Mining Functional Dependency in Relational Databases using FUN and Dep-Miner: A Comparative Study

Anupama A Chavan
M.Tech (2nd year)
Department of Computer Science and Engineering
Lord Krishna College of Technology

Vijay Kumar Verma
Asst. Professor M.Tech (CSE)
Department of Computer Science and Engineering
Lord Krishna College of Technology

ABSTRACT

Database is a collection of tables of data items, if the database is organized according to relational model it is called relational database. In a relational database, a logical and efficient design is just as critical. A poorly designed database may provide erroneous information, or may even fail to work properly may be difficult to use. Most of these problems are the result of two bad design features called redundant data and anomalies. Database normalization is the process of designing a database satisfying a set of integrity constraints, efficiently and in order to avoid inconsistencies when manipulating the database. Most of the research work has been devoted to functional dependencies. There are several algorithms have been developed in the past year like TANE, FD_Mine FD_Discover, Dep-Miner, FUN, FD Analysis using Rough sets, FD discovery by Bayes Net. In This paper we present a comparative study over Dep-Miner and FUN. We compare the working process of Dep-Miner and FUN using a simple example.

Keywords

Functional dependencies, closure of set, redundancy, normalization.

1. INTRODUCTION

To discover dependency existing in an instance of a relation received considerable interest as it allowed automatic database analysis. Knowledge discovery and data mining database management reverse engineering and query optimization are among the main applications benefiting from efficient dependencies discovery algorithms [6].

Redundancy is often caused by functional dependency. A functional dependency is a link between two sets of attributes in a relation. We can normalize a relation by removing unwanted FDs. Normalization transforms unstructured relation into separate relations, called normalized ones. The main purpose of this separation is to eliminate redundant data and reduce data anomaly. The data is inconsistent due insert, update, and delete operations and repetition of information. There are many different levels of normalization depending on the purpose of database designer such as 1NF, 2NF, 3NF, BCNF, 4NF, 5NF to make database free from all the anomalies Most database applications are designed to be either in the third, or the Boyce-Codd normal forms in which their dependency relations are sufficient for most organizational requirements. [2, 8]

2. BASIC CONCEPTS

2.1 Functional Dependency

Given a relation 'R', attribute 'Y' of 'R' is functional dependant on attribute 'X' of 'R' if- each 'X' value of "r' is associated with precisely one value of 'Y' in 'R' " . A functional dependency is a statement X →Y requiring that X functionally determines Y. For example
city → state
i.e. the state value depends on city value [7,8]

2.2 Free Set

A free set is a minimal set X of attributes in schema R such that for any subset Y of X, |r[Y]|<|r[X]|. Thus, every single attribute is a free set because they do not have a subset. If X is a free set, $A \in (R-X)$, and $|X|<|XA|$ and $|A| < |XA|$, then XA is another free set. The lhs of any minimal FD is necessarily a free set. The free set of relation r, denoted by Fr(r), is a set of all free sets on r.

2.3 Closure of Set

The closure of set X is calculated using cardinality as $X^+ = X + \{A | A \in (R-X) \wedge |r[X]|\}$. That is, X^+ contains attribute A on a node at next level if X→ A.

2.4 Quasi-closure of Set

The quasi-closure of X is
$X^o = X + (X - A_1)^+ + - - - - - - + (X - A_k)^+$. In fact X^o contains the attributes on all the parent nodes of X and all the dependent nodes of the parent nodes [1].

2.5 Maximal Equivalence Class

Let r be a stripped partition database. The set MC of maximal equivalence classes of r is defined as follows
$MC = max_{\subseteq}\{c \in n \mid n \in r\}$.

2.6 Agree Set

Let ti and tj be tuples and *X* an attribute set. The tuples ti and tj agree on X if ti[X] = tj [X]. The agree set of ti and tj is defined as follows:
$ag(ti,tj) = \{A \in R / ti[A]=tj[A]\}$. If r is a relation, $ag(r) = \{ag(ti,tj)/ti,tj \in r, ti \neq tj\}$.

2.7 Maximal Set

A maximal set is an attribute set X which, for some attribute A, is the largest possible set not determining A. We denote by max(dep(r),A) the set of maximal sets for A [4].

3. FUN ALGORITHM

N. Novelli and R. Cicchetti, in 2001 proposed FUN: An Efficient Algorithm for Mining Functional and Embedded Dependencies. FUN describes FD approach at a general level only without detailing the optimizations due to the stripped partition database. Concept of free set is used for deriving FDs in this algorithm.

FUN uses the cardinality of projection r[X] to test FD satisfaction: | r[X] | = | r[XA] | if X→A. We note that | r[X] | is the same as the number of equivalent classes in the partition of X.[1] A free set P is a set of attributes such that removing an attribute from P decreases the number of P's distinct values. Free sets correspond to left hand sides of FDs. The concept of closure and quasi-closure is used to define right hand sides of FDs. The closure of P denoted by P^+ is the union of the attributes in P and the additional attributes which can be added to P without increasing the number of P's distinct values. The quasi-closure of P, is the union of P and the closures of P's maximal subsets, while all attributes in P^+ are determined by P, only those not in P's quasi-closure yield minimal FDs. In summary, the minimal FDs satisfied in r are the FDs of the form P→A where P is a free set. Steps followed by algorithm are shown in below figure. [1, 5, 9].

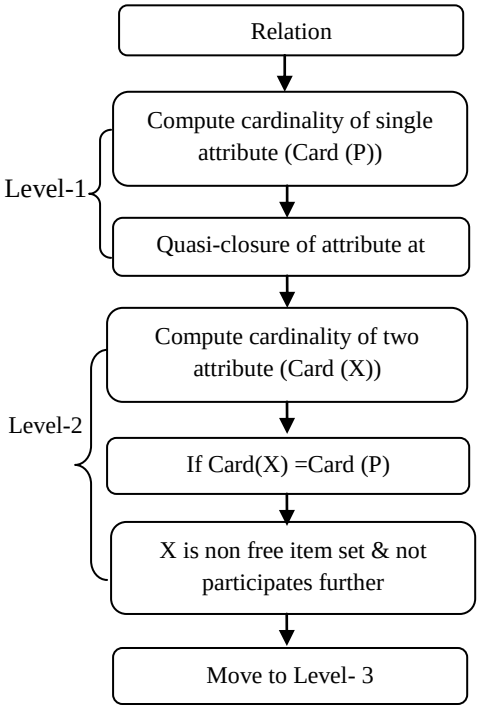

Fig 1: Steps in FUN Algorithm

In a levelwise manner, the algorithm searches for free sets of increasing sizes. At the level corresponding to FDs with a left hand side of size the algorithm knows from the previous level the free sets of size and their quasi-closure as well as the collection of candidate free sets of size s+1. It first computes the closure of the free

Sets of size s, and displays the FDs of the form X → A where X is a free set of size. Then, it computes the quasi-closure of the candidate free sets of size s+1 using the closure of the free sets of size s. Then, it prunes the candidate free sets X of size s" that are not free sets, based on the number of distinct values of X and of its maximal subsets that are free sets, Finally it generates the candidate free sets of size s +1 from the free sets of size s. Consider a simple employee data base shown in Table 1[5, 9].

Table 1. Simple employee data base

T No	Emp_No	Dep_No	Year	Dep_Name	Mgr_no
	A	B	C	D	E
1	1	1	2005	Production	5
2	1	5	2004	Marketing	12
3	2	2	2002	Sales	2
4	3	2	2008	Sales	2
5	4	3	2008	Purchase	2
6	5	1	1995	Production	5
7	6	5	1998	Marketing	12

From the above relation first find out Maximal equivalence class

{ {1,2},{1,6},{2,7},{3,4,5}}

In our example the concur set for the pair of tuples (1,2) is concur set con(1, 2)={A} Similarly, we have con(1,6) = con(2,7) = con(3,4) ={B,D,E}, con(3,5) = {E}, con(4,5) ={C,E} so the concur set of r are con(r)={A, BDE , E CE}. Complete working process of FUN is shown in the table 2.

Table 2. Working of FUN Algorithm

X	Cardinality	One item candidate set	Closure	FD
A	6	A	A	
B	4	B	B,D,E	B → D, E
C	6	C	C,E	C → E
D	4	D	B,D,E	D → B, E
E	3	E	E	
AB	Free set	A,B,D,E	A,B,C,D, E	AB → C
AC	Free set	A,C,E	A,B,C,D, E	AC →B, D
AD	Free set	A,B,D,E	A,B,C,D, E	AD → C
AE	Free set	A,E	A,B,C,D, E	AE → B,C,D
BC	Free set	B,C,D,E	A,B,C,D, E	BC → A
BD	Not free	------	------	
BE	Not free	-----	------	
CD	Free set	B,C,D,E	A,B,C,D, E	CD → A
CE	Not free	------	------	
DE	Not free	-------	------	

4. DEP-MINER ALGORITHM

St_ephane Lopes, Jean-Marc Petit, and Lot_ Lakhal in 2000 proposed a new efficient algorithm called Dep-Miner. Dep-Miner is used for discovering agree sets, maximal sets, left-hand sides of minimal non-trivial functional dependencies and real-world Armstrong relations. In Dep-Miner the underlying idea is based on the concept of agree set, which groups all attributes having the same value for a given pair of tuples. From these sets, maximal sets we can derive. The maximal sets for some attribute A are the largest possible sets of attributes not determining A. Then from the complements of

these maximal sets they derive the left hand sides of FDs using a levelwise algorithm for each attribute A it searches for left hand sides X by increasing the size of X. The steps followed by algorithm are shown in below figure.

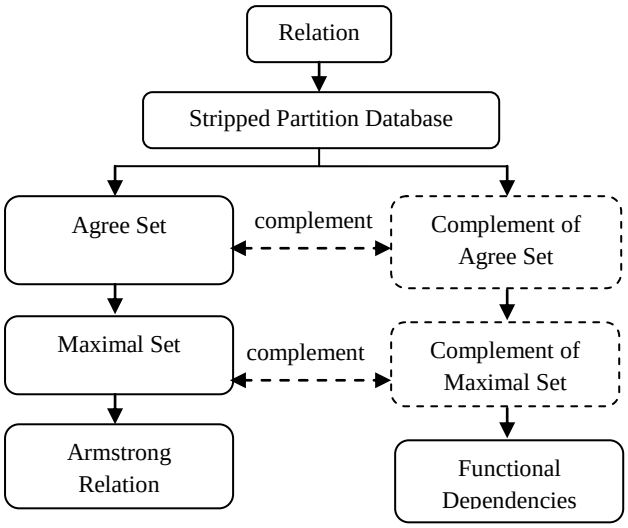

Fig 2: Steps in Dep-Miner Algorithm

Stripped partition databases, is used to compute agree sets. To avoid computing agree sets for all pairs of tuples by limiting themselves to the tuples within MC the set of maximal equivalence classes of the stripped partition database. So if we consider employee database given in table 1 maximal equivalence class is MC= {{1,2},{1,6},{2,7},{3,4,5}}.

For building agree sets, we only consider couples of tuples belonging to a common equivalence class of MC. In our example the agree set for the pair of tuples(1,2) is ag{A} Similarly we have ag(1,6), ag(2,7) ag(3,4) ={B_D_E} ag(3,5)={E} ag(4,5)={C,E},so agree sets of r are ag(r)={A,BDE, E,CE}.

The maximal sets from the agree sets as follows, in this example, we have max (A, r) = {BDE, CE} and the complement of the maximal set of A is cmax {A, r} = {AC, ABD}. Finally, the left-hand sides of FDs are defined from these complements of maximal sets and last generate Armstrong relation from maximal sets.

The entire working process of Dep-Miner is shown in the table 3 [4, 9].

Table 2. Working of Dep-Miner Algorithm

RHS	cmax (RHS,r)	Size1		Size2	
		Candidate	Traversal	Candidate	Traversal
A	{AC,ABD}	A,B,C,D	A	BC,BD, CD	BC, CD
B	{BCDE,ABD, ABCD}	A,B,C,D, E	B,D	AC,AE, CE	AC, AE
C	{BCDE,AC, ABCD}	A,B,C,D, E	C	AB,AD, AE,DB, BE,DE	AB, AD, AE
D	{BCDE,ABD, ABCD}	A,B,C,D, E	B,D	AC,AE, CE	AC, AE
E	{BCDE}	B,C,D,E	B,C, D,E	-	

So final FDs are

BC→A, CD→A, D→B, AC→B, AE→B, AB→C, AD→C, AE→C, B→D, AC→D, AE→D, B→E, C→E, D→ E

5. CONCLUSION

In FUN testing FDs is based on partition refinement. The dependency holds if the partition refines means if every equivalence class in is a subset of some equivalence class. In FUN approach describe a general level only without detailing the optimizations due to the stripped partition. Characterization of FDs is based on the concept of free sets. Free sets correspond to left hand sides of FDs. Right hand sides FDs define the closure and quasi-closure of an attribute set.

Dep-Miner discovers FDs by considering pairs of tuples, i.e. agree sets. First, a stripped partition database is extracted from the initial relation. Then, using such partitions, agree sets are computed and maximal sets are generated. Thus, a minimum FD cover according to these maximal sets is found. Dep-Miner employs a levelwise search. It combines the discovery of functional dependencies along with the construction of real-world Armstrong relations

6. REFERENCES

[1] Jixue Liu, Jiuyong Li, Chengfei Liu, and Yong Feng Chen "Discover dependencies from Data—A review" IEEE transactions on knowledge and data engineering, vol. 24, no. 2, February 2012

[2] Nittaya Kerdprasop and Kittisak Kerdprasop "Functional dependency discovery via Bayes net analysis" recent researches in computational techniques, non-linear systems and control ISBN: 978-1-61804-011

[3] Jalal Atoum, Dojanah Bader and Larafat Awajan "Mining functional dependency from relational databases using equivalent classes and minimal cover " Journal of computer science 4 (6): 421-426, 2008 ISSN 1549-3636© 2008 science publications

[4] St_ephane Lopes, Jean-Marc Petit, and Lot_ Lakhal "Dep-Miner Effective Discovery of Functional Dependencies and Armstrong Relations" Springer-Verlag Berlin Heidelberg 2000, pp. 350-364

[5] N. Novelli and R. Cicchetti, "Fun: An Efficient Algorithm for Mining Functional and Embedded Dependencies" Lecture Notes in Computer Science Volume 1973, 2001, pp 189-203

[6] Y. Huhtala, J. Karkkainen, P. Porkka, and H. Toivonen, "Tane : An Efficient Algorithm for Discovering Functional and Approximate Dependencies," Computer J., vol. 42, no. 2, pp. 100-111, 1999.

[7] Vijay Verma and Pradeep Sharma," Data Dependencies Mining In Database by Removing Equivalent Attributes" IJCSE, Vol.-1, Issue-1, July 2013

[8] Avi Silberschatz , Henry F. Korth ,S. Sudarshan,"Databse System Concepts, Sixth Edition, McGraw-Hill ISBN 0-07-352332-1

[9] Charlotte Vilarem, "Approximate Key and Foreign Key Discovery in Relational Databases", University Of Toranato

[10] Vijaya Lakshmi, Dr. E. V. Prasad a fast and efficient method to find the conditional functional dependencies in databases International journal of engineering research and development e-issn: 2278-067, P-ISSN: 2278-800x, www.ijerd.com volume 3, issue 5 (august 2012), pp. 56