CHAPTER 3

SCALABLE ASSOCIATION RULE MINING USING PARALLEL PRUNING

3.1 INTRODUCTION

Data mining is an area that involves interdisciplinary research field, which provides the latest research results in database related work, artificial intelligence, supervised and unsupervised machine learning, statistics and logistics, knowledge engineering, information retrieval for a transaction, data visualization technology and so on. Association rule mining is considered as one of the most important data mining problems that are yet to be covered.

Initially, it is proposed to use Clustering techniques, that are used to group up the transactions based on the relevancy. Transactions are partitioned into small groups. K-means clustering algorithm is one of the widely used clustering algorithms. The random transaction based centroid selection model may choose similar transactions. In this case the cluster accuracy is limited with respect to the distance between the centroid values. The proposed system is designed to improve the K-means clustering algorithm with efficient centroid estimation models.

The job of association rule mining is to discover the association relationship among a set of items. The mining of association rule include two sub problems.
1 To find all frequent itemsets that appears more often than a minimum support threshold 

2 To construct association rules using these frequent itemsets. 

Various types of association rule mining algorithms were used in many applications in order to identify interesting frequent itemsets. Among the different association rule mining algorithms, one of the association rule mining algorithm such as Apriori algorithm utilized the property of support and confidence value to construct frequent items. 

Further, it is proposed to present an improved Apriori algorithm, to minimize the number of candidate sets while generating association rules by evaluating quantitative information associated with each item that occurs in a transaction, which was usually discarded as traditional association rules focus just on qualitative correlations is proposed. The set of candidate sets found is pruned by a strategy that discards sets which contain infrequent subsets. This work evaluates the scalability of the algorithm by considering transaction time, number of itemsets used in the transaction and memory utilization. 

3.2 TRADITIONAL APRIORI ALGORITHM

Traditional Apriori algorithm can be described as an iterative association rule learning algorithm which computes the frequent itemsets where iteration j calculates all frequent j-itemsets and each iteration has a candidate construction step. Apriori is derived as a classic algorithm for studying association rules. Apriori is designed in such a way to operate on transaction data set containing transactions. As is common in association rule mining, given a group of itemsets, the algorithm maximizes to find subsets which are common to at least a minimum number Candidate of the itemsets.
Traditional Apriori uses a "bottom up" approach, where frequent subsets are extended one item at a time which is called as a candidate generation, and groups of candidates are tested against the data. The algorithm stops the loop when no further successful extensions are found. Traditional Apriori also utilizes breadth-first search and a tree structure to count candidate itemsets efficiently. It generates candidate itemsets of length \( l \) from itemsets of length \( l-1 \). Then it starts pruning the candidates which have an infrequent sub pattern. After that, it scans the transaction database to determine frequent itemsets among the candidates.

In many of the cases, the Apriori algorithm significantly reduces the size of candidate sets using the Apriori principle. Apriori suffers from a number of inefficiencies or trade-offs. They are, retrieving a huge number of candidate sets, repeatedly scanning the database and verifying the candidates by pattern matching. Candidate generation generates large numbers of subsets (the algorithm attempts to load up the candidate set with as many as possible before each scan). Bottom-up subset exploration (essentially a breadth-first traversal of the subset lattice) finds any maximal subset \( S \) only after all \( 2^{\left| S \right| - 1} \) of its proper subsets.

The Apriori algorithm is a dominant algorithm whose main job is to mine frequent itemsets which involves Boolean data type of association rules. The main ideas involved are:

**Frequent Itemsets** : The sets of item that has minimum support value which is denoted by \( F_i \) for \( i^{th} \) itemset.

**Apriori Property** : Any subset of the itemset that are frequent should also be frequent.

**Join Operation** : To find \( F_j \), which consists of a set of candidate \( j \)-itemsets is formed by joining \( F_{j-1} \).
To find the frequent itemsets: it is referred to as the set of items which have minimum support value. This states that a subset of an itemset that is frequent must also be a frequent itemset. For ex., if \{PQ\} is a frequent itemset, then both \{P\} and \{Q\} should also be a frequent itemset. Subsequently, we have to find all frequent itemsets with cardinality from 1 to j (j-itemset)

Generate association rules using the frequent itemsets. The pseudo code for traditional Apriori algorithm is given below:

**Join Step**: Rj is generated by joining Fj-1

**Pruning Step**: Any (j-1) itemset that is not frequent cannot be a subset of a frequent j- itemset.

**Pseudo code**

Rj: Candidate itemset of size j

Fj: Frequent itemset of size j

F1 = \{Frequent items\}

For(j=1;fj!=\mu;j++)

Do

Begin

Rj+1 = candidates generated from Fj

For each transaction T in database

Do

Add the count of candidates

Fj+1 = candidates in Rj+1 with min_sup

End

Return Fj

End
The working principle of pseudo code is explained with the help of an example

- Consider a transaction database (TD) that consists of 9 transactions performed.
- Assume that the minimum support count value is 2.
- Assume that the minimum confidence value required to be 70%.

The first step involved in the traditional Apriori algorithm is to find the frequent itemset using min_sup. The next step is to generate the association rules using min_conf.

<table>
<thead>
<tr>
<th>Tid</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>T100</td>
<td>RI1, RI2, RI5</td>
</tr>
<tr>
<td>T200</td>
<td>RI2, RI4</td>
</tr>
<tr>
<td>T300</td>
<td>RI2, RI3</td>
</tr>
<tr>
<td>T400</td>
<td>RI1, RI2, RI4</td>
</tr>
<tr>
<td>T500</td>
<td>RI1, RI3</td>
</tr>
<tr>
<td>T600</td>
<td>RI2, RI3</td>
</tr>
<tr>
<td>T700</td>
<td>RI1, RI3</td>
</tr>
<tr>
<td>T800</td>
<td>RI1, RI2, RI3, RI5</td>
</tr>
<tr>
<td>T900</td>
<td>RI1, RI2, RI3</td>
</tr>
</tbody>
</table>

**Phase I: Generating 1-itemset (Frequent Pattern)**

Scan the database for the count value

<table>
<thead>
<tr>
<th>Items</th>
<th>Support (Count)</th>
</tr>
</thead>
<tbody>
<tr>
<td>{RI1}</td>
<td>6</td>
</tr>
<tr>
<td>{RI2}</td>
<td>7</td>
</tr>
<tr>
<td>{RI3}</td>
<td>6</td>
</tr>
<tr>
<td>{RI4}</td>
<td>2</td>
</tr>
<tr>
<td>{RI5}</td>
<td>2</td>
</tr>
</tbody>
</table>
R1

<table>
<thead>
<tr>
<th>Items</th>
<th>Support (Count)</th>
</tr>
</thead>
<tbody>
<tr>
<td>{RI1}</td>
<td>6</td>
</tr>
<tr>
<td>{RI2}</td>
<td>7</td>
</tr>
<tr>
<td>{RI3}</td>
<td>6</td>
</tr>
<tr>
<td>{RI4}</td>
<td>2</td>
</tr>
<tr>
<td>{RI5}</td>
<td>2</td>
</tr>
</tbody>
</table>

F1 consists of the frequent 1-itemsets where each item is the member of candidate R1.

**Phase II: Generating 2-itemset (Frequent Pattern)**

Generate R2 candidates from F1.

<table>
<thead>
<tr>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>{RI1, RI2}</td>
</tr>
<tr>
<td>{RI1, RI3}</td>
</tr>
<tr>
<td>{RI1, RI4}</td>
</tr>
<tr>
<td>{RI1, RI5}</td>
</tr>
<tr>
<td>{RI2, RI3}</td>
</tr>
<tr>
<td>{RI2, RI4}</td>
</tr>
<tr>
<td>{RI2, RI5}</td>
</tr>
<tr>
<td>{RI3, RI4}</td>
</tr>
<tr>
<td>{RI3, RI5}</td>
</tr>
<tr>
<td>{RI4, RI5}</td>
</tr>
</tbody>
</table>

Scan the database for count value to determine candidates
In order to identify the frequent 2-itemsets \( F_2 \), the traditional Apriori algorithm uses \( F_1 \), Join operation performed for \( F_1 \) which in turn generates a candidate set of 2-itemsets \( R_2 \). The database \( D \) is scanned and the support count for each candidate itemset in \( R_2 \) is generated. The frequent
2-itemsets F2 is then achieved using min_sup. Based on the above concept any number of frequent itemsets can be generated. Finally association rules can be formed using the frequent itemsets generated.

Since, the number of database accesses of the Apriori algorithm equals to the overall size of the maximal frequent itemset, it takes k-times even when only one k-frequent itemset exists. If the size of the dataset is huge, the multiple database scans can result in the drawbacks of the Apriori algorithm. In the next section, a framework for scalable association rule mining is discussed with.

Prakash and Parvathi (2011) presented an Improved Apriori algorithm that not only reduces the number of itemsets generated but also the overall execution time. Transaction reduction is achieved by discarding the transactions that do not contain any frequent itemset. During the each iteration the Apriori algorithm uses the frequent sets from the previous iteration to generate the candidate sets.

3.2 FRAMEWORK FOR SCALABLE ASSOCIATION RULE MINING

In this section, the main features of proposed parallel pruning Apriori algorithm with the help of the Architecture of scalable association rule mining which overcomes the problems of minimized scan of candidate item, while discovering frequent itemsets in large size database (Giga Byte) is discussed.
The main purpose of data mining is to find meaningful patterns or generate the rules in large size datasets. It is considered as an interdisciplinary field. Data mining combines research from varied fields such as machine learning, statistics, high performance computing, and neural networks. The most common feature of data mining tasks is that they generally operate on enormous voluminous data. The data sources ranges from gigabytes to terabytes. This results in fast data mining algorithms that can prune large size databases within a fraction of time period.
The work proposes a parallel pruning Apriori algorithm that emphasizes in minimizing the number of candidate sets while building association rules by evaluating maximal information associated with each item that occurs in a given set of transaction and considering each item that occurs in a transaction, which was actually, discarded in traditional association rule mining. Figure 3.1 shows the architecture of scalable association rule mining performed using parallel pruning Apriori algorithm. Parallel pruning Apriori-based frequent item mining algorithms are based on the principle that the subset of a frequent itemsets must also be frequent. Frequent itemset mining finds sets of items that occur in a percentage of transactions with the percentage, called support value, larger than a given threshold.

The parallel pruning Apriori algorithm finds all frequent itemsets for bank transaction data set in multiple passes, given a support threshold. At the first pass, it finds the frequent items. In the algorithm, first all 1-item candidates are generated, test their frequencies by scanning the transaction database, then counts supports of these candidates and pruning those candidates whose supports are less than a given support threshold, then join the 1-item candidates to generate 2-item candidates and test the frequency of 2-items candidates. This process will be continued until there’s no frequent candidate left in the new generation. This results in reduced candidate itemset generation compared to that of traditional Apriori algorithm.

3.2.2 Parallel Pruning

In this section, the main features of parallel pruning Apriori algorithm that overcomes the problems of scalability by using parallel pruning while discovering frequent itemsets in large size database is described in brief. The proposed scalable association rule mining reduces not only the
number of itemsets using bank transaction data set [41], generated during pruning stage, but also the overall execution time of the algorithm. Reduction in the number of transaction is achieved by discarding the transactions that do not contain any frequent itemset in subsequent scans which in turn reduces overall execution time. Parallel pruning is done through the implementation of threads (in JAVA) which in turn reduces the execution time.

3.3 PARALLEL PRUNING APRIORI ALGORITHM - PROCESS

Initial research work started with the evaluation of weighted association rule mining in terms of item-value relational metrics. Then the number of item metrics is taken into account of the association rule mining with reduced candidate itemset. This may reduce not only the number of itemsets generated but also the overall execution time of the algorithm. Any valued attribute will be treated as item-value relational metrics and will be used to derive the minimal number of association rules which increased the rules information content.

The research work evaluates the scalability of the Fuzzy Optimal Search Space Pruning (car dataset and bank transaction data set are taken for evaluation) by considering the transaction time, number of itemsets used in the transaction and memory utilization. To evaluate the item-value relativity metric of the scalable association mining, optimal search on parallel pruning is planned for deployment as it can hold more number of associative information.
The parallel pruning Apriori algorithm is demonstrated in Figure 3.2.

Figure 3.2 Structure of Parallel Pruning Apriori

This research work, first analyze the scalability issues of association rule mining in large size data sets. Parallel pruning technique is deployed to mine the large transactional items simultaneously at different levels of items sets to improve the execution speed for generating frequent items and association rules.

The improvement of Apriori algorithm is done by increasing the efficiency of candidate pruning phase by reducing the number of candidates that are generated for further verification. The proposed pruning technique use information associated to the number of items to estimate the overlap items in the transactions. The basic elements considered in the development of the work scalability in association rule mining are number of transactions, average size of transaction, average size of the maximal large itemsets,
number of items, and distribution of occurrences of large itemsets. The algorithm for parallel pruning technique to generate informative rules and strong frequent items is presented as below:

The process of dynamic itemset counting is performed by adding new candidate itemsets only when all of their subsets are proved to be frequent. The frequent item ranges are used as input for generating higher order item ranges using the Apriori algorithm. During the each iteration of the algorithm, frequent sets are used from the previous iteration to generate the candidate sets and check whether their support is above the threshold. The set of candidate sets found is pruned that discards sets which contain infrequent subsets. The performance of the work is evaluated in terms of scalability of the algorithm by considering transaction time, number of itemsets used in the transaction and execution time.

Pratima Gautam and Pardasani (2011) presented a partition technique for the multilevel association rule mining problem. In this work association rules are performed at different levels which help in discovering more specific and applicable knowledge. Thus the author used partition method and Boolean methods for finding frequent itemsets at each concept levels which reduce the number of scans, I/O cost and also reduce CPU overhead.

3.3.1 Phases of Parallel Pruning Apriori Algorithm

The diagrammatic representation of Parallel Pruning Apriori algorithm with the extension provided from traditional Apriori algorithm is shown in Figure 3.3.
As shown in the Figure 3.3 it undergoes three phases which consists of the initialization phase, Join step phase and Pruning step phase. Finally the working of algorithm results in reduced candidate itemsets.

Initialization is done in the first phase. Join step and Pruning step is performed using the brute force method. The brute force method is designed in a way such that it consider all the candidate items and then apply the pruning process in order to identify and remove if there occurs any unnecessary candidate items during this stage.
The number of candidate itemsets \( (C_d) \) generated is derived as

\[
\sum_{d=1}^{k} C_d
\]

The working of parallel pruning Apriori algorithm is explained below. It involves a detailed analysis of three phases using bank data set provided with the number of transactions, minimum support value and the itemsets.

**Input** : Bank data set, Number of Transactions and items, minimum support value (min_sup) and itemset

**Output** : Reduced candidate itemsets, number of informative rules, frequent items, execution time.

**Procedure** : The parallel pruning Apriori algorithm is constructed as follows:

1. **Initialization**
   1.1 Initialize number of items, minimum support (min_sup) and transactions from large size data sets.
   1.2 Scan the Transaction Database (TD) to get support S for each item. Compare S with minimum support (min_sup) and retrieve a set of frequent 1-itemsets. Let it be F1.

2. **Join step**
   2.1 Use Fk-1, join Fk-1 to create a new set of candidate item k itemsets.
   2.2 Reduce the candidate item with relative item values
2.3 With probability ratio, generate frequent itemsets (i.e., satisfy minimum support)

3. **Pruning step**

3.1 Parallel prune the frequent items at different levels of the itemset

3.2 Any \((k - 1)\) size itemset which is not frequent cannot be a subset of a frequent \(k\) size itemset, hence those itemsets should be removed. Eliminate candidates that are infrequent, leaving only those that are frequent

3.3 With conditional probability on parallel pruned item levels, generate strong association rules.

3.4 Calculate execution time of frequent itemset and informative association rules

3.5 Sort the itemsets based on the frequency and information association

3.6 Iterate the steps 2.2 to 3.3 till the required scalability mining results are achieved.

With Bank data set transaction extracted from the Machine Learning Data repository which has been considered as input, proposed work of scalable association rule mining using parallel pruning adapt three phases as explained below. Scalable association rule mining using parallel pruning minimizes the number of candidate sets with efficient pruning time and search space optimization.

The first phase consists of initialization which performs the job of initializing number of items, minimum support \((\text{min}\_\text{sup})\) and transactions for
bank data set. Here scanning of transaction database is performed in order to obtain support $S$ for each item. Comparison is performed between support value $S$ and minimum support value (min_sup) to retrieve the first set of frequent items.

The second phase consists of joining the itemset to create a new set of candidate item using relative item values. By using the user provided minimum support value and with the help of probability ratio, frequent items are generated.

The third phase consists of pruning. The parallel pruning in this work provides improvement over traditional Apriori algorithm which executes faster while generation of frequent items and rules extracted for bank transaction data. It constructs all candidates based on the principle of n-level frequent itemsets on sorted database, and all frequent itemsets generated which can no longer be supported by transactions that still have to be processed. Proposed parallel pruning Apriori algorithm need not have to maintain the covers of all past itemsets.

In scalable association rule mining, the candidate item reduction object is updated in each of the iteration which helps in determining the processing items. In the traditional Apriori algorithm, the data item read, have to be compared against all candidates to determine the set of candidates whose total number of counts will be incremented.

3.4 EXPERIMENTAL RESULTS AND DISCUSSIONS

In order to evaluate the proposed methodology using parallel pruning Apriori algorithm and to measure the efficiency and scalability of Parallel Pruning Apriori algorithm, it is essential to perform an extensive performance study. In this section, comparison is performed on the
experimental results conducted on Apriori algorithm and parallel pruning Apriori algorithm. The experiment focused on evaluating scalability using parallel pruning Apriori techniques. Taking into consideration the best performance, parallel pruning Apriori algorithm is deployed using banking data set samples. 1 Giga Byte (GB) dataset is taken for experiment. It is observed that some of the valid results derived from the bank data set. To show the enhancement in transaction time, memory utilization, several experiments have conducted using bank data set.

The total number of distinct item is 1000 and the average number of items in a transaction is taken as 15. A confidence of 90% and support of 5% is used. Execution times using 1, 2, 3, and 4 threads are presented on the processor. With 1 thread, traditional Apriori does not have any significant overheads as compared to the parallel pruning Apriori algorithm. Therefore, this model is used for reporting all speedups.

The second experiment was conducted using a dataset with 20 distinct items, where the average number of items per transaction is 6. The total size of the dataset is 500 MB and a confidence level of 90% is used. Four support levels, 10%, 5%, 3%, and 2% are considered. The execution time efficiency is improved for the parallel pruning Apriori on frequent itemsets evaluation with the support count. This is shown in Figure 3.4. Using parallel pruning Apriori algorithm frequent itemsets are generated from bank data set using the methodology is explained in 3.3.1.

The experiment conducted using new parallel pruning Apriori algorithm provides more interesting rules than the previous one using the existing traditional Apriori algorithm.
The thread execution on the Traditional Apriori and Parallel Pruning Apriori are evaluated for the same data set. Here the initial thread requires more time, however consequent threads shows better scalable performance of Parallel Pruning Apriori.

Figure 3.5 illustrates the execution time of both Apriori and parallel pruning algorithm with respect to the number of transaction required for processing the bank data set. In Figure 3.5, X axis denotes the number of transaction and the execution time denoted in terms of nano seconds in the Y axis. Using parallel pruning Apriori algorithm as the number of transaction increases the time taken to execute will be in decreasing phase when compared to the traditional Apriori algorithm. From the Figure 3.5, it is clear that parallel pruning Apriori shows better results at mining very large transaction data sets when compared to the traditional Apriori algorithm.
Figure 3.5 Execution Time for Apriori and Parallel Pruning Apriori

Figure 3.6 shows the scalability of both Apriori and parallel pruning Apriori with respect to the transactions in bank data set. X axis shows the number of items i.e., 200, 400, 600, 800, 1000 respectively whereas the Y axis refers to the execution time. Various support threshold values were applied in both the algorithms. As the number of items increases the time taken to execute parallel pruning algorithm is decreasing in contrast with Apriori algorithm. From the Figure 3.6, it is identified that parallel pruning algorithm is efficient and scalable in very large size databases.
3.5 SUMMARY

Here, a method of Parallel Pruning Apriori Algorithm is presented. The performance of association mining algorithms is discussed. Some of the algorithms discussed are traditional Apriori algorithm, parallel pruning Apriori algorithm using bank data set. Software is implemented for the proposed algorithm and its framework and algorithm are explained in detail using diagrammatical representation. Several experiments are conducted to assess the performance of the algorithm.

The proposed parallel pruning Apriori algorithm provides improvement over traditional Apriori by considering faster execution for generating frequent items and association rules for deriving the result from transaction data. Parallel execution of item pruning reduced memory faintness and increases the execution speed. It constructs all candidates based on n-level frequent itemsets on sorted database, and all frequent itemsets that can no longer be supported by transactions that still have to be processed.
However as the pruning transaction item is more concerned in parallel, the search space for frequent item generation and item-value pair based maximal information sensitive association rules becomes complex. To overcome these facts, in the next chapter, the optimization of search space using fuzzy rule set, is described.