t, r_i)$, be denoted $update^*(t, r_i)$; $insert^*(t, r_i)$ and $delete^*(t, r_i)$ are similarly defined.

Proposition 5.1. Let $r$ be the database state associated with the UR schema $(O, K)$, and $r_i$ the relation associated with the basic object $O_i \in O$. $update^*(t, r_i)$ maps $r$ into a state that satisfies the containment condition and is minimal, that is, no proper subset of updates has this property.

Sketch of Proof: This can be verified by following the propagation of $update(t, r_i)$ in the order specified by the subgraph of the object digraph, induced by $Sup(O_i)$ and $Sub(O_i)$, for deletion and insertion, respectively.

---

(i) $delete( t, r_{ENAME} )$  (ii) $insert( t, r_{ENAME, PNAME, POSITION} )$

---

![Diagram](image)

Fig.8 Update Propagation Subgraphs over Universal-Relation Schema of Figure 6.
The UR non-basic objects are actually relational views expressed by natural joins. The major problem with views is the difficulty of interpreting view updates, that is, updates expressed over the view. Mapping view updates into updates of basic relations has been studied extensively (e.g. [DB], [Kel]). Side effect freeness, meaning that the underlying updates must produce no effects beyond the intended update on the view, is generally accepted as ensuring the correctness of the interpretation given to view updates. We shall investigate the conditions guaranteeing side effect free mappings of updates related to non-basic objects. The following definition characterizes the basic correctness criteria for the mapping of view updates [DB]:

**Definition 5.2 — Update Mapping Correctness.**

Let \( r_v \) be a relation associated with a view and derived from the database state \( r \), by some evaluation function \( Eval \), let \( u_v \) be an elementary update over \( r_v \), and let \( \{u_1 \cdots u_k\} \) be the underlying updates to which \( u_v \) is mapped; let \( r'_v = u_v(r_v) \), \( r' = \{u_1 \cdots u_k\}(r) \), and \( r'_v = Eval(r') \), as shown below:

\[
\begin{array}{c}
\xrightarrow{\text{Eval}(r)} \quad \xrightarrow{\text{Update Mapping}} \quad \xrightarrow{\text{Eval}(r')}
\end{array}
\]

Then \( \{u_1 \cdots u_k\} \) is said to

(i) perform \( u_v = \text{delete} \) iff \( r'_v \subseteq r^{'*}_v \); and perform \( u_v = \text{insert} \) iff \( r^{'*}_v \subseteq r'_v \);

(ii) exactly perform \( u_v \) iff \( r'_v \equiv r^{'*}_v \).

In other words, (i) states that the underlying updates produce the effect of the update on the view, (ii) states that the underlying updates produce no side effects beyond the update on the view.

Since the insertion of a tuple into a relation associated with some object \( O_i \), propagates to all the subobjects of \( O_i \), the update propagation alone is enough to perform an insertion related to a non-basic object. The deletion of a tuple from a relation associated with some object \( O_i \), propagates to all the superobjects of \( O_i \), therefore a deletion related to a non-basic object is performed by mapping it into deletions related to at least one of the subobjects of \( O_i \).
Definition 5.3 — Non-Basic Object Update ($T_{upd}$).

Let $r$ be the database associated with the UR schema $(O, K)$, $O_i$ a non-basic object, and $update^*(t, r_i)$ denote the set of updates triggered by $update(t, r_i)$, including $update(t, r_i)$ itself. Then we define the update over a non-basic object $O_i$ by mapping $T_{upd}$, as follows:

(i) $T_{upd}(\text{insert}(t, r_i)) \triangleq \text{insert}^*(t, r_i)$;

(ii) $T_{upd}(\text{delete}(t, r_i)) \triangleq \text{delete}^*(t, r_i) \cup \{\text{delete}^*(t', r_j) \mid O_j \in \text{Del}(O_i), t' = t\mid O_j\}$,

where $\text{Del}(O_i)$ is a nonempty subset of the minimal basic subobjects of $O_i$, that is, $\text{Del}(O_i) \subseteq \text{Sub}(O_j)$, such that $|\text{Del}(O_i)| \geq 1$.

Proposition 5.2. Let $r$ be a database associated with the UR schema $(O, K)$, $O_i$ a non-basic object associated with relation $r_i$, and $t$ a tuple to be inserted/deleted into/from $r_i$. Then $T_{upd}(update(t, r_i))$ performs $update(t, r_i)$.

Proof: follows directly from the definition 5.3.

Definition 5.4 — Side Effect Free Update Mapping.

Let $r$ be a database associated with the UR schema $(O, K)$, and $O_i$ a non-basic object associated with relation $r_i$. Then $T_{upd}$ is said to be side effect free for $O_i$ iff for any tuple $t$, $T_{upd}(update(t, r_i))$ exactly performs $update(t, r_i)$.

Proposition 5.3. Let $r$ be a database associated with UR schema $(O, K)$, and $O_i$ a non-basic object. Then $T_{upd}$ is (i) insert side effect free for $O_i$ if every minimal basic subobject of $O_i$, $O_j$, functionally determines $O_i$; and (ii) delete side effect free for $O_i$ if at least one minimal basic subobject of $O_i$, $O_j$, functionally determines $O_i$.

Sketch of Proof: The insertion related to $O_i$ propagates to the set of minimal superobjects of $O_i$, while the deletion related to $O_i$ propagates to the subset of minimal superobjects of $O_i$ belonging to $\text{Del}(O_i)$, where $\text{Del}(O_i)$ must consist of at least one minimal subobject of $O_i$. The key dependencies ensure that every tuple in a relation associated with a minimal subobject of $O_i$ corresponds to at most one tuple in the relation associated with $O_i$. The proposition follows immediately.

Proposition 5.3 shows that, for a non-basic object $O_i$, the side effect freeness under insertion depends on the specification of $O_i$, while the side effect freeness under deletion depends on how the set $\text{Del}(O_i)$ is chosen, that is, the insert side effect freeness condition is stronger than
the delete side effect freeness condition. Let, for instance, the basic objects \((AB)\) and \((BC)\), be associated with relations \(r_{(AB)} = \begin{bmatrix} 1 & 1 \\ 2 & 2 \end{bmatrix}\) and \(r_{(BC)} = \begin{bmatrix} 2 & 4 \\ 2 & 5 \end{bmatrix}\) respectively, and let the non-basic object \((ABC)\) be associated accordingly with \(r_{(ABC)} = \begin{bmatrix} 1 & 2 & 4 \\ 1 & 2 & 5 \end{bmatrix}\). Deletion side effect freeness is ensured by \(\text{Del}(ABC) = ((BC))\); the deletion of \((1 2 4)\) from \(r_{(ABC)}\), for instance, is then interpreted as the deletion of \((1 2)\) from \(r_{(BC)}\). On the other hand, insertion of \((1 3 5)\) into \(r_{(ABC)}\) is interpreted as the insertion of \((1 3)\) into \(r_{(AB)}\) and the insertion of \((3 5)\) into \(r_{(BC)}\), leading to the side-effect insertion of \((1 3 4)\) into \(r_{(ABC)}\). Note that the insert side effect freeness condition of proposition 5.3 is mentioned by [Kel] (in a different context, therefore with different concepts) as a requirement for defining views based on joins.

The side-effect freeness condition can be backtracked to the basic objects. For instance, in the previous example, key dependency \(BC \rightarrow A\), over non-basic object \((ABC)\) is implied by the key dependency \(B \rightarrow A\) over basic object \((AB)\).

6. Abbreviated Query Interpretation In Universal-Relation Databases.

UR interfaces attempt to free the user from specifying navigation details: users pose abbreviated queries, mentioning only the attributes they are interested in. UR interfaces are based on various methodologies for interpreting abbreviated queries. All these methodologies try to find a derivation corresponding to the view supposedly referred by the user. Following [MRW] we shall call such a derivation a window-function. Let \(X\) denote any set of attributes; the relation to be associated with \(X\) is generated by a window-function, defined in [MRW] as the union, of projections onto \(X\), of the relations associated with all the superobjects of \(X\):

\[
r(X) = \bigcup_{o_i \in o, o \subseteq X} \Pi_X(r_i).
\]

The window-function selects superobjects from the set of both basic and non-basic objects. Non-basic objects are treated differently by the various UR methodologies. For instance, [MRW] argues that the non-basic objects should be specified by the designer. However, the almost general idea underlying the specification of non-basic objects is to augment (extend) basic objects with other objects, following functional dependencies, so that all the implied joins would be lossless [Ull83]. This approach is based on the conjecture that a join makes sense if and only if it is lossless.

Informally, two relations have a lossless join if from the existence of a pair of matching
tuples taken from each relation one can imply the existence of a tuple that is their join. Let a join expression evaluate to a relation associated with attribute set $Z$; semantically, the join expression is said to be lossless if for each relation $u$ over the universal attribute set $U$ that satisfies the associated dependency set, when we substitute each operand of the join, $O_k$, with the projection of $u$ over $O_k$, the join evaluates to the projection of $u$ over $Z$. Syntactically, the join of the relations associated with the minimal basic subobjects of some non-basic object, is lossless whenever a key of one the minimal basic subobjects is also a key for the non-basic object (see e.g. [Ull82]). For instance, a join that is not lossless, is the join of the relations associated with objects $(AB)$ and $(BC)$; suppose that these objects are associated with relations $r_{(AB)} = \begin{bmatrix} 1 & 2 \\ 3 & 2 \end{bmatrix}$ and $r_{(BC)} = \begin{bmatrix} 2 & 4 \\ 2 & 5 \end{bmatrix}$ respectively, and let $u = \begin{bmatrix} 1 & 2 & 4 \\ 3 & 2 & 5 \end{bmatrix}$ be some UR state that satisfies the involved key dependencies; both the semantic and syntactic conditions for lossless join are obviously not met.

Since losslessness does not have a clear intuition (e.g. see [AP]), we propose an alternative for capturing the strongest connection between a given set of attributes. When a user with no knowledge of the database structure poses an abbreviated query, he refers to some view that he assumes to be represented in the database. Accordingly, he would expect his view to behave under update as any basic object in the database, that is, with no side effects. Consequently, we propose the existence of a side effect free view-update mapping to guide the specification of non-basic objects. Thus, the plausibility of such non-basic objects is graded as follows:

(0) least plausible the view has no side effect free update mapping.

(1) plausible the view has a side effect free delete mapping, but no side effect free insert mapping;

(2) most plausible the view has a side effect free update (delete and insert) mapping;

The plausibility criterion above can be incorporated in any of the UR methodologies for query interpretation. The following proposition states that for a non-basic object the losslessness of the associated join is equivalent to the existence of a delete side effect free mapping. Consequently, the third plausibility condition above is stronger than the lossless join condition.

**Proposition 6.1.** Let $(O, K)$ be an UR schema and $O_i$ a non-basic object. $O_i$ has a delete side effect free mapping iff the join of the relations associated with its minimal basic subobjects, is lossless with respect to $K$. 

Sketch of Proof: following proposition 5.3, a delete side effect free mapping for some non-basic object, $O_i$, is ensured if at least one minimal basic subobject of $O_i$, $O_j$, functionally determines $O_i$. The same condition ensures the losslessness of the join of the relations associated with the basic subobjects of $O_i$.

7. Default Query—Subgraphs In Entity—Relationship Diagrams.

Our purpose in this section is to apply the extended UR query interpretation methodology discussed in the previous section, to ER queries. As has been observed by many authors (e.g. [WK]), an ER query can be seen as selecting a subgraph of the ER diagram, and associating predicate conditions with the diagram vertices. Such a subgraph corresponds to interpreting an ER query since it represents all the joins necessary to evaluate the query in the underlying ER-consistent database. We use the term query-subgraph to denote a connected subgraph of the ER diagram that represents an ER query. For users with no knowledge of the ER diagram the provision of default query-subgraphs could allow the expression of abbreviated queries consisting solely of the specification of a set of possibly disconnected entity-sets and relationship-sets. Moreover, default query-subgraphs could also improve the conciseness of ER-oriented query languages. In ER-consistent databases the query-subgraphs correspond to proper, connected, ERD subgraphs (proposition 4.1(ii)). We define the plausibility of query-subgraphs as follows.

Definition 7.1 — Query—Subgraph Plausibility.
Let $(O,K)$ be an UR schema translation of ERD $G_{ER}$, and $G_{ER}'$ a query-subgraph over $G_{ER}$. Then the plausibility of $G_{ER}'$ is defined as the plausibility of $O_i$.

Given a set of entity-sets and/or relationship-sets, $Q$, we propose a procedure to determine the connected query-subgraphs spanning $Q$. Our goal is to obtain subgraphs that are minimal, proper ERD subgraphs, and having the highest possible plausibility. The procedure starts with the initial ERD and reduces it as far as possible, while preserving its connectivity and the condition of being a proper ERD subgraph. The reduction of an ERD over a subset of its vertices, called Red, is presented in figure 9, and exemplified in figure 10.
Red : ER-Diagram Reduction Procedure

Input : $G_{ER} = (V, H)$, an attribute-reduced ERD, and $Q \subseteq V$, a subset of vertices;

Output: $G'_{ER} = (V', H')$, a subgraph of $G_{ER}$ spanning $Q$;

Initially $G'_{ER} \equiv G_{ER}$;

Do While (possible)

Choose from $(V' - Q)$ a reducible vertex $X_i$ such that

(i) $G'_{ER} (V' - X_i)$ is connected; and

(ii) plausibility of $G'_{ER} (V' - X_i) \geq$ plausibility of $G'_{ER}$;

Replace $G'_{ER}$ by $G'_{ER} (V' - X_i)$;

EndDo.

Fig.9 Entity-Relationship Diagram Reduction.

Definition 7.2 − Reducible Vertex.

Let $G_{ER} = (V, H)$ be an ERD. A vertex $X_i \in V$ is said to be reducible iff $\notin X_j \in V$ such that $X_j \rightarrow X_i \in G_{ER}$. For example all r-vertices are reducible.

Lemma 7.1. Let $G_{ER} = (V, H)$ be an attribute-reduced ERD. A vertex $X_i \in V$ is reducible iff the subgraph of $G_{ER}$ induced by $(V - X_i)$ is a proper ERD subgraph of $G_{ER}$.

Sketch of Proof : If $X_i$ is reducible then clearly $G_{ER} (V - X_i)$ is a proper ERD subgraph of $G_{ER}$. Conversely, if $G_{ER} (V - X_i)$ is a proper ERD subgraph of $G_{ER}$, then it is simple to verify that $\notin X_j \in V$ such that $X_j \rightarrow X_i \in G_{ER}$.

Fig.10 Entity-Relationship Diagram of Figure 2 Reduced on DEPARTMENT and SPONSOR.
Proposition 7.2. Let $\overline{G}_{ER}=(V, H)$ be an attribute-reduced ERD, and $Q$ a subset of vertices of $V$, and let $\overline{G}'_{ER}=(V', H')$ be the result of reducing $\overline{G}_{ER}$ on $Q$ by Red. Then:

(i) $Q \subseteq V'$, and $\overline{G}'_{ER}$ is a proper, connected, ERD subgraph of $\overline{G}_{ER}$; and

(ii) $\forall X_j \in (V' - Q)$: either $X_j$ is not reducible, or disconnecting $X_j$ would destroy the connectivity of $G'_{ER}$, or disconnecting $X_j$ would decrease the plausibility of $G'_{ER}$.

Sketch of Proof: (i) follows from lemma 7.1, and (ii) follows from the procedure steps.

Let ERD $G_{ER}$ be associated with UR schema $(O, K)$. Condition (i) of proposition 7.2 guarantees that the reduction of $\overline{G}_{ER}$ on any non-empty subset of vertices, $Q$, results in an ERD subgraph that corresponds to a well-defined either basic or non-basic object, depending on $Q$: if the resulting subgraph has a root, then it corresponds to the basic object associated with the root; otherwise it corresponds to a non-basic object. Note that in any case some ERD subgraph is found, that is, any abbreviated query has at least associated some query-subgraph with certain plausibility. Another property of the reduction proposed by us is that it guarantees *faithfulness* [MRW]. Given some set of attributes, $X$, that equals a basic object, $O_i$, a faithful window-function on $X$ will return the relation that is associated with $O_i$. In the case of Red, for any set of attributes that equals the attribute-set of some basic object, Red returns a single subgraph rooted in the vertex corresponding to that object.

Several examples are given below:

(i) Consider a query over the ER diagram of figure 2 that requests the SPONSORS of the CS DEPARTMENT. The query-subgraph with the highest plausibility (2) is that of figure 10.

(ii) Consider a query over the ER diagram of figure 11(i) that requests the FACULTY members of all DEPARTMENT. The corresponding query-subgraph with the highest plausibility (2)

---

Fig.11 Entity-Relationship Diagram Examples.
is \{FACULTY, TEACH, COURSE, OFFER, DEPARTMENT\}. Note that for this query, the interpretation proposed by [MU] or [ZM] would result in the questionable union-combination of two query-subgraphs, the one above and a second join-graph, with plausibility 1, namely \{FACULTY, ADVISOR, STUDENT, ENROL, DEPARTMENT\}.

(iii) Consider a query over the ER diagram of figure 11(ii) that requests the list of all STUDENTS and their FACULTY. The query-subgraph with the highest plausibility (1) is the whole ERD. Note that according to the procedure proposed in [ZM] there is no interpretation for this query, although the join of all the relations associated with the objects corresponding to the ERD vertices is lossless.

8. Conclusion.

We have proposed to extend the UR methodology for query interpretation, and to apply it to the ER model. We have introduced a new criterion for interpreting abbreviated queries, namely the existence of a side effect free update mapping for the view implied by the query. We have proposed a methodology for interpreting abbreviated ER queries, in which an underlying UR formalism, including the above criterion, supports an ER-oriented graph procedure for determining default query-subgraphs over ER diagrams.

Although having a natural intuition, our methodology, like other UR methodologies, does not guarantee the selection of a single query-subgraph for a given abbreviated query. Furthermore, even when a single query-subgraph is found, we cannot guarantee that it agrees with the user's intended meaning. Therefore some strategy is needed for verifying whether the user's intentions have been satisfied, and for selecting from several query-subgraphs. Most UR methodologies simply union all the derived relations corresponding to the various query-subgraphs [Ull83]. The problem with such a strategy is that while a union is sometimes desirable, it is by no means the only reasonable choice. Indeed, selecting a single query-subgraph is more likely to be desired in most cases. The union employed by the UR methodologies requires the satisfaction of the One Flavor Assumption [MRW], meaning that all the relations involved in the union would represent the same semantic relationship. However, the One Flavor Assumption satisfaction or violation is impossible to prove because it depends on the designer's perception of the real-world.

Another obvious strategy is to consult the user. In the extreme, the user can be shown all the alternative query-graphs, and chose the desirable one. If such a strategy is employed, then
our methodology can order the choices according to plausibility, from most plausible to least plausible, with respect to the existence of side effect free update mappings. Yet another strategy could be to choose subgraphs that contain short paths. The rational is that short paths are less confusing for a user to imagine. Our methodology can be also extended by adding new plausibility levels. For instance, the database designer could label some subgraphs as more plausible, and use this as a guidance for selecting the query-subgraph most likely to agree with the user's intended meaning.

Acknowledgement: This work was supported by the Applied Mathematical Sciences Research Program of the Office of Energy Research, U.S. Department of Energy, under contract number DE-AC03-76SF00098.

References


