Journal of Innovative Engineering and Research (JIER)
Vol.- 1,Issue - 1, April 2018, pp. 37-40 (4 pages)

Mining Frequent Item Sets over Dynamic Data Streams by Efficiently Removing Aged Transaction from Current Sliding Window

DEEPIKA PATHAK[1], DHANRAJ VERMA[2]

[1]*Department of Computer Science & Engineering, RKDF University, Bhopal*
[2]*Department of Computer Science & Engineering, Dr. A. P. J. Abdul Kalam University, Indore*
Corresponding Author Email: dhanrajmtech@gmail.com

Abstract— Frequent pattern mining is a core data mining operation and has extensively studied over the last decades. Mining frequent itemsets in data stream applications is beneficial for a number of purposes such as knowledge discovery, trend learning, fraud detection, transaction prediction and estimation [1,3,8,9]. In data streams, new data are continuously coming as time advances. It is costly even impossible to store all streaming data received so far due to the memory constraint. It is assumed that the stream can only be scanned once and hence if an item is passed, it cannot be revisited, unless it is stored in main memory. Storing large parts of the stream, however, is not possible because the amount of data passing by is typically huge [5, 6, 8, 18]. In this paper, we introduce a new approach which makes two major contributions. First the proposed algorithms is a single pass algorithm for mining frequent item set second it does not require any out of core structure . The proposed method is also efficient in term of memory problem of finding frequent items in a continuous stream of items. A new frequency measure is introduced, based on a variable window length. We study the properties of the new method, and propose an incremental algorithm that allows producing the frequent itemsets immediately at any time. In our method, we used multiple segments for handling different size of windows. By storing these segments in a data structure, the usage of memory can be optimized. Our experiments show that our algorithm performs much better in optimizing memory usage and mining only the most recent patterns in very less time.

Index Terms— Incremental Algorithm, Optimizing Memory, Data Mining.

I. INTRODUCTION

A data stream is an ordered sequence of items that arrives in timely order. Different from data in traditional static databases, data streams [1,9,8] are continuous, unbounded, usually come with high speed and have a data distribution that often changes with time The characteristic of continually arriving data points introduces an important property of data streams which also poses the greatest challenge: the size of a data stream is unbounded. This leads to the following requirements for data stream processing algorithms [2, 4, 20, 21]

• Bounded storage: The algorithm can only store a very limited amount of data to summarize the data stream.

Single pass: The incoming data points cannot be permanently stored and need to be processed at once in the arriving order.

• Real-time: The algorithm has to process data points on average at least as fast as the arriving data.

• Concept drift: The algorithm has be able to deal with a data generation process which evolves over time (e.g., distributions change or new structure in the data appears).

II. RELATED WORK / PREVIOUS WORK

In the year 2008, Syed Khairuzzaman Tanbeer, Choudary Farhan Ahmad, Byeong-Soo Jeong, Young-Koo Lee had proposed a research paper "Efficient frequent pattern mining over data streams". In this paper they proposed a prefix-tree structure called CPS-tree (Compact Pattern Stream tree). The CPS 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. Proposed prefix-tree structure called CPS-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. So it causes more memory space as well as time.[7,10,11,19]

In the year 2009 Pauray S.M. Tsai proposed research paper "Mining frequent item sets in data streams using the weighted sliding window model". In this 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. Using the Apriori algorithm the user can generate the candidate item set also. When a candidate item set is generated we can determine whether it is frequent or not by using the WSW algorithm. 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.

Journal of Innovative Engineering and Research (JIER)
Vol.- 1,Issue - 1, April 2018, pp. 37-40 (4 pages)

When the window size increases, then the execution time of WSW decreases [5, 7, 10, 12].

In the year 2010, Yo Unghee Kim, Won young Kim and Ungmo Kim had proposed a research paper called "Mining frequent item sets with normalized weight in continuous data streams". They 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. In the first phase the data stream is divided into 3 categories such as frequent items, latent items, and infrequent items. In second phase the author present a novel tree structure, called WSFP-tree that stores compressed crucial information about frequent item sets. WSFP-tree structure is an extended form of FP-tree growth technique. At last phase the WSFI-mine method discovers frequent item sets. This WSFI-mine algorithm can mine all frequent item sets in one scan from the database. Data changes usually with time in the data stream. A currently infrequent pattern may become frequent in the future. Therefore be careful not to prune infrequent item sets too early. This is the main advantage of this algorithm [16,21, 22, 23],

In the year 2010 Chowdhury Farhan Ahmed, Syed Khairuzzaman Tanbeer and Byeong-Soo Jeong had proposed a research paper called "Efficient mining of high utility patterns over data stream with a sliding window model". In this paper they proposed a novel algorithm for sliding window based high utility pattern mining over data stream called as HUPMS (High Utility Pattern Mining in Stream data). This paper provides a novel algorithm for sliding window based high utility pattern mining over data stream. This HUMPS algorithm can capture only the recent change of knowledge in a data stream by using novel tree structure. It is easy to construct and maintain the tree during the sliding window-based stream data mining. It uses the property called build once mine many. This is only suitable for interactive mining.

In the year 2011 Jing Guo,Peng Zhang,jianlong Tan and Li Guo discussed how to mine frequent patterns across multiple data streams called "Mining frequent patterns across multiple data streams". In this paper they selected real time news paper data for analyzing. In multiple streams it is important to discover collaborative frequent patterns and comparative frequent patterns. There are lots of frequent item set mining algorithms have been proposed. But they are mined only single stream data. In many real world applications stream data are generated from multiple sources. So it is necessary to combine multiple data stream for mining. This algorithm is proposed to discover comparative and collaborative frequent patterns [13, 14, 15, 19].

III. BASIC CONCEPTS

A data stream DS can be defined as infinite sequence of transactions, i.e. DS = [t1, t2,…, tm], i [1, m] where ti is the ith arrived transaction. A window W can be referred to as a set of all transactions between the ith and jth arrival of transactions, where j > i and the size of W is |W|= j– i, means the number of transactions between ith and jth arrival of transactions [11, 16, 17,]

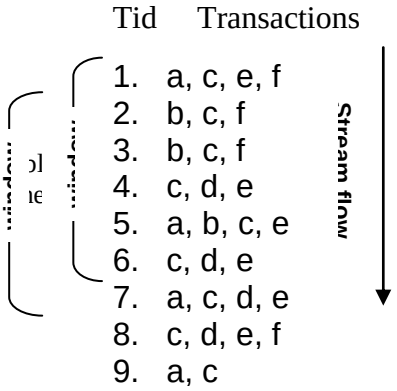

Table 1: Transactional database, where Window size |W| =8.

IV. PROPOSED METHOD

Consider a simple transactional data base given table one suppose minimum support threshold is 0.4 Frequent item set for SW1

{(c), (d), (e), (f), (c d),(c e), (c f), (d e), (c d e)} Frequent item set for SW2

{(c), (d), (e), (c d), (c e), (d e), (c d e)}

We can find that (f) and (c, f) and are frequent itemsets in SW1, but not frequent ones in SW2. We transform of transactions into bit stream sequence. Now we generate frequent pattern for CSW1 which consists of eight transactions form T1 to T8 Transforming Bit stream sequences are

One Itemset	Bit Stream sequence	Support Count
a	10001010	3
b	01101000	3
c	11111111	8
d	00010111	4
e	10011111	6
f	11100001	3

Table 2

One frequent item set { (c),(d) ,(e) ,(f) } For two candidate item set perform logical AND operation on bit stream sequence of the one frequent itemset

Two Itemset	Bit Stream sequence after performing logical AND operation	Support count
c d	00010111	4
c e	10011111	6
c f	11100001	4
d e	00010111	4
d f	00000001	1
e f	10000001	2

Table 3

Two frequent itemset are {(c d), (c e), (c f), (d e)}.

Journal of Innovative Engineering and Research (JIER)
Vol.- 1,Issue - 1, April 2018, pp. 37-40 (4 pages)

For three candidates item set perform logical AND operation on bit stream sequence of the two frequent item.

Three Itemset	Bit Stream sequence after performing logical AND operation	Support count
c d e	00010111	4
c d f	00000001	1
c e f	10000001	2

Table 4

Three frequent itemset are {(c d e)} Now we generate frequent pattern for SW2. SW2 consists of eight transaction form T2 to T9 transactions. Transforming into Bit stream sequence by using left shift bit wise operation. We have to only consider the last transaction T9.

One Itemset	Bit Stream sequence	Support count
a	00010101	3
b	11010000	3
c	11111111	8
d	00101110	4
e	00111110	6

Table 5

Frequent one item set {(c), (d), (e)}For two candidate item set perform logical AND operation on bit stream sequence of the one frequent itemset

One Itemset	Bit Stream sequence after performing logical AND operation	Support count
c d	01011100	4
c e	00111110	5
d e	00101110	4

Table 6

Two frequent itemset are {(c d),(c e),(d e)} For three candidate item set perform logical AND operation on bit stream sequence of the two frequent itemset Candidate 3-itemsets Frequent one item set {(c d e)}

One Itemset	Bit Stream sequence after performing logical AND operation	Support Count
c d e	01011100	4

Table 7

V. PROPOSED ALGORITHM

Algorithm is divided into two parts in first part we converted
Input: TDS (a transaction data stream), s (a user-defined minimum support threshold, and w (the user-specified sliding window size).
Output: a set of frequent itemsets, FI-Output.

Part 1
Begin
CSW = initialization
/*CSW consists of w transactions */
Repeat:
for each incoming transaction Ti in CSW do
for each item X in Ti do
Do bit stream -sequence transform(X);
end for
if CSW = FULL then
Do shift bit stream sequence by one position to the left of all items in CSW;
end if
end for
for each bit stream-sequence Bit(X) in CSW do
if sup(X) ≤ 0 then
Drop X from CSW;
end if
end for
part 2
The following is the frequent itemsets generation phase. The phase is performed only when requested by users
FI1 = {frequent 1-itemsets};
for (k=2; $FI_{k-1} \neq$ NULL; k++) do
Do bitwise AND to find the supports of CI_k;
for each candidate $c_k \in CI_k$ do
if $sup(c_k)^{CSW} \geq$ w.s then
$FI_k = \{c_k \in CI_k \mid sup(c_k)^{CSW} \geq w.s\}$;
end if
end for
end for
FI-Output
End

VI. PERFORMANCE EVALUATIONS

We will describe the experimental evaluation of the proposed algorithms. The programs are implemented using Microsoft Visual studio (VB.NET) version 10 and performed on a 3i GHz Pentium PC machine with 2 GB memory running on Windows 7 ., where T, I and D mean the average transaction length, the average length of the maximal frequent itemset, and the total number of transactions, respectively. We are using SQL server for data base management .To simulate data streams, we are using electronics database containing 50 different items ,maximum transaction length are 10 items .We are taking different number of transactions ,different number of window size, different number of support count to evaluate performance of the proposed methods

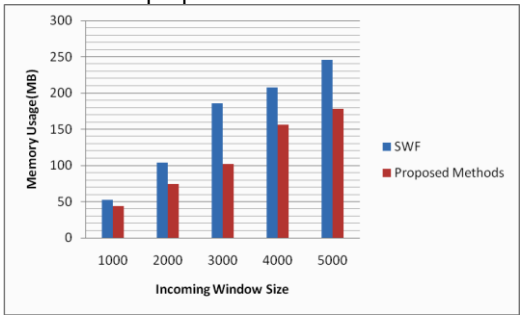

Figure 1: Memory Utilization graph

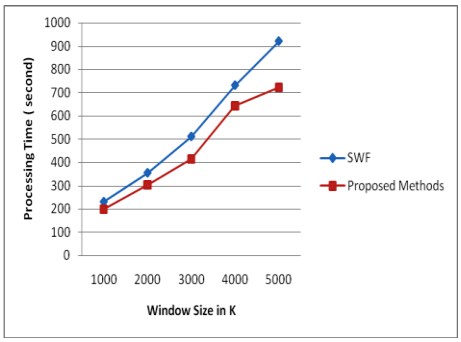

Figure 2: Execution Time graph

VII. CONCLUSION

In this paper, we propose an efficient single-pass algorithm, for mining the set of frequent itemsets over data streams with a efficiently removed aged transaction from current sliding window. We use an effective bit-stream sequence representation of items to enhance the performance of proposed algorithm. The proposed method is also efficient in term of memory problem of finding frequent items in a continuous stream of items. Proposed algorithm we proposed algorithm is efficient in term of execution time number of candidate generation and memory as compared to previous algorithms. In future we can enhance this algorithm to find out recurrent pattern from dynamic data stream.

REFERENCES

[1] J. Han and M. Kamber, Data Mining: Concepts and Techniques, Second ed., Elsevier. 2006.

[2] A. K. Pujari, Data mining techniques.

[3] C. C. Agrawal, Data Streams: Models and Algorithms, Springer, 2007.

[4] F. Chu, Mining Techniques for Data Streams and Sequences. Doctor of Philosophy Thesis: University of California, 2005.

[5] N. A. Chaudhry and M. Abdelgurefi, Stream Data Management, Springer, Vol. 30. 2005.

[6] A. Bifet and R. Kirkby, " Data Stream Mining A Practical Approach" Springer, 2011 .

[7] S. Muthukrishnan, Data streams: algorithms and applications. In Proceedings of the fourteenth annual ACM-SIAM symposium on discrete algorithms, 2003.

[8] C. Agarwal and P. S. Yu, A Framework for Clustering Evolving Data Streams, Int. Conf. on Very Large Data Bases, Berlin, Germany, Sept. 2003.

[9] C. Aggarwal, J. Han, J. Wang, and P. S. Yu, A Framework for Projected Clustering of High Dimensional Data Streams, Int. Conf. on Very Large Data Bases, Toronto, Canada, 2004.

[10] P. S. M. Tsai, Mining frequent item sets in data streams using the weighted sliding window model, Elsevier, 2009.

[11] S. K. Tabeer, B. S. Jeong and Y. K. Lee, "Efficient frequent pattern mining over data streams ', 2008

[12] G. Dong, J. Han, J. Pei, H. Wang and P. S. Yu. "Online mining of changes from data streams: Research problems and preliminary results" In Proceedings of the 2003 ACM SIGMOD Workshop on Management and Processing of Data Streams

[13] J. Guo, P. Zhang, J. Tan and L. Guo "Mining frequent patterns across multiple data streams", 2011.

[14] H. Wang, W. Fan, and J. Han, Mining Concept-Drifting Data Streams using Ensemble Classifiers, in the 9th ACM International Conference on Knowledge Discovery and Data Mining, Washington DC, USA, 2003.

[15] V. Ganti, J. Gehrke, R. Ramakrishnan, "Mining Data Streams under Block Evolution", SIGKDD Explorations vol. 3, no. 2, 2002.

[16] P. Domingos and G. Hulten, "Mining High-Speed Data Streams", In Proceedings of the Association for Computing Machinery Sixth International Conference on Knowledge Discovery and Data Mining, 2000.

[17] H. O. Hebah, "Stream Data Mining", International Journal of Web Applications, vol. 1, no. 4, 2009.

[18] H. F. Lia and M.K. Shan, "Online Mining Maximal Frequent Itemsets over Data Streams". Proceedings of the 15th International Workshop on Research Issues in Data Engineering: Stream Data Mining and Applications, 2005.

[19] J. Chandrika1, K. R. Ananda Kumar, "Frequent Itemset Mining in Transactional Data Streams Based on Quality Control and Resource Adaptation", International Journal of Data Mining & Knowledge Management Process, vol. 2, No.6, 2012.

[20] M. Matysiak, "Data Stream Mining Basic Methods and Techniques," vol. 29, 2012.

[21] C. Chang, M. S. Chen and C. H. Lee, "Mining General Temporal Association Rules for Items with Different Exhibition Periods", In Proceedings of the IEEE International Conference on Data Mining, 2002.

[22] A. Dafa-Alla, H. S. Shon, K. E.K. Saeed and M. Piao, "IMTAR: Incremental Mining of General Temporal", Journal of Information Processing Systems, vol. 6, No.2, 2010.

[23] Y. Kim, W. Y. Kim and U. Kim "Mining frequent item sets with normalized weight in continuous data streams", Journal of Information Processing Systems, 2010.