International Journal of Computational Intelligence Research
ISSN 0973-1873 Volume 13, Number 2 (2017), pp. 183-196

http://www.ripublication.com

CASW: Context Aware Sliding Window for Frequent Itemset Mining over Data Streams

V.Sidda Reddy[1], T.V.Rao[2] andA.Govardhn[3]

[1]*Department of CSE, Sai Tirumala NVR Engineering College, A.P, India.*

[2]*Department of Computing, K.L.University, A.P, India.*

[3]*Department of CSE, JNTUHE, T.S, India.*

Abstract

In recent years, advances in both hardware and software technologies coupled with high-speed data generation leads data streams and data stream mining. Data generation has been much faster in data stream applications and score of data is generated in quick turnaround time. Hence it becomes obvious to perform mining, data on arrival that is usually termed as data stream mining. General frequent pattern mining methods are envisaging limitations and do not support in responding to a massive quantum of data being streamed. In order to address such limitations, data mining researchers have focused on methods for conducting more efficient and effective mining tasks by scanning a database only once. As a process of evolution, sliding window model that perform mining operations focusing on updating accumulated parts over data streams, are proposed. It is hard to consider all of the frequent patterns in data stream environment as generated patterns were remarkably increasing as data streams get extended continuously. Hence, methods for efficiently compressing patterns that are generated are essential to address the limitations. Considering the challenges and shortcoming in the earlier solutions, in this paper, focus is on incremental mining of frequent patterns from the window and a solution of CASW (Context Aware Sliding Window) is proposed. There are well defined boundaries for frequent and infrequent patterns for specific patterns. In this research article, we adapt usage of window size change for representing conceptual drift in the information stream. An experimental study carried out on the model depicts significant developments and has affirmed that the algorithm has been designed with a more efficient system than that of existing solution.

Keywords: Data Streams, Data Stream Mining, Frequent Itemset, Frequent Itemset Mining, Sliding Window.

1. INTRODUCTION

Data mining solutions have become an integral part of analytical solutions and knowledge discovery process. Different data mining solutions have been applied for mining interesting patterns on huge amount of data. Among different data mining solutions frequent pattern mining is one of the significant areas of research with varied approaches and widely accepted in vivid areas of industrial sectors. Numerous algorithms were developed to find frequent patterns, a few of the well-known fundamental frequent pattern mining algorithms are Apriori [1] constituting breadth first search and FP-growth [2] that is reliant on depth first search. Many improvised solutions have been developed based on the aforesaid fundamental algorithms. For instance, certain algorithm models like frequent pattern mining comprising a minimum support threshold that is specified by users [3], [4], [5] and sequential frequent pattern mining [6], [7], [8]. Frequent pattern mining is utilized in vivid range of applications including in the medical domain [9] weblog and web page click analysis [10], [11].

Apart from applying frequent pattern mining in the databases that are static, it is also applied to the data streams with high score volume of data that is constantly updating in to the data systems in a real-time environment, which signifies continuous and unlimited features. However, certain requirements that are to be addressed by data stream mining are [12]:

- Only once the data elements required for data streaming has to be analyzed.
- Despite the fact that there is constant stream of data inflow in to the database, still the issues like the memory usage for mining has to be limited for having acceptable and constant range.
- The data elements that are updated in to the information system have to be processed in quick turnaround time.
- Results of data stream analysis should be of very insightful and resourceful and it can be accessible instantly.

The frequent pattern mining methods that were proposed in earlier times do not adhere to such standards and many methods scan database multiple times to mine frequent patterns. Some of the methods that could be effectively resourceful for mining approaches, and the one that could be resourceful in addressing the requirements are proposed in few of the studies [13], [14]. However the proposed data stream mining methods has succeeded in extraction of frequent patterns over data streams effectively, still one of the key issues that are encountered is with the data elements getting added regularly, and the size of the data is also raising high, it becomes significant challenge in mining the time spent on accumulation of transaction data, and thus resulting in lapse in addressing one of the requirements of data streaming and also the immediate processing. In order to address such issues, CFP and MFP notations [4], [12], [15-28] representing more compact forms of general frequent patterns, which can be utilized.

The MFP notation assures more efficient pattern of compression with more compact forms than the CFP notation, however there is a slight pattern loss that occur when the conversion of general ones to MFPs takes place. Also, if MFP method without outstanding compressibility is adapted for data streaming, valid patterns over data streams with more efficient system is observed. As data is accumulated over data streams continuously, importance of certain data in the system might be ignored or no longer be essential and relatively the importance of recently added data might be high.

Many window model based mining approaches were proposed in [5], [12], [14], [29 - 35] for the aforesaid context. Predominantly, the fixed window, sliding window and adaptive window techniques are chosen as per the requirements of data streams. Data streams are composed of varied items and every item denotes certain objects from real-world. For instance, in retail market data stream, item could represent regarding products in the supermarket section, in the case scenario of medical records, it could be records of the patients with varied classification etc. and significance to each item might actually be different. Hence, obtaining high-quality mining results reflecting not only items, frequency and also their importance by working on a weight factor in the data stream mining.

In this paper, the model is proposed for mining frequent itemsets using variable window size fixed by Context Aware Sliding Window (CASW). The proposed solution shall be effective in memory usage and computational complexity. Context variation analysis is used in the solution proposed reliant on dynamic window oriented transaction storage is used in the solution proposed and the mining of frequent item sets from the concluded window are performed using TIFIM [36].

2. RELATED WORK

In the contemporary models from the recent past, Apriori [1] proposed model performs only in the static database solutions using downward closure property. It scan database multiple times and generate more number of candidates which leads memory and processing time expensive. The algorithm FP-Growth [2] mine complete set of frequent itemsets from large database and implemented compact data structure FP-Tree to store information about frequent itemsets. As two fixed database scans are adapted for mining by FP-Growth, algorithm effectively performs and do not generate any candidate patterns.

Though mining methods that are reliant on FP-Growth have an effect on static databases, it might not right fit model for data streams that are accumulating data in continuous manner. The other challenge is that it may not deal with data streams instantly, as these methods have to perform scans of more than two databases. In the other dimension, as the trees are constructed with items that are remaining after infrequent items that are deleted, the trees that are generated earlier had to be discarded and new trees have to be built again whenever the new transaction data are added in to data streams. In data streams, though a certain item might be infrequent in current time frame, it might turn to be a frequent one according to addition of new transaction data.

But the challenge is that, the model for reading the databases right from start has to run both the scan based methods as they have already excluded the infrequent items in the earlier steps. For addressing such issues, mining methods comprising suitable data streams are proposed [37], [38], [13] that can perform mining tasks having only one database scan and also respond to the changes of data streams in immediate manner.

Accordingly, the sliding window-based frequent pattern mining approach is proposed [5], [30 - 35] which mine frequent patterns based on the latest transaction data of large data streams.

In [13], [14] it is discussed that if the general data sizes are large, implementation of static database that can cause computational overheads. In sliding window-based data stream models, only the latest windows are considered and the earlier ones are ignored, the overheads can be reduced, but still avoiding them could result in increasing size of windows or the volume of windows might go up.

Considering such implications, it is imperative that MFP notation which can compress the frequent patterns that are generated in to small forms that are compressed can be used for a mining process and many MFP based mining methods were proposed [12], [15- 16].

Vertical bitmap representation is adapted by MAFIA [20] which reduces the tree traversals. Once the bitmap is constructed, MAFIA can observe pattern’s frequency thorough and also the operation of bitmap despite that it might not traverse the trees. One of the significant methods of state-of-art MFP mining algorithm is FP Max [17] in which an additional data structure for mining MFPs in quick turnaround time was proposed. The model succeeds in decreasing tree traversal times in considerable manner. As FP-array has information of patterns, algorithm can more easily estimate them early even before the trees are actually traversed whilst working on the growth. Also, the technique not only reduces effectively the tree traversal operations, it also supports in pruning efficiency by curtailing any needless conditional trees being generated. However, the challenge in the aforesaid model is that they are profoundly the scan based processes and shall not be appropriate for process of data stream mining.

3. CONTEXT AWARE SLIDING WINDOW FOR FREQUENT ITEMSET MINING OVER DATA STREAM.

The main technique behind sliding window model is to keep window size fixed to mine real-time data streams. Several sliding window algorithms have been proposed for mine frequent itemsets by considering current data and ignoring past data. The main advantage of sliding window model reduced memory requirements and processing time. Sliding window is useful when user need to mine only current data without considering importance of history data and poses several mining challenges:

- The sliding window size fixed constant.
- It removes old data when new data arrived without analyzing whether significant of data changes have occurred or not.
- The size of the sliding defined by user without prior knowledge about the time and scale of the changes within the data.
- Sliding window models with fixed window and without prior knowledge about time will reflect mining results.

In regards to fulfil above observations and challenges, we proposed Context Aware Sliding Window (CASW) model for mining frequent itemsets over data streams. The sliding window (context window) will vary dynamically between minimum to maximum size based on content similarity observed in the input data streams. The proposed solution will be efficient in memory usage and processing time.

3.1 Model definition

Let *D* be the data stream generated sequence of transactions of size |D|. Now transaction *t* = (*tid, a*) is a tuple, where *tid* is a unique transaction-id and *a* is attribute set formed by transaction. Let *min* be the minimal size and *max* be the max size of window. Let w_c be the context window vary between *min* and *max* size and w_t be the transaction window fixed *min* constant size. Let *St* be the similarity threshold and *cv* be coverage (minimum support) defined by user to finalize w_c for frequent itemset mining. The initial values of *min max St,* and *cv* will be initialized at pre-processing step.

Now data stream *D* generated sequence of transactions $\{t_1, t_2, \ldots, t_i, \ldots\}$ of size |D| (total number of transaction generated). Initially, sequence of transactions will move into context window w_c followed by transaction window w_t. Whenever *min* count transactions moved into w_t then the model initiate content similarity analysis process to find content similarity between w_c , w_t .

3.2 Analyzing Content Similarity(CS)

The content similarity between context window w_c and transaction window w_t can be analyzed using hamming distance approach. Hamming distances is information analysis technique and very particle metric for measuring two strings or attribute sets distance based on content. It compares two strings or attributes and adds 0 to distance if both are similar else adds 1.

The value of hamming distance is observed in the model for denoting the content difference context window w_t and transaction window w_c. It is a one of the key method adapted for assessing the difference between elements in coding theory. Such strategy is applied for identifying the distance between unique attributes that are identified column wise in context window w_c and transaction window w_t .

Let $a(w_c)=\{c_{a1}, c_{a2},...,c_{am}\}$ be the unique column vector of context window w_c and $a(w_t)=\{t_{a1}, t_{a2},..., \text{t}_{\text{an}}\}$ be the unique column vector of transaction window w_t of size m and n respectively. If $m \neq n$ then accumulate constant value corresponding position in minimal size vector to make both vector same size ($m = n$).

Hamming distance column wise vector of $a(w_c), a(w_t)$ is assessed as follows:

$$hd_{a(wc)\leftrightarrow a(wt)} = \sum_{i=1}^{n} \begin{cases} 0 & if\,(c_{ai} \equiv t_{ai}) \\ 1 & \end{cases}$$

The average hamming distance of (w_c, w_t) computed by adding all columns wise hamming distance by dividing number of vectors formed:

$$hd(w_c \leftrightarrow w_t) = \frac{\sum_{k}^{cn} hda(w_c) \leftrightarrow a(w_t)}{cn}$$

In furtherance content similarity amidst of transaction window and context window cab be evaluated as:

$$S_{a(wc)\leftrightarrow a(wt)} = 1 - hd\,(w_c \leftrightarrow w_t)$$

If similarity scores $S_{a(wc)\leftrightarrow a(wc)}$ greater than or equal to the given similarity score threshold St then transactions of w_t will be moved to w_c by performing union operation. If w_c contains *Max* count transactions will be finalized for mining frequent itemsets by using TFIM, then w_c and w_t shall be cleared and process exploration for preparing window w_c will be sustained in further transactions streaming from data stream D. Even transaction window w_c contains less than *max* transactions finalized w_c for frequent mining by initiating TIFIM, then clear w_c and move transaction of w_t into w_c and w_t cleared for process next streaming transaction of data stream *D*.

The mining frequency itemsets finalized window will be carried out with Tree based Incremental Frequent Itemsets Mining over Data Streams (TIFIM) [36]. TIFIM is our first research contribution implemented by using compact data structure one hierarchy tree called bush to store useful information and incremental mining algorithm Frequent Itemset Finder (FIF) for mine frequent itemsets over data stream.

3.3 CASW algorithm

```
begin
  wc ← φ;
  wt ← φ;
  for i ← 1,2,...,|D| do
      if |wc| ≤ min then
          wc ← ti;
      else
          wt ← ti;
      end if
      if |wc| >= then
          S ← CS(wc, wt);
      end if
      if S ≥ St then
          wc ← wc + wt;
          if wc ≥ max then
              TIFIM(wc);
              wc ← φ;
              wt ← φ;
          end if
      else
          TIFIM(wc)
          wc ← φ;
          wc ← wt;
          wt ← φ;
     end if
   end for
end begin
```

3.4 Frequent Itemsets Mining using (TIFIM)

Primary representations pertaining to transaction of a data stream D has been detailed above. Asynchronous parallel processes are carried out for indentifying the frequent item sets in an incremental phase.

Bush indicates itemsets comprising two attribute pair like the attributes belonging to *wc* and transactions constituting the pair. For measuring the frequency of item sets, coverage (*cv)* can be defined by the user. Coverage of two attribute item sets with count of childs towards a bush is denoted for all the pair of attributes.

FIF (Frequent Itemset Finder) which is an asynchronous parallel process is performed as follows:

- Initially it selects the bushes comprising coverage more than the chosen coverage threshold cv.
- Develop new bushes from every two bushes by union of roots and intersects the Childs, and shall retain them in instance of a bush coverage which is new, being higher or equal to cv else discards.
- Process is continued until no new bush is formed.

3.4.1 Process of Pruning

A bush b_i considered to be sub-bush to bush b_j if $r_{b_i} \subseteq r_{b_j}$ any $cv_{(b_i)} \le cv_{(b_j)}$. As sub-bush b_i denoted by b_j, then bush b_i shall be pruned from the bush-set B.

3.4.2 Find frequent items

During an instance, frequent item sets are reviewed as:

Roots of bushes comprising higher value than the given cv shall be claimed as frequent itemset.

The bush b_i coverage shall be detected as:

If a bush b_j institute to be such that $b_i \subseteq b_j$ and coverage value of b_j turns higher than any other bush b_k such that $b_i \subseteq b_k$, in such instance, the coverage of b_i said to be $cv_{(b_j)} + cv_{(b_i)}$.

4. EMPIRICAL ANALYSIS

4.1. Characteristics of a Dataset

We evaluated the performance of CASW model with three metrics memory usage, execution time and scalability. We use two set of synthetic datasets generated by IBM data generator. Synthetic dataset contains 10000 transactions, with 125 different itemset and average size of transaction is 6 items. To achieve sparseness in the streaming transactions, the maximal window size ranges 100, 200, 300, 400, 500 and 600 and minimal window size ranges of 50, 100, 150, 200, 250, and 300, with minimum transaction.

4.2. Experimental results

Experimental results of the proposed solution are compared with the frequent item sets mining model for data streams that are devised in [29] the model of MFI (Matrix based frequent itemsets) and [36] TIFIM(Tree based Incremental Frequent Itemset

Mining over Streaming Data) are carried out in Java 7, and set of flat files as streaming data sources. Streaming environment is enumerated using Java RMI and also parallel process is adapted for the CASW which is achieved using Java Multi-threading concept. Three parameters for every synthetic dataset are cumulative of the aggregate volume of transactions. Every transaction of a data set is scanned only once in the experimental study, for simulating the data streams environment. For measuring computational cost and scalability, algorithms are performed under vivid coverage values in range of 2 to 8.

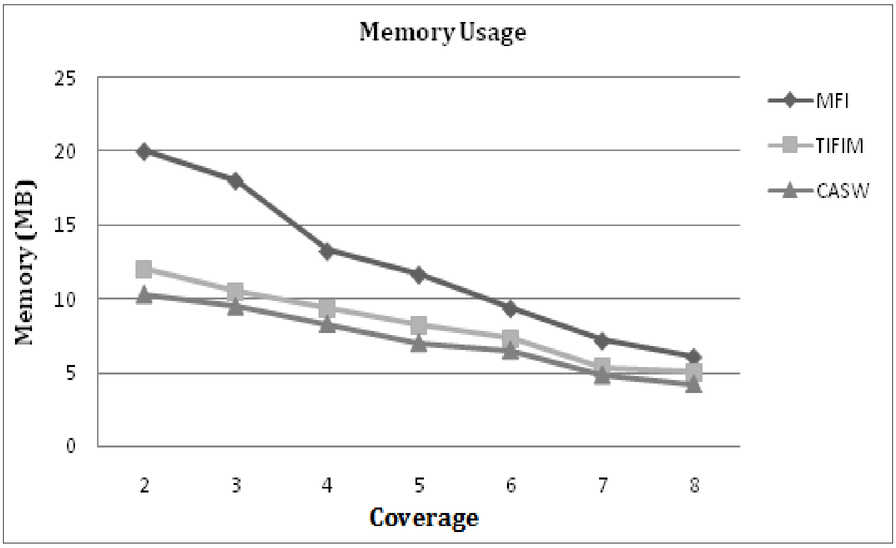

Figure 2: Memory usage of MFI, TIFIM and CASW

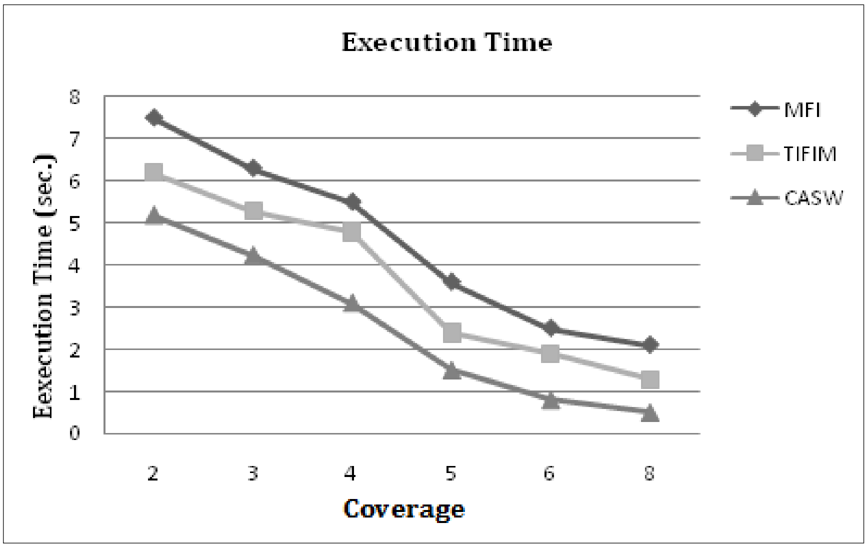

Figure 3: Execution time of MFI, TIFIM and CASW

Figure 2 and 3 depicts the comparative analysis of memory usage and execution time under vivid range of coverage values in range of 2 to 8 respectively. Figure 3 denotes the scalability in both the models of CASW and MFI and how CASW has outperformed compared to and MFI.

For graphical representation in Figure 2 and 3, the coverage given is denoted in horizontal axis and the memory in units of MBs and time mentioned in unit seconds are denoted in vertical axis respectively. Streaming of data size denoted in units of transactions is detailed in horizontal axis in Figure 4, and execution times denoted in units of seconds are represented in vertical axis. Also the percentage of time elapsed is also represented in units of seconds. From the Figure 2: represented memory usage in MBs coverage ranges 2 to 8 and average memory usage (MB) for MFI, TIFIM and the CASW are 13.7, 9.2 and 7.4 MBs respectively.

Figure 3 represent time execution of different coverage ranges 2 to 8 and average execution time of MFI, TIFIM and the CASW model are 4.5, 3.6 and 2.5 seconds respectively.

Performance of CASW is scalable as MFI represented in the Figure 4: maximal window size in horizontal axis ranges 100 to 600 and corresponding execution time is vertical axis. The average execution time of MFI and the CASW are 12.7 and 9.3 respectively based on maximal window size (no of maximal records per window).

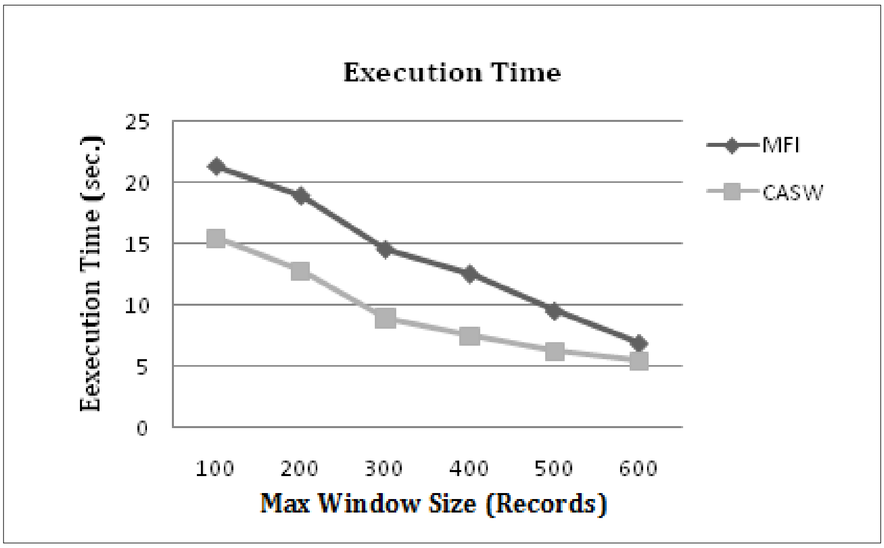

Figure 4: Scalability based on Maximum Window Size (Records).

5. CONCLUSION

In this paper, we proposed a scalable CASW (Context Aware Sliding Window) model for mining frequent itemset over streaming data. The size of sliding window dynamically varies between minimum to maximum size based on content similarity observed in the streaming data. We used hamming distance technique to analyze content similarity of streaming data. Using the content similarity analysis and fixing window size dynamically the proposed model has been more resourceful in terms of optimality and scalability. In this proposed, we used our earlier model TIFIM for mining frequent itemsets using the concept of cached bush structures and incremental mining.

In this paper, the solution is extended by introducing variable sliding window approach for streaming transactions to reduce memory usage and execution time. Experimental results from the study confirm that CASW model is scalable with coverage values and also with divergent streaming data size. In the future the solution can be extended to perform utility frequent itemset mining over data streams.

ACKNOWLEDGEMENT

The authors would like thanks to the anonymous reviewers for their valuable comments. We also grateful to the authors of all reference journals for valuable information and helping us set up the paper.

REFERENCES

[1] R. Agrawal and R. Srikant. "Fast algorithms for mining association rules", Proceedings of 20th VLDB Conference, 1994.

[2] J. Han, J. Pei and Y. Yin. "Mining frequent patterns without candidate generation", Proceedings of Conference on the Management of Data, ACM Press, 2000.

[3] K. Chuang, J. Huang and M. Chen. "Mining top-k frequent patterns in the presence of the memory constraint", The VLDB Journal, 1321-1344, 2007.

[4] Hua-Fu Li. "Interactive mining of top-k frequent closed itemsets from data streams", Expert Systems with Applications, 10779-10788, 2009.

[5] X. Zhang and Y. Zhang. "Sliding-window top-k pattern mining on uncertain streams." Journal of Computational Information Systems, 984-992, 2011.

[6] L. Chang, T. Wang, D. Yang, H. Luan and S. Tang. "Efficient algorithms for incremental maintenance of closed sequential patterns in large databases", Data & Knowledge Engineering, 68-106, 2009.

[7] M. Muhammal and R. Raman. "Mining sequential patterns from probabilistic databases", Knowledge and Information Systems, Springer, 325-358, 2015.

[8] Yun, Unil, Ryu, Ho K and Eunchul Yoon. "Weighted approximate sequential pattern mining within tolerance factors", Intelligent Data Analysis, 551-569, 2011.

[9] W. Liu, Y. Zheng, S. Chawla, J. Yuan and X. Xing. "Discovering spatio-temporal causal interactions in traffic data streams”, Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 1010-1018, 2011.

[10] Hua-Fu Li. "A sliding window method for finding top-k path traversal patterns over streaming web click-sequences." Expert Systems with Applications, 4382-4386, 2009.

[11] Hua-Fu Li. "DSM-PLW: Single-pass mining of path traversal patterns over streaming Web click-sequences", Computer Networks, 1474-1487, 2006.

[12] Z. Farzanyar, M. Kangavari and N. Cercone. "Max-FISM: Mining (recently) maximal frequent itemsets over data streams using the sliding window model", Computers & Mathematics with Applications, 1706-1718, 2012.

[13] S. K. Tanbeer, C.F. Ahmed, BS. Jeong and YK. Lee. "Efficient single-pass frequent pattern mining using a prefix-tree." Information Sciences, 559-583, 2009.

[14] S.K. Tanbeer, C.F. Ahmed, BS. Jeong and YK. Lee. "Sliding window-based frequent pattern mining over data streams", Information sciences, 3843-3865. 2009.

[15] D. Burdick, M. Calimlim, J. Flannick, J. Gehke and T.Yiu. "MAFIA: A maximal frequent itemset algorithm", IEEE Transactions on Knowledge and Data Engineering, 1490-1504, 2005.

[16] Y. Chen, R. Bie, and X. Chuan. "A new approach for maximal frequent sequential patterns mining over data streams", International Journal of Digital Content Technology and its Applications, 104-112, 2011.

[17] G. Grahne and J. Zhu J. "Fast algorithms for frequent itemset mining using FP-trees", IEEE Transactions on Knowledge and Data Engineering, 1347-1362, 2005.

[18] K. Gouda and M.J. Zaki . "GenMax: An efficient algorithm for mining maximal frequent itemsets", Data Mining and Knowledge Discovery, 223-242, 2005.

[19] Y. Huang, H. Xiong, W. Wu, P. Deng and Z. Zhang. "Mining maximal hyperclique pattern: a hybrid search strategy", Information Sciences 703-721, 2007.

[20] C. Luo C and S.M .Chung. "A scalable algorithm for mining maximal frequent sequences using a sample", Knowledge and Information Systems, 149-179, 2008.

[21] C. Luo C and S.M .Chung. "Parallel mining of maximal sequential patterns using multiple samples", The Journal of Supercomputing, 852-881, 2012.

[22] R. V. Priya, A.Vadivel and R. S. Thakur. "Maximal pattern mining using fast CP-tree for knowledge discovery", International Journal of Information Systems and Social Change (IJISSC), 56-74, 2012.

[23] S. Selvan and R. V. Nataraj. "Efficient mining of large maximal bicliques from 3D symmetric adjacency matrix", IEEE Transactions on Knowledge and Data Engineering, 1797-1802, 2010

[24] H. Shiozaki, T. Ozaki, and T. Ohkawa. "Mining closed and maximal frequent induced free subtrees", IEEE 6th International Conference Data Mining Workshops, 2006.

[25] L. T. Thomas, S.R. Valluri, and K. Karlapalem. "Margin: Maximal frequent subgraph mining", ACM Transactions on Knowledge Discovery from Data (TKDD), 2010.

[26] C. Yang, Y. Li, C. Zhang and Y. Hu. "A novel algorithm of mining maximal frequent pattern based on projection sum tree", IEET Fourth International Conference on Fuzzy Systems and Knowledge Discovery, 2007.

[27] U. Yun, H. Shin, KH. Ryu and EC. Yoon. "An efficient mining algorithm for maximal weighted frequent patterns in transactional databases", Knowledge-Based Systems, 53-64, 2012.

[28] X. Zeng, J. Pei, K. Wang K and J. Li. "PADS: A simple yet effective pattern-aware dynamic search method for fast maximal frequent pattern mining", Knowledge and Information Systems, 375-391, 2009.

[29] F. Guidan and Y. Shaohong. "A frequent itemsets mining algorithm based on matrix in sliding window over data streams", Intelligent System Design and Engineering Applications (ISDEA), IEEE, 2013.

[30] C.F. Ahmed, S.K. Tanbeer, BS. Jeong and YK. Lee. "An efficient algorithm for sliding window-based weighted frequent pattern mining over data streams", IEICE Transaction on Information and Systems, 1369-1381, 2009.

[31] H. Chen, LC. Shu, J. Xia and Q. Dengl. "Mining frequent patterns in a varying-size sliding window of online transactional data streams", Information Sciences, 15-36, 2012.

[32] M. Deypir, M.H. Sadreddini and H. Hashemi. "Towards a variable size sliding window model for frequent itemset mining over data streams", Computers & Industrial Engineering, 161-172, 2012.

[33] Li, Hua-Fu. "MHUI-max: An efficient algorithm for discovering high-utility itemsets from data streams", Journal of Information Science, 532-545, 2011.

[34] B. Mozafari, H. Thakkar, and C. Zaniolo. "Verifying and mining frequent patterns from large windows over data streams", IEEE 24th International Conference Data Engineering, 179-188, 2008.

[35] S. Bai-En, P. S. Yu and V. S. Tseng. "Efficient algorithms for mining maximal high utility itemsets from data streams with different models", Expert Systems with Applications, 12947-12960, 2012.

[36] V.sidda Reddy, Dr.T.V. Rao and Dr.A. Govardhan. "TIFIM: Tree based Incremental Frequent Itemset Mining over Streaming Data", International Journal of Computers & Technology, 1580-1586, 2013.

[37] C.H. Ahmed, S. K. Tanbeer, BS. Jeong and YK. Lee. "Single-pass incremental and interactive mining for weighted frequent patterns", Expert Systems with Applications, 7976-7994, 2012.

[38] L. Chen and C. Wang. "Continuous subgraph pattern search over certain and uncertain graph streams", IEEE Transactions on Knowledge and Data Engineering, 1093-1109, 2012.