IMPACT: International Journal of Research in Engineering & Technology (IMPACT: IJRET)
ISSN(E): 2321-8843; ISSN(P): 2347-4599
Vol. 2, Issue 3, Mar 2014, 17-24
© Impact Journals

AN EFFICIENT EXTENSION OF CONDITIONAL FUNCTIONAL DEPENDENCIES IN DISTRIBUTED DATABASES

SUCHITRA REYYA[1], M. PRAMEELA[2], G. VASAVI YADAV[3], K. SANDHYA RANI[4] & A. V. BHARGAVI[5]

[1]Assistant Professor, Department of Computer Science & Engineering, Lendi Institute of Engineering & Technology, Andhra Pradesh, India

[2,3,4,5]Department of Computer Science & Engineering, Lendi Institute of Engineering & Technology, Andhra Pradesh, India

ABSTRACT

This paper propose an efficient data cleaning by using extended conditional functional dependencies (eCFD's) which is an extension of conditional functional dependencies (CFD's) to reduce the inconsistency of data, efficient conditional functional dependencies intend to solve the multi valued inconsistencies to trounce drawbacks of CFD's which use pattern tableau to hold individual tuples in a table for cleaning relational data by supporting only single value attributes. One of the central problems for data quality is inconsistency detection. Given a database D and a set ∑ of dependencies as data quality rules, we want to identify tuples in D that violate some rules in ∑. When D is a centralized database, there have been effective eCFD's techniques for finding violations. It is however; far more challenging when data in D is distributed in which inconsistency detection often necessarily requires shipping data from one system to another.

KEYWORDS: CFD's, eCFD's, Nested Relational Database, Distributed Database

INTRODUCTION

Data cleaning is also called as data scrubbing, which deals with detecting and removing errors and inconsistencies from data in order to improve the quality of data. Thus data cleaning plays a major role of preprocessing. One way of achieving this is by using conditional functional dependencies (CFD's). CFD's were developed which aim at capturing violations for individual tuples by using the concept of pattern table 1. When these pattern tables are implied on the entire database the tuples which violate the pattern tables are detected and hence can be easily corrected. CFD's can hold only single value attributes but not for multi valued attributes. To overcome this problem we are using eCFD's for improving data quality which are extended from CFD's to work over the problem of multi valued attributes by using nested relational database[5][6].

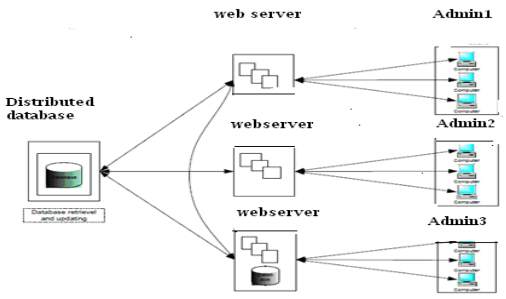

Figure 1

This paper develops techniques for detecting violations of CFD’s in relations that are fragmented and distributed across different systems. Distributed database is nothing but a collection of several different databases that look like a single database to the user. Data in distributed database system is stored across several systems, and each system is typically managed by DBMS that can run independent of each other systems, and eCFD’s are also used for detecting violations in multi valued inconsistencies.

Here we take the centralized system where we enter the eCFD’s data which eliminates the redundant data, where it stores that data in the centralized system and it distributes the data to Centralized Database (D1) and Centralized Database (D2) here if the D1 is cleaned and updated D2 and centralized system also will be cleaned and updated. And if D2 is cleaned and updated both the D1 and the centralized system will also be cleaned and updated automatically.

RELATED WORK

Our project is worked on the Effective Conditional Functional Dependencies (ECFD’s) based on the Conditional functional dependencies (CFD’s). Relations that contain redundant information may potentially endure from update anomalies. Functional dependencies are used to detect the redundancies by enforcing integrity constraints at schema level [1]. The constraints hold for all the tupelos in the table. The main idea behind FD is that the value of a particular attribute uniquely determines the value of some other attribute [8] A relation A->B if “for every valid instance of A, that value of A uniquely determines the value of B”. Data quality is recognized as one of the most important problems for data management [1].

A central technical problem for data quality concerns inconsistency detection, to identify errors in data. More specifically, given a database D and a set ∑ of dependencies serving as data quality rules, the detection problem is to find all the violations of ∑ in D, i.e., all the tuples in D that violate some rules in ∑. For a data quality tool to be effective in practice, it is

Definition of Conditional Functional Dependencies

Conditional Functional D were recently introduced for data cleaning. They extend standard FD’s by enforcing patterns of semantically related constants. CFD’s have been more effective than FD’s in detecting and repairing inconsistent data. A CFD ϕ over R is a pair (X->A, tp), where x is a set of attributes in attribute(R), and A is a single attribute in attribute(R), X -> A is a standard functional dependency, referred to as the functional dependency embedded in ϕ and tp is a pattern tuple with attributes in X and A, where for each B in X U{A}, tuple [B] is either a constant “a” in domain(B), or an unnamed variable “-” that draws values from domain(B). By using this CFD we can remove the unwanted data, it cannot support the multi valued inconsistencies so we can use the eCFD’s.

Definition of Efficient Conditional Functional Dependency

Let R be some relational schema given as (R:X→Y, Tp, Tm) where (1) X, Y, Tp, Tm ∈ attr(R); (2) X→Y is a Embedded FD; (3) Tp is a pattern tableau consisting of finite number of pattern tuples; (4) Tm is a multi tableau holding multiple values of Y for each X attribute. Conditional functional dependencies were recently introduced for data cleaning they extend standard functional dependencies by enforcing patterns of semantically related constants CFD’s is used to detecting inconsistencies of data.

ELIMINATE THE UNWANTED DATA IN DISTRIBUTED DATABASE USING NESTED RELATIONAL DATABASE

This paper proposes eCFD's which hold for multi value data. This is achieved using nested Database. The data after corrected will be distributed to different systems. Then overall design of the system is given below

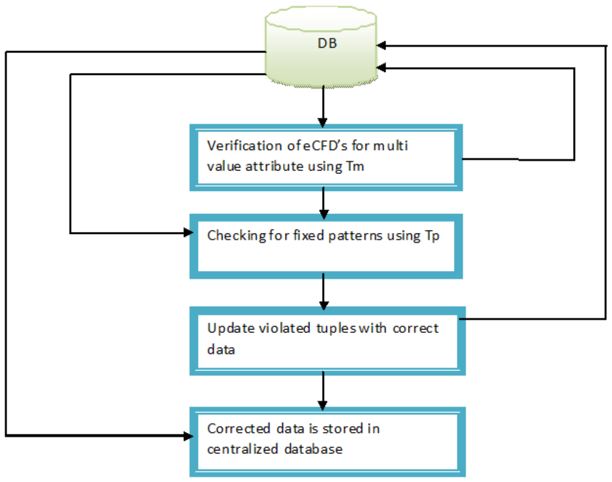
DB
Verification of eCFD's for multi value attribute using Tm
Checking for fixed patterns using Tp
Update violated tuples with correct data
Corrected data is stored in centralized database

Figure 2: A Model for Achieving eCFD's in Distributed Database

The model architecture describes the step by step flow of process carried to implement eCFD's. In the first step the data in the database is compared with table Tm to check for multi value inconsistencies. In the second step if the inconsistencies are found then it compares each tuple with matching fields from the database and table Tm with pattern table Tp. In third step the incorrect tuple is updated with matching value from multi tableau and saved in database. In step 4 corrected the data is stored in the Centralized Database and it can be send to the D1 &D2. In step 5 the data is cleaned and updated in D1 and automatically cleaned and updated in D2&Centralized system and vice versa.

Example

Let us consider a schema cust (id_num, Country code, Area code, Phone number, customer name, Area, city, pin).

In the below table we can observe that the city NYC and Vizag has different pin

Φ1: City ∈ {NYC} → ϕ with pin ∈ {07974, 01202, 01204}.

Φ2: City ∈ {VIZAG} → ϕ with AC ∈ {530013, 530014}.

Table 1: Customer Database (CD)

Id_Num	Country Code	Area Code	Phone Number	Customer Name	Area	City	Pin
1	1	908	111111	Mike	Tree ave	Nyc	07974
2	1	908	111111	Nick	Tree ave	Nyc	07974
3	1	212	222222	Joe	Elm str	Nyc	01202
4	1	212	222222	Jim	5th ave	Nyc	01205
5	5	0891	2535015	Ben	Eenadu	Vizag	530013
6	91	0891	2535015	Teena	Arilova	Vizag	530014
7	1	215	9885430122	Martin	Siripuri	Vzm	02349
8	1	216	9185430122	Meena	Oka ave	Hyd	832108
9	91	0891	2535015	Davidson	Troy	Vizag	530015

Hence in ϕ1 if the city is NYC then the pin must be one of the following values. Similarly in Φ2 if the is VIZAG the pin must be one of the above given values. In the above example tuple t4 contains City as NYC but pin is 01205 which is not one of the pin associated with NYC and tuple t9 has City as Vizag but pin is 530015. This problem is not overcome in CFD‟s as it does not hold multi-valued attributes. This multi-valued attribute problem can be solved with by less complexity by taking two tables Tm & Tp.

Table Tm: Tm is a multi tableau holding all the tuples consisting of multi- value of Y for a single attribute X. The Tm table for the given customer (cust) instance is shown below.

Table 2: Multi Tabelau-Tm

City	Pin
Nyc	{07974,01202,01204}
Vizag	{530013,530014}

Now to correct the pin in tuple t4 appropriate pin must be selected from multiple values in table Tm. So we use the pattern table Tp which consists of fields, Area and pin to select the pin that match the fields in the tuple. The pattern tableau for the cust relation is shown below

From cust relation we observe that in the tuple t4, area 5th ave and city is NYC which uniquely determines pin as 07974. Thus now the violation showed in tuple t4 can be eliminated by correcting it with pin as 07974 using these two tables.

Table 3: Pattern Tableau-Tp

Area	Pin
Tree ave	07974
Elm str	01202
5th ave	07974
Eanadu	530013
Arilova	530014
Troy	530013
Oka ave	832108

METHODOLOGY TO REDUCE REDUNDANCY

We introduce a query for solving multi value inconsistencies in a simplified way with less complexity. This query shows the violate tuples by considering customer relation table(C) and multi tableau(Tm) where the city’s area code (AC) does not match with the disjunction of options present in the multi tableau Tm.

```
Select All
    From cust C, table T
 Where (Tm.City ’@’)
    And Tm.pin [i….in] <> ‘@’)
 And (C.City=Tm.City
And C.pin<>Tm.pin [i….in]);
```

Once the violations are displayed the pin must be corrected with the appropriate one from the multiple pins.

This below query is used to update the pin by considering customer relation table (C), multi tableau(Tm) and pattern tableau (Tp) where the pin number of Tm matches with that of Tp and the street in customer relation matches in that of Tp.

```
Update
  Cust set
    C.pin=(select Tp.pin
  From table2 Tp
```

Algorithm: Eliminating inconsistent tuples in multi-valued attribute in distributed database using nested relational database

Input: Inconsistent multi-valued database (CD)

Output: Clean the database (CCD) and it is distributed to the different systems.

Method: Solving multi-valued inconsistencies in distributed database using Nested Relational database.

- Begin
- Consider a table CD with different fields.
- Take a new table multi tableau(Tm) having multituple values for attribute X i.e. X[i….in].
- For each tuple in table CD having attr X, compare it with X[i….in] of table Tm.
- If comparison successful then go to step15.
- Else
- Display tuples not satisfying step 4
- End for
- Consider a pattern tableau Tp having prefix constants and unnamed variables
- **For** updating the inconsistent values shown in step 7, for each tuple in table CD compare Tm.X[i…..in] with Tp.X and CD.Y with Tp.Y respectively.
- **If** comparision successful then
- Update attr X with tp.X
- **End if**
- **End for**
- Original data base(CD) is updated as clean Database(CCD)
- X attr stored in centralized system (CS).
- This attr are distributed to the CD1and CD2.

- The data will be cleaned and updated in CS automatically cleaned and updated in CD1 and CD2.
- **END**

Now after applying the above algorithm on Table 1, the consistencies in the table will be replaced by the accurate data, thereby making the database clean and error free. The below is the clean database (Table 4) obtained as a result of the algorithm. The pin in tuple t4 is replaced with 07974 and similarly tuple t9 with erroneous pin is replaced with appropriate pin as 530013.

Table 4: Centralized Database (CD)

Id_Num	Country Code	Area Code	Phone Number	Customer Name	Area	City	Pin
1	1	908	111111	Mike	Tree ave	Nyc	07974
2	1	908	111111	Nick	Tree ave	Nyc	07974
3	1	212	222222	Joe	Elm str	Nyc	01202
4	1	212	222222	Jim	5th ave	Nyc	07974
5	5	0891	2535015	Ben	Eanadu	Vizag	530013
6	91	0891	2535015	Teena	Arilova	Vizag	530014
7	1	215	9885430122	Martin	Siripuri	Vzm	02349
8	1	216	9185430122	Meena	Oka ave	Hyd	832108
9	91	0891	2535015	Davidson	Troy	Vizag	530013

Here the D1 is cleaned and update the data and automatically cleaned and update the data on centralized database and vice versa

Table 5: Centralized Database (D1)

Id_Num	Country Code	Area Code	Phone Number	Customer Name	Area	City	Pin
1	1	908	111111	Mike	Tree ave	Nyc	07974
2	1	908	111111	Nick	Tree ave	Nyc	07974
3	1	212	222222	Joe	Elm str	Nyc	01202
4	1	212	222222	Jim	5th ave	Nyc	07974

Table 6: Centralized Database (D2)

Id_Num	Country Code	Area Code	Phone Number	Customer Name	Area	City	Pin
1	5	0891	2535015	Ben	Eanadu	Vizag	530013
2	91	0891	2535015	Teena	Arilova	Vizag	530014
3	1	215	9885430122	Martin	Siripuri	Vzm	02349
4	1	216	9185430122	Meena	Oka ave	Hyd	832108
5	91	0891	2535015	Davidson	Troy	Vizag	530013

Here D2 will be cleaned and updated then automatically cleaned and update the data on centralized system and vice versa.

EXPERIMENTAL ANALYSIS

In this section, we present our findings about the performance of our models for reducing redundancies over multi-value databases.

Setup: For the experiments, we used postgre SQL on Windows XP with 1.86 GHz Power PC dual CPU with 3GB of RAM.

Efficiency of eCFD's: The efficiency of the eCFD's is graphically displayed by considering the multi-value database. When CFD's are posted on this database all tuples having multiple values for each attribute are changed to only one value and obtains duplicate data. But in eCFD's each attribute can hold multiple data thereby preserving the data.

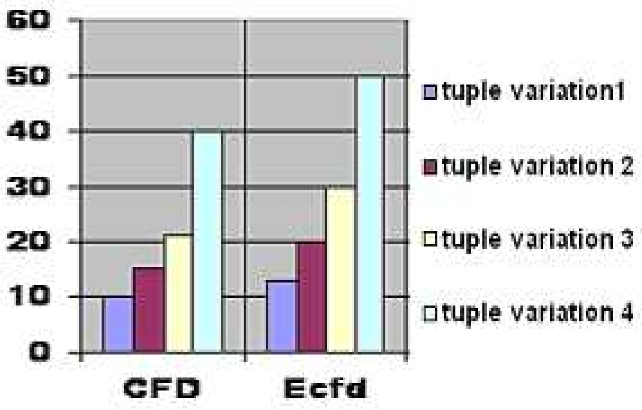

Figure 3: Graph Displaying the Efficiency of eCFD's

In this experiment we have analyzed how the redundancy is reduced by considering the space occupied by the pattern table (TP) and multi tableau (Tm) in CFD's and eCFD's respectively. It is observed Tp can hold only one value for each fixed pattern where as Tm can efficiently hold multiple values for each attribute in the pattern thus reducing tuple size.

Table 7: Comparison of Size of Tuples in Tp (CFD's) & Tm (eCFD's)

No. of Multiple Values	Tp	Tm
Pattern 1	1	2
Pattern 2	1	3
Pattern 3	1	4

CONCLUSIONS

We presented that nesting based eCFD's proved to be better than CFD's is eliminating redundancy in nested relational database. ECFDs acquire no extra complexity in inconsistency detection. We have developed eCFD's based technique for avoiding violations. Our experimental results for efficiency and size of the relation proved eCFD's to be more effective compared to CFD's. Finally we conclude that in which we can remove unwanted data by using eCFD's and Distributed to different sites by using Distributed Database. Updates are performed on the centralized database D1, D2… In the centralized database D1 the data will be cleaned and updated automatically the centralized database D2 will be cleaned and vice versa.

ACKNOWLEDGEMENTS

We express our sincere and profound gratitude to LENDI institute management members of our Chairman Sri P. Madhusudhana Rao, Vice-Chairman Sri P. Srinivasa Rao and Secretary Sri K. Siva Rama Krishna for their valuable support and providing good infrastructure. Our special thanks to Principal Dr. V. V. Rama Reddy for his motivation and encouragement. We also thank our Head of the Department Professor A. Rama Rao for giving guidance and continuous support.

REFERENCES

1. Philip Bohannon, Wenfei Fan, Floris Geerts and Xibei Jia3 Anastasios Kementsietsidis3: *Conditional Functional Dependencies for Data Cleaning.*

2. Wenfei Fan, Floris Geerts, Laks V.S. Lakshmanan and Ming Xiong: *Discovering Conditional Functional Dependencies*, IEEE conference on data engineering.

3. Suchitra Reyya, Sangeeta Viswanadham: *Eliminating Data Redundancy through Weak Multi valued Dependencies using Nesting.*

4. Suchitra Reyya, M. Sundara Babu, Ratnam Dodda: *An Efficient Storage of Spatial Database using Nested Relational Database*, International Journal of Engineering Research & Technology (IJERT) Vol. 1 Issue 7, September – 2012.

5. Erhard Rahm, Hong Hai Do: *Data Cleaning Problems and Current Approaches*, IEEE Techn. Bulletetin on Data Engineering, Dec. (2000).

6. Korth, H., Roth, M.: *Query languages for Nested Relational Databases, Nested Relations and Complex Objects in Databases*, Springer, (1991).