International Journal of Research
Available at https://journals.pen2print.org/index.php/ijr/

e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 06 Issue 10
September 2019

Study on Apriori Algorithm and FP-Growth Algorithm in Association Rule Mining

Kyi Zar Nyunt[1], Wint Aye Khaing[2], Thida Win[3]
[1]Faculty of Information Science, University of Computer Studies (Toungoo), Myanmar
[2,3] Faculty of Information Science, University of Computer Studies (Toungoo), Myanmar
kyizar81@gmail.com, wintayekhaing5@gmail.com, thidawin01@gmail.com

***Abstract*:**

Data mining is a technique dedicated to data analysis and understanding and to reveal the knowledge contained in data. It has become one of the important goals of the application of information technology in the future. Association rule mining is the technique that can discover set of frequent items in a transaction. The paper highlight about apriori algorithm and fp-growth algorithm in association rule mining and compare the performance between them. Apriori algorithm discovers the itemset which is frequent and generates candidate itemset. Fp-growth discovers the frequent itemsets without candidate itemset generation.

Keywords

Association Rules, Apriori Algorithm, FP-Growth Algorithm.

1. Introduction

Data Mining is the process of discovering knowledge. It is the process of extracting information from available raw data. The data are stored in databases. There are various kinds of data that can be used in data mining which includes transactional data, statistical data etc. Data mining includes various techniques for each purpose. Techniques include Association rule mining, classification and prediction, regression etc [6].

The overall goal of the data mining process is to extract information from a data set and transform it into an understandable structure for further use. Association rule learning is a popular method which is used in data mining for discovering relations between variables in extensive databases. These are applied in order to identify strong rules discovered in databases using different measures [4].

The volume of data is increasing dramatically as the data generated by day-to-day activities. Therefore, mining association rules from massive amount of data in the database is interested for many industries which help in much business can decision making processes, such as cross marketing, Basket data analysis, and promotion assortment. It helps to find the association relationship among the large number of database items and its most typical application is to find the new useful rules in the sales transaction database, which reflects the customer purchasing behavior patterns, such as the impact on the other goods after buying a certain kind of goods. These rules can be used in many fields, such as customer shopping analysis, additional sales, goods shelves design, storage planning and classifying the users according to the buying patterns, etc [1].

2. Association Rule Mining

Association rules reflect the interesting association between data items in databases, and frequent itemsets discovery is the key technology and steps in the application of association rules mining [9]. Association rule is applied on the large amount of data. For example, the association technique is used in market basket analysis to identify what products that customers frequently purchase together. Based on this data businesses can have corresponding marketing campaign to sell more products to make more profit [8].

Association rule are the statements that find the relationship between data in any database. Association rule has two parts “Antecedent” and “Consequent”. For example {bread} => {eggs}. Here bread is the antecedent and egg is the consequent. Antecedent is the item that is found in the database, and consequent is the item that is found in combination with the first [2].

Generally a data is gathered from some specific technique, but which technique or which rule/method is suitable for retrieving the appropriate data, and how many time it appears. **Support** means fraction of transaction that contains item-set. **Confidence** means percentage of data in item-set. Both give the occurrence of data which is followed by group of data. Following example shows the support and confidence of data.

Example: -Given a set of transactions, find rules that will predict the occurrence of an item based on the occurrences of other items in the transaction [8]

Table 1:-Market Basket Data Table

TID	Items
1	Bread, Milk
2	Bread, Cloths, Beer, Eggs
3	Milk, Cloths, Beer, Coke
4	Bread, Milk, Cloths, Beer
5	Cloths, Bread, Milk, Coke

{Cloths}→{Beer}
{Milk, Bread}→ {Eggs, Coke}
{Beer, Bread} → {Milk}

Where, Implication relates co-occurrence, not causality

DEFINITION OF FREQUENT ITEM SET

Item-set

- A collection of one or more items
  Example: {Milk, Bread, Cloths}
- k-item-set
  An item-set that contains k items

Support count (σ)

- Frequency of occurrence of an item-set
- E.g. σ ({Milk, Bread, Cloths}) = 2

Support

- Fraction of transactions that contain an item-set
- E.g. s({Milk, Bread, Cloths}) = 2/5

Frequent Item-set

- An item-set whose support is greater than or equal to a *minsup* threshold

DEFINITION OF ASSOCIATION RULE

An implication expression of the form X →Y, where X and Y are item-sets

Example:

{Milk, cloths} → {Beer}

Rule Evaluation Metrics

Support (s)

- Fraction of transactions that contain both X and Y

Confidence (c)

- Measures how often items in Y appear in transactions that contain X

From Table 1:

{Milk, Cloths}=>Beer

S= σ (Milk, Cloths, Beer) /|T|=2/5=0.4

C = σ (Milk, Cloths, Beer)/ σ (Milk, Cloths) =2/3=0.67

3. Apriori Algorithm

Apriori is an association rule mining technique which when given the input of transactional database, it mines all frequently occurring items in the transaction [6]. 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 [1].

```
(1) L1 = {large 1-itemsets};
(2) FOR (k=2; Lk-1≠Φ; k++) DO BEGIN
(3) Ck=apriori-gen(Lk-1); // Ck is a candidate set of k elements
(4) FOR all transactions t∈DDO BEGIN
(5) Ct=subset(Ck,t); // Ct is the candidate set element contained by all t
(6) FOR all candidates c∈Ct DO
(7) c.count++;
(8) END
(9) Lk={c∈Ck|c.count≥minsup_count}
(10) END
(11) Answer= ∪kLk;
```

Table.2 Sample transaction database

TID	Itemset
1	A, B, C, D
2	B, C, E
3	A, B, C, E
4	B, D, E
5	A, B, C, D

The implementation of the Apriori algorithm for the transaction database shown in table 1 is as follows (set minsupport =40%):

(1) L1 generation: generating candidate set and the number of support them by scanning the database, C1={(A, 3), (B, 5), (C, 4), (D, 3), (E, 3)}; minsup_count 2 set of items selected and 1- frequent itemsets L1={A, B, C, D, E}.

(2): L2 formation generated by the L1 2- candidate set and get their support number C2={by scanning the database (AB, 3), (AC, 3), (AD, 2), (AE, 1), (BC, 4), (BD, 3), (BE, 3). (CD, 2), (CE, 2), (DE, 1)}; minsup_count choose >2 project consisting of 2- frequent itemsets L2={AB, AC, AD, BC, BD, BE, CD, CE}.

(3): L3 formation generated by the L2 3- candidate set and get their support number C3={by scanning the database (ABC, 3), (ABD, 2), (ACD, 2), (BCD, 2), (BCE, 2)}; select minsup_count>2 project consisting of 3- frequent itemsets L3={ABC ABD, ACD, BCD, BCE}.

(4) L4 generation: the 4- candidate is generated by L3, and their support numbers C4={(ABCD, 2)} are obtained by scanning the database; selecting the minsup_count>2 itemsets to form the 4- frequent itemsets L4 ={ABCD}.

(5) L5 generation: the 5- candidate set L4 is generated by C5= Ø, L5= Ø, and the algorithm is stopped.

The frequent itemsets are {ABCD, BCE}. Obviously, the Apriori algorithm has two fatal performance bottlenecks:

1) You repeatedly scan the database, which needs a lot of I/O load. For each k loop, each element in the candidate Ck must be verified by scanning the database once to add Lk. If there is a frequent itemset containing 10 items, then you need to scan the transaction database at least 10 times.

2) It may generate huge candidate sets. The k- candidate Lk-1 produced by Ck is exponentially increasing, for example, 104 of the 1- frequent itemsets are likely to produce 2-candidates of nearly 107 elements. Such a large collection of selections is a challenge to both time and memory space [9].

4. FP-Growth Algorithm

Large databases are compressed into compact FP tree structure. FP tree structure stores necessary information about frequent item sets in a database [4]. FP Growth algorithm discovers the frequent itemset without the candidate generation. It follows two steps such as: In step one it builds a compact data structure called the FP-Tree, in step two it directly extracts the frequent itemsets from the FP-Tree. FP-Tree was proposed by Han [3]. The advantage is that it constructs conditional pattern base from database which satisfies minimum support, due to compact structure and no candidate generation it requires less memory. The disadvantage is that it performs badly with long pattern data sets.

FP-Tree was proposed by Han. FP-Tree represents all the relevant frequent information from a data set due to its compact structure. Each and every path of FP-Tree represents a frequent itemset and nodes in the path are arranged in a decreasing order of the frequency. The great advantage of FP-Tree is that all the overlapping itemsets share the same prefix path. Because of this the information of the data set is highly compressed. It scans the database only twice and it does not need any candidate generation [7].FP-Tree is constructed using 2 passes over the data set.

Pass1: It first scans the data and then find support for each item. Then it discards the infrequent itemsets and sort the frequent itemsets in decreasing order based on their support.

Pass2: Nodes corresponds to itemset and have a counter.

1. It first reads 1 transaction at a time and maps it to a path.
2. Fixed order is used so paths can overlap when transaction shares items (when they have same prefix) I this case, counters are incremented.
3. Pointers are maintained between nodes containing the same item, resulting a linked list (dotted lines). The compression will be high based on the more paths that overlap. FP-Tree may fit in the memory.
4. Frequent itemsets are extracted from the FP-Tree [5].

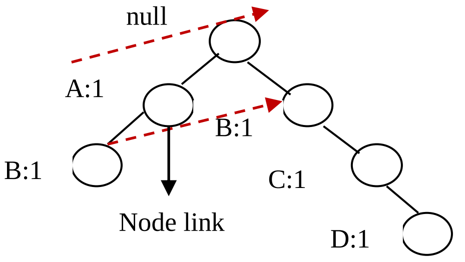

Fig 2: Tree after reading 2nd transaction

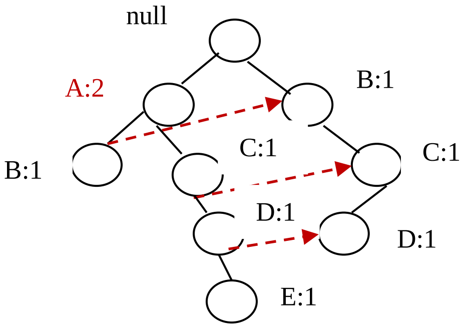

Fig 3: Tree after reading 3rd transaction

Table.3 Sample transaction database 2

TID	Items
1	{A,B}
2	{B,C,D}
3	{A,C,D,E}
4	{A,D,E}
5	{A,B,C}
6	{A,B,C,D}
7	{A}
8	{A,B,C}
9	{A,B,D}
10	{B,C,E}

Minimum support count=2

Scan database to find frequent 1-itemsets s(A)=8, s(B) s(C)=5, s(D)=5, s(E)=3

Item order (decreasing support) = A,B,C,D,E

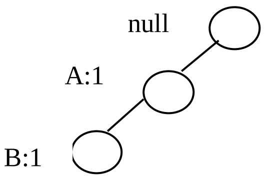

Fig 1: Tree after reading 1st transaction

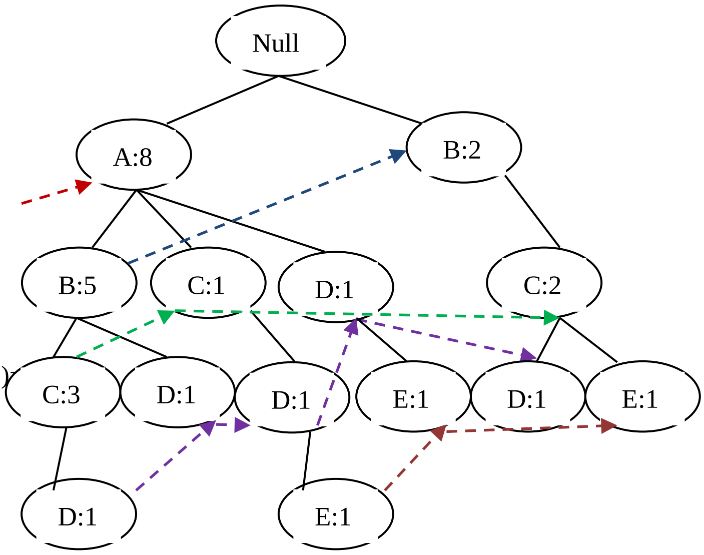

Fig 4: Pointers to speed up lookup

5. Comparison between Apriori Algorithm and FP-Growth Algorithm

Both Apriori and FP Growth algorithm are used to mine the frequent patterns from database. Both the algorithm uses some technique to discover the frequent patterns. Apriori algorithm works well with large database but FP Growth algorithm works badly with large database [12].

Apriori means "from what comes before" and uses breadth first search technique. Its implementation is easier than other algorithms and consumes less memory. However it has certain disadvantages also. It only explains the presence and absence of an item in transactional databases and requires a large number of database scan. Moreover the minimum support threshold used is uniform and the number of candidate itemsets produced is large. To overcome some of the bottlenecks of the Apriori algorithm Fp-growth algorithm was designed which is based on tree structure.

The frequent itemsets are generated with only two passes over the database and without any candidate generation process thus making it faster than the Apriori algorithm. FP-growth uses a compressed representation of the database thus the irrelevant information are pruned. However it cannot be used for interactive and incremental mining system as changes in threshold value or new insertions in database may lead to a repetition of the whole process if we employ FP-tree method [5].

Table.4 Performance of Apriori Algorithm and FP-Growth Algorithm

	Advantages	Disadvantages
Apriori	1.This algorithm has least memory consumption. 2.Easy implementation. 3.It uses Apriori property for pruning therefore, itemsets left for further support checking remain less.	1. It requires many scans of database. 2. It allows only a single minimum support threshold. 3. It is favourable only for small database. 4. It explains only the presence or absence of an item in the database.
FP-growth	1. It is faster than other association rule mining algorithm. 2.It uses compressed representation of original database. 3.Repeated database scan is eliminated.	1. The memory consumption is more. 2. It cannot be used for interactive mining and incremental mining. 3. The resulting FP-Tree is not unique for the same logical database [2]

6. Conclusion

Association rules are widely used in various areas such as telecommunication networks, risk and market management, medical diagnosis, inventory control etc. Association rule mining involves the relationships between items in a data set. It classifies a given transaction as a subset of the set of all possible items. Association rule mining finds out item sets which have minimum support and are represented in relatively high number of transactions. These transactions are simple known as frequent item sets. In this paper, we study on Apriori algorithm and FP-Growth algorithm in association rule mining. The techniques, association rule mining, advantages and disadvantages of both algorithms are discussed briefly. Apriori and FP-Growth are two algorithms for frequent itemset mining. Apriori utilize a level-wise approach where it will generate patterns containing 1 items, 2 items, 3 items, etc. Moreover, it will repeatedly scan the database to count the support of each pattern. On the other hand, FP-Growth utilizes a depth-first search instead of a breadth first search and uses a pattern-growth approach.

7. References

[1] Charanjeet Kaur, "Association Rule mining using Apriori Algorithm: A Survey", International Journal of *Advanced Research in Computer Engineering & Technology (IJARCET)*, Volume 2, Issue 6, June 2013.

[2] Gurneet Kaur, "Association Rule Mining: A Survey", *(IJCSIT) International Journal of Computer Science and Information Technologies*, Vol. 5(2), 2014, 2320-2324.

[3] Kamber, M., Han, J., and Chiang, J.Y. 1997. Metarule-guided mining of multi-dimensional association rules using data cube. In Proc. 1997 Int. Conf. Knowledge Discovery and Data Mining (KDD'97) Newport Beach, CA, pp.207-210.

[4] Manisha Girotra, Kanika Nagpal, Saloni Minocha and Neha Sharma, "Comparative Survey on Association Rule Mining Algorithms", *International Journal of Computer Applications (0975-8887)*, Volume 84- No 10, December 2013.

[5] Mrs. M.Kavitha and Ms.S.T.Tamil Selvi, "Comparative Study on Apriori Algorithm and Fp Growth Algorithm with Pros and Cons", *International Journal of Computer Science Trends and Technology (IJCST)*-Volume 4 Issue 4, Jul-Aug 2016.

[6] MS.J.Omana, MS.S.Monika and MS.B.Deepika, "SURVEY ON EFFICIENCY OF ASSOCIATION RULE MINING TECHNIQUES", *International Journal of Computer Science and Mobile Computing*, Vol.6 Issue 4, April-2017, pg. 5-8.

[7] Prashasti Kanikar, Twinkle Puri, Binita Shah, Ishaan Bazaz, Binita Parekh, "A Comparison of FP tree and Apriori algorithm", *International Journal of Engneering Research and Development*, Volume 10, Issue 6, pp 78-82, June 2014.

[8] Rahul B, Diwate and Amit Sahu, "Data Mining TEchniques in Association Rule : A Review", *(IJCSIT) International Journal of Computer Science and Information Technologies*, Vol. 5(1), 2014, 227-229.

[9] Yutang Liu and Qin Zhang, "Research on Association Rules Mining Algorithm Based on Large Data", Revista de la Facultad de Ingenieria U.C.V., Vol. 32, N 8, pp. 229-236, 2017.