ISSN (Print) : 0974-6846
Indian Journal of Science and Technology, *Vol 9(44), DOI: 10.17485/ijst/2016/v9i44/98864, November 2016* ISSN (Online) : 0974-5645

Performance Comparison of Frequent Pattern Mining Algorithms for Business Intelligence Analytics

S. Dharani[1*], B. Justus Rabi[2], S. S. Darly[3] and A. N. Nanda Kumar[4]

[1]Department of CSE, Dr.M.G,R. University, Chennai – 600095, Tamil Nadu; India; sd_mtech@yahoo.com
[2]Shri AndalAlagar College of Engineering, Mamandur- 603111, Tamil Nadu, India; bennisrobi@rediffmail.com
[3]Department of EEE, Anna University, Chennai - 600025, Tamil Nadu, India
[4]R.L.Jalappa Institute of Technology, Bangalore – 561203, Karnataka, India; nandakumar57@hotmail.com

Abstract

Objectives: In this paper, a simple and flexible partition algorithm has been proposed to mine frequent data item sets. This partition algorithm is different from other frequent pattern mining algorithm like Apriori algorithm, AprioriAllHybrid algorithm etc. **Method:** Partition algorithm concept has been proposed to increase the execution speed with minimum cost. Initially only for one time the database is scanned and separate partitions will be created for each sets of itemsets, which is 1-itemset, 2-itemsets, 3-itemsets etc. **Findings:** The scanning of whole database is not necessary to get the count of an itemset, it is enough to get the count of each data itemsets from its partition. This partition algorithm approach is implemented and evaluated against AprioriAllHybrid and Apriori algorithm. The candidate itemsets generated at each step is reduced and the scanning time is also reduced. The proposed methodology performance is significantly better than other algorithms and it promotes the faster execution time for mining frequent patterns. **Applications:** This proposed algorithm is used in areas like retail sales, production, universities, finance, banking systems and for business to plan and estimate the future values.

Keywords: AprioriAllHybrid, Apriori Algorithm, Data Mining, Frequent Pattern Mining, Partition

1. Introduction

In recent years, one of the most challenging research areas of data mining is how to effectively findthe frequent item sets present in the data warehouse or from massive datasets[1]. Data Mining is the most valuable and widely used process for the exploration and analysis of large quantity of data to acquire valid, novel, potentially useful and intelligent patterns hidden in database. Several algorithms are there for finding a sequential pattern from massive datasets. Apriori algorithm is the most standard algorithm for mining association rule and for findingfrequent patterns from huge datasets,needs the database to be scanned for multiple passes. Many algorithms have been proposed to mine the frequent patterns from large database D[2–4]. In this paper, a new efficient partition algorithm have been proposed, the basic idea is to identifying and projecting the frequent itemsets from large database in an efficient manner.

By having separate partitions for 1- itemsets, 2-itemsets, 3- itemsets, 4-itemsets etc., from the partition of those particular itemsets, get the number of occurrences of each itemsets, instead of from scanning the database D. The execution speed is decreased and the performance is increased. The proposed Partition algorithm is compared with Apriori and AprioriAllHybrid algorithm. All these algorithms are implemented using PL/SQL in Oracle Database and the performance is evaluated using execution time. Compared with other algorithm, partition algorithm is the fastest algorithm for discovering frequent itemsets or patterns.

2. Apriori Algorithm

Apriori algorithm uses previous information of properties of the frequent itemset to find the itemsets which is

* *Author for correspondence*

occurring frequently. Here, n-itemsets are used to get (n+1)-itemsets. To form the set of frequent 1-item set, scan the database for the occurrences of count of each item and collecting those items that satisfies the minimum support count. The resulting frequent 1- itemset is used to find frequent 2-itemset, the set of frequent 2-itemsets, is used to find frequent 3 - itemset, and so on, until no more frequent n-itemsets can be found. Joining and pruning are the two steps to discover frequent itemsets[5,6].In join, the candidate k itemsets$C_{k,}$ are generated by joining frequent k-1 itemsets with frequent k-1 itemsets.

In Pruning, for obtaining the number ofcandidate in C_k, the database is scanned, thecandidates whose count greater than or equal to the min_supcount are frequent and therefore belong to L_k. Using Apriori property the number of candidate K itemsets are reduced. Strong rules are a rule which satisfies both a minimum confidence threshold and minimum support threshold (*min_sup*). In this method, large numbers of candidate itemsets are generated and increase in records in the database results in too many I/O spending. To mine the frequent item sets many updated Apriori algorithm have been derived.

3. AprioriAllHybrid Algorithm

The main features of [7]AprioriAllHybrid algorithm are to efficiently discover all frequent patterns which satisfy the minimum support count. Letthe items with set of nunique attributes be I = $\{i_1,i_2,..,i_n\}$.Assume that all items in an itemset are occurring at the same time. An itemseti is denoted as $(i_1,i_2, i_3\ldots,i_n)$, where i_i is an item. An itemset with n items are called an-itemset. A sequence a is denoted as $(a_1\rightarrow a_2\rightarrow\ldots\rightarrow a_q)$,where the sequence element a_i is anitemset. A sequence with n-items is called an-sequence. For example, (X→YZ) is a 3-sequence. In an itemset, an item can occur only once, butin different itemsets of a sequence,multiple times it can occur.

A sequence a = $(a_1\rightarrow a_2\rightarrow\ldots\rightarrow a_n)$ is a subsequence of another sequence b = $(b_1\rightarrow b_2\rightarrow \ldots\rightarrow b_n)$, denoted as a→b, if there exist integers such that $a_\varphi\subseteq b_{ij}$for all a_j. The sequence (X→YZ) is a subsequence of (YX→W→YZV), the element sequence areX→YX and YZ→YZV. The sequence (YX→W) is not a subsequence of (YXW) and vice-versa.a is a proper subsequence of q, denoted as a⊂b, if a⊂b and a⊂b. A transaction contains a set of items I.Let the transaction list be $\{T_1, T_2\ldots Tn\}$. The database contains large number of transaction. A sequence is said to contain a sequence a, if a⊆b i.e., a is a subsequence of the customer sequence S.

The support of a sequence S, denoted as σ (a), is the total number of transactions that contains this sequence. With the minimum support, if a sequence occurs more than the minimum support times,then that sequence is appear to be frequent. $F_k$is denoted as the set of frequent k-sequences. The original database D is read only for one time and then a new temporary database D' is generated for the next iterations. Find the candidate 2 itemsets sequence using temporary database D',after completing the first iteration. Till thenthere is no candidate sequence or size of the temporary database becomes empty find the candidate k-itemset sequences. At this stage, both database size and number of candidate sequences generated were reduced with reduction in time complexity. For finding the candidate sequences, before beginning of the pass one, use set C_k. This algorithm has been applied in Cross Selling and Library transaction.

When this AprioriAllHybrid algorithm is used in the cross-selling environment, it helps the seller to know whether a particular item will sell out or not. These findings help to improve customer satisfaction and profit of sales. In Library transaction, it helps the librarian to find the frequently accessed books so that better services will be provided to the users by availing more number of copies of that book[8,9]. When compared with Apriori algorithm, AprioriAllHybridis two times faster than Apriori algorithm for large volumes of data, andhave better performance because,multiple passes over the original databases is reduced and total number of candidate sequences generated at each stage is decreased.

4. Partition Algorithm

Partition is a way to split tables, indexes, and index-organized tables into smaller pieces called partitions which enables the database objects to be managed and accessed at a finer level of granularity.Each partition is known by its specific name and has its own characteristics such as its storage and index. Partitioning is important, when the table size is greater than 2 giga bytes, tables contain historical data and the contents of a table need to be distributed across different types of storage devices. It is time efficient because it allows the maintenance and failure operation on a particular partition to be carried out on selected partitions while other partitions are available to users so that it improves the performance of operation. Many algorithms are there to mine frequent itemsets[10,11].

A partition concept has been proposed to increase the

execution speed with minimum cost. For each itemsets, that is one itemset 2-itemsets, 3-itemsets etc., a separate partition will be created during data insertion in to the table. Initially, frequent 1-itemsets is created by looking over the databaseand get the numbers of existences of each item from the partition of those particular items using the pointer, then that items sustaining the minimum support will be contained withinin the frequent 1-itemsets. To find the, frequent K itemsets in the available datasets, it is not required to scan the full database; it is ample to acquire the count of each data itemsets from its partition.

Table 1. Transaction database for partition algorithm.

TID	List of Items
T1	I1,I2,I4
T2	I1,I4,
T3	I3,I4,
T4	I1,I2,I5
T5	I2,I4,
T6	I1,I3,I2
T7	I1,I2,I5,I3
T8	I1,I3,I5

To find the count of each candidate in C_k, the partition of each itemsets will be checked and the count which is not less than minimum support count is frequent and belongs to L_k [12]. Then apply apriori property for reducing the size of the candidate k itemsets. Consider the Database, with eight transactions, here partitioning concept has been applied to store and retrieve the data from the database. Huge and different types of partitioning techniques are available; they are range, hash and list. The range partitioning has been used to increase the performancewhen mining the frequent patterns in the database. Ranges are always defined as an excluding upper boundary of a partition.

Table 1 shows a transaction database for partition algorithm and it contains 8 transactions.In this table, transaction T1 contains I1,I2,I3and transaction T2 contains I2, I4 and so on. From the candidate 1 itemsets produce the frequency 1 itemsets by computing the number of existences of the data items directly from the partition instead of scanning the whole database. The min_support is taken as 2. Candidate 1 itemset whichis satisfying the minimum support count will be included in frequent 1 itemset. The following Figure 1 shows, the generation of candidate 1 itemsets and frequent 1 itemsets. The candidate 2 itemsets are generated by joining frequent 1 itemset with frequent 1 itemset and check whether the subset of the frequent itemsets are also frequent.

In frequent 1 itemset all the items have been involved and there is no need to have pruning.For computing the support count, instead of look over the whole database, it is enough to get the count from the appropriate partition. The frequent 2 itemsets are formed from candidate 2 itemsets which is satisfying the minimum support count.

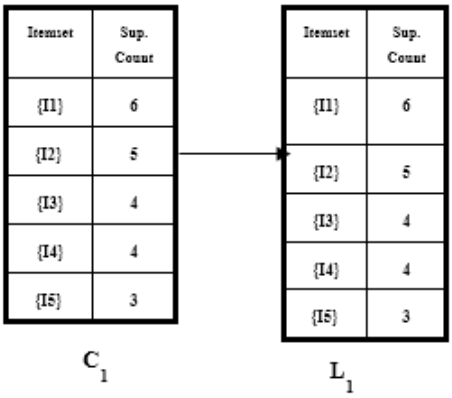

Figure 1. Generation of frequent 1- itemset using partition algorithm.

The Figure 2 shows the generation of frequent 2 itemsets using partition. Finally, 8, frequent 2 itemsets have been generated, and this 8, frequent 2 itemsets are used to generate the candidate 3 itemsets.

By joining L_2 by itself the candidate 3 itemsets are generated and find whether the subset of the frequent itemsets are also frequent, only the itemsets {I1,I2,I4},{I1 ,I2,I5},{I1,I2,I3} and{I1,I3,I5} have been included for the next step that is considered as candidate 3 itemsets since subsets of these itemsets are also a frequent itemset. The remaining itemsets have been removed because whose subsets are not frequent one.

For calculating the support count, instead of scanning the whole database, it is enough to get the count from the appropriate partition. The candidate 3 itemsets which is satisfying the minimum support count will be included in the frequent 3 itemsets.The following Figure 3 shows the generation of frequent 3 itemsets using partition. Table can be partitioned into smaller units using Partitioning techniques. SQL commands in oracle are used for managing partitioned tables, and that comprises including new partitions, breaking, combining, pruning and swapping partitions.Finally, 3 frequent 3 itemsets are generated, and thesethree frequent 3 itemsets are used to generate the candidate 4 itemsets.

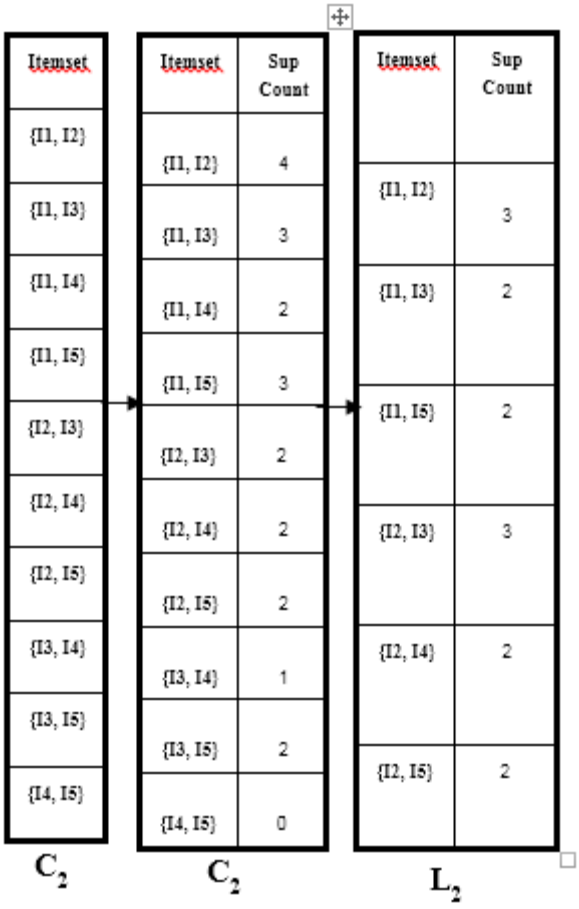

Figure 2. Generation of frequent 2- itemset using partition algorithm.

The candidate 4 itemsets are generated by joining L_3 by itself and check whether the subset of the frequent itemsets are also frequent, only the itemsets {I1,I2,I3,I5} have been included for the next step that is considered as candidate 4 itemsets since whose subset is also a frequent itemset[13].

Get the count directly from the appropriate rom the partition, instead of scanning the whole database. Here, the support count is 1, which is less than minimum support, so, L_4 = ¢ and algorithm terminates.With the frequent 3 itemsets, association rule is applied which shows the association between the data items. Instead of searching the whole database D, the itemsets can be directly taken from the corresponding partitions. So, the efficiency and performance of the algorithm has been increased for mining the frequent itemsets from large amount of datasets.

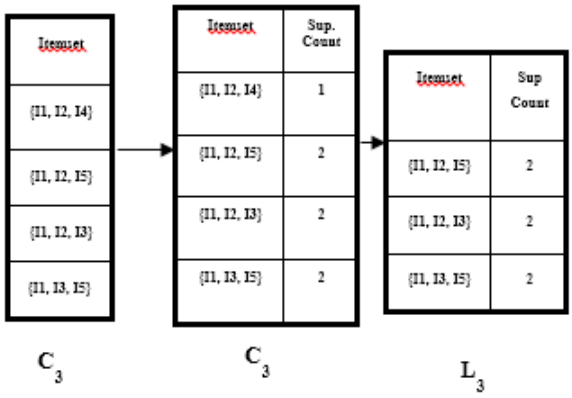

Figure 3. Generation of frequent 3- itemset using partition algorithm.

5. Comparison of Algorithms

Apriori algorithm, AprioriAllHybrid algorithm and Partition algorithm have been compared and evaluated by using PL/SQL in Oracle Database. The algorithms are compared with different sets of records like 30000, 20000 and 10000[14,15]. Many algorithms have been compared to mine the frequent patterns from large database. For 30000 records, Apriori algorithm takes 264 minutes, AprioriAllhybrid takes 135 min and partition algorithm takes 19 minutes only. For 20000 records, Apriori algorithm has taken 178 minutes, AprioriAllHybrid algorithm taken 82 minutes and Partition algorithm has taken 13 minutes.

For 10000 records, Apriori algorithm takes 86 minutes, AprioriAllHybrid algorithm takes 41 minutes and Partition algorithm takes 6 minutes. This performance evaluation is shown in the Table 2. From the above Table 2, depends on the number of candidate generation and scanningtime of the database, the execution time of the algorithm to find sequential pattern is calculated. With the increase in customer transactions in the database there will be increase in the execution time.

Table 2. Performance evaluation of Apriori, AprioriAllHybrid and Partition Algorithm.

S. No	ALGORITHMS	TOTAL NO OF RECORDS		
		30000	20000	10000
1	Apriori	264 min	178 min	86 min
2	AprioriAllHybrid	135 mins	82 min	41 min
3	Partition	19 mins	13 min	6 min

The partition algorithm is faster than any other algorithm like AprioriAllHybrid and Apriori Algorithm. Partition algorithm can reduce scheduled downtime and gains the performance, manageability and availability, because it allows the maintenance and failure operation on a particular partition to be carried out on selected partitions while other partitions are available to users so that it improves the performance of operation.

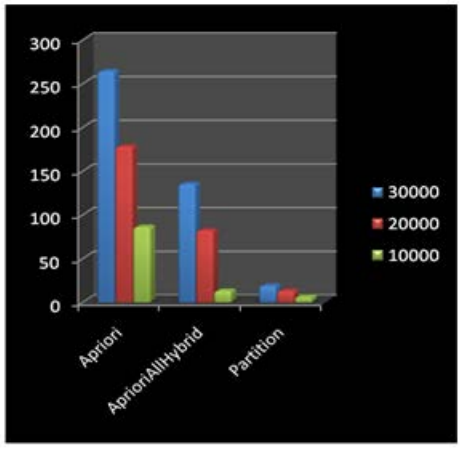

Figure 4. Performance graph of Apriori, AprioriAllHybrid and Partition algorithm.

Partition algorithm works very much faster for all datasets that ranges from gigabytes to terabytes. Table 2, specifies that, by having less execution time for discovering the frequent itemsets and association rule among the massive datasets, partition algorithm has performed powerfully and efficiently.The algorithms performance has shown in graphical representation in the Figure 4.

6. Conclusion

Apriori algorithm, AprioriAllHybrid and Partition algorithm have employed for large amount of data sets and the results are compared. To find the frequent patterns available in the massive database, Apriori algorithm is used. The disadvantage of this method is, the large number of candidates has been generated and scan time have increased because, for each time whole database have to be scanned. In AprioriAllHybrid algorithm, original database D is read only once, afterwards new temporary intermediate database is created at each step of iteration, as a result database size is reduced and no. of candidate sequences generated is also reduced.

The AprioriAllHybrid takes more execution time for finding frequent itemsets in giga or terabytes of data. To handle large amount of datasets more efficiently, partition algorithm concept has been included. In partition algorithm, separate partitions will be created for each 1 itemsets, 2 itemsets, 3 itemsets etc. so that the control will be directed to the appropriate partitions instead of scanning the entire database. Finally, the scan time is decreased and as a result performance is increased.Massive experimentation work was performed for evaluating and comparing Apriori, AprioriAllHybrid and Partition algorithms. The proposed partition algorithm approach consistently performs well than rest of the algorithm and will support many future researches in many ways.

7. Future Work

The proposed partition algorithm in this paper can be further improved in different perspectives. The future enhancements will focus on, the algorithm takes only table as input, which contains only text, and this can be improved by allowing any type of inputs. Association rule generated can be effectively optimized using GA or some other techniques, by producing strong association rules. Thus the algorithm can be enhanced in different perspective, which helps to improve the performance of the algorithm. By incorporating all the mentioned augmentations in the proposed system, an efficient, effective and intelligent method for mining frequent patterns can be developed.

8. References

1. Sun D, Teng S, Zhang W. An algorithm to improve the effectiveness of Apriori. Proceedings of IEEE International Conference on Cognitive Informatics; 2007.p. 385–90.
2. Zhou H, Zhejiang. Mining accurate top-K frequent closed itemset from data stream. Proceedings of IEEE International Conference on Computer Science and Electronics Engineering. 2012; p.180–4.
3. Wang F, Hua Y. An improved Apriorialgorithm based on the matrix. Proceedings of IEEE International Conference on BioMedical Information Engineering. 2008. p.152–5.
4. Guidan F, Shaohong Y. A frequent itemsets mining algorithm based on matrix in sliding window over data streams. Proceedings of IEEE International Conference on Intelligent System Design and Engineering Applications; 2013. p. 66–9.
5. Yang G, Zhao H, Wang L, Liu Y. An implementation of im-

proved Apriorialgorithm. Proceedings of IEEE International Conference on Machine Learning and Cybernetics; 2009. p. 1686–9.
6. Xu J, Wang Z, Hu L. Rough_Apriori algorithm and the application of an aid system of campus student major selection. Proceedings of IEEE International Conference on Research Challenges in Computer Science; 2009. p. 160–3.
7. Dharani S, Rabi JB, Nandakumar AN, Darly SS. Fast algorithm for discovering sequential patterns in massive datasets. Journal of Computer Science. 2011:1325–9.
8. Shi Y, Zhou Y. An improved Apriorialgorithm. Proceedings of IEEE International Conference on Granular Computing; 2010.p. 759–62.
9. Gupta A, Arora R, Sikarwar R, Saxena. Web usage mining using improved frequent pattern tree algorithms. Proceedings of IEEE International Conference on issues and Challenges in Intelligent Computing Techniques; 2014. p. 573–8.
10. Jain JK, Tiwari N, Ramaiya M. Mining positive and negative association rules from frequent and infrequent pattern using improved Genetic algorithm.Proceedings of IEEE International Conference on Computational Intelligence and Communication Networks; 2013. p.516–21.
11. Zhang S, Du Z, Wang JTL.New techniques for mining frequent patterns in unordered trees. IEEE Transactions on Cybernetics. 2015; 45(6):1113–25.
12. Addi AM, Cadi S, Tarik A. Comparative survey of association rule mining algorithms based on multiple –criteria decision analysis approach. Proceedings of IEEE international Conference on Control, Engineering and Information Technology; 2015. p. 1–6.
13. Chapter 1: Introduction – Shodhganga [Internet]. [cited 2015]. Available from: http://shodhganga.inflibnet.ac.in/bitstream/10603/42665/10.
14. Hashmi AS, Ahmad T. Big data mining techniques.Indian Journal of Science and Technology. 2016Oct; 9(37):1–5.
15. Jafarzadeh H, Torkashvand RR, Asgari C, Amiry A. Provide a new approach for mining fuzzy association rules using Apriorialgorithm. Indian Journal of Science and Technology. 2015 Apr; 8(8):127–34.