

Web Log Mining using Improved Version of Apriori Algorithm

Suneetha K R

Bharatidasan Institute of Technology
Anna University

Krishnamoorti R

Bharatidasan Institute of Technology
Anna University

ABSTRACT

Association Rule mining is one of the important and most popular data mining technique. It extracts interesting correlations, frequent patterns and associations among sets of items in the transaction databases or other data repositories. Most of the existing algorithms require multiple passes over the database for discovering frequent patterns resulting in a large number of disk reads and placing a huge burden on the input/output subsystem. In order to reduce repetitive disk read, a novel method of top down approach is proposed in this paper. The improved version of Apriori Algorithm greatly reduces the data base scans and avoids generation of unnecessary patterns which reduces data base scan, time and space consumption.

Keywords

Data mining, Association rule, Apriori algorithm, Frequent pattern.

1. INTRODUCTION

Association rule mining has been well studied in data mining, especially for basket transaction data analysis. Association rules also used in various areas such as telecommunication networks, market, risk management and inventory control etc. Aside from being applicable for e-commerce, business intelligence and marketing applications, it helps web designers to restructure their web site. The frequent pattern mining module gives the details of association rule mining technique. Association rules shows, attributes value conditions that occur frequently together in a given data set that provides information in the form of “if-then” statements. Literature survey reveals that identifying frequent item sets is computationally expensive process. Counting the occurrences of item sets requires a considerable amount of processing time. As a consequence, numbers of efficient algorithms are proposed. It is noticed that, most of the algorithms for discovering frequent patterns requires multiple passes over the database resulting in a large number of disk reads and placing a huge burden on the I/O subsystem. Apriori utilizes a complete bottom up search with a horizontal layout and enumerate all frequent item sets. The proposed improved version of Apriori algorithm utilizes top down approach, where the rules are generated by avoiding generation of un-necessary patterns. The major advantage of this approach is, the number of database scans is greatly reduced. Working of existing and proposed Apriori algorithm to generate Association rule is discussed in the following section.

The structure of the paper is as follows: Section 2 covers relative work. Section 3 presents processing principle of existing and proposed algorithm. Section 4 shows experimental results and Section 5 concludes the paper.

2. RELATED WORK

One of the most well known and popular data mining techniques is the Association rules or frequent item sets mining algorithm. The algorithm was originally proposed by Agrawal et al. [1] [2] for market basket analysis. Because of its

significant applicability, many revised algorithms have been introduced since then, and Association rule mining is still a widely researched area. Many variations done on the frequent pattern mining algorithm of Apriori is discussed in this section.

Association rule generation is used to relate pages that are most often referenced together in a single server sessions [13]. In the context of web usage mining, association rules refer to sets of pages that are accessed together with a support value exceeding some specified threshold.

Agrawal et. al. presented an AIS algorithm in [1] which generates candidate item sets on-the-fly during each pass of the database scan. Large item sets from previous pass are checked if they are present in the current transaction. Thus new item sets are formed by extending existing item sets. This algorithm turns out to be ineffective because it generates too many candidate item sets. It requires more space and at the same time this algorithm requires too many passes over the whole database and also it generates rules with one consequent item.

Agrawal et. al. [2] developed various versions of Apriori algorithm such as Apriori, AprioriTid, and AprioriHybrid. Apriori and AprioriTid generate item sets using the large item sets found in the previous pass, without considering the transactions. AprioriTid improves Apriori by using the database at the first pass. Counting in subsequent passes is done using encodings created in the first pass, which is much smaller than the database. This leads to a dramatic performance improvement of three times faster than AIS. A further improvement, called AprioriHybrid, is achieved when Apriori is used in the initial passes and switches to AprioriTid in the later passes if the candidate k-itemset is expected to fit into the main memory.

Even though different versions of Apriori are available, the problem with Apriori is that it generates too many 2-item sets that are not frequent. A Direct Hashing and Pruning (DHP) algorithm is developed in [8] that reduces the size of candidate set by filtering any k-item set out of the hash table, if the hash entry does not have minimum support. This powerful filtering capability allows DHP to complete execution when Apriori is still at its second pass and hence shows improvement in execution time and utilization of space.

Scalability is another important area of data mining because of its huge size. Hence, algorithms must be able to “scale up” to handle large amount of data. Eui-Hong et. al [4] tried to make data distribution and candidate distribution scalable by Intelligent Data Distribution (IDD) algorithm and Hybrid Distribution (HD) algorithm respectively. IDD addresses the issues of communication overhead and redundant computation by using aggregate memory to partition candidates and move data efficiently. HD improves over IDD by dynamically partitioning the candidate set to maintain good load balance.

Another scalability study of data mining is reported by introducing a light-weight data structure called Segment

Support Map (SSM) that reduces the number of candidate item sets needed for counting [11]. SSM contains the support count for the 1-item set. The individual support counts are added together as the upper bound for k-item sets. Applying this to Apriori, the effort to generate 1-item set is saved by simply inspecting those SSM support counts that exceed the support threshold. Furthermore, those 1-item sets that do not meet the threshold will be discarded to reduce the number of higher level item sets to be counted.

Evolutionary Algorithms (EA) are widely adopted in many scientific areas. EA borrows mechanisms of biological evolution and applies them in problem-solving, especially suitable for searching and optimization problems. Hence, the problem of mining with Association rules is a natural fit. Besides Association rule mining Evolutionary algorithms are also reported that can generate association rules [12]. It allows overlapping intervals in different item sets.

The quality of the association rule discovered is measured in terms of confidence. The rules with confidence above a certain level (threshold value) are considered as interesting and deserve attention. Most algorithms define interestingness in terms of user-supply thresholds for support and confidence. The problem is that these algorithms rely on the users to set suitable values. Another algorithm called APACS2 is proposed in [10], that makes use of an objective interestingness measure called adjusted difference. It also discovers both positive and negative association rules. APACS2 uses adjusted difference as an objective interestingness measure. Adjusted difference is defined in terms of standardized difference and maximum likelihood estimate.

A survey on different methods and algorithms used to find frequent patterns is presented in [14]. Analysis of algorithms and descriptions for AprioriTid, AprioriHybrid, Continuous Association Rule Mining Algorithm (CARMA), Eclat algorithm, and Direct hashing and Pruning (DHP) algorithm is explained in detail. Conclusions are drawn as, for dense databases Éclat algorithm is better, for sparse databases the Hybrid algorithm is the best choice and as long as the database fits in main memory the Hybrid algorithm (combination of optimized version of Apriori and Eclat) is most efficient one.

An improved version of original Apriori- All algorithm is developed for sequence mining in [15]. It adds the property of the userID during every step of producing the candidate set and every step of scanning the database to decide about whether an item in the candidate set should be used to produce next candidate set. The algorithm reduces the size of candidate set in order to reduce the number of database scanning.

Based on the temporal association rule [3] [5], retailers make better promotion strategies. The time dimension exists in all transaction, and is included in finding large item sets, especially when not all items exist throughout the entire data gathering period. The temporal concept introduced in [9] addition to the normal support and confidence. The temporal support is the minimum interval width. Thus, a rule is considered as long as there is enough support or temporal support.

Different works are reported in the literature to modify the Apriori logic so as to improve the efficiency of generating rules. Enhanced version of Apriori algorithm is presented in [16] where, the efficiency is improved by scanning the database in forward and backward directions. Xiang-wei Liu et.al [17] presented an improved association rule mining algorithm that reduces scanning time of candidate sets using hash tree. Another version of Apriori is reported in [18] as an algorithm

called IApriori algorithm, which optimizes the join procedure of frequent item sets generated to reduce the size of the candidate item sets. The algorithm presented in [19] scans the database only once to generate a frequent item sets, thereby saving time and increasing efficiency. These methods even though focused on reducing time and space, in real time still needs improvement.

Another way to improve Apriori is to use most suitable data structure such as frequent pattern tree. Han et. al., in [7] introduced an algorithm known as FP-Tree algorithm for frequent pattern mining. It is another milestone in the development of association rule mining and avoids the candidate generation process with less passes over the database. FP-Tree algorithm breaks the bottlenecks of Apriori series algorithms but suffers with limitations. It is difficult to use in an environment that users may change the support threshold with regard to the mining results, and once the support threshold changed, the old FP-Tree cannot be used anymore, hence additional effort is needed to re-construct the corresponding FP-Tree. It is not suitable for incremental mining, since as time goes on databases keep changing, new datasets may be inserted into the database or old datasets be deleted, and hence these changes lead to a re-construction of the FP-Tree[6].

Even though fast algorithms are reported for Association mining it still inherits the drawback of scanning the whole data base many times. The survey reveals that more attention is required to address the issues related to reduce the number of database scan, and also to reduce memory space with less execution speed. This results in a large number of disk reads and placing a huge burden on the I/O subsystem. These limitations and other related issues motivated us to continue the research work in this area. Comparing all these methods, in this work we propose a new improved version of Apriori algorithm which reduces time and space and the same is presented in the next section.

3. FREQUENT ITEM SET AND ASSOCIATION RULE

The aim of Association rule mining is exploring relations and important rules in large datasets. A dataset is considered as a sequence of entries consisting of attribute values also known as items. A set of such item sets is called an item set. Frequent item sets are sets of pages which are visited frequently together in a single server session. Only the list of session IDs and URLs is used during this process. Support is often utilized to limit the number of discovered patterns. Support of the subset $\{i_1 \dots i_n\}$ from a set D is defined as in equation (1)

$$S(i_1, i_n) = \text{count}(\{i_1, i_n\} \in D) / \text{Count}(D) \text{---- (1)}$$

Once the frequent item sets are discovered, we calculate for each item set the interest to objectively rank them. Interest is defined as in equation (2)

$$I(i_1, \dots, i_n) = S(i_1, \dots, i_n) / \prod_{j=1}^n S(i_j) \text{----- (2)}$$

Set of n frequent items are broken into n separate Association rules. The confidence of an association rule (as in equation (3)) is the fraction of sessions where the subsequent and the antecedent are present and sessions where only the subsequent is present.

For the rule $i_a \rightarrow i_{s1} \dots i_{sn}$ it is

$$C(i_a \rightarrow i_{s1}, \dots, i_{sn}) = S(i_a \rightarrow i_{s1}, \dots, i_{sn}) / S(i_a) \text{--- (3)}$$

The applications of frequent item sets and association rules are: business intelligence (e.g. cross promotional opportunities), web site restructuring, and documents pre-fetching. Association rules are of interest to both database community and data mining users. The support of an item is the percentage of transactions in which that item occurs. Confidence measures strength of the rule, where as support measures how often it should occur in the database. Typically, large confidence values and a smaller support are used.

The Apriori algorithm is used for mining frequent item sets. The algorithm to discover Association rules generally broken down into two steps:

1. Find all large item sets - A large item set is a set of items that exceeds the minimum support.
2. Generate rules from the large item sets.

Association rules are considered interesting if they satisfy both a minimum support threshold and minimum confidence threshold.

3.1 Proposed Work

The classical Apriori algorithm follows bottom up approach. The proposed algorithm uses top down approach, where in the rules are generated by avoiding generation of un-necessary patterns. Figure. 1. Illustrates step wise working of the proposed approach. The major advantage of this approach is that, the number of database scans is greatly reduced, since the number of combinations made is far less compared to the original Apriori algorithm and thus results in reduction of time and space.

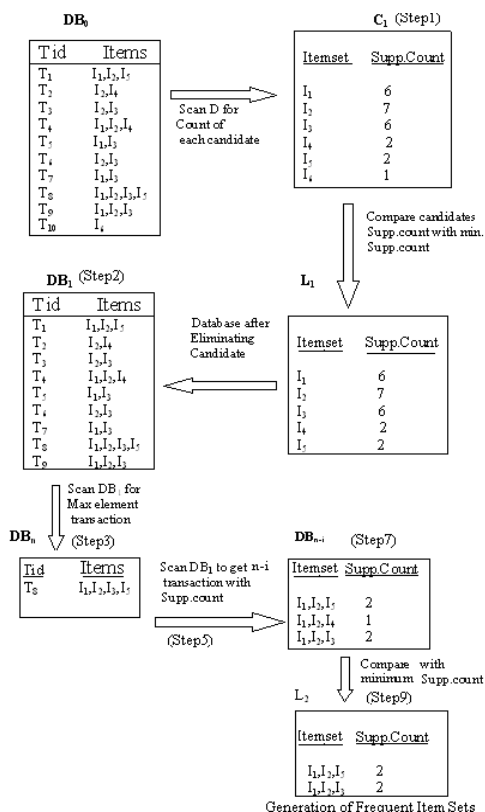


Fig 1: Working steps of new approach

The algorithmic steps written in form of Pseudo code is presented below.

3.1.1 Pseudo Code

```

Input: database (D), minimum support (min_sup).
Output: frequent item sets in D.
L1= frequent item set (D)
j=k; /* k is the maximum number of elements
in a transaction from the database*/
for k= maxlength to 1
{
for i=k to 2
{
for each transaction Ti of order i
{
if (Ti has repeated)
{ Ti.count++; }
m=0;
while (i<j-m)
{if ( Ti is a subset of each transaction Tj-m of
order j-m)
{ Ti.count++; m++; }
}
If (Ti.count >=min_sup)
{ Rule Ti generated
/*store the transaction in Rule Table*/}
}
}
}

```

3.1.2 Stepwise Descriptions of Pseudo code

Step1: Scan the Database to count number of occurrences of each item. In the first iteration of the algorithm each item is a member of the set of candidate-1 item set C₁.

Step 2: Compare candidate support count with minimum support count. The set of frequent-1 item set L₁ is determined, which consists of the candidate-1 item sets satisfying minimum support. This results in database DB₁.

Step3: Scan the Database DB₁ for maximum element (k) transactions. The resulting database is termed DB_n. The probability of forming the rule with maximum element transaction is very less because it is unique.

Step4: Initialize i=j=k.

Step5: Scan the database to find i element transaction say Ti.

Step6: If the transaction has repeated then increment the respective counter by 1.

Step7: Check whether the Transaction Ti is a subset of each Transaction of bigger element Transaction set Ti+1 (Not applicable for k element transaction because it itself is a bigger set).

Step8: If the Transaction Ti is a subset then increment the respective counter by 1.

Step 9: If the counts of the respective transaction are greater than the Minimum Support Count then Rule Ti is generated. (Add Ti to the Rule Table).

Step10: Repeat step 5 for all transactions of the order i (i element transactions).

Step 11: Decrement k by 1.

Step 12: Repeat step 4 until the value of k reduces to 1.

The algorithm is applied on the log file of www.enggresources.com obtained for the month of August 10. Comparison chart and list of Association rules is presented further.

3.2 Comparison of the performance

The theoretical estimation of the performance is presented in this section. The total run time for original Apriori algorithm is defined as in equation (4)

$$\sum_{i=1}^n (t_s * m_k + t_c + l_{k+1} * k + 1 / 2 * t_s * n_k / A) \text{-----(4)}$$

where, t_s be the time cost of a single scan of the database, t_c be the time cost of generating C_{k+1} from L_k , m_k is set to be the amount of item sets in C_k without n_k , the variable l_{k+1} is set to be the amount of item sets in C_{k+1} and the variable n_k is set to be the amount of item sets in L_k . A denotes the amount of the records in the database and n denotes the dimension of the data.

The total run time for proposed Improved version of Apriori algorithm is

- Time required to scan the database for first time is: $t_s * m_k$ where t_s : time for scanning data base and m_k : number of entries in data base
- Time required to scan for items considering support count: $t_k * n_k$ where n_k be the number of items and t_k is time to scan items comparing with minimum support
- Time taken to search for items of maximum length : $t_1 * n_1$ where t_1 : reduced database scan time
- Time taken to generate combinations : $\sum_{i=2}^p t_i * n_i$

Where p length of highest pattern, t_i is time to scan combination of length i and n_i is number of combinations of length i .

Total time taken for the proposed method is given as follows as in equation (5)

$$(t_s * m_k) + (t_k * n_k) + (t_1 * n_1) + \sum_{i=2}^p t_i * n_i \text{-----(5)}$$

From the above equations (4) and (5), the difference in the runtime between the two algorithms depends on N , the dimension of the data. And also compared with the classical Apriori algorithm, the number of scan reduces to 50% and the run time reduces nearly to 60% of the original Apriori.

4. EXPERIMENTAL RESULTS

In this work raw web log data is collected from the educational web site www.enggresources.com of month August 2010 (41MB). This web site focuses on engineering education and provides information related to engineering subjects, syllabus, courses, teaching guide lines, question banks, etc. Various experiments are conducted with the proposed framework to predict user behavior. This satisfies the demanding requirements of today’s applications such as web personalization, site modifications and business intelligence for the success of e-commerce. The data preprocessing and pattern extraction module worked on AMD ATHLON 4000+ processor with 256 MB of physical RAM and 98 MB of free disk space using language JAVA SDK 6.0 Windows 7 operating system.

The result shown in Figure. 2 indicates that improved version of Apriori algorithm is more efficient which takes less time, less memory and hence results in high efficiency. The experimental results shows improvement in generation of candidate sets, results in reduced number of data base scan, and also the time and space consumption.

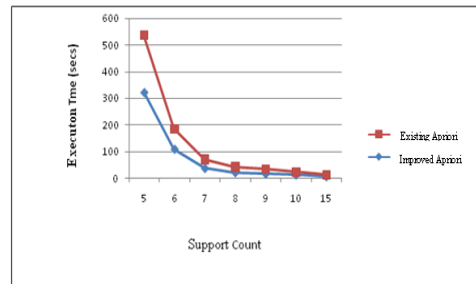


Fig 2: Execution time variation

It is evident from the graph that proposed improved version of Apriori algorithm takes less time and space to generate rules. Most of the algorithms require multiple passes over the database for discovering frequent patterns resulting in a large number of disk reads and placing a huge burden on the I/O subsystem. The major advantage of this approach is that, the number of database scans is greatly reduced by use of top down approach. Performance comparison between existing Apriori and the improved version is discussed in this section. To assess the performance of our mining algorithm, we use datasets from www.enggresources.com. To keep the comparison fair, we implemented algorithms using the same basic data structure.

We use test data set that includes 100 K transactions. Time(s)-axis refers to the running time in seconds, M-axis to the result size constraint, and MinSup-axis to the MinSup constraint. Tests performed using existing Apriori are shown in Figure with some values of MinSup constraint. The Figure. 3 shows that when MinSup constraint becomes low, Apriori tends to run into its exponential bottleneck. The Figure. 4 shows the comparison of and in a larger scale by removing unnecessary patterns. Apriori test results and from there we can see that improved version performs better than existing Apriori.

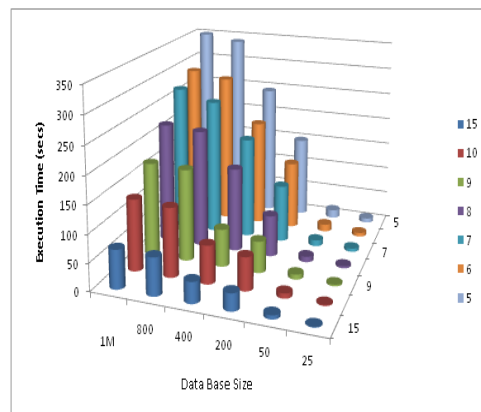


Fig 3: Existing Apriori

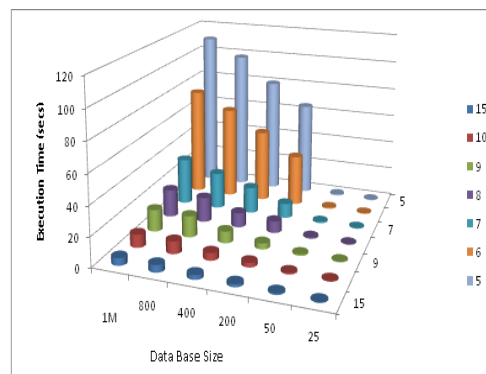


Fig 4: Improved version of Apriori

By tracing these interested usage patterns, the system guesses, with higher probability, a user's "next step". These patterns are analyzed to determine whether a web site is being used as it is intended.

5. CONCLUSIONS

In this paper, the improved version of Apriori algorithm is proposed to overcome the deficiency of the basic Apriori algorithm. The basic Apriori algorithm follows bottom up approach which suffers from increased number of data base scan. The new proposed method follows top down approach which reduces the number of database scans. The improved version Apriori algorithm is more efficient which takes less time, less memory and hence reflects in high efficiency.

6. REFERENCES

- [1] Agrawal, R., Imielinski, T., and Swami, A. N. Mining Association Rules Between Sets of Items in Large Databases. Proceedings of the ACM SIGMOD, International Conference on Management of Data, pp.207-216, 1993.
- [2] Agrawal. R., and Srikant. R., Fast Algorithms for Mining Association Rules, Proceedings of 20th International Conference of Very Large Data Bases. pp.487-499,1994.
- [3] Agrawal. R., and Srikant. R. Mining Sequential Patterns. Proceedings of 11th International Conference on Data Engineering, IEEE Computer Society Press, pp.3-14, 1995.
- [4] Eui-Hong Han, George Karypis, and Kumar, V. Scalable Parallel Data Mining for Association Rules. IEEE Transaction on Knowledge and Data Engineering, 12(3), pp.728-737, 2000.
- [5] Han, J., Dong, G., and Yin, Y. Efficient Mining of Partial Periodic Patterns in Time Series Database. Proceedings of 15th IEEE International Conference on Data Engineering, pp.106–115, 1999.
- [6] Han, J., Jian, Pei., and Yiwen, Yin. Mining Frequent Patterns without Candidate Generation. Proceedings of ACM International conference on Management of Data, 29(2), pp.1-12, 2000.
- [7] Han, J., Jian, Pei., Yiwen, Yin, and Runying, Mao. Mining Frequent Pattern without Candidate Generation: A Frequent-Pattern Tree Approach. Journal of Data Mining and Knowledge Discovery, 8, pp.53-87, 2004.
- [8] Jong Park, S., Ming-Syan, Chen, and Yu, P. S. Using a Hash-Based Method with transaction Trimming for Mining Association Rules. IEEE Transactions on Knowledge and Data Engineering, 9(5), pp.813-825,1997.
- [9] Juan, M. A., Gustavo, H., and Rossi. An Approach to Discovering Temporal Association Rules. Proceedings of the ACM Symposium on Applied Computing, 1, pp.234-239,2000.
- [10] Keith, Chan., and Wai-Ho, A. An Effective Algorithm for Mining Interesting Quantitative Association Rules. Proceedings of the ACM Symposium on Applied Computing, pp. 88-90,1997.
- [11] Lakshmanan, V., S., Carson Kai-Sang, L., and T. Raymond. The Segment Support Map: Scalable Mining of Frequent Itemsets. Journal of ACM SIGKDD Explorations Newsletter, 2(2), pp.21-27, 2000.
- [12] Mata, J., Alvarez, J. L., and Riquelme, J. C. Evolutionary Computing and Optimization: An Evolutionary Algorithm to Discover Numeric Association Rules. Proceedings of ACM Symposium on applied Computing, pp. 590-594, 2002.
- [13] Srivastava, J., Cooley, R., Deshpande, M., and Tan, P. N. Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data. Journal of ACM Special Interest Group on Knowledge Discovery and Data Mining Explorations, 1(2), pp.12-23, 2000.
- [14] Velu, C. M., Ramakrishnan, M., Somu, V., and Logzathan, V. Efficient Association Rules for Data Mining. International Journal of Soft Computing , 2, pp.21-36, 2007.
- [15] Wang Tong, and He Pi-Lian. Web Log Mining by Improved Apriori All Algorithm. Transaction on Engineering Computing and Technology, 4, pp.97-100, 2005.
- [16] Wei Zhang, Zhang Wei, Dongme Sun Shaohua Teng and Haibin Zhu. An Algorithm to Improve Effectiveness of Apriori. Proceedings of 6th IEEE International Conference on Cognitive Informatics, pp.385-390, 2007.
- [17] Xiang-Wei Liu, and Pi-Lian He. The Research of Improved Association Rules Mining Apriori Algorithm. Proceedings of 3rd International Conference on Machine Learning and Cybernetics, pp.1577-1579, 2004.
- [18] Yiwu Xie, Yutong Li, Chunli Wang, and Mingyu Lu. The Optimization and Improvement of the Apriori Algorithm. Proceedings of IEEE International Symposium on Intelligent Information Technology Application Workshops, pp. 1101-1103, 2008.
- [19] Zhao Hong, Gang yang, Lei Wang, and Ying liu. An Implementation of Improved Apriori Algorithm. Proceedings of 8th IEEE International Conference on Machine Learning and Cybernetics, pp.1565-1569, 2009.