International Journal For Technological Research In Engineering
Volume 1, Issue 9, May-2014 *ISSN (Online): 2347 - 4718*

FREQUENT PATTERN MINING OF CONTINUOUS DATA OVER DATA STREAMS

Aneri Patel[1] (M.E. Scholar), M.B. Chaudhari[2] (Professor and HOD)
Computer Science and Engineering Department
Government Engineering College, Sec-28, Gandhinagar, Gujarat, India.

Abstract: Frequent pattern mining is one of the important tasks used in data mining domain. Frequent pattern mining is used to find interesting patterns from databases, such as association rules, correlations rules, sequence rules, classifier rules, and cluster rules. The main goal of the association rule is, to analyze the purchased products of a customer in a supermarket transactional data. Association rule is used to describe how frequently items are purchased together. It is mainly used in transactional database. There is the efficient technique to discover the complete set of recent frequent patterns from a high-speed data stream over a sliding window. A New MOMENT algorithm with CET(Closed Enumeration Tree) to capture the recent stream data content and efficiently remove the obsolete, old stream data content. The complete set of recent frequent patterns is obtained from the CET of the current window using a transactional window sliding. Extensive experimental analyses show that our method is highly efficient in terms of memory and execution time when finding recent frequent patterns from a high-speed data stream.

Index Terms: Frequent Pattern, Data Stream, Data Mining

I. INTRODUCTION

As IT field is growing in market now days, Organizations are working with computer technologies with large database. These organization sectors include marketing, telecommunications, manufacturing, banking, transportation etc. To extract valuable data, it is necessary to explore the databases completely and efficiently. Data mining helps to identify valuable information in such huge databases.

Data mining [1] a technique that deals with the extraction of hidden predictive information large database. It uses sophisticated algorithms for the process of sorting through large amounts of data sets and picking out relevant information. Data mining tools predict future trends and behaviors, allowing businesses to make proactive, knowledge-driven decisions. With the amount of data doubling each year, more data is gathered and data mining is becoming an increasingly important tool to transform this data into information. Long process of research and product development evolved data mining. This evolution began when business data was first stored on computers, continued with improvements in data access, and more recently, generated technologies that allow users to navigate through their data in real time. Data mining takes this evolutionary process beyond retrospective data access and navigation to prospective and proactive information delivery. Data mining is ready for application in the business community because it is supported by three technologies that are now sufficiently mature:

- Massive data collection
- Powerful multiprocessor computers
- Data mining algorithms

Applications examples of data mining [2]

• Health Care
  - Drug development, Medical diagnostics, Insurance claims analysis, Bio-Informatics, Genomics etc.

• Business and Finance
  - Banks, Credit card companies, Lenders, Financial Planning, Risk Analysis, Marketing Plans, Fraud detection etc.

• Sports and Gambling
  - Gaming industry, Sports Fanatics etc.

• Education
  - Enrollment Management, Retention/Graduation Analysis, Donation Prediction Smart Infrastructures etc.

• Retail
  - Basket analysis, Sales forecasting, Merchandise planning and allocation etc.

• Other Application Areas
  - Insurance, Stock Market, Weather Prediction, Smart Infrastructures, Telecommunications, Land-use, Urban Planning etc.

Data Stream Mining is the process of extracting knowledge structures from continuous, rapid data records. A data stream is an ordered sequence of instances that in many applications of data stream mining can be read only once or a small number of times using limited computing and storage capabilities.

Data Stream Mining Application [3]

In this section we discuss some of the main data stream application domains in which concept drift plays an important role.

• Monitoring systems are characterized by large data streams which need to be analyzed in real time. Classes in these systems are usually highly imbalanced, which makes the task even more complicated.

- Network security, Telecommunications, Finance, Transportation, Industrial monitoring

• Personal assistance can include individual assistance for personal use, customer profile, and recommendation systems. The costs of mistakes are relatively low compared to other applications, and the class labels are mostly "soft". Personal assistance systems do not have to react in real time, but are usually affected by more than one source of drift.

- News feeds, Spam filtering, Recommendation systems, Economics

• Decision support with concept drift includes diagnostics and evaluation of creditworthiness. The true answer whether a decision is correct is usually delayed in these systems. Decision support and diagnostic applications typically involve limited amount of data and are not required to be made in real time. The cost of mistakes in these systems is large, thus the main challenge is high accuracy.

- Finance, Biomedical applications

• Artificial intelligence

Learning in dynamic environments is a branch of machine learning and AI where concept drift plays an important role. Ubiquitous Knowledge Discovery (UKD), which deals with mobile distributed systems such as navigation systems, vehicle monitoring, household management systems, is also prone to concept drift.

- Navigation systems, Smart homes and virtual reality

Frequent Pattern Mining [4]

Many techniques have been developed in data mining amongst which primarily, Association rule mining is very important and best-known technique which results in association rules. Association rule mining is a technique for discovering unsuspected data dependencies. It is also known identified as "Frequent pattern mining". The basic idea is to identify from a given database, consisting of item sets, whether the occurrence of specific items, implies also the occurrence of other items with a relatively high probability. In principle the answer to this question could be easily found by complete exploration of all possible dependencies, which is however prohibitively expensive. These rules are applied on market based analysis, medical applications, science and engineering, music data mining, banking etc for decision making. For example, in market based analysis, when customer buys items, some of them are having dependencies on each other. Finding these dependencies can be very helpful to market based analysis and these dependencies can be found by association rule mining in which important patterns are found. This is known as extracting pattern from database. From these patterns association rules are found.

II. RELATED WORK

"Efficient frequent pattern mining over data streams"[5] paper they proposed a prefix-tree structure called CPS-tree (Compact Pattern Stream tree) uses technique called as dynamic tree restructuring technique to handle the stream data. This tree constructs a compact-prefix tree structure with single pass scanning. For restructuring the CPS tree they used an efficient restructuring mechanism called as BSM method and path adjusting method. The main disadvantage of this algorithm is every time a new item is arrives, it reconstructs the tree. So it causes more memory space and time. "Mining frequent item sets in data streams using the weighted sliding window model" [6] paper the author proposed a new technique called the weighted sliding window WSW algorithm. This model allows the user to specify the number of windows for mining, the size of the window and the weight each window. Using this algorithm the user can specify minimum weighted threshold value. They split the transaction into equal number of windows. Using the WSW algorithm they calculate the weight of each transaction in each window. If the weighted support count of an item is greater than or equal to minimum weighted threshold value it is called as frequent item set. The main advantage of this algorithm is it scanned the database only once. It does not take more than one scan to find out the frequent item set. When the window size increases, then the execution time of WSW decreases. This is because when the window size small the number of transaction containing frequent item sets in each window is small. Therefore the probability of choosing a candidate item set to be not a frequent item set of early windows is high. "Mining frequent item sets with normalized weight in continuous data streams" [7] proposed an efficient algorithm WSFI mine (Weighted Support Frequent Item sets mining) with normalized weight over data stream. The proposed WSFI-mine method is designed to mine all frequent item sets from one scan in the data stream. This algorithm uses three phases. WSFP-tree structure is an extended form of FP-tree growth technique. At last phase the WSFI-mine method discovers frequent item sets. A currently infrequent pattern may become frequent in the future. Therefore the author has to be careful not to prune infrequent item sets too early. There are lots of confusion arises after the pruning. In future we should discover some new techniques to avoid this confusion at the time of pruning. "An Efficient Algorithm for Mining Closed Weighted Frequent Pattern over Data Streams"[8] Weighted frequent pattern mining is suggested to discover more important frequent pattern by considering different weights of each item, and closed frequent pattern mining can reduces the number of frequent patterns and keep sufficient result information. In this paper, we propose an efficient algorithm DS _ CWFP to mine closed weighted frequent pattern mining over data streams. We present an efficient algorithm based on sliding window and can discover closed weighted frequent pattern from the recent data. A new efficient DS_ CWFP data structure is used to dynamically maintain the information of transactions and also maintain the closed weighted frequent patterns has been found in the current sliding window. The new combined DS_CWFP structure is to dynamically maintain the information of transactions in the current sliding window. And we describe three optimization strategies taken at different stages in order to improve the efficiency of the implementation of the algorithm. Then the framework and main processes of algorithm DS CWFP are described in detail. Experiments prove the efficiency and effectiveness of our algorithm. But in this algorithm there is the issue of time complexity.

III. PROBLEM IDENTIFICATION

Finding frequent patterns from data streams has become one of the important and challenging problems, since capturing the stream content memory efficiently with a single-pass and efficient mining have been major issues. The FP-growth mining technique is one of the efficient algorithms where the achieved performance gain is mainly based on the highly compact frequency-descending FP-tree structure that ensures the tree to maintain as much prefix sharing as possible. However, the two database scans and prior threshold knowledge requirements of the FP-tree restrict its use in data stream. DSTree uses the FPgrowth mining technique to mine exact set of recent frequent patterns from stream data with a single-pass. However, it provides poor compactness in tree structure and inefficient mining phase, since it uses frequency-independent canonical order tree structure. The CP tree (Compact Pattern tree) uses a new technique called as dynamic tree restructuring technique to handle the stream data. This tree constructs a compact-prefix tree structure with single pass scanning. Its performance is as same as FP tree growth technique. [9] After creating the CP tree we can refresh the tree at each window. For restructuring the CP tree they used an efficient restructuring mechanism called as Branch sorting method (BSM) [10] and path adjusting method. Once the CP tree is constructed current window the algorithm uses bottom up technique to generate exact set of recent frequent patterns. This prefix-tree structure called CP-tree that introduces dynamic tree restructuring mechanism in data stream and find recent frequent patterns. The main disadvantage of this algorithm is every time a new item is arrives, it reconstructs the tree. It is the tree structure for single pass frequent pattern mining.

IV. PROPOSED METHODOLOGY

There is some problem in CP-tree that is it reconstructs the tree every time when a new item is arrives. So it causes more memory space as well as time. So my proposed algorithm is NewMOMENT algorithm that is use for finding frequent pattern from data stream. This algorithm uses in-memory data structure, the closed enumeration tree to monitor a dynamically selected small set of item set at any time. The closed enumeration tree (CET), which monitors closed frequent item sets as well as item sets that form the boundary between the closed frequent item sets and the rest of the item sets. CET maintains the boundary between closed frequent item set and rest of item set, which makes the boundary relatively stable, whenever any item set changes its state (frequent to non-frequent vice-versa), ultimately reducing the updating cost. An efficient algorithm to incrementally update the CET, which update the CET when newly arrived transaction change the content of window or oldest transaction being deleted from the window. [11]

When a transaction arrives/expires, Moment traverse the part of CET related to that transaction. For each node visited, Moment, update the CET by incrementing/decrementing its frequency. The merit of Moment is that it computes the exact set of closed frequent itemset over a sliding window. Although an update to a node may result in a propagation of the node insertion and deletion in the CET, most of the nodes related to an incoming or expiring transaction do not change their type often. Therefore, the average update cost in the

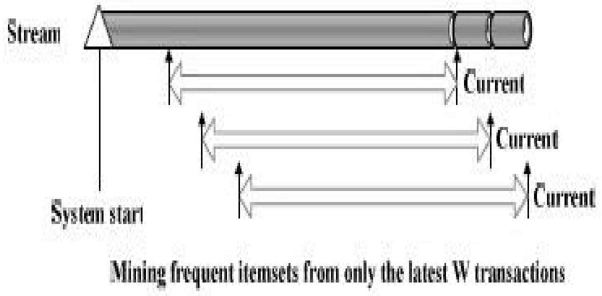
Stream
Current
Current
Current
System start
Mining frequent itemsets from only the latest W transactions

Fig. 1: Mining Frequent Item set for CET

CET is small. This algorithm is the work for incremental mining of closed frequent itemset over a data streams sliding window. It performs an exact mining of the set of frequent closed itemset, using an update per transaction policy of window method. A transaction data stream is a sequence of incoming transactions and an excerpt of this stream is called window. A sliding window can be either time-sensitive or transaction-sensitive. A time-sensitive window consists of a sequence of a fixed length of time units. The window slides forward for every time unit. A transaction-sensitive window consists of a sequence of batches. The window slides forward for an every number of transactions equal to the batch size.

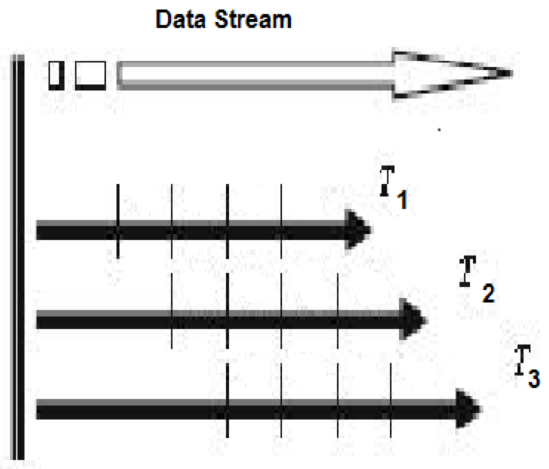

Fig. 2: Sliding Window model

Window Sliding

The window sliding phase consists of new pane addition and old pane deletion. A new pane, composed of Tidsets of items, is constructed and updated by arrival of a new transaction. After arrival of a number of transactions equal to the pane size, the set of Tidsets is ready for updating the prefix tree. This process includes updating of old itemset stored in the prefix tree, removing of insignificant itemsets from the tree and also insertion of newly identified significant itemsets to the prefix tree. [12]

Algorithm
Input: - Stream_data, Window_Size, Support, Item_Size
Output: - No. Of frequent itemset

I/P File:- .txt file, .ascii file
O/P File:- .txt file

1. Construct a FP-Tree.
2. First window sliding.
3. Explore Nodes.
4. Update proposed tree.
5. Adding a next transaction.
6. Deleting first old transaction.
7. Second window sliding.

Algorithm for Explore nodes

```
1. check support of node.
2. If support < minimum support
   assign it as infrequent node.
3. else if support > minimum support
   assign it as frequent node.
4. else create new child.
5. compute support and explore new child, database and
   mininmum support.
6. insert frequent node into table.
```

Algorithm for Adding nodes

```
Addition (TId, Itemset, FP-Tree, min_sup)
1. traverse the parts of the tree that are related to
   transaction T.
2. insert new items propogated down.
3. find the location in FP.
4. update support, tid_sum.
5. add the item to the end of currentPrefix.
6. check node type.
7. if node become infrequent to frequent build the
   occurrence lists
8. Recursively update each child.
9. call explore.
```

Algorithm for Deleting nodes

```
Deletion (TId, Itemset, FP-Tree)
1. traverse the parts of the tree that are related to
   transaction T.
2. remove the parentItemset components that were
   infrequent in the above level and copy the itemset to
   pass down to new Itemset.
3. update the support and tidsum of corresponding
   items
4. recount the support of node's children
5. recursively update the children of each family.
6. remove old value from the table.
7. if exist nodes to be removed because it became
   infrequent at parent's level.
```

V. EXPERIMENTAL ANALYSIS

I present the results of my comprehensive experimental analysis of the performance of the proposed new MOMENT algorithm for real datasets. Table 5.1 provides some statistical information about the dataset used in the experimental analyses. A dense dataset, in contrast, has many items per transaction with few distinct items. Mushroom and Connect are real-world datasets, has been used extensively in the AI area. Each record in the dataset is in standard format such as customer id, transaction id, no. of items, and list of items.

Database	**No.of Items**	**Avg. length**	**Max length**	**#Records**	**Window size**
Connect	129	43	43	67,557	1000
Mushroom	120	23	23	8124	1000

Table. 1: Dataset Characteristics

The program is written in Microsoft Visual C++ and run with Windows 7 on a 1 GHz CPU with 4 GB memory. Runtime specifies CPU and I/Os and includes, unless otherwise specified, tree construction, and mining time. The results shown in this section are based on the average of multiple runs for each case.

- Mushroom Dataset

Min_Sup(%)	**Frequent Itemset No.**	**CET Node**	**Explore call**	**Added node**	**Deleted node**
35	255	2912	0.54	7.01	7.75
30	359	4338	0.64	7.91	9.06
20	635	10356	0.87	19.37	16.4

Table. 2: Data output result for Mushroom dataset

- Connect Dataset

Min_Sup(%)	Frequent ItemsetNo.	CET Node	Explore call	Added node	Deleted node
99	41	1698	5.9	101.54	104.01
95	223	6699	7.31	159.81	146.50
90	577	14783	14.36	343.52	284.06

Table. 3: Data output result for Connect dataset

For each data set we have used one sliding-window sizes and different support value. And another case is for each data set we have used one support value and different window size.

- Performance under different minimum support

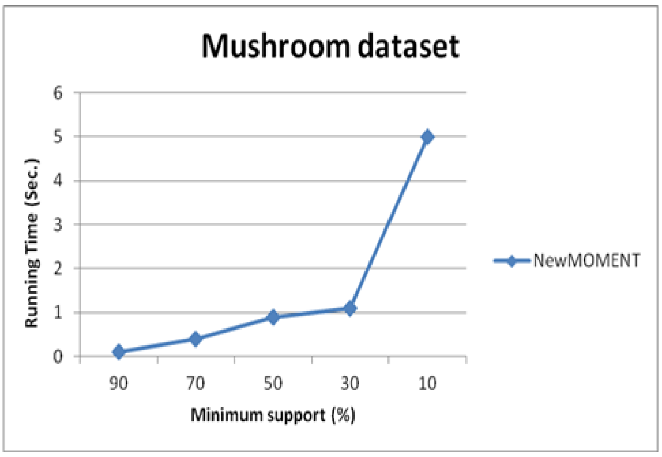

Fig. 3: Graph for compare the support and time for Mushroom dataset

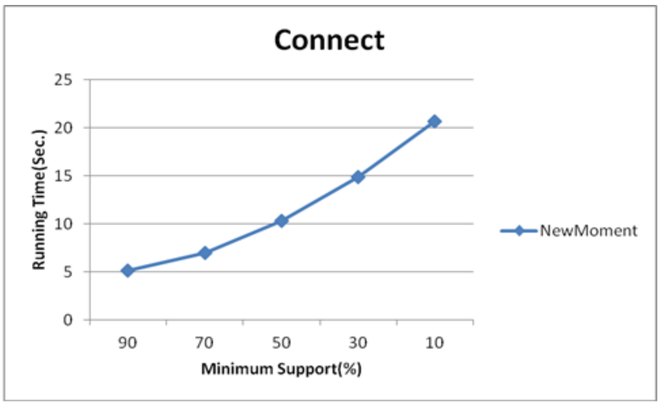

Fig. 4: Graph for compare the support and time for Connect dataset

Here these two graphs show the results of support and execution time for two datasets mushroom and connect. In both dataset when support decreases execution time will be increase.

- Performance under different Window size

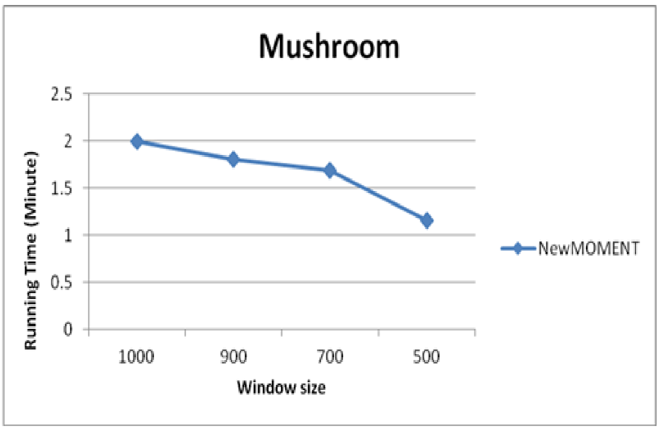

Fig. 5: Graph for compare the Window size and time for Mushroom dataset

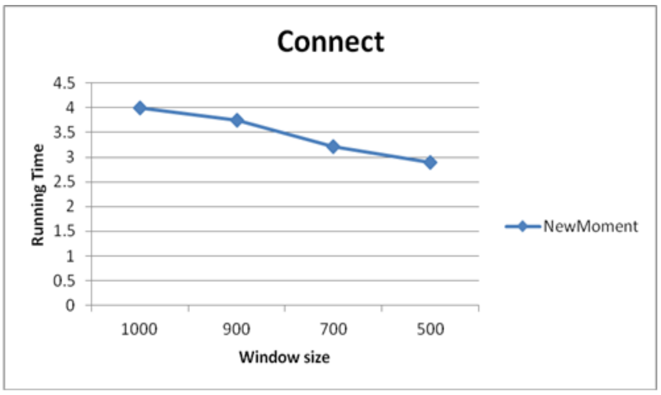

Fig. 6: Graph for compare the Window size and time for Connect dataset

Here these two graphs show the results of window size and execution time for two datasets mushroom and connect. In both dataset when window size decreases execution time will also be decrease.

- Comparison between different methods

Here it shows the different results among three methods data stream tree (DS- tree), Compact Pattern tree (CP- tree) and our proposed algorithm that is New Moment method. This is the result shows for support vs. time and for the two dataset Mushroom and Connect. In both the dataset it shows that it takes very less execution time compare to the other methods. So the method for finding frequent pattern from the data streams is little bit fast compare to the other methods. As it shows the above graphs when window size decrease it also decrease the running time and when support increase it increase the running time for both the dataset.

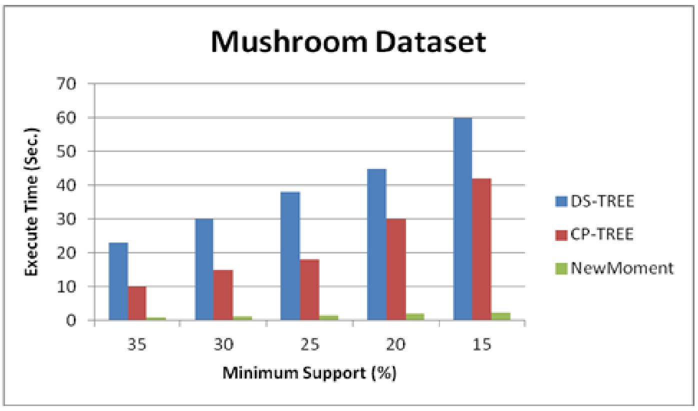

Fig. 7: Graph for compare between different methods for Mushroom

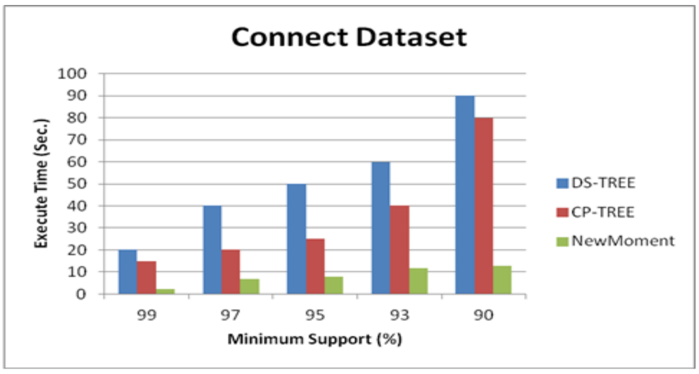

Fig. 8: Graph for compare between different methods for Connect dataset

VI. CONCLUSION

Data Stream Mining is the process of extracting knowledge structures from continuous, high speed and rapid data records. Data streams include computer network traffic, phone conversations, ATM transactions, web searches, and sensor data etc. From all this data finding frequent pattern is very complex work and so that I have try to find frequent pattern data from the stream of data. I have proposed NewMOMENT algorithm for discover and maintain all the frequent itemset using the sliding window method. The merit of NewMoment is that it computes the exact set of closed frequent itemset over a sliding window. And in this method it reduces the time when window size decreases and it is very efficient method for data stream.

REFERENCES

[1] Data mining Concepts and Techniques by Jiawei Han and Micheline Kamber

[2] “Data Mining Technologies and Decision Support Systems for Business and Scientific Applications” http://www.inf.uni-konstanz.de/dbis/teaching/ws0607/busintelligence/papers/ DMDW

[3] MINING DATA STREAMS WITH CONCEPT DRIFT, Poznan University of Technology http://www.cs.put.poznan.pl/dbrzezinski/publications/ConceptDrift.pdf

[4] Kumar,” Association analysis: basic concepts and algorithms” http://www-users.cs.umn.edu/~kumar/dmbook/ch6.pdf

[5] Syed Khairuzzaman Tabeer, Chowdary Farha ahmed, Byeong-Soo Jeong, Young Koo Lee “Efficient frequent pattern mining over data streams”, Elsevier publication, 2008

[6] Pauray S.M.Tsai “Mining frequent item sets in data streams using the weighted sliding window model”, Elsevier publication, 2009

[7] Yo-unghee Kim, Won Young Kim and Ungmo Kim “Mining frequent item sets with normalized weight in continuous data streams” Journal of information processing systems, 2010

[8] Wang Jie School of Management, Capital Normal University Beijing 100089, China "An Efficient Algorithm for Mining Closed Weighted Frequent Pattern over Data Streams", IEEE, 2012

[9] S. In Proc. Of PAKDD,Lect Notes Artif Int, 1022-1027, “CPtree: a tree structure for single-pass frequent pattern mining” 2012

[10] Koh, J.-L., and Shieh, S.-F. 2004. “An efficient approach for maintaining association rules based on adjusting FP-tree structures”. In Lee Y-J, Li J, Whang K-Y, Lee D (eds) Proc. of DASFAA 2004. Springer-Verlag, Berlin Heidelberg New York, 417–424

[11] Vikas Kumar, Sangita Satapathy Department of Computer Science and engineering, “A Review on Algorithms for Mining Frequent Item set Over Data Stream”, IJARCSSE, 2013

[12] MHMOOD DEYPIR, JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 29, “An Efficient Sliding Window Based Algorithm for Adaptive Frequent Item set Mining over Data Streams”, 2013

[13] Xun Zhu1, Hongtao Deng2 School of Mathematics & Computer Science Jianghan University Wuhan, China, "A Brief Review on Frequent Pattern Mining ", IEEE, 2011