ISSN: 2321-7782 (Online)
Volume 2, Issue 7, July 2014
International Journal of Advance Research in Computer Science and Management Studies
Research Article / Survey Paper / Case Study
Available online at: www.ijarcsms.com

BDAG- A Proficient Approach with Closed Frequent Pattern

Prof. C. Suganya[1]
Dept. of Computer Engineering
BDCE, Sevagram
Maharashtra – India

Prof. Dezy B.Wankhede[2]
Dept. of Computer Engineering
BDCE, Sevagram
Maharashtra – India

Abstract: Bidirectional Directed Acyclic Graph (BDAG) is an enhanced method of UDDAG algorithm [1]by including closed frequent pattern. For that closed frequent pattern growth algorithm has been implemented using Closed FP-tree, for storing compressed frequency information. An efficient FP-array technique greatly reduces the need to traverse FP-trees, as a result obtaining significantly improved performance for FP-tree based algorithms. Additionally a very effective closed frequent pattern[2] method is used for variation of the FP-tree data structure in combination with the FP-array technique efficiently.

BDAG approach is for detecting efficient pattern mining. This approach detects patterns from both ends ie. prefixes and suffixes due to which frequent pattern has been found with less levels of database projection compared to traditional approaches. This will improve the time efficiency and memory usage is less than Spade[4] and LapinSpam[4].

Keywords: Data mining algorithm, Up Down Directed Acyclic Graph, Closed frequent pattern, FP-array technique, FP-tree.

I. INTRODUCTION

Sequential pattern mining is very important to identify which pattern or combination of items is mostly used in data mining. But traditional approaches gave redundant data which result in time complexity and huge memory usage. To overcome the redundant problem closed frequent pattern mining algorithm is incorporated in this system. Closed sequential pattern [3] mining eliminates the redundant patterns in sequential pattern. BDAG approach finds complete set of pattern with minimum support. It is highly efficient, scalable with little amount of database scan.

II. PROBLEM STATEMENT AND RELATED WORK

Problem Statement

Let I = $\{i_1, i_2, . . . i_n\}$ be a set of items, an item set is a subset of I, denoted by $(x_1, x_2, . . . x_k)$, where $x_i \in I$; $i \in \{1, . . ., k\}$. Let S= $\{s_1, s_2, s_3 \ldots s_n\}$ be a list of itemsets where s_i is an item set, $i \in \{1, . . .,n\}$. The number of instances of item sets in S is called the length of S.

A tuple <tid; S> contains a sequence B if B< S. The sequence B in a sequence database D has absolute support as $SupD(B)=|\{<tid; S>|(B< S)\wedge(<tid; S> \in D)\}|$.The minSup value is taken as the support threshold B is called a sequential pattern in D if $SupD(B) \geq minSup$.

III. RELATED WORK

Mining closed frequent pattern

Mining closed frequent itemsets[2] is composed into two steps. (1) To sort frequent itemsets; (2) check new frequent itemset is closed.

For the construction of closed Frequent Pattern tree, a minimum support is supposed to be taken to create transaction. Usually FP-tree contains all frequent itemset, In this method Closed FP-tree is supposed to constructed due to conditional pattern base and conditional FP-tree. the constructed is shown in figure 2 which contains all closed frequent itemsets. A node x:

l: c means that the node is for item x, its level is l and its count is c. The header table in the Closed Frequent Pattern tree is the same as that of the FP-tree.

Start from the first FP-tree T in figure 1. Since T contains more than one path, a bottom-up search has to be done. First, because item g'conditional FP-tree contains only one path, we get the first frequent itemset {a, g}.Obviously it is closed, so it is inserted into the Closed FPtree directly. Similarly, for item b and e, we can get closed frequent itemsets {d, b} and {a, c, e}. For item f,construct f'conditional FP-tree (Tf), because Tf contains more than one path, Closed FP-growth is recursively called,we get closed frequent itemsets {a, d, f}, {a, f}. Similarly,we deal with item d until item a. Finally, we get Closed FP-tree as shown in figure2.

To minimize the number of frequent itemsets, maximal frequent itemset concept is used. An effective closed frequent pattern[3] mining algorithm based on FP-growth, which is named, closed FP-growth. In the algorithm, a variation of the FP-tree is used, named Closed FP-tree. In a Closed FP-tree, each node in the sub tree has four fields namely item-name, count, node link and level. Level is used for closed frequent item set testing.

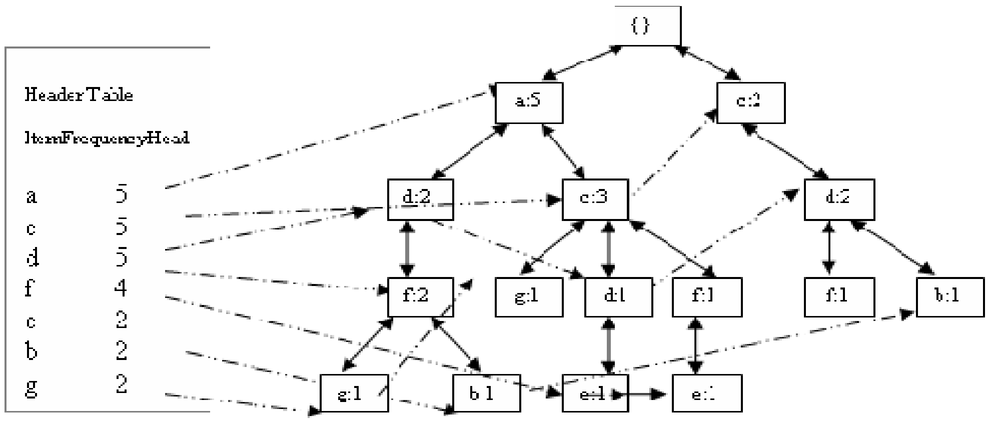


Fig. 1 FP Tree

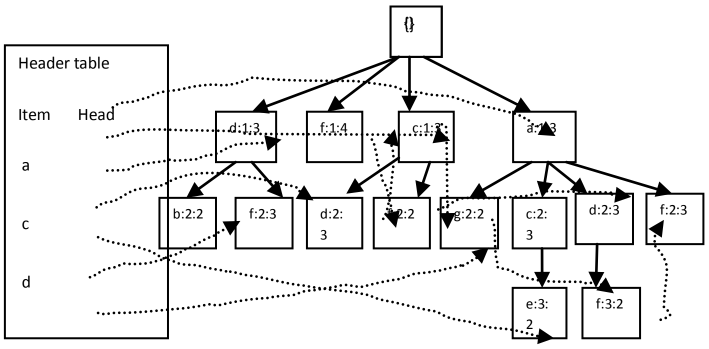


Fig. 2. Closed FP Tree

Up Down Directed Acyclic pattern

Given the following database:

1. < i d e h c f>,

2. < c i d e h c a e>,

3. < c h b d f c i>,

4. < b h d c f>,

5. < i f c>,

^{h}D is,

1. < d e h c f>,

2. <c d e h c a e >,

3. <c h b d f c>,

4. < b h d c f>.

If minSup is 2, the 8th (h) subset of patterns is

{<h>, <c h>, <d h>, <e h>, <d e h>, <h c>, <h d>, <h f>, <h c f>, <h d c>, <h d f>, <c h c>, <e h c>, <d e h c >}

Observing the patterns in the 8th subset, except for <h>, which only contains h and can be derived directly, all other patterns can be clustered and derived as follows:

1. {<c h>; <d h>; <e h>; <d e h>}, the patterns with 8 at the end. This cluster can be derived based on the prefix subsequences of h in ^{h}D, or Pre (^{h}D), which is,

1. <d e>;

2. <c d e>;

3. <c>; and

4. <b>.

By concatenating the patterns {<c>, <d>, <e>, <d e>) of Pre (^{h}D) with h, we can derive patterns in this cluster.

2. {<h c>, <h d>, <h f>, <h c f>, <h d c>, <h d f>}, the patterns with h at the beginning. This cluster can be derived based on the suffix subsequences of h in ^{h}D, or Suf(^{h}D), which is

1. <c f>;

2. <c a e>;

3. <b d f c>;

4. <d c f>.

By concatenating 8 with the patterns (<c>, <d>, <f>, <c f>, <d c>, <d f >) of Suf(^{h}D), we can derive patterns in this cluster.

3. {<c h c>, <d h c>, <e h c>, <d e h c>}, the patterns with h in between the beginning and end of each pattern. This cluster can be mined based on the patterns in Pre (^{h}D) and Suf(^{h}D). In this case, a pattern (e.g., <d h c>) can be derived by concatenating a pattern of Pre (^{h}D) (e.g., d) with the root item h and a pattern of Suf(^{h}D) (e.g., c).

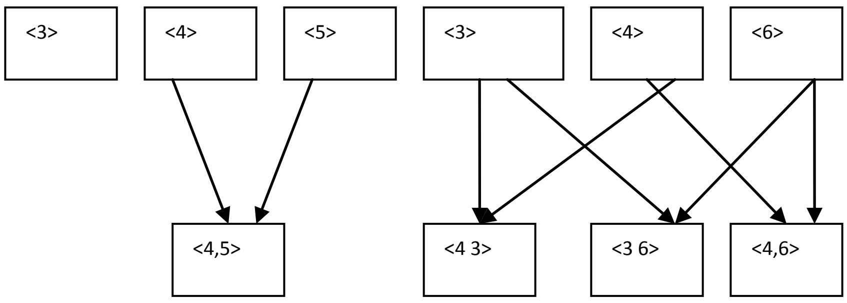


Fig.3. Up/Down DAGs of patterns

IV. SYSTEM DESIGN

Bidirectional Directed Acyclic Graph (BDAG) is the enhanced method of UDDAG algorithm by including closed frequent pattern. Mining association rule consist of two steps such as finding all the frequent itemsets and generating strong association rules. This results in huge amount of frequent itemsets. To reduce the number of frequent itemsets Multi Item Frequent Itemset MFI method is used. MFI method has some drawbacks. To overcome the deficiency of MFI , closed frequent pattern is implemented.To find all closed frequent itemsets, consists of two steps as to find frequent itemset and verify whether the new frequent itemset is closed or not.

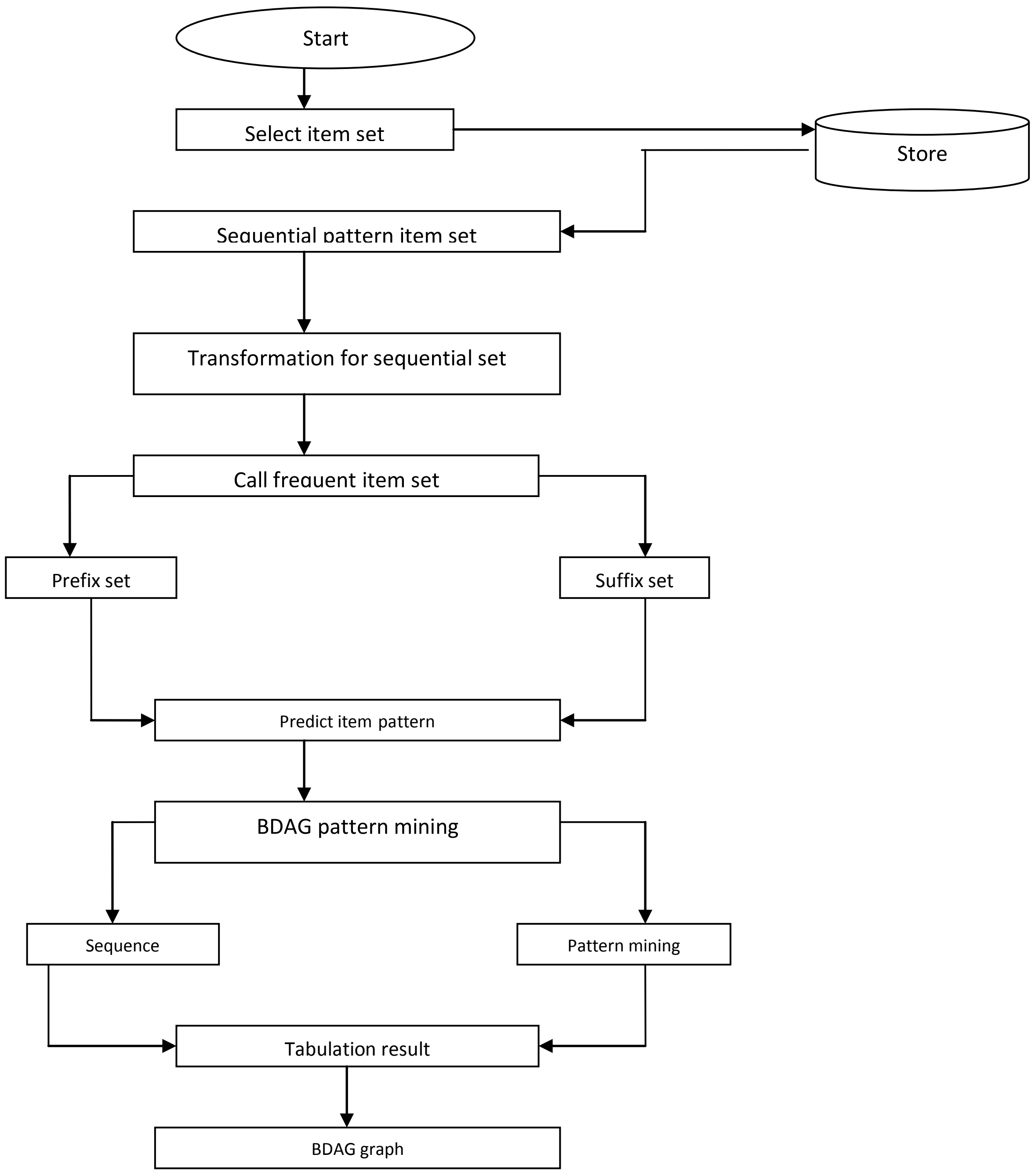


Fig.4 System Design

UDDAG pattern[1] mining approach consists of the three major steps Database Transformation, Problem Partitioning, UDDAG-Based Pattern Mining. This approach first transforms a database based on frequent item sets, then partitions the problem, and finally detects each subset using UDDAG .Closed Frequent Mining algorithm consists of the steps: Closed FP-tree construction, FP-array technique. In this approach, a new algorithm is developed to integrate Closed Frequent Mining algorithm with UDDAG Based Pattern Mining algorithm.

V. Bidirectional Directed Acyclic Graph

A. *SET THE SEQUENTIAL PATTERN*

Sequential pattern is detected from number of transaction id and number of product sequence by using minimum and maximum product in the sequential product.

Transacction_	**Sequence pattern**
10	<a(abc)(ac)d(cf)>
20	<(ad)c(bc)(ae)>
30	<(ef)(ab)(df)cb>
40	<eg(af)cbc>

Tab.1. Sequence Database

B. *DATABASE TRANSFORMATION*

In a sequence database, the absolute support for an item set can be calculated by the number of tuples whose sequences contain the item set. An item set with a support larger than minSup is called a frequent item (FI) set.

The sequence database D is transformed into an alternative representation, for which Unique id is assigned to each Frequent Itemsets FI in D. Then each itemset in each sequence with the ids of all the FIs contained in the item set is replaced.

For example, for the database in Tab.1., the FIs are: (1),(2), (3), (4), (5), (6), (1,2), (2,3). By assigning a unique id to each FI, e.g., (1)-1, (1,2)-2, (2)-3, (2,3)-4, (3)-5, (4)-6, (5)-7, (6)-8, we can transform the database as shown in Tab.2. (Infrequent items are eliminated).

Seq.Id	**Sequence**
1	<1 (1,2,3,4,5) (1,5) 6 (5,8)>
2	<(1,6) 5 (3,4,5) (1,7)>
3	<(7,8) (1,2,3) (6,8) 5 3>
4	<7 (1,8) 5 3 5>

Tab.2. Transformed Database

An item pattern is a sequential pattern with exactly 1 item in every item set it contains. Let D be a database and P be the complete set of sequential patterns in D, D0 be its transformed database, substituting the ids of each item pattern contained in D0 with the corresponding item sets, and denoting the resulted pattern set by P0, we have P ¼ P0.

The sequential data from user interaction in this table data is store into database and the proceeding the further process is continued. Then set threshold value for minimum support and set the predicting data in this value is given by user. Then the predicted data sequence in separate from the input data.

Example: 1. <9 4 5 8 3 6>;

2. <3 9 4 5 8 3 1 5>;

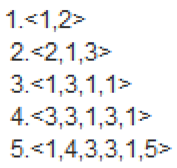
3. <3 8 2 4 6 3 9>;

4. <2 8 4 3 6>;

5. <9 6 3>,

"8 is the user given predicted data product", *Item pattern*

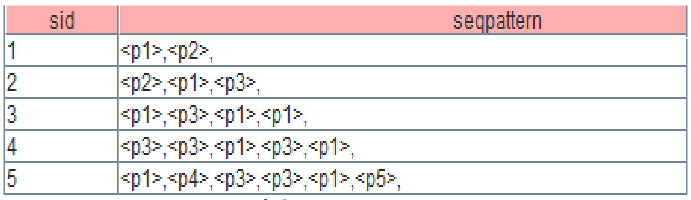
In this module the extracting product data is to be displayed and eliminate less minimum support data from sequential data set.



1.<1,2>
2.<2,1,3>
3.<1,3,1,1>
4.<3,3,1,3,1>
5.<1,4,3,3,1,5>

C. FREQUENT ITEMSET

A set of items is referred to as an itemset. An item set that contains k items is a k-item set. The set {P1, P2, and P3} is a 2-itemset. The frequency occurrence of of an itemset is considered as the number of transactions that contain the item set. This is also known as the frequency, support count, or count of the item set. The occurrence frequency is called the absolute support. If the relative support of an item set I satisfies a prespecified minimum support threshold (i.e., the absolute support of I satisfies the corresponding minimum support count threshold), then I is a frequent item set. The set of frequent k-item sets is commonly denoted by Frequent Pattern.

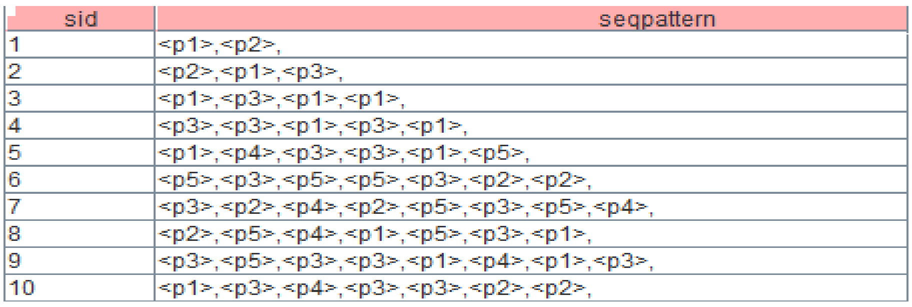


sid	seqpattern
1	<p1>,<p2>,
2	<p2>,<p1>,<p3>,
3	<p1>,<p3>,<p1>,<p1>,
4	<p3>,<p3>,<p1>,<p3>,<p1>,
5	<p1>,<p4>,<p3>,<p3>,<p1>,<p5>,

Tab.3 Frequent Itemset

A TS- Based Pattern Mining

The algorithm implementation part is started to this module already the data set's are split from sequential data set and calculate the frequent pattern support to the each single data and multiple data sequence. In this module ,used to calculating prefix and suffix tree support from the give data's because our algorithm is find unidirectional data is called prefix and suffix calculation.

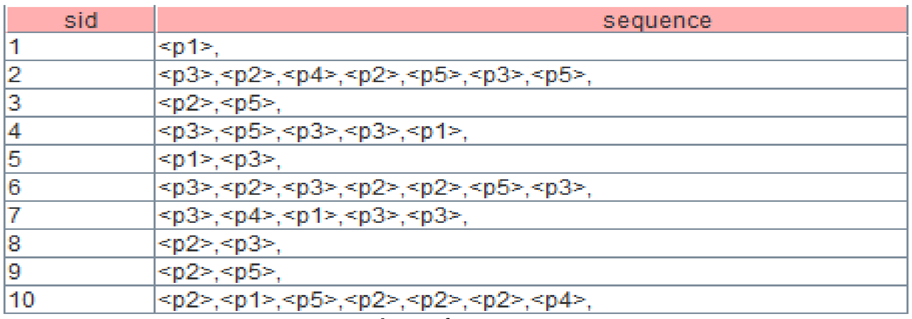


sid	seqpattern
1	<p1>,<p2>,
2	<p2>,<p1>,<p3>,
3	<p1>,<p3>,<p1>,<p1>,
4	<p3>,<p3>,<p1>,<p3>,<p1>,
5	<p1>,<p4>,<p3>,<p3>,<p1>,<p5>,
6	<p5>,<p3>,<p5>,<p5>,<p3>,<p2>,<p2>,
7	<p3>,<p2>,<p4>,<p2>,<p5>,<p3>,<p5>,<p4>,
8	<p2>,<p5>,<p4>,<p1>,<p5>,<p3>,<p1>,
9	<p3>,<p5>,<p3>,<p3>,<p1>,<p4>,<p1>,<p3>,
10	<p1>,<p3>,<p4>,<p3>,<p3>,<p2>,<p2>,

Tab.4 Sequential Pattern Itemset

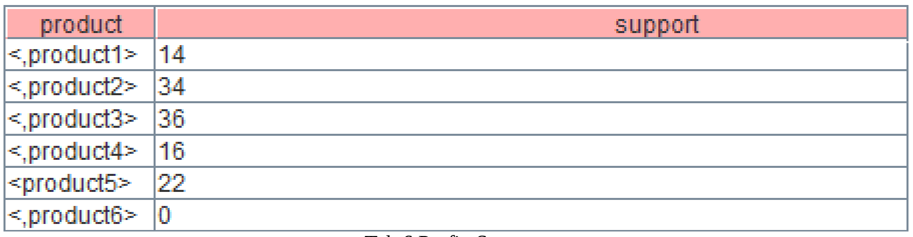
Prefix tree

In this algorithm is working into calculate the prefix value in the sequential data set. This data's are getting from above the minimum threshold level. The prefix graph fig. has generated based on the count greater than minimum support count as shown in Tab.6



sid	sequence
1	<p1>,
2	<p3>,<p2>,<p4>,<p2>,<p5>,<p3>,<p5>,
3	<p2>,<p5>,
4	<p3>,<p5>,<p3>,<p3>,<p1>,
5	<p1>,<p3>,
6	<p3>,<p2>,<p3>,<p2>,<p2>,<p5>,<p3>,
7	<p3>,<p4>,<p1>,<p3>,<p3>,
8	<p2>,<p3>,
9	<p2>,<p5>,
10	<p2>,<p1>,<p5>,<p2>,<p2>,<p2>,<p4>,

Tab.5 Prefix sequence

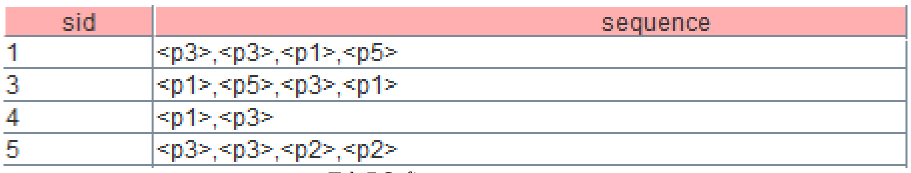


product	support
<,product1>	14
<,product2>	34
<,product3>	36
<,product4>	16
<product5>	22
<,product6>	0

Tab.6 Prefix Support

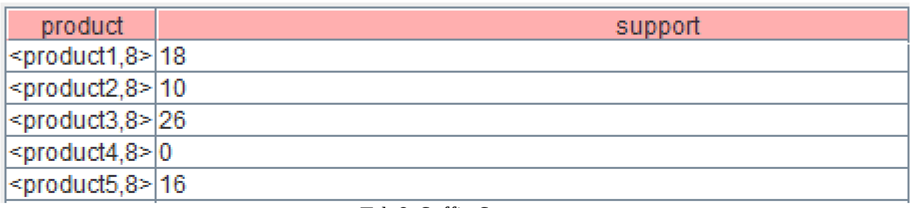
Suffix tree

This cluster can be derived based on the suffix value in the sequential data set. This data's are getting from above the minimum threshold level. The prefix graph fig. has generated based on the count greater than minimum support count as shown in Tab.7



sid	sequence
1	<p3>,<p3>,<p1>,<p5>
3	<p1>,<p5>,<p3>,<p1>
4	<p1>,<p3>
5	<p3>,<p3>,<p2>,<p2>

Tab.7 Sufix sequence



product	support
<product1,8>	18
<product2,8>	10
<product3,8>	26
<product4,8>	0
<product5,8>	16

Tab.8 Suffix Support

In PrefixSpan[12], patterns are partitioned based on common prefix and grown unidirectionally along the suffix direction of detected patterns. At each level of recursion, the length of detected patterns is only grown by 1. In this method patterns grow bidirectionally along both ends of detected patterns; the patterns may grow in parallel at each level of recursion. This new approach finds suitable method for partitioning, projection and detection strategies that allow for faster pattern growth.

To support bidirectional pattern growth, partitioning patterns is based on common root items instead of prefix[8]. Consider the transformed database with n different frequent itemset; its sequential patterns can be divided into n disjoint subsets. The i^{th} subset $(1 \leq i \leq n)$ contains the root item of the subset and items are smaller than i. To identify the ith subset, checking the subset of tuples whose sequences contain i in database D. In the ith subset, each pattern can be divided into two parts, prefix and suffix of i.

D. Closed Frequent Pattern Bidirectional Directed Acyclic Graph

Challenges on Sequential Pattern Mining

A huge numbers of possible sequential patterns are hidden in databases. A mining algorithm should find the complete set of patterns, when possible, satisfying the minimum support (frequency) threshold, be highly efficient, scalable, involving only an adequate number of database scans be able to incorporate various kinds of user-specific constraints. The transformed database has to be scanned to get support count for each candidate sequence . Candidate length-(k+1) sequences is generated from length-k frequent sequences using Apriori.

Problem Definition

Data mining from databases consists of the following important processes.

1) Creating a target data set

2) Data cleaning and preprocessing.

3) Data reduction and projection.

4) Exploratory analysis and model and hypothesis selection.

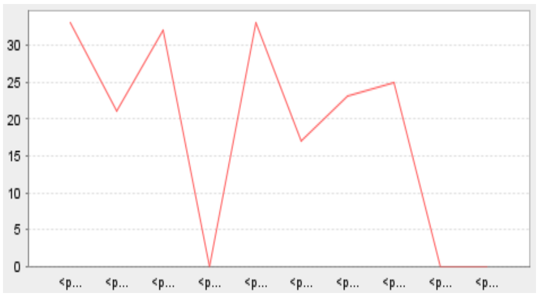


Fig. 5. Prefix Graph

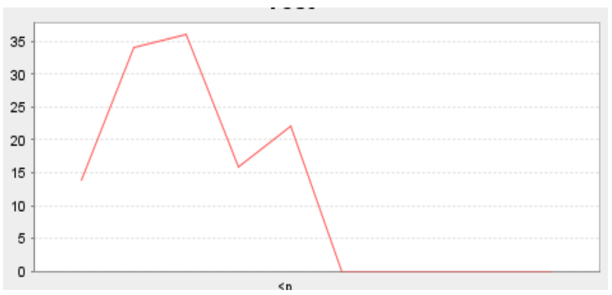


Fig.6 Suffix Graph

VI. CONCLUSION

This new method grows and detects patterns from both ends (prefixes and suffixes) of closed frequent patterns. Due to this results in faster pattern growth because of less levels of database projection compared to traditional approaches.

One major feature of BDAG is that it supports efficient pruning of invalid candidates and avoid the redundant item set. Due to which frequent itemsets are limited and easy to find frequent pattern. This represents a promising approach for applications involving searching in large spaces. Thus, it has great potential to related areas of data mining and artificial intelligence

References

1. Jinlin chen (2010) “An UpDown Directed Acyclic Graph Approach for Sequential Pattern Mining”, IEEE Transactions on Knowledge and Data Engineering, Vol.22, No.
2. Nancy P. Lin, Wei-Hua Hao, Hung-Jen chen, Hao-En Chueh and Chung-I chang (2008) "Fast Mining of Closed Sequential Patterns", WSEAS Transactions on Computers, vol. 7, no. 4, pp. 133-139
3. Yan. X, Han. J and Afshar. R (2003) "CloSpan: mining closed sequential patterns in large datasets", In Proceedings of the 3rd SIAM International Conference on data mining, pp. 166-177, San Francisco, CA.
4. Zhang.Z and Kitsuregawa.M (2005), “LAPIN-SPAM: An Improved Algorithm for Mining Sequential Pattern,” Proc. Int’l Special Workshop Databases for Next Generation Researchers, pp. 8-11.
5. Berkovich.S, Lapir.G, and Mack.M (2000) “A Bit-Counting Algorithm Using the Frequency Division Principle,” Software: Practice and Experience, vol. 30, no. 14, pp. 1531-1540.
6. Chen.J and Cook.T (2007) “Mining Contiguous Sequential Patterns from Web Logs,” Proc. World Wide Web Conf. (WWW ’07) Poster Session.
7. Chen.J and Xiao.K, “BISC: A Binary Itemset Support Counting Approach towards Efficient Frequent Itemset Mining,” to be published in ACM Trans. Knowledge Discovery in Data.
8. Grahne.G and Zhu.J (2003) “Efficiently Using Prefix-Trees in Mining Frequent Itemsets,” Proc. Workshop Frequent Itemset Mining Implementations (FIMI ’03).
9. Lin.M.Y and Lee.S.Y (2005) “Fast Discovery of Sequential Patterns through Memory Indexing and Database Partitioning,” J. Information Science and Eng., vol. 21, pp. 109 -128.
10. Masseglia F, Cathala.F and Poncelet.P (1998) “The PSP Approach for Mining Sequential Patterns,” Proc. European Symp. Principle of Data Mining and Knowledge Discovery, pp. 176-184.
11. Agrawal.R and Srikant.R (1995), “Mining Sequential Patterns,” Proc. Int’l Conf. Data Eng. (ICDE ’95), pp. 3-14.
12. Agrawal.R and Srikant.R (1994), “Fast Algorithms for Mining Association Rules,” Proc. 20th Int’l Conf. Very Large Data Bases (VLDB), pp. 487-499.

AUTHOR(S) PROFILE

C.SUGANYA received the B.E degree in Computer Science and Engineering from E.G.S.P Engineering College, Nagapattinam, Tamilnadu, India and the M.E in Computer Science and Engineering from Anna University, Trichy,Tamilnadu, India.

Presently working as Assistant Professor in Computer Engineering at B.D.C.E, RTMNU University, Sevagram, Maharashtra, India

DEZY B.WANKHEDE received the B.E degree in Computer Technology Engineering from R.C.E.R.T Engineering College, Chandrapur, Nagpur University, Maharashtra and the M.Tech in Software Engineering from Karunya University, coimbatore, Tamilnadu.

Presently working as Assistant Professor in Computer Engineering at B.D.C.E, RTMNU University, Sevagram, Maharashtra, India