International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 2 Issue 7, July - 2013

Apriori Based Algorithms And Their Comparisons

Rupali
Student, Computer Engineering
University College of Engineering
Punjabi University
Patiala, India

Gaurav Gupta
Department of Computer Engineering
University College of Engineering
Punjabi University
Patiala, India

***Abstract* — Mining frequent itemsets from the large transactional database is a very critical and important task. The first algorithm for mining all frequent itemsets and strong association rules was the AIS algorithm by [2]. Shortly after that, the algorithm was improved and renamed Apriori. Apriori algorithm is the most classical and important algorithm for mining frequent itemsets. This paper aims to presents a basic Concepts of some of the algorithms (Direct Hashing and Pruning (DHP), Partitioning, Sampling, Dynamic Itemset Counting (DIC), Improved Apriori algorithm) based upon the Apriori like algorithm for mining the frequent item sets along with their capabilities and comparisons.**

***Keywords – Data mining, Apriori variations, Frequent Itemsets*.**

1. INTRODUCTION

Frequent item set mining is one of the most important and common topic of research for association rule mining in data mining research area. As an association rule mining is defined as the relation between various itemsets. Association rule mining takes part in pattern discovery techniques in knowledge discovery and data mining (KDD). As performance of association rule mining is depends upon the frequent itemsets mining, thus is necessary to mine frequent item set efficiently.

The process of extracting association rule mining consists of two parts firstly, mine all frequent itemsets pattern each of these pattern should satisfy the minimum support threshold. Once these entire frequent patterns are mined, then only second phase of mining i.e. association rules are produced from these frequent itemsets. These association rules must satisfy the minimum support and minimum confidence. This minimum support and confidence should be defined by the user. In this way the generation of association rule is largely depends upon the generation of the frequent items in the first phase. [1, 2, 3]

A large number of algorithms with different mining efficiencies were proposed by many researchers for generation of frequent itemsets. Any algorithm should find the same set of rules though their computational efficiencies and memory requirements may be different. The best known mining algorithm is Apriori algorithm.

The further organisation of this paper is as follows. In Section 2, we briefly define the problem statement for finding the frequent itemsets from transactional database. Section 3 defines the classic Apriori algorithm. Sections 4 define the existing techniques based upon the Apriori algorithm. Section 5 defines their comparisons. Section 6 concludes the paper.

2. PROBLEM STATEMENT

The problem of mining association rules over market basket analysis was introduced in [4] .i.e. finding associations between the items that are present in the transaction from the database. The database may be from any retail shop, medical or from any other applications [5]. As defined in [2] the problem is stated as follows:

Let I= i_1, i_2 . . . i_m be a set of literals, called items and m is considered the dimensionality of the problem. Let D be a set of transactions, where each transaction T is a set of items such that T ∈ I. A unique identifier TID is given to each transaction. A transaction T is said to contain X, a set of items in I. X ∈ T An *association rule* is an implication of the form "X → Y ", where X ∈ I , Y ∈ I, and X ∩ Y = ∅. An itemset *X* is said to be *large* or *frequent* if its *support s* is greater or equal than a given minimum support threshold σ. An itemset X satisfies a constraint C if and only if C (X) is *true*. The rule X → Y has a *support s* in the transaction set D if s% of the transactions in D contain X ∪ Y. In other words, the support of the rule is the probability that X and Y hold together among all the possible presented cases. It is said that the rule X → Y holds in the transaction set D with *confidence* c if c% of transactions in D that contain X also contain Y . In other words, the confidence of the rule is the conditional probability that the consequent Y is true under the condition of the antecedent X. The problem of discovering all association rules from a set of transactions D consists of generating the rules that have a *support* and *confidence* greater than a given threshold. These rules are called Strong Rules. This association-mining task can be broken into two steps:

1. Finding the frequent k-itemset from the large database.
2. Generate the association rule from these frequent item sets.

3. APRIORI ALGORITHM

The first algorithm for mining all frequent itemsets and strong association rules was the AIS algorithm by [1]. Shortly after that, the algorithm was improved and renamed Apriori. Apriori algorithm is, the most classical and important algorithm for mining frequent itemsets. Apriori is used to find all frequent itemsets in a given database DB. The key idea of Apriori algorithm is to make multiple passes over the database. It employs an iterative approach known as a breadth-first search (level-wise search) through the search space, where k-itemsets are used to explore (k+1)-itemsets.

The working of Apriori algorithm is fairly depends upon the Apriori property which states that" All nonempty subsets of a frequent itemsets must be frequent" [6]. It also described the anti monotonic property which says if the system cannot pass the minimum support test, all its supersets will fail to pass the test [6, 1]. Therefore if the one set is infrequent then all its supersets are also frequent and vice versa. This property is used to prune the infrequent candidate elements.

It is no doubt that Apriori algorithm successfully finds the frequent elements from the database. But as the dimensionality of the database increase with the number of items then:

- More search space is needed and I/O cost will increase.
- Number of database scan is increased thus candidate generation will increase results in increase in computational cost.

Therefore many variations have been takes place in the Apriori algorithm to minimize the above limitations arises due to increase in size of database. These subsequently proposed algorithms adopt similar database scan level by level as in Apriori algorithm, while the methods of candidate generation and pruning, support counting and candidate representation may differ. The algorithms improve the Apriori algorithms by:

- Reduce passes of transaction database scans
- Shrink number of candidates
- Facilitate support counting of candidates

4. APRIORI ALGORITHM VARIATIONS

4.1. Direct Hashing and Pruning (DHP):

It is absorbed that reducing the candidate items from the database is one of the important task for increasing the efficiency. Thus a DHP technique was proposed [7] to reduce the number of candidates in the early passes C_k for k>1 and thus the size of database. In this method support is counted by mapping the items from the candidate list into the buckets which is divided according to support known as Hash table structure. As the new itemset is encountered if item exist earlier then increase the bucket count else insert into new bucket. Thus in the end the bucket whose support count is less the minimum support is removed from the candidate set.

In this way it reduce the generation of candidate sets in the earlier stages but as the level increase the size of bucket also increase thus difficult to manage hash table as well candidate set.

4.2. Partitioning Algorithm:

Partitioning algorithm [8] is based to find the frequent elements on the basis partitioning of database in n parts. It overcomes the memory problem for large database which do not fit into main memory because small parts of database easily fit into main memory. This algorithm divides into two passes,

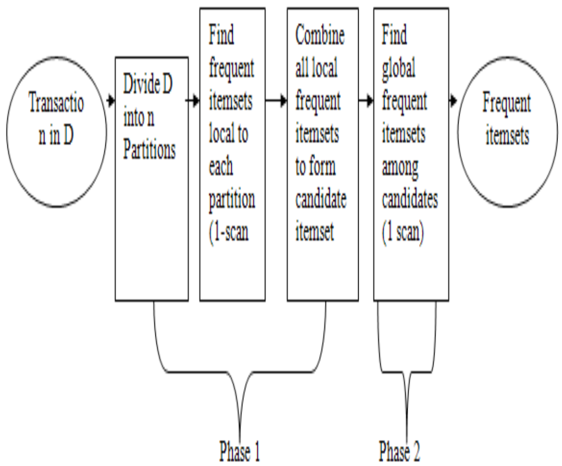

Figure 1: Mining Frequent itemsets using Partition algorithm [9]

It should be noted that if the minimum support for transactions in whole database is min_sup then the minimum support for partitioned transactions is min-sup number of transaction in that partition.

A local frequent itemset may or may not be frequent with respect to the entire database thus any itemset which is potentially frequent must include in any one of the frequent partition.

As this algorithm able to reduce the database scan for generating frequent itemsets but in some cases, the time needed to compute the frequency of candidate generates in each partitions is greater than the database scan thus results in increased computational cost.

4.3. Sampling Algorithm:

This algorithm [10] is used to overcome the limitation of I/O overhead by not considering the whole database for checking the frequency. It is just based in the idea to pick a random sample of itemset R from the database instead of whole database D. The sample is picked in such a way that whole sample is accommodated in the main memory. In this way we try to find the frequent elements for the sample only and there is chance to miss the global frequent elements in that sample therefore lower threshold support is used instead of actual minimum support to find the frequent elements local to sample. In the best case only one pass is needed to find all frequent elements if all the elements included in sample and if elements missed in sample then second pass are needed to find the itemsets missed in first pass or in sample [11].

Thus this approach is beneficial if efficiency is more important than the accuracy because this approach gives the result in very less scan or time and overcome the limitation of memory consumption arises due to generation of large amount of datasets but results are not as much accurate.

4.4. Dynamic Itemset Counting (DIC):

This algorithm [12] also used to reduce the number of database scan. It is based upon the downward disclosure property in which adds the candidate itemsets at different point of time during the scan. In this dynamic blocks are formed from the database marked by start points and unlike the previous techniques of Apriori it dynamically changes the sets of candidates during the database scan. Unlike the Apriori it cannot start the next level scan at the end of first level scan, it start the scan by starting label attached to each dynamic partition of candidate sets.

In this way it reduce the database scan for finding the frequent itemsets by just adding the new candidate at any point of time during the run time. But it generates the large number of candidates and computing their frequencies are the bottleneck of performance while the database scans only take a small part of runtime.

5. COMPARISON OF ALGORITHMS

Algorithm / Parameter	Apriori Algorithm	DHP Algorithm	Partition Algorithm	DIC Algorithm	Sample Algorithm
Storage Structure	Array Based	Array Based	Array Based	Array Based	Array Based
Technique	Use Apriori property and join and prune method	Use hashing tech for finding frequent itemsets	Partition the database for finding local frequent item first	Based upon database for finding local frequent item first	Pick any random sample for checking frequency of whole database at lower threshold support
Memory Utilization	Due to large amount of candidate are produced, require large memory space	Require less space at earlier passes but more in later stages	Each partition is easily occupy in main memory	Require different amount of memory at different point of time	Very less amount of memory is needed
Databases	Suitable for sparse datasets as well as dense datasets	Suitable for medium databases	Suitable for large databases	Suitable for medium and low databases	Suitable for any kind of datasets but mostly not give accurate results
Time	Execution time is more as time wasted in producing candidate at everytime	Execution time is small databases	Execution time is more because of finding locally frequent then globally	Execution time is small because dynamic itemsets are added acc to situation	Execution time is very much small

6. CONCLUSION

In this paper we studied the basic algorithm i.e. Apriori, to mine the frequent item sets. Apriori algorithm is, the most classical and important algorithm for mining frequent itemsets. Apriori is used to find all frequent itemsets in a given database DB. Apriori algorithm is associated with certain limitations of large database scans. Thus variations of Apriori come into existence. The various variations we discussed are DHP, Partition, DIC and Sampling algorithms. We also studied their comparisons.

REFERENCES

1. R.Agrawal, R.Srikant, “Fast algorithms for mining association rules”, Proceedings of the 20th Very Large DataBases Conference (VLDB’94), Santiago de Chile, Chile, 1994, pp. 487-499.
2. Jiawei Han, M.Kamber, “Data Mining- Concepts and Techniques”, Morgan Kanufmann Publishers, Sam Francisco, 2009.
3. R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In Proc. 1993 ACMSIGMOD Int. Conf. Management of Data, pages 207–216, Washington, D.C., May 1993.
4. J.Han, J.Pei and Y.Yin., “Mining frequent patterns without candidate Generation”, in: Proceeding of ACM SIGMOD International Conference Management of Data, 2000, pp.1-12
5. M.-L. Antonie and O. R. Za¨ıane. Text document categorization by term association. In IEEE International Conference on Data Mining, pages 19–26, December 2002.
6. Aggrawal.R, Imielinski.t, Swami.A. “Mining Association Rules between Sets of Items in Large Databases”. In Proc. Int’l

Conf. of the 1993 ACM SIGMOD Conference Washington DC, USA.

7. Park. J. S, M.S. Chen, P.S. Yu. "An effective hash-based algorithm for mining association rules". In Proc. ACM-SIGMOD Int'l Conf. Management of Data (SIGMOD), San Jose, CA, May 1995, pages 175–186.
8. A. Savasere, E. Omiecinski, and S. Navathe. "An efficient algorithm for mining association rules in large databases". In Proc. Int'l Conf. Very Large Data Bases (VLDB), Sept. 1995, pages 432–443.
9. Pei.J, Han.J, Lu.H, Nishio.S. Tang. S. and Yang. D. "H-mine: Hyper-structure mining of frequent patterns in large databases". In Proc. Int'l Conf. Data Mining (ICDM), November 2001
10. Toivonen.H. "Sampling large databases for association rules". In Proc. Int'l Conf. Very Large Data Bases (VLDB), Sept. 1996, Bombay, India, pages 134–145.
11. "Data mining Concepts and Techniques" by By Jiawei Han, Micheline Kamber, Morgan Kaufmann Publishers, 2006.
12. Brin.S, Motwani. R, Ullman. J.D, and S. Tsur. "Dynamic itemset counting and implication rules for market basket analysis". In Proc. ACM-SIGMOD Int'l Conf. Management of Data (SIGMOD), May 1997, pages 255–264.
13. "Data mining Concepts and Techniques" by By Jiawei Han, Micheline Kamber, Morgan Kaufmann Publishers, 2006.
14. Dongme Sun, Shaohua Teng, Wei Zhang, Haibin Zhu. "An Algorithm to Improve the Effectiveness of Apriori". In Proc. Int'l Conf. on 6th IEEE Int. Conf. on Cognitive Informatics (ICCI'07), 2007.