International Journal of Computer Applications (0975 – 888)
Volume 48– No.20, June 2012

A Proficient Approach of Incremental Algorithm for Frequent Pattern Mining

Endu Duneja
R.I.T.S, Bhopal
M.P., India

A.K. Sachan
PhD ,R.I.T.S., Bhopal
M.P., India

ABSTRACT

The conundrum of mining association rules has drawn a lot of attention in the research community. In spite of their practical benefits, it is nontrivial to perform incremental mining or efficient mining of constrained association rules. Many researchers have recently focused on providing discrete solutions for these two problems. It is belief that constrained mining will be in tradition, incremental mining of constrained rules will be obligatory. In this paper, a novel algorithm for incremental mining is proposed which satiates the gap between incremental & constrained mining researchers. The proposed algorithm can discover sequential frequent pattern itemsets in incremental database. We developed new method that considers sequential data mining of marketing websites as an effective tool that participates in having well-structured websites. The advantage of this method is that is saves a lot maintenance efforts.

General Terms

Association Rule Mining, Frequent itemset mining, Incremental mining, constrained mining.

Keywords

Frequent Itemset, Association Rule, Incremental mining.

1. INTRODUCTION

Ensuing the instable augmentation of the amount of data generated by transactional systems, defy for finding new techniques to excerpt useful patterns from such a huge amount of data arose. Data mining come forward as a new research area to meet this challenge. Recently, data mining fascinated a lot of research consideration.

In the data mining field two significant issues are uncovered, namely incremental mining & interactive mining. The former issue, incremental mining, exhibits in normal working environments. Data starts to accumulate in smaller increments, after preliminary investigation of the stored data. It would be natural to try to adapt mining tasks to handle those increments in a proficient way using earlier revealed patterns rather than mining from scratch [2, 3, 4, 5, 6, 7, 8, 10].

The later issue, interactive mining is owing to the large extent of time it takes virtually all mining tasks to accomplish. Moreover, if those tasks are left untraced they can yield a large number of patterns making the users bewildered trying to excerpt what is really remarkable for them. Therefore, interactive mining through user-defined constrained queries in which the user mentions his remark (constrained mining) is crucial to avert data mining from becoming itself a knowledge problem. [11, 12, 13, 14, 15, 16, 17, 18, 19]

In spite of their obvious practical benefits, it is not possible to perform incremental mining of association rules. Innumerable research efforts tried to present discrete solutions for these problems. Besides, the future is for constrained association rule mining & many see that it should become the benchmark [20, 9, 21, 15]. The challenge lies in how to competently maintain discovered rules that satisfy certain constrictions incrementally without having to accomplish the whole task from task scratch.

2. ASSOCIATION RULE MINING AND FREQUENT ITEMSET MINING

In this section we will introduce association rule mining problem in detail. Different concerns in Association Rule Mining (ARM) will be elucidated together.

Association Rule Problem Firstly it was stated in [Agrawal et al. 1993] by Agrawal that the conventional statement of association rule mining problem was discovering the interesting association or correlation relationships among a large set of data items.

A fundamental problem for mining association rules is mining frequent itemsets. In a market basket transaction dataset, frequent itemset mining is the process of searching for itemsets that a set of customers likely to purchase in a given visit to the store. The study of the behavior of frequent itemsets with respect to time is done through mining for frequent itemsets in different time periods. Top level goal of temporal analysis is to filter for interesting frequent itemsets.

Let I be a set of items. A set $X = \{i_1, \ldots, i_K\} \subseteq I$ is called an itemset, or a k-itemset if it contains k items. A transaction over I is a couple $T = (tid, I)$ where tid is the transaction identifier and I is an itemset. A transaction $T = (tid, I)$ is said to support an itemset $X \subseteq I$, if $X \subseteq I$. A transaction database D over I is a set of transactions over I.

The cover of an itemset X in D consists of the set of transaction identifiers of transactions in D that support X:

$$cover(X, D) = \{tid \mid (tid, I) \in D, X \subseteq I\}.$$

The support of an itemset X in D is the number of transactions in the cover of X in D:

$$support(X,D) = |cover(X,D)|$$

The frequency of an itemset X in D is the probability of X occurring in a transaction T∈D:

$$frequency(X, D) = P(X) = support(X, D) / |D|$$
$$|D| = support(\{ \}, D).$$

An itemset is called frequent if its support is no less than a given absolute minimal support threshold σ_{abs}, with $0 \leq \sigma_{abs} \leq |D|$. When working with frequencies of itemsets instead of their supports, we use a relative minimal frequency threshold σ_{rel} , with $0 \leq \sigma_{rel} l \leq 1$. Obviously, $\sigma_{abs} = \lceil \sigma_{rel} \cdot |D| \rceil$.

Definition1. Let ***D*** be a transaction database over a set of items ***I***, and σ a minimal support threshold. The collection of frequent itemsets in ***D*** with respect to σ is denoted by

$F(D, \sigma) = \{X \subseteq I \mid support(X,D) \geq \sigma\}$, or simply ***F*** if ***D*** and σ are clear from the context.

The main question arise here in itemset mining is that, given a set of items ***I***, a transaction database ***D*** over ***I***, and minimal support thresholdσ, find $F (D,\sigma)$.

In practice we are not only interested in the set of itemsets F, but also in the actual supports of these itemsets. An association rule is an expression of the form $X \Rightarrow Y$, where ***X*** and ***Y*** are itemsets, and $X \cap Y = \{ \}$. Such a rule expresses the association that if a transaction contains all items in **X**, then that transaction also contains all items in **Y**. **X** is called the body or antecedent, and **Y** is called the head or consequent of the rule.

The support of an association rule $X \Rightarrow Y$ in ***D*** is the support of $X \cup Y$ in D, and similarly, the frequency of the rule is the frequency of $X \cup Y$. An association rule is called frequent if its support (frequency) exceeds a given minimal support (frequency) threshold σ_{abs} (σ_{rel}). Again, we will only work with the absolute minimal support threshold for association rules and omit the subscript abs unless explicitly stated otherwise.

The confidence or accuracy of an association rule $X \Rightarrow Y$ in ***D*** is the conditional probability of having ***Y*** contained in a transaction, given that ***X*** is contained in that transaction:

$$Confidence\ (X \Rightarrow Y, D) = P(Y\ |X) = support(X \cup Y, D) / support(X, D)$$

The rule is called confident if $P\ (Y \mid X)$ exceeds a given minimal confidence thresholdΥ, with $0 \leq \Upsilon \leq 1$.

Association rule mining is a two-step process:

Step 1: Find all frequent itemsets. By definition, each of these itemsets will occur at least as frequently as a predetermined minimum support count.

Step 2: Generate strong association rules from the frequent itemsets. By definition, these rules must satisfy minimum support and minimum confidence.

Typically, association rules are considered interesting if they satisfy both a minimum support threshold and a minimum confidence threshold. Such thresholds can be set by users or domain experts.

Rules that satisfy both a minimum support threshold (min sup) and a minimum confidence threshold (min conf) are called strong association rules.

K-itemset, an itemset that contains *k* items. The set {computer, antivirus} is a 2-itemset. The occurrence frequency of an itemset is the number of transactions that contain the itemset. This is also known, simply, as the frequency or support count of the itemset. An itemset satisfies minimum support if the occurrence frequency of the itemset is greater than or equal to the product of min sup and the total number of transactions in **D**. If an itemset satisfies minimum support, then it is a frequent itemset. The set of frequent k-itemsets is commonly denoted by $\mathbf{L_K}$ [1].

3. INCREMENTAL MINING

Certainly constrained mining of association rules is a stipulation for interactive mining but on the same basis incremental mining is also inevitable if we are ever implementing true practical data mining systems [20]. Many research endeavors showed that producing the new set of rules after updates to the database can be appreciably profited & expedited by hitherto discovered association rules. This gives a great improvement over the naïve approach of running a traditional approach algorithm over the new set of data no matter how good such an algorithm.

The various explorers concentrated on providing discrete solutions for the two problems. Incremental association mining algorithms converged on maintain the discovered rules that have the equivalent support constraints, namely those which qualify for the same setting of minimum support as the fundamental database afore updates. On other side, the framework commenced in [21] and the pioneering of other kinds of constraints mutually with the conception of anti-monotonicity & terseness of constraints aided the purpose of augmenting the execution of the mining algorithm for a set of user-specified constraints.

The so far discussed points can be précised by mentioning that the significance of mining association rules & the obligation of interactive & ad-hoc mining simultaneously with the need for incremental mining of associations highly motivate hanker for effectual techniques to incrementally mine association rules with the constraints other than the conventional minimum support [15]. Hitherto, no such effectual algorithms have been proposed in the literature.

4. PROPOSED ALGORITHM

Problem statement: Let Database ***D*** with item transactions and db be the incremental updates of this database. The database displays only the items purchased. While the quantities of items purchased are not concerned. While Itemset is a non-empty set of items, $< i_1\ i_2\ i_3 \ldots n >$. A Sequence is an ordered list of itemsets, $< s_1\ s_2\ s_3 \ldots n >$.

A sequence $< a_1\ a_2\ a_3 \ldots n >$ is contained in $< b_1\ b_2\ b_3 \ldots n >$ if there exist $i_1 < i_2 < ... < i_n$ such that $a_1 \subseteq bi_1, a_2 \subseteq bi_2, \ldots a_n \subseteq bi_n$.

E.g., < (3)(4 5)(8) > $\subseteq$ < (7)(3 8)(9)(4 5 6)(8) >,
since (3) $\subseteq$ (3 8), (4 5) $\subseteq$ (4 5 6) and (8) $\subseteq$ (8)

However, note that sequence < (3)(5) > $\not\subseteq$ < (3 5) > (and vice versa)

A sequence of transactions ("shopping baskets") ordered by transaction times

$$T_i : < itemset(T_1)\ itemset(T_2) \ldots itemset(T_n) >$$

The task is to find the maximal sequences among all sequences in the given a database ***D*** of transactions with the incremental db datasets which are added after a time period. Each such maximal sequence represents a sequential pattern, which is the output of proposed algorithm.

The study proposed novel incremental frequent pattern mining method, which can discover frequent pattern itemsets. It can efficiently identify all frequent itemsets that occur in incremental database in particular periods when the new transaction data are added into the original transaction database.

Consider an original and an incremental customer transaction database. Incremental database may contain new transactions for new customers. To compute the set of sequential patterns in the updated database, we want to avoid counting everything from the scratch. Some main things one has to consider are as follows:

• Discover all sequential patterns not frequent in the original database but become frequent with the increment.

• Examine all transactions in the original database which can be extended to become frequent.

• Old frequent sequences may become invalid when adding new entries.

The proposed algorithm follows the approaches of Update with early pruning and Improved Apriori algorithms. It prunes an item set that will become small from the set of generated candidates as early as possible by a dynamic look ahead pruning strategy. It generates and counts the less number of candidates in the new database.

The Length of a sequence is the number of itemsets in the sequence. A sequence of length ***k*** is called ***k***-sequence. A sequence concatenated from sequences ***x*** and ***y*** is denoted by ***x.y***. ***DB*** is the original database, while db is the increment database. ***U = DB ∪ db*** is the updated database containing all sequences from ***DB*** and ***db***. ***LDB*** is the set of frequent sequences in DB. The task is to find frequent sequences in ***U***, noted L^U.

Counting supports of candidates is a serious problem. The total number of candidates can be very large. One transaction may contain many candidates.

We used the Tokens which solves the problem easily & fast. A token stores the intermediate data at multiple levels.

Main Algorithm

```
Forall 1to k-sesequence in db
    find the counts of all the sequences of $C^K_{db}$ in db
    $T^K_{db}$= All K-sequence in $C^K_{db}$ with support $\geq$ minsup in db
    $L^K_{DB} - T^K_{db}$
Call pruning algo
        Add X to $L_{DB+db}$ and $L^k_{db}$
      Add X to $L_{DB+db}$ and $L_{db}$
for ( k=n; k>1; k--)
      for each k-sequences $S_k$ do
      delete all subsequences from $L_{DB+db}$
end for
```

Pruning Algorithm

1. Start pruning with dataset until dataset ends
2. Retrieve first set from dataset into X
3. Find frequent data in database, if not frequent remove set from DB.
4. If dataset is frequent add at last to lower set of database and remove temp dataset.

Advantages of Proposed Algorithm:

1. Effective tool that participates greatly in having well structured retail websites.
2. The Algorithm generate less no. of candidate sets.
3. Saves a lot of maintenance effort needed in the future.
4. It traverses the database only where it is really required.
5. The algorithm can be used for dynamic database.
6. Introduced a measure of interestingness.

Moreover, when a database is incremented there is need to make changes in the application following the proposed algorithm. The only thing important is requirement of multithreaded important.

Dynamic Look Ahead strategy is used in pruning. When database is updated it is done in a special way in which existing huge itemsets are detected & removed that may be no more remain longer after adding new transactions

5. RESULTS

The presented algorithm is executed on the real transaction datasets of Frequent Itemset Mining Dataset Repository, which could be downloaded from http://fimi.cs.helsinki.fi/data/. The proposed algorithm worked efficiently with the said datasets after some revisions and generated the correct output.

Dataset 1= T10I4D100K.DAT	(3.83 MB)
Dataset 2=T40I10D100K.DAT	(14.8 MB)

The dataset contains the day wise entries of purchased items irrespective of their quantities. The used dataset has the records of over 200 days, making it potentially useful dataset for mining purposes. The dataset is first filtered as the requirements and then the task of finding the maximal sequence pattern which occurred frequently is performed. Thereafter, we have incremented the dataset with db datasets and again executed the program to analyze the effect. In the below table, the results clearly show the inclination of refinements with respect to the increments of datasets. The task of mining in the chosen dataset was time consuming as the dataset were real and having the size of over 14 Mb.

The results are as follows:

Table 1. Result with split database (80% & 20%)

	Test 1	
	80% (DB)	+20% (db_1)
T10I4D100K	(21,59,73,108,225, 392,790,801,962)	(21,59,73,108,225,392,**631**,790, 801,962)
T40I10D100K	(10,104,576,598,641,753,849,883,988)	(10,**24**,104,576, 598,641,753,849,883,**887**,**913**,988)

Graph 1: Result with split database (80% & 20%)

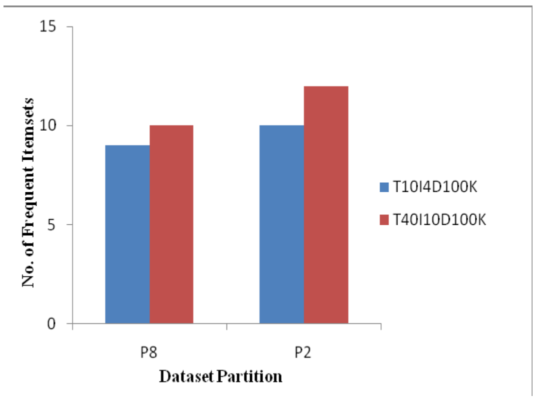
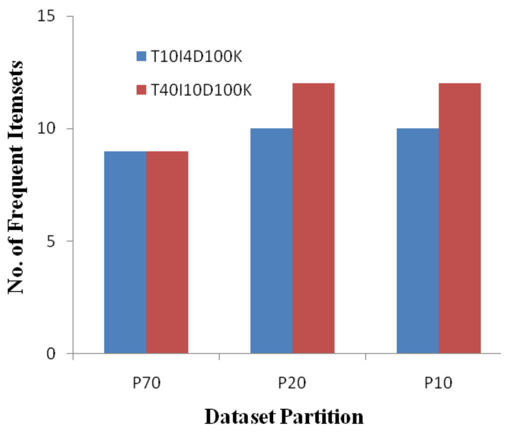


Table 2: Result with split database (70%, 20% & 10%)

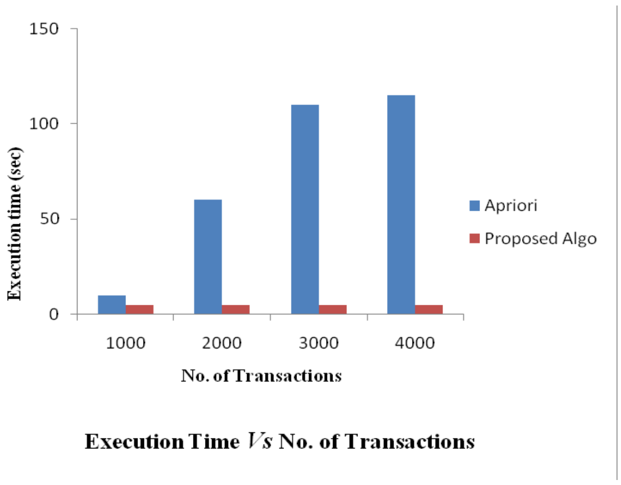
	Test 2		
	70% (DB)	+20% (db_1)	+10% (db_2)
T10I4D100K	(21,59,73, 108,225,392,790,801,962)	(21,59,73,108,225,392, **631**,790,801,962)	(21,59,73,108, 225,392,631,790,801,962)
T40I10D100K	(10,104,576,598,641,753,849, 883,988)	(10,**24**,104, 576,598,641,753,849,883,**913**,988)	(10,24,104,576,598,641,753,849,883,**887,** 913,988)

Graph 2: Result with split database (70%, 20% & 10%)



We compared the proposed algorithm with Apriori & get the results illustrated in the figure shown below:

Graph 3: Apriori Vs Proposed Algo



Further, the algorithm encountered some filtering problems in initial phases, which were rectified at the time of realization. The process of mining is repeated over several cut-sets of dataset, while here only the normalized versions are shown.

6. ACKNOWLEDGMENTS

Authors are greatly thankful to Dr. J.L. Rana, Director of RGI, Bhopal & Prof. Ashish Khare, Head of Department, at R.I.T.S, Bhopal, Dept. of Computer Science and Engineering, for giving his valuable time and resources that helped me to do my research work. Also, we would like to thank our faculty members and Radharaman Group of Institutions, Bhopal, India for providing valuable suggestions in this research work.

7. REFERENCES

[1] Jiawei Han & Michelien Kamber, Edition 2003. “Data Mining: Concepts & Techniques”, published by Elsevier, ISBN: 81-8147-049-4, pp 226-230

[2] Ayan N.F., Tansel A.U & Arkun E., 1999. “Ann efficient algorithm to update large itemsets with early pruning”, In Proc. of 5th ACM SIGKDD international conference on

Knowledge Discovery & Data Mining, CA USA, pp 23-29.

[3] D. Cheung, J. Han, C.Y. Wong, V. Ng, 1996. "Maintenance of discovered association Rules in large Databases: An Incremental Updating Technique", In Proc. of 1996 Int'l Conf. on Data Engg. (ICDE'96), USA.

[4] D. Cheung, S.D. Lee & Kao, 1997. "A general incremental technique for maintaining discovered association rules", In Proc. of Int'l Conf. on Database systems for adcanced Apllications, Australia, pp. 185-194.

[5] R. Feldman, Y. Aumann, A. Amir & H. Mannila, 1997. "Effeicient algorithms for discovering Frequent sets in Incremental databases", nI Proc. of SIGMOD Workshop on DMKD, Arizona.

[6] V. Ganti, J.E. Gehrke & Rmakrishanan, 2000. "Mining & monitoring evolving data", In Proc. of 16th International Conf. on Data Engg., San Diego.

[7] D. Cheung, S.D. Lee, 1997. "Maintaenance of discovered association rules : When to update? ", In Proc. of ACm-SIGMOD workshop on DMKD, Tucson.

[8] E. Omiecinski, A. Savasere, 1998. "An efficient mining of association rules in large dynamic databases" In Proc. of BNCOD, pp.13-24.

[9] N.L. Sarda & N. V. Srinivas, 1998. "An adaptive algorithm for incremental mining of association rules", In Proc. Of DEXA workshop, pp. 24-245.

[10] S. Thomas, S. Bodagala K. Alsabti & S. Ranka, 1997. "An efficient algorithm for incremental updation of association rules in large databases", In Proc. Of 3rd Int'l Conf. on KDD, CA.

[11] J. Bayardo, R. Agarwal & D. Gunoplulos, 1999. "Constraint based rule mining in large dense databases", In Proc. Of 15th Int'l Conf. on Data Engg., pp188-197.

[12] P. Bardely, U. Fayyad & O. Mangasarian, 1998. "Data mining: Overview & optimization opportunities" Microsoft Research Report MSR-TR-98-04.

[13] L.V.S. Laxmanan, R. Ng., J. Han & A. Pang, 1999. "Optimization of Constrained Frequent set queries with 2-variable constraints", In Proc. Of ACM-SIGMOD Conf. on Management of data, pp.157-168.

[14] R. Srikant & R. Agarwal, 1997. "Mining association rules with item constraints", In Proc. Of 3rd Int'l Cf. on Knowledge Discovery in databases & Data Mining, pp.67-73.

[15] J. Han & J. Pei & Y. Yin, 2000. "Mining frequent patterns without candidate generation" In Proc. of ACM-SIGMOD Int'l Conf. on Management of data, Dallas.

[16] M. Klemettinen, P. mannila & A. Verkamo, 1994. "Finding interesting rules from large sets of discovered association rules", In Proc. of 3rd Int'l Conf. on Information & Knowledge management, pp.401-407.

[17] H. Toivonen, M.Klementtinen & K. Hatonen, 1995. "Pruning & grouping discovered association rules", In MLnet workshop on Statistics, machine Learning & discovery in Databases, pp.47-52.

[18] C. Agarwal & P.H. Yu, 1998. "A new framework for itemset generation", In Proc. of 7th ACM-SIGACT-SIGMOD-SIGART Symposium on Principles of database systems, Washington.

[19] R.J. Bayardo & R. Agarwal, 1999. "Mining the most interesting rules", In Proc. of 5th ACM-SIGKDD Int'l Conf. on KDD, CA, USA.

[20] J. Han, L.V.S. Laxmanan, R. Ng., 1999. "Constraint-based, Multidimensional data mining",IEEE Computer Special issue on data mining.

[21] R.T. Ng., Lasshmanan, J.Han & A. Pang, 1998. "Exploratory mining & pruning optimizations of constrained association rules", in Proc. of ACM-SIG-MOD Int'l Conf. on Management of Data, pp. 13-24