Computer Science Department Technical Report
University of California
Los Angeles, CA 90024-1596

IMPLICIT REPRESENTATION FOR EXTENSIONAL ANSWERS

Chung-Dak Shum
Richard Muntz

August 1988
CSD-880067

Implicit Representation for Extensional Answers

Chung-Dak Shum
Richard Muntz

Computer Science Department
University of California, Los Angeles

Abstract

An exhaustive list of atomic objects is not always the best means of information exchange. This paper concerns the implicit representation of extensional answers. Expressions for answers are given in terms of concepts and individuals. Exceptions within individual concepts are allowed. Two criteria are defined as measures of the *goodness* of such expressions: (i) minimizing the number of terms; (ii) positive terms preferred over negative terms. Expressions satisfying these two criteria are called optimal expressions. It is shown that under a strict taxonomy of concepts, any two optimal expressions for an extensional answer share the same set of terms. The inductive proof elicits an algorithm for obtaining such expressions. Generalizing the strict taxonomy of concepts to a join-semilattice of concepts eliminates the term uniqueness property and also makes the problem of finding an optimal expression intractable. The problem under multiple taxonomies, although it involves a restricted type of join-semilattice, remains intractable.

1. Introduction

Conventional responses in database systems, usually given as lists of atomic objects, although sufficient to serve the purpose of conveying information, do not necessarily provide efficent and effective communications between a user and the system. Recently, new notions of answers to queries have been receiving more research interest. For example, in [1], an answer to a query is expressed in terms of both atomic facts and general rules; in [2,3], intensional descriptions or concepts are being used as part of an answer. This latter notion of answers is particularly helpful when the number of entities or objects which satisfy the query is very large. Consider the personnel database of a large corporation and the query

$$\langle x/employee \mid (\exists y/annual\_pay)\ salary(x,y) \wedge (y > 30{,}000) \rangle.$$

If there is a large number of employees whose salaries are more than 30,000, and if it turns out that all engineers, all managers and a few secretaries are, then it seems reasonable for the system to let the user know of the situation. In this paper, we are interested in answers of this type and will refer to them as *abstract responses*.

Our work is closely related to that by Corella [3]. In his paper, the answer to a query is expressed not as a set of individuals, but as a set of concepts or predicates, whose extensions may not be explicitly represented. Now suppose that we have retrieved a set of individuals as the answer to a conventional database query, and we want to re-express the answer in terms of a set of concepts. Those concepts must, of course, be pre-defined; otherwise, the user may not be familiar with them and the answer in terms of those concepts will thus make not too much sense. One of the immediate drawbacks to such an approach of expressing answers is that the extensions of the pre-defined

concepts often do not satisfy the query conditions as a whole. As a result, we cannot express answers the way we want except in very rare cases. An obvious solution to this problem is to allow *exceptions* within individual concepts [4] to be used as part of the answer. For example, if all engineers except John Smith satisfy the above query, we should be able to say just exactly that in the answer.

Here we will study the different ways of expressing abstract responses. In Section 2, we introduce the notion of an expression which will be used as answers to queries throughout the paper. Criteria for comparing such expressions will be defined in Section 3, and algorithms for generating the so-called *good* expression under a strict taxonomy of concepts will be given. The strict taxonomy of concepts is generalized to a join-semilattice of concepts in Section 4. Section 5 is devoted to the study of obtaining expressions over multiple taxonomies. In the last Section, we conclude the paper with suggestions of possible future work in this area.

2. Definitions and Notations

We consider a finite domain **D** of individuals, and *concepts* relative to **D**. A concept is a unary predicate $C(\cdot)$ defined over **D**, where C, with possible subscripts, is the label of the concept. For convenience, we will also denote the extension of the predicate $\{x \mid C(x)\}$ by C. The context should suffice to disambiguate. A concept C_1 is said to be subsumed by another concept C_2 if and only if $C_1 \subseteq C_2$. We shall use both set terminology (union, intersection, complementation, set inclusion, difference) and logic terminology (disjunction, conjunction, negation, subsumption) when referring to concepts. The *extensional* answer **A** to a query is simply a subset of **D** whose elements satisfy the query conditions. The problem is to re-express the answer in terms of some pre-defined concepts and individuals.

We are not dealing with an arbitrary collection of concepts; instead, we are interested in a *taxonomy* of concepts.

Definition 2.1 A *taxonomy* is a finite tree whose nodes are labeled by concepts. Any node other than the leaf node has two or more successors. The successor concept of each node is subsumed by its parent concept. The union of all successor concepts of any non-leaf node is equal to the parent concept. A taxonomy is called *strict* if all sibling concepts are mutually exclusive.

Since we will be working mostly with strict taxonomies, the word taxonomy will simply be used to refer to a strict taxonomy unless otherwise stated. An extensional answer **A** to a query is related to a taxonomy by the following definition.

Definition 2.2 A set of individuals **A** is *classifiable* by a taxonomy **T** iff the root concept of **T** contains **A**.

Next we look at how to express an extensional answer **A** in terms of concepts from a taxonomy **T** and individuals from the root concept of **T** given that **A** is classifiable by **T**. To this end, we need the notion of an expression.

Definition 2.3 The alphabet of an expression defined over a taxonomy **T** is composed of the following:

1. *Concepts*: $C_1, C_2, \cdots$,
   Each concept is a label of a node in **T**.
2. *Individuals*: $d_1, d_2, \cdots$,
   Each individual is an element of the root concept of **T**.
3. *Empty*: ε,

This denotes the empty expression.

4. *Signs*: +,−.

Next we introduce the notion of a term, followed by the syntax of an expression.

Definition 2.4 There are two types of *terms*:

1. If x is an individual or a concept, then $+x$ is a *positive* term.
2. If x is an individual or a concept, then $-x$ is a *negative* term.

Definition 2.5 An *expression* over the taxonomy **T** is defined inductively as follows:

1. A term (positive or negative) is an expression.
2. ε is an expression.
3. If e_1 and e_2 are expressions, so is $e_1 e_2$.

Expressions are introduced so that extensional answers can be re-expressed in terms of high level concepts and, consequently, abstract responses can be offered to the user which enhance his understanding with regard to the state of the world. Thus it seems natural to associate the meaning of an expression with a set of individuals.

Definition 2.6 The meaning of an expression over a taxonomy **T** is given by a mapping ξ: $Exp \rightarrow 2^R$, where Exp is the set of expressions over **T** and R is the root concept of **T**.

1. $\xi(+d) = \{d\}$ if d is an individual;
   $\xi(+C) = C$ if C is a concept;
   $\xi(-x) = \varnothing$ if x is an individual or a concept.
   $\xi(\varepsilon) = \varnothing$
2. $\xi(e+d) = \xi(e) \cup \{d\}$ if e is an expression and d is an individual;
   $\xi(e+C) = \xi(e) \cup C$ if e is an expression and C is a concept;
   $\xi(e-d) = \xi(e) - \{d\}$ if e is an expression and d is an individual;
   $\xi(e-C) = \xi(e) - C$ if e is an expression and C is a concept;
   $\xi(e\ \varepsilon) = \xi(e)$ if e is an expression.

We assume the $\cup$ and $-$ operators within a set expression have the same precedence and the left associative rule is used in the evaluation of such an expression. With the meaning of an expression defined, we can compare two expressions.

Definition 2.7 Two expressions e_1, e_2 over a taxonomy **T** are *equivalent* iff $\xi(e_1) = \xi(e_2)$.

Notice that by way of definition, expressions with same terms do not necessarily carry the same meaning. Obviously, if a concept $C = \{d_1, d_2\}$ and two expressions $e_1 = +C-d_1$ and $e_2 = -d_1+C$, e_1 and e_2 contain the same terms, but they are not equivalent since $\xi(e_1) = \{d_2\}$ whereas $\xi(e_2) = C$.

To fix ideas, let us illustrate how expressions can be used as responses to queries via a more realistic example. Suppose a large organization, say a government agency, has to keep record of a large number of vehicles. These vehicles include many diverse types such as cars, trucks, and helicopters. Each type is a set of individual vehicles and may be thought of as defining a concept. Figure 1 shows a *taxonomy* of the vehicle types. The successor concept of each node in the taxonomy is subsumed by its parent concept; for example, all *cars* are *road vehicles*. Now assume that we are interested in the speed of the vehicles and the following question is asked

"Which vehicle can go at a speed of 60 mph or more?"

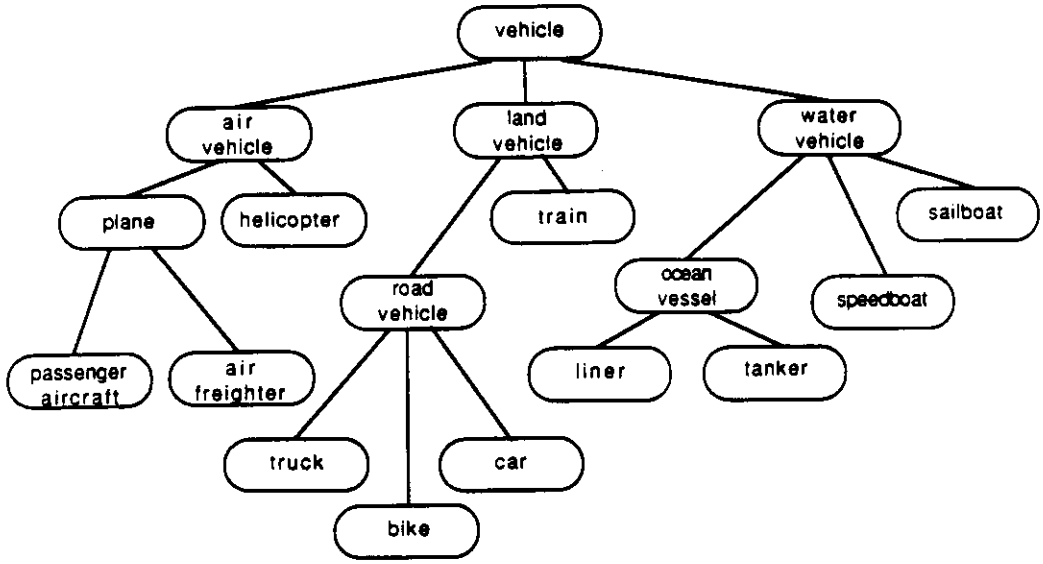

Figure 1 A Taxonomy of Vehicle Types

If each individual vehicle has an identification number and its associated maximum speed recorded, the response could, then, be an enumeration of vehicle identification number. But it could also be simpler and the following are example forms of answers in terms of concepts. It is not difficult to see that all three expressions are equivalent.

"*passenger aircraft* + *air freighter* + *truck* + *car* + *train* + *speed boat*";

"*air vehicle* + *road vehicle* − *bike* + *train* + *speed boat*"; or

"*vehicle* − *bike* − *water vehicle* + *speed boat*".

Notice that in the third expression, *speed boat* is subsumed by *water vehicle*. Thus, the last two terms in that expression basically say that all *water vehicles* are excluded besides *speed boats*. The question of which expression is preferable will be discussed next.

3. Expression Complexity

We have defined alternative ways for presenting an answer to a user, not as an exhaustive list of individuals, but rather as an expression of concepts and individuals. In fact, our definition of expression actually covers the case in which just individuals are listed. Such is the case when the expression is made up solely of individuals and all the terms within the expression are positive. It is not difficult to see that given an extensional answer **A** to a query and a taxonomy **T** by which **A** is classifiable, there may exist many expressions e_i over **T** such that $\xi(e_i) = \mathbf{A}$. Thus presenting an answer in the form of an expression over **T** is not necessarily unique. In this Section we will present our view of what constitutes a good expression, illustrate certain properties associated with such expressions, and give algorithms for obtaining them.

An extensional answer may not always be the best way of presenting a response, but sometimes it is. This is true when there are only a few individuals that satisfy the conditions of the query. Hence, the number of *items* appearing in a answer is clearly one of the criteria against which to measure how good an answer is, in the sense of how easy it is for a user to comprehend. In our case, the number of terms involved in an expression is the important measure. Since a term in our expression may involve an individual or a concept, weighting each term equally, we are assuming that the user has similar understandings with respect to concepts as well as to individuals.

Definition 3.1 Let **A** be an extensional answer to a query classifiable by a taxonomy **T**. An *answer set* **E** for **A** is given by:

$$\{e \mid \xi(e) = \mathbf{A}\}$$

A subset **M** of an answer set **E** is called a *minimal answer set* for **A** if for each $e \in \mathbf{M}$, $|e| \leq |e'|$ for all $e' \in \mathbf{E}$, where $|x|$ is the number of terms in expression x. We call each expression in **M** a *minimal expression*.

The following proposition for the minimal answer set should be obvious from the definitions.

Proposition 3.1 All expressions in a minimal answer set are equivalent and they all have the same (minimum) number of terms.

Here we will look at some interesting properties of minimal expressions. In particular, we define what we mean by a subexpression and prove two lemmas that will be useful in establishing a later theorem. For convenience, let us attach individual elements of leaf concepts to their corresponding terminal nodes in the taxonomy and refer to $\mathbf{T}_+$ hereafter, as the *modified tree taxonomy*. This way we have a uniform treatment for individuals and concepts. Each individual, informally, is just a concept with a single element.

Definition 3.2 A *path* **P** in a taxonomy **T** is a sequence of nodes $<C_{P_1}, C_{P_2}, \cdots, C_{P_n}>$ in which successive nodes are connected by edges in **T**. A *rooted path* is any path whose first node C_{P_1} is the root node. A *terminal path* is any path ending at a leaf node. A *full path* is a rooted terminal path.

Definition 3.3 Let e be an expression for an answer set **A** over a taxonomy **T** and **P** be a full path in **T**. A *subexpression* of e over **P** is an expression formed by removing from e all those terms whose associated concepts are not in **P**.

Lemma 3.1 Let e_M be a minimal expression and $e_P = \pm C_{P_1} \pm C_{P_2} \cdots \pm C_{P_k}$ be a subexpression of e_M over some full path **P**. Then for all $i > j$, $C_{P_i} \subset C_{P_j}$.

Proof: Assume the contrary, then there exists some i, such that $C_{P_i} \subset C_{P_{i+1}}$.

Let $e_M = e_H \pm C_{P_i}\ e_{notP} \pm C_{P_{i+1}}\ e_T$ where e_H and e_T represent respectively the head and tail portion of e_M and e_{notP} represents the portion between $\pm C_{P_i}$ and $\pm C_{P_{i+1}}$.

Notice that by the definition of subexpression of e_M over **P** all concepts associated with e_{notP} are not in **P**.

Consider the expression $e_M' = e_H\ e_{notP} \pm C_{P_{i+1}}\ e_T$.

Since $\pm C_{P_{i+1}}$ will override any effect e_{notP} has over $\pm C_{P_i}$, it is not difficult to see that $\xi(e_M)$ is equivalent to $\xi(e_M')$. But e_M' has one term less than that of e_M.

Therefore, e_M is not a minimal expression. **Contradiction.**

Lemma 3.2 Let e_M be a minimal expression. Then any subexpression $e_P = \pm C_{P_1} \pm C_{P_2} \cdots \pm C_{P_k}$ of e_M over some full path **P** must have alternate positive and negative terms.

Proof: Assume the contrary. There are two cases:

(i) $e_M = e_H + C_{P_i}\, e_{notP} + C_{P_{i+1}}\, e_T$
where e_H and e_T represent respectively the head and tail portion of e_M and e_{notP} represents the portion between $+C_{P_i}$ and $+C_{P_{i+1}}$.

Notice that by the definition of subexpression of e_M over **P** all concepts associated with e_{notP} are not in **P**, and by Lemma 3.1, $C_{P_{i+1}} \subset C_{P_i}$. Thus, individuals in $C_{P_{i+1}}$, being introduced by the term $+C_{P_i}$, are not affected by e_{notP}. So the term $+C_{P_{i+1}}$ is redundant.

Therefore, e_M is not a minimal expression. **Contradiction.**

(ii) $e_M = e_H - C_{P_i}\, e_{notP} - C_{P_{i+1}}\, e_T$
The proof of this case is similar to (i).

Since the answer set with the minimum number of terms condition still leaves us with potentially many minimal expressions, we are led to consider another measure of merit in order to further distinguish the so-called good expressions. Suppose an extensional answer **A** has two individuals d_1 and d_2, and a concept $C = \{d_1, d_2, d_3\}$. Now two minimal expressions exist: $+d_1 + d_2$ and $+C - d_3$. Most people would prefer the former response because it answers the question directly. Thus the next criterion for a good expression is *positive terms are preferred over negative terms*.

Definition 3.4 An *optimal answer set* Ω is a subset of a minimal answer set **M** such that for all $e \in \Omega$, e has the maximal number of positive terms. We called each expression in Ω an *optimal expression*.

Our two simple criteria for good expressions, namely, the minimum number of terms condition and positive terms over negative terms, give rise to an optimal answer set with an interesting property which is stated in the following theorem.

Theorem 3.1 Any two optimal expressions in a given optimal answer set Ω have exactly the same set of terms. (*term uniqueness*)

Proof: By induction on the height of the tree taxonomy. The inductive proof also elicits an algorithm for obtaining an optimal expression. Here we highlight the inductive steps and leave some details to an Appendix.

height = *1* (a root concept R with individuals d_i attached)

Consider two ways of expressing the extensional answer **A** classifiable by taxonomy **T**:

(i) without root concept R: $e_1^{-R} = \begin{cases} +d_{i_1} \cdots +d_{i_k} & \mathbf{A} \neq \varnothing \\ \varepsilon & \mathbf{A} = \varnothing \end{cases}$

(ii) with root concept R: $e_1^{+R} = +R\ e_1^{-}$ where $e_1^{-} = \begin{cases} \varepsilon & \mathbf{A} = R \\ -R & \mathbf{A} = \varnothing \\ -d_{j_1} \cdots -d_{j_m} & m < \dfrac{|R|+1}{2} \\ -R+d_{i_1} \cdots +d_{i_k} & otherwise \end{cases}$

We claim that e_1^{-R} is an "optimal expression" conditioned on R being excluded and e_1^{+R} is "optimal" conditioned on R being included. The simple proofs are omitted.

Next, we observe that except for $\mathbf{A} = R$, e_1^{+R} has more negative terms than e_1^{-R}.

Thus only one of e_1^{+R} and e_1^{-R} must be optimal (e_1^{opt}). Notice that both e_1^{+R} and e_1^{-R} may not be unique since a reordering of terms may give an equivalent "optimal" expression, but the set of terms forming each is unique.

height = k

Assume that (1) e_k^{-R} is an "optimal expression" conditioned on the root concept being excluded and $e_k^{+R} = +R\ e_k^{-}$ is an "optimal expression" conditioned on the root concept being included and the set of terms forming each is unique; (2) except for $\mathbf{A} = R$, e_k^{+R} has more negative terms than e_k^{-R}; (3) one of e_k^{-R} and e_k^{+R} is an optimal expression e_k^{opt}.

height = $k+1$

Assume that the root R has n successor $(S_1, \cdots, S_n)$ and without loss of generality, each successor is the root of a tree with height k. For each subtree $\mathbf{T}_i$ rooted at successor S_i $(1 \le i \le n)$, $e_{k_i}^{-R}$ and $e_{k_i}^{+R} = S_i\ e_{k_i}^{-}$ exist and have the properties as described above (*height* = k).

Again consider two ways of expressing the extensional answer **A**:

(i) without root concept R: $e_{k+1}^{-R} = e_{k_1}^{opt} \cdots e_{k_n}^{opt}$

We claim that e_{k+1}^{-R} is an "optimal expression" conditioned on the root concept being excluded. Without the root the subtrees $\mathbf{T}_i$ are mutually independent. Thus the only "optimal" way to express **A** is to concatenate optimal expressions from each subtree.

(ii) with root concept R: $e_{k+1}^{+R} = +R\ e_{k+1}^{-}$

where $e_{k+1}^{-} = \begin{cases} \varepsilon & \mathbf{A} = R \\ -R & \mathbf{A} = \varnothing \\ OPT(e_{k_1}^{-} \cdots e_{k_n}^{-}, -R\ e_{k_1}^{opt} \cdots e_{k_n}^{opt}) & otherwise \end{cases}$

The *OPT* function simply returns the argument expression which is optimal of the two. We note that expressions (α) $e^-_{k_1} \cdots e^-_{k_n}$ and (β) $-R\ e^{opt}_{k_1} \cdots e^{opt}_{k_n}$ cannot have the same number of positive and negative terms. The details are shown in Appendix A.

Next we claim that e^{+R}_{k+1} is an "optimal expression" conditioned on the root concept being included. The cases where $\mathbf{A} = R$ and $\mathbf{A} = \emptyset$ are trivial. We only show the last case. Suppose $e^{+R}_{k+1} = +R\ e^-_{k_1} \cdots e^-_{k_n}$.

Assume that there exist $e^{+R}_{k+1}{}' = +R\ e^-_{k_1} \cdots e^-_{k_i}{}' \cdots e^-_{k_n}$, which is "optimal" conditioned on R being included. By Lemma 3.2, for any full path, any concept in $e^-_{k_i}$ or $e^-_{k_i}{}'$ which is closest to R must be negative.

Thus for subtree $\mathbf{T}_i$ we can get $+S_i\ e^-_{k_i}{}'$ which is "optimal" conditioned on S_i being included instead of $e^{+R}_{k_i}$. **Contradiction.**

Similarly, we can show the same for $e^{+R}_{k+1} = +R\ -R\ e^{opt}_{k_1} \cdots e^{opt}_{k_n}$.

Also e^{+R}_{k+1} has more negative terms than e^{-R}_{k+1}. This is true since each $e^{+R}_{k_i}$ has more negative terms than $e^{-R}_{k_i}$.

Hence one of e^{-R}_{k+1} and e^{+R}_{k+1} must be an optimal expression and other equivalent optimal expressions must share the same terms.

As mentioned, the proof of Theorem 3.1 is constructive in nature, an algorithm for obtaining an optimal expression for an extensional answer **A** over a taxonomy **T** falls naturally out of its induction step. Basically, we do a *postorder traversal* of **T**, that is, before a node is visited, all its descendants must have been traversed. Expressions $e^{-R}_{k_i}$ and $e^{+R}_{k_i}$ are constructed as each node (indexed by k_i) is being encountered and the optimal expression is readily obtainable once the root has been visited.

By Theorem 3.1, the set of terms occurring in different optimal expressions must be identical. Thus, the ordering among terms is the only way by which two optimal expressions can differ. However, it is not true that any ordering will do; instead, Lemma 3.1 tells us that the ordering of terms is related to a partial ordering of their associated concepts.

Theorem 3.2 Let $\{T_1, \cdots, T_k\}$ be the set of terms associated with an optimal answer set Ω and corresponding to each term T_i is a concept C_i. Then $T_{\pi(1)} \cdots T_{\pi(k)}$ is an optimal expression if and only if for any $C_{\pi(i)} \subset C_{\pi(j)}$, $i > j$, where π is a permutation of these k elements.

Proof: *only-if part*

Assume the contrary, that is, there exists some i and j such that $C_{\pi(i)} \subset C_{\pi(j)}$ and $i < j$.

Since the taxonomy is a tree, $C_{\pi(i)} \subset C_{\pi(j)}$ implies that $C_{\pi(i)}$ and $C_{\pi(j)}$ must lie in some full path P. It follows from Lemma 3.1 that $i > j$. **Contradiction.**

if part

An important observation is that for an optimal expression, if the concepts associated with two adjacent terms are disjoint, switching them makes another optimal expression. This should be obvious from the meaning of expressions.

For any arbitrary expression which satisfies the "*if condition*", we can transform an optimal expression to that form through a series of switching operations as described above. The proof of this is omitted for simplicity.

Since each switching results in another optimal expression, any arbitrary expression which satisfies the "*if condition*" is also an optimal expression. Thus the Theorem is established.

To summarize thus far, using our notion of an expression, we studied formally how to re-express an extensional answer into a more abstrast form. Two simple criteria: (i) the minimum number of terms condition, and (ii) positive terms over negative terms, were introduced as a measure of how *good* such an abstract response is. Under a *strict* taxonomy, we were able to establish an optimal answer set whose members satisfy the two criteria and have the same set of terms. Moreover, any permutation of those terms which satisfies a partial ordering induced by the taxonomy belongs to the optimal answer set.

As far as the ordering of terms within an optimal expressions is concerned, there is still another criterion that could be imposed in order to get an even smaller set of *good* expressions. Consider two abstract responses

"*all engineers and all managers except John Smith*" and
"*all engineers except John Smith, and all managers*".

Assume that *engineers* and *managers* are disjoint sets and that *John Smith* belongs to *engineers*. Notice that these two responses could be present in the same optimal answer set and they do carry the same meaning if the user is aware of the assumptions. Most people would, however, prefer the second one as a response since it groups together similar concepts and/or individuals.

Definition 3.5 A *normalized answer set* **N** is a subset of an optimal answer set Ω such that for all $e \in \mathbf{N}$, the ordering of terms in e satisfies a *preordering* of the taxonomy **T**.

Notice that a *preordering traversal* of a tree is one in which a node is traversed first before its subtrees. We still have one dimension of freedom left. The different orders by which the subtrees are traversed constitutes different normalized answers. However, we choose not to distinguish between those, since answers like

"*all managers and all engineers*" or
"*all engineers and all managers*"

apparently makes no difference to a user. The ordering of subtrees has basically this kind of flavor. Thus, we assume that any expression in the normalized answer set is equally *good*.

4. Extensions

A collection of concepts relative to a domain D does not necessarily fall into a strict (tree) taxonomy. In other words, for two concepts, it may neither be the case that their intersection is empty nor one is the subset of the other. For example, consider two job categories of the employees of a company: *computer scientist* and *electrical engineer*. It is not surprising that some employees belong to both categories, whereas others belong exclusively to one category. In fact, if enough members belong to the intersection, it is quite natural to form another concept, say, *computer engineer*, so that we can denotate them collectively as an aggregate. A simple extension beyond the strict (tree) taxonomy leads us to consider a *join-semilattice* of concepts [5].

Definition 4.1 A *partially ordered set* (poset) $P, \leq$ is a *join-semilattice* if and only if for any pair elements $x,y \in P$, there is a *unique* element $z \in P$ such that $x \leq z$ and $y \leq z$, and for all $u \in P$, if $x \leq u$ and $y \leq u$, then $z \leq u$.

In our case, each element in the poset P is a concept and the partially ordered relation $\leq$ is taken as the *subset* $\subseteq$ relation. That is, for $C_1, C_2 \in P$, $C_1 \leq C_2$ if and only if $C_1 \subseteq C_2$.

If we substitute the strict (tree) taxonomy with the more general join-semilattice of concepts, and still consider the same two criteria for optimal expressions, it is easy to show that term uniqueness no longer holds. Again we treat individuals as concepts with single elements for uniformity and simplicity. Consider the join-semilattice of concepts as shown in Figure 2. The filled nodes represent the extensional answer.

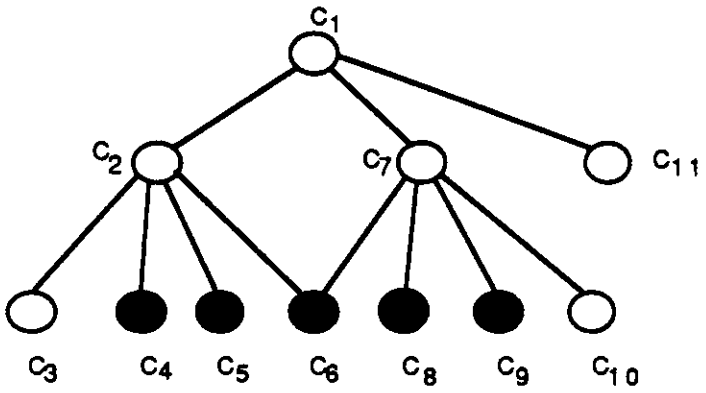

Figure 2 A Join-Semilattice of Concepts

It is not difficult to see that $+C_2-C_3+C_8+C_9$ is an optimal expression as well as $+C_7-C_{10}+C_4+C_5$. In this particular example, in fact, none of the terms in these two optimal expressions is the same. Thus, Theorem 3.1 is not always true under a join-semilattice of concepts.

Without worrying about the term uniqueness in optimal expressions, we would like to know whether any optimal expression can be obtained through an efficient algorithm. By efficient, we mean an algorithm whose time complexity function is $O(p(n))$ for some polynomial function p, where n is used to denote the input length. Unfortunately, we are able to show that the problem at hand belongs to the class of *NP-complete* problems [6]. No efficient algorithm has been found for any problem in this class.

Let us restate our problem in terms of a decision problem. The extensional answer **A** to a query is simply a subset of singleton concepts. Analogous to Definition 2.2, we define *classifiability* as follows.

Definition 4.2 An extensional answer **A** is *classifiable* by a join-semilattice **L** iff the *greatest upper bound* of **L** contains $\cup$**A**.

Here is the decision version of our problem.

MINIMAL EXPRESSION

INSTANCE: An extensional answer **A** to a query classifiable by a join-semilattice **L** and a bound $B \in Z^+$ (where Z^+ denotes the positive integers).

QUESTION: Is there an expression e such that $\xi(e) = \cup$**A** and the number of terms in e is no more than **B**?

Clearly, if we could find a minimal expression for **A** classifiable by **L** in polynomial time, we could also solve the associated decision problem described above in polynomial time. All we need to do is find the minimal expression, determine the number of terms involved, and compare it with the given bound **B**. Thus, the decision problem can be no harder than the corresponding optimization problem. If we could demonstrate the MINIMAL EXPRESSION problem is NP-complete, we would know that its optimization problem is at least as hard.

Theorem 4.1 MINIMAL EXPRESSION is NP-complete.

A join-semilattice is still a somewhat general structure. In the next Section, we are able to show that the same problem under a *restricted* join-semilattice still remains NP-complete, and thus, Theorem 4.1 follows.

5. Multiple Taxonomies

Consider again the query

$$<x/employee \mid (\exists y/annual\_pay)\ salary(x,y) \wedge y > 30{,}000>.$$

Suppose all managers and nearly half of all engineers satisfy the query, and the number of engineers is large. Then our expression would consists of a single concept *manager* and a long list of individual engineers. Now assume further that an employee's salary is related to his education level and it so happens that all and only the engineers that have master degrees earn more than 30,000. Apparently, the best answer, then, to the above query is

"*all managers and all engineers with master degrees*".

Notice that there are three concepts in the expression. Of the three concepts involved, the first two, *manager* and *engineer*, may belong to, say, a *job category* taxonomy, whereas the third one, *master*, may belong to an *education level* taxonomy. In this Section, we are concerned with finding optimal expressions whose concepts are not restricted to a single taxonomy, but can, possibly, come from different taxonomies.

To express the set of "*all engineers with master degrees*", we introduce the notion of a *composite* concept.

Definition 5.1 Let $\mathbf{T}_i$ $(1 \le i \le k)$ be a set of taxonomies and C_i be a concept of $\mathbf{T}_i$. Then the tuple $<C_1, \cdots, C_k>$ is a *composite* concept over k taxonomies. We also referred to each C_i as a simple concept.

If C_1 of $\mathbf{T}_1$ denotes the set of *engineers* and C_2 of $\mathbf{T}_2$ denotes the set of *masters*, we use the composite concept $<C_1, C_2>$ to denote the set of *all engineers with master degrees*. As is true for all concepts, a composite concept

represents a set of individuals; but, instead of being defined inherently as in simple concepts, it is given in terms of other simple concepts.

Definition 5.2 Let $<C_1, \cdots, C_k>$ be a composite concept over k taxonomies. The extension of $<C_1, \cdots, C_k>$ is defined as $\bigcap_{1 \le i \le k} C_i$.

The collection of composite concepts that we are dealing with is not arbitrary. Their relationships with one another can be derived from the following definition.

Definition 5.3 The *direct product* $P_1 \cdots P_k$ of k posets P_i, $1 \le i \le k$, is the set of all tuples $<x_1, \cdots, x_k>$ with $x_i \in P_i$, partially ordered by the rule that $<x_1, \cdots, x_k> \le <y_1, \cdots, y_k>$ if and only if $x_i \le_{P_i} y_i$ for all i $(1 \le i \le k)$.

If the partial order relations are taken as the subsumption relations, it is not difficult to see that our definition of composite concepts satisfies the ordering introduced by Definition 5.3 and our collection of composite concepts over k taxonomies corresponds to the direct product of the k taxonomies.

With the meaning of composite concepts defined, we can form expressions in terms of these composite concepts, and consider the optimality of such expressions. Notice that expressions are still defined the same way as in Definition 2.3 with the simple concepts being replaced by composite concepts. To distinguish such expressions, we sometimes referred to them as expressions over k taxonomies. Again, we would like to know whether any optimal expression can be obtained through an efficient algorithm. To this end, we first look at a decision version of a simpler problem.

MULTIPLE-TAXONOMY EXPRESSION

INSTANCE: An arbitrary number of taxonomies, $\mathbf{T}_i$ $(1 \le i \le n)$, an extensional answer **A** classifiable by each taxonomy $\mathbf{T}_i$, and a positive integer **K**.

QUESTION: Is there an expression e over n taxonomies, such that $\xi(e) = \mathbf{A}$ and the number of terms in e is no more than **K**?

If we could demonstrate the MULTIPLE-TAXONOMY EXPRESSION problem is NP-complete, we would know that its corresponding optimization problem, namely, finding a minimal expression under multiple taxonomies, is at least as hard.

Theorem 5.2 MULTIPLE-TAXONOMY EXPRESSION is NP-complete.

Proof: See Appendix B.

Next we study the relationship between the MINIMAL EXPRESSION problem and the MULTIPLE-TAXONOMY EXPRESSION problem. Taxonomies are not arbitrary posets, they are posets with *tree* structures. But the direct product of two non-trivial taxonomies, that is, those with more than one node, is no longer a taxonomy. A simple example illustrates this in Figure 3. Nevertheless, the direct product of any number of taxonomies is still a join-semilattice.

Theorem 5.3 The direct product LM of any two join-semilattices is a join-semilattice.

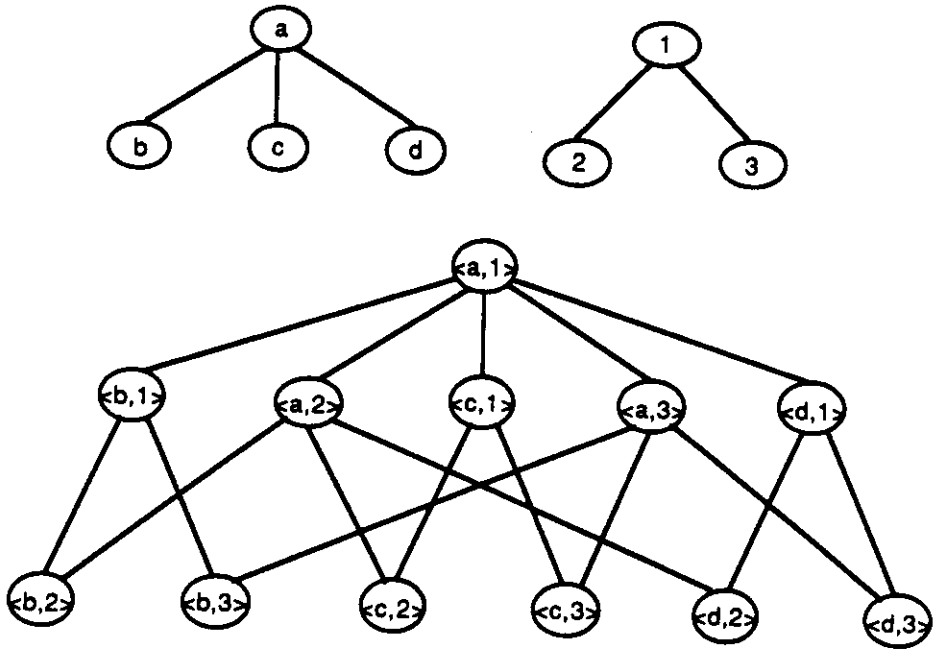

Figure 3 A Direct Product of two Taxonomies

Proof: For any two elements $<x_1,y_1>$, $<x_2,y_2>$ in LM, there exists $<x,y> \in$ LM such that x is the least upper bound of x_1 and x_2, and y is the least upper bound of y_1 and y_2. Thus, by definition of direct product, $<x_1,y_1> \leq <x,y>$ and $<x_2,y_2> \leq <x,y>$.

Moreover, every other upper bound (u,v) of the two $<x_1,y_1>$ and $<x_2,y_2>$ satisfies $x_1 \leq_L u$, $x_2 \leq_L u$ and hence, by definition of least upper bound, $x \leq_L u$; likewise, $y \leq_M v$, and so $<x,y> \leq <u,v>$. This completes the proof that any two elements in LM have a *unique* least upper bound and therefore, LM is join-semilattice.

Corollary 5.1 The direct product of any number taxonomies is a join-semilattice.

Proof: Since a taxonomy is a join-semilattice, this follows trivially from Theorem 5.3.

Thus any arbitrary instance of the MULTIPLE-TAXONOMY EXPRESSION problem can be restated in form of some instance of the MINIMAL EXPRESSION problem. Since the MULTIPLE-TAXONOMY EXPRESSION problem is NP-complete, so is the MINIMAL EXPRESSION problem and Theorem 4.1 follows.

In this Section, we show that the general problem of obtaining optimal expressions over k taxonomies is intractable. This does not, however, mean that responses like

"all managers and all engineers with master degrees"

are not obtainable. Recall that our main reason for introducing multiple taxonomies is that concepts in a single taxonomy do not necessarily characterize the set of individuals in the extensional answer well enough; that is, long list of individuals may still remain in the optimal expression. Such situation occurs when, say, nearly half of the individuals in a certain concept satisfies the query and the total number of individuals in the concept is large. A plausible solution is to further classify these individuals by another taxonomy, obtain an optimal expression for them over that taxonomy and integrate it with the original expression to form a combined expression. Or we may also look at it from the point of view of a grafted taxonomy.

Definition 5.1.1 Let T_1 be a taxonomy and L be a leaf concept of T_1 such that L is classifiable by another taxonomy T_2. Then **T** formed by grafting T_2 at L is called a *grafted* taxonomy. In **T**, the labels of all the concepts which originally belonged to T_1 remain unchanged, and all those which originally belonged to T_2 except the root have the form $<L,C>$, where C is a concept of T_2.

Notice that $<L,C>$ is actually a composite concept. Since a grafted taxonomy is still a taxonomy, all our results in Section 3 remain valid. In particular, we can obtain an optimal expression for an extensional answer **A** over a grafted taxonomy **T**.

6. Conclusions

We have considered the problem of providing abstract responses to database queries. They are given in terms of expressions of pre-defined concepts with exceptions allowed. Two criteria are defined as measures of the goodness of such expressions: (i) the minimum number of terms condition; and (ii) positive terms preferred over negative terms. We call an expression for a response satisfying the two criteria an optimal expression. Under a strict taxonomy of concepts, we are able to show that any optimal expression have exactly the same set of terms. Moreover, an algorithm for obtaining such expressions falls naturally out of its inductive proof. Generalizing the strict taxonomy of concepts to a join-semilattice of concepts, optimal expressions no longer share the same set of terms, and no efficient algorithm for obtaining such expressions can be found. Under multiple taxonomies, again, the general problem of obtaining an optimal expression is intractable, though, grafted taxonomy, involving multiple taxonomies in a more restrictive sense, can be considered.

Our definition of optimal expression is, by no means, the only way by which the goodness of an abstract response could be measured. Other reasonable criteria include putting a bound on the maximum number of negative terms in the expression, and assigning weights to concepts at different levels in a taxonomy.

Throughout the paper, we assume the extensional answer set is given a priori. An interesting question is how much the finding of optimal expressions can be integrated with conventional query processing and, as a result, abstract responses obtained more efficiently.

APPENDIX A

Here we demonstrate that expressions (α) $e^-_{k_1} \cdots e^-_{k_n}$ and (β) $-R\ e^{opt}_{k_1} \cdots e^{opt}_{k_n}$ cannot have the same number of positive and negative terms.

To show this, we compare each pair of terms $e^-_{k_i}$ and $e^{opt}_{k_i}$ in α and β respectively. Recall that $|\cdot|$ denotes the number of terms in an expression. Consider the following cases:

1. if $|e^{-R}_{k_i}| > |e^{+R}_{k_i}|$ then
   $e^{opt}_{k_i} = e^{+R}_{k_i} = S_i\ e^-_{k_i}$ and $e^{opt}_{k_i}$ has an extra *positive* term over $e^-_{k_i}$.
2. if $|e^{-R}_{k_i}| = |e^{+R}_{k_i}|$ then
   $e^{opt}_{k_i} = e^{-R}_{k_i}$ and $e^{opt}_{k_i}$ has one more term but less *negative* terms than $e^-_{k_i}$.
3. if $|e^{-R}_{k_i}| + 1 = |e^{+R}_{k_i}|$ then
   $e^{opt}_{k_i} = e^{-R}_{k_i}$ and $e^{opt}_{k_i}$ has less *negative* terms than $e^-_{k_i}$.
4. otherwise $|e^{-R}_{k_i}| + 2 = |e^{+R}_{k_i}|$ and thus
   $e^{opt}_{k_i} = e^{-R}_{k_i}$ and $e^-_{k_i} = -S_i\ e^{-R}_{k_i}$ and $e^-_{k_i}$ has an extra *negative* term.

Assume that α and β can have the same number of positive and negative terms and there are w case 1, x case 2, y case 3, and z case 4. If α and β have the same number of terms,

$$w + x + 1 = z \tag{I}$$

The + 1 comes from $-R$ in β. If the expressions have the same number of negative terms,

$$x + y + z \leq 1 \tag{II}$$

Again the 1 on the right hand side of the inequality comes from $-R$ in β. Furthermore, by our definition of taxonomy, $n \geq 2$, that is,

$$w + x + y + z \geq 2 \tag{III}$$

Since w, x, y and z are all natural numbers, it should be obvious that no solution can satisfy (I), (II) and (III) simultaneously. **Contradiction**. Thus, expressions α and β cannot have the same number of positive and negative terms.

APPENDIX B

Theorem 5.2 MULTIPLE-TAXONOMY EXPRESSION is NP-complete.

Proof: It is easy to see that MULTIPLE-TAXONOMY EXPRESSION ∈ NP since a non-deterministic algorithm need only guess an expression e and check in polynomial time whether $\xi(e) = \mathbf{A}$ and has the appropriate number of terms.

The remaining part of the proof involves the transformation of some already known NP-complete problem to our problem and showing that the transformation can be done in polynomial time. Below we define the MINIMUM COVER problem which is known to be NP-complete [7]. And then continue with a construction that transform MINIMUM COVER to MULTIPLE-TAXONOMY EXPRESSION.

MINIMUM COVER

INSTANCE: Collection C of subsets of a finite set S, positive integer $K \leq |C|$.

QUESTION: Does C contain a cover for S of size K or less, i.e., a subset $C' \subseteq C$ with $|C'| \leq K$ such that every element of S belongs to at least one member of C'?

Our transformation basically follows that of the proof that the MINIMAL DISJUNCTIVE NORMAL FORM problem is NP-complete [8]. Let $S = \{s_1, \cdots, s_n\}$ be a finite set, $C = \{c_1, \cdots, c_m\}$ be a collection of subsets of S, and a positive integer $K \leq m$ be an instance of MINIMUM COVER. We must construct taxonomies $\mathbf{T}_i$ $(1 \leq i \leq n')$, an extensional answer $\mathbf{A}$ classifiable by each taxonomy $\mathbf{T}_i$, and a positive integer $\mathbf{K}'$ such that an expression for $\mathbf{A}$ has no more than $\mathbf{K}'$ terms if and only if C has a cover of size K or less.

First, we construct n 3-concept taxonomies $\mathbf{T}_i$ $(1 \leq i \leq n)$, with r_i's as the roots, and x_i and x_i' as the only successors for each root. Taxonomies $\mathbf{T}_{n+1}$ and $\mathbf{T}_{n+2}$ are also 3-concept taxonomies, with p_1, p_2 as their roots, and y_1, y_1' and y_2, y_2', respectively, as their successors. They are named differently because, as we will see later, they serve different purposes. We will be dealing with composite concepts over n' $(=n+2)$ taxonomies. For convenience, we will refer to them as composite concepts and those which compose of no root concepts as *base* concepts.

For each element $s_i \in S$, we associate it with a base concept

$$\omega_i = \langle x_1, \cdots, x_i', \cdots, x_n, y_1, y_2 \rangle.$$

For each $c_j \in C$, we associate it with a composite concept

$$\Omega_j = \langle z_{j_1}, \cdots, z_{j_n}, y_1, y_2 \rangle \text{ where } z_{j_k} = \begin{cases} x_k & s_k \notin c_j \\ r_k & s_k \in c_j \end{cases}$$

It should be clear from Definition 5.3 that if $s_i \in c_j$, $\omega_i \leq \Omega_j$; but there are base concepts $\omega \leq \Omega_j$ such that $\omega \neq \omega_i$ for all i $(1 \leq i \leq n)$. Let D be a subset of all base concepts such that $\omega \in D$ iff $\omega \leq \Omega_j$ for some j $(1 \leq j \leq m)$ and $\omega \neq \omega_i$ for all i $(1 \leq i \leq n)$. Define

$D_{even} = \{\langle z_1, \cdots, z_n, y_1, y_2' \rangle \mid \omega = \langle z_1, \cdots, z_n, y_1, y_2 \rangle \in D$ and number of non-primed x's is even $\}$

$\Omega_{even} = \{\langle z_1, \cdots, z_n, y_1, p_2 \rangle \mid \langle z_1, \cdots, z_n, y_1, y_2' \rangle \in D_{even}\}$

$D_{odd} = \{\langle z_1, \cdots, z_n, y_1', y_2 \rangle \mid \omega = \langle z_1, \cdots, z_n, y_1, y_2 \rangle \in D$ and number of non-primed x's is odd $\}$

$\Omega_{odd} = \{\langle z_1, \cdots, z_n, p_1, y_2 \rangle \mid \langle z_1, \cdots, z_n, y_1', y_2 \rangle \in D_{odd}\}$

Notice that every base concept in D_{even} (D_{odd}) is contained exactly by one composite concept in Ω_{even} (Ω_{odd}). Moreover, every base concept in D is also contained by exactly one composite concept in either Ω_{even} or Ω_{odd}. The even-odd arrangement and the involvement of two more taxonomies, $\mathbf{T}_{n+1}$ and $\mathbf{T}_{n+2}$, are to ensure that each newly introduced composite concept has to be included in a minimal expression if $\cup D \subseteq \mathbf{A}$.

Let us complete the construction by specifying the extensional answer $\mathbf{A}$ and $\mathbf{K}'$.

$$\mathbf{A} = \bigcup \left\{ D \cup D_{even} \cup D_{odd} \cup \{\omega_i \mid 1 \leq i \leq n\} \right\}$$

$$\mathbf{K}' = K + |D|$$

Since our construction basically follows that from [3], we omit showing that the transformation can be done in polynomial time.

We claim that there is an expression for **A** with no more than $\mathbf{K}'$ terms if and only if C has a cover of size K or less. Before we show this, let us note an important observation: any expression e for **A** has an equivalent expression e' such that $|e'| \leq |e|$ and $|e'|$ has no negative terms. The proof of this is somewhat lengthy, we refer the readers to [9] for details. With this observation, negative terms can be ignored from here on.

Suppose C has a cover of size K or less. We can form an expression e_1 such that for all composite concepts $\Omega \in \Omega_{even} \cup \Omega_{odd}$, $+\Omega$ is a term in e_1. Notice that $\xi(e_1) = \cup\{D \cup D_{even} \cup D_{odd}\}$ and $|e_1| = |\Omega_{even}| + |\Omega_{odd}| = |D|$. Now all that remains of **A** is $\cup\{\omega_i \mid 1 \leq i \leq n\}$. Recall that $\omega_i \leq \Omega_j$ if $s_i \in c_j$. Since C has a cover of size K or less, we can form an expression e_2 with no more than K terms. Thus, there exist an expression $e = e_1\ e_2$ for **A** such that $|e|$ is no more than $\mathbf{K}' = K + |D|$ terms.

Suppose an expression e for **A** has no more than $\mathbf{K}'$ terms. By the way of construction, e_1 must be part of e and $|e_1| = |D|$. Finally, for each $+\Omega_j$ in e, we pick the corresponding c_j for the cover C'. Thus there is cover C' with size $K = \mathbf{K}' - |D|$ or less.

Acknowledgements

The authors would like to thank D. Stott Parker for many helpful suggestions. This research was carried out as part of the Tangram project at UCLA and was partially supported by the Defense Advanced Projects Agency under contract No. F29601-87-K-0072.

References

[1] Imielinski, T., "Intelligent Query Answering in Rule Based Systems", *J. Logic Programming*, Vol.4, No.3. September 1987.

[2] Porto, A., "Semantic Unification for Knowledge Base Deduction", *Foundations of Deductive Databases and Logic Programming*, Minker, J.(ed.), August 1986.

[3] Corella, F., "Semantic Retrieval and Levels of Abstraction", *Expert Database Systems*, Kerschberg, L. (ed.), Benjamin Cummings, New York, 1985.

[4] Lassez J.-L., Marriott K., "Explicit Representation of Terms Defined by Counter Examples", *J. Automated Reasoning* 3, 1987.

[5] Ait-Kaci, H., "Type Subsumption as a Model of Computation", *Expert Database Systems*, Kerschberg, L. (ed.), Benjamin Cummings, New York, 1985.

[6] Garey, M., Johnson, D., *Computers and Intractability - A Guide to the Theory of NP-Completeness*, Freeman, New York, 1979.

[7] Karp, R.M., "Reducibility among combinatorial problems", *Complexity of Computer Computations*, Thatcher J.W. (ed.), Plenum Press, New York, 1972.

[8] Gimpel, J.F., "A Method of Producing a Boolean Function having an Arbitrarily Prescribed Prime Implicant Table", *IEEE Trans. Computers*, Vol.14, No.3, 1965.

[9] Shum, C.D., Muntz, R., "Abstract Responses", *Technical Report*, Computer Science Department, University of California, Los Angeles, October, 1987.