ISSN No. 0976-5697

Volume 2, No. 1, Jan-Feb 2011

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

A Tabular Approach for Frequent itemset mining

G.Krishna Mohan
Reader, P.B.Siddhartha College,
Vijayawada-10, Andhrapradesh, India.
km_mm_2000@yahoo.com

Abstract: Frequent patterns are patterns that appear in a data set frequently. For example, a set of items, such as milk and bread that appear frequently together in a transaction data set is a frequent itemset. Frequent pattern mining searches for recurring relationships in a given data set. With massive amounts of data continuously being collected and stored, many industries are becoming interested in mining such patterns from their databases. The discovery of interesting correlation relationships among huge amounts of business transaction records can help in many business decision-making processes, such as catalog design, cross-marketing, and customer shopping behavior analysis. If we think of the universe as the set of items available at the store, then each item has a Boolean variable representing the presence or absence of that item. Each basket can then be represented by a Boolean vector of values assigned to these variables. The Boolean vectors can be analyzed for buying patterns that reflect items that are frequently associated or purchased together. A set of items is referred to as an itemset. An itemset that contains k items is a k-itemset. The occurrence frequency of an itemset is the number of transactions that contain the itemset. This is also known, simply, as the frequency, support count, or count of the itemset. The set of frequent k-itemsets is commonly denoted by L_k.

Keywords: Apriori; table; AND operation; database; frequent itemsets

I. INTRODUCTION

A set of items is referred to as an itemset. An itemset that contains k items is a k-itemset. The occurrence frequency of an itemset is the number of transactions that contain the itemset. This is also known, simply, as the frequency, support count, or count of the itemset. The set of frequent k-itemsets is commonly denoted by L_k.

The efficiency of frequent itemset mining algorithms is determined mainly by three factors: the way candidates are generated, the data structure that is used and the implementation details. Most papers focus on the first factor, some describe the underlying data structures, but implementation details are almost always neglected. Indexed Apriori algorithm operates on sorted transactions in the database according to the number of items in the transaction.

II. LITERATURE REVIEW

Apriori [1] is the most popular algorithm to find all the frequent sets. It is also called the level-wise algorithm. The nicety of the algorithm is that before reading the database at every level, it graciously prunes many of the sets which are unlikely to be frequent sets.

The first pass of the algorithm simply counts item occurrences to determine the frequent itemsets. A subsequent pass, pass k, consists of two phases. First, the frequent itemsets L_{k-1} found in the (k-1)th pass are used to generate the candidate itemsets C_k, using the apriori candidate generation procedure. Next, the database is scanned and the support of candidates in C_k is counted. For fast counting, we need to efficiently determine the candidates in C_k contained in a given transaction 't'. The set of candidate itemsets is ubjected to a pruning process to ensure that all the subsets of the candidate sets are already known to be frequent itemsets. The candidate generation process and the pruning process are the most important parts of this algorithm.

III. PROPOSED PROCEDURE

The sample database considered to be as shown in the table I. The items involved are {I1,I2,I3,I4,I5,I6,I7,I8,I9,I10} and the minsupport is considered as 3.

Table I. Sample Database

Transactions	*items*
T1	**I3,I4,I6**
T2	**I1,I2,I3,I4,I5,I6**
T3	**I4,I5,I6,I9**
T4	**I4,I6,I8,I10**
T5	**I1,I5,I6,I7,I8,I10**
T6	**I3,I4,I5,I6,I7,I8,I10**
T7	**I1,I3,I5,I7,I9**
T8	**I2,I9**
T9	**I3,I5,I7,I8,I9,I10**
T10	**I1,I4,I9**

Construct the Initial table as, Items as rows and Transactions as columns. Map the cells in the table with '1' where the items occurred in a transaction. Calculate all the RCs.

Table II. Initial Table

	T1	T2	T3	T4	T5	T6	T7	T 8	T 9	T 10	R C
I1		**1**			**1**		**1**			**1**	**4**
I2		**1**						**1**			**2**
I3	**1**	**1**				**1**	**1**		**1**		**5**
I4	**1**	**1**	**1**	**1**		**1**				**1**	**6**
I5		**1**	**1**		**1**	**1**	**1**		**1**		**6**
I6	**1**	**1**	**1**	**1**	**1**	**1**					**6**
I7					**1**	**1**	**1**		**1**		**4**
I8				**1**	**1**	**1**			**1**		**4**

I9			**1**				**1**	**1**	**1**	**1**	**5**
I10				**1**	**1**	**1**			**1**		**4**

SimplifyTab1(D,minsupport):

Check the RCs in the Initial table, if there are any RCs less than minsupport then remove the corresponding items from the table. If there are no items left in the table, which implies there are no frequent item sets.

Table III. SimplifiedTab1

	T 1	**T 2**	**T 3**	**T 4**	**T 5**	**T 6**	**T 7**	**T 8**	**T 9**	**T 10**	**R C**
I1		1			1		1			1	4
I3	1	1				1	1		1		5
I4	1	1	1	1		1				1	6
I5		1	1		1	1	1		1		6
I6	1	1	1	1	1	1					6
I7					1	1	1		1		4
I8				1	1	1			1		4
I9			1				1	1	1	1	5
I10				1	1	1			1		4
C C	3	5	4	4	6	7	5	1	6	2	

Find -1 :

Generate all items which were left in the simplifiedTab1 as 1_itemset frequent itemsets.

L_1= {I1},{I3},{I4},{I5},{I6},{I7},{I8},{I9}

SimplifyTab2(SimplifiedTab1,2,minsupport) :

Remove the transactions whose CCs is less than 2 and then remove the corresponding items whose RCs is less than minsupport from the SimplifiedTab1.

Table IV. SimplifiedTab2

	T 1	**T 2**	**T 3**	**T 4**	**T 5**	**T 6**	**T 7**	**T 9**	**T 10**	**RC**
I1		1			1		1		1	4
I3	1	1				1	1	1		5
I4	1	1	1	1		1			1	6
I5		1	1		1	1	1	1		6
I6	1	1	1	1	1	1				6
I7					1	1	1	1		4
I8				1	1	1		1		4
I9			1				1	1	1	4
I10				1	1	1		1		4

for item = I1

NewTab←GenerateNewTab(SimplifiedTab2,item)

Identifying the first item in the SimplifiedTab2, construct a NewTab considering all the transactions in which that item exists.

Table V. NewTab

	T2	**T5**	**T7**	**T10**	**RC**
I1	1	1	1	1	4
I3	1		1		2
I4	1			1	2
I5	1	1	1		3
I6	1	1			2
I7		1	1		2
I8		1			1
I9			1	1	2
I10		1			1
CC	5	6	5	3	

SimplifyTab2(NewTab,2,minsupport) :

Remove the transactions whose CC is less than 2 and then remove the corresponding items whose RCs is less than minsupport from the NewTab.

Table VI. SimplifiedTab

	T2	**T5**	**T7**	**T10**	**RC**
I1	1	1	1	1	4
I5	1	1	1		3

Find -2 :

Generate all item combinations which were left in the SimplifiedTab as 2_itemset frequent itemsets for an item.

L_2= {I1,I5}

SimplifyTab2(NewTab,3,minsupport) :

Remove the transactions whose CC is less than 3 and then remove the corresponding items whose RCs is less than minsupport from the SimplifiedTab.

Find -3 :

L_3= NULL

for item = I3

NewTab←GenerateNewTab(SimplifiedTab2,item)

Identifying the next item in the SimplifiedTab2, construct a NewTab considering all the transactions in which that item exists.

Table VII. NewTab

	T1	**T2**	**T6**	**T7**	**T9**	**RC**
I3	1	1	1	1	1	5
I4	1	1	1			3
I5		1	1	1	1	4
I6	1	1	1			3
I7			1	1	1	3
I8			1		1	2
I9				1	1	2
I10			1		1	2
CC	3	4	7	4	6	

SimplifyTab2(NewTab,2,minsupport) :

Remove the transactions whose CC is less than 2 and then remove the corresponding items whose RCs is less than minsupport from the NewTab.

Table VIII. SimplifiedTab

	T1	**T2**	**T6**	**T7**	**T9**	**RC**
I3	1	1	1	1	1	5
I4	1	1	1			3
I5		1	1	1	1	4
I6	1	1	1			3
I7			1	1	1	3

Find -2 :
Generate all item combinations which were left in the simplifiedTab as 2_itemset frequent itemsets for an item.

L_2= {I3,I4} {I3,I5} {I3,I6} {I3,I7}

SimplifyTab2(SimplifiedTab,3,minsupport) :
Remove the transactions whose CCs is less than 3 and then remove the corresponding items whose RCs is less than minsupport from the SimplifiedTab.
Find -3 :
Identify the 3 item transactions and check whether the items as a set are frequent , comparing with the rest of the transactions. If so, generate the items as a frequent itemset and remove the transaction along with the transactions which are of same.

	T1	T2	T6	T7	T9	RC
I3	1	1	1	1	1	5
I4	1	1	1			3
I5		1	1	1	1	4
I6	1	1	1			3
I7			1	1	1	3
	3	4	5	3	3	

L_3= {I3,I4,I6},

SimplifyTab2(SimplifiedTab,3,minsupport) :
Remove the transactions whose CC is less than 3 and then remove the corresponding items whose RCs is less than minsupport from the SimplifiedTab.

	T2	T6	T7	T9	RC
I3	1	1	1	1	4
I4	1	1			2
I5	1	1	1	1	4
I6	1	1			2
I7		1	1	1	3
	4	5	3	3	

	T2	T6	T7	T9	RC
I3	1	1	1	1	4
I5	1	1	1	1	4
I7		1	1	1	3
	2	3	3	3	

	T6	T7	T9	RC
I3	1	1	1	4
I5	1	1	1	4
I7	1	1	1	3
	3	3	3	

L_3= {I3,I5,I7}

for item = I4
NewTab←GenerateNewTab(SimplifiedTab2,item)
Identifying the next item in the simplifiedTab2, construct a NewTab considering all the transactions in which that item exists.

Table IX. NewTab

	T1	T2	T3	T4	T6	T10	RC
I4	1	1	1	1	1	1	6
I5		1	1		1		3
I6	1	1	1	1	1		5
I7					1		1
I8				1	1		2
I9			1			1	2
I10				1	1		2
CC	2	3	4	4	6	2	

SimplifyTab2(NewTab,2,minsupport) :
Remove the transactions whose CC is less than 2 and then remove the corresponding items whose RCs is less than minsupport from the NewTab.

Table X. simplifiedTab

	T1	T2	T3	T4	T6	T10	RC
I4	1	1	1	1	1	1	6
I5		1	1		1		3
I6	1	1	1	1	1		5

Find -2 :
Generate all item combinations which were left in the simplifiedTab as 2_itemset frequent itemsets for an item.

L_2= {I4,I5} {I4,I6}

SimplifyTab2(SimplifiedTab,3,minsupport) :
Remove the transactions whose CCs is less than 3 and then remove the corresponding items whose RCs is less than minsupport from the SimplifiedTab.

	T1	T2	T3	T4	T6	T10	RC
I4	1	1	1	1	1	1	6
I5		1	1		1		3
I6	1	1	1	1	1		5
CC	2	3	3	2	3	1	

Find -3 :
Identify the 3 item transactions and check whether the items as a set are frequent comparing with the rest of the transactions. If so, generate the items as a frequent itemset and remove the transaction along with the transactions which are of same.

	T2	T3	T6	RC
I4	1	1	1	3
I5	1	1	1	3
I6	1	1	1	3
CC	3	3	3	

L_3= {I4,I5,I6}

for item = I5
NewTab←GenerateNewTab(SimplifiedTab2,item)
Identifying the next item in the simplifiedTab2, construct a NewTab considering all the transactions in which that item exists.

Table XI. NewTab

	T2	T3	T5	T6	T7	T9	RC
I5	1	1	1	1	1	1	6
I6	1	1	1	1			4
I7			1	1	1	1	4
I8			1	1		1	3
I9		1			1	1	3
I10			1	1		1	3
CC	2	3	5	5	3	5	

Find -2 :
Generate all item combinations which were left in the simplifiedTab as 2_itemset frequent itemsets for an item.

L_2= {I5,I6} {I5,I7} {I5,I8}{I5,I9} {I5,I10}

SimplifyTab2(SimplifiedTab,3,minsupport) :
Remove the transactions whose CC is less than 3 and then remove the corresponding items whose RCs is less than minsupport from the SimplifiedTab.

	T2	T3	T5	T6	T7	T9	RC
I5	1	1	1	1	1	1	6
I6	1	1	1	1			4
I7			1	1	1	1	4
I8			1	1		1	3
I9		1			1	1	3
I10			1	1		1	3
CC	2	3	5	5	3	5	

	T3	T5	T6	T7	T9	RC
I5	1	1	1	1	1	5
I6	1	1	1			3
I7		1	1	1	1	4
I8		1	1		1	3
I9	1			1	1	3
I10		1	1		1	3
CC	3	5	5	3	5	

Find -3 :
Identify the 3 item transactions and check whether the items as a set are frequent comparing with the rest of the transactions. If so, generate the items as a frequent itemset and remove the transaction along with the transactions which are of same.

	T5	T6	T7	T9	RC
I5	1	1	1	1	4
I6	1	1			2
I7	1	1	1	1	4
I8	1	1		1	3
I9			1	1	2
I10	1	1		1	3
CC	5	5	3	5	

	T5	T6	T7	T9	RC
I5	1	1	1	1	4
I7	1	1	1	1	4
I8	1	1		1	3
I10	1	1		1	3
CC	4	4	2	4	

	T5	T6	T9	RC
I5	1	1	1	4
I7	1	1	1	4
I8	1	1	1	3
I10	1	1	1	3
CC	4	4	4	

L_3= {I5,I7,I8} {I5,I7,I10} {I5,I8,I10}

Find -4 :
Generate all item combinations which were left in the simplifiedTab as 4_itemset frequent itemsets for an item.

L_4= {I5,I7,I8,I10}

for item = I6
NewTab←GenerateNewTab(SimplifiedTab2,item)
Identifying the first item in the simplifiedTab2, construct a NewTab considering all the transactions in which that item exists.

Table XII. NewTab

	T1	T2	T3	T4	T5	T6	RC
I6	1	1	1	1	1	1	6
I7					1	1	2
I8				1	1	1	3
I9			1				1
I10				1	1	1	3
CC	1	1	2	3	4	4	

SimplifyTab2(NewTab,2,minsupport) :
Remove the transactions whose CC is less than 2 and then remove the corresponding items whose RCs is less than minsupport from the NewTab.

	T1	T2	T3	T4	T5	T6	RC
I6	1	1	1	1	1	1	6
I7					1	1	2
I8				1	1	1	3
I9			1				1
I10				1	1	1	3

	T3	T4	T5	T6	RC
I6	1	1	1	1	4
I7			1	1	2
I8		1	1	1	3
I9	1				1
I10		1	1	1	3

Table XIII. SimplifiedTab

	T3	T4	T5	T6	RC
I6	1	1	1	1	4
I8		1	1	1	3
I10		1	1	1	3

Find -2 :

Generate all item combinations which were left in the simplifiedTab as 2_itemset frequent itemsets for an item.

L_2= {I6,I8} {I6,I10}

SimplifyTab2(SimplifiedTab,3,minsupport) :
Remove the transactions whose CCs is less than 3 and then remove the corresponding items whose RCs is less than minsupport from the SimplifiedTab.

	T4	T5	T6	RC
I6	1	1	1	6
I8	1	1	1	3
I10	1	1	1	3
CC	3	3	3	

Find -3 :
Identify the 3 item transactions and check whether the items as a set are frequent comparing with the rest of the transactions. If so, generate the items as a frequent itemset and remove the transaction along with the transactions which are of same.

L_3= {I6,I8,I10}

for item = I7
NewTab←GenerateNewTab(SimplifiedTab2,item)
Identifying the first item in the simplifiedTab2, construct a NewTab considering all the transactions in which that item exists.

Table XIV. NewTab

	T5	T6	T7	T9	RC
I7	1	1	1	1	4
I8	1	1		1	3
I9			1	1	2
I10	1	1		1	3
CC	3	3	2	4	

SimplifyTab2(NewTab,2,minsupport) :
Remove the transactions whose CC is less than 2 and then remove the corresponding items whose RCs is less than minsupport from the NewTab.

	T5	T6	T7	T9	RC
I7	1	1	1	1	4
I8	1	1		1	3
I9			1	1	2
I10	1	1		1	3

Table XV. simplifiedTab

	T5	T6	T7	T9	RC
I7	1	1	1	1	4
I8	1	1		1	3
I10	1	1		1	3

Find -2 :
Generate all item combinations which were left in the simplifiedTab as 2_itemset frequent itemsets for an item.

L_2= {I7,I8} {I7,I10}

SimplifyTab2(SimplifiedTab,3,minsupport) :
Remove the transactions whose CCs is less than 3 and then remove the corresponding items whose RCs is less than minsupport from the SimplifiedTab.

	T5	T6	T7	T9	RC
I7	1	1	1	1	4
I8	1	1		1	3
I10	1	1		1	3
CC	3	3	1	3	

	T5	T6	T9	RC
I7	1	1	1	3
I8	1	1	1	3
I10	1	1	1	3
CC	3	3	3	

Find -3 :
Identify the 3 item transactions and check whether the items as a set are frequent comparing with the rest of the transactions. If so, generate the items as a frequent itemset and remove the transaction along with the transactions which are of same.

L_3= {I7,I8,I10}

for item = I8
NewTab←GenerateNewTab(SimplifiedTab2,item)
Identifying the first item in the simplifiedTab2, construct a NewTab considering all the transactions in which that item exists.

Table XVI. NewTab

	T4	T5	T6	T9	RC
I8	1	1	1	1	4
I9				1	1
I10	1	1	1	1	4
CC	2	2	2	3	

SimplifyTab2(NewTab,2,minsupport) :
Remove the transactions whose CC is less than 2 and then remove the corresponding items whose RCs is less than minsupport from the NewTab.

Table XVII. simplifiedTab

	T4	T5	T6	T9	RC
I8	1	1	1	1	4
I10	1	1	1	1	4

Find -2 :
Generate all item combinations which were left in the simplifiedTab as 2_itemset frequent itemsets for an item.

L_2= {I8,I10}

SimplifyTab2(NewTab,3,minsupport) :
Remove the transactions whose CCs is less than 3 and then remove the corresponding items whose RCs is less than minsupport from the SimplifiedTab.

L_3= NULL

for item = I9
NewTab←GenerateNewTab(SimplifiedTab2,item)

Identifying the first item in the simplifiedTab2, construct a NewTab considering all the transactions in which that item exists.

Table XVIII. NewTab

	T3	T7	T9	T10	RC
I9	1	1	1	1	4
I10			1		1
CC	1	1	2	1	

SimplifyTab2(NewTab,2,minsupport) :
Remove the transactions whose CC is less than 2 and then remove the corresponding items whose RCs is less than minsupport from the NewTab.

Table XIX. simplifiedTab

	T3	T7	T9	T10	RC
I9	1	1	1	1	4
I10			1		1
CC	1	1	2	1	

	T9	RC
I9	1	1
I10	1	1
CC	2	

Find -2 :
Generate all item combinations which were left in the simplifiedTab as 2_itemset frequent itemsets for an item.

L_2= NULL

IV. ALGORITHM

Input: database D, minsupport
Output : All the frequent itemsets in D

1) [*Produce the initiaTtable*]
SimplifiedTab1 ← SimplifyTab1(D,minsupport)
If SimplifiedTab1 = NULL
End, No frequent itemsets
Else
[*Finding 1_itemset frequent itemsets*]
L1 ← find_1
SimplifiedTab2←
SimplifyTab2(SimplifiedTab1,2,minsupport)

2) [*Repeat the procedure from first to last with one item in the table*]
[*start with the initial item in simplifiedTab2 as item*]
NewTab←GenerateNewTab(SimplifiedTab2, item)
[*finding 2_itemset frequent itemsets with item*]
L2 ← find_2.
[*finding 3_itemset frequent itemsets to maximum column_count frequent itemsets*]
While N from 3 to Max(CC) in the New Tab
{
SimplifiedTab ← simplifyTab2(NewTab ,N, minsupport)
if (SimplifiedTab != NULL)
While (Any column count = N)
L_N ← find_N [collection of exact N_item frequent itemsets in the New Tab]
SimplifiedTab ← Remove(SimplifiedTab ,Transaction)
SimplifiedTab ← SimplifyTab2(SimplifiedTab ,N, minsupport)
if (SimplifiedTab != NULL)
[*find out all other N frequent itemsets*]
[*Find out all the possible N item combinations from New Tab items starting with item*]
L_N ← find_N. [*collection of all N_frequent itemsets in the New Tab*]
Increment N
NewTab=SimplifiedTab.
End while
}
Take next item in the table as item and continue the above procedure
End procedure
3) [find out all frequent item sets]
Answers ← L_1 U L_2 U L_3 U L_4 ------ U $L_{maxitem.}$
4) [END]

V. RESULTS

The Frequent itemsets that were generated on the sample database with a minimum support of 3 are L_1 U L_2 U L_3 U L_4
Where,
L_1={I1},{I3},{I4},{I5},{I6},{I7},{I8},{I9}.
L_2={I1,I5}, {I3,I4},{I3,I5},{I3,I6},{I3,I7}, {I4,I5},{I4,I6}, {I5,I6},{I5,I7},{I5,I8},{I5,I9},{I5,I10}, {I6,I8},{I6,I10},{I7,I8},{I7,I10},{I8,I10}.
L_3={I3,I4,I6},{I3,I5,I7},{I4,I5,I6},{I5,I7,I8},{I5,I7,I10}, {I5,I8,I10}, {I6,I8,I10}, {I7,I8,I10}.
L_4={I5,I7,I8,I10}.

VI. CONCLUSION

The performance of the proposed algorithm on some of the sample databases which contain the number of transactions as 20, 40, 60, 80, 100 is shown in the following graph in terms of CPU execution time.

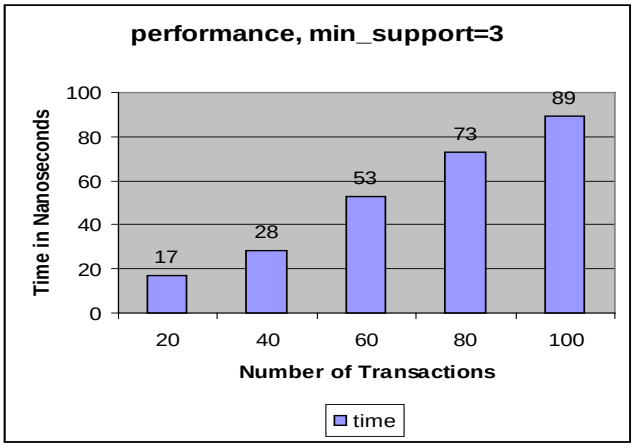

Figure 1. performance of the proposed algorithm

VII.REFERENCES

[1] Jiawei Han, Micheline Kamber, "Data Mining: Concepts and Techniques", 2nd Edition.

[2] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. The International Conference on Very Large Databases, pages 487–499, 1994.

[3] Arun K.Pujari, "Data Mining: Techniques".

[4] http://fimi.cs.helsinki.fi/data/