# An advance Algorithm for Association Rules Mining: Comparative Study

**Girja Shankar** [1]
*Master of Engineering student*
*Indore Institute of Science and Technology*
*INDORE (M.P.) INDIA*

**Lalita Bargadiya** [2]
*Assistant Professor*
*Indore Institute of Science and Technology*
*INDORE (M.P.) INDIA*

## Abstract

*Data mining is the process of finding hidden, previously unknown and valid and useful information from large database. An association rule mining is the data mining task which is used for finding the important relationship among the item-sets. In this paper, we describe the classical Apriori algorithm and defects of this algorithm which is used for association rule mining. And then we describe the advance algorithm over the classical algorithm and at the end of this paper we will show the experimental results which proved our improvement over Apriori algorithm.*

**Key words:** Data Mining, KDD, Association Rule Mining, GPS, LPS

## 1. Introduction

In current days the use of computer system in markets and todays the daily life product will come with bar-codes, so more and more data are stored in database. It is important to find useful information from this large database. It is not possible to find this useful information with the help of the traditional methods. Hence the efficient methods are required to find information. This problem will be eliminated by the use of data mining techniques. Data mining is formally known as knowledge discovery in database (KDD) [3].

Association rule mining is one of the most important application fields of the data mining tasks. Association rule is used to find out the dependency of among multiple domains based on the given degree of support and confidence.

Apriori algorithm is the classical approach for finding the frequent item-sets which is use for association rule mining. It represents the candidate generation approach. It generates candidate (K+1) item-sets based on frequent k- item-sets.it is level vice search algorithm.

## 2. Basic conception

The concept of the association rules mining was proposed by Agrawal and Srikant in 1994. It is used to predict the customer's buying pattern in the supermarket. Association Rule Mining can be formally defined as:

Defination1: Let I={i1,i2…..in} be the finite item-sets. D is a transaction database, where $i_k$(k=1,2,3…..n) is an item. Tid is denoting the exclusive identifier of transaction T in transaction database.

Definition 2: The implication of the form X => Y is called association rules. Where $X \subset I$, $Y \subset I$, and $X \cap Y \neq \emptyset$.

Definition 3: let D is a transitional database. If the percentage of transactions in D that contain X U Y is s% that rule X => Y holds in D with Support s. If the percentage of transactions in D containing X that also contain Y is c%, the rule X => Y has Confidence c. the definitions of probability are:

Support (X => Y) = P (X U Y)
Confidence (X => Y) = P (Y | X)

Rules that satisfy both minimum support and confidence threshold is call strong rules.

Definition 4: If the support of item-sets X is greater than or equal to minimum support threshold then X is called frequent item-sets, if the support of item-sets X is smaller than the minimum support is called infrequent item-sets.

## 3. The basic idea of Apriori algorithm

Apriori algorithm [4] is a level wise algorithm and it is based on the anti-monotonic property of set theory which states that every subset of a frequent item-set is also frequent, which is to say that an item-set is frequent, all possible subsets of the same are also deemed to be frequent. Apriori is a candidate generation algorithm and proceeds in a level-wise fashion.

Apriori algorithm is two step procedures:

i) Candidate Generation.
ii) Pruning.

A candidate item-set is basically an item-set that could either be frequent or infrequent with respect to the user minimum

support threshold. Higher level candidate item-sets ( $C_i$ ) are generated by joining previous level frequent item-sets or Li-1 with itself. For this aspect that Apriori algorithm is treated as a level-wise algorithm.

Second step of Apriori algorithm helps in filtering out candidate item-sets whose subsets are not frequent. This is based on the anti-monotonic property as a result of which every subset of a frequent item-set is also frequent. Thus, a candidate item-set which is composed of one or more infrequent item-sets of a priori level is pruned from the process of frequent item-set and association mining.

### Apriori algorithm

**Input:** transactional database D and minimum support threshold min_sup.
**Output:** L, frequent item-sets in D.
**Method:**

1. $L_1$ = Frequent items of length 1.
2. For (k=1;$L_k$!=K++) do,
3. Ck+1 = Candidates generated from $L_k$.
4. For each transaction t in database do,
5. Increment the count of all candidates in $C_{k+1}$ that are contained in t.
6. $L_{k+1}$= Candidates in $C_{k+1}$ with minimum support.
7. End do.
8. Return the $L_k$ as the set of all possible frequent item-sets.

### Limitations of Apriori algorithm

Apriori algorithm, despite its simple logic and inherent pruning advantage, suffers from limitations of a huge number of repeated scans of entire transaction database. Since it is a level wise algorithm hence it requires separate scans of the database and over the entire frequent item-set mining process, this become tedious and is a serious limitation.

Another limitation of the Apriori algorithm is the generation of candidate sets which can become cumbersome and time consuming when the number of frequent 1 item-set is large.

## 4. Improvement of Apriori Algorithm

In this paper we describe the improvement of classical Apriori algorithm in the following two aspects:

a) Reducing the passes of database scan.
b) Reducing the unnecessary candidate generation

In our approach, require only two scan of the database. In the first scan we find the frequent one item-set list $L_1$, according to the user min_sup threshold value. Then generate the all combination of the items available in $L_1$ named this as Global_Power_Set (GPS), initialize with item-set_count =0.

At the second scan read the transaction database one-by-one. For each transaction following operation will be done:

First we compare these items with the $L_1$ and remove item from current transaction which is not present in $L_1$. And then generate all possible combination with remaining item, name this as local power set (LPS). Second compare the LPS with GPS if the match found then increment the item-set_count of that item-set by 1 of the GPS.

When the second scan is completed the GPS will store the all candidate item-set with it item-set_count. Pruning will be applied into the GPS with min_sup threshold value. Finally after pruning GPS will hold the all frequent item-sets.

## 5. Advance Algorithm

Input: transaction database D and user minimum support min_sup threshold.
Output: L, frequent item-sets.

Methods:

1) $L_1$= frequent item-set of length 1,
2) Generate power set of $L_1$ and named as GPS initialize with item-set_count=0, it will global for entire algorithm.
3) For each transaction t in database Do.
   a) For each item I in t Do,
      Compare I with $L_1$
      If (not match) then delete item from transaction t.
      End Do.
   b) Generate power set of t and named as LPS
   c) Compare item-sets of GPS with LPS
   d) If (item-set match) increase the item-set_count by 1 of GPS.
4) End Do.

Pruning phase:

5) For each item-set I in GPS Do,
6) If item-set_count of I is less than to min_sup threshold the delete I
7) End Do,
8) Remaining item-set in GPS will be frequent item-sets which holds min_sup threshold.

## 6. Experimental Result

We compared the performance of our advance algorithm with the classical Apriori Algorithm. The experimental platform is Intel Pentium® CPU B950 2.10GHz, 2GB RAM Windows 7 Operating System. The algorithm adopts C# program and compiles in Visual Studio 2008 experiment environment.

Our experiment uses a supermarket transactional database which consists of 10000 TDs and 10 items.

Experiment 1: Number of Items : 10
Number of TD : 10000.

From fig.1, we can see that the execution time of Advance algorithm is shorter than Apriori algorithm while under different support degree.

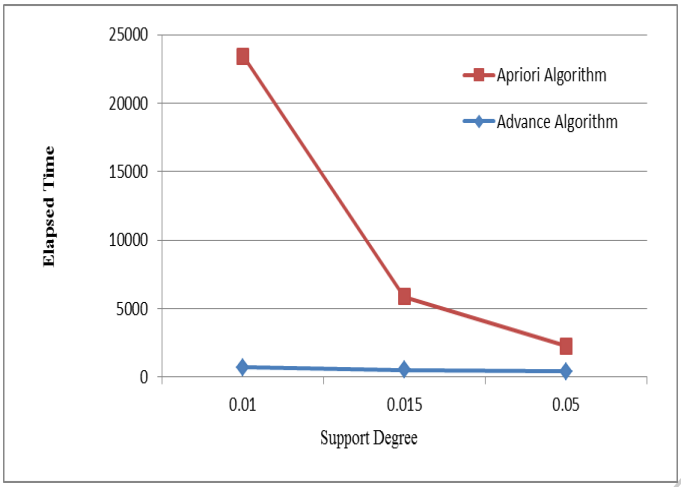


Figure 1 Experiment Contrast with different Support degree

Experiment 2: Support Degree: 0.01
Number of items: 10

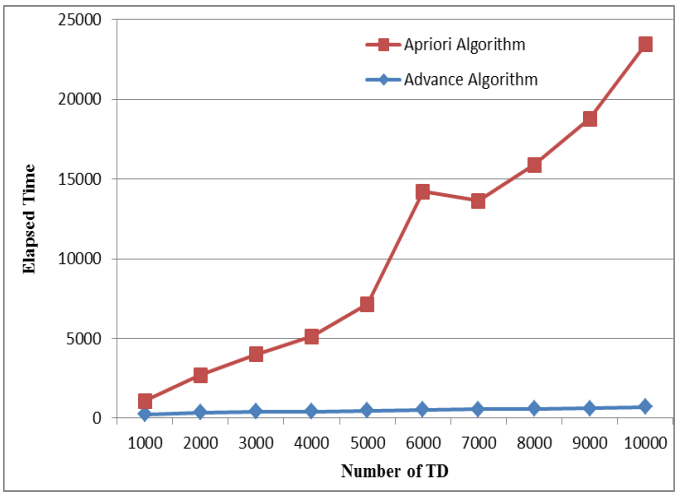


Figure 2 Contrast with different Number of TD

In fig.2, we can see that the execution time of Advance algorithm is shorter than Apriori algorithm while under different number of TD.

Experiment 3: Number of TD: 1000 to 10000
Support degree: 0.01

Table 1 Complexity comparisons of Algorithms with support degree 0.01%

| Number of TD. | Apriori Algorithm (Time complicity in millisecond) | Advance Algorithm (Time complicity in millisecond) |
|---|---|---|
| 1000 | 850 | 230 |
| 2000 | 2370 | 340 |
| 3000 | 3610 | 400 |
| 4000 | 4730 | 410 |
| 5000 | 6690 | 470 |
| 6000 | 13690 | 520 |
| 7000 | 13060 | 560 |
| 8000 | 15310 | 590 |
| 9000 | 18170 | 610 |
| 10000 | 22740 | 710 |

Experiment 4: Number of TD: 1000 to 10000
Support degree: 0.015

Table 2 Complexity comparisons of Algorithms with support degree 0.015%

| Number of TD. | Apriori Algorithm (Time complicity in millisecond) | Advance Algorithm (Time complicity in millisecond) |
|---|---|---|
| 1000 | 880 | 240 |
| 2000 | 2240 | 310 |
| 3000 | 3890 | 360 |
| 4000 | 4830 | 440 |
| 5000 | 8320 | 430 |
| 6000 | 14370 | 500 |
| 7000 | 4270 | 430 |
| 8000 | 3530 | 450 |
| 9000 | 4030 | 510 |
| 10000 | 5330 | 550 |

Experiment 5: Number of TD: 1000 to 10000
Support degree: 0.05

**Table 3 Complexity comparisons of Algorithms with support degree 0.05%**

| Number of TD. | Apriori Algorithm (Time complicity in millisecond) | Advance Algorithm (Time complicity in millisecond) |
|---|---|---|
| 1000 | 920 | 230 |
| 2000 | 2020 | 330 |
| 3000 | 830 | 240 |
| 4000 | 1550 | 290 |
| 5000 | 1410 | 380 |
| 6000 | 1860 | 440 |
| 7000 | 1590 | 410 |
| 8000 | 1710 | 390 |
| 9000 | 1620 | 390 |
| 10000 | 1850 | 430 |

From table 1, 2, 3, we can see that execution time of Advance algorithm is shorter than the Apriori algorithm while under of different number of TD and different support degree.

## 7. Conclusions

In this paper an advance algorithm is proposed which enhance the performance of improved Apriori algorithm. Our improvement consists of two parts which are reducing the number of database passes and reducing the unnecessary generation of candidate item-sets by removing the item which are not available in one item-set list. Validated by the experiments, the improvement is notable.



### REFERENCES

[1] Agrawal; Rakesh;, "Fast Algorithms for Mining Association Rules in Large Databases", Proceedings of the ACM SIGMOD International Conference Management of Data, Washington, 1993, pp.207-216.

[2] AgrawaI.R; Imielinski.T; Swami.A; Mining Association rules between Sets of Items in large Databases [Cl In Proceedings of the ACM-SIGMOD Conference on Management of Data,1993:207-2l6

[3] Fayyad, Piatetsky-Shapiro, Smyth, "From Data Mining to Knowledge Discovery: An Overview", in Fayyad, Piatetsky-Shapiro, Smyth, Uthurusamy, *Advances in Knowledge Discovery and Data Mining*, AAAI Press / The MIT Press, Menlo Park, CA, 1996, pp.1-34

[4] M. Dhanabhakyam; M. Punithavalli, A Survey on Data Mining Algorithm for MarketBasketAnalysis.Global Journal of Computer Science and Technology. Vol 11, July 2011.

[5] Chen, M. S.; Han, J.; and Yu, P.S. Data Mining: An Overview from a Database Perspective. IEEE Transactions on Knowledge and Data Engineering, 8(6): 866-883, 1996.

[6] Agrawal and R. Srikant. 1994. Fast algorithms for mining association rules. In Proceedings of the 20th international conference on very large databases held in Santiago, Chile, September 12-15, 1994, 487-99.

[7] M. Housman and A. Swami.Set-oriented mining of association rules. Research Report RJ 9567, IBM Almaden Research Center, San Jose, Cali-fornia, October 1993.