Building A Security-Aware Query Answering System Based On Hierarchical Data Masking

Seunghyun Im
[1] University of North Carolina at Charlotte
Department of Computer Science,
Charlotte, N.C. 28223, USA
sim@uncc.edu

Zbigniew W. Ras[1,2]
[2] Polish Academy of Sciences
Institute of Computer Science,
Ordona 21, 01-237 Warsaw, Poland
ras@uncc.edu

Agnieszka Dardzińska[3,1]
[3] Bialystok Technical University
Mathematics Dept.,
15-351 Bialystok, Poland
adardzin@uncc.edu

Abstract

Knowledge based query answering system takes advantage of data mining techniques to provide answers to user queries that involve incomplete or foreign attribute values. However, such process may cause a security issue when the system contains confidential data required to be protected. The confidential data as well as other values can be treated as missing or incomplete, and vulnerable to improper disclosure of their true values. To minimize such vulnerability, data transformation techniques are often utilized. In this paper, we present a method that exploits hierarchical structure of attributes to minimize the changes from the original information system while accommodating a given security requirement. In particular, our method replaces the existing data with more generalized ones in such a way that the value replacement cannot be used to predict the confidential data.

1 Introduction

Knowledge based Query Answering Systems (*QAS*) is to discover rules either locally or at remote sites (if system is distributed) and use these rules in a query answering process. There are two different situations within this objective. The first is when attributes are incomplete and we may need rules to approximate the incomplete values to answer to a query. The second is when users want to ask queries based on some attributes which are not listed in a local domain. Since these attributes are locally not available, we can only search for their definitions at remote sites and use them to approximate given queries [6] [8]. One way to design query answering systems more flexible is to apply a hierarchical structure to their attributes [12]. Unlike single-level attribute system, data collected with different granularity levels can be assigned into an information system with their semantic relations. For example, when the age of a person is recorded, the value can be *17* or *young* .

We can expect higher probability of answering user queries successfully by using a *QAS*. However, it may create a security problem when a set of data is confidential (e.g. age or salary) and protection is required. The exact value of confidential data can be concealed from an information system, for example, by replacing them with null values. However, users can send a query to *QAS* asking the confidential data as incomplete or foreign, and *QAS* returns the hidden data.

When we design a protection method against such improper disclosure, one approach is to transform a set of data to null values [5]. In other word, we completely hide a set of existing values that are used to predict the confidential data. Another approach, which we will discuss in this paper, is to mask the exact value by substituting it with more generalized values at higher level of a hierarchical attribute structure. For example, instead of showing a person is *17* years old we may show that she is *young* if disclosure of the value *young* does not compromise the privacy of the person. The advantage of the second approach that users will be able to acquire more explicit answers to their queries.

Clearly, we need to assume that a hierarchical attribute structure is given to each attribute and they are part of the common ontology which is large and approximately the same among sites. They should come from the same world (e.g. medical information) and, consequently, rules generated from different sites are close in terms of their meanings. In addition, each site is forced to accept a new version of ontology if any change has been made. Also the hierarchical structure must be seen by users. Users have freedom of querying any level of values in the hierarchy.

The tradeoff between security risk and information availability is relatively clear. As the amount of hidden or rough data continues to grow the disclosure of confidential data drops. However, it is important to retain the original sources of information to the maximum extent possible to maintain the *QAS* to return more precise answers. In that respect, the method presented in this paper aims to minimize the amount of data replacement in terms of value granularity while making sure that *QAS* will not reveal the data below the safe levels of granularity.

2 Null Value Imputation in Distributed Query Answering System

2.1 Distributed Query Answering System and Chase

In real life, data are often collected and stored in information systems residing at many different locations, built independently, instead of collecting them and storing at only one single location. In such cases we talk about distributed (autonomous) information systems. It is very possible that an attribute is missing or hidden in one of them while it occurs in many others. Also, in one information system, an attribute might be partially hidden, while in other systems the same attribute is either complete or close to being complete. Assume that user submits a query to one of the information systems (called a client) which involves some hidden or non-local attributes. In such a case, network communication technology is used to get definitions of these unknown or hidden attributes from other information systems (called servers). All these new definitions form a knowledge base which can be used to chase both missing and hidden attributes at the client site.

In Figure 1, we present two consecutive states of a distributed information system consisting of S_1, S_2, S_3. In the first state, all values of all hidden attributes in all three information systems have to be identified. System S_1 sends request q_{S_1} to the other two information systems asking them for definitions of its hidden attributes. Similarly, system S_2 sends request q_{S_2} to the other two information systems asking them for definitions of its hidden attributes. Now, system S_3 sends request q_{S_3} to the other two information systems also asking them for definitions of its hidden attributes. Next, rules describing the requested definitions are extracted from each of these three information systems and sent to the systems which requested them. It means, the set $L(D_1)$ is sent to S_2 and S_3, the set $L(D_2)$ is sent to S_1 and S_3, and the set $L(D_3)$ is sent to S_1 and S_2.

The second state of a distributed information system, presented in Figure 1, shows all three information systems with the corresponding $L(D_i)$ sets, $i \in \{1, 2, 3\}$, all abbreviated as KB. Now, the $Chase$ algorithm [9] is run independently at each of our three sites. Resulting information systems are: $Chase(S_1)$, $Chase(S_2)$, and $Chase(S_3)$. Now, the whole process is recursively repeated. It means, both hidden and incomplete attributes in all three new information systems are identified again. Next, each of these three systems is sending requests to the other two systems asking for definitions of its either hidden or incomplete attributes and when these definitions are received, they are stored in the corresponding KB sets. Now, $Chase$ algorithm is run again at each of these three sites. The whole process is repeated till some fixed point is reached (no changes in attribute values assigned to objects are observed in all 3 systems). When this step is accomplished, a query containing some hidden attribute values can be submitted to any S_i, $i \in \{1, 2, 3\}$ and processed in a standard way.

2.2 Null Value Imputation Algorithm Chase

Let us examine in more detail the *Chase* algorithm. Assume that information is stored in an information system $S = (X, A, V)$, where X is a set of objects, A is a finite set of attributes, and V is a finite set of their values. In particular, we say that $S = (X, A, V)$ is an incomplete information system of type λ if the following three conditions hold:

- $a_S(x)$ is defined for any $x \in X$, $a \in A$,
- $(\forall x \in X)(\forall a \in A)[(a_S(x) = \{(a_i, p_i) : 1 \leq i \leq m\}) \to \sum_{i=1}^{m} p_i = 1]$,
- $(\forall x \in X)(\forall a \in A)[(a_S(x) = \{(a_i, p_i) : 1 \leq i \leq m\}) \to (\forall i)(p_i \geq \lambda)]$.

Incompleteness is understood by having a set of weighted attribute values as a value of an attribute. Now, suppose that $L(D) = \{(t \to v_c) \in D : c \in In(A)\}$ (called a knowledge-base) is a set of all rules extracted from $S = (X, A, V)$ by $ERID(S, \lambda_1, \lambda_2)$, where $In(A)$ is the set of incomplete attributes in S and λ_1, λ_2 are thresholds for minimum support and minimum confidence, correspondingly. *ERID* [11][2] is the algorithm for discovering rules from incomplete information systems, and used as a part of null value imputation algorithm *Chase* [8]. Assume now that a

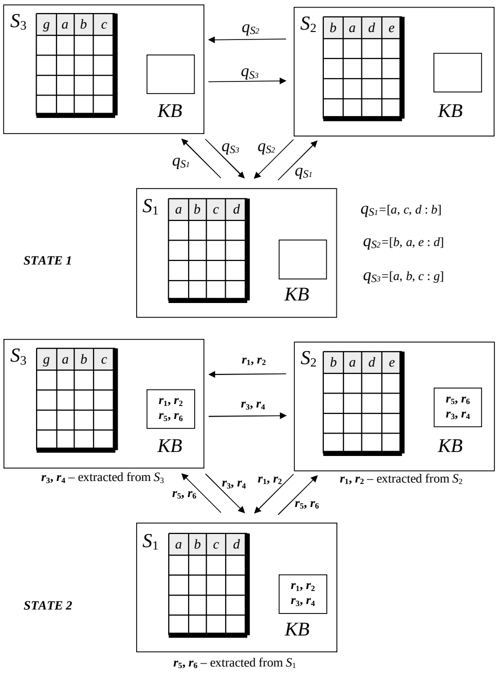


Figure 1. Global extraction and exchange of knowledge

query $q(B)$ is submitted to system $S = (X, A, V)$, where B is the set of all attributes used in $q(B)$ and that $A \cap B \neq \emptyset$. Attributes in $B - [A \cap B]$ are called either foreign or hidden in S. If S is a part of a distributed information system, definitions of such attributes can be extracted at remote sites for S [7].
The new definition replaced by the imputation algorithm is computed as following. Let $R_s(x_i) \subseteq L(D)$ be the set of rules that the conditional part of the rules is equal to the attribute values in $x_i \in S$, and d be a null value. Then there are three cases:

- $R_s(x_i) = \phi$ In this case, d cannot be predicted.
- $R_s(x) = \{r_1 = [t_1 \to d_1], r_2 = [t_2 \to d_1], ..., r_k = [t_k \to d_1]\}$ In this case, every rule implies a single decision attribute value, and $d = d_1$.
- $R_s(x_i) = \{r_1 = [t_1 \to d_1], r_2 = [t_2 \to d_2], ..., r_k = [t_k \to d_k]\}$ In this case, rules imply multiple decision values.

The confidence for the attribute value d for x_i driven by KB is defined as following [5]. Assuming that support and confidence of a rule r_i is $[s_i, c_i]$, and the product of the weight of each attribute value that matches to $a(x) \in t_i$ is $\prod p_{a(t_i)}$, for $i \leq k$.

$$conf(d') = \frac{\sum\{[\prod p_{a(t_i)}] \cdot s_i \cdot c_i : [d' = d_i]\}}{\sum\{[\prod p_{a(t_i)}] \cdot s_i \cdot c_i\}}, 1 \leq i \leq k$$

We replace the null value with d' when each $conf(d') > \lambda$.

2.3 Inconsistency

As we already pointed out, the knowledge base $L(D)$, contains rules extracted locally at the client site (information system queried by user) as well as rules extracted from information systems at its remote sites. Since rules are extracted from different information systems, inconsistencies in semantics, if any, have to be resolved before any query can be processed. There are two options:

- a knowledge base $L(D)$ at the client site is kept consistent (in this scenario all inconsistencies have to be resolved before rules are stored in the knowledge base)
- a knowledge base at the client site is inconsistent (values of the same attribute used in two rules extracted at different sites may be of different granularity levels and may have different semantics associated with them).

In general, we assume that the information stored in ontologies [4], [14] and, if needed, in inter-ontologies (if they are provided) is sufficient to resolve inconsistencies in semantics of all sites involved in $Chase$. Inconsistencies related to the confidence of conflicting rules stored in $L(D)$ do not have to be resolved at all (algorithm $Chase$ does not have such a requirement). The fact, that rules stored in $L(D)$ can be extracted at different sites and under different interpretations of incomplete values, is not pleasant assuming that we need to use them in $Chase$. In all such cases, following the same approach as in [7], rough semantics can be used for interpreting rules in $L(D)$.

One of the problems related to an incomplete information system $S = (X, A, V)$ is the freedom how new values are constructed to replace incomplete values in S, before any rule extraction process begins. This replacement of incomplete attribute values can be done either by $Chase$ or/and by a number of available statistical methods [3]. This implies that semantics of queries submitted to S and queries processed by the query answering system QAS based on $Chase$, may often differ. In such cases, following again the approach in [7], rough semantics can be used by QAS

to handle this problem. In this paper we assume that the semantic of attribute hierarchy is consistent among all the sites. For example, if $a \in A_i \cap A_j$, then only the granularity levels of a in S_i and S_j may differ but conceptually its meaning, both in S_i and S_j is the same.

2.4 Rule Extraction and Hierarchical Attribute

Before we discuss *Chase* and data security, we need to examine how rules are generated from S that is represented in hierarchical attribute structures. Assume that an information system $S = (X, A, V)$ is a partially incomplete information system of type λ, and a set of tree-like attribute hierarchy H_S is assigned to S where $h_a \in H_S$ represents all possible values of an attribute $a \in A$. If we denote a node in t_a as a_i, the set $\{a_{ik} : 1 \leq k \leq m\}$ contains all the children of a_i as shown in Figure 2.

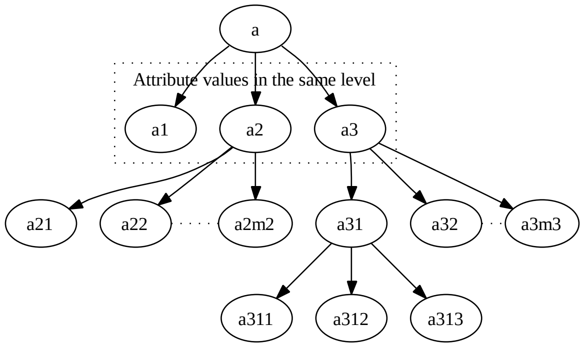


Figure 2. Hierarchical Attribute Structure

Many different combinations of attribute levels can be chosen for rule extraction. To extract rules at particular levels of interest in S, we need to transform attribute values before the rule extraction algorithm *ERID* is executed. In the following, we will use the term 'generalization' of $a(x)$ to refer to the transformation of $a(x)$ to a node value on the path from $a(x)$ to the root node in the hierarchy, and 'specification' to mean a transformation of $a(x)$ to a node value on the path to the leaf node. As defined, each attribute value in an incomplete information system is a value/weight pair $(a(x), p)$. When attribute values are transformed, the new value and weight are interpreted as the following,

- if $a(x)$ is specialized, it is replaced by a null value. This means that a parent node is considered as a null value for any child node.
- if $a(x)$ is generalized, it is replaced by $a_i \in t_a$ at the given level on the path. The weight of the new value is the sum of the children nodes. Intermediate nodes placed along the path, if exist, are computed in the same way. That is $p'_{a(x)} = \sum p_{a(x)ik}, (1 \leq k \leq m, p_{a(x)ik} \geq \lambda)$.

Clearly, the root node in each tree is an attribute name, and it is equivalent to a null value. Null value assigned to an object is interpreted as all possible values of an attribute with equal confidence assigned to all of them. Now, let L_H be the set of level of attributes to be used, λ_1 be the support, and λ_2 be the confidence value. *ERID* for hierarchical attributes is represented as *ERID-H*$(S, H_S, L_H, \lambda_1, \lambda_2)$.

3 Data Security and Chase Applicability

3.1 Problem of Data Confidentiality

To illustrate the data confidentiality problem, let's consider the following example. Suppose a local information system $S \in S_i$ for $i \in I$ operates in a distributed *QAS* as shown in Table 1. We assume that values in S are stored under hierarchical attribute structures H_S that is part of global ontology [7] for *QAS*. Now, an attribute d in S contains a set of confidential data, and S_d (see Table 2) has been built by replacing the exact values of d with values at a higher level in the hierarchical attributes structure that is considered as a secure level. User queries are now responded by S_d in replace of S. However, disclosure risk still remains because users may treat the confidential data as incomplete or foreign and contact remote sites in *QAS*. Clearly, $d_{S_d}(x)$ predicted by *Chase* can be equal to $d_S(x)$ for a number of objects [10]. Another vulnerability may be present in a similar way that users employ locally generated rules to predict attribute values contained in the rules that are extracted from remote sites [5]. For example, suppose we have generalized a set of additional attribute values to block the rules extracted from remote sites. However, there may be some local rules that reconstruct those additionally replaced values, and d can be predicted again.

3.2 Chase Applicability

Suppose that a knowledge base *KB* for S contains a set of rules. In order for the $Chase$ algorithm to be applicable to S, it has to satisfy the following conditions [7]:

- attribute value used in the decision part of a rule form *KB* has the granularity level either equal to or finer than the granularity level of the corresponding attribute in S.
- the granularity level of any attribute used in the classification part of a rule from *KB* should be either equal or softer than the granularity level of the corresponding attribute in S.

X	*A*	*B*	*C*	*D*	*E*	*F*
x_1	$(a_{[1,1,1]},\frac{2}{3})(a_{[2,1,1]},\frac{1}{3})$	$b_{[1,1,1]}$	$c_{[1,1,1]}$	$d_{[1,1,2]}$	$e_{[1,1,1]}$	$f_{[1,1,1]}$
x_2	$a_{[2,2,1]}$	$b_{[2,1,1]}$	$c_{[1,1,1]}$	$d_{[2,3,2]}$	$e_{[2,1,1]}$	$f_{[1,3,2]}$
x_3	$a_{[1,1,2]}$	$(b_{[1,1,1]},\frac{1}{2})(b_{[2,1,2]},\frac{1}{2})$	$c_{[1,1,1]}$	$d_{[1,3,2]}$	$e_{[1,1,1]}$	$f_{[1,3,1]}$
x_4	$a_{[2,2,1]}$	$b_{[2,1,2]}$	$c_{[2,1]}$	$(d_{[1,1,1]},\frac{2}{3})(d_{[2,1,1]},\frac{1}{3})$	$e_{[2,1,1]}$	$f_{[1,1,1]}$
x_5	$a_{[1,1,2]}$	$b_{[2,3,2]}$	$c_{[1,2,1]}$	$d_{[1,2]}$	$e_{[2,3,2]}$	$f_{[1,2,2]}$
x_6	$(a_{[1,1,1]},\frac{2}{3})(a_{[2,1,1]},\frac{1}{3})$	$b_{[2,2,1]}$	$c_{[1,3,1]}$	$d_{[1,1,2]}$	$e_{[2,3,2]}$	$f_{[1,1,2]}$
			.			
			.			
x_i	$a_{[1,2,1]}$	$b_{[2,1,1]}$	$c_{[1,1]}$	$d_{[1,2,2]}$	$e_{[1,1,1]}$	$f_{[1,1,1]}$

Table 1. Information System S

The set of values predicted by *Chase* may consist of horizontally and vertically different values. That means when two or more rules are supported by an object x_i, the predicted set can contain multiple values which granularity levels in H_S are the same (e.g. $\{d_{[1]}, d_{[2]}\}$), or different (e.g. $\{d_{[1]}, d_{[1,2]}\}$). We use the same confidence calculation function discussed in section 2.2 and allow all such cases to be valid predictions.

4 Method Description

We present an algorithm that protects values of a confidential attribute from $Chase$ algorithm. We continue to assume that attribute $d \in S$ contains confidential values, and all $d(x_i)$ are generalized to the values at the 2^{nd} level of H_S. The new information system S_d is shown in Table 2. The structure of each attribute hierarchy is same as that of $a \in A$ as illustrated in Figure 1 that has four levels and one to three nodes in each level. The rules in the knowledge base *KB* are summarized in table 3. For instance $r_1 = [c_{[1,1,1]} \rightarrow d_{[1,1,2]}]$ is an example of a rule belonging to *KB*. We use $\lambda = 0.3$ and $\tau = 0.8$. Based on above assumptions, we define the following sets:

- $\alpha(x)$, the set of attribute values used to describe x in S_d
- $\alpha(t)$, the set of attribute values used in t, where t is their conjunction
- *R*$(x) = \{(t \rightarrow d) : \alpha(t) \subseteq \alpha(x)\} \subseteq$ *KB*, the set of rules in *KB* where the attribute values used in t are contained in $\alpha(x)$
- $\beta(x) = \cup\{\alpha(t) \cup \{d\} : [t \rightarrow d] \in$ *R*$(x)\}$.

Our protection strategy consists of two phases: First, we identify the set of attribute values that are used for prediction for the confidential values. In the second phase, the values identified in the first phase are generalized. Before we discuss the phases in detail, we introduce the notion of chase closure and validity of prediction.

4.1 Chase Closure and Validity of Prediction

To find the minimum amount of values that are used for prediction for the confidential values, a bottom up approach has been adapted. We check the values that will remain unchanged starting from a singleton set containing attribute value a by using *chase closure* and increase the initial set size as much as possible. Chase closure is similar to transitive closure [1] except that if the weight of a predicted value is less than λ, the value is not added to the closure. For example, an object x_5 supports three rules, $\{r_6, r_7, r_8\} \in KB$ that predict $\{b_{[2,3,1]}, b_{[2,3,2]}\}$. In this case, $b_{[2,3,1]}$ is not included in the closure because $p'_{d_{[2,3,1]}} = \frac{45}{235} < 0.3$.

Identification method based on chase closure automatically rules out any superset of must-be-hidden values, and minimizes the computational cost. The justification of this is quite simple. Chase closure has the property that the superset of a set s also contains s. Clearly, if a set of attribute values predicts d_1, then the set must be hidden regardless of the presence/abscence of other attribute values.

When a confidential value has been predicted by *Chase*, the weight of the predicted value may be substantially different from that of the actual value. If this is the case, protection is not required because an adversary cannot have enough confidence in the confidential value. In order to determine whether a prediction is valid we define a measurement function and compare it to the threshold value τ. Suppose that the weight of an actual confidential value d_i is denoted as $p_{d_{[i]}}$ and weight of the predicted value is denoted as $p'_{d_{[i]}}$. The degree of validity associated with the prediction is defined as,

X	*A*	*B*	*C*	*D*	*E*	*F*
x_1	$(a_{[1,1,1]},\frac{2}{3})(a_{[2,1,1]},\frac{1}{3})$	$b_{[1,1,1]}$	$c_{[1,1,1]}$	$d_{[1,1]}$	$e_{[1,1,1]}$	$f_{[1,1,1]}$
x_2	$a_{[2,2,1]}$	$b_{[2,1,1]}$	$c_{[1,1,1]}$	$d_{[2,3]}$	$e_{[2,1,1]}$	$f_{[1,3,2]}$
x_3	$a_{[1,1,2]}$	$(b_{[1,1,1]},\frac{1}{2})(b_{[2,1,2]},\frac{1}{2})$	$c_{[1,1,1]}$	$d_{[1,3]}$	$e_{[1,1,1]}$	$f_{[1,3,1]}$
x_4	$a_{[2,2,1]}$	$b_{[2,1,2]}$	$c_{[2,1]}$	$(d_{[1,1]},\frac{2}{3})(d_{[2,1]},\frac{1}{3})$	$e_{[2,1,1]}$	$f_{[1,1,1]}$
x_5	$a_{[1,1,2]}$	$b_{[2,3,2]}$	$c_{[1,2,1]}$	$d_{[1,2]}$	$e_{[2,3,2]}$	$f_{[1,2,2]}$
x_6	$(a_{[1,1,1]},\frac{2}{3})(a_{[2,1,1]},\frac{1}{3})$	$b_{[2,2,1]}$	$c_{[1,3,1]}$	$d_{[1,1]}$	$e_{[2,3,2]}$	$f_{[1,1,2]}$
			.			
			.			
x_i	$a_{[1,2,1]}$	$b_{[2,1,1]}$	$c_{[1,1]}$	$d_{[1,2]}$	$e_{[1,1,1]}$	$f_{[1,1,1]}$

Table 2. Information System S_d

Rule	*A*	*B*	*C*	*D*	*E*	*F*	*Sup*	*Conf*
r_1			$c_{[1,1,1]}$	$(d_{[1,1,2]})$			100	0.9
r_2	$a_{[1,1,1]}$			$(d_{[1,1,2]})$		$f_{[1,1,1]}$	110	1
r_3		$b_{[1,1,1]}$			$(e_{[1,1,1]})$		120	1
r_4	$(a_{[1,1,1]})$				$e_{[1,1,1]}$		100	1
r_5	$(a_{[1,1,1]})$	$b_{[1,1,1]}$				$f_{[1,1,1]}$	90	1
r_6	$a_{[1,1,2]}$	$(b_{[2,3,1]})$					50	0.9
r_7		$(b_{[2,3,2]})$	$c_{[2,3,2]}$				100	1
r_8		$(b_{[2,3,2]})$			$e_{[2,3,2]}$	$f_{[1,2,2]}$	100	0.9
r_9	$a_{[1,1,1]}$		$c_{[1,1]}$	$(d_{[1,1,2]})$			100	0.9
r_{10}	$a_{[1,1,1]}$			$(d_{[1,1,2]})$		$f_{[1,1]}$	100	0.9

Table 3. Rules in KB

$$v = 1 - \frac{p_{d_{[i]}} - p'_{d_{[i]}}}{p_{d_{[i]}}}$$

and, we say d_i is secure against *Chase* if $v < \tau$. For example, assume that $\tau = 0.8$ for a confidential attribute d. A confidential attribute value is $\{(d_{[1]}, \frac{3}{4})), (d_{[2]}, \frac{1}{4})\}$ and it is predicted as $\{(d_{[1]}, \frac{1}{4})), (d_{[3]}, \frac{4}{4})\}$. In this case, $d_{[1]}$ is not considered as a valid prediction because $1 - (0.5/0.75) < 0.8$.

4.2 Phase One : Identification

We start phase one with a set $\beta(x)$ for the object x_1 which construction is supported by 5 rules $\{r_1, r_2, r_3, r_4, r_5\}$ from *KB*, and check the chase closure of each singleton subset $\delta(x)$ of that set. If the chase closure of $\delta(x)$ contains classified attribute value d_1, then $\delta(x)$ does not sustain, it is marked, and it is not considered in later steps. Otherwise, the set remains unmarked. In the second iteration of the algorithm, all two-element subsets of $\beta(x)$ built only from unmarked sets are considered. If the chase closure of any of these sets does not contain d_1, then such a set remains unmarked and it is used in the later steps of the algorithm. Otherwise, the set is getting marked. If either all sets in a currently executed iteration step are marked or we have reached the set $\beta(x)$, then the algorithm stops. Since only subsets of $\beta(x)$ are considered, the number of iterations will be usually not large.

So, in our example the following singleton sets are considered:

$\{a_{[1,1,1]}\}^+ = \{a_{[1,1,1]}\}$, unmarked
$\{b_{[1,1,1]}\}^+ = \{b_{[1,1,1]}, e_{[1,1,1]}, a_{[1,1,1]}\}$, unmarked
$\{c_{[1,1,1]}\}^+ = \{c_{[1,1,1]}, d_{[1,1,2]}\} \supseteq \{d_{[1,1,2]}\}$, $v_{[1,1,2]} = 1$ > 0.8 marked *
$\{e_{[1,1,1]}\}^+ = \{e_{[1,1,1]}, a_{[1,1,1]}\}$, unmarked
$\{f_{[1,1,1]}\}^+ = \{f_{[1,1,1]}\}$, unmarked.

Clearly, $\{c_{[1,1,1]}\}$ has to be concealed. The next step is to build terms of length 2 and determine which of the set can remain.

$\{a_{[1,1,1]}, b_{[1,1,1]}\}^+ = \{a_{[1,1,1]}, b_{[1,1,1]}\}$ unmarked
$\{a_{[1,1,1]}, e_{[1,1,1]}\}^+ = \{a_{[1,1,1]}, e_{[1,1,1]}\}$ unmarked
$\{a_{[1,1,1]}, f_{[1,1,1]}\}^+ = \{a_{[1,1,1]}, f_{[1,1,1]}, d_{[1,1,2]}\} \supseteq \{d_{[1,1,2]}\}$, $v_{[1,1,2]} = 1 > 0.8$ marked *
$\{b_{[1,1,1]}, e_{[1,1,1]}\}^+ = \{b_{[1,1,1]}, e_{[1,1,1]}\}$ unmarked

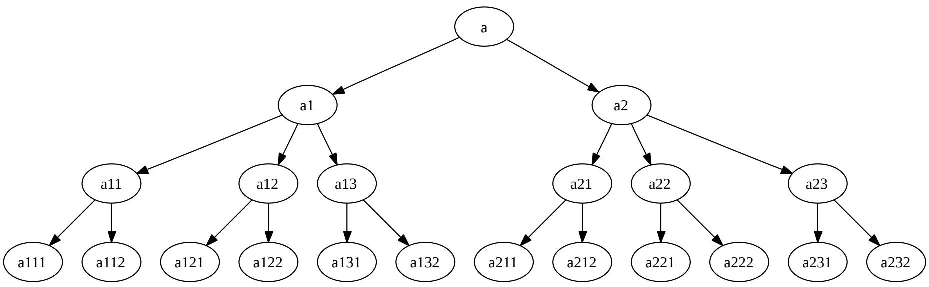


Figure 3. Attribute Hierarchy for $a \in S$

$\{b_{[1,1,1]}, f_{[1,1,1]}\}^+ = \{b_{[1,1,1]}, f_{[1,1,1]}, a_{[1,1,1]}, d_{[1,1,2]}\} \supseteq \{d_{[1,1,2]}\}$, $v_{[1,1,2]} = 1 > 0.8$ marked *
$\{e_{[1,1,1]}, f_{[1,1,1]}\}^+ = \{e_{[1,1,1]}, f_{[1,1,1]}\}$ unmarked

Now we build 3-element sets from previous sets that have not been marked.

$\{a_{[1,1,1]}, b_{[1,1,1]}, e_{[1,1,1]}\}^+ = \{a_{[1,1,1]}, b_{[1,1,1]}, e_{[1,1,1]}\}$ unmarked

We have $\gamma(x_1) = \{a_{[1,1,1]}, b_{[1,1,1]}, e_{[1,1,1]}\}$ as unmarked set that contains the maximum number of elements and does not have the chase closure containing d.

4.3 Phase Two : Generalization

Now, we need to generalize attribute values in $\epsilon(x_1) = \{\alpha(x_1) - \gamma(x_1)\}$. There are two issues. One is that we may have more than one $\gamma(x_1)$. If that is the case, we have to choose one of them. The other issue is that each of $\gamma(x_1)$ contains multiple attribute values that can be generalized. Several strategies can be considered to reduce the amount of generalization. One approach is to compute the minimum amount of generalization for each $\gamma(x_1)_i$ in terms of the number of layer transformations, and compare the result. However, it is difficult to say that the approach will produce the best result because (1) the granularity distance between a parent and a child can be different among $h_a \in H_S$, and (2) some attribute values are semantically more significant than others in QAS. In this case, a priority can be given by the system. In this paper, we assume that the amount of abstraction between a parent and a child is identical across all the attributes, and generalization is applied equally to each attribute value.

From phase one, we acquired the set $\epsilon(x_1) = \{c_{[1,1,1]}, f_{[1,1,1]}\}$. Our strategy in the second phase is that attribute values in $\epsilon(x_1) \subseteq (\epsilon(x_1) \cup \gamma(x_1))$ are generalized against the rules in *KB*, without modifying $\gamma(x_1)$, until the chase closure of the newly created set does not contain $d_{[1,1,2]}$. This strategy works because, as was exhibited in the first phase, if we replace all attribute values in $\epsilon(x_1)$ with null values, $d_{[1,1,1]}$ cannot be predicted. So, between a node connected to the root node (null value) and the current node, $d_{[1,1,1]}$ will not be predicted by *Chase*.

$\{\{c_{[1,1,1]}, f_{[1,1,1]}\}, \{a_{[1,1,1]}, b_{[1,1,1]}, e_{[1,1,1]}\}\}^+$
$= \{a_{[1,1,1]}, b_{[1,1,1]}, e_{[1,1,1]}, c_{[1,1,1]}, f_{[1,1,1]}, d_{[1,1,1]}\}$
$\supseteq \{d_{[1,1,2]}\}$, $v_{[1,1,2]} = 1$ marked*

We start from c by generalizing $c_{[1,1,1]}$ to $c_{[1,1]}$.

$\{\{c_{[1,1]}, f_{[1,1,1]}\}, \{a_{[1,1,1]}, b_{[1,1,1]}, e_{[1,1,1]}\}\}^+$
$= \{a_{[1,1,1]}, b_{[1,1,1]}, e_{[1,1,1]}, c_{[1,1]}, f_{[1,1,1]}, d_{[1,1,1]}, d_{[1,1,2]}\}$
$\supseteq \{d_{[1,1,2]}\}$, $v_{[1,1,2]} = 1$ marked*

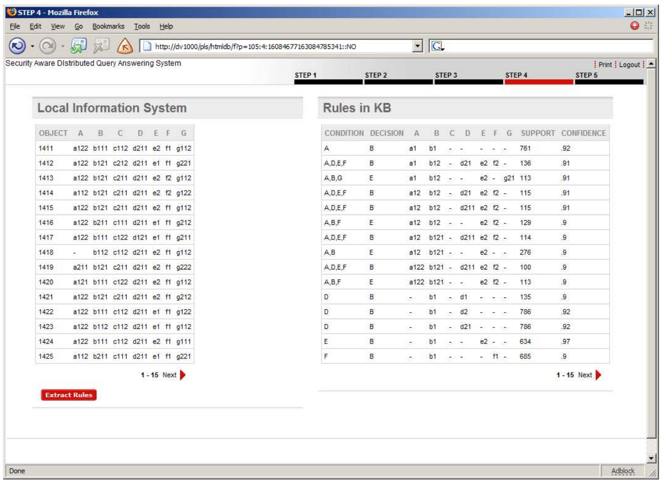
$\{\{c_{[1,1]}, f_{[1,1]}\}, \{a_{[1,1,1]}, b_{[1,1,1]}, e_{[1,1,1]}\}\}^+$
$= \{a_{[1,1,1]}, b_{[1,1,1]}, e_{[1,1,1]}, c_{[1,1]}, f_{[1,1]}, d_{[1,1]}\}$
$\supseteq \{d_{[1,1,2]}\}$, $v_{[1,1,2]} = 1$ marked*

$\{\{c_{[1]}, f_{[1,1]}\}, \{a_{[1,1,1]}, b_{[1,1,1]}, e_{[1,1,1]}\}\}^+$
$= \{a_{[1,1,1]}, b_{[1,1,1]}, e_{[1,1,1]}, c_{[1]}, f_{[1,1]}, d_{[1,1]}\}$ unmarked

We have $\{c_{[1]}, f_{[1,1]}, a_{[1,1,1]}, b_{[1,1,1]}, e_{[1,1,1]}\}$ as a set that cannot be used to predict $d_{[1,1,2]}$. In a similar way, we compute the maximal sets for any object x_i.

5 Implementation and Conclusion

The algorithm was written in *PL/SQL* language, and executed in the Oracle 10g database running on Windows XP. A web based user interface was implemented using *HTML DB* as shown in Figure 4. The sample table that contains 3,000 objects with 7 attributes was randomly extracted from the census bureau database of the UCI Knowledge Discovery in Databases Archive [13]. A set of simple hierarchical attribute structure with a maximal depth of 3 was built on the basis of the interpretation of data. Each level of the hierarchical tree contains one to three nodes. The table was randomly partitioned into 3 tables that each have 1,000 tuples. One of these tables is called a client and the remaining 2 are called servers. We extracted 26 rules that describe the values of a confidential attribute from the servers, and 33


STEP 4 - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://dv1000/pls/htmldb/f?p=105:4:16084677163084785341::NO
Security Aware DIstributed Query Answering System
Print Logout
STEP 1
STEP 2
STEP 3
STEP 4
STEP 5
1 - 15 Next
Extract Rules
1 - 15 Next
Done
Adblock

Local Information System

OBJECT	A	B	C	D	E	F	G
1411	a122	b111	c112	d211	e2	f1	g112
1412	a122	b121	c212	d211	e1	f1	g221
1413	a122	b121	c211	d211	e2	f2	g112
1414	a112	b121	c211	d211	e2	f2	g122
1415	a122	b121	c211	d211	e2	f1	g112
1416	a122	b211	c111	d211	e1	f1	g212
1417	a122	b111	c122	d121	e1	f1	g211
1418	-	b112	c112	d211	e2	f1	g112
1419	a211	b121	c211	d211	e2	f1	g222
1420	a121	b111	c122	d211	e2	f1	g112
1421	a122	b121	c211	d211	e2	f1	g212
1422	a122	b111	c112	d211	e1	f1	g122
1423	a122	b112	c112	d211	e1	f1	g112
1424	a122	b111	c112	d211	e2	f1	g111
1425	a112	b211	c111	d211	e1	f1	g221

Rules in KB

CONDITION	DECISION	A	B	C	D	E	F	G	SUPPORT	CONFIDENCE
A	B	a1	b1	-	-	-	-	-	761	.92
A,D,E,F	B	a1	b12	-	d21	e2	f2	-	136	.91
A,B,G	E	a1	b12	-	-	e2	-	g21	113	.91
A,D,E,F	B	a12	b12	-	d21	e2	f2	-	115	.91
A,D,E,F	B	a12	b12	-	d211	e2	f2	-	115	.91
A,B,F	E	a12	b12	-	-	e2	f2	-	129	.9
A,D,E,F	B	a12	b121	-	d211	e2	f2	-	114	.9
A,B	E	a12	b121	-	-	e2	-	-	276	.9
A,D,E,F	B	a122	b121	-	d211	e2	f2	-	100	.9
A,B,F	E	a122	b121	-	-	e2	f2	-	113	.9
D	B	-	b1	-	d1	-	-	-	135	.9
D	B	-	b1	-	d2	-	-	-	786	.92
D	B	-	b1	-	d21	-	-	-	786	.92
E	B	-	b1	-	-	e2	-	-	634	.97
F	B	-	b1	-	-	-	f1	-	685	.9

Figure 4. User Interface

local rules that are used to describe the values of remaining attributes.

To evaluate whether the use of hierarchical attributes is effective, we compared the total number of slots replaced by null values with that obtained by the same method without using hierarchical attribute structures. This was achieved by running the program without the execution of the generalization step. Without generalization step, 570 slots are replaced with null values. When the attribute hierarchy was applied, 187 slots are replaced by null values with 952 level transformations that include the number of transformations to null values. The result shows that 383 more attribute values can be shown to users. These values consist of values at the same level or values at higher levels. Clearly, the amount of improvements may not be the same when different set of rules and information systems are used. However, the use of hierarchical attributes will reduce the number of null values.

References

[1] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison Wesley, 1995.

[2] A. Dardzińska and Z. Raś. On rules discovery from incomplete information systems. In *Proceedings of the ICDM'03 Workshop on Foundations and New Directions of Data Mining*, Melbourne, Florida, November 2003.

[3] P. Giudici. *Applied Data Mining, Statistical Methods for Business and Industry*. Wiley, 2003.

[4] N. Guarino and P. Giaretta. *Ontologies and knowledge bases, towards a terminological clarification*, pages 25–32. IOS Press, 1995.

[5] S. Im and Z. Ras. Ensuring data security against knowledge discovery in distributed information system. In *Proceedings of the 10th International Conference on Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing*, Regina, Canada, September 2005.

[6] Z. Raś and A. Dardzińska. Collaborative query processing in dks controlled by reducts. In *Proceedings of Rough Sets and Current Trends in Computing 2002 Symposium*, Malvern, PA, October 2002.

[7] Z. Raś and A. Dardzińska. Ontology based distributed autonomous knowledge systems. *Information Systems International Journal*, 29(1):47–58, 2004.

[8] Z. Raś and A. Dardzińska. Query answering based on collaboration and chase. In *Proceedings of the 6th International Conference On Flexible Query Answering Systems*, Lyon, France, June 2004.

[9] Z. Raś and A. Dardzińska. Chase-2: Rule based chase algorithm for information systems of type lambda. In *Proceedings of the Second International Workshop on Active Mining*, Maebashi City, Japan, October 2005.

[10] Z. Raś and A. Dardzińska. *Data security and null value imputation in distributed information systems*, pages 133–146. Springer-Verlag, 2005.

[11] Z. Raś and A. Dardzińska. *Extracting Rules from Incomplete Decision Systems: System ERID*, pages 143–154. Springer, 2005.

[12] Z. Raś, A. Dardzińska, and O. Gurdal. Knowledge discovery based query answering in hierarchical information systems. In *Proceedings of the 10th International Conference on Rough Sets, Fuzzy Sets, Data Mining and Granular Computing*, Regina, Canada, September 2005.

[13] C. B. S. Hettich and C. Merz. UCI repository of machine learning databases. 1998.

[14] G. Van Heijst, A. Schreiber, and B. Wielinga. Using explicit ontologies in kbs development. *International Journal of Human and Computer Studies*, 46(2/3), 1997.