International Journal of Modern Engineering Research (IJMER)
www.ijmer.com Vol.2, Issue.4, July-Aug. 2012 pp-2508-2511 ISSN: 2249-6645

CB Based Approach for Mining Frequent Itemsets

K. Jothimani[1], Dr. Antony Selvadoss Thanamani[2]
*(PhD Research Scholar, Research Department of Computer Science, NGM College, Pollachi ,India)
** (Professor and Head, Research Department of Computer Science, NGM College, Pollachi ,India

ABSTRACT : *In this paper, we propose a new method for discovering frequent itemsets in a transactional data stream under the sliding window model. Based on a theory of Chernoff Bound, the proposed method approximates the counts of itemsets from certain recorded summary information without scanning the input stream for each itemset. Together with an innovative technique called dynamically approximating to select parameter-values properly for different itemsets to be approximated, our method is adaptive to streams with different distributions. Our experiments show that our algorithm performs much better in optimizing memory usage and mining only the most recent patterns in very less time performance with pretty accurate mining result.*

Keywords: *Chernoff Bound, DataStream, Frequent Itemsets*

I. INTRODUCTION

The most significant tasks in data mining are the process of mining frequent itemsets over data streams. It should support the flexible trade-off between processing time and mining accuracy. 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]. However, the characteristics of stream data – unbounded, continuous, fast arriving, and time- changing – make this a challenging task.

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 can not 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. In this paper, we study the problem of finding frequent items in a continuous stream of items.Many real-world applications data are more appropriately handled by the data stream model than by traditional static databases. Such applications can be: stock tickers, network traffic measurements, transaction flows in retail chains, click streams, sensor networks and telecommunications call records. In the sameway, as the data distribution are usually changing with time, very often end- users are much more interested in the most recent patterns [3]. For example, in network monitoring, changes in the past several minutes of the frequent patterns are useful to detect network intrusions [4].

Many methods focusing on frequent itemset miningover a stream have been proposed. [13] proposed *FPStream*to mine frequent itemsets, which was efficientwhen the average transaction length was small; used lossy counting to mine frequent itemsets; [7],[8],and [9] focused on mining the recent itemsets, which used a regression parameter to adjust and reflect theimportance of recent transactions; [27] presented the*FTP-DS* method to compress each frequent itemset; [10] and [1] separately focused on multiple-level frequentitemset mining and semi-structure stream mining; [12]proposed a group testing technique, and [15] proposeda hash technique to improve frequent itemset mining;[16] proposed an in-core mining algorithm to speed upthe runtime when distinct items are huge or minimumsupport is low; [15] presented two methods separatelybased on the average time stamps and frequency-changingpoints of patterns to estimate the approximate supportsof frequent itemsets; [5] focused on mining a streamover a flexible sliding window; [11] was a block-basedstream mining algorithm with *DSTree* structure; [12] useda verification technique to mine frequent itemsets over astream when the sliding window is large; [11] reviewedthe main techniques of frequent itemset mining algorithmsover data streams and classified them into categoriesto be separately addressed.

The above methods are mostly based on the (ε, δ) approximation scheme,and depend on error parameter ε to control the memory consumption and the running time. While adilemma between mining precision and memory consumption will be caused by the error parameter ε. Alittle decrease of ε may make memory consumption large, and a little increase of ε may degrade outputprecision. Since uncertain streams are highly time-sensitive, most data items are more likely to be changedas time goes by, besides people are generally more interested in the recent data items than those in the past.

Thus this needs an efficient model to deal with the time-sensitive items on uncertain streams.Sliding-window model is the most reprehensive approach to coping with the most recent items in manypractical applications. So we introduce a Chernoff bound with Markov's inequality to deal this problem

The remainder of this paperis organized as follows. In Section 2, we give an overview of the relatedwork and present our motivation for a new approach. Section 3 goes deeper into presenting the problems and gives an extensive statement of our problem. Section 4 presents our solution. Experiments are reported Section 5, and Section 6 concludes the paper with the features of our work.

II. RELATED WORK

Many previous studies contributed to the efficient mining of frequent itemsets (FI) in streaming data [4, 5]. According to the stream processing model [20], the research of mining frequent itemsets in data streams can be divided into three categories: landmark windows [15,

International Journal of Modern Engineering Research (IJMER)
www.ijmer.com Vol.2, Issue.4, July-Aug. 2012 pp-2508-2511 ISSN: 2249-6645

12, 19, 11, 13], sliding windows [5, 6, 14, 16, 17, 18], and damped windows [7, 4], as described briefly as follows. In the landmark window model, knowledge discovery is performed based on the values between a specific timestamp called landmark and the present. In the sliding window model, knowledge discovery is performed over a fixed number of recently generated data elements which is the target of data mining.

According to the existential uncertain stream model, an uncertain stream US is a continuous and unboundedsequence of some transactions, {T1 T2 … Tn …}. Each transaction Ti in US consists of a number of items,and each item x in Ti is associated with a positive probability PTi(x), which stands for the possibility (orlikelihood) that x exists in the transaction Ti. For a given uncertain stream, there are many possibleinstances called worlds that are carried by the stream. The possible worlds semantics has been widely used

in [8, 9], which can be adopted in this paper to illustrate uncertain streams clearly. Each probability PTi(x)associated with an item x deduces two possible worlds, pw1 and pw2. In possible world pw1, item x exists in the transaction Ti, and in possible world pw2, item x does not exist in Ti. Each possible world pwi is annotated with an existence probability, denoted as P(pwi) that the possible world pwi happens. By this semantics, we can get P(pw1)=PTi(x) and P(pw2)=1-PTi(x). In fact, a transaction often contains several items.

We introduce a new approach which combines the mathematical model Chernoff bound for this problem. The main attempts were to keep some advantages of the previous approach and resolve some of its drawbacks, and consequently to improve run time and memory consumption. We also revise the Markov's inequality, which avoids multiple scans of the entire data sets, optimizing memory usage, and mining only the most recent patterns. In this paper, we propose a remarkable approximating method for discovering frequent itemsets in a transactional data stream under the sliding window model. Based on a theory of Combinatorial Mathematics, the proposed method approximates the counts of itemsets from certain recorded summary information without scanning the input stream for each itemset.

III. PROBLEM DESCRIPTION

The problem of mining frequent itemsets was previously defined by [1]: Let I = {i1, i2, . . . im} be a set of literals, called items. Let database DB be a set of transactions, where each transaction T is a set of items such that T ⊆ I . Associated with each transaction is a unique identifier, called its T I D. A set X ⊆ I is also called an itemset, where items within are kept in lexicographic order. A k-itemset is represented by (x1, x2, . . . xk) where x1 < x2 < . . . <xn . The support of an itemset X , denoted support(X), is the number of transactions in which that itemset occurs as a subset. An itemset is called a frequent itemset if support(X) ≥ σ × |DB| where $\sigma\epsilon(0,1)$isauser-specifiedminimum support threshold and $|DB|$ standsforthesizeofthedatabase.

Theproblem ofminingfrequentitemsets istomineallitemsets whosesupport isgreater orequalthan $\sigma\times|DB|$ inDB.

Thepreviousdefinitionsconsiderthatthedatabaseisstatic. Letusnowassumethatdataarrivessequentially intheformof continuousrapid streams. Let*datastream* DS=$B^{b_i}_{a_i}$,$B^{b_{i+1}}_{a_{i+1}}$,...,$B^{b_n}_{a_n}$beaninfinitesequenceofbatches

whereeachbatch isassociated with atime period[a_k,b_k],i.e. B^{b_k},and let$_{a_n}$bethemostrecentbatch

Each batch Each batch Bak consists of a setof transactions; that is, Each batch Bbk = [T1 , T2 , T3, ..., Tk]. We also assume that batches do not have necessarily the same size. Hence, the length

The support of an itemset *X* at a specific time interval [*ai* , *bi*] is now denoted by the ratio of the number of customers having *X* in the current time window to the total number of customers. Therefore, given a user-defined minimum support, the problem of mining itemsets on a data stream is thus to find all frequent itemsets *X* over an arbitrary time period [*ai* , *bi*], i.e. verifying:

*bi*Xsupport*t*(*X*) ≥ *σ* × |*Bbi* |,*t=ai*

of the streaming data using as little main memory as possible.Given a transaction sets X, we are interested in finding strong bounds

Low End: Pr[X < a]

High End: Pr[X ≥ a]

Lemma 1.1 (Markov's Inequality) Let X be a non-negative random variable(transaction sets) with finite expectation μ. Then for any α> 0: Pr[X ≥ α] ≤ μ/α.

Lemma 1.2 (Chebyshev's Inequality) Let X be a random variable(transaction sets) with finite expectation μ and standard deviation σ. Then for any α> 0: Pr[|X − μ| ≥ α σ] ≤ $1/\alpha^2$.

Chebyshev's inequality follows fromMarkov's inequality for the variableY = $(X-\mu)^2$ (so E[Y] =Var[X]) and the fact that $x \rightarrow x^2$ is a strictly monotonic function on R_0 .

$$\Pr[|X-\mu| \geq \alpha\sigma] = \Pr[(X-\mu)^2 \geq \alpha^2\sigma^2]$$

$$\leq \frac{E[(X-\mu)^2]}{\alpha^2\sigma^2} = 1/\alpha^2$$

This makes it tempting to use even higher moments to get better bounds. One way to do thisnicely is by considering an exponential of the basic form of e^x.

Proof.Lemma 1.3 (Simplified Chernoff Bounds)

Low End:

$$\Pr[X < (1-\delta)\mu] < e^{-\mu\delta 2/2} \quad 0 < \delta \leq 1$$

High End:

$$\Pr[X \geq (1+\delta)\mu] < e^{-\mu\delta 2/3} \quad 0 < \delta \leq 1$$

$$\Pr[X \geq (1 + \delta)] < e^{-(1+\delta)\mu} 2e - 1 < \delta$$

Proof.

For the LowEnd let $D = (1 - \delta)^{1-\delta}$. By Taylor expansion we get

$\ln D = (1 - \delta) \ln(1 - \delta)$

$= (1 - \delta)(-\delta - \delta^2/2 - \delta^3/3 - \dots)$

$= -\delta + \delta^2/2 + \delta^3/6 + \dots$

$> -\delta + \delta^2/2$

By Applying this simplified method in the series we can get the following equation for low end windows.

Hence $\left(\frac{e^{-\delta}}{\left((1-\delta)^{1-\delta}\right)}\right) = e^{-\mu\delta 2/2}$

IV. CBMFI ALGORITHM

CBMFI (ChernoffBound Based Mining Frequent Itemsets) algorithm relies on a verifier function and it is an exact and efficient algorithm for mining frequent itemsets sliding windows over data streams. The performance of CBMFI improves when small delays are allowed in reporting new frequent itemsets, however this delay can be set to 0 with a small performance overhead.

The following algorithm shows how it combines with CB for Mining frequent itemsets.

CBMFI

Initialization
For all u pick a route rt(u).
Place all M_u such that D(u) =6 u into C_e where e is the first window on rt(u).
Main Loop r = 0
while(there is a packet not at its destination)
in parallel, for all window ends e = (x, y)
do
ifC_e is not empty,
pick the highest priority itemsets in C_e
y = itemsets(T)
if (a moved itemset is not at its destination)
C_e'= next end transaction
r = r + 1

We would like to use a Chernoff bound, so we need to find some independent Poisson trials somewhere. This requires a bit of thought, lots of random variables associated with this algorithm are not independent;

The expected length of a route is k/2, so the total number of edges on all routes is expected to benk/2. The total number of edges in a hypercube is nk, so we have E[R(e)] = 1/2.It follows that $E[H] \leq k/2$. Now apply the simplified Chernoff bound where $\delta > 2e - 1$:

$$\Pr[H \geq 6k] < 2{-}6k$$

This is allowed since $6k = (\delta + 1)\mu$ implies $\delta > 11$. By applying this in the CBMFI algorithm we can get the efficient result.

V. EXPERIMENTAL RESUTLS

All experiments were implemented with C#, compiledwith Visual Studio 2005 in Windows Server 2003 andexecuted on a Xeon 2.0GHz PC with 4GB RAM. We used 1 synthetic datasets and 1 real-life datasets, which are well-known benchmarks for frequent itemset mining. The T40I10D100K dataset is generated with the IBM synthetic data generator.

We firstly compared the average runtime of these twoalgorithms under different data sizes when the minimumsupport was fixed. As shown in all of the images inFig.5.1, the running time cost of MFI and CBMFI increase following the data size; these results verify thatboth algorithms are sensitive to data size, which is due tothe using of unchanged absolute minimum support

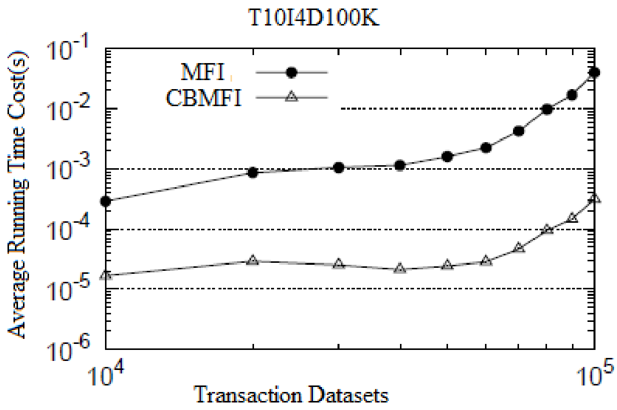

Fig: 5.1 Runtime vs number of Records

Memory Cost Evaluation

To evaluate the memory cost of our algorithm, we comparedthe count of generated items in *MFI*and *CBMFI*.As shown in Fig.5.2 and Fig.5.3, when we fix the minimumsupport and increase the data size, the generated itemsof both algorithm become more, but the overall itemscount of *CBMFI*is less than that of *MFI*since*CBMFI*uses simplified method foritemsets.

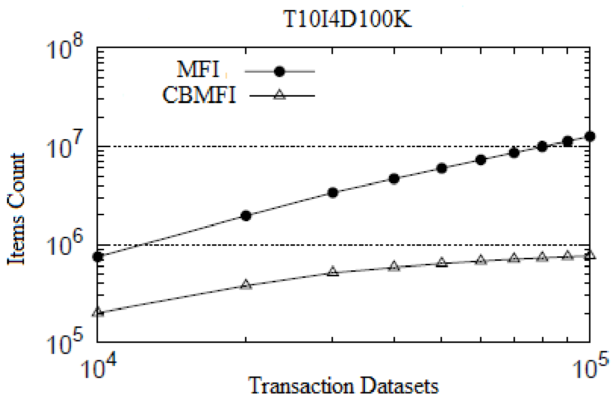

Fig: 5.2 Itemset Count cost vsTransation Set

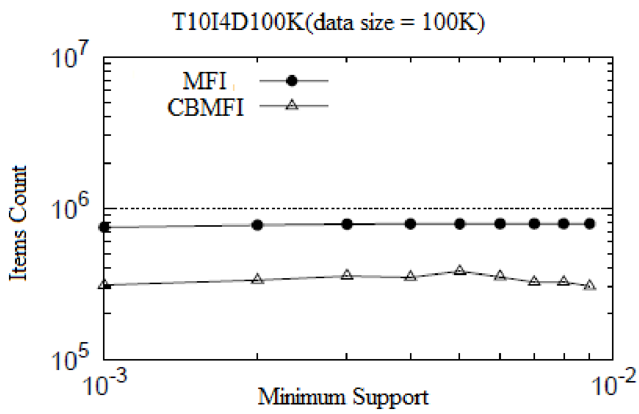

Fig: 5.3 Itemsets Count cost vs. Minimum Support

Furthermore, when the minimum support increases,the incremental scope of *CBMFI*is much reduced thanthat of *MFI*, that is because more redundant items when the overall items count isfixed but the itemsets count increases.

VI. Conclusion

In this paper we considered a problem that how tomine frequent itemset over stream sliding window using Chernoff Bound. We compared twoalgorithmsMFI and CBMFI and introduce a compact datastructure, which can compress the itemsetstorage and optimize itemset computation, based oncombinatorial mathematical. Our experimental studies showed that ouralgorithm is effective and efficient.

References

[1] R. Agrawal, T. Imielinski, andA. Swami. Mining associationrules betweensetsofitems inlargedatabase.In*Proc. oftheIntl. Conf. onManagement ofData(ACM SIGMOD 93)*,1993.

[2] Y.Chen, G.Dong,J. Han, B.Wah, and J. Wang. Multi-dimensional regression analysisoftime-series data streams. In*Proc. ofVLDB'02 Conference*, 2002.

[3] Y.Chi, H.Wang, P.S. Yu, andR.R. Muntz.Moment: Maintainingclosedfrequentitemsets over a stream sliding window. In *Proc. ofICDM'04Conference*, 2004.

[4] P. Dokas, L. Ertoz, V. Kumar, A. Lazarevic, J. Srivastava, and P.-N. Tan. Data mining for network intrusion detection. *In Proc. of the 2002 National Science Foundation Workshop on Data Mining*, 2002.

[5] G.Giannella, J.Han,J.Pei,X.Yan,andP.Yu. Miningfrequentpat-ternsindata streamsatmultipletimegranularities.In*NextGeneration DataMining,MIT Press*,2003.

[6] J. Han, J. Pei, B. Mortazavi-asl, Q. Chen, U. Dayal, andM. Hsu.Freespan: Frequentpattern-projectedsequential patternmining. In *Proc.ofKDD'00Conference*, 2000.

[7] Y. Chi, H. Wang, P.S. Yu, & R.R. Muntz, *"Moment: maintaining closed frequent itemsets over a stream sliding window"*, Proc. 4th IEEE Conf. on Data Mining, Brighton, UK, 2004, pp. 59–66.

[8] N. Jiang & L. Gruenwald, "*CFI-Stream: mining closed frequent Itemsets in data streams*", Proc. 12th ACM SIGKDD Conf. on Knowledge Discovery and Data Mining, Philadelphia, PA,USA, 2006, pp. 592–597.

[9] H.F Li, S.Y. Lee, M.K. Shan, *"An Efficient Algorithm for Mining Frequent Itemsets over the Entire History of Data Streams*", In Proceedings of First International Workshop on Knowledge Discovery in Data Streams 9IWKDDS, 2004.

[10] H.F Li, S.Y. Lee, M.K. Shan, *"Online Mining (Recently) Maximal Frequent Itemsets over Data Streams*", In Proceedings of the 15th IEEE International Workshop on Research Issues on Data Engineering (RIDE), 2005.

[11] J.H. Chang & W.S. Lee, *"A sliding window method for finding recently frequent itemsets over online data streams"*, Journal of Information science and Engineering, 20(4), 2004, pp. 753–762.

[12] J. Cheng, Y. Ke, & W. Ng,*" Maintaining frequent itemsets over high-speed data streams"*, Proc. 10th Pacific-Asia Conf. on Knowledge Discovery and Data Mining, Singapore, 2006, pp.462–467.

[13] C.K.-S. Leung & Q.I. Khan, *"DSTree: a tree structure for the mining of frequent sets from data streams,"* Proc. 6th IEEE Conf. on Data Mining, Hong Kong, China, 2006, pp. 928–932.

[14] B. Mozafari, H. Thakkar, & C. Zaniolo, "*Verifying and mining frequent patterns from large windows over data streams",* Proc. 24th Conf. on Data Engineering, Mexico, 2008, pp. 179–188.

[15] K.-F. Jea& C.-W. Li, "*Discovering frequent itemsets over transactional data streams through an efficient and stable approximate approach, Expert Systems with Applications*", 36(10), 2009, pp. 12323–12331.