Journal of Theoretical and Applied Information Technology
10th December 2013. Vol. 58 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

AN EFFICIENT METHOD FOR RETRIEVING CLOSED FREQUENT ITEM SETS USING HATCI ALGORITHM

[1]MALA. A, [2]Dr F.RAMESH DHANASEELAN
[1]Associate Professor, PSN College of Engg & Tech, Tamilnadu, India.
[2]Professor & Head, Department of Computer Applications,
St. Xavier's Catholic College of Engineering, Tamil Nadu, India
E-mail: [1]malaa0570@gmail.com

ABSTRACT

A Data Stream is a real-time ordered sequence of transactions, which grows continuously and in non-constant rapid rate. The size of data stream is un-bounded. The item sets that appear in a data set with a frequency equal to or above a specified threshold are called frequent item sets. It is difficult to mine all frequent item sets, but easy to mine the closed frequent item sets. The number of Frequent Item sets (FI) grows exponentially when the support threshold is reduced. In such cases, identifying only the Closed Frequent Item sets (CFI) is a better choice as the number of CFIs is very less compared to that of FIs and at the same time the complete information about the FIs can be extracted from the CFIs. In this paper, a new algorithm named as HATCI Algorithm, is used for generating the table for Closed Item sets(CI). Along with this table, a table of supersets and subsets of the CIs and a separate transaction table to store the CIs generated by each transaction, are also maintained. For generating and maintaining these tables, a sliding window model is used, that performs only one scan over the data stream. The size of the transaction table is equal to the size of the sliding window. Whenever the user requests for the CFIs, the CI table is accessed sequentially, and then all the CIs are checked out. The CIs with support count greater than or equal to the support threshold are extracted and returned as CFIs. The proposed methodology is implemented using JAVA platform. The experimental results show that the proposed methodology can effectively retrieve the CFIs on the user's request.

Keywords:- *Data Streams, Closed Frequent Item sets, Sliding window, HATCI Algorithm, AddNewTransaction Algortihm, RemoveOldestTransaction Algorithm.*

1. INTRODUCTION

In the data mining area, data stream mining has turned into an eye-catching research region in recent times. This is owing to the developmentof a set of streaming applications such as traffic network study, web click stream mining, network intrusion detection etc. [18] [16]. Data streams have special features like high arrival rate, massive volume of incoming data, changing nature of data, etc. Mining dynamic data streams is more challenging than traditional static databases due to these properties [8] [15]. Data mining is the procedure of retrieving samples from data and change these data into useful information using some specialized devices.[9] [13]. It is generally applied in a broad range of functions such as marketing, surveillance, fraud detection and scientific discovery [17].

Data streams own different computational features, such as unidentified or limitless length, probably extremely quick arrival rate, incapability to backtrack over formerly arrived items (only one sequential pass over the data is permitted), and a need of system control over the order in which the data arrive. It is difficult to accumulate the whole data into main memory as data streams are of uncontrolled length. On the other hand, for several applications, it is significant to maintain the capacity to perform queries that refer to long-ago data. To hold up such questions is a most important challenge of data stream processing. Conversely, in most applications very old information is regarded less constructive than latest data. We require recognizing the latest changes for the sake of study of data streams [21].

For examining data sets across a broad range of applications, frequent item set mining is a trendy and significant, initial step in data mining. It plays an important position in many significant data mining tasks [20]. Frequent item set mining in fixed databases is comprehensively studied so that a large number of algorithms have been suggested.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Dealings must be scrutinizened only once in frequent pattern mining over data streams and the algorithm must adjust itself to latest concept drifts of data streams. Frequent item set mining algorithm must engage a restricted memory and be as quick as feasible due to high speed and large amount of incoming information [10] [12]. Frequent Closed Itemset Mining is the core significant method in numerous data mining applications (e.g., classifiers, association rules, etc.), for signifying constructive removing of samples (or itemsets) from very large candidate patterns inside a transaction database and for working out the problem of a frequent itemset mining [11]. The set of closed frequent patterns have the entire information pertaining to its related frequent patterns [19]. Hence, closed item set mining over data streams is more enviable than finding the entire set of frequent item sets [14]. Lately, a number of algorithms for mining closed item sets and other type of compressed depictions of item sets have been suggested.

A novel algorithm that is HATCI, Hash Table of Closed Item sets, is suggested which builds tables to signify the closed item sets, their supersets and subsets, and a detach table to accumulate the closed item sets produced by every transaction in this study. The closed item sets in the transactions of the current sliding window are accumulated in a hash table. The value is the closed item set and its representation is the pair (f,n) where f specifies the first item in that closed item set and n specifies the number of clsoed item sets that start with this exact item. This makes certain that a clsoed item set can be directly accessed, when it has to be revised or verified during more processing. Experimental assessments explain that HATCI has superior runtime and devours lower amount of memory with reverence to earlier algorithms.

The remaining of the paper is categorized as follows. In the subsequent section, some associated efforts are assessed. The suggested algorithm is launched in Section 3. Experimental answers are offered in Section 4. Lastly Section 5 ends the paper.

2. LITERATURE REVIEW

Data mining has drawn much consideration in database communities due to its extensive applicability. One of the most important functions is mining association rules in huge transaction databases. Frequent item sets are the item sets that come into view in a data set with frequency. It is hard to mine all frequent item sets, but simple to mine the closed frequent item sets. A concise method for mining the Closed Frequent Item set is specified beneath.

An efficient and competent algorithm for closed frequent item set mining over data streams operating in the sliding window model has been brought in by Fatemeh Noria *et al.* [1]. For stocking up transactions of the window and related frequent closed item sets, this algorithm employs a new data structure. In addition, the support of a novel frequent closed item set was competently calculated and an old pattern was eradicated from the monitoring set when it was no longer frequent closed item set. Wide-ranging experiments on both actual and synthetic data streams demonstrated that the suggested algorithm was better to formerly planned algorithms in terms of runtime and memory usage.

An efficient bit-sequence based, one-pass algorithm, called MFI-Trans-SW (Mining Frequent Item sets within a Transaction-sensitive Sliding Window), to mine the set of frequent item sets from data streams inside a transaction-sensitive sliding window which contains a rigid number of transactions has been suggested by Hua-Fu Li and Suh-Yin Lee [2]. The suggested MFI-TransSW algorithm contains three phases: window initialization, window sliding and pattern generation. Initially, every item of each transaction is programmed in an efficient bit-sequence representation in the window initialization phase. The suggested bit-sequence representation of the item was applied to diminish the time and memory required to slide the windows in the subsequent phases. Next, MFI-TransSW applied the left bit-shift method to slide the windows competently in the window sliding phase. At last, the entire set of frequent item sets inside the current sliding window was created by a level-wise technique in the pattern generation phase. Experimental studies demonstrated that the suggested algorithm not only achieves highly precise mining results, but also run considerably faster and devoured less memory than do presented algorithms for mining frequent itemsets over data streams with a sliding window.

Using a sliding window method, retrieving top-k frequent closed item sets from data streams has been suggested by Pauray S.M. Tsai [3]. For the generation of top-k frequent closed item sets of length no more than max_l, a single pass algorithm, called FCI_max, is built up. Two difficulties happenning in mining association rules were observed. Initially, the user must state a minimum

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

support for mining. Characteristically it may need tuning the value of the minimum support many times before a set of constructive association rules could be attained. There are regularly a lot of frequent item sets produced in the mining result secondly. It will effect in the production of a large number of association rules, giving mount to difficulties of applications. The suggested method can competently resolve the revealed two difficulties in association rule mining, which helps the usability of the mining result in practice.

A new algorithm (called FP-CDS) that can confine all frequent closed item sets and a novel storage structure (called FP-CDS tree) that can be vigorously adjusted to replicate the evolution of item sets' frequencies over time has been offered by Xuejun Liu*et et al.* [4]. A landmark window was separated into numerous basic windows and these basic windows were applied as revising units. Potential frequent closed item sets in every basic window were mined and accumulated in FP-CDS tree based on some suggested approaches.

An algorithm, CLICI, for mining all latest closed item sets in a landmark window model of the online data stream has been suggested by Anamika Gupta *et al.* [5]. The algorithm contains an online component, which progresses the dealings arriving in the stream without candidate generation and revises the abstract properly. The offline element was raised on demand to mine all frequent closed item sets. By stipulating the support threshold vigorously, the user can discover and conduct test.

A novel structure for data stream mining, called the weighted sliding window model has been suggested by Pauray S.M. Tsai [6]. The suggested model permitted the user to indicate the number of windows for mining, the size of a window, and the weight for every window. As a result users can indicate a higher weight to a more important data section, which will formulate the mining result closer to user's obligations. The document has suggested a single pass algorithm, called WSW, based on the weighted sliding window model, to competently find out all the frequent item sets from data streams. An enhanced algorithm, called WSW-Imp, was progressed by examining data features, to added reduce the time of deciding whether a candidate item set is regular or not. Empirical effects demonstrated that WSW-Imp outperforms WSW under the weighted sliding window model.

A technique for compressing the data of all the item sets into a structure with a fixed size by a hash-based method has been suggested by En Tzu Wang and Arbee L. P. Chen [7]. This hash-based strategy proficiently reviews the data of the whole data stream by applying a hash table, offers a new method to calculate approximately the support counts of the non-frequent item sets, and retains only the frequent item sets for speeding up the mining process. The objective of optimizing memory space utilization can be accomplished. The precision guarantee, error study, and parameter setting of this strategy are studied and a series of experiments is executed to demonstrate the efficiency and the competence of this strategy.

In several of the novel applications, data flows through the Internet or sensor networks. It is challenging to widen the mining methods to such a vibrant surrounding. All the frequent itemsets and their support counts obtained from the original database are saved in these techniques. When transactions are added or expired, the support counts of the frequent itemsets contained in them are recomputed. By using again the frequent itemsets and their support counts, the number of candidate itemsets produced during the mining process can be diminished. All these techniques have to rescan the unique database since non-frequent itemsets can become frequent after the database is revised. Thus, they can never work without seeing the complete database and can never be used to data streams. The main challenges consist of a rapid reply to the incessant request, a compressed synopsis of the data stream, and a method that acclimatizes to the restricted resources.

3. PROPOSED METHODOLOGY DURING THE TENURE OF THE RESEARCH

Generally data streams contain large amount of data with old and new information. People need newer information rather than the oldest information. This leads to the sliding window concept. Sliding window needs less amount of storage space compared with that of an entire data stream. And also the sliding window shows the current information of a data stream due to its up-to-date position.

In this paper, a new algorithm namely HATCI is proposed which constructs tables to represent the closed item sets, their supersets and subsets, and a separate table named as transaction table, to store the Closed Item sets generated by each transaction. For generating and maintaining these tables, a sliding window model is used, that performs one scan over the data stream. The size of the transaction table is equal to the size of the sliding window. When a user makes a request for finding

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Closed Item sets, the CI table is accessed sequentially and it retrieved the results using Hash tables. The proposed methodology for HATCI Algorithm is given in the figure 1.

3.1. HATCI Algorithm

HATCI Algorithm, Hash Table of Closed Item sets, is used with a hash table for generating the table for Closed Frequent Item sets (CFIs) with its table of supersets and subsets and also for generating a separate transaction table to store the closed item sets generated by each transaction. For generating and maintaining these tables, a sliding window model is used, that scans one time, over the data stream. The size of the sliding window is equal to the size of the transaction table.

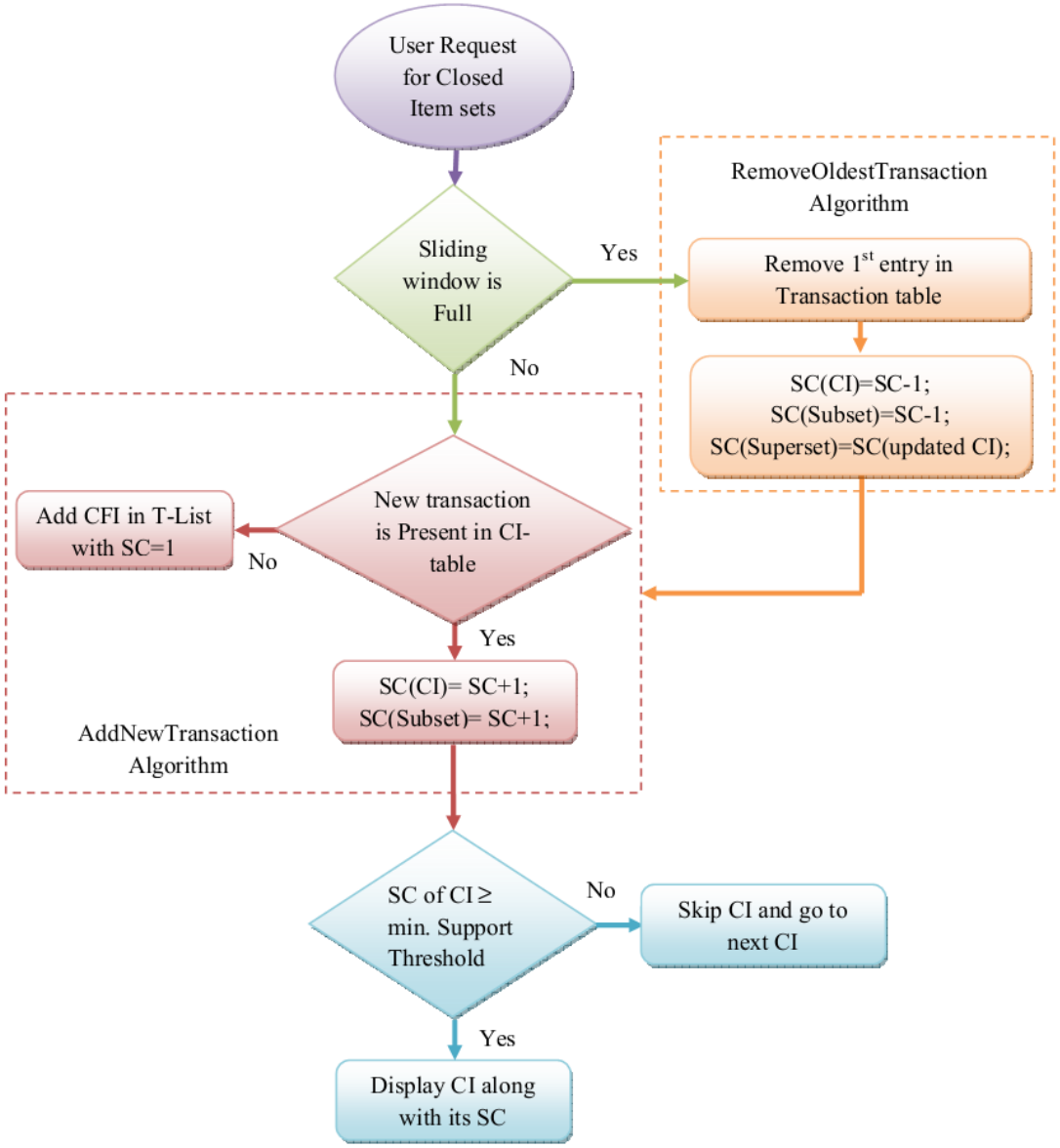

Figure 1 : Proposed Diagram for HATCI Algorithm

3.1.1. Generation of the Table for Transactions

Transaction-table is an array of Closed Item sets created by each transaction in the window. The transaction table contains the transactions in the data stream and the related Closed Item sets. In this transaction table two columns are presented. The

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

general structure for transaction table is given in table 1.

Table 1: Transaction table Structure

T_{id}	Key
T_i	(f,n)

Column "T_{id}"

The column "T_{id}" in the transaction table represents the transactions "T_i" in the data stream. Here i takes the value of transaction number.

Column "Key"

The "Key" column indicates the pair (f,n), in which f indicates the order number of first term of the Closed Item set and n indicates the order number of the Closed Item sets that begins with this particular first term.

For example, consider that the Closed item sets: $\{ab, abc, bc, bcd, cd\}$. The f value for the Closed item set ab, is 1. Because, the order number of the first term a in the item set ab is 1. If we consider bc means, then the order number of b is 2; so the f value of bc is also 2. And also, the n value of the Closed Item set ab is 1 and the n value of abc is 2. Totally the Closed item sets that starts with this particular first term a is 2, so one of the Closed Item set ab has 1 as the value of n and the other item set abc has 2 as the value of n.

3.1.2. Generation of the Table for Closed Item Sets

The hash table stores the Closed Item sets (CI) that is presented in the transactions of the current sliding window. In CI hash table, two columns are used for the processing. First column is for "Key". Second column is for "Value" and the last is for "SC-Support Count". The general structure for the Closed Item set table is given below in table 2.

Table 2: Table Structure for Closed Item sets

Key	Value	SC
(f,n)	Closed Item sets	S

"Key" column

The "Key" column indicates the pair (f,n), in which f indicates the order number of first term of the Closed Item set and n indicates the order number of the Closed Item sets that begins with this particular first term.

"Value" column

The "Value" column indicates the Closed Item sets from the data stream.

"Support Count" column

The "Support Count" column indicates S, the number of times the particular item set occurs in the data set.

3.1.3. Generation of the table for Supersets

The Closed Item sets that are present in the CI table have supersets. These supersets are listed in this superset table. The "Key" in the supersets indicates the value of this hash table with the same key pair (f,n).

3.1.4. Generation of the table for Subsets

The Closed Item sets that are present in the CI table also have subsets. These subsets are listed in this subset table. The "Key" in the supersets indicates the value of this hash table with the same key pair (f,n).

3.2. Phases of HATCI algorithm

HATCI Algorithm has three phases. They are,

1. Initialization Phase
2. Sliding Phase
3. Generation Phase

3.2.1. Initialization Phase

The Initialization phase uses the Algorithm, the AddNewTransaction. This algorithm continues until sliding window is full. The arrival of new transactions, automatically checks for the sliding window, whether it is full or not. If the sliding window is not full, then the CI table is updated by using AddNewTransaction algorithm.

AddNewTransaction Algorithm

// **Global:** N - size of the sliding window
// CI-table - hash table of closed item sets
// Superset-table - hash table of supersets of each CI
// Subset-table - hash table of subsets of each CI
// Transaction-table - array of CIs created by each transaction in the window
// **Input:** new transaction - T
// **Output:** updated tables

1. Take the first item of the new transaction.
2. Check the CI-table to see if this transaction is already presented as a Closed Item set.
3. If it is present, then increment it supports count and also the support count of all its subsets listed in the Subset-table and returns.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

4. If it is not present, then add 'T' as a Closed Item set in a temporary list, T-list with support count, $SC = 1$.
5. For each CI C, in the CI-table do the following

a. Take intersection I , of 'C' and 'T'.

b. If I not null

i. Find key of intersection

ii. If $I \neq T$

Add T as superset of I and I as subset of T

iii. If $I \neq C$

Add C as superset of I and I as subset of C

iv. If I not in T-List

Add I with $SC = Closed\ Item\ set's\ SC + 1$ to the T-List at determined key position.

Otherwise, increment SC

v. For each superset S, of CI, listed in Superset-table, do

Add I as subset of S and S as superset of I

6. For each CI, i in T-list, TCI do

For each remaining CI, j in the T-list do

If $i \subset j$

Set i as subset of j and j as superset of i

else if $j \subset i$

Set i as superset of j and j as subset of i

7. Each CI in the T-list is inserted into or updated in the CI-table.
8. Each CI is also added to the list in the Transaction-table for this new transaction.

Whenever CI table is updated, the updating of superset table and subset table is also performed simultaneously. Transaction table shows each transaction with their Closed Item set. If the CI is created or affected by one transaction, then these details are updated in the transaction table. One temporary list, "T-list" is also generated for comparison. Whenever the transaction arrives, the particular transaction is checked against the CI table and the T-list. After checking, if the transaction is already present as a Closed Item set either in the CI table or in the T-list, then its Support Count SC, is increased, along with the support count of its subsets and then the algorithm returns. If not, then the new transaction is added as a Closed Item set in the T-list with the SC value 1. We have to take the intersection of each Closed Item set in the CI table with a new transaction. If the intersection is not presented in the T-list, update the T-list with this Closed Item set and also increment the SC value of the particular Closed Item set. It also makes changes in the superset and subset tables.

3.2.2. Sliding Phase

The RemoveOldestTransaction Algorithm is used in the Sliding Phase. Initially the sliding window is checked out for the transactions. If the current sliding window is full, then the sliding phase can be started. In this phase, we can remove the old transaction using the Algorithm, RemoveOldestTransaction and also we can add the new transaction using AddNewTransaction Algorithm.

RemoveOldestTransaction Algorithm

// Used to remove the oldest transaction by updating the global CI-table, Superset-table, Subset- table and Transaction-table

1. For each CI in the first entry in the Transaction-table (oldest transaction) do

a. Reduce SC by one. If $SC = 0$ mark CI as invalid.

b. For each of its subset in the Subset-table do

Reduce SC by one. If $SC = 0$ mark CI as invalid.

2. For each superset of updated CIs do

If SC of superset = SC of the updated CI then mark CI as invalid

If the old transaction affects the CIs, then this effect is found by taking the first entry in the transaction with the help of RemoveOldestTransaction Algorithm. And then decrease the SC value by one. Then the CIs are checked out, whether the SC value of the CIs is same as that of any of its superset and subset table. If so, then the particular CIs become invalid and then the corresponding CIs are deleted from the CI table, Superset table and Subset table.

3.2.3. Generation Phase

Generation phase starts with the user's requests. When the user queries for the Closed Item set, the CI table is scanned sequentially for the requested Closed Item set, and then checked out for all the CIs whether the SC of all CIs is greater than or

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

equal to the Support threshold. If that condition is satisfied, then that particular CIs are extracted.

Generate-CFI Algorithm

// **Global:** CI-table - Table of CIs in current window

// S_t - Minimum support threshold

// **Input:** User request

// **Output:** CFIs in current window

1. For each entry in the CI-table do

   If $Support\ Count\ of\ CI \geq S_t$ then

   Display CI value along with its Support Count

Else

Skip that CI and go to the next CI

3.3. Computational Part of Hatci Algorithm

Consider the data stream D , with five transactions $\{T_1, T_2, T_3, T_4, T_5\}$, that have support threshold $S_t = 50\%$ and window size = 4. The representation of the data stream is, $D = \{T_1, T_2, T_3, T_4, T_5\}$

The data stream with five transactions and with their corresponding Item sets are given in the table 3.

Table 3: Data stream with 5 transactions

Transactions	Item sets
T_1	cd
T_2	ab
T_3	abc
T_4	abc
T_5	bc

The size of the sliding window is 4, so the two windows W_1 and W_2 are,

$W_1 = \{T_1, T_2, T_3, T_4\}$

$W_2 = \{T_2, T_3, T_4, T_5\}$

For the window W_1 , Closed Item sets are ab, abc, cd, c and the transaction table, CI table, Superset table and Subset table are given below in figure 2.

Transaction Table

TID	CI-Keys
T_1	(3,1)
T_2	(1,1)
T_3	(1,1), (1,2), (3,2)
T_4	(1,1), (1,2), (3,2)

s

e

CI Table

Key	Value	SC
(1,1)	ab	3
(1,2)	abc	2
(3,1)	cd	1
(3,2)	c	3

Superset Table

Key	Value
(1,1)	(1,2)
(1,2)	-
(3,1)	-
(3,2)	(3,1), (1,2)

Subset Table

Key	Value
(1,1)	-
(1,2)	(1,1), (3,2)
(3,1)	(3,2)
(3,2)	-

Figure 2: Transaction, CI, Superset, Subset tables for window W_1.

Journal of Theoretical and Applied Information Technology
10th December 2013. Vol. 58 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

In figure 2, the two pointers s, e are used in the transaction table. The pointer s indicates the start of the current window and the pointer e indicates the end of the current window.

When a new transaction T_5, is arrived as in the illustration, the sliding window checks out for the transactions. If the current window is full, then the sliding phase is started. The oldest transaction T_1 is removed using the Algorithm which is called, RemoveOldestTransaction. While removing any transaction, the changes are made in all the tables corresponding to that particular transaction. In the transaction table, the updation change is made for the two pointers s and e. Now, the start of the current window s is T_2. In CI table, the key for the transaction T_1 is (3,1) and the Closed Item set value is cd. This row is deleted from the CI table. Now, the removal also has made changes in the Support Count column. Since the transaction T_1 is removed, the Support Count values in the CI table for the particular key and other related keys are also reduced by one. Now the particular transaction T_1 is eliminated also from the CI, Superset and Subset tables. The updation in all the tables, while removing the transaction T_1 is shown in figure 3. Thus all the tables are updated for the removal of the transaction T_1. Hence the transaction T_1 is removed.

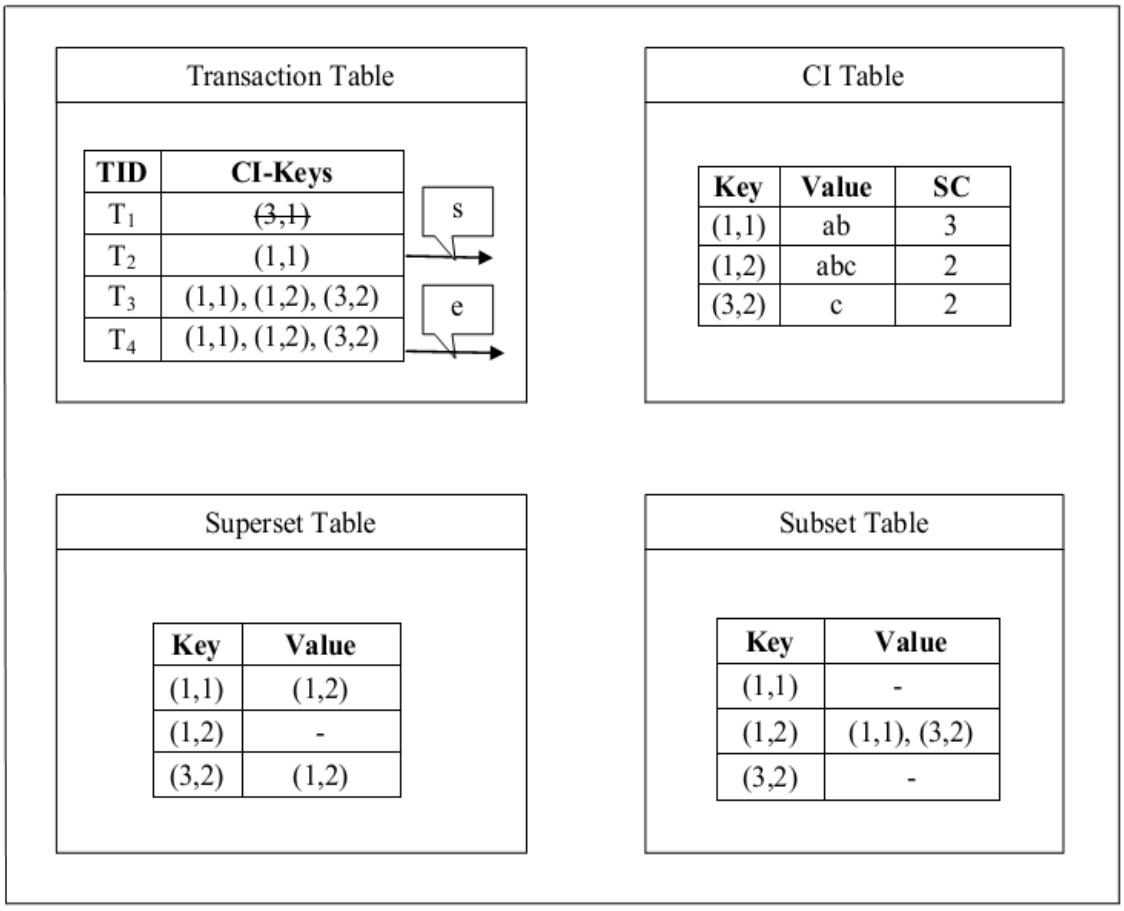

Figure 3: Updation of Transaction, CI, Superset, Subset tables after the removal of transaction T_1.

The arrival of new transaction T_5 is added using the Algorithm, AddNewTransaction. Since the oldest transaction T_1 is removed, the space for that transaction is empty in the transaction table. This space in transaction table is occupied by the new transaction T_5. In the transaction table, the space for each transaction is being used in a circular fashion. Now the pointers have interchanged their place also. Here, the start of the current window s is T_2 and the end of current window e is T_5. The new addition of transaction T_5 is having two new additional Closed Item sets bc and b. The Support Count for the CI value bc is 3 and for b

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

is 4. The Closed Item set value is no longer a CI, since it has a superset, bc having the same SC. So, it has been deleted from all table entries. These SC values of bc and b are updated to the CI table and further corresponding updations are performed in Superset and Subset tables. Hence, the addition of new transaction T_5 is done.

The updation of the tables transaction, CI, Superset and Subset for the arrival of the new transaction T_5 is shown in the following figure 4.

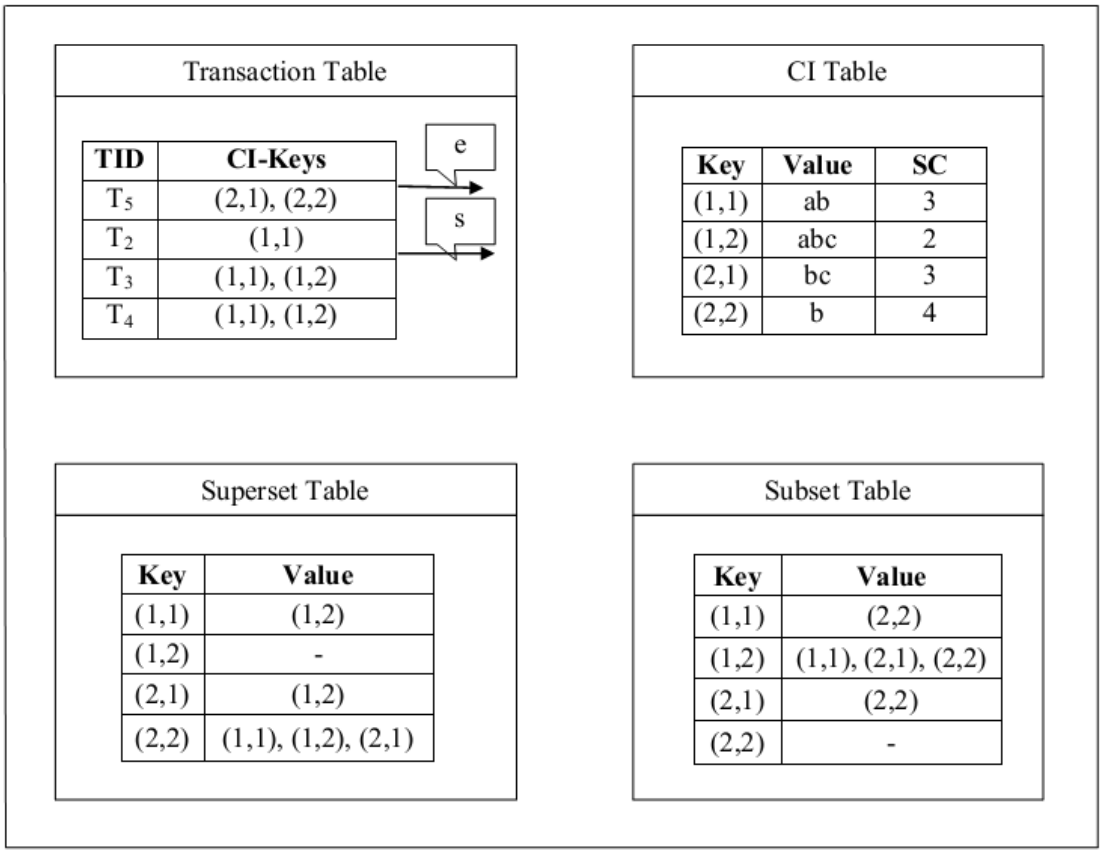

Figure 4: Updations of Transaction, CI, Superset, Subset tables after including the transaction T_5.

When the user requests for the Closed Item set, the CI table is scanned sequentially for the requested Closed Item set, and then check out all the CIs whether the SC of all CIs is greater than or equal to the Support threshold. If that condition is satisfied, then that particular CIs are extracted.

4. PERFORMANCE EVALUATION

The proposed Closed Frequent Itemset Retrieval system is implemented in JAVA 1.6. We have used two datasets for evaluating the performance of the retrieval system. The two datasets used are Mushroom and Chess datasets.

4.1. Dataset Description

Mushroom dataset: This data set includes descriptions of hypothetical samples corresponding to 23 species of gilled mushrooms in the Agaricus and Lepiota Family. Each species is identified as definitely edible, definitely poisonous, or of unknown edibility and not recommended. This latter class was combined with the poisonous one. The Guide clearly states that there is no simple rule for determining the edibility of a mushroom; no rule like ``leaflets three, let it be" for Poisonous Oak and Ivy.

Chess dataset: This dataset contains domain theories, in addition to that, it contains some useful prolog routines. One routine takes a generic chess board description and a domain theory name, and produces a prolog state description suitable for use with the given domain theory.

Journal of Theoretical and Applied Information Technology
10th December 2013. Vol. 58 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

4.2. Evaluation Measures

Two measures are used for analyzing the peroformance of our proposed system. They are given below.

1. Number of Retrievals
2. Computational Time

Number of Retrievals evaluates the total number of Closed Item Sets that are retrieved from the data stream. Normally, in a data stream, the Closed Item Sets are less than that of the Frequent Item Sets.. The Computational Time indicates the total time that is taken for retrieving the Closed Item Sets. The Time for retrieving should be low for a good system.

4.3. Experimental Results

The experimental results of the proposed Closed Frequent Itemset Retrieval system are presented in this section. We analyze the Computational time for retrieval and the number of retrievals of Closed Item Sets for our proposed method and also the results are presented in this Section. The following figure 5 and figure 6 shows the comparison graphs of Frequent Itemset with Closed Frequent Itemset using both data sets Chess and Mush room.

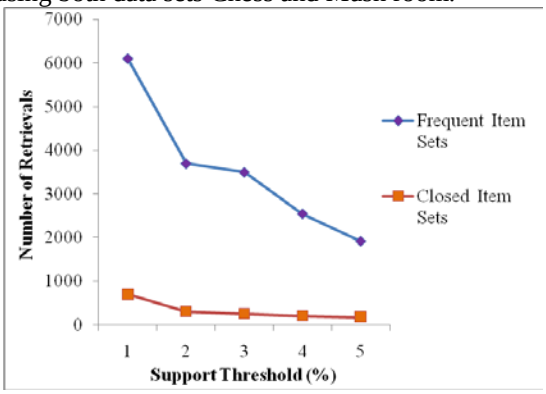

Figure 5: Graph for Comparing Frequent ItemSet and Closed Frequent Itemset – Chess data set

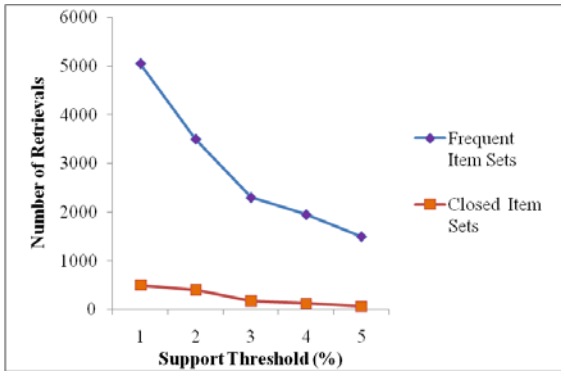

Figure 6: Graph for Comparing Frequent ItemSet and Closed Frequent Itemset – Mushroom data set

From the above figure 5 and figure 6, we can say that Closed Frequent Itemset method can retrieve the Closed Frequent Itemset more accurately, from both the data sets Chess and Mushroom. When the support threshold (%) is a low value, we can retrieve more number of Closed Frequent Itemsets. Normally all data streams contain less number of Closed Item Sets when compared with the Frequent Sets.

4.3.1. Number of retrievals of Closed Item Sets

For getting the number of retrievals of Closed Item Sets, in this paper we compare the moment algorithm with our Proposed HATCI algorithm. The figure for comparing the moment algorithm and proposed HATCI algorithm for finding out the total number of Closed Item Sets retrieval is given in below figure 7 and figure 8 using both Chess and Mushroom data sets respectively.

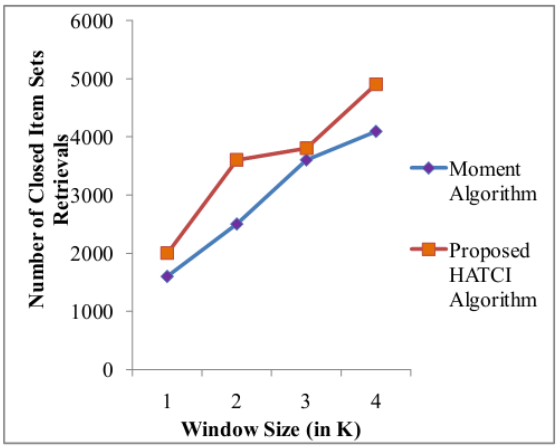

Figure 7: Graph for finding the number of retrievals by comparing Moment and our Proposed HATCI Algorithms – Chess data set.

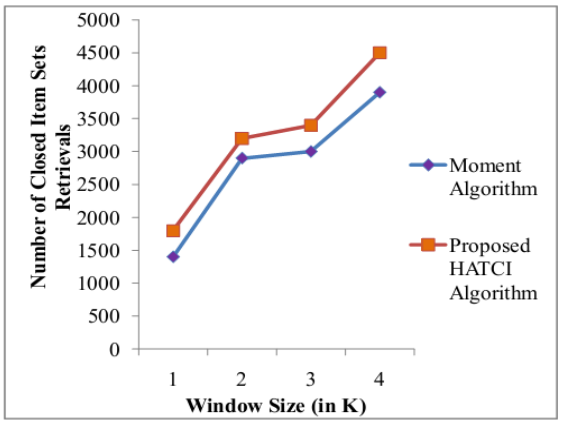

Figure 8: Graph for finding the number of retrievals by comparing Moment and our Proposed HATCI Algorithms– Mushroom data set.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

From figure 7 and figure 8, we can prove that our Proposed HATCI Algorithm has the ability to retrieve more number of Closed Item Sets when compared with the moment algorithm by using both the Chess and Mushroom data set. According to the window size, the number of retrievals of the Closed Item Sets is changed. If the Window size is small, then low number of Closed Item Sets are retrieved; Otherwise more number of Closed Item Sets are retrieved.

4.3.2. Computational Time for the Closed Item Sets Retrieval

For getting the retrieval execution time, we compare the moment algorithm with our Proposed HATCI algorithm by using the two data sets Chess and Mushroom. The graph for comparing the moment algorithm and proposed HATCI algorithm is given in figure 9 and figure 10 below:

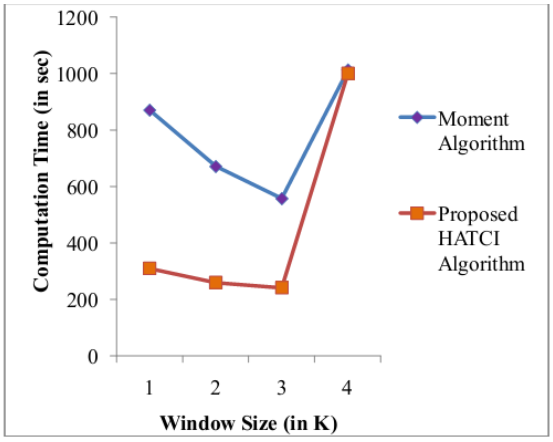

Figure 9: Graph for finding the computational time by comparing Moment and our Proposed HATCI Algorithms – Chess data set

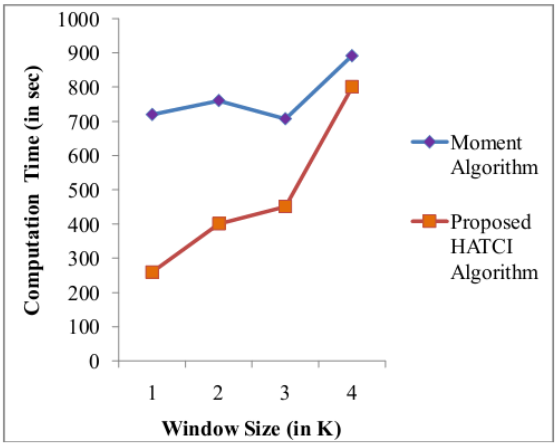

Figure 10: Graph for finding the computational time by comparing Moment and our Proposed HATCI Algorithms – Mushroom data set

From figure 9 and figure 10, we can prove that our Proposed HATCI Algorithm requires less execution time when compared with the moment algorithm by using both data sets Chess and Mushroom. We can understand from these figures that the computational time changes according to the window size. When the window size is increased, the Computational time for retrieving Closed Item Sets will also get increased. From this section, it can be proved that our Proposed HATCI algorithm performed better than the existing algorithms.

5. CONCLUSION

In this paper, the Closed Item sets that are requested by users were obtained using our proposed HATCI Algorithm. HATCI Algorithm, was used for generating the table for Closed Frequent Item sets with its table of supersets and subsets and also for generating a separate transaction table to store the closed item sets generated by each transaction. For generating and maintaining these tables, a sliding window model was used, that scanned one time, over the data stream. When the user requested for the Closed Item set, the CI table was scanned sequentially for the requested Closed Item set, and then checked out all the CIs whether the SC of all CIs is greater than or equal to the Support threshold. If that condition was satisfied, then that particular CIs were extracted. The proposed methodology was implemented using JAVA platform. The experimental results showed that the proposed methodology effectively retrieved the Closed Frequent Item sets on the user's request.

REFERENCES

[1] Fatemeh Noria, Mahmood Deypirb and Mohamad Hadi Sadreddini, "A sliding window based algorithm for frequent closed item set mining over data streams", *The Journal of Systems and Software*, Vol. 86, pp.615– 623, 2013.

[2] Hua-Fu Li and Suh-Yin Lee, "Mining frequent item sets over data streams using efficient window sliding techniques", *Expert Systems with Applications*, Vol. 36, No. 2, pp. 1466–1477, 2009.

[3] Pauray S.M. Tsai,"Mining top-k frequent closed itemsets over data streams using the slidingwindow model", *Expert Systems with Applications*, Vol. 37, No. 10, pp. 6968–6973, 2010.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[4] Xuejun Liu, JihongGuanb and Ping Huc,"A sliding window based algorithm for frequent closed itemset mining over datastreams", *Computers and Mathematics with Applications*, Vol. 57, pp. 927-936, 2009.

[5] Anamika Gupta, VasudhaBhatnagar and Naveen Kumar,"Mining Closed Itemsets in Data Stream Using Formal Concept Analysis", *Data Warehousing and Knowledge Discovery*, Vol. 6263, pp. 285-296, 2010.

[6] Pauray S.M. Tsai, "Mining frequent itemsets in data streams using the weighted sliding window model", *Expert Systems with Applications*, Vol. 36, No. 9, pp. 11617-11625, 2009.

[7] En Tzu Wang andArbee L. P. Chen,"A novel hash-based approach for mining frequent item sets over data streams requiring less memory space", *Data Mining and Knowledge Discovery*, Vol. 19, No. 1, pp. 132-172, 2009.

[8] Nan Jiang and Le Gruenwald, "CFI-Stream: Mining Closed Frequent Itemsets in Data Streams", *In Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 592-597, 2006.

[9] Yun Chi, Haixun Wang, Philip S. Yu and Richard R. Muntz,"Catch the moment: maintaining closedfrequent itemsets over a data streamsliding window", *Knowledge and Information Systems*, Vol. 10,No. 3, pp. 265 – 294, 2006.

[10] En Tzu Wangand Arbee L. P. Chen,"A novel hash-based approach for mining frequentitemsets over data streams requiring less memory space", *Data Mining and Knowledge Discovery*, Vol. 19, No. 1, pp. 132-172, 2009.

[11] Chakarin Vajiramedhin, and Jeeraporn Werapun, "EBPA: An Efficient Data Structure for Frequent Closed Itemset Mining", *Applied Mathematical Sciences*, Vol. 7, No. 30, pp. 1483 - 1506, 2013.

[12] K Jothimani and Dr. Antony SelvadossThanamani,"An Overview of Mining Frequent Itemsets OverData Streams Using Sliding Window Model", *International Journal of Emerging Trends & Technology in Computer Science*, Vol. 1, No. 1, 2012.

[13] JiaDongRenand Cong Huo,"Mining Closed Frequent Itemsets in Sliding Window over Data Streams", *In Proceedings of 3rd International Conference on Innovative Computing Information and Control*, pp. 76-80, 2008.

[14] Guanglu Zhang, Jingsheng Lei andXinghui Wu,"Mining frequent closed itemsets over data stream based on Bitvector and digraph", *In Proceedings of 2nd International Conference on Future Computer and Communication*, Vol. 2, pp. V2-241- V2-246,2010.

[15] FatemehNori,MahmoodDeypir, MohamadHadi and KoroshZiarati, "A New Sliding Window Based Algorithm for Frequent Closed Itemset Mining OverData Streams", *In Proceedings of1st International eConference on Computer and Knowledge Engineering (ICCKE)*, pp. 249- 253, 2011.

[16] S. Ben Yahia,T. Hamrouni andE. MephuNguifo,"Frequent closed itemset based algorithms: a thorough structural and analytical survey",*ACM SIGKDD Explorations Newsletter*,Volume 8 Issue 1, Pages 93-104, 2006.

[17] Jian Pei, Jiawei Han andRunying Mao,"CLOSET: An Efficient Algorithm for Mining Frequent Closed Itemsets", *In ACM {SIGMOD} Workshop on Research Issues in Data Mining and Knowledge Discovery*, pp. 21-30, 2000.

[18] J. TahmoresNezhad andM.H.Sadreddini, "PTclose: A novel algorithm for generation ofclosed frequent itemsets from dense andsparse datasets", *In Proceedings of the World Congress on Engineering,London, U.K.*, Vol.1, 2007.

[19] Prabha S, Shanmugapriya S, and Duraiswamy K, "A Survey on Closed Frequent Pattern Mining ", *International Journal of Computer Applications*, Vol. 63, No. 14, pp. 0975-8887, February 2013.

[20] M.Rajalakshmi, Dr. T. Purusothaman and Dr.R.Nedunchezhian,"Maximal Frequent Itemset GenerationUsing Segmentation Approach", *International Journal of Database Management Systems (IJDMS)*, Vol.3, No.3, 2011.

[21] Congying Han, Lijun Xu, and Guoping He, "Mining Recent Frequent Itemsets In Sliding Windows Over Data Streams", *Computing and Informatics*, Vol. 27, pp. 315-33, 2008.