A Rough Set based Data Inconsistency Checking Method for Relational Databases

Hyontai Sug,

Division of Computer & Info. Eng., Dongseo University, Busan, Korea

Summary
In order to deal with data inconsistency problems in relational databases, a new method based on rough set theory which checks data consistency solely based on data is presented. The inconsistent data that exist under the attribute sets in the relations having possible functional dependencies can be found effectively by applying the suggested rough set based consistency checking method. The method is illustrated by examples.

Key words:
Rough sets, data integrity, functional dependency.

1. Introduction

Nowadays, computers as well as related software systems like database management systems are available widely, so that lots of databases are created and used. Moreover, it is relatively easy to use the database management systems for small and medium sized computers because of the development of good user interfaces. But, this ease of use might play a role of some bad points in the respect of data integrity.

Because relational databases are consisted of relations that resemble conventional tables, users or designers of the databases might consider that the relations are just like conventional tables and they want to store data in a small number of tables as much as possible, because the complexity of making queries from the tables is increased as the number of tables is increased. Therefore, it is highly possible that the relations are not normalized well, so that the relations might contain some redundant information due to the small number of tables.

For example, consider that we have a book lending shop where the shop can lend some books to customers. The shop's database has a table called LENDING to store the book lending information. The table has an attribute set like {customerNumber, customerName, telephoneNumber, address, bookRegistrationNumber, lentDate, returnDate}. The underlines represent the attributes' role as a primary key, and there is a functional dependency, {customerName, telephoneNumber} $\rightarrow$ {address}. The following table contains some example data that contain redundant data.

Table 1: An example table, LENDING

C#	CNm	Tel.	Addr.	Book#	lntDt	rtnDt
C1	John	555..	25 m..	B089	…	…
C2	Tom	555..	11 s..	B010	…	…
C3	Mary	555..	13 o..	B400	…	…
C4	Judy	555..	44 h..	B101	…	…
C3	Mary	555..	13 o..	B653	…	…
C2	Tom	555..	11 s..	B356	…	…

In the table C#, CNm, Tel., Addr., Book#, lntDt, rtnDt represent customerNumber, customerName, telephoneNumber, address, bookRegistrationNumber, lentDate, returnDate respectively.

The designer of the database might not want to have a separate table to store the customer's information, because reports to be printed out need most of the data in the table LENDING. Separate table structures like LNT{bookRegistrationNumber, lentDate, returnDate}, and CUST{customerNumber, customerName, telephone_number, address} make printing out the reports slightly more troublesome, because we need an additional join operation.

Table 2: An example table, LNT

Book#	lntDt	rtnDt
B089	07.05.05	07.05.25
B010	08.10.03	
B400	06.04.25	06.05.23
B101	07.11.01	07.12.20
B653	08.09.15	
B356	08.07.14	08.07.28

Table 3: An example table, CUST

C#	CNm	Tel.	Addr.
C1	John	555..	25 m..
C2	Tom	555..	11 s..
C3	Mary	555..	13 o..
C4	Judy	555..	44 h..
C3	Mary	555..	13 o..
C2	Tom	555..	11 s..

If some customers borrow books more than once, redundant data for attributes {customerName,

Manuscript received November 5, 2008
Manuscript revised November 20, 2008

telephoneNumber, address} can reside in the table many times. This redundancy may cause data inconsistency problem, if the redundant data have not been updated unanimously. The inconsistency problem might happen whenever some customers moved somewhere else. The problem can become worse when the structure of the database is more complex, which is common in real world situations.
This paper suggests a method to solve the problem of data inconsistency based on an approach inspired by rough set theory. The method is applied to the attribute sets in a relation that have possible functional dependencies between attributes. We will first discuss related works in section 2, in section 3 we present our method in detail and in section 4 we illustrate our method through examples. Finally in section 5, we present conclusions. This paper is a modified version of a conference paper [1].

2. Related Work

Because rough set theory considers data dependency solely based on data, many researchers tried to investigate attribute dependency in algebraic aspects [2], or in statistical aspects [3]. There are also many researchers who tried to find decision rules from databases [4, 5]. ROSETTA [6] and RSES [7] are some examples of data mining tools for such efforts. There is some size limitation of input data set for the systems due to time complexity. Because rough set theory concerns concepts that exit in a table, some researchers tried to combine rough set theory with other well-known theories. Ytow et al. [8] combined formal concepts having objects and attributes with rough sets to have upper and lower approximations, and Guo and Tanaka [9] showed similarity between possibility theory and rough set theory. In paper like [10] we can find a survey on feature subset selection based on rough set theory to optimize knowledge models for given data sets.

3. Proposed Method

3.1 Definitions

The main advantage of rough set theory is that concept approximation is solely based on data, so it does not need any preliminary or additional information about the data.

Definition 1: If we are given a finite set $U \neq \varnothing$ of objects, called a universe, and R is an equivalence relation over U, then U/R represents the family of all equivalence classes of R called categories, and $[x]_R$ indicates a category of $x \in U$ in R.

The following definition extends the equivalence relation R in definition 1 to the family of equivalence relations over U.

Definition 2: If we are given a finite set $U \neq \varnothing$ of objects, called a universe, and a family of equivalence relations over U, called **R**, then a relational system K = (U, **R**) is a knowledge base. A family of equivalence relations **R** represents the set of equivalence relations having the following properties.
· If **R** is a family of equivalence relations over U then U/**R** means the family of all equivalence classes of **R**.
· IND(**P**) means the intersection of all equivalence relations belonging to **P** and is called an indiscernibility relation over **P** where $\mathbf{P} \subseteq \mathbf{R}$ and $\mathbf{P} \neq \varnothing$.
· U/IND(**P**) means the family of all equivalence classes of the IND(**P**).
· [x]**p** indicates a category of $x \in U$ in **P**.

Definition 3: Rough sets are sets that are defined using two approximations, upper approximations and lower approximations. Let $X \subseteq U$ and $R \in IND(\mathbf{R})$ then
· R-lower approximation, $R_{lower}X = \{ Y \in U/\mathbf{R}: Y \subseteq X \}$
· R-upper approximation, $R^{upper}X = \{ Y \in U/\mathbf{R}: Y \cap X \neq \varnothing \}$
· R-boundary region of X, $BN_R(X) = R^{upper}X - R_{lower}X$
Additionally, we can define the following terms:
· R-positive region of X, $POS_R(X) = R_{lower}X$
Let P and Q be equivalence relations over U, then
· P-positive region of Q is $POS_P(Q) = \cup_{x \in U/Q} P_{lower}x$.

Let $IND(\mathbf{P}) \subseteq IND(\mathbf{R})$, then the same definitions are applied for IND(**P**) as R above. So, we have the following definition 4 for the degree of dependency between **P** and $\mathbf{Q} \subseteq \mathbf{R}$.

Definition 4: The degree of dependency between **P** and **Q** where **P**, $\mathbf{Q} \subset \mathbf{R}$ is defined as follows.
$\mathbf{P} \Rightarrow_k \mathbf{Q}$ where $k = |POS_{\mathbf{P}}(\mathbf{Q})| / |U|$.
· k = 1 : **Q** totally depends on **P**.
· 0<k<1 : **Q** partially depends on **P**.
· k = 0 : **Q** is independent on **P**.

So, if we have a larger positive region, we can see more dependency between **P** and **Q**. So, we apply the definition 4 to check dependency between sets of attributes in relations. In any functional dependencies of a relational table **P** corresponds to the left hand side of the functional dependency and **Q** corresponds to the right hand side of the functional dependency, and moreover, k = 1.

3.2 Suggested Method

The functional dependencies we want to use in checking data inconsistency have the property that each right hand side (RHS) of the functional dependencies consists of one attribute. But there is no restriction on the left hand side (LHS) of the functional dependencies. If some functional dependencies have several attributes in their right hand side, we need to separate the attributes of the RHS one by one for efficiency. But, this separation doesn't matter, because we can always decompose the right hand side of a functional dependency, and the two functional dependencies made from the separation is equivalent to the original one by Armstrong's axiom [11]. For example, the functional dependency $A \rightarrow \{B, C\}$ is equivalent to functional dependencies $A \rightarrow B$, and $A \rightarrow C$. In order to find inconsistent data in a given relation we apply the following steps for each user-selected functional dependency in the relation.

For each user-selected functional dependency in the relation **do**

(i) Select a functional dependency (FD) for data inconsistency check.
(ii) Project the relation with respect to the attribute set in the FD.
(iii) Find inconsistent objects where the attribute values of the subsets of RHS are different, even though attribute values of LHS are the same.
(iv) Calculate the degree of dependency, k.
(v) Display k and the sets of inconsistent objects.

End do

Note that in (v) users consider the value of degree of dependency and the set of objects' multiple values, and users can determine whether the attribute set has a real functional dependency or not. And more importantly, users can determine whether data are in inconsistency or not.

4. Examples

Let's see a relation in table 4 having two functional dependencies, $\{A, B\} \rightarrow C$, $\{A, B\} \rightarrow D$.

Table 4: An example table

Object Number	A	B	C	D
1	0	0	1	1
2	0	1	2	1
3	0	2	2	1
4	1	0	1	2
5	1	1	1	2
6	1	1	2	2

If we represent the table in functional dependency form with values like $A_iB_j \Rightarrow C_k$, $A_iB_j \Rightarrow D_k$ where A, B, C, and D represent attribute names and i, j, and k represent respective values, then we have the following two tables, table 5 and table 6.

Table 5: FD $\{A, B\} \rightarrow C$ with attribute values

Object number	FD with values
1	$A_0B_0 \Rightarrow C_1$
2	$A_0B_1 \Rightarrow C_2$
3	$A_0B_2 \Rightarrow C_2$
4	$A_1B_0 \Rightarrow C_1$
5, 6	$A_1B_1 \Rightarrow \{C_1, C_2\}$

Table 6: FD $\{A, B\} \rightarrow D$ with attribute values

Object number	FD with values
1	$A_0B_0 \Rightarrow D_1$
2	$A_0B_1 \Rightarrow D_1$
3	$A_0B_2 \Rightarrow D_1$
4	$A_1B_0 \Rightarrow D_2$
5	$A_1B_1 \Rightarrow D_2$
6	$A_1B_1 \Rightarrow D_2$

Therefore, we can find inconsistent data sets for the functional dependency, $\{A, B\} \rightarrow C$, and the degree of dependency for the functional dependency is 0.67. On the other hand, there is no inconsistency in the values of the functional dependency, $\{A, B\} \rightarrow D$, so that the degree of dependency is 1.

Note that because we have no inconsistency in data according to the table, we may also find an additional functional dependency between attribute A and D as indicated by table 7.

Table 7: A possible FD $A \rightarrow D$ with attribute values

Object number	FD with values
1, 2, 3	$A_0 \Rightarrow D_1$
4, 5, 6	$A_1 \Rightarrow D_2$

But we have to consider the meaning of attribute A and D precisely to decide whether there is functional dependency between A and D or not, because in the future we may have some inconsistency, if there is no functional dependency between A and D.
By considering these values of degree of dependencies and the objects in boundary region, users can determine inconsistent data effectively.

5. Conclusions

As the development of computer technology as well as computer industry, it is not difficult to use the database management systems because of the advancement of user interface technologies, so that nowadays lots of small and medium databases are created and used. But, this ease of use might play a role of some bad points in the respect of data integrity. Because relational databases have relations that resemble conventional relations and the complexity in making queries from the relations is increased as the number of tables is increased, the designers of the databases might want to use smaller number of tables. But, due to the small number of tables it is highly possible that the relations might contain some redundant data, and the redundant data might cause some inconsistency as a result of negligent updates to the redundant information.
This paper suggests an effective method to find such inconsistent data based on possible functional dependencies between attributes in a relation. Rough set theory based method can be applied effectively to find out the inconsistent data. The method measures the size of positive region to reflect the dependency between the left hand side and the right hand side of the functional dependency. In addition, the method also can find some hidden functional dependencies so that it is also useful for data integrity for the future.

References

[1] H. Sug, "Applying rough sets to maintain data consistency for high degree relations", NCM'2008, Vol. 22, 2008, pp. 244-247.
[2] I. Düntsch, and G. Gediga, "Algebraic aspects of attribute dependencies in information systems", Fndamenta Informaticae, Vol. 29, 1997, pp. 119-133.
[3] I. Düntsch, and G. Gediga, "Statistical evaluation of rough set dependency analysis," International journal of human-computer studies, Vol.46, 1997.
[4] T.Y. Lin, and H. Cao, "Searching decision rules in very large databases using rough set theory," Lecture notes in artificial intelligence, Ziarco and Yao eds., 2000, pp. 346-353.
[5] R. Stowinski, ed., Intelligent decision support: Handbook of applications and advances of the rough set theory, Kluwer Academic Publishers, 1992.
[6] A. Øhrn, Discernibility and rough sets in medicine: tools and applications, PhD thesis, Department of computer and information science, Norwegian University of Science and Technology, 1999.
[7] J.G. Bazan, M.S. Szczuka, and J. Wroblewski, "A new version of rough set exploration system," Lecture notes in artificial intelligence, Vol.2475, 2002, pp. 397-404.
[8] N. Ttow, D.R. Morse, and D.M. Roberts, "Rough set approximation as formal concept," Journal of advanced computational intelligence and intelligent informatics, Vol.10, No.5, 2006, pp. 606-611.
[9] P. Guo, and H. Tanaka, "Upper and lower possibility distributions with rough set concepts,", In Rough set theory and granular computing, M. Inuiguchi, S. Hirano, and S. Tsumoto eds., Springer, 2002, pp. 243-250.
[10] R. Jensen and Q. Shen, "Rough set based feature selection: A review," http: //hdl.handle.net/2160/ 490, in Rough computing: theories, technologies and applications, A.E. Hassanien, Z. Suraj, D. Slezak, and P. Lingras, eds. IGI global, 2007.
[11] C.J. Date, An Introduction to Database Systems, 8th ed., Addison Wesley, 2004.

Hyontai Sug received the B.S. degree in Computer Science and Statistics from Pusan National University, M.S. degree in Computer Science from Hankuk University of Foreign Studies, and Ph.D. degree in Computer and Information Science and Engineering from University of Florida in 1983, 1986, and 1998 respectively. During 1986-1992, he worked for Agency of Defense Development (ADD) as a researcher, and during 1999-2001, he was a full-time lecturer of Pusan University of Foreign Studies. He is now with Dongseo University as an associate professor since 2001.