Programming Assignment 1 Report

Name: Rohan Sadale

Email: sadal005@umn.edu

ID: 5165501

Data Structures used to store Projected Database

A ***map< int, vector<int> >*** is used to store transactions in projected database. Here ***key (int)*** for map is the transaction_id and the ***value*** is a ***vector*** that contains list of all items in the transaction.

A ***vector < pair<int, int> >*** is used to store frequency of elements in projected Database

Implementation of Projected Database

int treeProjection(int item, itemsets &t);

vector < pair<int, int> > getFrequency(itemsets &t);

map< int, vector<int> > projectedDatabase(int item, itemsets &t);

The above three functions are implemented to generate frequent itemsets as per tree projection algorithm.

- TreeProjection() - 3 step tree projection algorithm
- getFrequency() - returns frequent elements in projected database. Based on options value, the elements sorted in ascending or descending order.
- ProjectedDatabase() - returns projected Database for that element.

Algorithm for treeProjection():

treeProjection(element, transactionDB):

frequentItems = getFrequency(transactionDB)

for each frequentItem in frequentItems:

```
t = projectedDatabase(frequentItem,  transactionDB);

treeProjection( frequentItem, t)
```

Algorithm for projectedDatabase():

```
projectedDatabase(element, transactionDB):
    map< int, vector<int> > temp;          //empty transaction database
    for each transaction in transactionDB:
        if element present in transaction:
            remove element from transaction;
            copy the transaction to temp;
    return temp;
```

Here in projectedDatabase(), I used binary search to find element in transaction. This substantially increases performance as my search window for each transaction decreases.

The approach I used is as follows:

1. Initially treeProjection is called with parameters (0, transationDB).

2. I then find frequent elements in the transactionDB and return it through getFrequency(). For options=2 or options=3 this will return elements sorted in respective forms.

3. Then, for each element in frequent elements, I call projectedDatabase function to get it's projected DB.

4. Once I get projectedDB for that element, I recursively call treeProjection() for that element and hence the recursion goes deeper(depth first) all the way till it becomes empty or it has no frequent items. Here for each next sibling, the size of transaction database decreases because I am deleting the element from transaction whose projection has been complete. Thus, size of transaction database goes on decreasing as the program move from left to right. The effect is substantially observed when the program is run for option = 2.

5. The approach that I used can be well understood by above two algorithms.

Note: I am passing the transactionDatabase by reference each time it is called. Hence, there will be no multiple copies of transactionDB

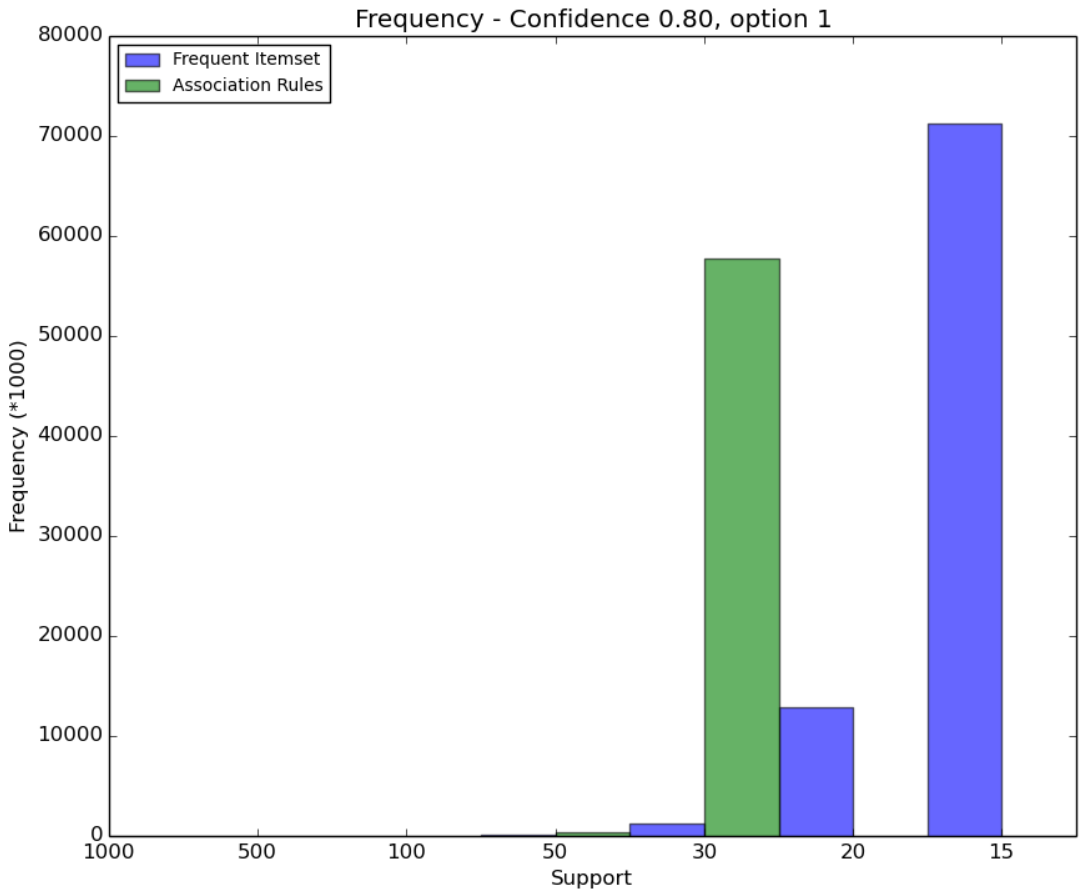
Which approach is faster?

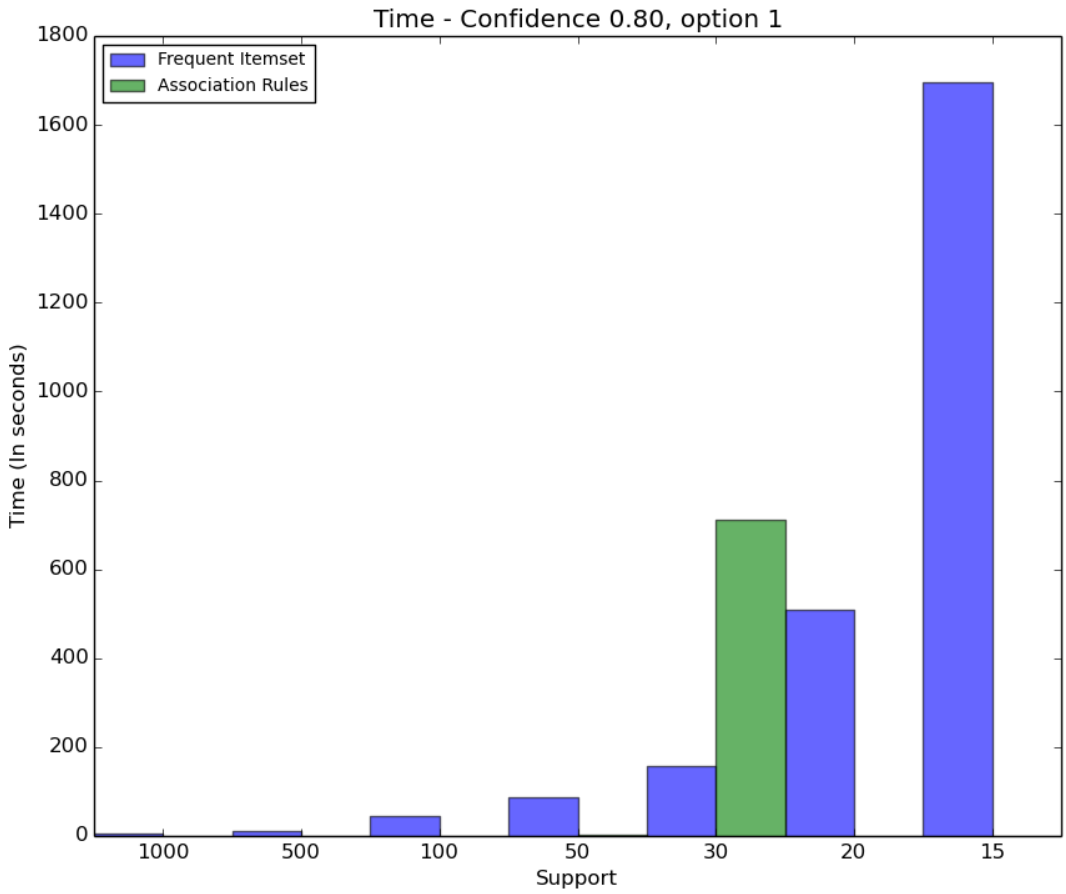
According to the below results, options = 2 will give the most fastest frequent itemsets because, if we order the items from least frequent to most frequent then the computationally heavier branches of tree will have fewer relevant transactions. Thus we can maximize the selectivity of projections and will result in better efficiency. Thus projected databases which are no longer needed for future computations can be deleted and hence the approach will be memory efficient.

Graphs and Readings:

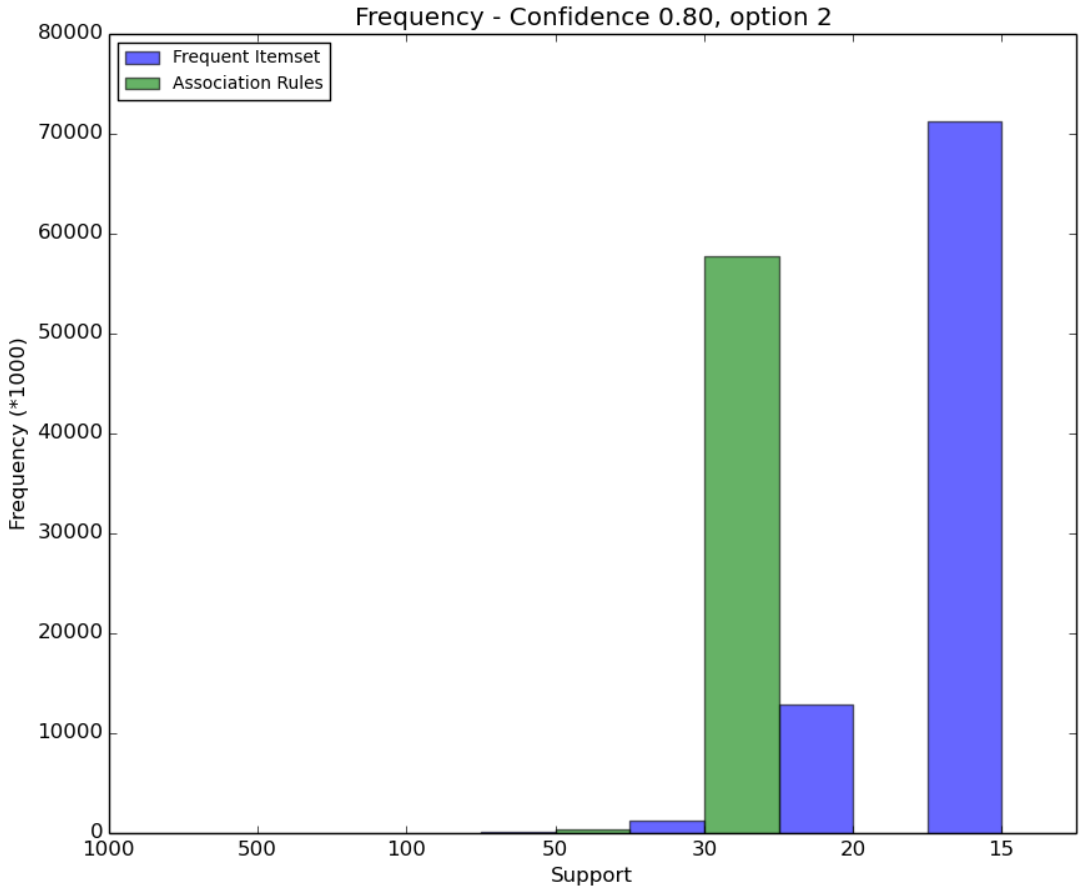
minsup	minconf	options	frequentItems	associationRules	frequentItemsTime	associationRulesTime
1000	0.8	1	635	37	5.829	0.003
1000	0.8	2	635	37	5.555	0.005
1000	0.8	3	635	37	5.69	0.003
1000	0.9	1	635	3	5.731	0
1000	0.9	2	635	3	5.579	0
1000	0.9	3	635	3	5.677	0
1000	0.95	1	635	0	5.735	0
1000	0.95	2	635	0	5.603	0
1000	0.95	3	635	0	5.68	0
500	0.8	1	2303	168	11.478	0.006
500	0.8	2	2303	168	10.046	0.008
500	0.8	3	2303	168	11.678	0.005
500	0.9	1	2303	15	11.502	0.006
500	0.9	2	2303	15	10.024	0.007
500	0.9	3	2303	15	11.66	0.005
500	0.95	1	2303	1	11.502	0.002
500	0.95	2	2303	1	9.964	0.002
500	0.95	3	2303	1	11.647	0.002
100	0.8	1	39278	5403	45.865	0.089
100	0.8	2	39278	5403	29.119	0.115
100	0.8	3	39278	5403	52.516	0.097
100	0.9	1	39278	768	49.398	0.075
100	0.9	2	39278	768	29.079	0.074

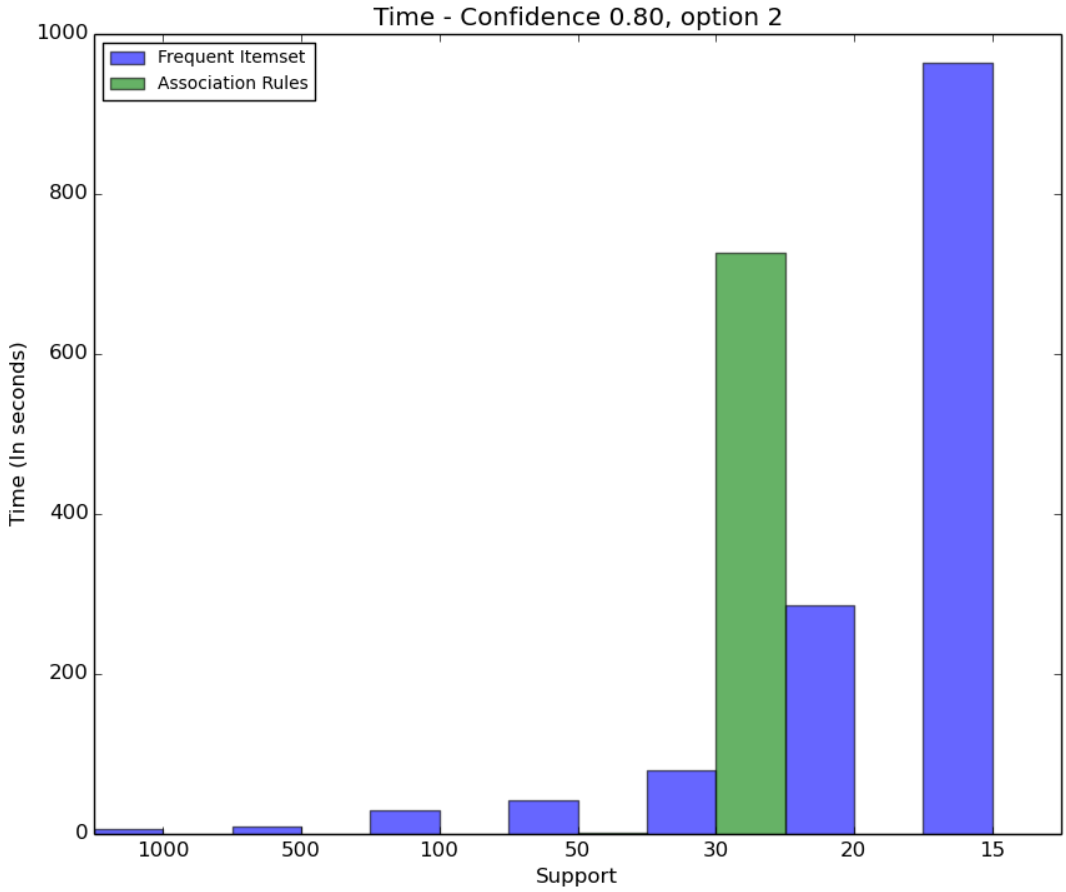
100	0.9	3	39278	768	49.247	0.074
100	0.95	1	39278	197	49.332	0.074
100	0.95	2	39278	197	29.186	0.073
100	0.95	3	39278	197	51.529	0.071
50	0.8	1	157915	329304	88.56	2.206
50	0.8	2	157915	329304	41.578	2.229
50	0.8	3	157915	329304	96.705	2.222
50	0.9	1	157915	160077	86.481	1.485
50	0.9	2	157915	160077	40.92	1.434
50	0.9	3	157915	160077	96.665	1.491
50	0.95	1	157915	65679	88.386	0.94
50	0.95	2	157915	65679	41.561	0.921
50	0.95	3	157915	65679	91.41	0.883
30	0.8	1	1197364	57726221	158.744	711.85
30	0.8	2	1197364	57726221	80.445	725.987
30	0.8	3	1197364	57726221	183.519	740.434
30	0.9	1	1197364	26168852	159.376	366.808
30	0.9	2	1197364	26168852	79.989	364.718
30	0.9	3	1197364	26168852	183.797	374.042
30	0.95	1	1197364	11569806	159.297	176.981
30	0.95	2	1197364	11569806	80.546	180.643
30	0.95	3	1197364	11569806	180.938	182.318
20	-1	1	12917208	0	509.877	0
20	-1	2	12917208	0	285.892	0
20	-1	3	12917208	0	664.45	0
15	-1	1	71274569	0	1695.353	0
15	-1	2	71274569	0	964.362	0
15	-1	3	71274569	0	1897.934	0

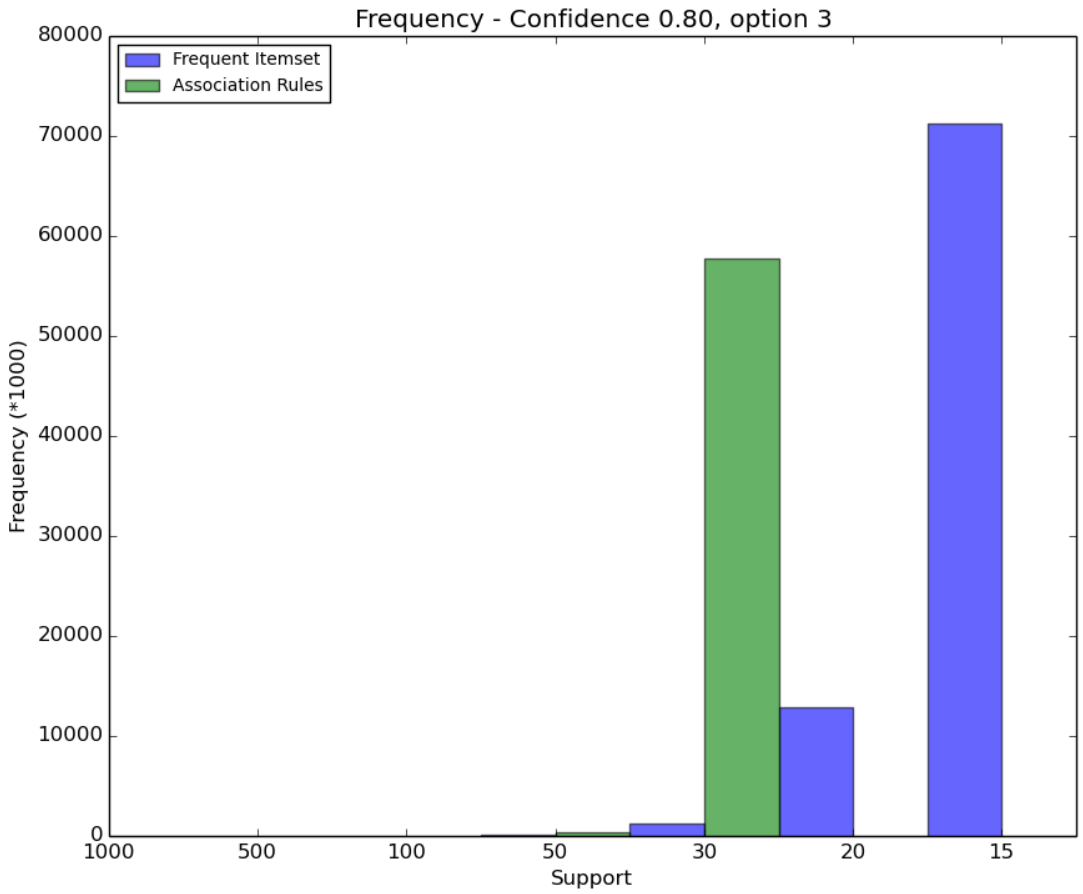

Frequency - Confidence 0.80, option 1
Frequent Itemset
Association Rules
Frequency (*1000)
80000
70000
60000
50000
40000
30000
20000
10000
0
1000
500
100
50
30
20
15
Support

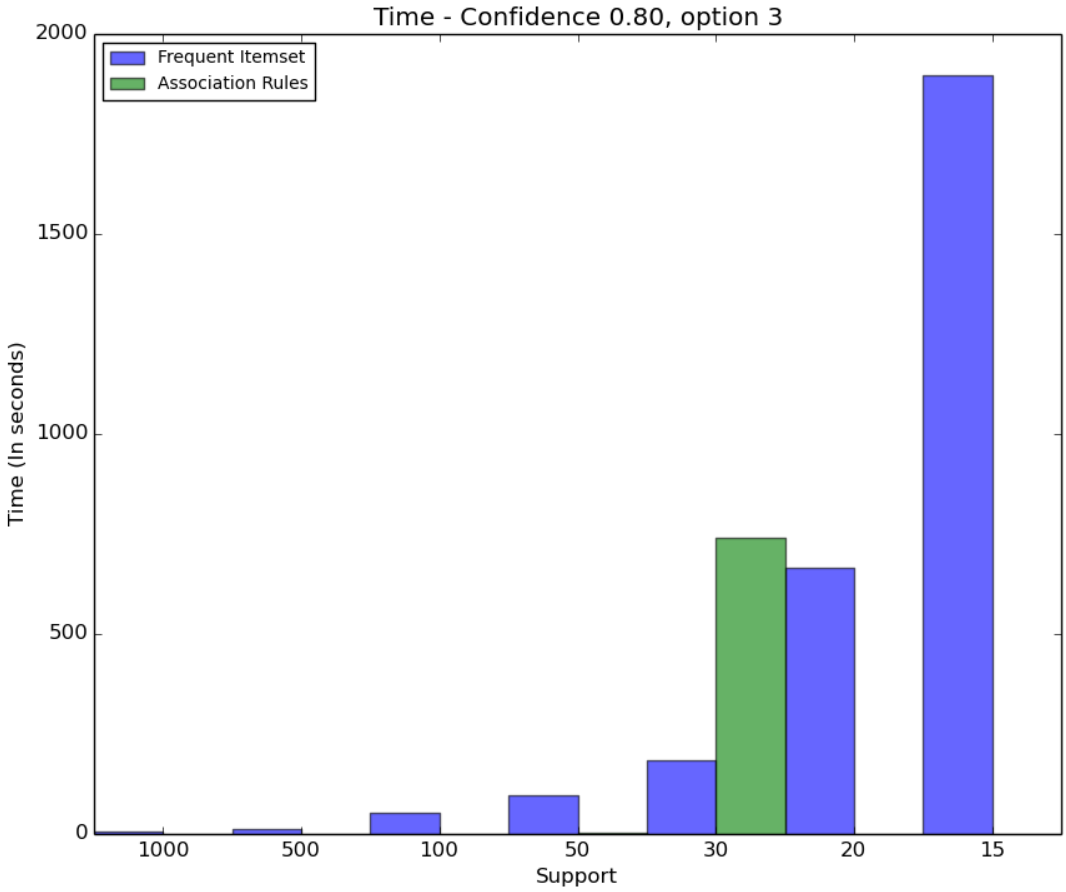

Time - Confidence 0.80, option 1
Frequent Itemset
Association Rules
Time (In seconds)
0
200
400
600
800
1000
1200
1400
1600
1800
1000
500
100
50
30
20
15
Support

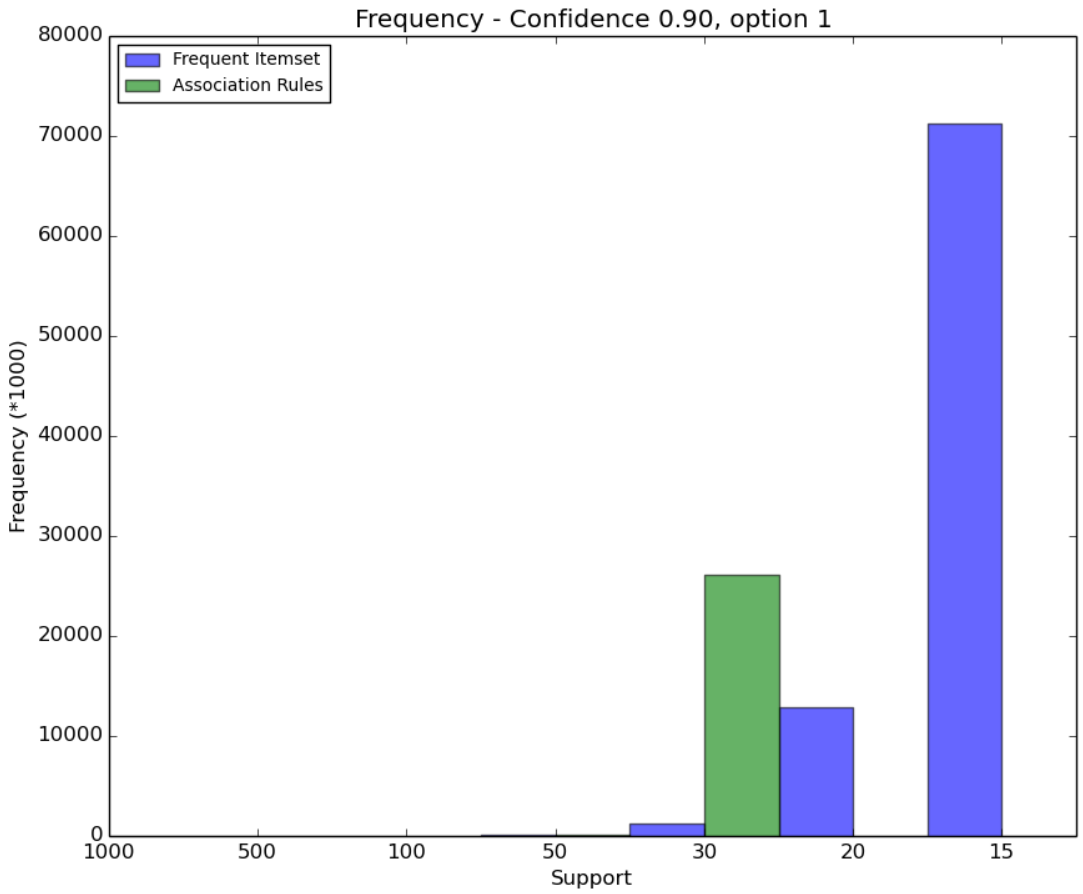
vc

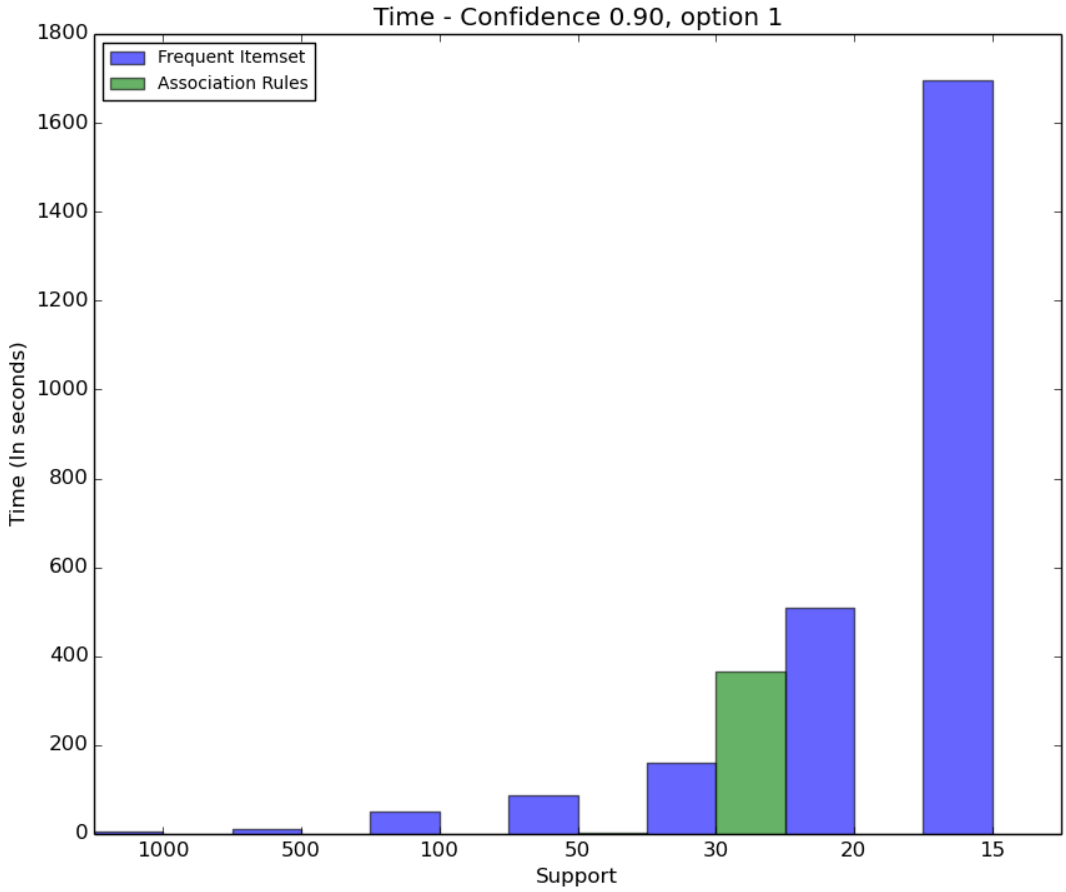

Frequency - Confidence 0.80, option 2
Frequent Itemset
Association Rules
Frequency (*1000)
80000
70000
60000
50000
40000
30000
20000
10000
0
1000
500
100
50
30
20
15
Support

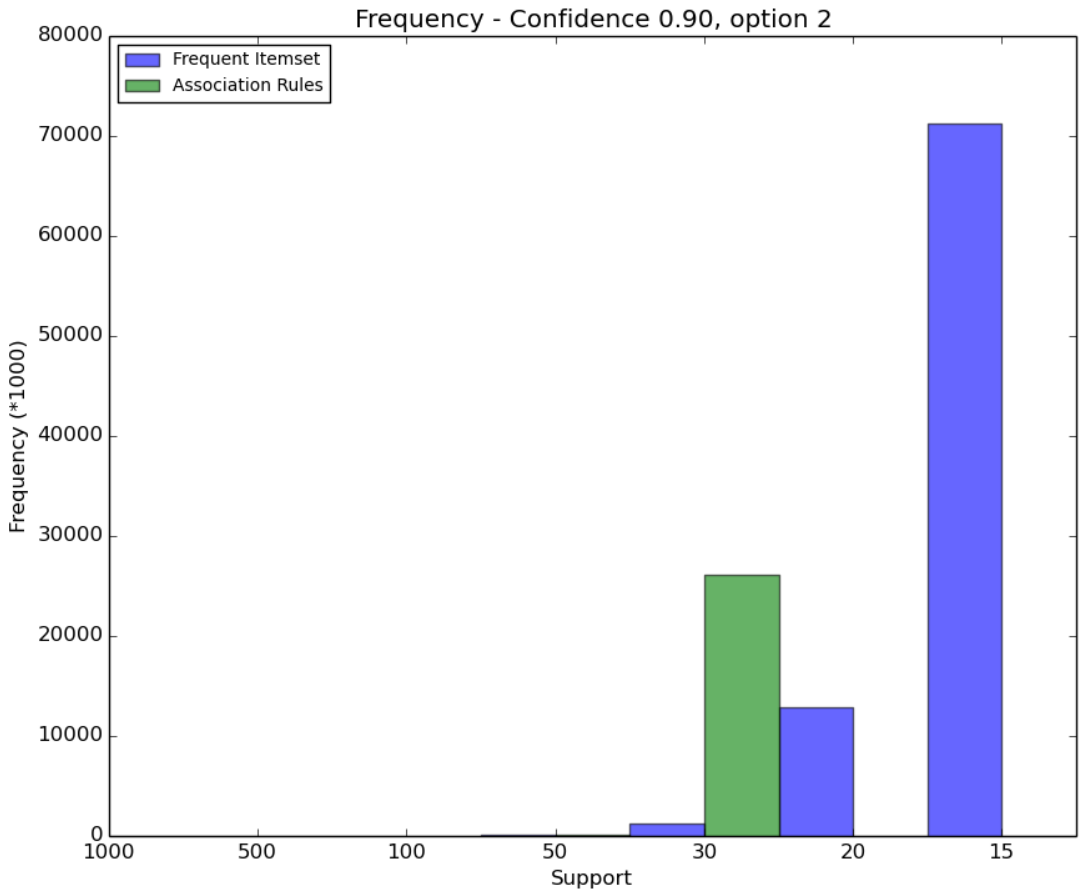

Time - Confidence 0.80, option 2
Frequent Itemset
Association Rules
Time (In seconds)
1000
800
600
400
200
0
1000
500
100
50
30
20
15
Support

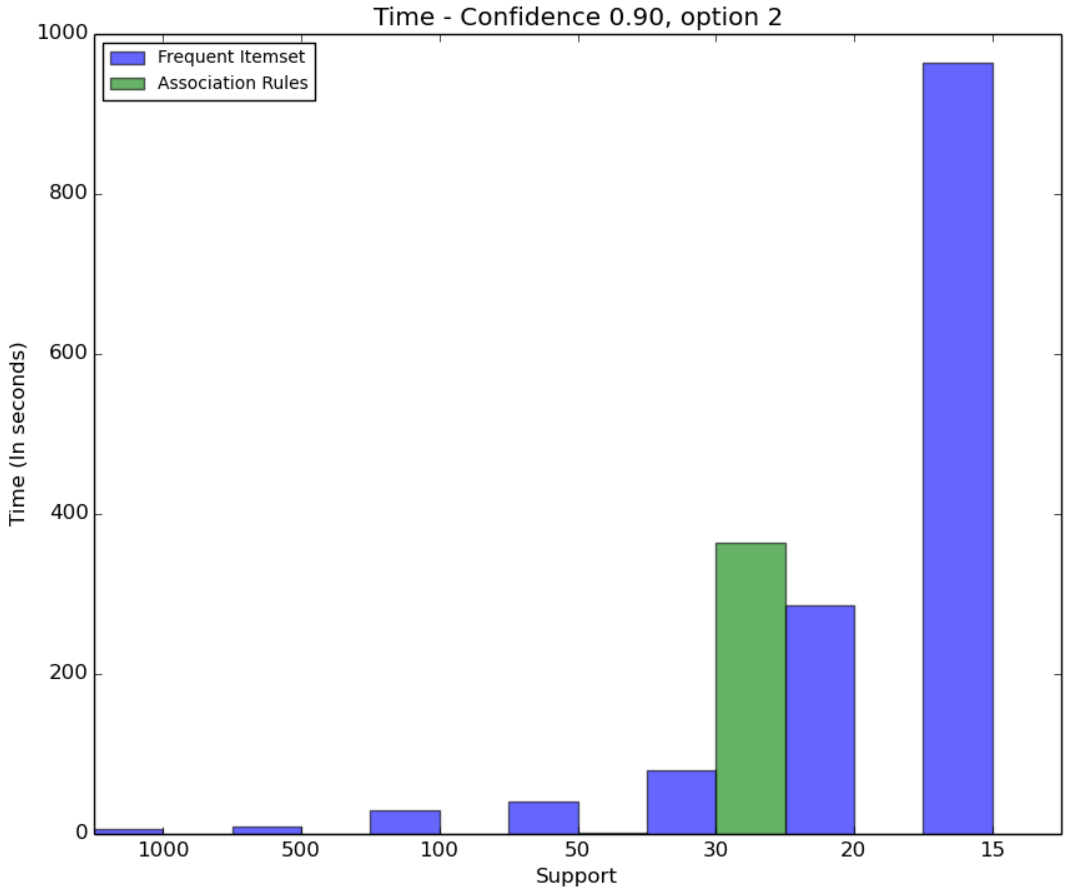

Frequency - Confidence 0.80, option 3
Frequent Itemset
Association Rules
Frequency (*1000)
80000
70000
60000
50000
40000
30000
20000
10000
0
1000
500
100
50
30
20
15
Support

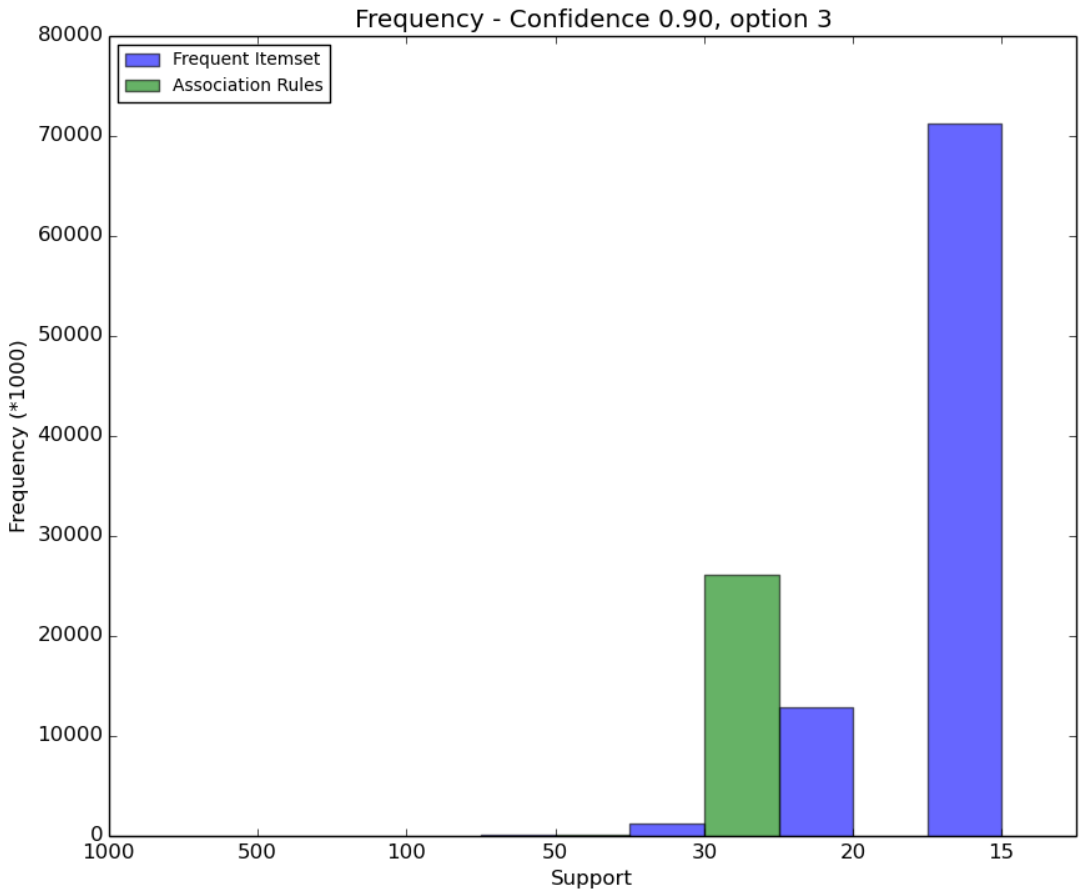

Time - Confidence 0.80, option 3
Frequent Itemset
Association Rules
Time (In seconds)
2000
1500
1000
500
0
1000
500
100
50
30
20
15
Support

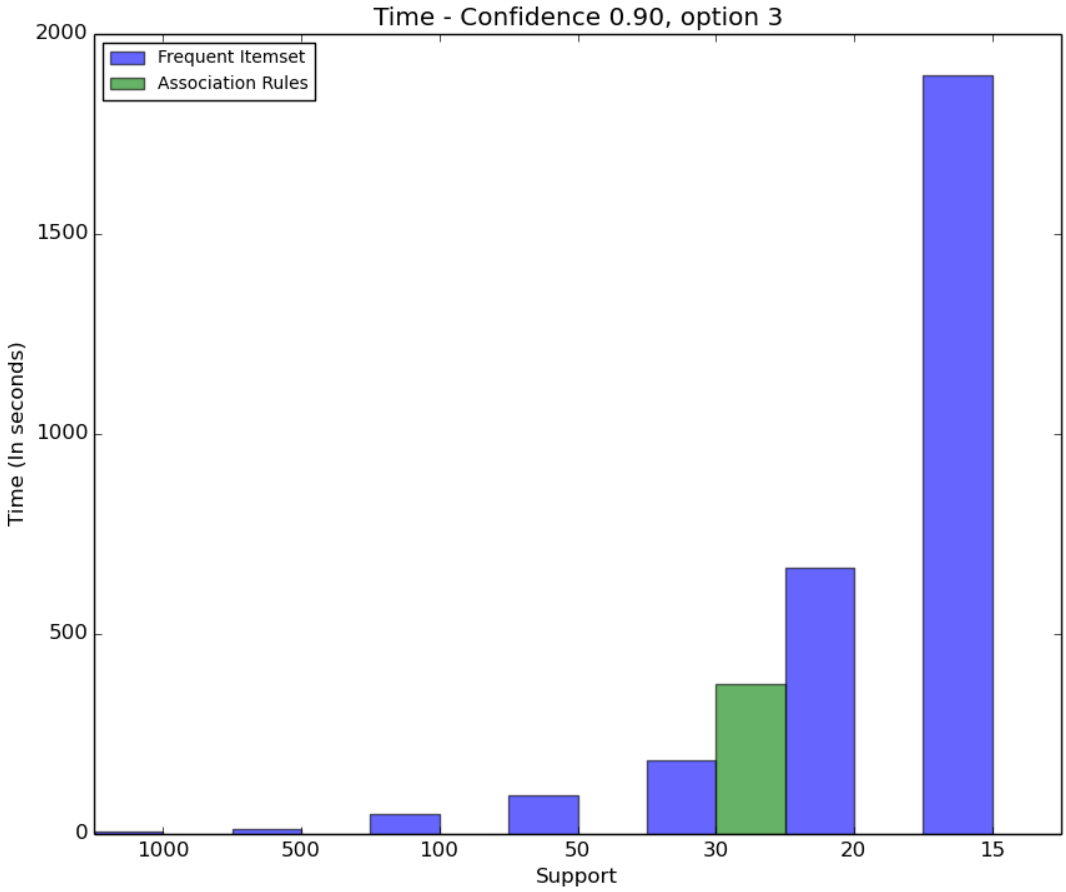

Frequency - Confidence 0.90, option 1
Frequent Itemset
Association Rules
Frequency (*1000)
80000
70000
60000
50000
40000
30000
20000
10000
0
1000
500
100
50
30
20
15
Support

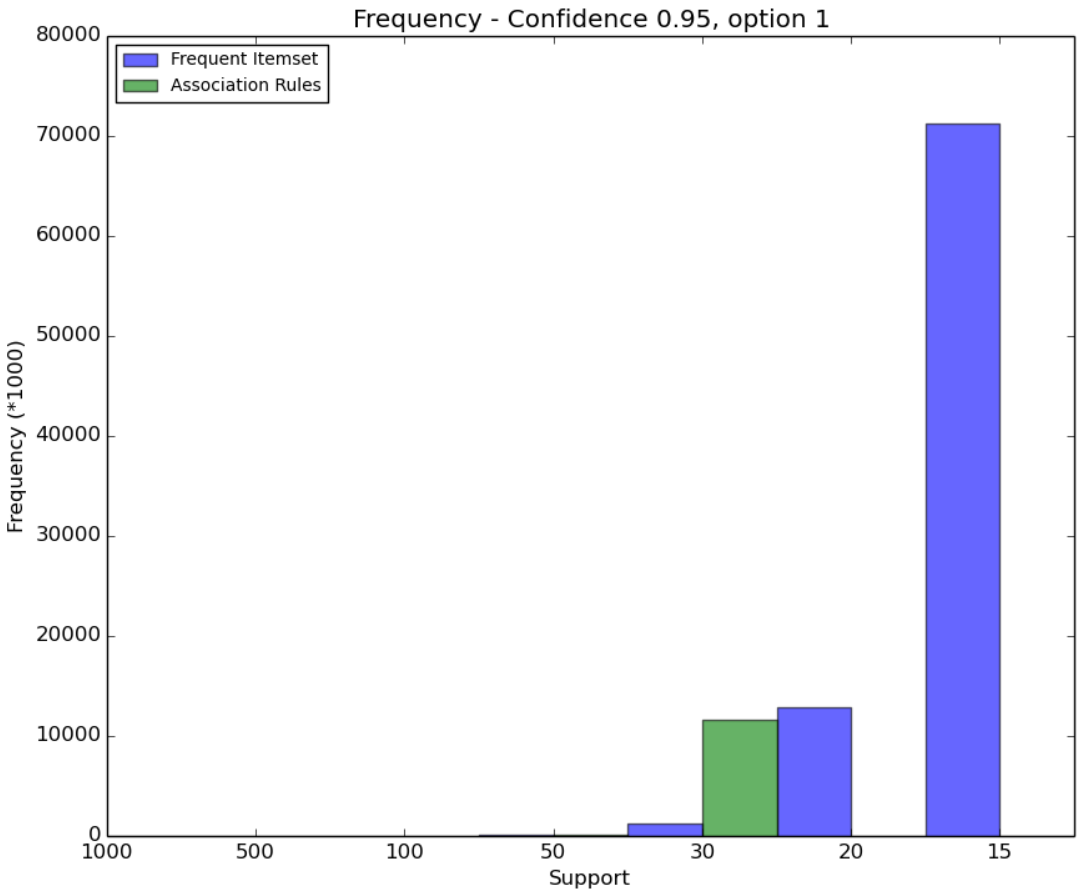

Time - Confidence 0.90, option 1
Frequent Itemset
Association Rules
Time (In seconds)
1800
1600
1400
1200
1000
800
600
400
200
0
1000
500
100
50
30
20
15
Support

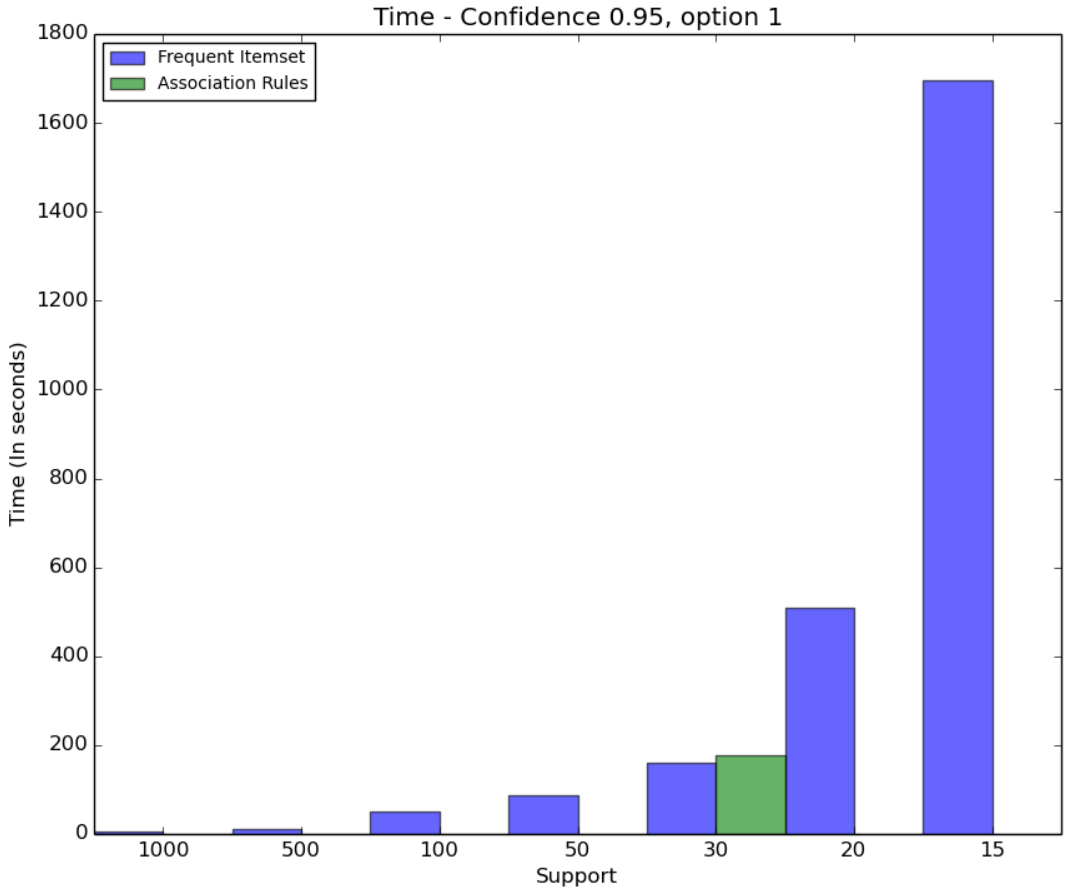

Frequency - Confidence 0.90, option 2
Frequent Itemset
Association Rules
Frequency (*1000)
80000
70000
60000
50000
40000
30000
20000
10000
0
1000
500
100
50
30
20
15
Support

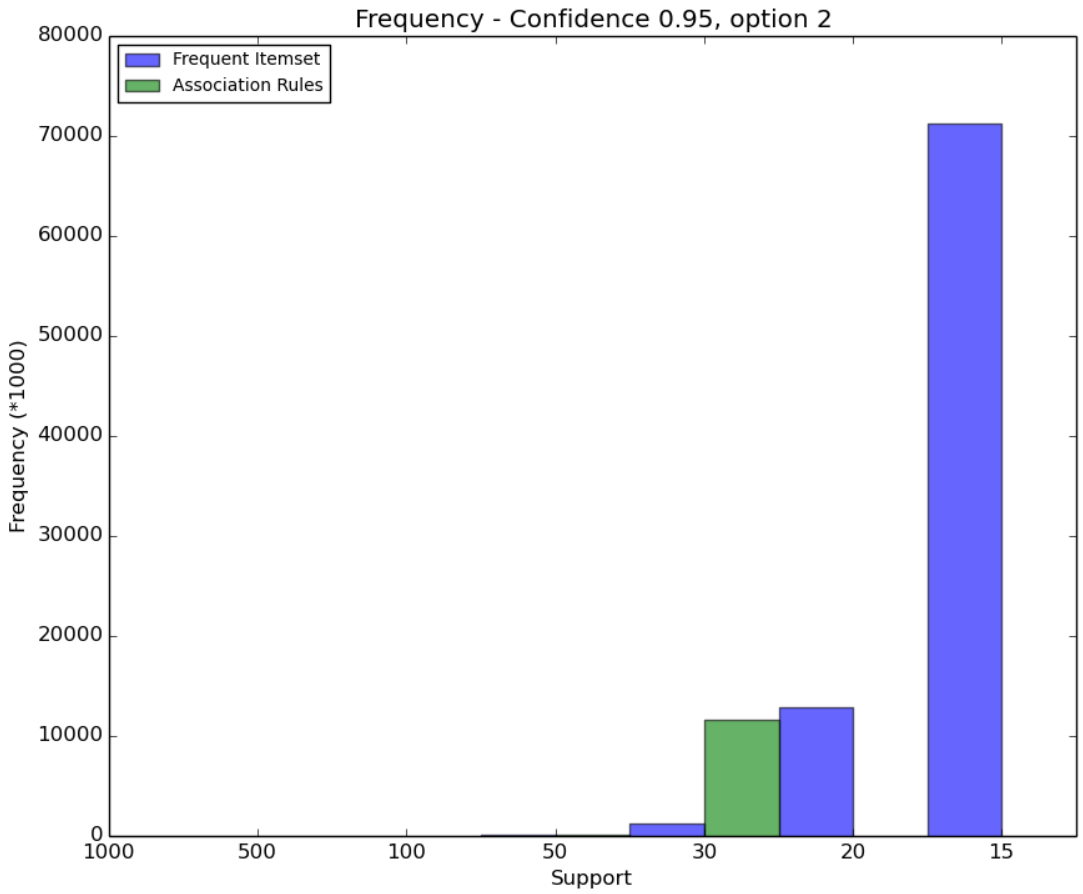

Time - Confidence 0.90, option 2
Frequent Itemset
Association Rules
Time (In seconds)
1000
800
600
400
200
0
1000
500
100
50
30
20
15
Support

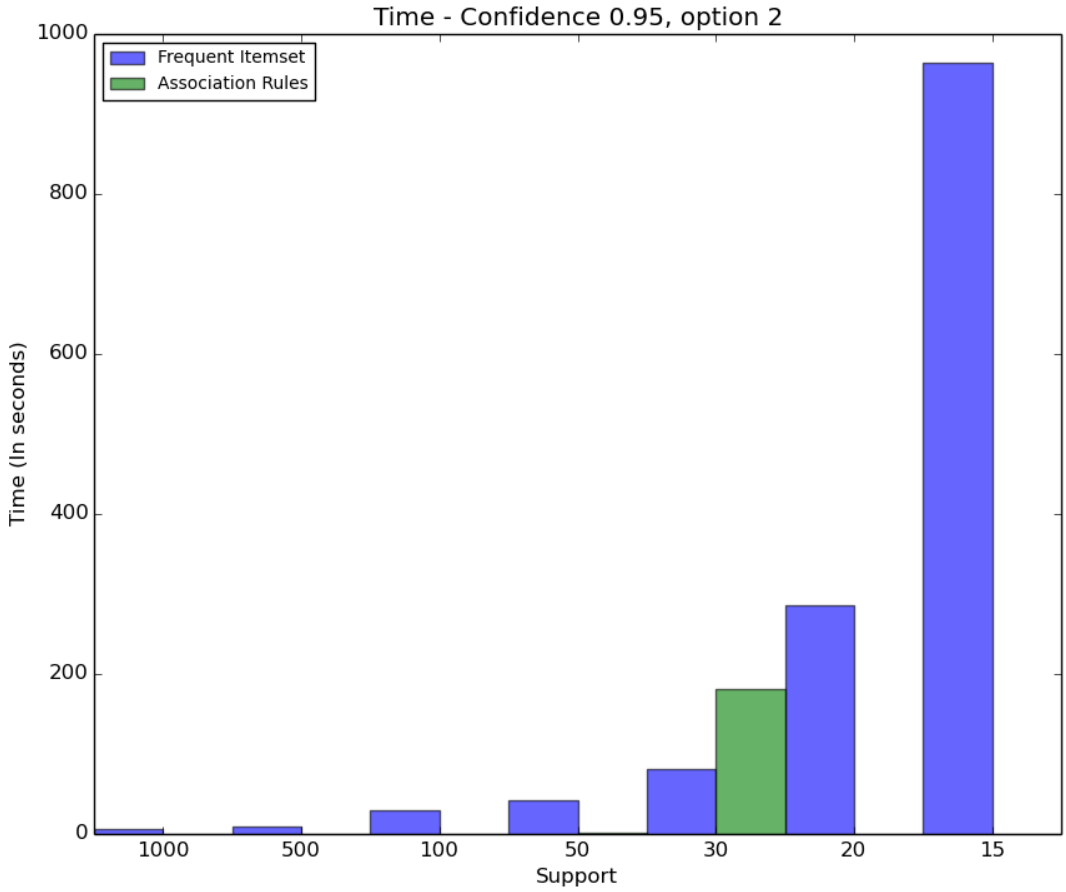

Frequency - Confidence 0.90, option 3
Frequent Itemset
Association Rules
Frequency (*1000)
80000
70000
60000
50000
40000
30000
20000
10000
0
1000
500
100
50
30
20
15
Support

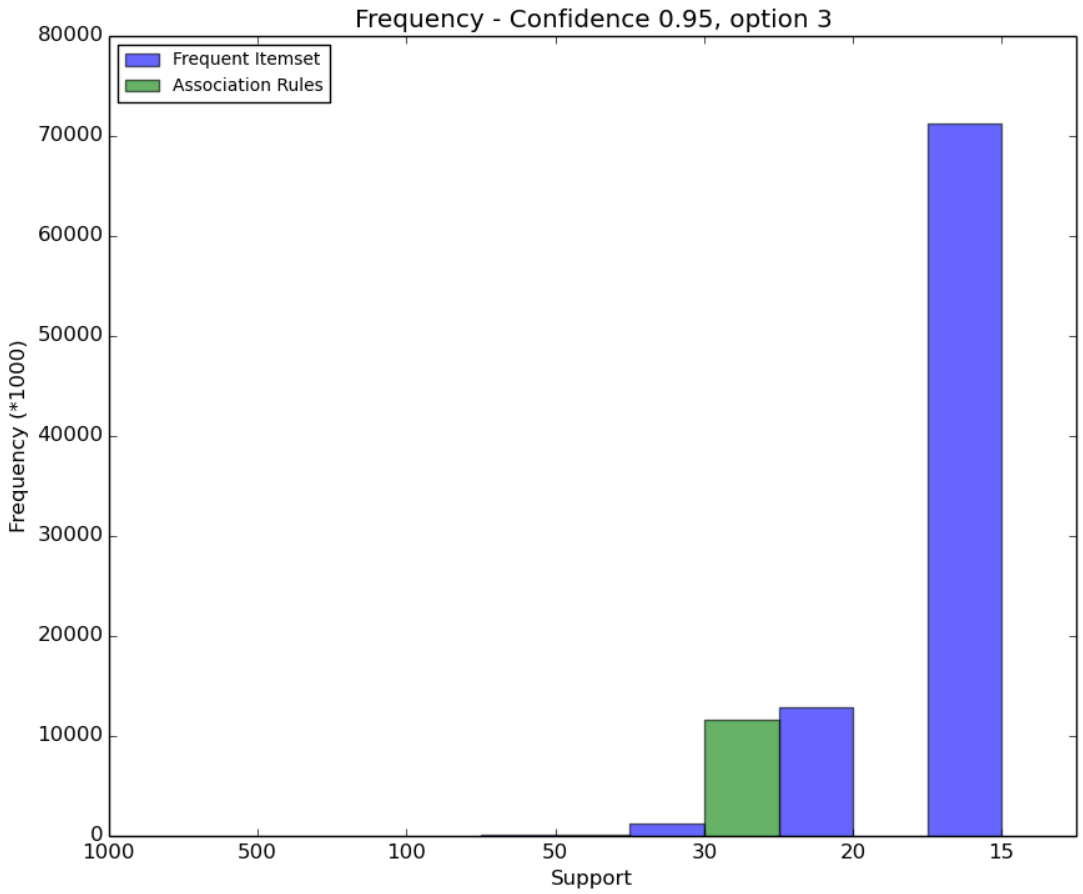

Time - Confidence 0.90, option 3
Frequent Itemset
Association Rules
Time (In seconds)
2000
1500
1000
500
0
1000
500
100
50
30
20
15
Support

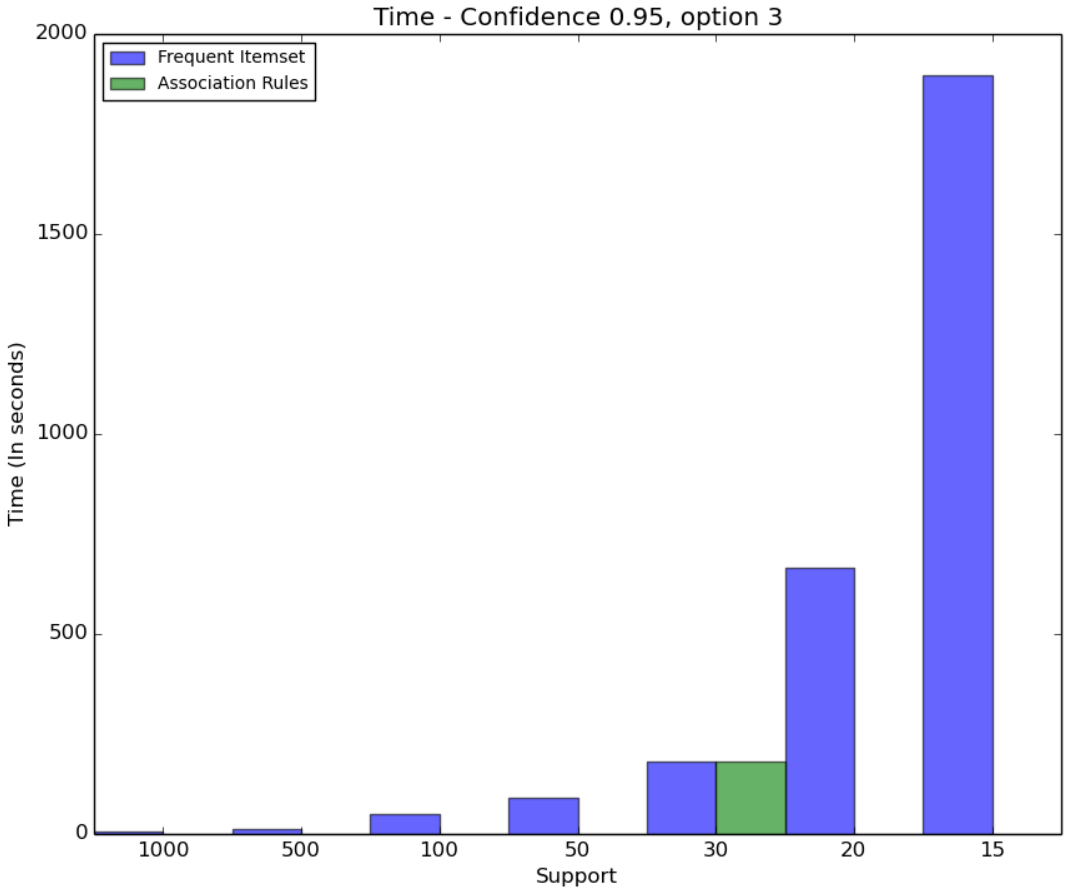

Frequency - Confidence 0.95, option 1
Frequent Itemset
Association Rules
Frequency (*1000)
80000
70000
60000
50000
40000
30000
20000
10000
0
1000
500
100
50
30
20
15
Support


Time - Confidence 0.95, option 1
Frequent Itemset
Association Rules
Time (In seconds)
1800
1600
1400
1200
1000
800
600
400
200
0
1000
500
100
50
30
20
15
Support


Frequency - Confidence 0.95, option 2
Frequent Itemset
Association Rules
Frequency (*1000)
80000
70000
60000
50000
40000
30000
20000
10000
0
1000
500
100
50
30
20
15
Support


Time - Confidence 0.95, option 2
Frequent Itemset
Association Rules
Time (In seconds)
1000
800
600
400
200
0
1000
500
100
50
30
20
15
Support


Frequency - Confidence 0.95, option 3
Frequent Itemset
Association Rules
Frequency (*1000)
80000
70000
60000
50000
40000
30000
20000
10000
0
1000
500
100
50
30
20
15
Support


Time - Confidence 0.95, option 3
Frequent Itemset
Association Rules
2000
1500
1000
500
0
Time (In seconds)
1000
500
100
50
30
20
15
Support