# An Efficient Binary to Decimal Conversion Approach for Discovering Frequent Patterns

Kapil Chaturvedi
Department of Computer Application
Rajiv Gandhi Technical University
Bhopal, MP, India

Ravindra Patel, Ph.D
Associate Professor
Rajiv Gandhi Technical University
Bhopal, MP, India

D.K. Swami, Ph.D
Professor
VNS Institute of Technology
Bhopal, MP, India

## ABSTRACT
Association Rule Mining(ARM) is a most vital field of data mining to discover interesting relationship between items from huge transaction databases it analysis the data and discover strong rules using different measures such as (support, confidence, lift, conviction) etc, various ARM algorithms are available in literature for discovering frequent patterns. Market Basket analysis is one of the most essential applications of ARM; other applications are pattern recognition, weblog data mining and special data analysis etc. In this paper we proposed B2DCARM algorithm to discover frequent pattern which use Boolean matrix based technique. This algorithm adopts binary to decimal conversion approach to discover frequent itemsets from huge transaction database which outperforms in both of the cases where support threshold is low or high and also better performs from efficiency point of view compare to available tree based approaches.

## Keywords
ARM, B2DCARM, Frequent Pattern mining, Boolean matrix

## 1. INTRODUCTION
Data mining is the process of extraction of hidden predictive information from large databases [1, 2], it is also known as knowledge discovery process (KDD) and the major characteristic of data mining is to provide "proactive information delivery from the business perspective**".** Data mining is the most popular research field since last two decades, many of mining techniques already exist and well investigated in literature[10], data mining has several applications like DNA pattern recognition, market analysis, web mining etc. Association rule mining is one of the most important and well researched field of data mining, which was first introduced by Zhang C Q, Zhang S C in 1993 [3, 4]. It aims to extract interesting correlations, frequent patterns, associations or casual structures among sets of items in the transaction databases or other data repositories, an association rule can be represent in the form of an implication expression as $A \rightarrow B$ where A and B are the disjoint itemsets, i.e. $A \cap B = \emptyset$ here support and confidence are two measures for finding the strength of an association rule with additional measurement factors like Lift and Conviction. These terms are briefly discussed below.

- **Support** Determines the occurrence frequency of item/rule in given dataset.

$$\mathbf{Support}(\mathbf{A} \rightarrow \mathbf{B}) = \frac{\mathbf{Count}(\mathbf{A} \cup \mathbf{B})}{\mathbf{No\ of\ transactions}}$$

- **Confidence** ascertains how frequently RHS of the rule present in the transaction where LHS of rule will also be present (i.e. items in B appear in transactions that contain A).

$$\mathbf{Confidence}(\mathbf{A} \rightarrow \mathbf{B}) = \frac{\mathbf{Support}(\mathbf{A} \cup \mathbf{B})}{\mathbf{Support}(\mathbf{A})}$$

- **Lift** is defines as "ratio of the observed support to that expected (if A & B were independent)"

$$\mathbf{Lift}(\mathbf{A} \rightarrow \mathbf{B}) = \frac{\mathbf{Support}(\mathbf{A} \cup \mathbf{B})}{\mathbf{Support}(\mathbf{A}) \times \mathbf{Support}(\mathbf{B})}$$

- **Conviction** is the ratio of the expected frequency of occurrence of A without B, that means "the frequency that the rule makes an incorrect prediction (if A & B were independent)"

$$\mathbf{Conviction}(\mathbf{A} \rightarrow \mathbf{B}) = \frac{\mathbf{1 - Support}(\mathbf{B})}{\mathbf{1 - Confidence}(\mathbf{A} \rightarrow \mathbf{B})}$$

For example the rule (PEN, PENCIL) $\rightarrow$ NOTE BOOK found in the sales transaction database of a stationery shop, would indicate that if a customer buy PEN and PENCIL together would also buy NOTE BOOK such information can be use full in decision making about marketing activities. Traditional ARM techniques can be classified in three categories.

1. Candidate generation approaches [5, 7].
2. Tree based approaches [6].
3. Matrix based approaches [8, 9].

There are many algorithms are available and well investigated in the literature based on these techniques.
This paper proposes a frame work for binary to decimal conversion Association Rule Mining (B2DCARM) algorithm for discovering the frequent patterns(in section III), section IV shows the experimental result and section V is the conclusion part which shows outcome of experiments performed.

## 2. RELATED WORK
### 2.1 Classical ARM approaches
Association rules are widely used in various areas such as telecommunication networks, market and risk management, inventory control etc. Typical association rule mining algorithms are Apriori and FP-Growth, first association rule mining algorithm was Apriori algorithm introduced by Agrawal and R. Srikant in (1994) [5] which uses a candidate generation approach to find frequent items. Another

algorithms is FP-growth (Frequent Pattern Growth) algorithm proposed by J. Han, J. Pei, Y. Yin in 2000 [6] which do not use candidates to discover the frequent patterns, it is two phase method which reads the database only twice and stores in form of a tree in the main memory; FP-growth uses a divide-and-conquer approach to decompose both the mining tasks and the databases, it avoid the costly process of candidate generation. Other approaches are matrix based approaches [8, 11, 12] which use Boolean logical operations to generate the association rules; it stores all data in the form of bits, so it needs less memory space.

A classification of traditional ARM approaches based on their features is as follows.

| **Algorithm** | **Key Feature/ Strategy** | **Working Procedure** | **Pros. /Cons.** |
|---|---|---|---|
| Apriori algorithm [5] | Counting/ BFS | Uses a candidate generation approach to find frequent items on the basis of support[5] | Outperform when high support count and number of items are less. |
| FP-Growth algorithm [6] | Counting/ DFS | Two Pass Technique: 1. counts occurrence (attribute-value pairs) in the dataset & store them to "HEADER" table 2. builds the FP-tree structure by inserting instances | FP Growth outperform only when low support, large result count and fast result are needed. Require much storage space in case large transaction set given to design & store a tree structure |
| Eclat algorithm [7] | Intersecting/ DFS | Use transaction id set intersections to compute the support of a candidate itemset avoiding the generation of subsets that does not exist in the prefix tree. | For very low support thresholds, it performs better comparativ ely other existing algorithms. |
| Matrix based approaches [9, 13, 16, 17] | It uses Boolean logical operations to generate the association rules. | Two Pass Technique: 1. Scan database once to generate Boolean matrix. 2. Apply algorithm to discover frequent item sets. | Stores all data in the form of bits, so it needs less memory space. Capable to apply on large relational databases. |

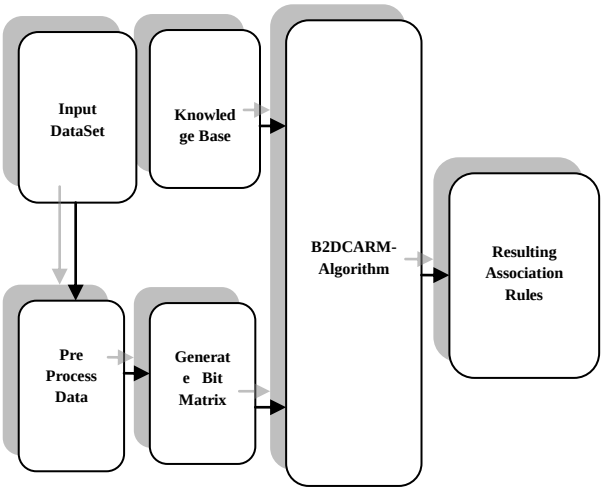
**3.1**

**3.2**

**3.3**

# 3. PROPOSED WORK

## 3.1 Architecture of B2DCARM Algorithm frame work




## 3.2 Steps involved in proposed algorithm are as follows

**PHASE 1:** Construct a matrix based on the presence and absence of items i.e. **"1" & "0"** indicate the presence and absence of items respectively.

**PHASE 2:** Preprocess the data by following two step procedure -

2.1 Count the number of 1's in column to check support count of an item.

2.2 Remove items which don't have minimum support count.

**PHASE 3:** Apply binary to decimal conversion association rule mining (**B2DCARM**) approach on bit matrix.

**PHASE 4:** End.

## 3.3 Procedure to generate Matrix

We generate a Boolean matrix which contains only 0 and 1 (where 0 and 1 represents the presence and absence of item respectively in transaction database). In matrix, row indicated the items occurred in transaction. B2DARM accepts that

Boolean matrix as input and derive frequent patterns. The process of generating the matrix is as follows:

Let $T=[T_1,T_2,\ldots..T_n]$ is the set of transactions represented as column and $I=[I_1, I_2,\ldots.,I_m,]$ be the set of items represented as rows. Then the generating matrix $MTX=\{MTX_{ij}\}$, $(i=1,2,3,\ldots..n; j=1,2,3\ldots.m)$ is an m×n matrix, where $M_{ij}=0$ or 1 determined by following rule,

$$MTXij = \begin{cases} 1, & \text{if } \mathbf{I}\,ij \in I \\ 0, & \text{if } \mathbf{I}\,ij \text{ Not} \in I \end{cases}$$

## 3.4 Algorithm-1 (Generate Matrix)

n - Number of Transactions
T- Transaction set
I - Set of items.
INPUT: D - Data base of Transactions
OUPUT: A Bit Matrix [Containing '0' and '1']
METHOD:

**STEP 1** START
**STEP 2** for(Each Transaction i=1 to n)
a. if I[i] Is Present in T
i. MTX[I,j]=1
else
ii. MTX[I,j]=0
b. End if
**STEP 3** End for
**STEP 4** END

## 3.5 Algorithm-2 (B2DCARM)

Mtx: Matrix
Gen: Generate
Conv: Conversion
Sup_Count: Support Count

**STEP 1** Begin

**STEP 2** Read Mtx

**STEP 3** Pre-Process Mtx(Mtx)
[Generating the bit (Boolean) matrix which contain 0 & 1]

**STEP 4** New_Mtx= Gen_Candidate(Bit_Mtx, Length)
[Generating combinations of items in the form of binary numbers]

**STEP 5** Decimal_Mtx=Bin2Dec_Conv(New_Mtx)
[Converting binary sequences in to corresponding Decimal numbers]

**STEP 6** Result_Mtx = Prun_Mtx(Decimal_Mtx, Sup_Count)
[Discard those combination which do not support pre defined thresholds]

**STEP 7** Repeat Step- 4 to 6

**STEP 8** End

## 3.6 Description

Let T=[T1, T2, T3, T4, T5, T6] is the set transactions and I=[A, B, C, D] be the set of items, here threshold is set 50% that means, item that is supported by at least three transactions would be frequent item. Table 1-I shows a Database contain six transactions, assume support count SC=50%, Table 1 contains 6 records, it means item that supported by atleast three transactions, would be frequent item.

**Table 1-I. Shows a list of transactions where T1, T2, T3, T4, T5 & T6 are transaction numbers and A, B, C, D, E are items.**

**Table 1-II. Assign the identification number (Item_ID) to the items occurred in transaction.**

| Table 1-I | | | Table 1-II | |
|---|---|---|---|---|
| **Tran_No.** | **Items** | | **Items** | **Item_ID** |
| **T1** | C D | | A | 1 |
| **T2** | A B C D E | ➔ | B | 2 |
| **T3** | A B C | | C | 3 |
| **T4** | A B D | | D | 4 |
| **T5** | A C D | | E | 5 |
| **T6** | A B C E | | | |

**Phase I**

**Table-2. Scan table once and generate Boolean matrix.**

| Tran_No. | A | B | C | D | E |
|---|---|---|---|---|---|
| **T1** | 0 | 0 | 1 | 1 | 0 |
| **T2** | 1 | 1 | 1 | 1 | 1 |
| **T3** | 1 | 1 | 1 | 0 | 0 |
| **T4** | 1 | 1 | 0 | 1 | 0 |
| **T5** | 1 | 0 | 1 | 1 | 0 |
| **T6** | 1 | 1 | 1 | 0 | 1 |

**Phase II.** Preprocess the data

**Table-3. Remove items which don't have minimum support count.**

| Tran_No. | A | B | C | D | ~~E~~ |
|---|---|---|---|---|---|
| **T1** | 0 | 0 | 1 | 1 | ~~0~~ |
| **T2** | 1 | 1 | 1 | 1 | ~~1~~ |
| **T3** | 1 | 1 | 1 | 0 | ~~0~~ |
| **T4** | 1 | 1 | 0 | 1 | ~~0~~ |
| **T5** | 1 | 0 | 1 | 1 | ~~0~~ |
| **T6** | 1 | 1 | 1 | 0 | ~~1~~ |
| **Supp. Count** | **5** | **4** | **5** | **4** | ~~**2**~~ |

**Phase III**

**TABLE-4. Generate combination of two items, in the form of binary sequences.**

| Tran_No. | AB | AC | AD | BC | BD | CD |
|---|---|---|---|---|---|---|
| **T1** | 00 | 01 | 01 | 01 | 01 | 11 |
| **T2** | 11 | 11 | 11 | 11 | 11 | 11 |
| **T3** | 11 | 11 | 10 | 11 | 10 | 10 |
| **T4** | 11 | 10 | 11 | 10 | 11 | 01 |
| **T5** | 10 | 11 | 11 | 01 | 01 | 11 |
| **T6** | 11 | 11 | 10 | 11 | 10 | 10 |

**TABLE-5. Swap Binary sequence by corresponding decimal numbers & discard which does not support given thresholds.**

| Tran_No. | AB | AC | AD | BC | ~~BD~~ | CD |
|---|---|---|---|---|---|---|
| T1 | 0 | 1 | 1 | 1 | ~~1~~ | 3 |
| T2 | 3 | 3 | 3 | 3 | ~~3~~ | 3 |
| T3 | 3 | 3 | 2 | 3 | ~~2~~ | 2 |
| T4 | 3 | 2 | 3 | 2 | ~~3~~ | 1 |
| T5 | 2 | 3 | 3 | 1 | ~~1~~ | 3 |
| T6 | 3 | 3 | 2 | 3 | ~~2~~ | 2 |
| Supp. Count | 4 | 4 | 3 | 3 | 2 | 3 |

**Table-6. Dataset after prune**

| **Tran_No.** | AB | **AC** | **AD** | **BC** | **CD** |
|---|---|---|---|---|---|
| **T1** | 0 | 1 | 1 | 1 | 3 |
| **T2** | 3 | 3 | 3 | 3 | 3 |
| **T3** | 3 | 3 | 2 | 3 | 2 |
| **T4** | 3 | 2 | 3 | 2 | 1 |
| **T5** | 2 | 3 | 3 | 1 | 3 |
| **T6** | 3 | 3 | 2 | 3 | 2 |
| **Supp. Count** | **4** | **4** | **3** | **3** | **3** |

**TABLE-7. Generate combination of three items, in the form of binary sequences.**

| Tran_No. | ABC | ABD | ACD | BCD |
|---|---|---|---|---|
| **T1** | 001 | 001 | 011 | 011 |
| **T2** | 111 | 111 | 111 | 111 |
| **T3** | 111 | 110 | 110 | 110 |
| **T4** | 110 | 111 | 101 | 101 |
| **T5** | 101 | 101 | 111 | 011 |
| **T6** | 111 | 110 | 110 | 110 |

**TABLE-8. Swap Table-7's binary sequences by corresponding decimal numbers & discard which does not support given thresholds.**

| **Tran_No.** | **ABC** | **~~ABD~~** | **~~ACD~~** | **~~BCD~~** |
|---|---|---|---|---|
| **T1** | 1 | ~~1~~ | ~~3~~ | ~~3~~ |
| **T2** | 7 | ~~7~~ | ~~7~~ | ~~7~~ |
| **T3** | 7 | ~~6~~ | ~~6~~ | ~~6~~ |
| **T4** | 6 | ~~7~~ | ~~5~~ | ~~5~~ |
| **T5** | 5 | ~~5~~ | ~~7~~ | ~~3~~ |
| **T6** | 7 | ~~6~~ | ~~6~~ | ~~6~~ |
| **Supp. Count** | **3** | **~~2~~** | **~~2~~** | **~~1~~** |

**TABLE-9. Shows the resulting rule generated by B2DCARM Algorithm,**

| **Tran_No.** | **ABC** |
|---|---|
| **T1** | 1 |
| **T2** | **7** |
| **T3** | **7** |
| **T4** | 6 |
| **T5** | 5 |
| **T6** | **7** |
| **Supp. Count** | **3** |

Fig-1 shows a MATLAB environment, where B2DCARM approach were implemented. Fig-2 shows the resulting rules generated by the algorithm implementation.

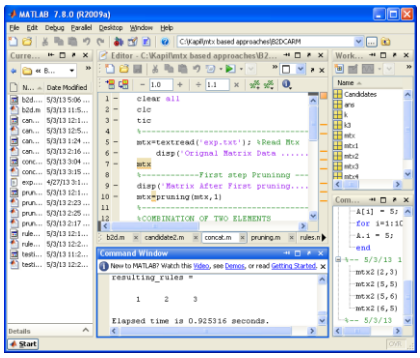


**Fig-1: MATLAB environment, where B2DCARM approach were implemented**

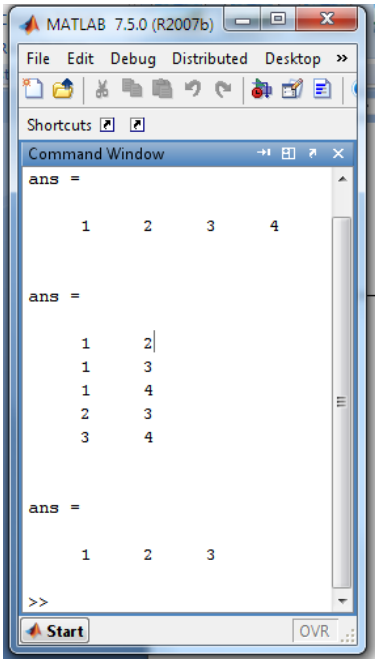



**Fig-2: Shows the resulting rules generated by B2DCARM Algorithm – MATLAB implementation.**

## 4. EXPERIMENTAL RESULT

To evaluate the performance of proposed B2DCARM algorithm, several comparative experiments has conducted on FP-Growth and B2DCARM algorithm to test efficiency and scalability of new approach, for this purpose we select datasets from [14][15] ( 1000X8 Database, Synthetic#1, Syntahatic#2 etc ) apply both algorithms on same number of transaction and compare the execution time shown in Fig-3, all experiments are performed on Intel core i3, 3.07GHz processor and 2GB of RAM, the program developed in MATLAB 7.5, Microsoft windows 2007 platform, the implementation of FP-Growth was downloaded from[15],

Fig-3 shows a performance evolution of FP-Growth and B2DCARM algorithms, here B2DCARM outperforms.

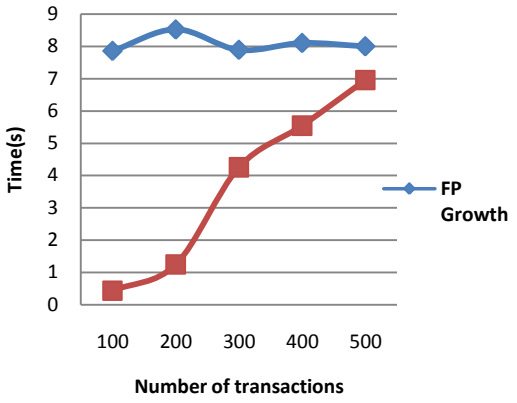



**Fig-3 Comparative graph between FP-Growth and B2DCARM**

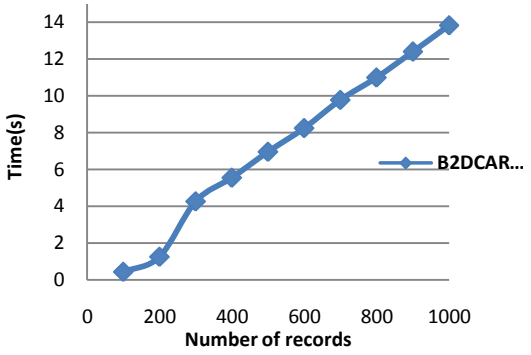



**Fig-4: Effect of Increasing the size of transactions, Support Count=20%**

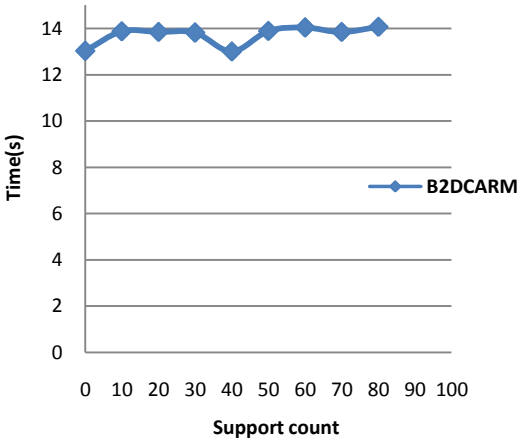



**Fig-5: test behavior of B2DCARM algorithm at different support threshold**

## 5. CONCLUSION

We proposed a new approach B2DCARM algorithm for discovering frequent pattern among Boolean databases of transactions. We compared the new approach with FP-Growth algorithm and illustrated the experimental result in Fig-3, Fig-4 and Fig-5 shows that the proposed technique performs better in order to time efficiency.

## 6. ACKNOWLEDGMENTS

We would like to thank to Mr. Sunil Joshi sir, Assistant Professor, Department of Computer Application, SATI (Degree) Vidisha for their help and suggestions.

## 7. REFERENCES

[1] U.M. Fayyad, et al.: "From Data Mining to Knowledge Discovery: An Overview", "Advances in Knowledge Discovery and Data Mining", AAAI Press/ MIT Press, pp 1-34, 1996.

[2] J. Han, M. Kamber, "Data Mining Concepts and Techniques", Morgan Kaufmann Publishers, San Francisco, USA, ISBN 1558604898, 2001.

[3] C Q Zhang, S C Zhang. "Association Rule Mining: Models and Algorithms". New York: Springer, 2002.

[4] R. Agrawal, T. Imielinski, and A. Swami, "Mining Association Rules between Sets of Items in Large Databases", In Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 207-216, 1993.

[5] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules", Proceedings 20th Very Large Databases Conference, Santiago, Chile, pp.487–499, 1994.

[6] J.Han, J.Pei,and, Y.Yin. "Mining frequent patterns without candidate generation", In Proceedings of the 2000 ACM SIGMOD international conference on Man agement of data, pp 1–12, 2000.

[7] M.J. Zaki. "Fast vertical mining using diffsets. Technical" Report 01-1, Rensselaer Polytechnic Institute, Troy, New York, 2001

[8] Hunbing Liu and Baishen Wang, "An Association Rule Mining Algorithm Based On Boolean Matrix", Data Science Journal,Volume 6,Supplement9, 2007 pp-63-66.

[9] ZHANG ZONG-YU, ZHANG YA-PING,"A parallel algorithm of frequent itemsets mining based on bit matrix", International Conference on Industrial Control and Electronics Engineering, 2012, pp.1210-1213

[10] Agrawal, R., Imielinski, T., Swami,A., "Database mining: A performance perspective. IEEE Trans. Knowledge and Data Eng.", 5(6) ,1993, pp-914-925.

[11] Yubo Yuan, Tingzhu Huang, "A Matrix Algorithm for Mining Association Rules", Springer-Verlag Berlin Heidelberg 2005, pp-370-379

[12] Pratima Gautam, K. R. Pardasani, "A Fast Algorithm for Mining Multilevel Association Rule Based on Boolean Matrix", (IJCSE) International Journal on Computer Science and Engineering Vol. 02, No. 03, 2010, pp 746-752

[13] Pav´on, J., S. Viana, and S. G´omez, "Matrix apriori: Speeding up the search for frequent patterns". In Proceedings of the 24th IASTED International Conference on Database and Applications, DBA'06, Anaheim, CA, USA ACTA Press, 2006 pp. 75–82.

[14] Database URL: http://www2.cs.uregina.ca/~dbd/cs831/notes/itemsets/datasets.php.

[15] An Implementation of FP-growth, "http://cgi.csc.liv.ac.uk/~frans/KDD/Software/FPgrowth/fpGrowth.html", Department of Computer Science, The University of Liverpool.

[16] Neelu Khare, Neeru Adlakha, K. R. Pardasani, "An Algorithm for Mining Multidimensional Association Rules using Boolean Matrix", IEEE International Conference on Recent Trends in Information, Telecommunication and Computing, 2010, pp. 95-99

[17] Xuezhi Chi, "A New Matrix-Based Association Rules Mining Algorithm", 9th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2012), 2012,pp. 633-636