International Journal of Scientific Research in Computer Science & Engineering
Research Paper Vol-1, Issue-4 ISSN: 2320-7639

Data Dependencies Mining In Database by Removing Equivalent Attributes

Pradeep Sharma and Vijay Kumar Verma

[1]*Department of Computer Science Holker Science College Indore, India*
[*2]*Department of Computer Science & Engg. Lord Krishna College of Technology Indore. India*

Available online at www.isroset.org

Received: 28 July 2013 Revised: 08 August 2013 Accepted: 20 August 2013 Published: 30 August 2013

***Abstract*-**data Dependency plays a key role in database normalization, which is a systematic process of verifying database design to ensure the nonexistence of undesirable characteristics. Bad design could incur insertion, update, and deletion anomalies that are the major cause of database inconsistency [1, 2]. The discovery of Data Dependency from databases has recently become a significant research problem this paper, we propose a new algorithm, called DM_EC (dependency mining using Equivalent Candidates) for the discovery of all Dependency from a database. DM_EC takes advantage of the rich theory of Functional dependencies [1, 3, 4]. The use of Functional dependencies theory can reduce both the size of the dataset and the number of FDs to be checked by pruning redundant data and skipping the search that follow logically from the Functional dependencies already discovered. We show that our method is sound, that is, the pruning does not lead to loss of information. Experiments on datasets show that DM_EC can prune more candidates than previous methods [5].

***Keywords*-** DBMS Normalization, Data Dependencies Mining, Data Mining

I. INTRODUCTION

Database design methodology normally starts with the first step of conceptual schema design in which users' requirements are modeled as the entity relationship (ER) diagram. The next step of logical design focuses on the translation of conceptual schemas into relations or database tables. Physical design concerns the performance issues such as data types, indexing option and other parameters related to the database management system. Conceptual schema and logical designs are two important steps regarding correctness and integrity of the database model. Database designers have to be aware of specifying thoroughly primary keys of tables and also determining extensively relationships between tables. Data normalization is a common mechanism employed to support database designers to ensure the correctness of their design [2, 5, 7].

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 (i.e., data inconsistency as a result of insert, update, and delete operations). There are many different levels of normalization depending on the purpose of database designer. 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 [6].

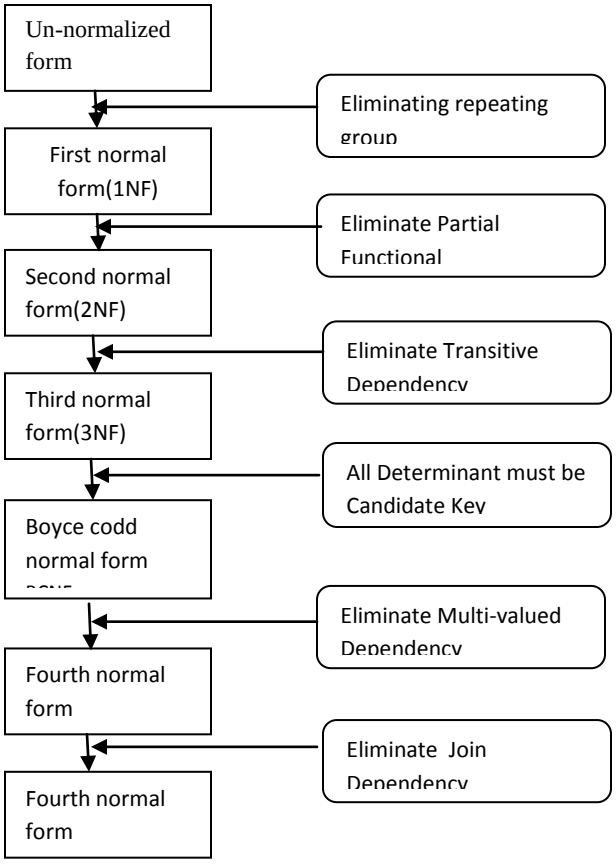

Figure 1 Normalization steps

Corresponding Author: *Vijay Verma*
Department of Computer Science & Engg. Lord Krishna College of Technology Indore. India

II. RELATED WORK

Our main objective is the induction of functional dependency relationships from the database instances.
Silva and Melkan off was the first team attempting to discover functional dependencies (FDs) through the data mining technique. The complexity of discovering FDs from existing database instances has been studied by Mannila and Raiha [8, 9]. Early work on FD discovery handled the complexity problem by means of partitioning the set of rows according to their attribute values and performs a level-wise search for desired solution. The later work of Wyss et al and Atoum et al. applied the minimal cover concept. Researchers in the application area of database reverse engineering are also interested in the same Objective. Lee and Yoo [10] proposed a method to derive a conceptual model from object-oriented databases. The final products of their method are the object model and the scenario diagram describing a sequence of operations. The work of Perez et al. emphasized on relational object-oriented conceptual schema extraction. Their technique is based on a formal method of term rewriting. Rules obtained from term rewriting are then generated to represent the correspondences between relational. Chen et al. Also apply association rule mining to discover new concepts leading to a proper design of relational database schema [11,12]. The work of Pannurat et al. and Alashqur are also in the line of association mining technique application to the database design. Besides functional dependencies, other kinds of database relationships are also explored. De Marchi et al. studied the problem of inclusion dependencies [11, 13]. Fan et al. proposed the idea to capture conditional FDs. Calders et al. Introduced a notion of roll-up dependency to be applied to the OLAP context. Approximate FDs concept has been recently applied to different subfield of data mining such as decision tree building, data redundancy detection, and data cleaning [9, 14, 15].

III. PROBLEM STATEMENT

Early methods for discovering of FDs were based on repeatedly sorting and comparing tuples to determine whether or not these tuples meet the FD definition.
Consider a simple transactional data base

Tuple No	A	B	C	D	E
T1	0	0	0	2	0
T2	0	1	0	2	0
T3	0	2	0	2	2
T4	0	3	1	2	0
T5	4	1	1	1	4
T6	4	3	1	1	2
T7	0	0	1	2	0

Table 1 transactional database

For example, in Table 1, the tuples are first sorted on attribute A, then each pair of tuples that have the same value on attribute A is compared on attribute B, C, D, and E, in turn, to decide whether or not

$A \rightarrow B$, $A \rightarrow C$, $A \rightarrow D$ $A \rightarrow E$ holds.

Then the tuples are sorted on attribute B and examined to decide whether or not

$B \rightarrow A$, $B \rightarrow C$, $B \rightarrow D$ $B \rightarrow E$

This process is repeated for C, D, E, AB, AC, AD, and so on. After the last candidate BCDE has been checked, all FDs will have been discovered. All candidates of five attributes are represented in Figure 2.

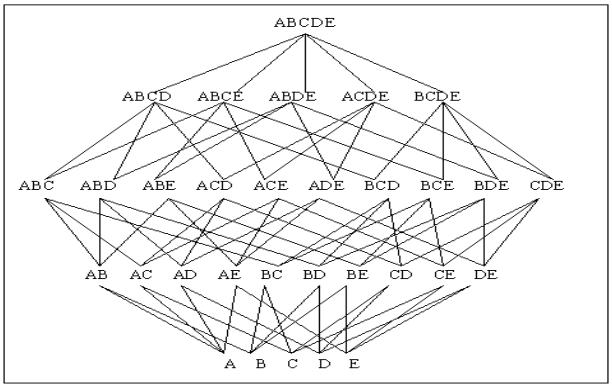

Figure 2 all candidate set for five attribute

This approach is inefficient because of this extra sorting and because it needs to examine every value of the candidate attributes to decide whether or not a FD holds. As a result, this approach is highly sensitive to the number of tuples and attributes. It is impracticable for a large dataset.

IV. EQUIVALENT ATTRIBUTES

a. Equivalent Attribute: - Let X and Y be candidates over a dataset D, if $X \rightarrow Y$ and $Y \rightarrow X$ hold, then X and Y are said to be equivalent candidates, denoted as $X \leftrightarrow Y$

b. Armstrong augmentation and transitivity rules:-

(1) Let X, Y and Z be candidates over D. If $X \leftrightarrow Y$ and $XW \rightarrow Z$ holds, then $YW \rightarrow Z$ holds.

(2) Let X, Y and Z be candidates over D. If $X \Phi Y$ and $WZ \rightarrow X$ hold, then $WZ \rightarrow Y$ holds

c. Nontrivial closure:- Let F be a set of FDs over a dataset D and X be a candidate over D. The
Closure of candidate X with respect to F, denoted Closure(X), is defined as {Y | $X \rightarrow Y$ can be deduced from F by Armstrong's axioms}. The nontrivial closure of candidate X with respect to F, denoted Closure'(X), is defined as Closure'(X) = Closure(X) – X.

d. Cardinality of the partition:- Let t1, t2, …, tn be all tuples in a dataset D, and X be a candidate over D. The partition over X, denoted $|\pi_x|$ is a set of the groups, such that t_i and t_j are in the same group if $t_i[X] = t_j[X]$. The number of the groups in the partition is called the cardinality of the partition, denoted $|\pi_x|$ [13, 14, 15].

V. PROPOSED METHODS

The proposed methods first reduce the size of the database to be searched, and the remainder reduces the number of candidates. Before giving pruning rule about equivalent candidates, we henceforth restrict the term equivalent candidate XΦY to a canonical form such that X is always the candidate that is generated earlier than candidate Y. For example, for a relational schema R={A, B, C, D}, canonical form corresponds to alphabetical order. So AΦB and BC→D are in canonical form, but DΦB and CB→D are not. We use following pruning rules

Pruning rule 1. If XΦY, then candidate Y can be deleted.

Pruning rule 2. If X is a key, then any superset XY of X does not need to be checked.

Proposed Algorithm

Purpose: To discover all functional dependencies in a dataset.

Input: Database D and its attributes X1, X2, ... Xm

Output: FD_SET, EQ_SET and KEY_SET

{

1. Initialization Step

set R = {X_1, X_2, ..., X_m}, set FD_SET = ϕ,

set EQ_SET = ϕ, set KEY_SET = ϕ

set CANDIDATE_SET = {X_1, X_2, ..., X_m}

∀Xi ∈ CANDIDATE_SET, set Closure'[Xi] = ϕ

2. Iteration Step

While CANDIDATE_SET≠ ϕ do

{

∀Xi ∈ CANDIDATE_SET do

{

ComputeNonTrivialClosure(Xi)

ObtaintFDandKey

}

ObtainEQSet (CANDIDATE_SET)

PruneCandidates (CANDIDATE_SET)

GenerateNextLevelCandidates

(CANDIDATE_SET)

}

3. Display (FD_SET, EQ_SET, KEY_SET)

}

Now we can explain the working of proposed algorithms through a example .consider the given database in table 1

Level 1

Candidates	|πx|	Closure	FD
A	2	D	A→D
B	4	Φ	
C	2	Φ	
D	2	A	D→A
E	4	Φ	

Table 2

At level 1

FD_Set = {A→D, D→A} and EQ_Set= {A, D}

Prune_Set = {A, B, C, E}

At Level 2

FD_Set = { AB→E, BE→A, CE→A } and EQ_Set= {AB,

BE} Prune_Set = {AB, AC, AE, BC,CE}

Candidates	|πx|	Closure	FD
AB	6	E	AB→E
AC	3	∂	
AE	5	Φ	
BC	6	Φ	
BE	6	A	BE→A
CE	6	Φ	CE→A

Table 3

At level 3 no candidate generates so search is terminated. Final dependencies are

FD_Set= {A→D, D→A, AB→E, BD→E, BE→A,

BE→D, CE→A, CE →D}

EQ_set={A, D},{AB, BE}

VI. EXPERIMENTS AND PERFORMANCE ANALYSIS

The working of Proposed Algorithms can be denoted by a simple diagram on the database given in table 1

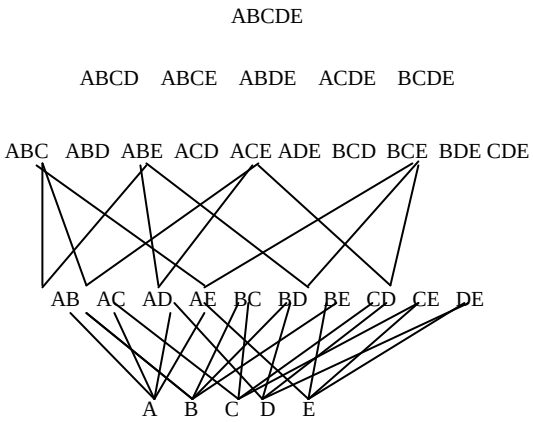

Figure 3(working of proposed methods)

At level1 there are five candidates in which A and D are fond to be equivalent so d can be remove at next there are 20 comparisons are made and at next level there are only 12 comparisons are made.

Now he working of TANE Algorithms can be denoted on the same on the database given in table 1

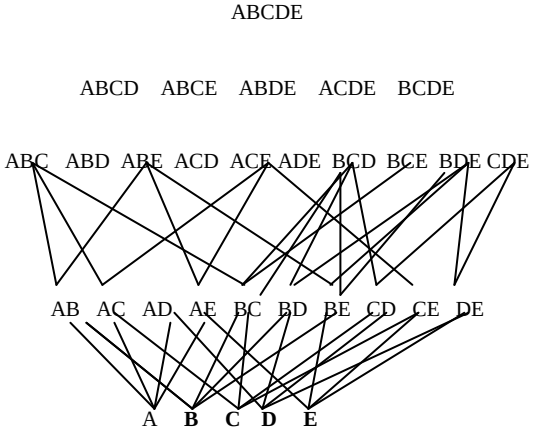

Figure 4(working of proposed methods)

Comparison of both algorithms using number of dependency checking table

Search level	Proposed algorithms	TANE
Level1	5	5
Level1	20	20
Level1	12	21
Total	37	46

Table 4 Dependency Comparison table

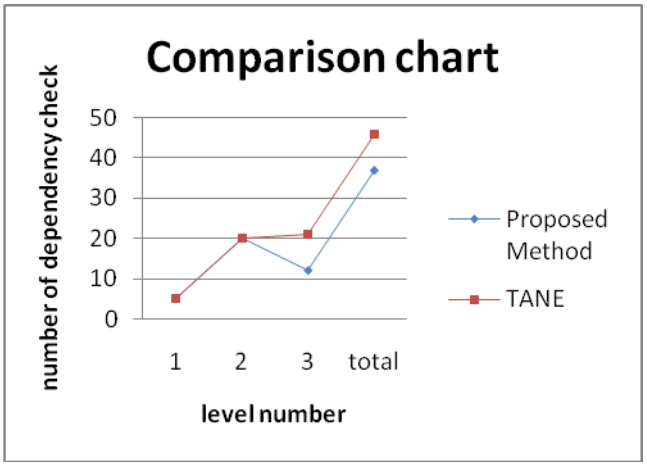

Figure 5 Comparison graph

VII. CONCLUSION

In this paper we identify several properties of functional dependencies, equivalences, and nontrivial closures that allow them to be used during the knowledge discovery process Like TANE, proposed algorithm is based on partitioning the database and comparing the number of partitions and provides additional pruning rules, based on our analysis of the

Theoretical properties of functional dependencies. These pruning rules are guaranteed not to eliminate any valid candidates, they reduce the size of the dataset or the number of checks required. Form the example and the results show that the pruning rules in the proposed algorithm are valuable because they increase the pruning of candidates and reduce the overall amount of checking required to find the same FDs.

REFERENCE

[1] St. Fephane Lopes, Jean-Marc Petit, and Lot_ Lakh Efficient Discovery of Functional Dependencies and Armstrong Relations C. Zaniolo et al. (Eds.): EDBT 2000, LNCS 1777, pp. 350{364, 2000. Springer-Verlag Berlin Heidelberg 2000.

[2] 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.

[3] Catharine Wyss, Chris Giannella, and Edward Robertson FastFDs: A Heuristic-Driven, Depth-First Algorithm for Mining Functional Dependencies from Relation Instances Computer Science Department, Indiana University, Bloomington, IN 47405, USA

[4] Fabien De Marchi CLIM: Closed Inclusion dependency mining in databases This work has been partially Funded by the French National Research Agency

DEFIS 2009 Program, project DAG ANR-09-EMER-003-01

[5] Katalin Tunde Janosi Rancz And Viorica Varga A Method For Mining Functional Dependencies In Relational Database Design Using Fca Studia Univ. Babes_{Bolyai, Informatics, Volume Liii, Number 1, 2008

[6] Wenfei Fan Dependencies Revisited for Improving Data Quality *PODS'08,* June 9–12, 2008, Vancouver, BC, Canada.Copyright 2008 ACM

[7] Pierre Allard⋆, Sebastien Ferr´e, and Olivier Ridoux Discovering Functional Dependencies and IRISA, Universities de Rennes 1, Campus de Beaulieu 35042 Rennes Cedex, France Association Rules by Navigating in a Lattice of OLAP Views

[8] Y. V. Sreevani1, Prof. T. Venkat Narayana Rao2 Identification and Evaluation of Functional Dependency Analysis using Rough sets for Knowledge Discovery (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 1, No. 5, November 2010

[9] Fabien De Marchi1, St´ephane Lopes2, and Jean-Marc Petit1 Efficient Algorithms for Mining Inclusion Dependencies C.S. Jensen et al. (Eds.): EDBT 2002, LNCS 2287, pp. 464–476, 2002. Springer-Verlag Berlin Heidelberg

[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: 2278-067X, p-ISSN: 2278-800X, www.ijerd.com Volume 3, Issue 5 (August 2012).

[11] Hong Yao · Howard J. Hamilton Mining functional dependencies from data Received: 15 September 2007 Springer Science Business Media,

[12] Daisy Zhe Wang Michael Franklin Luna Dong Anish Das Sarma Alon Halevy Discovering Functional Dependencies in Pay-As-You- Go Data Integration Systems Electrical Engineering and Computer Sciences University of California at Berkeley

[13] Jalal Atoum, Dojanah Bader and 1Arafat Awajan Mining Functional Dependency from Relational Databases Using Equivalent Classes and Minimal Cover Journal of Computer Science 4 (6): 421-426, 2008

[14] Nittaya Kerdprasop And Kittisak KerdprasopData Engineering Research Unit Functional Dependency Discovery via Bayes Net Analysis Recent Researches in Computational Techniques, Non-Linear Systems and Control ISBN: 978-1-61804-011-4

[15] Mark Levene and Millist W. Vincent Justification for Inclusion DependencyNormal Form IEEE Transactions On Knowledge And Data Engineering, Vol. 12, No. 2, March/April 2000

