Intelligent Query Answering in DAKS

Zbigniew W. Ras

University of North Carolina
Department of Comp. Science, Charlotte, N.C. 28223, USA
and Polish Academy of Sciences,
Institute of Computer Science, Ordona, Warsaw, Poland
ras@uncc.edu or ras@wars.ipipan.waw.pl

Abstract

The paper concerns discovery of knowledge needed to establish the shared meaning of attributes in a Distributed Autonomous Information Systems (DAIS). We present a Distributed Autonomous Knowledge System (DAKS) which is an extension of DAIS and it uses discovery mechanisms to define attributes missing in one information system by mining other information systems. The applications include knowledge and data sharing at the large scale by intelligent information systems, public verification of knowledge and others. In this paper we mainly focus on intelligent answers to queries.

1 Introduction

Query Answering System (QAS) for Distributed Autonomous Information Systems (DAIS) is concerned with identifying objects satisfying a given description in one of systems in DAIS. For example an information systems S in DAIS might contain information about students in a school and classify them using attributes such as hair color, eye color, gender and size. But S has no data concerning the nationality of a student.

A simple query might be to find in a system S all Swedes who are males with brown hair. One option of handling such a query is to retrieve from S all males who have brown hair. Clearly this way we have no information about the nationality of the retrieved students. Another option of handling such a query is to ask other information systems in DAIS for definitions of the term $Swede$. Assume that a definition discovered in a system S_1 is represented by a certain rule in S1 and it states that:

$(eyes, blue) \wedge (gender, male) \Rightarrow (nationality, Swede)$.

Also, assume that another definition discovered at S_1 is represented by a possible rule and it states that:

$(eyes, brown) \wedge (hair, blond) \Rightarrow (nationality, Swede)$.

Rule is called certain if its confidence is 100 percent and is called possible if its confidence is greater than 0 percent. The above two rules discovered at S_1 can be used to replace the word $Swede$ in a query
$q = [male.Swede.with.brown.hair]$

translating q to its rough representation $[q1; q2] =$
$[male.with.blue.eyes.and.brown.hair;$

$male.with.brown.eyes.and.hair.which.is.blond.and.brown]$.

Assuming that there is no person which satisfies statement $q2$, the original query will be replaced by the query
$[male.with.blue.eyes.and.brown.hair]$.

Clearly this new query can be handled in S without any problem. The pair $[q1, q2]$ is called a rough representation of q if $q1$ is seen as the description of certain objects described by q and $q2$ is seen as the description of possible objects described by q.

There is a number of strategies which allow us to find certain and possible rules describing decision attributes in terms of classification attributes. We should mention here such systems like $LERS$ (developed by J. Grzymala-Busse), $DQuest$ (developed by W. Ziarko), $C4.5$ (developed by J. Quinlan) or $AQ17$ (developed by R. Michalski). System $FortyNiner$ developed by J. Zytkow allows to describe one attribute as a function of other attributes. It is especially useful when attributes are numerical and when rough representation of a query is too general.

In this paper we show how to use definitions (of attribute values) extracted at sites of $DAIS$ and how to use reducts to build better query answering systems in a distributed environment.

For more than 10 years research has been devoted to the question of information retrieval from heterogenous distributed databases. This research has sought to provide integrated access to such databases and has focused on distributed databases, multidatabases, federated databases and their interoperability. The main purpose of integrated access is to enable a number of heterogeneous distributed databases to be queried as if they were a single homogeneous database. Common practice in integrating database systems involves manual integration of each database schema into a global schema [1]. This approach does not work when the number of database systems is large. Navathe and Donahoo [16] propose to allow the database designers to develop a metadata description of their database schema. The collection of metadata descriptions can be automatically processed by a schema builder to create a partially integrated global schema. The heterogeneity problem can be eliminated ([15]) by using an intermediate model that controls the knowledge translation from a source database or knowledgebase. The intermediate model they developed is based on the concept of abstract knowledge representation. It has two components: a modeling behavior which separates the knowledge from its implementation, and a performative behavior which establishes context abstraction rules over the knowledge.

Data mining provides tools for summarizing database contents and transforming low-level heterogeneous data into high-level homogeneous knowledge about the domain described by data. To deal with semantic heterogeneity in multiple autonomous databases, J. Han [10], [11] suggests to use generalization technique to transform low level diverse data into relatively high level, commonly sharable information. The construction of a Multiple Layered Database (MLD) composed of several layers of information is based on this methodology [11]. The lowest layer of MLD corresponds to the primitive information stored in a conventional database. Its higher layers store more general information extracted, by data mining techniques, from the lower layers. Many techniques proposed in the area of cooperative query answering ([3], [4], [7], [5]) in a single-layer databases can be naturally extended to cooperative query answering in MLD.

To address the problem of semantic heterogeneity in multiple autonomous databases we introduce a Distributed Autonomous Knowledge Systems ($DAKS$) where each knowledge system consists of a database (information system), a knowledgebase and, an agent. We assume that:

- Databases in $DAKS$ consist of data catalogs, data tables, and metadata (additional information about data).
- Knowledgebases in $DAKS$ consist not only of knowledge derived from data (rules, classification trees, equations or, taxonomies) but also they contain knowledge of other domains, knowledge of other sites and, communication knowledge. The knowledge derived from the data is stored either in a discovery layer or a knowledge layer. Discovery layer for a given site contains knowledge extracted only from that site. Knowledge layer for each site contains knowledge extracted at other sites of $DAKS$. Initially, the knowledge layers in $DAKS$ are empty.
- Agents in $DAKS$ interact with one another and use meaning unification and knowledge discovery to induce from accessible data sources a new knowledge needed to answer queries. Each agent is represented by an Intelligent Query Answering System ($IQAS$), knowledge discovery algorithms (Forty Niner, Rough Sets Library, AQ15, C4.5) and communication protocols to coordinate knowledge.

Figure 1 shows the proposed initial structure of a Distributed Autonomous Knowledge System. Discovery layers linked with databases at two different sites of $DAKS$ can easily be inconsistent because the knowledge they contain is extracted from two different sources. So, we added a shared (higher) discovery layer which contains knowledge extracted and repaired (if inconsistent) from the discovery layers of those sites which are frequently exchanging their knowledge. The shared discovery layers divide $DAKS$ into separate clusters. The process of answering queries will be greatly simplified because the knowledge exchange between sites of the same cluster is not needed.

We consider two types of queries called local (use only local attributes) and global (use some attributes which are locally not available). Global queries are

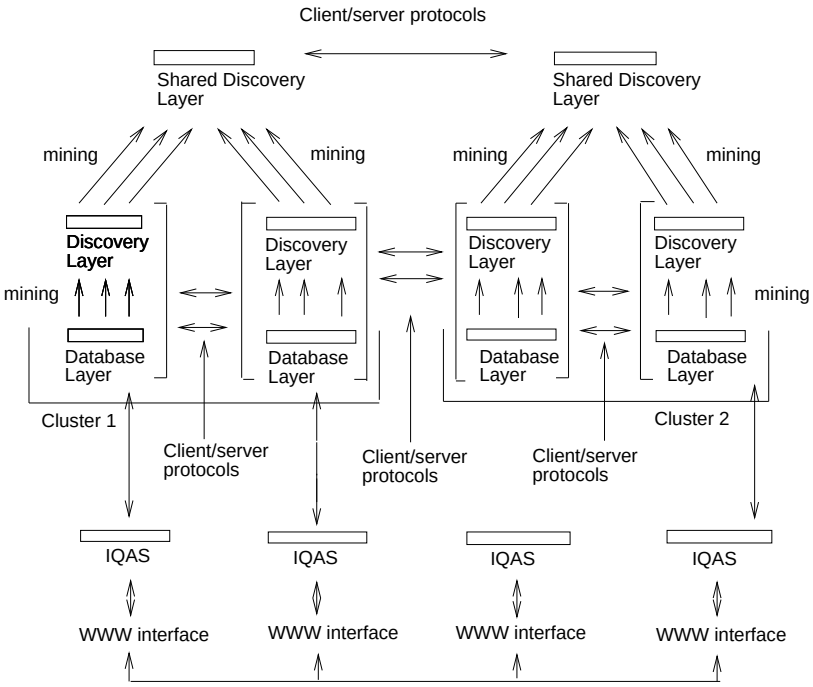

Fig. 1. Distributed Autonomous Knowledge System

queries which can be resolved only through the interaction of agents (exchanging knowledge in a form of definitions of attributes) either of the same or of different clusters in $DAKS$. Local queries are resolved entirely by a query answering system of a local agent who will access, if needed, the local discovery layer and/or the shared discovery layer stored in a locally shared memory of the agent's cluster. For example, a query

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

will be called global for a database

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

if the attribute $aircraft$ is not defined in both local and locally shared discovery layers.

A shared discovery layer of any cluster in $DAKS$ can also be seen as a shared knowledge layer (since its knowledge is extracted from many sites of a single cluster).

2 Basic Definitions

By an information system we mean a structure $S = (X, A, V)$, 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 often write $a_{[S]}$ to denote that a in an attribute in S.

We assume here that the reader is familiar with the basic definitions in rough sets theory including the notion of a covering. The main idea of the use of rough set theory for discovering rules is presented in Figure 2.

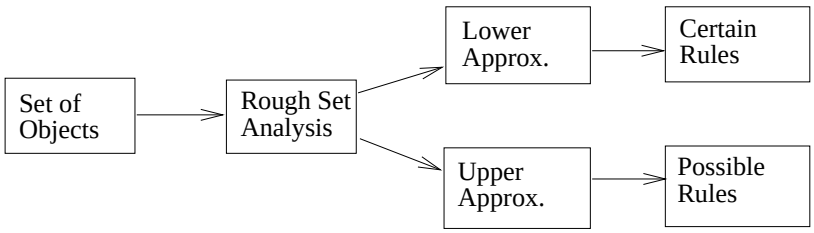

Fig. 2. Rough Set Rule Generation

By a distributed information system [20], [21], [23] we mean a pair $DS = (\{S_i\}_{i \in I}, L)$ where:

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

Distributed information system $DS = (\{S_i\}_{i \in I}, L)$ is consistent if:

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

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

We will use A to name the set of all attributes in DS, $A = \bigcup\{A_j : j \in I\}$.

The shared semantics is defined for the set A of all attributes in all information systems in DS. For each attribute a in A, the operational meaning of a is defined by:

1. the set of information systems in which a is available directly: $S_i : a \in A_i$;

2. the set of information systems in which a has been defined; and the set of definitions in each information system. Definitions can be equations, boolean forms, etc.
3. the set of information systems in which the definitions of a can be used, because the defining attributes are available there. An attribute a is a defined attribute in an information system S if:
   (a) a definition DEF of a has been discovered in one of information systems in DS;
   (b) all other attributes in the definition DEF are present in S; in such cases they can be put together in a JOIN table and DEF can be directly applied.

3 Syntax of Definitions

Now, we define the syntax of definitions in the form of equations. Partial definitions are included, as they are often useful. They can be automatically discovered by system 49er (developed by J. Zytkow). In the next subsection we give the interpretation of partial definitions.

Functors are the building blocks from which equations and inequalities can be formed. Those in turn are the building blocks for partial definitions. Assume that x is a variable over X_i and $r_1, r_2, ..., r_k$ are functors. Also, we assume here that m_j is the number of arguments of the functor r_j, $j = 1, 2, .., k$. The number of arguments can be zero. A zero argument functor is treated as a constant.

By a set of *s(i)-atomic-terms* we mean a least set $T0_i$ such that:

- $\mathbf{0}, \mathbf{1} \in T0_i$,

for any symbolic attribute $a \in A_j$,

- $[a(x) = w] \in T0_i$ for any $a \in A_i$ and $w \in V_{ia}$,
- $\sim [a(x) = w] \in T0_i$ for any $a \in A_i$ and $w \in V_{ia}$,

for any numerical attributes $a, a_1, a_2, ..., a_{m_j}$ in A_i,

- $[a(x)\ \rho\ r_j(a_1, a_2, ..., a_{m_j})(x)] \in T0_i$, where $\rho \in \{=, \leq, \geq\}$.

s(i)-atomic-terms of the form $[a(x) = w]$ and $[a(x) = r_j(a_1, a_2, ..., a_{m_j})(x)]$ are called equations.

By a set of *s(i)-partial-definitions* (s(i)-p-defs in short) we mean a least set T_i such that:

- if $t(x) \in T0_i$ is an equation, then $t(x) \in T_i$,
- if $a \in A_i$ and $t(x)$ is a conjunction of *s(i)-atomic-terms* and $s(x)$ is an equation, then $[t(x) \longrightarrow s(x)] \in T_i$,
- if $t_1(x), t_2(x) \in T_i$, then $(t_1(x) \vee t_2(x)), (t_1(x) \wedge t_2(x)) \in T_i$.

For simplicity we often write t instead of $t(x)$.

The set of *s(I)-defs* is defined in a similar way to *s(i)-p-defs*: the set V_i is only replaced by $\bigcup\{V_j : j \in I\}$ and the set A_i is replaced by $\bigcup\{A_j : j \in I\}$. *s(I)-defs* represents all possible candidate definitions built from attributes that can come from different databases (information systems).

Standard interpretation M_i of *s(i)-p-defs* in a distributed information system $DS = (\{S_j\}_{j \in I}, L)$ is defined as follows:

- $M_i(\mathbf{0}) = \emptyset$, $M_i(\mathbf{1}) = X_i$
- $M_i(a(x) = w) = \{x \in X_i : a_{[S_i]}(x) = w\}$,
- $M_i(\sim (a(x) = w)) = \{x \in X_i : a_{[S_i]}(x) \neq w\}$,
- for any $\rho \in \{=, \leq, \geq\}$,
  $M_i(a(x)\ \rho\ r_j(a_1, a_2, ..., a_{m_j})(x)) =$
  $\{x \in X_i : a_{[S_i]}(x)\ \rho\ r_j(a_{1[S_i]}(x), a_{2[S_i]}(x), ..., a_{m_j[S_i]}(x))\}$,
- $M_i([t \longrightarrow s]) = \{x \in X_i :$ if $[x \in M_i(t)]$ then $[x \in M_i(s)]\}$,
- if t_1, t_2 are *s(i)-p-defs*, then
  $M_i(t_1 \vee t_2) = M_i(t_1) \cup M_i(t_2)$,
  $M_i(t_1 \wedge t_2) = M_i(t_1) \cap M_i(t_2)$,
  $M_i(t_1 = t_2) =$ (if $M_i(t_1) = M_i(t_2)$ then $True$ else $False$).

Let us assume that $[t1 \longrightarrow (a_1(x) = w1)], [t2 \longrightarrow (a_2(x) = w2)]$ are *s(i)-p-defs*. We say that they are S_i-consistent, if either $a_1 \neq a_2$ or $M_i(t_1 \wedge t_2) = \emptyset$ or $w1 = w2$. Otherwise, these two *s(i)-p-defs* are called S_i-inconsistent.

Similar definitions apply when $w1$ and $w2$ in those partial definitions are replaced by $r_1(a_1, a_2, ..., a_{m_j})(x)$ and $r_2(a_1, a_2, ..., a_{m_j})(x)$.

4 Discovery layer

In this section, we introduce the notions of a discovery layer and a distributed autonomous knowledge system. Also, we introduce the concept of a dynamic operational semantics to reflect the dynamics of constantly changing discovery layers.

Notice that while in the previous sections $s(i)-p-defs$ have been interpreted at the sites at which all relevant attributes have been present, we now consider $s(I) - defs$ imported from site k to site i.

By a discovery layer D_{ki} we mean any $s(i)$-consistent set of $s(k)-p-defs$, of the two types specified below, which are satisfied, by means of the interpretation M_k, by most of the objects in S_k:

- $[t \longrightarrow [(a = r_m(a_1, a_2, ..., a_m))(x)]]$, where $a_1, a_2, ..., a_m \in A_i$ and $a \in A_k$ and t is a conjunction of atomic terms that contain attributes that occur both in A_i and in A_k
- $[t \longrightarrow (a(x) = w)]$, where $a \in A_k$ and t satisfies the same conditions as above.

Suppose that a number of partial definitions have been imported to site i from a set of sites K_i. All those definitions can be used at site i.

Thus, the discovery layer for site $i \in I$ is defined as a subset of the set $D_i = \bigcup\{D_{ki} : k \in K_i\}$ where K_i is a set of sites.

By Distributed Autonomous Knowledge System ($DAKS$) we mean $DS = (\{(S_i, D_i)\}_{i\in I}, L)$ where $(\{S_i\}_{i\in I}, L)$ is a distributed information system and D_i is a discovery layer for a site $i \in I$.

Let M_i be a standard interpretation of $s(i)$-p-defs in $DS = (\{S_j\}_{j\in I}, L)$ and $C_i = \bigcup\{V_k : k \in I\} - V_i$. By i-operational semantics of $s(I)$-defs in $DS = (\{(S_i, D_i)\}_{i\in I}, L)$ where $S_i = (X_i, A_i, V_i)$ and $V_i = \bigcup\{V_{ia} : a \in A_i\}$, we mean the interpretation N_i such that:

- $N_i(\mathbf{0}) = \emptyset$, $N_i(\mathbf{1}) = X_i$
- for any $w \in V_{ia}$,
  $N_i(a(x) = w) = M_i(a(x) = w)$,
  $N_i(\sim (a(x) = w)) = M_i(\sim (a(x) = w))$
- for any $w \in C_i \cap V_{ka}$ where $k \neq i$,
  $N_i(a(x) = w) = \{x \in X_i : ([t \longrightarrow [a(x) = w]] \in D_i \wedge x \in M_i(t))\}$
  $N_i(\sim (a(x) = w)) = \{x \in X_i : (\exists v \in V_a)[(v \neq w) \wedge ([t \longrightarrow [a(x) = v]] \in D_i) \wedge (x \in M_i(t))]\}$

- for any $w \in C_i \cap V_{ka}$ where $k \neq i$ and a is a numeric attribute,
  $N_i((a(x) = w)) = \bigcup\{x \in X_i : (\exists y \in X_k)$
  $([a_{1[S_i]}(x) = a_{1[S_k]}(y)] \wedge [a_{2[S_i]}(x) = a_{2[S_k]}(y)] \wedge ... \wedge [a_{m[S_i]}(x) = a_{m[S_k]}(y)] \wedge$
  $[a(y) = w = r_m(a_1, a_2, ..., a_m)] \in D_i)\}$
  $N_i(\sim (a(x) = w)) = X_i - N_i(a(x) = w)$

- for any $s(I)$-terms t_1, t_2
  $N_i(t_1 + t_2) = N_i(t_1) \cup N_i(t_2)$,
  $N_i(t_1 * t_2) = N_i(t_1) \cap N_i(t_2)$,
  $N_i(\sim (t_1 + t_2)) = N_i(\sim t_1) \cap N_i(\sim t_2)$,
  $N_i(\sim (t_1 * t_2)) = N_i(\sim t_1) \cup N_i(\sim t_2)$,
  $N_i(\sim\sim t) = N_i(t)$.
- for any $s(I)$-terms t_1, t_2
  $N_i(t_1 = t_2) = (\text{ if } N_i(t_1) = N_i(t_2) \text{ then } True \text{ else } False)$

Rules and equations in a Knowledge Layer are collected from many sites of $DAKS$. Next, any inconsistencies among rules and equations are resolved so the resulting knowledge has more chance to be globally true.

Formal logic has been chosen to represent knowledge in $DAKS$ and to give foundations of handling $s(I) - defs$. Many other representations (not necessarily based on mathematical logic) are, of course, possible. We have chosen formal logic because of the need to manipulate $s(I) - defs$ syntactically without changing their semantical meaning. This syntactical manipulation of $s(I) - defs$ will be

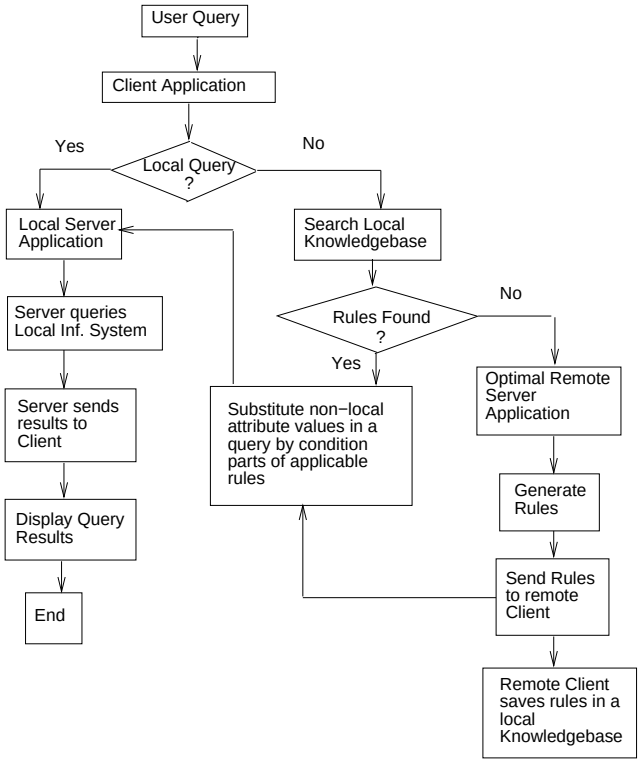

Fig. 3. Working Flow of $DAKS$

described in a separate paper. We need an assurance that the transformation process for $s(I) - defs$ based on logical axioms either will not change their semantical meaning or will change it in a controlled way (it will produce $s(i) - partial-definitions$ approximating initial $s(I)-defs$). Clearly, such a property is very much needed. Without it, we may be looking for an answer to queries which are semantically entirely different from the queries asked by the user. Such a situation has to be avoided.

5 Distributed Autonomous Knowledge System

The conventional distributed database system can be implemented using a three-tiered structure: Client, Server and a database. The $DAKS$ adds one more tier between server and a database (information system), so that the structure becomes a four-tiered one. This tier is called a knowledgebase and it contains a discovery layer and a knowledge layer. Both layers are used to store rules acquired either locally or from remote hosts.

The working flow of $DAKS$ is described using a flow diagram (see Figure 3). It demonstrates a way to implement $DAKS$. When user sends a query to the local client application, the client decides whether the query is local or not, i.e. whether the query contains locally unknown attributes. If the answer is negative, then the query is sent to the local server application. The question raises when client decides that the query is a non-local one. In this case the client sends two lists to an optimal remote server: a singleton list containing the unknown attribute and the list of its locally known attributes. Then the remote server calls RSL (Rough Sets Library) to generate rules from its local database describing the remote-client-unknown attribute in terms of the remote-client-known attribute list. These rules are sent back to the remote client. The client saves the rules in the local knowledge layer for a future use. Also the client applies these rules to approximate the non-local query by a local query. This approximation can have a form of a rough query if either RSL, LERS, or Rosetta is used for knowledge discovery (both certain and possible rules have to be extracted by remote server to construct a rough query).

Now, we briefly discuss the strategy of chosing an optimal site k for a discovery of definitions of locally unknown attribute values. Let us assume that for a site k there are sites I_k which meet the conditions needed for mining unknown attribute values. In [20], [21] any site from $\{i \in I_k : card(A_k \cap A_i) = max\}$ can be chosen as a favorite one. Clearly the reducts and coverings for each site $i \in I_k$ should also be taken into consideration.

Assume that $c \in A_i$ and $O_p(k,i,c)$ denotes the set of coverings of c at site i where each of the coverings is a superset of some reduct at site k. We claim that any site from $\{i \in I_k : (\exists R_i)[R_i \subset A_k \cap A_i \& R_i \in O_p(k,i,c)\}$ is optimal for discovering a definition of the attribute c.

If $O_p(k,i,c)$ is empty then some heuristic function weakening assumptions used to build this set should be used helping to choose an optimal site for discovering the definition of c.

References

1. Batini, C., Lenzerini, M., Navathe, S., "A comparative analysis of methodologies for database schema integration", in *ACM Computing Surveys*, Vol 18, No. 4, 1986, 325-364
2. Bosc, P., Pivert, O., "Some approaches for relational databases flexible querying", in *Journal of Intelligent Information Systems*, Kluwer Academic Publishers, Vol. 1, 1992, 355-382
3. Chu, W.W., "Neighborhood and associative query answering", in *Journal of Intelligent Information Systems*, Kluwer Academic Publishers, Vol. 1, 1992, 355-382
4. Chu, W.W., Chen, Q., Lee, R., "Cooperative query answering via type abstraction hierarchy", in *Cooperating Knowledge-based Systems* (ed. S.M. Deen), North Holland, 1991, 271-292

5. Cuppers, F., Demolombe, R., "Cooperative answering: a methodology to provide intelligent access to databases", in *Proceedings 2nd International Conference on Expert Database Systems*, Virginia, USA, 1988
6. Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R., "Advances in Knowledge Discovery and Data Mining", AAAI Press/MIT Press, 1996
7. Gaasterland, T., Godfrey, P., Minker, J., "An overview of cooperative answering", *Journal of Intelligent Information Systems*, Kluwer Academic Publishers, Vol. 1, 1992, 123-158
8. Grzymala-Busse, J., "Managing uncertainty in expert systems", Kluwer Academic Publishers, 1991
9. Han, J., Cai, Y., Cercone, N., "Data-driven discovery of quantitative rules in relational databases", in *IEEE Trans. Knowledge and Data Engineering*, No. 5, 1993, 29-40
10. Han, J., Huang, Y., Cercone, N., Fu, Y. "Intelligent query answering by knowledge discovery techniques", in *IEEE Trans. Knowledge and Data Engineering*, No. 8, 1996, 373-390
11. Han, J., Fu, Y., Ng, R., Dao, S., "Dealing with semantic heterogeneity by generalization-based data mining techniques", in *Cooperative Information Systems: Current Trends and Directions*, Academic Press, 1998, 207-231
12. Kryszkiewicz, M., Ras, Z.W., "Query rough-answering system for CKBS", in *Proceedings of the Fourth International Workshop on Rough Sets, Fuzzy Sets and Machine Discovery*, RSFD-96, Tokyo, Japan, November 6-8, 1996, pp. 162-167
13. Lin, T.Y., Cercone, N. (eds.), "Rough Sets and Data Mining : Analysis of Imprecise Data", Kluwer, 1997
14. Maitan, J., Ras, Z.W., Zemankova, M., "Query handling and learning in a distributed intelligent system", in *Methodologies for Intelligent Systems*, 4, (ed. Z.W. Ras), North Holland, 1989, 118-127
15. 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
16. 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
17. Pawlak, Z., "Rough sets and decision tables", in *Proceedings of the Fifth Symposium on Computation Theory*, Springer Verlag, Lecture Notes in Computer Science, Vol. 208, 1985, 118-127
18. Pawlak, Z., "Mathematical foundations of information retrieval", *CC PAS Reports*, No. 101, Warsaw, 1973
19. Ras, Z.W., "Dictionaries in a distributed knowledge-based system", in *Proceedings of Concurrent Engineering: Research and Applications Conference*, Pittsburgh, August 29-31, 1994, Concurrent Technologies Corporation, 383-390
20. Ras, Z.W., "Collaboration control in distributed knowledge-based systems", in *Information Sciences Journal*, Elsevier, Vol. 96, No. 3/4, 1997, pp. 193-205
21. Ras, Z.W., "Cooperative knowledge-based systems", in *Intelligent Automation and Soft Computing Journal*, Vol. 2, No. 2, 1996, pp. 193-202
22. Ras, Z.W., "Resolving queries through cooperation in multi-agent systems", in *Rough Sets and Data Mining*, Eds. T.Y. Lin, N. Cercone, Kluwer Academic Publishers, 1997, pp. 239-258
23. Ras, Z.W., Zytkow, J.M., "Discovery of equations and the shared operational semantics in distributed autonomous databases", in *Proceedings of PAKDD'99*,

Springer Verlag, Lecture Notes in Computer Science, Vol. 1574, 1999, 453-463
24. Sheth, A.P., Larson, J.A., "Federated database systems for managing distributed, heterogeneous, and autonomous databases", in *ACM Comput. Surv.*, No. 22, 1990, 183-236
25. Walker, M.G., Wiederhold, G., "Acquisition and Validation of Knowledge from Data", in *Intelligent Systems: State of the Art and Future Directions*, (Eds. Z. Ras and M. Zemankova), Ellis Horwood, New York, 1990, 415-430
26. Ziarko W. (ed.), "Rough Sets, Fuzzy Sets and Knowledge Discovery", Springer-Verlag, 1994

This article was processed using the LaTeX macro package with LMAMULT style