Research Issues in Data Stream Association Rule Mining

Nan Jiang and Le Gruenwald
The University of Oklahoma, School of Computer Science, Norman, OK 73019, USA
Email: {nan_jiang, ggruenwald} @ou.edu

ABSTRACT

There exist emerging applications of data streams that require association rule mining, such as network traffic monitoring and web click streams analysis. Different from data in traditional static databases, data streams typically arrive continuously in high speed with huge amount and changing data distribution. This raises new issues that need to be considered when developing association rule mining techniques for stream data. This paper discusses those issues and how they are addressed in the existing literature.

1. INTRODUCTION

A data stream is an ordered sequence of items that arrives in timely order. Different from data in traditional static databases, data streams are continuous, unbounded, usually come with high speed and have a data distribution that often changes with time [Guha, 2001]. As the number of applications on mining data streams grows rapidly, there is an increasing need to perform association rule mining on stream data. An association rule is an implication of the form $X \Rightarrow Y$ (s, c), where X and Y are frequent itemsets in a transactional database and $X \cap Y = \varnothing$, s is the percentage of records that contain both X and Y in the database, called support of the rule, and c is the percentage of records containing X that also contain Y, called the confidence of the rule. Association rule mining is to find all association rules the support and confidence of which are above or equal to a user-specified minimum support and confidence, respectively.

One example application of data stream association rule mining is to estimate missing data in sensor networks [Halatchev, 2005]. Another example is to predict frequency estimation of Internet packet streams [Demaine, 2002]. In the MAIDS project [Cai, 2004], this technique is used to find alarming incidents from data streams. Association rule mining can also be applied to monitor manufacturing flows [Kargupta, 2004] to predict failure or generate reports based on web log streams, and so on.

Data streams can be further classified into offline streams and online streams. Offline streams are characterized by regular bulk arrivals [Manku, 2002]. Among the above examples, generating reports based on web log streams can be treated as mining offline data streams because most of reports are made based on log data in a certain period of time. Other offline stream examples include queries on updates to warehouses or backup devices. Queries on these streams are allowed to be processed offline.

Online streams are characterized by real-time updated data that come one by one in time. From the above examples, predicting frequency estimation of Internet packet streams is an application of mining online data streams because Internet packet streams is a real-time one packet by one packet process. Other online data streams are stock tickers, network measurements and sensor data. They have to be processed online and must keep up with the rapid speed of online queries. They have to be discarded right after arrived and being processed. In addition, unlike with offline data streams, bulk data processing is not possible for online stream data.

Due to the characteristics of stream data, there are some inherent challenges for stream data association rule mining. First, due to the continuous, unbounded, and high speed characteristics of data streams, there is a huge amount of data in both offline and online data streams, and thus, there is not enough time to rescan the whole database or perform a multi-scan as in traditional data mining algorithms whenever an update occurs. Furthermore, there is not enough space to store all the stream data for online processing. Therefore, a one scan of data and compact memory usage of the association rule mining technique are necessary. Second, the mining method of data streams needs to adapt to their changing data distribution; otherwise, it will cause the concept drifting problem [Wang, 2003], which we will discuss in Section 2.3.1. Third, due to the high speed characteristics of online data streams, they need to be processed as fast as possible; the speed of the mining algorithm should be faster than the data coming rate, otherwise data approximation techniques, such as sampling and load shedding, need to be applied which will decrease the accuracy of the mining results. Fourth, due to the continuous, high speed, and changing data distribution characteristics, the analysis results of data streams often keep changing as well. Therefore, mining of data streams should be an incremental process to keep up with the highly update rate, i.e. new iterations of mining results are built based on old mining results so that the results will not have to be recalculated each time a user's request is received. Fifth, owing to the unlimited amount of stream data and limited system resources, such as memory space and CPU power, a mining mechanism that adapts itself to available resources is needed; otherwise, the accuracy of the mining results will be decreased.

Traditional association rule mining algorithms are developed to work on static data and, thus, can not be applied directly to mine association rules in stream data. The first recognized frequent itemsets mining algorithm

for traditional databases is Apriori [Agrawal, 1993]. After that, many other algorithms based on the ideas of Apriori were developed for performance improvement [Agrawal, 1994, Han, 1999]. Apriori-based algorithms require multiple scans of the original database, which leads to high CPU and I/O costs. Therefore, they are not suitable for a data stream environment, in which data can be scanned only once. Another category of association rule mining algorithms for traditional databases proposed by Han and Pei [Han, 2000] are those using a frequent pattern tree (FP-tree) data structure and an FP-growth algorithm which allows mining of frequent itemsets without generating candidate itemsets. Compared with Apriori-based algorithms, it achieves higher performance by avoiding iterative candidate generations. However, it still can not be used to mine association rules in data streams since the construction of FP-tree requires two scans of data.

As more and more applications generate a large amount of data streams every day, such as web transactions, telephone records, and network flows, much research on how to get frequent items, patterns and association rules in a data stream environment has been conducted [Chang, 2003, Chang, 2004, Charikar, 2004, Chi, 2004, Cormode, 2003, Demaine, 2002, Giannella, 2003, Huang, 2002, Jin, 2003, Karp, 2003, Li, 2004, Lin, 2005, Manku, 2002, Relue, 2001, Yang, 2004, Yu, 2004]. However, these algorithms are focused on one or more application areas, and none of them fully addresses the issues that need to be solved in order to mine association rules in data streams.

In [Gaber, 2005], Gaber et al briefly discussed some general issues concerning stream data mining. They did not provide a thorough discussion for issues that need to be considered in the specific area of data stream association rule mining; they merely addressed the state of the art solutions. In this paper, we focus on research issues concerning association rule mining in data streams and, whenever possible, review how they are handled in the existing literature.

The rest of this paper is organized as follows. Section 2 discusses general issues that need to be considered for all data association rule mining algorithms for data streams. Section 3 describes application dependent issues. Section 4 summarizes the merits and lessons learned from the existing studies and concludes the paper.

2. GENERAL ISSUES IN DATA STREAM ASSOCIATION RULE MINING

The characteristics of data streams as pointed out in Section 1 indicate that when developing association rule mining techniques, there are more issues that need to be considered in data streams than in traditional databases. In this section, general issues are discussed. These issues are crucial and need to be taken into account in all applications when developing an association rule mining technique for stream data.

2.1. Data Processing Model

The first issue addresses which parts of data streams are selected to apply association rule mining. From the definition given in Section 1, data streams consist of an ordered sequence of items. Each set of items is usually called “transaction”. The issue of data processing model here is to find a way to extract transactions for association rule mining from the overall data streams. Because data streams come continuously and unboundedly, the extracted transactions are changing from time to time.

According to the research of Zhu and Shasha [Zhu, 2002], there are three stream data processing models, Landmark, Damped and Sliding Windows. The Landmark model mines all frequent itemsets over the entire history of stream data from a specific time point called landmark to the present. A lot of research has been done based on this model [Charikar, 2004, Cormode, 2003, Jin, 2003, Karp, 2003, Li, 2004, Manku, 2002, Yang, 2004, Yu, 2004]. However, this model is not suitable for applications where people are interested only in the most recent information of the data streams, such as in the stock monitoring systems, where current and real time information and results will be more meaningful to the end users.

The Damped model, also called the Time-Fading model, mines frequent itemsets in stream data in which each transaction has a weight and this weight decreases with age. Older transactions contribute less weight toward itemset frequencies. In [Chang, 2003] and [Giannella, 2003], they use exactly this model. This model considers different weights for new and old transactions. This is suitable for applications in which old data has an effect on the mining results, but the effect decreases as time goes on.

The Sliding Windows model finds and maintains frequent itemsets in sliding windows. Only part of the data streams within the sliding window are stored and processed at the time when the data flows in. In [Chang, 2004, Chi, 2004, Lin, 2005], the authors use this concept in their algorithms to get the frequent itemsets of data streams within the current sliding window. The size of the sliding window may be decided according to applications and system resources. The mining result of the sliding window method totally depends on recently generated transactions in the range of the window; all the transactions in the window need to be maintained in order to remove their effects on the current mining results when they are out of range of the sliding window.

All these three models have been used in current research on data streams mining. Choosing which kind of data process models to use largely depends on application needs. An algorithm based on the Landmark model can be converted to that using the Damped model by adding a decay function on the upcoming data streams. It can also be converted to that using Sliding Windows by keeping track of and processing data within a specified sliding window.

2.2. Memory Management

The next fundamental issue we need to consider is how to optimize the memory space consumed when running the mining algorithm. This includes how to decide the information we must collect from data streams and how to choose a compact in-memory data structure that allows the information to be stored, updated and retrieved efficiently. Fully addressing these issues in the mining algorithm can greatly improve its performance.

2.2.1. Information to Be Collected and Stored in Memory

Classical association rule mining algorithms on static data collect the count information for all itemsets and discard the non-frequent itemsets and their count information after multiple scans of the database. This would not be feasible when we mine association rules in stream data due to the two following reasons. First, there is not enough memory space to store all the itemsets and their counts when a huge amount of data comes continuously. Second, the counts of the itemsets are changing with time when new stream data arrives. Therefore, we need to collect and store the least information possible, but enough to generate association rules.

In [Karp, 2003], the most frequent items and their counts are stored in the main memory. This technique stores the most important information. However, because it discards infrequent items and their counts and discarded items may become frequent in the future, it cannot get the information associated with non-frequent items when later they become frequent. In [Yang, 2004], the available computer memory is used to keep frequency counts of all short itemsets (itemsets with $k \leq 3$, where k is the maximum size of frequent itemsets), thus the association rule mining for short itemsets in data streams becomes trivial. But as pointed out by the authors, this technique only suits limited applications where $k \leq 3$ and $n \leq 1800$ (n is the total number of data items). We can see that there is a trade off between the information we collect and the usage of system resources. The more information we collect to get more accurate results, the more memory space we use and the more processing time is needed.

2.2.2. Compact Data Structure

An efficient and compact data structure is needed to store, update and retrieve the collected information. This is due to bounded memory size and huge amounts of data streams coming continuously. Failure in developing such a data structure will largely decrease the efficiency of the mining algorithm because, even if we store the information in disks, the additional I/O operations will increase the processing time. The data structure needs to be incrementally maintained since it is not possible to rescan the entire input due to the huge amount of data and requirement of rapid online querying speed.

In [Manku, 2002], a lattice data structure is used to store itemsets, approximate frequencies of itemsets, and maximum possible errors in the approximate frequencies. In [Li, 2004], the authors employ a prefix tree data structure to store item ids and their support values, block ids, head and node links pointing to the root or a certain node. In [Giannella, 2003], a FP-tree is constructed to store items, support information and node links. A proper data structure is a crucial part of an efficient algorithm since it is directly associated with the way we handle newly arrived information and update old stored information. A small and compact data structure which is efficient in inserting, retrieving and updating information is most favorable when developing an algorithm to mine association rules for stream data.

2.3. One Pass Algorithm to Generate Association Rules

Another fundamental issue is to choose the right type of mining algorithms. Association rules can be found in two steps: 1) finding large itemsets (support is ≥ user specified support) for a given threshold support and 2) generate desired association rules for a given confidence. In the following subsections, we discuss the issues that need to be considered to generate and maintain frequent itemsets and association rules in data streams.

2.3.1. Frequent Itemsets

There exist a number of techniques for finding frequent itemsets in data streams. Based on the result sets produced, stream data mining algorithms can be categorized as exact algorithms or approximate algorithms.

In exact algorithms, the result sets consist of all of the itemsets the support values of which are greater than or equal to the threshold support. In [Karp, 2003] and [Yang, 2004], the authors use the exact algorithms to generate the result frequent itemsets. It is important for many applications to know the exact answers of the mining results; however, additional cost is needed to generate the accurate result set when the processing data is huge and continuous. The technique proposed in [Karp, 2003] takes two scans to generate the exact result set, and in [Yang, 2004], the algorithm generated can only mine short itemsets, which cannot be applied to large itemsets. Another option to get the exact mining results with relatively small memory usage is to store and maintain only special frequent itemsets, such as closed or maximal frequent itemsets, in memory. In [Chi, 2004] and [Mao, 2005], the authors proposed algorithms to maintain only closed frequent itemsets and maximal frequent itemsets over a sliding window and landmark processing model, respectively. In both of these cases, how we can get all the information to further generate association rules based on these special itemsets is an additional issue that needs to be considered.

Approximate algorithms generate approximate result sets with or without an error guarantee. Approximate mining frequent patterns with a probabilistic guarantee can take two possible approaches: false positive oriented and false negative oriented. The former includes some infrequent

patterns in the result sets, whereas the latter misses some frequent patterns [Yu, 2004].

Since data streams are rapid, time-varying streams of data elements, itemsets which are frequent are changing as well. Often these changes make the model built on old data inconsistent with the new data, and frequent updating of the model is necessary. This problem is known as concept drifting [Wang, 2003]. From the aspect of association rule mining, when data is changing over time, some frequent itemsets may become non-frequent and some non-frequent itemsets may become frequent. If we store only the counts of frequent itemsets in the data structure, when we need the counts for potential non-frequent itemsets which would become frequent itemsets later, we cannot get this information. Therefore, the technique to handle concept drifting needs to be considered. In [Chi, 2004], Chi et al proposed a method to reflect the concept drifts by boundary movements in the closed enumeration tree (CET).

From the above discussions, we can see that when designing a stream data association rule mining algorithm, we need to answer a number of questions: should we use an exact or approximate algorithm to perform association rule mining in data streams? Can its error be guaranteed if it is an approximate algorithm? How to reduce and guarantee the error? What is the tradeoff between accuracy and processing speed? Is data processed within one pass? Can this algorithm handle a large amount of data? Up to how many frequent itemsets can this algorithm mine? Can this algorithm handle concept drifting and how?

In the current works published in this area, [Karp, 2003] [Yang, 2004] [Chi, 2004] and [Mao, 2005] proposed exact algorithms, while [Li, 2004], [Yu, 2004], [Chang, 2004], [Manku, 2002], [Charikar, 2004] and [Giannella, 2003] proposed approximate algorithms. Among them [Yu, 2004] uses the false negative method to mine association rules, while the other approximate algorithms use the false positive method. [Chi, 2004] considered the concept drifting problem in its proposed algorithm.

2.3.2. Mechanism to Maintain and Update Association Rules

The next step after we get frequent itemsets is to generate and maintain desired association rules for a given confidence. As we can see from the previous discussions, mining association rules involves a lot of memory and CPU costs. This is especially a problem in data streams since the processing time is limited to one online scan. Therefore, when to update association rules, in real time or only at needs, is another fundamental issue.

The problem of maintaining discovered association rules was first addressed in [Cheung, 1996]. The authors proposed an incremental updating technique called FUP to update discovered association rules in a database when new transactions are added to the database. A more general algorithm, called FUP2, was proposed later in [Cheung, 1997] which can update the discovered association rules when new transactions are added to, delete from, or modified in the database. However in a data stream environment, stream data are added continuously, and therefore, if we update association rules too frequently, the cost of computation will increase drastically.

In [Lee, 1997], the authors proposed an algorithm, called DELI, which uses a sampling technique to estimate the difference between the old and new association rules. If the estimated difference is large enough, the algorithm signals the need of an update operation; otherwise, it takes the old rules as an approximation of the new rules. It considers the difference in association rules, but does not consider the performance of incremental data mining algorithms for evolving data, which is especially the situation in data stream mining. [Zheng, 2003] proposed a metric distance as a difference measure between sequential patterns and used a method, called TPD, to decide when to update the sequential patterns of stream data. The authors suggested that some initial experiments be done to discover a suitable incremental ratio and then this ratio be used to decide when would be better to update sequential patterns. The TPD method is only suitable for streams with little concept drifting, that is to say the change of data distribution is relatively small.

2.4. Resource Aware

Resources such as memory space, CPU, and sometimes energy, are very precious in a stream mining environment. They are very likely to be used up when processing data streams which arrive with rapid speed and a huge amount. What should we do when the resources are nearly consumed? If we totally ignore the resources available, for example the main memory, when processing the mining algorithm, data will be lost when the memory is used up. This would lead to the inaccuracy of the mining results, thus degrade the performance of the mining algorithm. Shall we just shed the incoming data or adjust our technique to handle this problem?

In [Gaber, 2003, Gaber, 2004, Teng, 2004], the authors discussed this issue and proposed their solutions for resource-aware mining. Gaber et al. proposed an approach, called AOG, which uses a control parameter to control its output rate according to memory, time constrains and data stream rate [Gaber, 2003, Gaber, 2004]. Teng et al. proposed an algorithm, called RAM-DS, to not only reduce the memory required for data storage but also retain good approximation of temporal patterns given limited resources like memory space and computation power [Teng, 2004].

3. APPLICATION DEPENDENT ISSUES

Different data stream application environments may have different needs for an association rule mining algorithm. In this section, we discuss issues that are application dependent.

3.1. Timeline Query

Stream data come continuously over time. In some applications, user may be interested in getting association rules based on the data available during a certain period of time. Then the storage structure needs to be dynamically adjusted to reflect the evolution of itemset frequencies over time. How to efficiently store the stream data with timeline and how to efficiently retrieve them during a certain time interval in response to user queries is another important issue.

In [Giannella, 2003], the authors proposed a method to incrementally maintain tilted-time windows for each pattern at multiple time granularities, which is convenient for applications where users are more interested in getting detailed information from the recent time period. In [Lin, 2005] a time-sensitive sliding window model is created to mine and maintain the frequent itemsets during a user defined time interval.

3.2. Multidimensional Stream Data

In applications where stream data are multi-dimensional in nature, multi-dimensional processing techniques for association rule mining need to be considered. Take a sensor data network as an example and assume that it gets and distributes the weather information. It is possible that when the temperature for one sensor S goes up, its humidity will decrease and the temperature from the sensors in close vicinity and toward the same wind direction of the sensor S will also increase. Here, temperature and humidity are the multidimensional information of the sensor. How to efficiently store, update and retrieve the multidimensional information to mine association rules in multidimensional data streams is an issue we need to consider in this situation.

[Pinto, 2001] proposed a method to integrate multidimensional analysis and sequential data mining, and [Yu, 2005] proposed an algorithm to find sequential patterns from d-dimensional sequence data, where $d > 2$.

3.3. Online Interactive Processing

In some applications, users may need to modify the mining parameters during the processing period, especially when processing data streams because there is not a specific stop point during the mining process. Therefore, how to make the online processing interactive according to user inputs before and during the processing period is another important issue.

In [Parthasarathy, 1999], the authors presented techniques for maintaining frequent sequences upon database updates and user interaction and without re-executing the algorithm on the entire dataset. In [Veloso, 2003], the interactive approach makes use of selective updates to avoid updating the entire model of frequent itemsets. Ghoting and Parthasarathy proposed a scheme in [Ghoting, 2004] which gives controlled interactive response times when processing distributed data streams.

3.4. Distributed Environment

In a distributed environment, stream data comes from multiple remote sources. Such an environment imposes excessive communication overhead and wastes computational resources when data is dynamic. In this situation, how to minimize the communication cost, how to combine frequency counts from multiple nodes, and how to mine data streams in parallel and update the associated information incrementally are additional issues we need to consider.

Otey discussed this problem and presented an approach making use of parallel and incremental techniques to generate frequent itemsets of both local and global sites in [Otey, 2003] and [Otey, 2004]. In [Veloso, 2003b], the authors proposed a distributed algorithm which imposes low communization overhead for mining distributed datasets. Schuster et al presented a distributed association rule mining algorithm called D-ARM to perform a single scan over the database [Schuster, 2003]. The scheme proposed in [Ghoting, 2004] gives controlled interactive response times when processing distributed data streams. Wolff and Schuster proposed an algorithm to mine association rules in large-scale distributed peer-to-peer systems [Wolff, 2004], by which every node in the system can reach the exact solution.

3.5. Visualization

In some data stream applications, especially monitoring applications, there is a demand for visualization of association rules to facilitate the analysis process. An interactive use of visualized graphs can help the users understand the relationship between related association rules better so that they can further select and explore a specific set of rules from the visualization.

In [Hofmann, 2000], the authors showed how Mosaic plots can be used to visualize association rules. In [Bruzzese, 2004], Bruzzese and Buono proposed a visual strategy to both overview the association rule structure and further investigate inside a specific set of rules selected by the user. In [Cai, 2004], the authors developed a set of visualization tools which can be served for continuous queries and mining displays; they trigger alarms and give messages when some alarming incidents are being detected based on the ongoing stream data.

4. CONCLUSIONS

In this paper we discussed the issues that need to be considered when designing a stream data association rule mining technique. We reviewed how these issues are handled in the existing literature. We also discussed issues that are application-dependent.

From the above discussions, we can see that most of the current mining approaches adopt an incremental and one pass mining algorithm which is suitable to mine data streams, but few of them address the concept drifting problem. Most of these algorithms produce approximate results [Li, 2004, Yu, 2004, Chang, 2004, Manku, 2002,

Charikar, 2004, Giannella, 2003, Lin, 2005]. This is because due to the huge amount of data streams and limited memory, there is not enough space to keep frequency counts of all itemsets in the whole data streams as we do in traditional databases. A few of the proposed algorithms generate exact mining results by maintaining a small subset of frequent itemsets from data streams and keeping their exact frequency counts [Yang, 2004, Chi, 2004, Mao, 2005]. To keep track of the exact frequency counts of target itemsets with limited memory space, one way is to adopt the sliding window data processing model, which maintains only part of the frequent itemsets in sliding window(s) as in [Chi, 2004]. Another way is to maintain only special itemsets such as short frequent itemsets, closed frequent itemsets or maximal frequent itemsets as in [Yang, 2004, Mao, 2005].

The current stream data mining methods require users to define one or more parameters before their execution; however, most of them do not mention how users can adjust these parameters online while they are running. It is not desirable/feasible for users to wait until a mining algorithm to stop before they can reset the parameters. This is because it may take a long time for the algorithm to finish due to the continuous arrival and huge amount of data streams. Some proposed methods let users adjust only certain parameters online, but these parameters may not be the key ones to the mining algorithms, and thus are not enough for a user friendly mining environment. For example, in [Ghoting, 2004], the authors proposed a method to mine distributed data streams which allows the users, to modify online only one of the mining parameters, the response time, to trade off between the query response time and accuracy of the mining results. For further improvement, we may consider to either let users adjust online or let the mining algorithm auto-adjust most of the key parameters in association rule mining, such as support, confidence and error rate.

Research in data stream association rule mining is still in its early stage. To fully address the issues discussed in this paper would accelerate the process of developing association rule mining applications in data stream systems. As more of these problems are solved and more efficient and user-friendly mining techniques are developed for the end users, it is quite likely that in the near future data stream association rule mining will play a key role in the business world.

Acknowledgement

This material is based upon work supported by (while serving at) the National Science Foundation (NSF), the NASA Grant No. NNG05GA30G issued through the Office of Space Science and the OSU Grant. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

5. REFERENCES

[Agrawal, 1993] Rakesh Agrawal, Tomasz Imielinski, Arun Swami; Mining Association Rules between Sets of Items in Massive Databases; Int'l Conf. on Management of Data; May 1993.

[Agrawal, 1994] Rakesh Agrawal, Ramakrishnan Srikant; Fast Algorithms for Mining Association Rules; Int'l Conf. on Very Large Databases; September 1994.

[Bruzzese, 2004] Dario Bruzzese, Paolo Buono; Combining Visual Techniques for Association Rules Exploration; The Working Conf. on Advanced Visual Interfaces; May 2004.

[Cai, 2004] Y. Dora Cai, Greg Pape, Jiawei Han, Michael Welge, Loretta Auvil; MAIDS: Mining Alarming Incidents from Data Streams; Int'l Conf. on Management of Data; June 2004.

[Chang, 2003] Joong Hyuk Chang, Won Suk Lee, Aoying Zhou; Finding Recent Frequent Itemsets Adaptively over Online Data Streams; ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining; August 2003.

[Chang, 2004] Joong Hyuk Chang, Won Suk Lee; A Sliding Window Method for Finding Recently Frequent Itemsets over Online Data Streams; Journal of Information Science and Engineering; July 2004.

[Charikar, 2004] Moses Charikar, Kevin Chen, Martin Farach-Colton; Finding Frequent Items in Data Streams; Theoretical Computer Science; January 2004.

[Cheung, 1996] David W. Cheung, Jiawei Han, Vincent T. Ng, C.Y. Wong; Maintenance of Discovered Association Rules in Large Databases: An Incremental Updating Technique; IEEE Int'l Conf. on Data Mining; November 1996.

[Cheung, 1997] David W. Cheung, S.D. Lee, Benjamin Kao; A General Incremental Technique for Maintaining Discovered Association Rules; Int'l Conf. on Database Systems for Advanced Applications; 1997.

[Chi, 2004] Yun Chi, Haixun Wang , Philip S. Yu , Richard R.; Moment: Maintaining Closed Frequent Itemsets over a Stream Sliding Window; IEEE Int'l Conf. on Data Mining; November 2004.

[Cormode, 2003] Graham Cormode, S.Muthukrishnan; What's Hot and What's Not: Tracking Most Frequent Items Dynamically; ACM Transactions on Database Systems; March 2005.

[Demaine, 2002] Erik D. Demaine, Alejandro Lpez-Ortiz, J. Ian Munro; Frequency Estimation of Internet Packet Streams with Limited Space; European Symposium on Algorithms; September 2002.

[Gaber, 2003] Mohamed Medhat Gaber, Shonali Krishnaswamy, Arkady Zaslavsky; Adaptive Mining Techniques for Data Streams Using Algorithm Output Granularity; The Australasian Data Mining Workshop; December 2003.

[Gaber, 2004] Mohamed Medhat Gaber, Arkady Zaslavsky and Shonali Krishnaswamy; Resource-Aware Knowledge Discovery in Data Streams; Int'l Workshop on Knowledge Discovery in Data Streams; September 2004.

[Gaber, 2005] Mohamed Medhat Gaber, Arkady Zaslavsky, Shonali Krishnaswamy; Mining Data Streams: A Review; ACM SIGMOD Record Vol. 34, No. 2; June 2005.

[Ghoting, 2004] Amol Ghoting, Srinivasan Parthasarathy; Facilitating Interactive Distributed Data Stream Processing and Mining; IEEE Int'l Symposium on Parallel and Distributed Processing Systems; April 2004.

[Giannella, 2003] Chris Giannella, Jiawei Han, Jian Pei, Xifeng Yan, Philip S. Yu; Mining Frequent Patterns in Data Streams at Multiple Time Granularities; Data Mining: Next Generation Challenges and Future Directions, AAAI/MIT; 2003.

[Guha, 2001] Sudipto Guha, Nick Koudas, Kyuseok Shim; Data Streams and Histograms; ACM Symposium on Theory of Computing; 2001.

[Han, 1999] Jiawei Han, Guozhu Dong, Yiwen Yin; Efficient mining of partial periodic patterns in time series database; IEEE Int'l Conf. on Data Mining; March 1999.

[Han, 2000] Jiawei Han, Jian Pei, Yiwen Yin; Mining Frequent Patterns without Candidate Generation; Int'l Conf. on Management of Data; May 2000.

[Halatchev, 2005] Mihail Halatchev and Le Gruenwald; Estimating Missing Values in Related Sensor Data Streams; Int'l Conf. on Management of Data; January 2005.

[Hofmann, 2000] Heike Hofmann, Arno P. J. M. Siebes, Adalbert F. X. Wilhelm; Visualizing Association Rules with Interactive Mosaic Plots; ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining; August 2000.

[Huang, 2002] Hao Huang, Xindong Wu, Richard Relue; Association Analysis with One Scan of Databases; IEEE Int'l Conf. on Data Mining; December 2002.

[Jin, 2003] Cheqing Jin, Weining Qian, Chaofeng Sha, Jeffrey X. Yu, Aoying Zhou; Dynamically Maintaining Frequent Items over a Data Stream; Int'l Conf. on Information and Knowledge Management; 2003.

[Kargupta, 2004] Hillol Kargupta, Ruchita Bhargava, Kun Liu, Michael Powers, Patrick Blair, Samuel Bushra, James Dull, Kakali Sarkar, Martin Klein, Mitesh Vasa, David Handy; VEDAS: A Mobile and Distributed Data Stream Mining System for Real-Time Vehicle Monitoring; SIAM Int'l Conf. on Data Mining; 2004.

[Karp, 2003] Richard M. Karp, Scott Shenker; A Simple Algorithm for Finding Frequent Elements in Streams and Bags; ACM Transactions on Database Systems; March 2003.

[Lee, 1997] S.D. Lee, David W. Cheung; Maintenance of Discovered Association Rules: When to update?; Research Issues on Data Mining and Knowledge Discovery; 1997.

[Li, 2004] Hua-Fu Li, Suh-Yin Lee, and Man-Kwan Shan; An Efficient Algorithm for Mining Frequent Itemsets over the Entire History of Data Streams; Int'l Workshop on Knowledge Discovery in Data Streams; Sept. 2004.

[Lin, 2005] Chih-Hsiang Lin, Ding-Ying Chiu, Yi-Hung Wu, Arbee L. P. Chen; Mining Frequent Itemsets from Data Streams with a Time-Sensitive Sliding Window; SIAM Int'l Conf. on Data Mining; April 2005.

[Manku, 2002] Gurmeet Singh Manku, Rajeev Motwani; Approximate Frequency Counts over Data Streams; Int'l Conf. on Very Large Databases; 2002.

[Mao, 2005] Guojun Mao, Xindong Wu, Chunnian Liu, Xingquan Zhu, Gong Chen, Yue Sun, Xu Liu; Online Mining of Maximal Frequent Itemsequences from Data Streams; University of Vermont, Computer Science Technical Report CS-05-07; June 2005.

[Otey, 2003] Matthew Eric Otey, Chao Wang, Srinivasan Parthasarathy, Adriano Veloso, Wagner Meira Jr.; Mining Frequent Itemsets in Distributed and Dynamic Databases; IEEE Int'l Conf. on Data Mining; 2003.

[Otey, 2004] Matthew Eric Otey, Srinivasan Parthasarathy, Chao Wang, Adriano Veloso, Wagner Meira Jr.; Parallel and Distributed Methods for Incremental Frequent Itemset Mining; IEEE Transactions on Systems, Man and Cybernetics; December 2004.

[Parthasarathy, 1999] S. Parthasarathy, M. J. Zaki, M. Ogihara, S. Dwarkadas; Incremental and interactive sequence mining; Int'l Conf. on Information and Knowledge Management; 1999.

[Pinto, 2001] Helen Pinto, Jiawei Han, Jian Pei, Ke Wang, Qiming Chen, Umeshwar Dayal; Multi-Dimensional Sequential Pattern Mining; Int'l Conf. on Information and Knowledge Management; 2001.

[Relue, 2001] Richard Relue, Xindong Wu, Hao Huang; Efficient runtime generation of association rules; Int'l Conf. on Information and Knowledge Management; October 2001.

[Schuster, 2003] Assaf Schuster, Ran Wolff, and Dan Trock; Distributed Algorithm for Mining Association Rules; IEEE Int'l Conf. on Data Mining; November 2003.

[Teng, 2004] Wei-Guang Teng, Ming-Syan Chen, and Philip S. Yu; Resource-Aware Mining with Variable Granularities in Data Streams; SIAM Int'l Conf. on Data Mining; 2004.

[Veloso, 2003] Adriano Veloso, Wagner Meira Jr., Marcio Carvalho, Srini Parthasarathy, Mohammed J. Zaki; Parallel, Incremental and Interactive Mining for Frequent Itemsets in Evolving Databases; Int'l Workshop on High Performance Data Mining: Pervasive and Data Stream Mining; May 2003.

[Veloso, 2003b] Adriano Veloso, Matthew Eric Otey, Srinivasan Parthasarathy, Wagner Meira Jr.; Parallel and Distributed Frequent Itemset Mining on Dynamic Datasets; Int'l Conf. on High Performance Computing; 2003.

[Wang, 2003] Haixun Wang, Wei Fan, Philip S. Yu, Jiawei Han; Mining Concept-Drifting Data Streams using Ensemble Classifiers; ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining; August 2003.

[Wolff, 2004] Ran Wolff, Assaf Schuster; Association Rule Mining in Peer-to-Peer Systems; IEEE Transactions on Systems, Man and Cybernetics, Part B, Vol. 34, Issue 6; December 2004.

[Yang, 2004] Li Yang, Mustafa Sanver; Mining Short Association Rules with One Database Scan; Int'l Conf. on Information and Knowledge Engineering; June 2004.

[Yu, 2004] Jeffrey Xu Yu, Zhihong Chong, Hongjun Lu, Aoying Zhou; False Positive or False Negative: Mining Frequent Itemsets from High Speed Transactional Data Streams; Int'l Conf. on Very Large Databases; 2004.

[Yu, 2005] Chung-Ching Yu, Yen-Liang Chen; Mining Sequential Patterns from Multidimensional Sequence Data; IEEE Transactions on Knowledge and Data Engineering; January 2005.

[Zheng, 2003] Qingguo Zheng, Ke Xu, Shilong Ma; When to Update the Sequential Patterns of Stream Data; Pacific-Asia Conf. on Knowledge Discovery and Data Mining; 2003.

[Zhu, 2002] Yunyue Zhu, Dennis Shasha; StatStream: Statistical Monitoring of Thousands of Data Streams in Real Time; Int'l Conf. on Very Large Data Bases; 2002.