Method presented for finding Frequent Itemsets in web data streams

Farzaneh Kaviani¹*, Mohammad Reza Khayyambashi²

(1) Department of computer, Isfahan (Khorasgan) Branch, Islamic Azad University, Isfahan, Iran
(2) Faculty of Computer Engineering, University of Isfahan, Isfahan, Iran

Abstract

Continual data checking is considered as one of the most common search tools for frequent itemsets which requires storage on memory. On the other hand, according to properties of data stream which are unlimited productions with a high-speed, it is not possible saving these data on memory and we need for techniques which enables online processing and finding repetitive standards. One of the most popular techniques in this case is using sliding windows. The benefits of these windows can be reducing memory usage and also search acceleration. In this article, a new vertical display and an algorithm is provided based on the pins in order to find frequent itemsets in data streams. Since this new display has a compressed format itself so, the proposed algorithm in terms of memory consumption and processing is more efficient than any other similar algorithms.

Keywords: Data stream, Sliding windows, pin, collection of frequent data items.

1 Introduction

One of the applications of Data Mining may be finding important and frequent itemsets in large amounts of various types of data. One kind of these data is Data Streams. It means an unlimited sequence of data which is being produced consistently and with a very high speed. If repetition of an itemset is more than the certain amount that is diagnosed as a Frequent Itemset. Frequent Itemset exploration was first presented in 1993 by Mr.Agrawal and his colleagues [1]. Due to the extremely wide application of this field in business, industry and different sciences, base Apriori algorithm was presented in 1994. This algorithm searches for data several times in order to find frequent itemsets. Unfortunately it is not possible to do this on data streams because these streams are being produced indefinitely and with a very high speed therefore, it is not possible to save them for repeating exploration and consequently we are also not able to use these methods for finding frequent itemsets in data streams.
It is essential to draw plans which find frequent itemsets by scanning data once.

* Corresponding Author. Email address: kaviyanfarzaneh@gmail.com
In this context, the problem has been fixed by introduction of new display for storage of a summary from received data. The presented algorithm maintains part of the last data received and saves them as Bit Based Representation to search for frequent itemsets. In the new display the effects of transactions which font data in them does not appear, will be removed from the summary data stored and as a result the less memory is used. On the other hand for finding a repeated data font collection fewer operations is done and thus increased computing speed is expected. In recent years, according to data stream applications some methods are introduced in ways that make these itemsets extracted with a considerable speed and also accuracy. Presented methods must scan each data element only once, and the memory usage must be low. It also must process produced new elements very fast otherwise as mentioned above because of high speed in producing data, the process of input data is not possible anymore and input data will be lost.

2 An overview of the data mining algorithms

2.1. MFi-TransSW method
This algorithm is presented based on Apriori algorithm in 2009. The algorithm searches full set of duplicate items in the current window after the user request. This algorithm used bit based representation for data items so that a sequence of bits is considered for each data item. Presence of one data item in one transaction is presented with 1 and absence of them is presented by 0. Shifting to the left bit is used to add modern transaction and remove old transaction from window, this method has been composed of three phases: The first phase relates to initializing windows and obtaining bit sequence of each data item. Length of each bit sequence is equal to length of the window. In the next phase that is the same window sliding, old transactions are removed and modern transactions are added and to observe this sliding in the bit sequence of data item, bit sequence of data item are shifted left and in the third phase, frequent itemsets are explored. If this algorithm tries to increase speed of exploration by bit operation, but this technique does not have time efficiency since the same Apriori constrains that is production and test of data itemset are available. It means there are that high volume of candidate itemsets in cases that number of items are a lot and set of frequent items with high length are available in this algorithm .on the other hand, since this algorithm saves information for presence of repetition and also absence of repetition , high volume of memory is required. It means that a constant string should be considered proportional to length of the window.

2.2. Eclat algorithm
This algorithm is the depth first method for finding frequent items and operates based on vertical representation of databases. Vertical representation of databases means that list of transactions including that item is maintained for each item inside databases that backup of one data item is equal to length of this list and then by sharing on these lists backup of each data itemset is found and one tree is built supposedly by one depth firs survey that frequent data itemset is found by using it. Since Eclet method reads all data one time and then explore operation is conducted on them, this algorithm can be used for exploring data stream but disadvantage of this method is that it is not suitable for distributed data item. For eliminate this problem, another method was identified; instead of maintaining ordinary list for each data, itemset uses complementary list but these lists also have problem. The problem is that they are not suitable for dense data set. For this reason, modern method named LDS was identified by Mr. Diper that removes this problem. In the LDS method similar to Eclet method for each data item one list of transaction No. is maintained. Now if length of this list of transaction No. is more than half of window length, complementary list is maintained. Complementary means List of transaction No. inside the window that there is no ordinary list related to this data item. In result the shortest list for each data item is maintained and finally explore operation is done when user requests.
2.3. MFPN algorithm

This algorithm was identified by the Mr. Zhin et al. This algorithm is one of the developed algorithms of Eclet. This method is in category of pin algorithm that is updating operation is conducted based on one pin. This algorithm is used to explore network data.

In this method, instead of numbering transaction inside the window, they are numbered inside each pin and operation of building transaction list is done for each item inside this pin. Of course, by numbers that are presented inside each pin not those inside each window. Then explore operation is performed at the same time when the modern pin is prepared and frequent data itemset are found. So results are ready to be delivered whenever the user requests them.

3 Proposed algorithm

Proposed algorithm used siding window for exploring data stream. Since in most applications of data stream, the recent data are more important than old data so a part of recent data are considered as one window and exploring operation are conducted on them. This window has a constant size equal to constant number of transactions and this window is updated when a new transaction is appeared. But since speed of producing transactions in the data stream is very high, window should be updated with very high speed and in many times that this issue has great deal of overhead and decreases efficiency of system. To solve this problem, each window is divided into some subwindow with the same length called pin and whenever the new transaction appeared with the size of this pin, updating operation of the window is performed. Updating trend is conducted by eliminating the oldest pin from the window and adding the new pin to the window. The proposed algorithm has been composed of three phases.

3.1. The first phase: initial volume for the window

This phase includes the primary to SW and finding frequent data item. The process advances this way: input stream is monitored and the data item is found when each transaction is crucial and the dynamic vector list of data item is created and this pin is added to the window simultaneously whenever a transaction appears with a size of the pin.

3.2. Second phase – window sliding

This phase forms by popping up of a new window, that is by creation of a transactions which is the same size of a pin, the window must be updated. That is a new pin must be added to the window and the oldest (expired one) must be deleted from the window, and 1) data items of the new pin should be added to the previous pin. Overally the results obtained from addition and deletion of these pins must be applied on the new data item 1.

3.3. Third phase: creation of a set of frequent data items

The set of the recurrent are stored in a tree and the first set of recurrent data items are created by the survey of the first depth at this tree this is the same method as represented in Eclat

4 Evaluation and results

DATA SETS T40I10D100K which have been artificially created using IBM data generator software is used for testing.
Given that the proposed method is a predictive, the efficacy level of this method has been compared with MFT-TransSW (H. Li, S. Lee, and M. Shan and LDS (S.K. Tanbeer, C.F. Ahmed, 2009 [8]), MFPN (H. Liu, S. Lin, J. Qiao, 2008 [9], Shan, 2004 [4])

In the real world, a data stream is an unlimited extension that may continue forever, but in testing data streams are simulated by reading from the aforesaid datasets and the window sliding process continues until data streams are finished. Anyway, due to high speed of data entry in data streams any comparison among different methods of finding recurrent itemsets must be done in consideration of 1, amount of memory used, and 2, execution time. That is why in tests the amount of used memory and the time of execution have been investigated.

In the first test, the amount of memory consumed has been investigated for different window sizes.

For MFI-TransSW the amount of the memory of a window has been obtained by adding up the lengths of the bit strings. It is because the length of a bit string for a data item in a window exactly equals as the number of transactions in a window. Results are shown in figure 1. As it is can be seen in the figure, the amount of consumed memory increases with the increasing of the window size. This is natural because a greater volume of data must be surveyed, but in normal conditions it is seen that the amount of consumed memories is less in the new method presented here. That's because this method uses a different encoding such that this encoding redundant data which were not used are ignored on one hand, and on the other this follows a single-bit method, which in turn consumes less memory

The DPB-BA method follows a dynamic bit pin list for each data item and since this method uses a dynamic pin list which is also efficient in terms of memory, the represented method performs better in terms of memory consumption.

The proposed method is swifter than other methods. The bigger the size of memory, the higher the efficiency will be. The reason behind the higher efficiency of the proposed method is the representation of the efficiency used in displaying items. Because the larger the window, the more frequent itemsets will be recognized and stored. However, since an efficient displaying method that uses the minimum amount of memory has been used for storing proposed method for storing each frequent itemset less computation is needed for finding out the frequent itemsets in this method. In other methods the amount of calculations is more and requires more execution time.

Figure 1: Comparison of consumed memory on IBM data
The more the backup rate the less will be the data found out to be recurrent. Consequently, as it is shown in figure 3, the algorithm for building the tree in question in the third phase will require less data and the time consumed by the algorithm will be decreased.

The time of the DBP-BA execution is less than the other two algorithms. However, as it can be seen, generally, the minimum support of execution time in all algorithms will decrease.

**Figure 3**: The effect of minimum backup rate on algorithm execution time using IBM data

**5 Conclusion**

Data stream is a new data model which have received much attention recently. Since you are facing a great volume of data, you cannot store them in the memory. So you need methods to immediately process them and reveal the knowledge underlying them, a very important challenge in the treatment of these data. A topic which has received much attention is how to find frequent items in data streams. This issue has been subject to a great deal of discussion in static databases, but it is still a problem which is yet to be discussed in data streams.

Today, data production speed is much higher than before due to developments. Therefore, previous algorithms used in data mining are no more usable because of the great amount of memory consumption and consequently they cannot process the input data, and a lot of entered data are lost in this way. It has been
seen that algorithms provided are not of adequate capabilities and produce rough results that are not of adequate accuracy. That is why new algorithms should be provided that are precise enough to meet this purpose.

This study examines the algorithms used for mining frequent itemsets in the data streams and they have been classified in various terms. One of these classifications was the time needed for data mining such that it could be done any moment or delayed as desired by the user. Clearly the first method is swifter. Other classifications were based on the windows used in them. They have also been classified into two groups. This paper provides a new method for mining frequent itemsets in data streams. This method uses sliding windows method and reduces the amount of memory consumed by the algorithm by providing anew representation of memory. This algorithm is among the algorithms that does data mining at the request of the use. It can be seen that the method thus introduced is superior to other methods in comparison to other methods.

References


http://dx.doi.org/10.5120/316-484


http://dx.doi.org/10.1016/j.is.2004.02.001


http://dx.doi.org/10.1177/0165551505050785


http://dx.doi.org/10.1016/j.eswa.2007.11.061


http://dx.doi.org/10.1016/j.ins.2009.07.012


http://dx.doi.org/10.1109/icicta.2008.227
http://dx.doi.org/10.1145/2076623.2076659

http://dx.doi.org/10.1016/j.eswa.2010.06.012

http://dx.doi.org/10.1016/j.datak.2007.12.006


http://dx.doi.org/10.1145/1656274.1656287

http://dx.doi.org/10.1016/j.protcy.2013.12.424

http://dx.doi.org/10.1016/j.ipl.2014.09.009