World Academy of Science, Engineering and Technology
International Journal of Biotechnology and Bioengineering
Vol:2, No:12, 2008

Open Science Index, Biotechnology and Bioengineering Vol:2, No:12, 2008 publications.waset.org/12669.pdf

Proposing an Efficient Method for Frequent Pattern Mining

Vaibhav Kant Singh, Vijay Shah, Yogendra Kumar Jain, Anupam Shukla, A.S. Thoke, Vinay Kumar Singh, Chhaya Dule, Vivek Parganiha

***Abstract*—**Data mining, which is the exploration of knowledge from the large set of data, generated as a result of the various data processing activities. Frequent Pattern Mining is a very important task in data mining. The previous approaches applied to generate frequent set generally adopt candidate generation and pruning techniques for the satisfaction of the desired objective. This paper shows how the different approaches achieve the objective of frequent mining along with the complexities required to perform the job. This paper will also look for hardware approach of cache coherence to improve efficiency of the above process. The process of data mining is helpful in generation of support systems that can help in Management, Bioinformatics, Biotechnology, Medical Science, Statistics, Mathematics, Banking, Networking and other Computer related applications. This paper proposes the use of both upward and downward closure property for the extraction of frequent item sets which reduces the total number of scans required for the generation of Candidate Sets.

***Keywords*—**Data Mining, Candidate Sets, Frequent Item set, Pruning.

Vaibhav Kant Singh is a Research Scholar in the Department of Computer Science & Engineering Samrat Ashok Technological Institute, Vidisha affiliated to Rajeev Gandhi Prodyogiki Vishwavidyalaya Bhopal, India. (phone; +919424174443);(e-mail:vibhu200427@gmail.com).

Vijay Shah is currently Reader in Department of Computer Science & Engineering Samrat Ashok Technological Institute, Vidisha affiliated to RGPV Bhopal, India. (e-mail:vijayvidisha@gmail.com).

Yogendra Kumar Jain is currently HOD in the Department of Computer Science & Engineering Samrat Ashok Technological Institute, Vidisha affiliated to RGPV Bhopal, India. (e-mail:ykjain_p@yahoo.co.in).

Dr. Anupam Shukla is currently Assistant Professor in the Department of Information Technology ABV-Indian Institute of Information Technology & Management Gwalior, India. (email; anupamshukla@iiitm.ac.in).

Dr. A.S. Thoke is HOD Department of Electrical Engineering National Institute of Technology Raipur, India. (email; asthoke@yahoo.co.in)

Vinay Kumar Singh is Research Scholar and also with Department of Master of Computer Science & Application Guru Ghasidas University Bilaspur India (email; vks_123123@rediffmail.com)

Chhaya Dule is Assistant Professor in CSVTU and research scholar in the Department of Computer Science and Engineering Samrat Ashok Technological Institute, Vidisha affiliated to RGPV Bhopal, India (email; chhaya_dule@rediffmail.com)

Vivek Parganiha is with the Department of Electrical Engineering National Institute of Technology Raipur, India (Mobile; +919827402322); (email; vivekparganiha@rediffmail.com)

I. INTRODUCTION

THE introduction of computing technology has significantly influenced our society and the two major impacts of this include:-

- Business Data Processing
- Scientific Computing

Due to widespread computerization and due to affordable storage facilities, there is an enormous Wealth of information embedded in huge database belonging to different Enterprises. In the Domain of Scientific, computing the Major problem is to infer some valuable Information on from observed data. The Key idea of Data Mining is to find Effective ways to combine the Computer's Power to Process Data with human eye's Ability to detect patterns. The techniques Of Data-Mining are designed for and Work best with, Large Data Sets.

Today, the Computer Processor is having speed that is underutilized due to improper localization of the various parameters if these would be properly localized than the performance of the system can be improved a lot. This can be done using several cache conscious mechanisms that are going to help in optimal use of the resources for better outcome. Together with the proposed approach, the efficiency of frequent pattern mining is going to improve by some amount.

A. Evolution of Data Mining

The Evolution of Data Mining was a result of the support of three Technologies. The three Technologies are:-

- Massive Data Collection
- High Performance Computing
- Data Mining Algorithms

Of the above three, the first two technologies have a very vital role in the advent of Data Mining, which is playing a very significant role in today's information processing environment.

Several factors have contributed to bring Data mining to the forefront. Some of the factors are:-

- Untapped value in Databases.
- Concept of Data warehousing.
- Drop in Cost/Performance ratio.

Open Science Index, Biotechnology and Bioengineering Vol:2, No:12, 2008 publications.waset.org/12669.pdf

B. *Datamining*

Data mining is the exploration and analysis of large datasets, in order to discover meaningful patterns and rules. Data mining is a component of a wider Process called (KDD) Knowledge Discovery from Database. Before a data set is mined, it first has to be cleaned. This removes, errors, ensures consistency and takes missing values into account. Data mining may use quite simple or highly sophisticated data analysis. Data mining is a component of Data ware housing but it can be a stand-alone process for data analysis, even in the absence of a Data warehouse.

C. *Data Mining V/S Knowledge Discovery From Database*

The term "Data Mining" refers to the finding of relevant and useful information from database. Data mining and knowledge discovery in the database is a new interdisciplinary field, merging ideas from statistics, machine learning and parallel computing.

Data Mining is only one of the many steps involved in the database. The KDD process tends to be highly iterative and interactive under computation.KDD is the Process of identifying a valid, potentially useful and ultimately understandable structure in data.

STAGES OF KDD

1. Selection
2. Preprocessing
3. Transformation
4. Data Mining
5. Interpretation
6. Data Visualization

D. *Data Mining Techniques*

The two fundamental goals of Data-Mining are:-

1. Prediction
2. Description

Prediction

Prediction makes use of the existing variables in the database in order to predict unknown or future values of interest.

Description

Description focuses on finding patterns describing the data and the subsequent presentation for user interpretation.

There are Several Data-Mining techniques fulfilling to the above goals. This paper mainly deals with Association. Some of them along with brief introduction are listed below:-

- *Association :-* The Presence of one set of items in a transaction implies other set of items
- *Classification: -* Develops profiles of different groups.
- *Sequential Patterns: -* Identifies sequential patterns subject to user constraints.
- *Clustering: -* Segments database into subsets or clusters.

II. THE APRIORI ALGORITHM:-

One of the first algorithms proposed for association rules mining was the AIS algorithm [1]. The problem of association rules mining was introduced in [1] as well. This algorithm was improved later to obtain the Apriori algorithm [2]. The Apriori algorithm employs the downward closure property if an item set is not frequent, any superset of it cannot be frequent either. The Apriori algorithm performs a breadth-first search in the search space by generating candidate k+1-itemsets from frequent k itemsets. The frequency of an item set is computed by counting its occurrence in each transaction.

Apriori is an influential algorithm for mining frequent itemsets for Boolean association rules. Since the Algorithm uses prior knowledge of frequent item set it has been given the name Apriori.

Apriori is an iterative level wise search Algorithm, where k-itemsets are used to explore (k+1)-itemsets. First, the set of frequents 1- itemsets is found. This set is denoted by L_1. L_1 is used to find L_2 , the set of frequent 2-itemsets , which is used to find L_3 and so on , until no more frequent k-itemsets can be found . The finding of each L_k requires one full scan of database.

There are two steps for understanding that how L_{k-1} is used to find L_k:-

1. The join step :-

To find L_k , a set of candidate k-itemsets is generated by joining L_{k-1} with itself . This set of candidates is denoted C_k.

2. The prune step:-

C_k is a superset of L_k , that is , its members may or may not be frequent , but all of the frequent k-itemsets are included in C_k .

A scan of the database to determine the count of each candidate in Ck would result in the determination of $L_k.C_k$, however, can be huge, and so this could involve heavy computation. To reduce the size of C_k , the Apriori property is used as follows .

Any (k-1)-item set that is not frequent cannot be a subset of frequent k-item set.

Hence, if (k-1) subset of a candidate k item set is not in L_{k-1} then the candidate cannot be frequent either and so can be removed from C.

III. TECHIQUES TO OVERCOME APRIORI

A. *Previous Approaches*

FP-growth [3] is a well-known algorithm that uses the FP-tree data structure to achieve a condensed representation of the database transactions and employs a divide and-conquer approach to decompose the mining problem into a set of

Open Science Index, Biotechnology and Bioengineering Vol:2, No:12, 2008 publications.waset.org/12669.pdf

smaller problems. In essence, it mines all the frequent itemsets by recursively finding all frequent itemsets in the conditional pattern base which is efficiently constructed with the help of a node link structure. A variant of FP-growth is the H-mine algorithm [4]. It uses array-based and trie-based data structures to deal with sparse and dense datasets respectively. PatriciaMine [5] employs a compressed Patricia trie to store the datasets. FPgrowth* [6] uses an array technique to reduce the FP-tree traversal time. In FP-growth based algorithms, recursive construction of the FP-tree affects the algorithm's performance.

Eclat [8] is the first algorithm to find frequent patterns by a depth-first search and it has been devised to perform well. It uses a vertical database representation and counts the item set supports using the intersection of tids. However, because of the depth-first search, pruning used in the Apriori algorithm is not applicable during the candidate itemsets generation. VIPER [9] and Mafia [10] also use the vertical database layout and the intersection to achieve a good performance. The only difference is that they use the compressed bitmaps to represent the transaction list of each item set. However, their compression scheme has limitations especially when tids are uniformly distributed. The dEclat [11] uses the vertical database representation. They store the difference of tids called diffset between a candidate k item set and its prefix k-1 frequent itemsets, instead of the tids intersection set. They compute the support by subtracting the cardinality of diffset from the support of its prefix k-1 frequent item set. This algorithm has been shown to gain significant performance improvements over Eclat[8]. However, when the database is sparse, diffset will lose its advantage over tidset.

The Mafia algorithm [10] uses vertical database layout and does intersection to achieve good performance. The search strategy of the algorithm integrates a depth-first traversal of the item set lattice with effective pruning mechanisms that significantly improve mining performance. The Mafia algorithm [10] uses vertical database layout and intersection .It uses compressed bitmaps to represent the transaction list of each item set.

The dEclat algorithm [11] makes use of the vertical database representation where each item maintains a set of transaction ids where this item is contained. They store the difference of ids, called the diffset, between the candidate item set and its prefix frequent item sets, instead of the tids intersection set. They compute the support by subtracting the cardinality of diffset from the support of its prefix frequent item set.

B. *Current Proposed RSTDB Algorithm and Use of Hardware based Cache Conscious approach*

The paper proposes an algorithm called RSTDB [15] which reduces the number of scans involved in Apriori which is implemented in C++.The Outputs are shown in Fig. 1.It is clear from the description of the working of Apriori Algorithm that the major computational challenges that Apriori was facing were Multiple Scans of Transaction database, Huge number of candidates and Tedious workload of support counting for candidates.RSTDB tries to overcome the above problem by Reducing passes, Shrinking number of candidates and facilitating support counting of candidates.RSTDB uses heuristic function which calculates the overall number of times the scanning is going to be done , before actually iteration starts , this reduces the number of passes required for frequent item set estimation. The Algorithm is a combination of both bottom up and Top down approach. The algorithm uses Heuristic function to implement upward closure property. Heuristic function has been implemented in C++ to perform the objective. Use of dynamic memory allocation property of variables in C++ has been done to make the Algorithm Efficient. The performance of Algorithm depends upon the Efficiency of the Heuristic function used. Heuristic function taken depends on two constraints the first constraint is the number of different items in the Database and the Total number of Transactions in the TDB.

RSTDB Algorithm:-

➢ STEP 1:
Calculate the size of each transaction in the Transaction Database.

➢ STEP 2:
Evaluate the transaction set having maximum size.

➢ STEP 3:
Check for the Transaction set size having frequency or support value more than the given threshold value. Set this Transaction size as the maximum value up to which scanning & candidate-generation step has to proceed. This will be the maximum value of k up to which iteration has to be done.

Value HeuFn[no. of items,TDB size] = Max k (1)

Step 4 & Step 5 iterates until k = Max k
Value of k lies between 1 and Max k.

➢ STEP 4:
Candidate-Generation
gen_cand_itemsets with the given L_{k-1} as follows

$C_k = \phi$ (2)

```
for all item set $I_1 \in L_{k-1}$ do
for all item set $I_2 \in L_{k-1}$ do
If $I_1[1]= I_2[1] \wedge I_1[2]= I_2[2] \wedge ... \wedge I_1[k-1]< I_2[k-1]$
then c= $I_1[1], I_1[2]….. I_1[k-1], I_2[k-1]$
    $C_k = C_k \cup \{ c \}$                                  ( 3 )
```

➢ STEP 5:
Candidate Set Pruning

```
Prune($C_k$)
for all $c \in C_k$
for all ( k-1 ) subsets d of c do
If $d \notin L_{k-1}$
then $C_k = C_k \backslash \{c\}$
                                                        ( 4 )
```

Here,
k is the number of passes required.
I is the item set present in TDB.

Open Science Index, Biotechnology and Bioengineering Vol:2, No:12, 2008 publications.waset.org/12669.pdf

L_k is a set of candidate k-item set.
C_k is a superset of L_k.

RSTBD V/S APRIORI

For Evaluating the performance difference between the Apriori Algorithm and RSTDB we will be considering the example of the below transaction database. Consider the below database having five elements A, B, C, D, E.In the below table there are 10 transactions. We will mathematically show as to the difference between the two approaches:-

TABLE I
TDB1 CONSISTING OF A, B, C, D, E

Transaction ID	Items
100	A
101	B
102	C
103	D
104	E
105	B
106	B
107	A
108	B
109	C

The two Algorithms for the Above Transaction Database show slight difference in terms of utilization of Memory and execution time for different set of Threshold values. For the TDB the two approaches RSTDB and Apriori give the following results in terms of Space and execution time shown by graphs of Fig. 2 and Fig. 3 calculated mathematically [14].

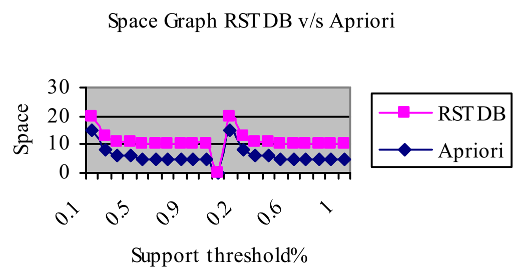


Fig. 2 Space complexity Graph for TDB1

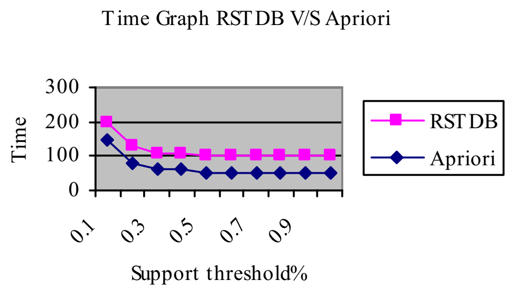


Fig. 3 Time Complexity Graph for TDB1

Limitations of RSTDB

1. It does not work for all conditions as it depends on the heuristic function
2. The increased efficiency is very less.
3. Overhead is associated with heuristic function evaluation.

Result and discussion

1. The Proposed algorithm depends on the heuristic function.
2. It is more efficient for lower threshold values.
3. It depends both on the number of different items and on total number of transactions.

Cache Conscious Optimizations

There are several mechanisms for improving the efficiency of frequent pattern mining using cache conscious optimizations [13]. Some of them are-

1. Prefix Trees
2. FP-Growth Algorithm
3. Spatial Locality Related Enhancements
4. Prefetching
5. Temporal Locality Related Enhancements.

1. Prefix Trees

A prefix tree is a data structure that provides a compact representation of transaction data set [13]. Each node of the tree stores an item label and a count, with the count representing the number of transactions, which contain all the items in the path from the root node to the current node.

2. FP-Growth Algorithm

The FP-growth algorithm [3] is one of the fastest approaches for frequent item set mining. The FP-growth algorithm [3] uses the FP-tree data structure to achieve a condensed representation of the database transaction and employees a divide-and-conquer approach to decompose the mining problem into a set of smaller problems. In essence, it mines all the frequent itemsets by recursively finding all frequent itemsets in the conditional pattern base which is efficiently constructed with the help of a node link structure.

.

3. Spatial Locality Related Enhancements

A Cache Conscious prefix-tree a data structure designed to significantly improve cache performance through spatial locality [13]. A cache conscious prefix tree is a modified prefix tree which accommodates fast bottom up traversals and improves cache line usage.

4. Prefetching

Cache line Prefetching is a popular technique for reducing the effect of cache line misses, particularly when applications do not perform a significant amount of computation per cache line[13].

5. Temporal Locality Related Enhancements

Temporal locality states that recently accessed memory locations are likely to be accessed again in near future [13]. Cache designers assume that programs will exhibit good

Open Science Index, Biotechnology and Bioengineering Vol:2, No:12, 2008 publications.waset.org/12669.pdf

temporal locality, and store recently accessed data in the cache accordingly.Therefore,it is imperative that we find any existing temporal locality in the algorithm and restructure computation to exploit it.

6. *Results and Discussion*

This part mathematically proves that for the given set of transaction databases even if dynamic memory allocation is implemented by previous basic approach i.e. Apriori, the prefix tree approach is more efficient in terms of processor utilization due to the availability of the content in the memory

Consider Table-II consisting of records of 5 Transaction databases. Also consider Fig. 4 consisting of FP-trees of the Transaction Databases in Table-II.

The graph in Fig. 5 that has been drawn from the results obtained after comparing the FP-trees drawn in Fig. 4 , shows as to how the prefix tree approach is better in performance as compared to the previous basic approach i.e. Apriori ,in terms of memory utilization which in turn causes inefficiency in terms of Processor utilization due to weak Hit/Miss Ratio.

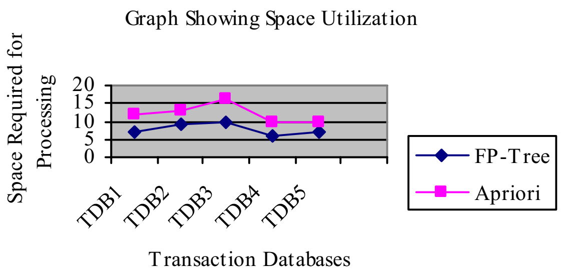


Fig. 5 Graph Representing Space Utilization for Transaction Data Bases in TABLE II.

Graph in Fig. 5 shows the performance difference between the two algorithms.

IV. CONCLUSION

Apriori is the simplest algorithm which is used for mining of frequent patterns from the transaction database. The purpose of reducing the number of scans of database to extract frequent item set has been resolved by proposed algorithm by using both upward and downward closure property

RSTDB although not that efficient increases the efficiency in frequent pattern mining by some amount. It mainly concerns with reducing the number of scans of database involved in mining process.

The Cache optimization techniques described are going to improve the Data mining system as they are going to improve Hit/Miss ratio.

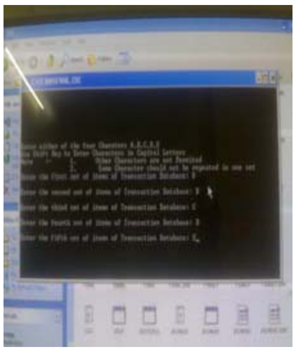
The Paper concludes that it would be very beneficial if we implement a system from combination of both the proposed approaches i.e. RSTDB using Heuristic function employing both upward and downward closure properties and Cache Coherence Techniques such as Prefix Tree for storing TDB which creates data structure that increases the Hit/Miss ratio and in turn increases efficiency of the overall frequent pattern Mining Process.

ACKNOWLEDGEMENT

Authors would like to thank Professor and Head MBA Dr. L.P.Pateria and Professor and Head Mathematics, Former Dean Faculty of Science, Founder Fellow International Academy of Physical Science (Allahabad) Dr. S.P. Singh G.G. University, Bilaspur for there helps in our research work.

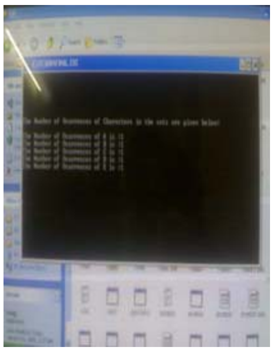
REFERENCES

[1] R. Agrawal, T. Imielinski, and A.N. Swami, "Mining association rules between sets of items in large databases," Proceedings of ACM SIGMOD International Conference on Management of Data, ACM Press, Washington DC, pp.207-216, May 1993.
[2] Mohammed J. Zaki, "Scalable Algorithms for Association Mining," IEEE Transactions on Knowledge and Data Engineering, vol.12, no. 3, pp. 372-390, May/June 2000.
[3] J. Han, J. Pei, and Y. Yin,"Mining Frequent Patterns without Candidate Generation," Proceedings of ACM SIGMOD International Conference on Management of Data, ACM Press, Dallas, Texas, pp. 1-12, May 2000.
[4] J. Pei, J. Han, H. Lu, S. Nishio, S. Tang, and D. Yang, "Hmine: Hyper-Structure Mining of Frequent Patterns in Large Databases," Proceedings of IEEE International Conference on Data Mining, pp. 441-448, 2001.
[5] Pietracaprina, and D. Zandolin, "Mining Frequent Item sets Using Patricia Tries," FIMI '03, Frequent Itemset Mining Implementations, Proceedings of the ICDM 2003 Workshop on Frequent Item set Mining Implementations, Melbourne, Florida, Dec. 2003.
[6] G. Grahne, and J. Zhu, "Efficiently using prefix-trees in mining frequent itemsets," FIMI '03, Frequent Itemset Mining Implementations, Proceedings of the ICDM 2003 Workshop on Frequent Itemset Mining Implementations, Melbourne, Florida,December 2003.
[7] Doug Burdick, Manuel Calimlim, Jason Flannick, Johannes Gehrke, "MAFIA: A Maximal Frequent Itemset Algorithm," IEEE Transactions on Knowledge and Data Engineering, vol.17, no. 11, pp. 1490-1505, Nov. 2005.
[8] M.J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li, "New algorithms for fast discovery of association rules," Proceedings of the Third International Conference on Knowledge Discovery and Data Mining, AAAI Press, pp. 283-286, 1997.
[9] P. Shenoy, J. R. Haritsa, S. Sudarshan, G. Bhalotia, M. Bawa, and D. Shah, "Turbo-charging vertical mining of large databases," Proceedings of ACM SIGMOD Intnational Conference on Management of Data, ACM Press, Dallas, Texas, pp. 22-23, May 2000.
[10] Burdick, M. Calimlim, and J. Gehrke, "MAFIA: a maximal frequent item set algorithm for transactional databases," Proceedings of International Conference on Data Engineering, Heidelberg, Germany, pp. 443-452, April 2001.
[11] M.J. Zaki, and K. Gouda, "Fast vertical mining using diffsets," Proceedings of the Nineth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington,D.C., ACM Press, New York, pp. 326-335, 2003.
[12] R. Agrawal, C. Aggarwal, and V. Prasad, "A Tree Projection Algorithm for Generation of Frequent Item Sets," Parallel and Distributed Computing, pp. 350-371, 2000.
[13] Amol Ghoting,Gregory Buehrer,Srinivasan Parthasarthy,Daehyun Kim,Anthony Nguyen,Yen-Kuang Chen and Pradeep Dubey "Cache-conscious Frequent Pattern Mining on a Modern Processor" Proceedings of the 31st VLDB Conference,Trondheim,Norway,2005.
[14] Vaibhav Kant Singh and Vijay Shah "Minimizing Space Time Complexity in Frequent Pattern Mining by Reducing Database Scanning and Using Pattern Growth Method" To be appeared in Chhattisgarh Journal of Science & Technology, Coming Volume ISSN 0973-7219.
[15] Vaibhav Kant Singh and Vinay Kumar Singh "Minimizing Space Time Complexity by RSTDB a new method for Frequent Pattern Mining" To be appeared in Proceeding of the First International Conference on Intelligent Human Computer Interaction ,Allahabad,2009.

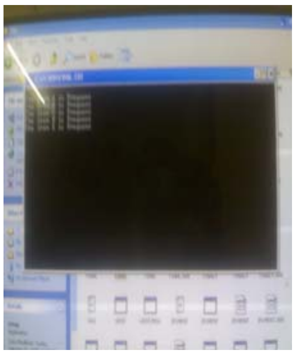


```
[illegible]
```

```
[illegible]
```



```
[illegible]
```



```
[illegible]
```

Fig. 1 The Output Windows in C++ of RSTDB Algorithm showing working of RSTDB.

TABLE II
TABLE REPRESENTING RECORD OF 5 TDB'S

TDB1		TDB2		TDB3		TDB4		TDB5	
TID	Item	TID	Item	TID	Item	TID	Item	TID	Item
T10	ABC	T20	AC	T30	ABCDE	T40	AB	T50	BC
T11	AB	T21	BCD	T31	ABC	T41	AE	T51	BCD
T12	ABE	T22	BE	T32	BCD	T42	A	T52	BE
T13	AD	T23	CE	T33	AC	T43	ADC	T53	AC
T14	BE	T24	ABCD	T34	ADE	T44	AC	T54	A

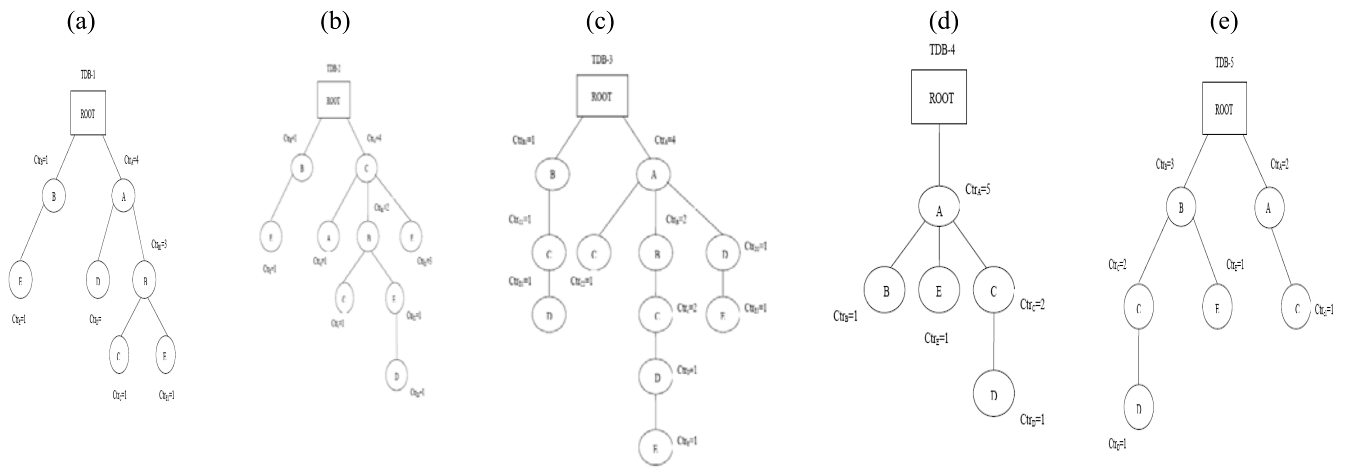


Fig. 4 Figure Representing FP-trees for the 5 TDB's in TABLE-II.(a)FP-Tree for TDB-1,(b) FP-Tree for TDB-2,(c) FP-Tree for TDB-3 ,(d) FP-Tree for TDB-4 ,(e) FP-Tree for TDB-5.

Open Science Index, Biotechnology and Bioengineering Vol:2, No:12, 2008 publications.waset.org/12669.pdf