International Academic Institute for Science and Technology

International Academic Journal of Science and Engineering
Vol. 3, No. 1, 2016, pp.201-207.
ISSN 2454-3896

International Academic Journal of Science and Engineering

www.iaiest.com

New parallel algorithm to frequent item set mining based on bit matrix in the form of bilateral process

Jaber Hosseinzadeh[a] , Abdoreza Savadi[b]

[a] *Laboratory of Data and Communication Security, computer engineering, Ferdowsi university of Mashhad.*

[b] *Computer engineering, Ferdowsi university of Mashhad.*

Abstract

Frequent item set mining is of the most significant subjects of association rules, which are one of the most important branches of data analysis. Frequent item set mining is the solution of this matter and the key point of the research as well. Primary researches include series of algorithms such as Apriori and FP-growth. Parallel algorithms were introduced after series ones in order to reduce work performance time by increasing the number of item sets. The present article proposes a novel parallel algorithm based on bit matrix in the form of bilateral process. The proposed algorithm tries to reduce work performance time and also take the benefits of frequent and non-frequent sets properties related to decreasing the number of investigable sets by means of mixing up-down and bottom-up problem solving approaches. Moreover, it uses a bilateral process to solve problems and to find frequent and infrequent sets. To be just and have authentic results, the same computer system and circumstances were utilized for the experiments and applications. Analyses and evaluations of the experiments results revealed an approximately 20% promotion in performance time in compared with the source algorithm of this study so that the effectiveness of the proposed algorithm is proved.

Keywords: Parallel algorithm, Bilateral process, Frequent item set mining, Bit matrix, Association rules

Introduction:

Association rules are one of the most important branches of data analysis and frequent item set mining is the solution of this matter as well as the key point of the research (Han, J W. and Kamber, M. 2005). Frequent item set miningwas first introduced by Agrawal et al. İn 1993 (Agrawal, R. 1993). Numerous databases can be expressed based on frequent item set having supermarket database as the most important one, in which transactions and data items are equivalent to the customers and the goods in the store, respectively. Thus, each customer selects their desired goods (Tias Guns et al,2011). Primary researches beared series algorithms like Apriori (Agrawal, R. 1993) and FP-growth Han, Jiawei et al 2004). Apriori probes -1, -2,..., -k frequent item sets in a repetitive manner. Where k, is the maximum number of items in the item set. İn each repeatition the algorithm produces the candidate item set and determines if it is frequent or infrequnt via caunting the number of set item determining transactions. On the other hand, FP-growth works in form of deviding and solving. İt functions reversely creating database and conditional FP-tree. Then, the FP-tree is analyezed in pattern growth method, which works through mixing productive frequent patterns of conditional FP-tree (Wenbin, Fang et al 2009). The higher the number and aspecs of the item set, the less efficiency of series algorithms. İn other words, for enormous databases parallel algorithm research is assumed as another focous on frequent item set mining (ZONG-YU and YA-PING, 2012).

Taking I={I1,I2,…,In}a set of items and A any subset of l including k as a member, A is a k-item set and D={T1,T2,…,Tm} is a database of m transactions and any T_k transaction is a set of items selected by the transaction, while every transaction is non-empty and a subset of l. For each set one can define a parameter, φ, in form of a function with an extent of a subdivision of items set and their domain, which is a subdivision of transactions, is described as follow:

İf T is transactions identity and D database is defined as bit matrix, then

$$T = \{1,2,\dots,m\} \qquad (1)$$

$$\varphi(I) = \{t \in T \mid \forall i \in I : D_{ti} = 1\} \qquad (2)$$

(A) support is the number of transactions that have selected A set and are actually the extent of φ parameter (Luc De, Raedt et al 2010, Luc De, Raedt et al 2008 ,Siegfried, Nijssen et al 2009). Conditioning that the number of these transactions are not fewer than the minimum support then the result set is frequent. Otherwise, it is infrequent (ZONG-YU and YA-PING 2012, Tias, Guns et al. 2011, Luc De, Raedt et al 2010, Luc De, Raedt et al 2008 ,Siegfried, Nijssen et al 2009). Finding frequent sets is defined as the goal.

Considering the benefits of computer bit function, the present article proposes a new parallel algorithm which takes into account a bit matrix in a bilateral approach. It is effectively applied by multicore computers illustrating its advantage towards series algorithms like Apriori and parallel algorithms such as (ZONG-YU and YA-PING, 2012) as well.

New parallel algorithm for frequent item set mining

In this section a new algorithm is introduced based on a bit matrix (a matrix with columns of items and rows of transaction). The amount of matrix equals one if the itemed transactions are adopted, or else, it is zero. The proposed algorithm is based on a bilateral process. The main idea of this algorithm is established on two principles:

1. Any supraset of an infrequent set is an infrequent set.
2. Any subset of frequent set is a frequent set.

To explain the first principle, one can say that if a new element is added to infrequent set, the resulted set is infrequent. As for the second principle, if a set is frequent, then every non-empty member of it is frequent. The algorithm combines up-down and bottom-up methods. Up-down mining is used to specialize subsets to the subsidiary frequency analyzing threads while bottom-up mining is utilized to report the frequency state to the main thread. Then, the main thread conveys subsets to the subsidiary threads and they subsequently send the report back to the main thread. The previous algorithm in (ZONG-YU and YA-PING, 2012) employs a one-way method that follows principle 1. The proposed algorithm in the article uses both principles consuming less time in its intermediate state rather than (ZONG-YU and YA-PING, 2012) algorithm and series ones. Since the processors belong to the same type it is possible to apply the same number of them to analyze basic principles in bilateral form. Half of the processors evaluate principle 1 beginning from smaller subsets and extending to the bigger ones. The other half begins with the bigger subsets and move toward smaller ones.

Algorithm steps

1. Create bit matrix in this way: if $I_j \in T_i$ in bit matrix, a matrix with columns of items and rows of transactions, it is one. Unless its amount in the matrix is zero.
2. the first step main thread devotes half of he subsidiary threads to evaluating the frequency state of -1(extension toward bigger sets) and the other half are devoted to –n frequency state assesment (decreasing toward smaller sets), where n is total number of items.
3. Yielded reports from subsidiary threads in the second step are evaluated by the main thread and throughly analyzed subsets, which are determined either frequent or infrequent, are labeled by it in order not to be experimented in the following steps.
4. Finishing all -1 frequents, half of the processors analyze -2 frequents and the other half work on -(n-1) frequent sets after completing frequent –n set.

An example of algorithm

An application of algorithm is illustrated as an example in table.1, which shows database of the example.

Table.1. database of the example in bit matrix form

	I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7
T_0	1	1	1	0	1	0	1	1
T_1	0	0	1	1	1	1	0	0
T_2	1	1	1	0	1	0	1	0
T_3	1	1	1	1	1	1	1	0
T_4	1	1	1	1	1	1	1	0
T_5	1	1	1	1	1	1	1	0
T_6	0	0	1	1	1	1	0	0
T_7	0	0	1	1	1	1	0	0

Considering min-support = 4 and the number of processor=-4, table.1 gains through applying step 1. In step 2, the main thread specializes I_0 and I_1 one-membered sets to the two first processors and {I_0, I_1, I_2, I_3, I_4, I_5, I_6, I_7} and {I_0, I_1, I_2, I_3, I_4, I_5, I_6} sets to the two next processors. The two first processors report infrequency in order to filtrate next steps and the two next processors report frequency for filtering for next steps. Analyzing I_0 set the first processor find it frequent and so does the second processor for I_1. The third and fourth processors analyze {I_0, I_1, I_2, I_3, I_4, I_5, I_6, I_7} and {I_0, I_1, I_2, I_3, I_4, I_5, I_6} sets and find them infrequent and frequent, respectively. These reports are sent to the main thread. In the next step the reports received from subsidiary threads are analyzed and applied for next steps filtration by the main thread. To do this, the report received from the forth processors is used to frequently mark all {I_0, I_1, I_2, I_3, I_4, I_5, I_6} subsets. For instance, {I_0} and {I_1} sets will be labeled frequent. In other words, in this case {I_0} and {I_1} sets will not be evaluated by the third and fourth processors and analyses are limited to {I_3} and {I_4} subsets in first step, before the selected subsets in the following steps by the first and second processors. After applying these filters in the next step the first processor analyzes {I_7} set and finds it infrequent. The second processor analyzes {I_0, I_7} and reveals it infrequent. {I_0, I_1, I_2, I_3, I_4, I_5, I_7} and {I_1, I_2, I_3, I_4, I_5, I_6, I_7} sets are analyzed by the third and fourth processors, respectively, and are determined infrequent. The analyses will continue with the same manner to get to a state that the opposite side processors meet each other, i.e. second side gets to the evaluated element by the first side processor. Table. 2 illustrates the result of once application of processors.

Table. 2. The result of once application of processors

نخهای فرعی	P1	P2	P3	P4
Process set	{I_0}	{I_1}	{ I_0,I_1, I_2, I_3, I_4, I_5, I_6,I_7 }	{ I_0,I_1, I_2, I_3, I_4, I_5, I_6 }
Frequent sets	-----	-----	Every subset of this set will be marked as frequent	-----
Infrequent sets	-----	-----	-----	-----

The result of the second time application of processors is presented in table. 3.

Table. 3. The result of once application of processors

Subsidiary threads	P1	P2	P3	P4
Process set	{I_7}	{I_0,I_7}	{ I_0,I_1, I_2, I_3, I_4, I_5, I_7 }	{ I_1, I_2, I_3, I_4, I_5, I_6,I_7 }
Frequent sets	-----	-----	----	-----
Infrequent sets	Every supraset of this set will be infrequent	Every supraset of this set will be infrequent	-----	-----

Research method

The proposed algorithm (in this work) and (ZONG-YU and YA-PING, 2012) are applied in the same system and situations. Database is generated by system as a random sample and both algorithms have been evaluated at the same circumstances and database. Both algorithms have been applied on a computer with intel dual core 1.86 1.87 processors, work frequency of 2 Giga byte with ground state of 10 data items and 3000 transactions with random samples for database and 4 subsidiary threads with threshold amount of 3. **Windows 8** in **visual C#2010** programming language is the operating system. The results are showed in fig. 1 through a graph representing the percent of database selected items (the percent of selected items by database transactions) and performance time in millisecond.

Findings

To express the efficiency of the proposed algorithm, the author has performed it on a computer with intel dual core 1.86 1.87 processors, work frequency of 2 Giga byte with ground state of 10 data items and 3000 transactions with random samples for database and 4 subsidiary threads with threshold amount of 3. **Windows 8** in **visual C#2010** programming language is the operating system. According to fig. 1. It can be said that in case the percentage of items selected by database is zero, all the subsets will be infrequent. Therefore, every basic principle discussed in section 2 will not seem to be practically effective so that its performance time shows a great difference with the case of having 49.87466% item selection, representing a time promotion of 29.5 percent. As item selection percentage increases it is expected to have the time reduced. However, we should not ignore the significant fact that item selection percentage does not explain anything about the amounts distribution in the database. The more sporadic the distribution is, which means item selection amounts in database are distributed among numerous transactions, the closer yielded distribution gets to zero item selection percent time. And as the distribution gets denser, which means item selection amounts in database show a tendency toward a special transaction, its performance time differs greatly from zero item selection percent time, then we expect a remarkable time promotion. It must be noted that this situation will not be permanently true. Furthermore, as we get closer to 100 item selection percent state, the performance time gets nearer to zero item selection percent time. There are many subsets to be analyzed and in100 item selection percent state all the subsets appear in the final answer. Little difference in performance time of two states roots in the long connection time amongst processors in100 item selection percent state. In a state with49.885666 item selection percent the time increases in compared with 49.87466 state, because random variable creating algorithms more or less use previous amounts with some changes to have two independent quantities. If item selection quantities in the first state show a tendency to a certain transaction it is expected to have this tendency maintained in the second state. Even though these two quantities are independent, the independency is also true among total data. So dispersion increases rather than the first state and then the resulted performance time will be extended. Continuation of this process leads to a state with a much more dispersion rather than 49.87466 state. For example, take two amounts of item selection percentages equal to 49.87466 and 49.96633. The dispersion in latter state is more than the former and the performance time of the algorithm with 49.96633 item selection percent varies extremely from the one with 49.87466 percent. By selecting random quantities obtained from random sample with 49.87466 item selection percent, the random quantities go back to the dense state. As a result there is a tendency toward specific transactions and a 32% promotion is observed in its performance time. In all of the above-mentioned cases the performance time of the proposed algorithm on average represent better outcomes in compared with (ZONG-YU and YA-PING, 2012) algorithm. According to the results, an approximately 20% promotion is observed on average mode.

Item selection percent in all database

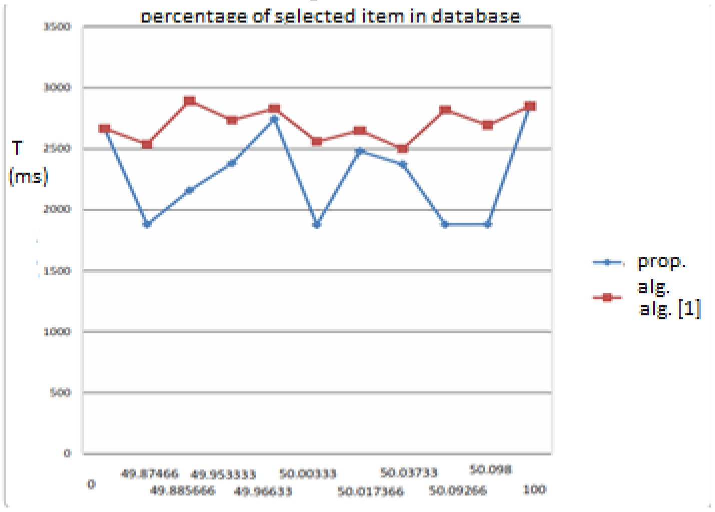

Fig. 1. Performance of the proposed algorithm and (ZONG-YU and YA-PING, 2012) algorithm on database with random sample

Discussion and conclusion

Bilateral process has never been used in previous approaches yet. For instance, in mentioned algorithm in (ZONG-YU and YA-PING, 2012) a one way process has been utilized. It means that developing and creating new sets is along with generating bigger subsets until getting general set of items. In the present algorithm principle 1 is used in order to reduce the number of sets which are going to be analyzed. The proposed algorithm in this article is the first algorithm which takes bilateral process into account to reduce the number of evaluable sets. The recommended principles stated in "new parallel algorithm to mine frequent item set" section are the main basis for this function. So that through the first principle and finding the infrequent set, the supra sets obtained from this set are assumed and introduced as infrequent. Then the second principle, which is only used in this work, introduces the subsets of a set, which is analyzed as a frequent set, as frequent sets. Two bilateral process approaches (in this work) and one way process (ZONG-YU and YA-PING, 2012) were applied and evaluated. The result certifies a 20% promotion in performance time of the proposed bilateral algorithm.

References:

Agrawal, R. (1993). Mining association rules between sets of items in large databases. Washington, DC. Proceedings of ACM SIGMOD Conference on Management of Data,1993,pp207-216.

Han, J W. and Kamber, M. (2005). Data mining: concepts and techniques,Translator Fan Ming,Meng Xiao-feng. BeiJing. China Machine Press,2005.

Han, Jiawei. and Pei, Jian and Yin, Yiwen and Mao, Runying. (2004). . Mining frequent patterns without candidate generation: A frequent-pattern tree approach. Data Mining and Knowledge Discovery, 2004.

Luc De, Raedt and Tias, Guns and Siegfried, Nijssen. (2008). Constraint Programming for Itemset Mining, KDD'08, August 24–27, 2008, Las Vegas, Nevada, USA.

Luc De, Raedt and Tias, Guns and Siegfried, Nijssen. (2010). Constraint Programming for Data Mining and Machine Learning. Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI-10), 2010

Siegfried, Nijssen and Tias, Guns and Luc De, Raedt, (2009). Correlated Itemset Mining in ROC Space: A Constraint Programming Approach. KDD'09, June 28–July 1, 2009, Paris, France.

Tias, Guns and Siegfried, Nijssen and Luc De, Raedt. (2011). Itemset mining: A constraint programming perspective, Artificial Intelligence 175 (2011) 1951–1983, 2011 Elsevier

Wenbin, Fang and Mian, Luand and Xiangye, Xiaoand and Bingsheng, Heand and Qiong, Luo. (2009). Frequent Itemset Mining on Graphics Processors, Proceedings of the Fifth International Workshop on Data Management on New Hardware (DaMoN 2009) June 28, 2009, Providence, Rhode-Island

ZONG-YU, ZHANG. and YA-PING, ZHANG.(2012). A parallel algorithm of frequent itemsets mining based on bit matrix. International Conference on Industrial Control and Electronics Engineering. IEEE 2012