A Quick Algorithm for Incremental Mining Closed Frequent Itemsets over Data Streams

Shankar B. Naik
Dept. of Computer Science
S.S.A. Govt. College, Pernem, Goa
+919420897135
xekhar@rediffmail.com

Jyoti D. Pawar
Dept. of Computer Science and Technology
Goa University
+919422059112
jyotidpawar@gmail.com

ABSTRACT

In this paper we have proposed an efficient algorithm QMINE to find closed frequent itemsets over a data stream. Our approach performs a few operations. Experiments have shown that our approach outperforms the previous approaches, significantly.

Categories and Subject Descriptors
E.m [**Data**]: Miscellaneous- *Patterns in Data Stream*

General Terms
Algorithms

Keywords
Data streams, Data mining, Sliding window, Closed frequent itemsets

1. INTRODUCTION

Frequent itemset mining aims at generation of frequent itemsets from transactional data streams [8].

Mining frequent itemsets from a data stream is a challenging task because of its unknown and huge size, and errors in the results[2]. The three approaches used to mine frequent itemsets in data streams are landmark windows [6], damped windows [3], and sliding windows [4].

A closed itemset is an itemset which has no proper superset with similar support [7][8]. A closed itemset is frequent if its support is not less than the minimum support threshold.

The purpose of this work is to mine closed frequent itemsets from transactional data streams using a sliding window model. An efficient algorithm QMINE is proposed to mine frequent closed itemsets from a transactional data stream.

The remainder of the paper is organized as follows-Section 2 is on the related work. The problem is defined in section 3. The proposed algorithm is presented in Section 4. Section 5 describes the experimental results and Section 6 concludes the paper.

CODS '15, Mar 18-21, 2015, Bangalore, India
ACM 978-1-4503-3436-5/15/03.
http://dx.doi.org/10.1145/2732587.2732611

2. RELATED WORK

Two main algorithms to find closed frequent itemsets from transactional data streams are Moment[5], NewMoment [7]. In this paper we have proposed an algorithm QMINE which allows the user to generate frequent itemsets by providing the value of minimum support online. The number of itemsets is less as they are closed itemsets, The proposed algorithm reads a transaction only once unlike some of the previous approaches. It does not have to access the sliding window while a new transaction is added or deleted. Instead, it maintains closed itemsets only by referring to contents of the summary data structure. This saves computational time considerably.

3. PROBLEM DEFINITION

3.1 Problem Statement

Given a sliding window *SW* of size *w* and a minimum support threshold *s*, the problem is to find the set of closed frequent itemsets in *SW* which contains the latest *w* transaction of the data stream *D*. The value of *s* is not fixed and can be specified by the user online.

3.2 Proposed data structure INL (INdexed List)

The summary data structure *INL* consists of (1) *ISets*; and (2) *ISIndex*. The details of the structures are described as follows (Figure 1). *ISets* contains the set of closed frequent itemsets. *ISets* is a table with three fields: (1) *ItemId*; (2) Itemset; and (3) Support. *ItemId* is the key or the identifier of the Itemset in *ISets* table. Support is the support of Itemset in the current sliding window.

ISIndex

Item	Vector
a	
b	
c	
d	

ISets

ItemId	Itemset	Support

Figure 1 Summary data structure *INL*

ISIndex is a table with two fields: *Item*, and *Vector*. The Vector is a bit-sequence in which the i^{th} bit is set to one if the Item belongs to the Itemset in *ISets* table and *i* is the *ItemId* of Itemset in *ISets*. *ISIndex* is very efficient in searching for itemsets in

ISets table. This is the index table which improves the efficiency in searching the itemsets in *ISets* table.

4. ALGORITHM QMINE (QUICK MINING OF CLOSED FREQUENT ITEMSET)

In this section we describe the working of QMINE algorithm. The algorithm consists of two steps: (1) Add; and (2) Delete. The Add step is performed when a transaction arrives at the sliding window. The Delete step is performed to update the closed frequent itemsets when a transaction leaves the sliding window.

4.1 Add Step

This step is performed when a new transaction arrives at the sliding window. When a transaction containing the itemset *X* arrives at the sliding window, the supports of subsets of *X* will change. QMINE searches for these subsets of X in *ISets* and updates their supports.

4.2 Delete Step

When a transaction containing the itemset *X* is removed from the sliding window the supports of only the subsets of *X* should be decreased by one. Some of the subsets of *X* which are in *ISets* table may not remain closed itemsets after the transaction is deleted. The Delete step first finds all subsets of *X*. After having found these subsets it then checks for those subsets which have become non-closed and eliminates them from the *ISets*. When an itemset X_r with *ItemId i* is removed from *ISIndex*, the i^{th} bit in the Vector of the items belonging to the X_r in the *ISIndex* table are changed from 1 to *0*.

In order to generate the subsets of *X* the Delete step performs intersection of *X* with the Itemsets in *ISets* containing atleast one item in *X*. The Delete step uses Vectors in *ISIndex* table locate these Itemsets in *ISets* table. In order to locate their *ItemId*s, a bitwise OR is performed between the Vectors of the items belonging to *X*. The positions of bits with value 1 represent the *ItemId*s of the subsets of *X* present in *ISets* table. This reduces the search time considerably.

5. EXPERIMENTAL RESULTS

Experiments were performed to compare the performance of QMINE with NewMoment algorithm. All experiments are done on 2.26GHz Intel® Core™ i3 PC with 3 GB memory and running on Windows 7 system. The proposed algorithm is implemented in C++ and compiled using GNU GCC compiler. The synthetic dataset is generated using IBM Synthetic Data Generator [1]. The experiments were performed by changing the sliding window size w from 10K to 100K. The value of minimum support threshold is set to 0.2. QMINE requires less time for a transition of a sliding window as compared to NewMoment. These observations are done by taking average of 50 transitions. QMINE requires less time than NewMoment for lower values of minimum support. For higher values of minimum support NewMoment performs better in term of time requirements. This happens because QMINE is independent of minimum support and generates all closed itemsets, whereas, NewMoment generates closed itemsets which are frequent. For higher values of minimum support the number of frequent closed itemsets generated is less, hence NewMoment is more time efficient than QMINE for higher values of minimum support. QMINE can be made time efficient by restricting it to generate closed itemsets which are frequent.

As QMINE maintains all closed frequent itemsets in its summary data structure, QMINE requires more memory than NewMoment for higher values of minimum support.

6. CONCLUSION

A quick algorithm to mine closed frequent itemsets over transactional data streams is proposed. It uses a summary data structure which is efficient in searching itemsets stored in it. The proposed algorithm QMINE generates all close itemsets and allows the user to specify the minimum support threshold online. The experiments show that QMINE outperforms NewMoment algorithm in time for lower values of minimum support. For higher values of minimum support, QMINE can be made time efficient by having it to generate only the frequent itemsets, which motivates us for our future work.

REFERENCES

[1] Agrawal, R. and Shrikant, R.1994. Fast algorithms for mining association rules.*Proceedings of the 20th international conference on very large databases,* 487-499, (1994).

[2] Babock, B., Babu, S., Motwani, R. and Widom. J. 2002 Models and issues in data stream systems. *Proceedings of the 21st ACM SIGMOD-SIGACT-AIGART symposium on principles of database systems,* 1-16, (2002).

[3] Chang, J., and Lee, W.(2004) Decaying obsolete information in finding recent frequent itemsets over data stream. *IEICE Transaction on Informations and Systems,* 6, (2004).

[4] Chang, J., and Lee, W. 2004. A sliding window method for finding recently frequent itemsets over online data streams. *Journal of information science and engineering*, 4, (2004).

[5] Chi Y., Wang, H., Yu, P., and Muntz. R..2004. MOMENT: Maintaining closed frequent itemsets over a stream sliding window, *Proceedings of the 4th IEEE international conference on data mining*, 59-66, (2004).

[6] Jin, R. and Agrawal, G.2005. An algorithm for in-core frequent itemset mining on streaming data. *Proceedings of the 5th IEEE international conference on data mining, (*2005).

[7] Lee, H.F. and Ho, C.C,.and.Lee, S.Y. 2009 Incremental updates of closed frequent itemsets over continuous data streams, *Expert system with application,* 2-36, (2009).

[8] Naik, S.B., and Pawar, J.D.2013 An Efficient Incremental Algorithm to mine frequent itemsets in data streams. *Proceeding of the 19th International Conference on Management of Data (COMAD),* 117-120, (2013).