2012 International Conference on Innovation and Information Management (ICIIM 2012)
IPCSIT vol. 36 (2012) © (2012) IACSIT Press, Singapore

Collective Sequential Pattern Mining in Distributed Evolving Data Streams

Amany F. Soliman[1,a+], Gamal A. Ebrahim[1,b] and Hoda K. Mohammed[1,c]

[1] Computer and Systems Engineering Department, Faculty of Engineering, Ain Shams University, Cairo, Egypt

Abstract. The advances in processing and communication techniques resulted in a multitude of emerging applications that interact with streams of data. Traditional data mining systems store arriving data, collect them for later mining, and make multiple passes over the collected data. Unfortunately, these systems are prohibitively slow when they deal with data streams with massive amounts of data arriving at high rates. This paper introduces a new model for mining sequential patterns on distributed data streams environments. It focuses on evolving data streams that originate from multiple distributed sources. Moreover, the mining process is achieved without compromising the privacy of the individual data streams of the participant nodes. Simulation results show that the proposed model scales linearly with the number of distributed nodes. In addition, it reduces the overhead in the distributed mining process.

Keywords: Sequential Pattern Mining, Collective Data Mining, Distributed Data Streams

1. Introduction

Mining sequential patterns in data streams has attracted significant attentions [1, 2, 3]. However, several needs of real-world applications that exhibit distributed nature of data streams have not been met. Several applications utilize data streams originating from multiple distributed sources such as online intrusion detection systems, sensor networks, and web-click streams. These applications are geographically distributed stream-oriented systems by their nature. Additionally, data from all sources should be gathered in a single location to reach the result. In several situations, the cost of centralizing data can be prohibitive since streaming data rates may exceed the capabilities of the storage, communication, and processing infrastructure. In addition, privacy issues may be encountered in such environments. Consequently, additional challenges may be imposed on the design and development of mining techniques to be able to interact with these huge continuous multiple data streams. Thus, there is a real need for a new model of mining sequential patterns in distributed data streams that preserves the privacy of the individual nodes. It mainly helps in the situations where participant nodes may not fully trust each other in terms of the distribution of their own data.

Sequential pattern mining in distributed databases has been addressed by several researchers [4, 5, 6, 7]. In addition, several sequential pattern-mining algorithms in data streams are introduced in [8, 9, and 10]. However, these approaches are not capable of dealing with distributed streams. Mainly because collecting data streams in a single location before processing them has a prohibitive cost and can compromise the owner's privacy. On the other hand, the problem of mining distributed data streams was addressed in many areas of data mining such as mining association rules [11, 12], clustering [13, 14], and mining frequent itemsets [15, 16]. Conversely, mining sequential patterns in multiple data streams is addressed in [17] and [18]. However, the MILE algorithm introduced in [17] is a one-time fashioned algorithm that cannot retain previous mining results. Hence, it may take prohibitive amount of time in re-mining. This problem has been

+ Corresponding author
E-mail address: [a] a_f_soliman@hotmail.com, [b] gamal.ebrahim@eng.asu.edu.eg, [c] hoda.korashy@eng.asu.edu.eg

avoided in IAspam algorithm introduced in [18] that incrementally mines across-streams sequential patterns to maintain the newest mining results. However, it utilizes a centralized setting strategy, which relies on a sliding window at the same working node to read and sample data streams. Meanwhile, a framework for Collective Data Mining (CDM) has been proposed in [19]. The main objective of CDM is to ensure that partial models produced from local data at the different sites are correct. Moreover, these models can be utilized as building blocks for forming a global model. To the best of our knowledge, no study has been achieved to mine sequential patterns from distributed data streams in a distributed manner, which is the major motivation behind this work. Instead of gathering all data streams from different sources on a single location and performing data mining, a new model based on collective data mining is presented in this paper. This model relies on multiple working nodes, which mine data streams locally and forward only the summarized information to a coordinator node for a global mining process.

The rest of the paper is organized as follows: next section presents the problem formulation followed by the details of the proposed model. Then, a theoretical analysis of the proposed model is introduced followed by a detailed simulation study. Finally, the paper is summarized in the last section.

2. Problem Formulation

Given a set of $d \geq 1$ data streams DS^1, DS^2 ... DS^d on d identical nodes N_1, N_2 ... N_d and each stream DS^i contains a stream of sequences $DS^i = s^i_1, s^i_2$ ... *etc.* Let DS be the sequence-preserving union of these streams $DS = DS^1 \cup DS^2 \cup \ldots \cup DS^d$. If α is a sequential pattern then α is a local frequent sequential pattern at node i if and only if the support of α (the ratio of the number of sequences containing α to the number of all sequences) in the local data stream DS^i is greater than or equal to the local minimum support σ_L. In addition, α will be a global frequent sequential pattern if and only if the support of α in DS is greater than or equal to the global minimum support σ_G.

The problem of collaborative sequential pattern mining is to find all sequential patterns GS, whose support value is equal to or greater than a fixed global minimum support threshold σ_G for the known part of DS at a given time. Furthermore, the problem of privacy-preserving sequential pattern mining in distributed data streams is to discover sequential patterns embedded in DS by imposing a major constraint on the mining process. This constraint is to prevent any sharing of private data among participant nodes during the mining process.

3. Proposed Model

The proposed model tries to minimize the communication requirements among the nodes. In addition, it does not need raw data exchange among participant nodes. All nodes are assumed identical and one of the nodes is selected randomly to act as a coordinator node. The coordinator node collects stream summary from the rest of the nodes. The proposed model accomplishes the mining process without compromising the privacy of individual data streams at different participant nodes.

The existence of a large number of participant nodes and the rapid incoming streams may lead to a significant bottleneck at the coordinator node. This is mainly because all participant nodes should periodically send some information to the coordinator node. Hence, this may result in excessive communication and high space requirements at the coordinator node for processing a large number of incoming sequences. The load on the coordinator node could be significantly reduced by arranging the nodes in a hierarchical communication structure. At any node in the intermediate level i, a set of frequent sequential patterns combined with their supports is received from the nodes at the lower level ($i - 1$). Then, integration is done at the intermediate node and a new set of frequent sequential patterns is mined with the level minimum support σ_i, where $\sigma_{i-1} < \sigma_i$. This process continues until the root node is reached and a set of global sequential patterns is mined. Consequently, the proposed model consists of three phases; mining sequential patterns at local nodes, an integration phase, and the global mining phase as shown in Fig. 1. On the other hand, Fig. 2 shows pseudo-code of each phase of the proposed model.

3.1. Mining Sequential Patterns at Local Nodes

SPEDS introduced by us in [20] is adopted to mine sequential patterns at local nodes. It uses a lexicographical tree *T* to hold the subsequences extracted from the data stream so far. Each path from the root of the tree to a node represents a sequence where the items are the nodes along the path. Each node in the tree contains a *tilted-time window table* constructed based on a logarithmic time scale. The entries of this table represent the numbers of sequences seen in the data stream in the corresponding tilted-time window.

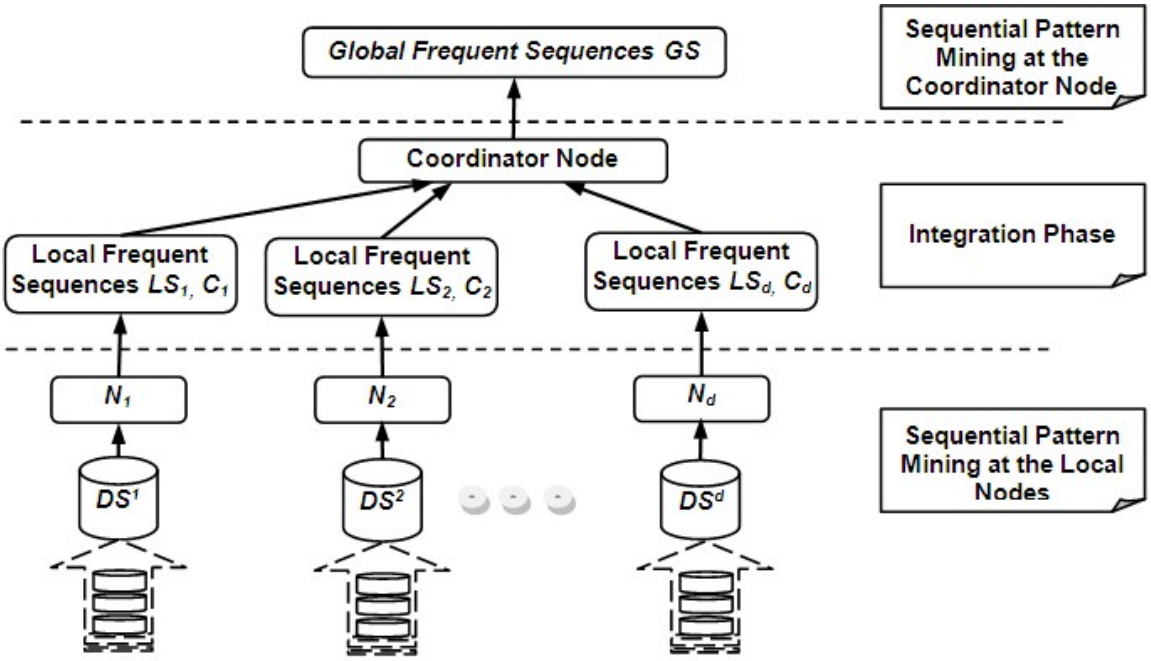

Fig.1. The architecture of the proposed model

At any participant node N_i corresponding to a data stream DS^i, a lexicographical tree T_i is constructed and SPEDS is applied locally to mine the set of local frequent sequences LS_i. It utilizes a local minimum support threshold σ_L, and a significance threshold ε $\left(0 \le \varepsilon \le \sigma_L\right)$. After processing a predefined number of batches *n* of local sequences at a participant node *i*, a set of local frequent sequences LS_i combined with a list C_i of their corresponding support are mined and forwarded to the coordinator node for further global mining. The support of a sequence *s* at any given time using the time fading scheme described in [20] is calculated by Eq. (1).

$$\left|s\right| = \sum_{\substack{1 \le k \le n, \\ f_k > 0}} \omega^{n-k} f_k + \sum_{\substack{1 \le k \le n \\ f_k = 0}} \omega^{n-k} \alpha L_k \ . \tag{1}$$

where the fading factor ω < 1, *n* is the number of all received batches, $f_1, f_2 \ldots f_n$ are the numbers of occurrences of *s* in tilted-time windows $t_1, t_2 \ldots t_n$ respectively, α is the batch support threshold $\left(0 \le \alpha \le \varepsilon\right)$, and L_k is the number of all sequences in the k^{th} tilted-time window.

A sequences *s* is frequent if its support $\left|s\right| \ge \left(\sigma - \varepsilon\right) N$, where σ is the minimum support threshold, ε is the significance threshold $\left(0 \le \varepsilon < \sigma\right)$, and *N* the number of sequences computed by:

$$N = \sum_{k=1}^{n} \omega^{n-k} L_k \ . \tag{2}$$

Mining local frequent sequential patterns at local node N_i

```
Repeat
  While the number of the received batches < n Do
      Apply SPEDS on the DS^l
  Calculate N_L using Eq. (2)
  For every sequence s in T_i Do
    Calculate |s| using Eq. (1)
    If |s| ≥ (σ_L - ε)N_L Then
       add s to LS_i and |s| to C_i
  Send LS_i combined with a list C_i to the coordinator node
  Initialize tree T_i
```

Integration phase at the coordinator node

```
For any incoming LS_i and C_i at time T_i Do
  For every sequence s_j with support c_j in LS_i and C_i Do
    If not end of pruning period Then
      If not end of window size Then
        search T_coordinator for s_j
          If s_j is not in T_coordinator Then
            insert s_j in T_coordinator
            set the corresponding count to zero
          Increment count of the corresponding node by c_j
      Else                                  //end of window
        Scan T_coordinator
        For each node in T_coordinator Do
          Update tilted-time table
          Set count to zero
    Else                                    //end of pruning period
      Prune tree
```

Mining global frequent sequential patterns at the coordinator node

```
Calculate N_G using Eq. (4)
For every sequence s in T_coordinator Do
        Calculate |s| using Eq. (3)
        If |s|≥ (σ_G - ε)N_G Then
              add s to GS
```

Fig.2. The pseudo-code of the three phases of the proposed model

Integration Phase. At the coordinator node, a lexicographical tree $T_{coordinator}$ similar to the one at local nodes is created and upon receiving LS_i and C_i from participant node i, an integration is started by inserting all incoming local frequent sequences into $T_{coordinator}$. SPEDS algorithm is modified and applied at the coordinator node to mine the set of global frequent sequences *GS*. Eq. (1) above is modified at the coordinator node to be:

$$|s| = \sum_{\substack{1\le k\le m,\\ f_k>0}} \omega^{m-k} f_k + \sum_{\substack{1\le k\le m\\ f_k=0}} \omega^{m-k}(\alpha_L - \varepsilon)N_L \ . \tag{3}$$

where m is the number of all received sequences and N_L is calculated using a modified version of Eq. (2), which is given by Eq. (4).

$$N_G = \sum_{k=1}^{m} \omega^{m-k} N_L \ . \tag{4}$$

The pruning period δ is represented by the number of incoming sequences, and it is a multiple of the window size. Every time the pruning period is up, the *tree* is pruned by calculating $|s|$ for each node using Eq. (3) and if $|s| \le \varepsilon N_G$ then the entire sub-tree rooted at this node is pruned.

Mining Global Sequential Patterns at the Coordinator Node. At the coordinator node, upon requesting the set of global frequent sequences *GS,* the time fading scheme presented in [20] is applied on $T_{coordinator}$ using the global minimum support σ_G, where $\sigma_G > \sigma_L$.

4. Theoretical Analysis

The local mining process at the participant nodes is based on SPEDS, which has been proven to produce no false negatives [20]. Assume there is a sequence s' that is frequent and was not included into the global frequent sequences. Two cases for s' can be identified:

1. s' was a frequent sequence at the local nodes $N_{i_1}, N_{i_2} \ldots N_{i_n}$ with frequencies $f_{i_1}, f_{i_2} \ldots f_{i_n}$ in the tilted-time windows $i_1, i_2 \ldots i_3$ at the coordinator node respectively. Hence, $f_{i_1}, f_{i_2} \ldots f_{i_n} \geq (\sigma_L - \varepsilon)N_L$, where σ_L is the local minimum support and N_L is calculated using Eq. (2).
2. s' was not frequent in the remaining local nodes $N_{j_1}, N_{j_2} \ldots N_{j_n}$ with frequencies $f_{j_1}, f_{j_2} \ldots f_{j_n}$ in tilted-time windows $j_1, j_2 \ldots j_3$ at the coordinator node respectively. Hence, $f_{j_1}, f_{j_2} \ldots f_{j_n} < (\sigma_L - \varepsilon)N_L$ and the corresponding counts in entries $j_1, j_2 \ldots j_3$ of its tilted-time window table would be zeros. However, during calculating the sequence support using Eq. (3), its frequency is assumed to be $(\sigma_L - \varepsilon)N_L$, which is greater than the true count of the sequence at the corresponding nodes. If such a sequence is not pruned from the tree then its support calculated using Eq. (3) is greater than the true count of the sequence. In this case, if it was not generated as an output then it would not be frequent. If such a sequence was pruned n times from the tree then every time it is pruned from the tree $|s'| \leq \varepsilon N_G$, where ε is the significance threshold $(0 \leq \varepsilon < \sigma_G)$ and $|s'|$ is its support calculated by Eq. (3). Consequently, if it was not an output, it would not be frequent. Hence, sequence s' does not exist, which leads to the following claim:

Claim 1. *The proposed model produces no false negatives.*

SPEDS guarantees that all false positives have true support of at least $(\sigma - \varepsilon)N$. Where N is the stream length, σ is the minimum support threshold, and ε is the significance threshold $(0 \leq \varepsilon < \sigma)$. To calculate the minimum support of false positives in the proposed model for mining sequential patterns from distributed data streams, let us first calculate the maximum error e_{max} in the support of a sequence using Eq. (3). Assume that there is a sequence s' that has the minimum support at the coordinator node, for such a sequence to be an output from a local node and forwarded to the coordinator node, its minimum local support at the local node(s) should be $(\sigma_L - \varepsilon)N_L$. Hence, the minimum global support $|s'_{\min}|$ for any sequence in $T_{coordinator}$ using Eq. (3) is $|s'_{\min}| = \sum_{1 \leq k \leq m} \omega^{m-k}(\sigma_L - \varepsilon)N_L$ and from Eq. (4): $|s'_{\min}| = (\sigma_L - \varepsilon)N_G$. This leads to $e_{\max} \leq (\sigma_L - \varepsilon)N_G$. Consequently, for any sequence s' that has true support c with Eq. (3) utilized to calculate $|s'|$ then:

$$|s'| \leq c + (\sigma_L - \varepsilon)N_G. \tag{5}$$

If s' was included in the output then $|s'| \geq (\sigma_G - \varepsilon)N_G$, which is the minimum support for a sequence to be an output. On the other hand, from Eq. (5): $|s'| \leq c + (\sigma_L - \varepsilon)N_G$, hence, for any sequence s' to be an output it should have true count c satisfying $c + (\sigma_L - \varepsilon)N_G \geq (\sigma_G - \varepsilon)N_G$. Consequently, it leads to the following claim:

Claim 2. *In the proposed model, all false positives have true support of at least* $(\sigma_G - \sigma_L)N_G$.

5. Experimental Results

Several experiments have been carried out with a fading factor $\omega = 0.5$, a batch size of 100 sequences, $\alpha = 0.0095$, and $\varepsilon = 0.01$. Fig. 3 shows the results of this experimental study.

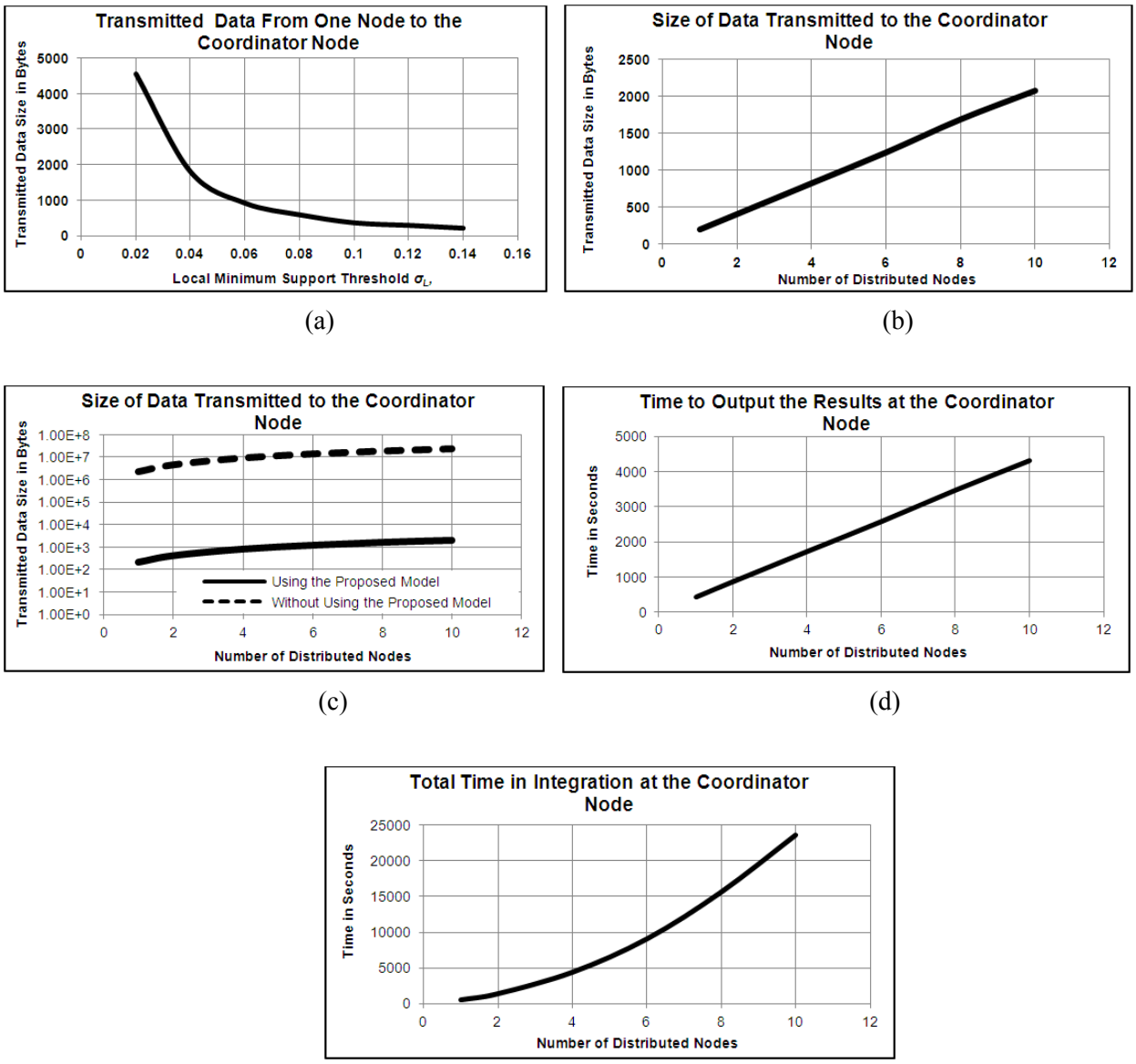

Fig.3 Results of the experimental study

Synthetic data streams are adopted in the experiments. They have been generated using the sequential data generator introduced in *IlliMine* system package. The first experiment is carried out to show the effect of changing local minimum support σ_L on the size of transmitted data from any distributed node. The number of processed batches at any distributed node is 1000 batches, σ_L is varied and the average size of transmitted data to the coordinator node in bytes is measured.

The results of this experiment are shown in Fig. 3a. As shown in this figure, if σ_L is high then there will be only a limited number of sequential patterns and, hence, the pattern length will be short.

The second experiment is carried out to assess the communication load at the coordinator node. In this case, the number of distributed nodes is changed and the communication load at the coordinator node is measured in terms of the size of received data from all distributed nodes. The number of processed batches at any distributed node is 1000 batches, and σ_L = 0.14. As shown in Fig. 3b the communication load scales linearly with the number of distributed nodes. These results are compared with the communication load at the coordinator node if the mining process is performed in a centralized setting (Fig. 3c). As shown in this figure, there is a considerable reduction of the data transmitted to the coordinator node when adopting the

proposed model. This is mainly because the coordinator node only collects the stream summary from the participant nodes. This reduction in the transmitted data contributes to the scalability of the proposed model.

The third experiment is carried out to analyze the timing requirements at the coordinator node for the integration and mining of the global frequent sequences and generating the results. The number of processed batches at any distributed node is 1000 batches, σ_L = 0.14 and σ_G = 0.2. The number of distributed nodes is changed during the experiment. Then, the execution time of the integration phase and time required to generate the results at the coordinator node are measured. As shown in Fig. 3d, the average execution time for outputting the global frequent sequences scales linearly with the number of distributed nodes, while Fig. 3e shows that the execution time of integrating local frequent sequences at the coordinator node increases with the number of distributed nodes.

6. Summary

Most prior work on sequential pattern mining in data streams considers the case of a single stream or multiple data streams in a centralized setting. Hence, a distributed data stream mining model has been proposed in this paper that is able to mine sequential patterns from multiple distributed evolving data streams. It is proven that the proposed model produces no false negatives and imposes a lower bound of the support of false positives. Simulation study has been carried out to analyze the performance of the proposed model. Simulation results show that the proposed model reduces the communication overhead in the distributed mining process. Most importantly, it scales linearly with the number of distributed nodes, which contributes to the scalability of the proposed model.

7. References

[1] L. Vinceslas, J.-E. Symphor, A. Mancheron, and P. Poncelet: SPAMS: A Novel Incremental Approach for Sequential Pattern Mining in Data Streams. Advances in Knowledge Discovery and Management Vol. 292 (2010), pp. 201-216.

[2] P.A. Laur, J.-E. Symphor, R. Nock, and P. Poncelet: Mining Sequential Patterns on Data Streams: A Near-Optimal Statistical Approach. In Proc. of the 2nd International Workshop on Knowledge Discovery from Data Streams (2005).

[3] A. Marascu and F. Masseglia: Mining Sequential Patterns from Data Streams: A Centroid Approach. Journal of Intelligent Information Systems Vol. 27 (2006), pp. 291-307.

[4] V. Kapoor, P. Poncelet, F. Trousset, and M. Teisseire: From Collaborative to Privacy-Preserving Sequential Pattern Mining, Privacy and Anonymity in Information Management Systems. Advanced Information and Knowledge Processing Part 3 (2010), pp. 135-156.

[5] C. Zhang, K. Hu, Z. Chen, L. Chen, and Y. Dong: ApproxMGMSP: A Scalable Method of Mining Approximate Multidimensional Sequential Patterns on Distributed System. In: Proc. of the 4th International Conference of Fuzzy Systems and Knowledge Discovery (2007), pp. 730-734.

[6] H.-C. Kum, J. H. Chang, and W. Wang: Sequential Pattern Mining in Multi-Databases via Multiple Alignment. Journal of Data Mining and Knowledge Discovery Vol. 12 (2006), pp. 151-180.

[7] V. Kapoor, P. Poncelet, F. Trousset, and M. Teisseire: Privacy Preserving Sequential Pattern Mining in Distributed Databases. In: Proc. of the 15th ACM International Conference on Information and knowledge management (2006), pp. 758-767.

[8] C. Raıssi and P. Poncelet: Random Sampling Over Data Streams for Sequential Pattern Mining. In: Proc. of the 1st European Workshop on Data Streams (2007), pp. 61-66.

[9] L.F. Mendes, B. Ding, and J. Han: Stream Sequential Pattern Mining with Precise Error Bounds. In: Proc. of the IEEE International Conference on Data Mining (2008), pp. 941-946.

[10] C.-C. Ho, H.-F. Li, F.-F. Kuo, and S.-Y. Lee: Incremental Mining of Sequential Patterns over a Stream Sliding Window. In: Proc. of the 6th IEEE International Conference on Data Mining (2006), pp.677-681.

[11] M. Zhang, and G. Mao: An Approach for Distributed Streams Mining Using Combination of Naïve Bayes and Decision Trees. In: Proc. of the 3rd International Conference on Advances in Databases, Knowledge, and Data Applications (2011), pp. 29-33.

[12] G. Folino, C. Pizzuti, and G. Spezzano: Mining Distributed Evolving Data Streams using Fractal GP Ensembles. In: Proc. of the 10th European Conference on Genetic Programming (2007), pp. 160-169.

[13] A. Ciampi, A. Appice, and D. Malerba: Discovering Trend-Based Clusters in Spatially Distributed Data Streams. International Workshop of Mining Ubiquitous and Social Environments (2010), pp 107-122.

[14] R.Wolff, K. Bhaduri, and H. Kargupta: A Generic Local Algorithm for Mining Data Streams in Large Distributed Systems. IEEE Transactions on Knowledge and Data Engineering Vol. 21 (2009), pp. 465-478.

[15] E.T. Wang, and A.L. Chen: Mining Frequent Itemsets over Distributed Data Streams by Continuously Maintaining a Global Synopsis. Data Mining and Knowledge Discovery Vol. 23 (2011), pp. 252-299.

[16] E. Cesario, A. Grillo, C. Mastroianni, and D. Talia: A Sketch-Based Architecture for Mining Frequent Items and Itemsets from Distributed Data Streams. In: Proc. of the 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (2011), pp. 245-253.

[17] G. Chen, X. Wu, and X. Zhu: Sequential Pattern Mining in Multiple Streams. In: Proc. of the 5th IEEE International Conference on Data Mining (2005).

[18] S. Yang, C. Chao, P. Chen, and C. Sun: Incremental Mining of Across-streams Sequential Patterns in Multiple Data Streams. Journal of Computers Vol. 6 (2011), pp.449-457.

[19] H. Kargupta, B.-H. Park, D. Hershberger, and E. Johnson: Collective Data Mining: A New Perspective Towards Distributed Data Mining. In: Advances in Distributed and Parallel Knowledge Discovery, editied by H. Kargupta and P. Chan, chapter, 5, AAAI Press / The MIT Press (2000).

[20] A.F. Soliman, G.A. Ebrahim, and H.K. Mohammed: SPEDS: A Framework for Mining Sequential Patterns in Evolving Data Streams. In: Proc. of the IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (2011), pp. 464-469.