QUERY ANSWERING BASED ON DISTRIBUTED KNOWLEDGE MINING

ZBIGNIEW W. RAŚ

University of North Carolina,
Department of Computer Science
Charlotte, N.C. 28223, USA
E-mail: ras@uncc.edu

Traditional query processing provides exact answers to queries. It usually requires that users fully understand the database structure and content to issue a query. Due to the complexity of the database applications, so called global queries can be posed which traditional query answering systems can not handle. In this paper a query answering system based on distributed data mining is presented to rectify these problems.

1 Introduction

In many fields, such as medical, banking and educational, similar databases are kept at many sites. Each database stores information about local events and uses attributes suitable for a local task, but since the local situations are similar, the majority of attributes are compatible among databases. An attribute may be missing in one database, while it occurs in many others.

Missing attributes lead to problems. A user may issue a query to a local database S_1 in search for objects that match a desired description, only to realize that one component a_1 of that description is missing in S_1 so that the query cannot be answered. The same query may work in other databases but the user is interested in identifying suitable objects only in S_1.

Clearly, the task of integrating established database systems is complicated not only by the differences between the sets of attributes but also by differences in structure and semantics of data. We call such systems heterogeneous. The notion of an intermediate model, proposed by [Maluf and Wiederhold][1], is very useful in dealing with the heterogeneity problem, because it describes the database content at a relatively high abstract level, sufficient to guarantee homogeneous representation of all databases. Discovery layers and action layers introduced in this paper, can be used for a similar purpose. Discovery layer contains rules extracted from a database. Actions layer contains, so called, action rules (see [Ras and Wieczorkowska][7]) showing what minimal changes in a database are needed to re-classify some of its objects.

2 Distributed Knowledge Systems

In this section, we recall the notion of an information system and a distributed information system (DIS). Next, we introduce local queries and give their standard semantics. Finally, we show the structure of discovery layers and action layers.

By an *information system* we mean $S = (X, A, V)$, where X is a finite set of objects, A is a finite set of attributes, and $V = \bigcup\{V_a : a \in A\}$ is a set of their values. We assume that:

- V_a, V_b are disjoint for any $a, b \in A$ such that $a \neq b$,
- $a : X \longrightarrow V_a$ is a function for every $a \in A$.

Instead of a, we may write $a_{[S]}$ to denote that a in an attribute in S.

By a *distributed information system* we mean a pair $DS = (\{S_i\}_{i \in I}, L)$ where:

- I is a set of sites.
- $S_i = (X_i, A_i, V_i)$ is an information system for any $i \in I$,
- L is a symmetric, binary relation on the set I.

A distributed information system $DS = (\{S_i\}_{i \in I}, L)$ is consistent if the following condition holds:

$$(\forall i)(\forall j)(\forall x \in X_i \cap X_j)(\forall a \in A_i \cap A_j)\ (a_{[S_i]}(x) = (a_{[S_j]})(x)).$$

In the remainder of this paper we assume that $DS = (\{S_i\}_{i \in I}, L)$ is consistent. Also, we assume that $S_j = (X_j, A_j, V_j)$ where $V_j = \bigcup\{V_{ja} : a \in A_j\}$, for any $j \in I$.

We use A to denote the set of all attributes in DS, $A = \bigcup\{A_j : j \in I\}$. Also, by V we mean $\bigcup\{V_j : j \in I\}$.

Before, we introduce the notion of a discovery layer, we begin with a definition of $s(i)$-terms and their standard interpretation M_i in $DS = (\{S_j\}_{j \in I}, L)$, where $S_j = (X_j, A_j, V_j)$ and $V_j = \bigcup\{V_{ja} : a \in A_j\}$, for any $j \in I$.

By a set of $s(i)$-terms (also called a set of local queries for site i) we mean a least set T_i such that:

- $\mathbf{0}, \mathbf{1} \in T_i$,
- $w \in T_i$ for any $w \in V_i$,

- if $t_1, t_2 \in T_i$, then $(t_1 + t_2), (t_1 * t_2), \sim t_1 \in T_i$.

By a set of $s(i)$-formulas we mean a least set F_i such that:

- if $t_1, t_2 \in T_i$, then $(t_1 = t_2) \in F_i$.

Definition of DS-terms (also called a set of global queries) and DS-formulas is quite similar (we only replace T_i by $\bigcup\{T_i : i \in I\}$ and F_i by F in two definitions above).

We say that:

- $s(i)$-term t is *primitive* if it is of the form $\prod\{w : w \in U_i\}$ for any $U_i \subseteq V_i$,
- $s(i)$-term $t = \prod\{w : w \in U_i\}$ where $U_i \subseteq V_i$ is *simple* if $U_i \cap V_{ia}$ is a singleton set for any $a \in A_i$,
- $s(i)$-term is in *disjunctive normal form* (DNF) if $t = \sum\{t_j : j \in J\}$ where each t_j is primitive.

Similar definitions we have for DS-terms.

Clearly, it is easy to give an example of a local query. The expression:

select $*$ from $Flights$
where $airline =$ "$Delta$"
and $departure\_time =$ "$morning$"
and $departure\_airport =$ "$Charlotte$"

is an example of a non-local query (DS-term) in a database

$Flights(airline, departure\_time, arrival\_time,$
$departure\_airport, arrival\_airport)$.

Semantics of $s(i)$-terms is defined by standard interpretation M_i in a distributed information system $DS = (\{S_j\}_{j \in I}, L)$ as follows:

- $M_i(\mathbf{0}) = \emptyset$, $M_i(\mathbf{1}) = X_i$
- $M_i(w) = \{x \in X_i :$ if $w \in V_{ia}$ then $w = h_i(x, a)\}$ for any $w \in V_i$,
- if t_1, t_2 are s(i)-terms, then

  $M_i(t_1 + t_2) = M_i(t_1) \cup M_i(t_2)$,
  $M_i(t_1 * t_2) = M_i(t_1) \cap M_i(t_2)$,
  $M_i(\sim t_1) = X_i - M_i(t_1)$.
  $M_i(t_1 = t_2) =$
  (if $M_i(t_1) = M_i(t_2)$ then T else F)
  where T stands for $True$ and F for $False$

The sound and complete axiomatization of the above semantics is quite standard and for instance is given in paper by [Ras][6].

Now, we are ready to introduce the notion of (k,i)-rules, for any $i \in I$. We use them to form a discovery layer at site $i \in I$.

By (k,i)-rule in $DS = (\{S_j\}_{j \in I}, L)$, $k, i \in I$, we mean a triple (c,t,s) such that:

- $c \in V_k - V_i$,
- t, s are $s(k)$-terms in DNF and they both belong to $T_k \cap T_i$,
- $M_k(t) \subseteq M_k(c) \subseteq M_k(t+s)$.

For any (k,i)-rule (c,t,s) in $DS = (\{S_j\}_{j \in I}, L)$, we say that:

- $(t \rightarrow c)$ is a k-certain rule in DS,
- $(t + s \rightarrow c)$ is a k-possible rule in DS.

Let us assume that $r_1 = (c_1, t_1, s_1)$, $r_2 = (c_2, t_2, s_2)$ are (k,i)-rules. We say that: r_1, r_2 are strongly consistent, if either c_1, c_2 are values of two different attributes in S_k or a DNF form equivalent to $t_1 * t_2$ does not contain simple conjuncts.

Now, we are ready to define a discovery layer D_{ki}. Its elements can be seen as approximate descriptions of values of attributes from $V_k - V_i$ in terms of values of attributes from $V_k \cap V_i$.

To be more precise, we say that D_{ki} is a set of (k,i)-rules such that:
if $(c,t,s) \in D_{ki}$ and $t_1 = \sim (t+s)$, then $(\sim c, t1, s) \in D_{ki}$.

By a discovery layer for site i, denoted by D_i, we mean any subset of $\bigcup\{D_{ki} : (k,i) \in L\}$.

3 Actions Layer

In this section we introduce the notion of actions layer which is a basic part of a distributed knowledge system (DKS).

Information systems can be seen as decision tables . In any decision table together with the set of attributes a partition of that set into conditions and decisions is given. Additionally, we assume that the set of conditions is partitioned into stable conditions and flexible conditions . Attribute $a \in A$ is called stable for the set X if its values assigned to objects from X can not be

changed in time. Otherwise, it is called flexible. Date of birth is an example of a stable attribute. Interest rate on any customer account is an example of a flexible attribute. For simplicity reason, we consider decision tables with only one decision. We adopt the following definition of a decision table:

A decision table is any information system of the form $S = (X, A_1 \cup A_2 \cup \{d\}, V)$, where $d \notin A_1 \cup A_2$ is a distinguished attribute called decision. The elements of A_1 are called stable conditions, whereas the elements of $A_2 \cup \{d\}$ are called flexible conditions.

The goal is to change values of attributes in A_1 for some objects in X so the values of the attribute d for these objects may change as well. Rules in a discovery layer defining d in terms of $A_1 \cup A_2$ are extracted from S and used to discover new rules called action rules [7]. These new rules provide suggestions for re-classification of objects from S in terms of the attribute d. It can be done because d is flexible.

Now, let us assume that $(a, v \longrightarrow w)$ denotes the fact that the value of attribute a has been changed from v to w. Similarly, the term $(a, v \longrightarrow w)(x)$ means that $a(x) = v$ has been changed to $a(x) = w$. Saying another words, the property (a, v) of object x has been changed to property (a, w).

Assume now that $S = (X, A_1 \cup A_2 \cup \{d\}, V)$ is a decision table, where A_1 is a set of stable attributes and A_2 is a set of flexible attributes. Assume that rules r_1, r_2 have been extracted from S and $r_1/A_1 = r_2/A_2, d(r_1) = k_1, d(r_2) = k_2$ and $k_1 < k_2$. Also, assume that $(b_1, b_2, ..., b_p)$ is a list of all attributes in $Dom(r_1) \cap Dom(r_2) \cap A_2$ on which r_1, r_2 differ and $r_1(b_1) = v_1, r_1(b_2) = v_2, ..., r_1(b_p) = v_p$.

By (r_1, r_2)-action rule on $x \in X$ we mean a statement:

$[(b_1, v_1 \longrightarrow w_1) \wedge (b_2, v_2 \longrightarrow w_2) \wedge ..... \wedge (b_p, v_p \longrightarrow w_p)](x) \Rightarrow [(d, k_1) \longrightarrow (d, k_2)](x)$.

If the value of the above rule is true on x then the rule is valid for x. Otherwise is false.

Action layer for a site i, denoted by Act_i, contains (r_1, r_2)-action rules constructed from rules r_1, r_2 in a discovery layer D_i.

4 Distributed Knowledge System

In this section, we introduce the notion of a distributed knowledge system.

By Distributed Knowledge System (DKS) we mean $DS = (\{(S_i, D_i, Act_i)\}_{i \in I}, L)$ where $(\{S_i\}_{i \in I}, L)$ is a distributed information system, $D_i = \bigcup \{D_{ki} : (k, i) \in L\}$ is a discovery layer and Act_i is an action layer for $i \in I$.

Figure 1 shows the basic architecture of DKS (a query answering system QAS that handles global queries is also added to each site of DKS). Opera-

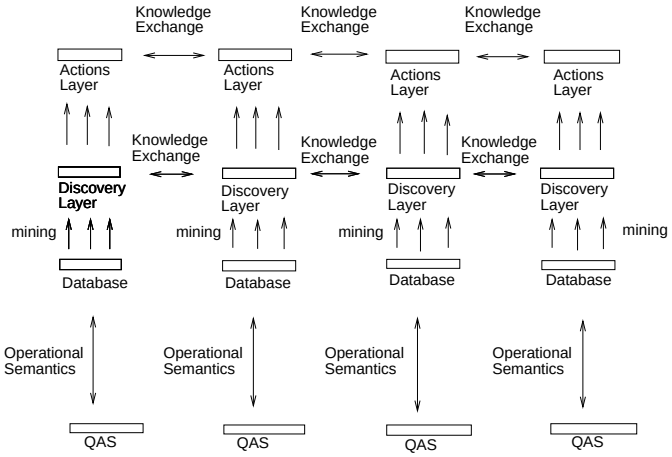

Figure 1: Distributed Knowledge System (DKS)

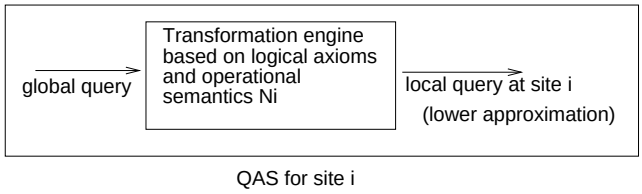

Figure 2: Query Answering System (QAS)

tional semantics reflects the dynamic nature of definitions of attribute values in a query (see [Ras and Zytkow[8]).

Figure 2 shows a part of QAS which is responsible for query transformation. This part of QAS can be replaced by a rough transformation engine shown in Figure 3.

If for each non-local attribute we collect rules from many sites of DKS and then resolve all inconsistencies among them (see [Ras[5]), then the local confidence in resulting operational definitions is high since they represent consensus of many sites.

Assume now that N_i is a standard interpretation of global queries as introduced for instance in [Ras[8]. It corresponds to a pessimistic approach to

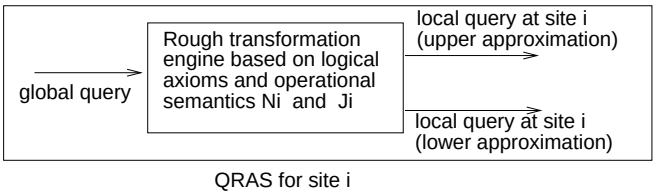

Figure 3: Query Rough Answering System (QRAS)

evaluation of global queries because of the way the non-local attribute values are interpreted (their lower approximation is taken).

We can replace N_i by a new interpretation J_i representing optimistic approach to evaluation of global queries. Namely, we define:

- $J_i(w) = X - N_i(\sim w)$,
- $J_i(\sim w) = X - N_i(w)$,
- $J_i(t) = N_i(t)$ for any other t.

In optimistic approach to evaluation of queries, upper approximation of non-local terms w, $\sim w$ is taken.

Following this line of thought, we can propose rough operational semantics R_i defined as $R_i(t) = [N_i(t), J_i(t)]$ for any global query t. Rough operational semantics has a natural advantage of either N_i or J_i. Clearly, if interpretations N_i and J_i of a term t give us the same sets of objects, then both approximations (lower and upper) are semantically equal.

5 Query Answering Based on Reducts

In this section we recall the notion of a reduct (see [Pawlak[3]) and show how it can be used to improve query answering process in DKS.

Let us assume that $S = (X, A, V)$, is an information system and $V = \bigcup\{V_a : a \in A\}$. Let $B \subset A$. We say that $x, y \in X$ are indiscernible by B, denoted $[x \approx_B y]$, if $(\forall a \in B)[a(x) = a(y)]$.

Now, assume that both B_1, B_2 are subsets of A. We say that B_1 depends on B_2 if $\approx_{B_2} \subset \approx_{B_1}$. Also, we say that B_1 is a covering of B_2 if B_2 depends on

B_1 and B_1 is minimal.

By a reduct of A in S (for simplicity reason we say A-reduct of S) we mean any covering of A.

Example. Assume the following scenario:

- $S_1 = (X_1, \{c, d, e, g\}, V_1)$, $S_2 = (X_2, \{a, b, c, d, f\}, V_2)$, $S_3 = (X_3, \{b, e, g, h\}, V_3)$ are information systems,
- User submits a query $q = q(c, e, f)$ to the query answering system QAS associated with system S_1,
- Systems S_1, S_2, S_3 are parts of DKS.

Attribute f is non-local for a system S_1 so the query answering system associated with S_1 has to contact other sites of DKS requesting a definition of f in terms of $\{d, c, e, g\}$. Such a request is denoted by $< f : d, c, e, g >$. Assume that the system S_2 is contacted. The definition of f, extracted from S_2, involves only attributes $\{d, c, e, g\} \cap \{a, b, c, d, f\} = \{c, d\}$. There are three f-reducts (coverings of f) in S_2. They are: $\{a, b\}, \{a, c\}, \{b, c\}$. The optimal f-reduct is the one which has minimal number of elements outside $\{c, d\}$. Let us assume that $\{b, c\}$ is chosen as an optimal f-reduct in S_2.

Then, the definition of f in terms of attributes $\{b, c\}$ will be extracted from S_2 and the query answering system of S_2 will contact other sites of DKS requesting a definition of b (which is non-local for S_1) in terms of attributes $\{d, c, e, g\}$. If definition of b is found, then it is sent to QAS of the site 1. Figure 4 shows the process of resolving query q in the example above.

We will use the graph in Figure 5 to represent visually the fact: $R[i]$ is an a-reduct at site i containing attribute b.

Let us adopt the following definition. By $< a_1, A >$-linear set of reducts we mean a set $\{< a_i, R[i] >: 1 \leq i \leq k\}$ such that:

- $a_i \notin A$, for any $1 \leq i \leq k$
- $a_{i+1} \in R[i]$, for any $1 \leq i \leq k-1$
- $R[i]$ is an a_i-reduct at site i and $\mathrm{card}(A - R[i])$=1, for any $1 \leq i \leq k-1$
- $R[k] \subseteq A$.

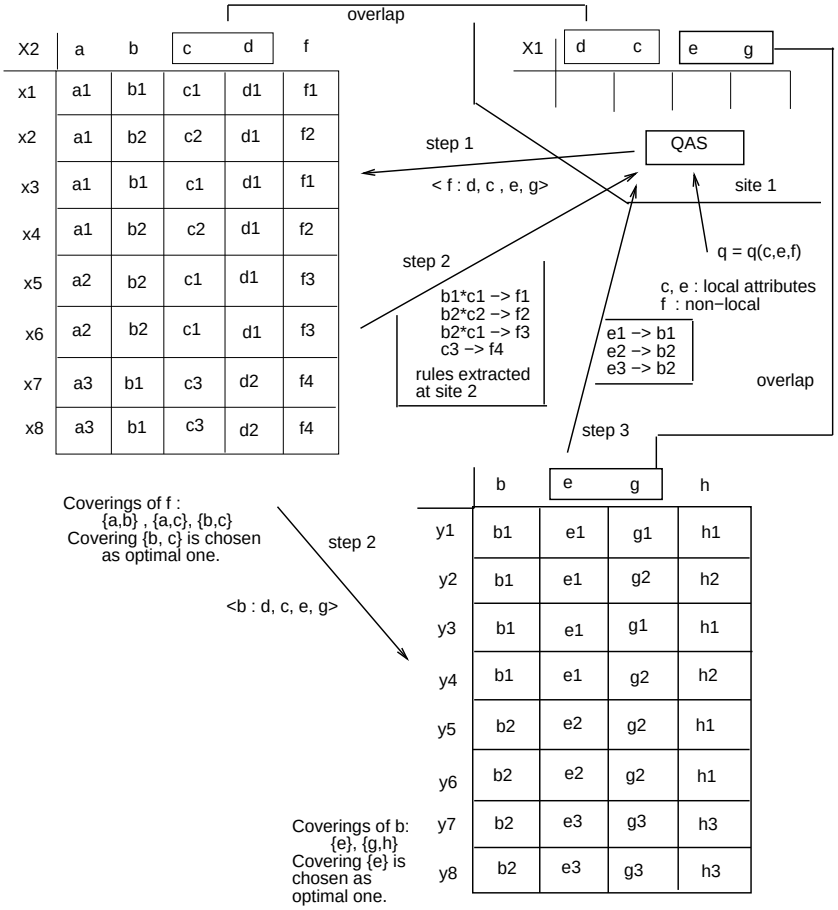

Figure 4: Process of resolving a query by QAS in DKS

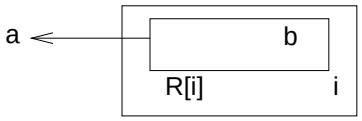

Figure 5: R[i]: a-reduct at site i containing attribute b

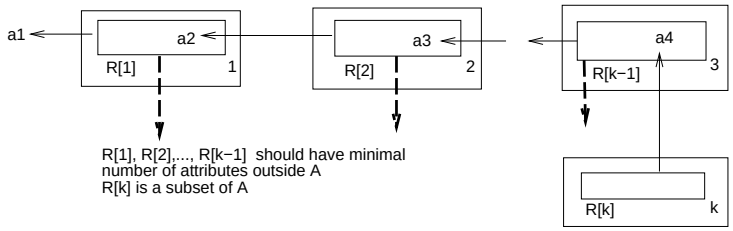

Figure 6: $< a_1, A >$-linear set of reducts

Figure 6 visually represents $< a_1, A >$-linear set of reducts. Clearly, the existence of $< a, A >$-linear set of reducts is sufficient for attribute a to be definable in DKS. The existence of $< a, A >$-directed set of reducts (defined below) is necessary for attribute a to be definable in DKS.

By $< a_1, A >$-directed set of reducts we mean a smallest, non-empty set $\{< a_i, R[i], s_i >: 1 \leq i \leq k\}$ such that:

- $a_i \notin A$, for any $1 \leq i \leq k$
- s_i is a site of DKS, for any $1 \leq i \leq k$
- $R[i]$ is an a_i-reduct at site s_i, for any $1 \leq i \leq k$
- $(\forall a \in \bigcup\{R[i] : i \leq k\} - A)(\exists j \leq k)[a = a_j]$
- $R[k] \subseteq A$

Clearly, for every (a_1, A) we have to search for the smallest $< a_1, A >$-directed set of reducts, to guarantee the smallest number of steps needed to learn the definition of attribute a_1 while keeping the confidence of what we learn still the highest.

6 Conclusion

Query answering system for DKS can handle two types of queries:

Queries asking for all objects at a site i which satisfy a given description (any attributes are allowed to be used here). In such a case, query answering system will search for operational definitions of all attributes not-existing at the site i, before it can process the query locally.

Queries asking for actions which have to be undertaken in order to change the classification of some objects at site i. Such queries can be processed

entirely at site i or moved for remote processing to other sites of DKS. In the last case, operational definitions of all attributes from the site i in terms of attributes from another site are needed. But, this problem will be a topic of a separate paper.

References

1. Maluf, D., Wiederhold, G., "Abstraction of representation for interoperation", in *Proceedings of Tenth International Symposium on Methodologies for Intelligent Systems*, LNCS/LNAI, Springer-Verlag, No. 1325, 1997, 441-455
2. Navathe, S., Donahoo, M., "Towards intelligent integration of heterogeneous information sources", in *Proceedings of the Sixth International Workshop on Database Re-engineering and Interoperability*, 1995
3. Pawlak, Z., "Rough classification", in *International Journal of Man-Machine Studies*, Vol. 20, 1984, 469-483
4. Prodromidis, A.L. & Stolfo, S., "Mining databases with different schemas: Integrating incompatible classifiers", in *Proceedings of The Fourth Intern. Conf. onn Knowledge Discovery and Data Mining*, AAAI Press, 1998, 314-318
5. Ras, Z., "Dictionaries in a distributed knowledge-based system", in *Concurrent Engineering: Research and Applications, Conference Proceedings*, Pittsburgh, Penn., Concurrent Technologies Corporation, 1994, 383-390
6. Ras, Z., "Resolving queries through cooperation in multi-agent systems", in *Rough Sets and Data Mining* (Eds. T.Y. Lin, N. Cercone), Kluwer Academic Publishers, 1997, 239-258
7. Ras, Z., Wieczorkowska, A., "Action Rules: how to increase profit of a company", in *Principles of Data Mining and Knowledge Discovery*, D.A. Zighed, J. Komorowski, J. Zytkow (Eds), Proceedings of PKDD'00, Lyon, France, LNCS/LNAI, No. 1910, Springer-Verlag, 2000, 587-592
8. Ras, Z., Żytkow, J., "Mining for attribute definitions in a distributed two-layered DB system", *Journal of Intelligent Information Systems*, Kluwer, Vol. 14, No. 2/3, 2000, 115-130
9. Ras, Z., Żytkow, J.,"Discovery of equations to augment the shared operational semantics in distributed autonomous BD System", in *PAKDD'99 Proceedings*, LNCS/LNAI, No. 1574, Springer-Verlag, 1999, 453-463
10. Żytkow, J.M., Zhu, J., and Zembowicz R. Operational Definition Refinement: a Discovery Process, *Proceedings of the Tenth National Conference on Artificial Intelligence*, The AAAI Press, 1992, p.76–81.