REPORT

By: Jagdeep Singh

2017csb1080

All Dataset details:

1. T40I10D100K : Total number of transactions: 9021
   Average Width of transactions:39.53
   Size of maximal frequent itemset for minsup(0.01): 17
   Number of maximal frequent itemsets: 1
   Total number of items: 356652

2. T10I4D100K : Total number of transactions: 81319
   Average Width of transactions: 10.1
   Size of maximal frequent itemset for minsup(0.01): 3
   Number of maximal frequent itemsets: 1
   Total number of items: 821886

3. retail : Total number of transactions: 14532
   Average Width of transactions: 10.16
   Size of maximal frequent itemset for minsup(0.01): 4
   Number of maximal frequent itemsets: 8
   Total number of items: 147770

4. groceries – groceriers.csv : Total number of transactions: 9835
   Average Width of transactions: 4.4
   Size of maximal frequent itemset for minsup(0.01): 3
   Number of maximal frequent itemsets: 32
   Total number of items: 43274(excluding first column of every transaction as it is just number of items in transaction and excluding first row which is just for making dataset convenient)

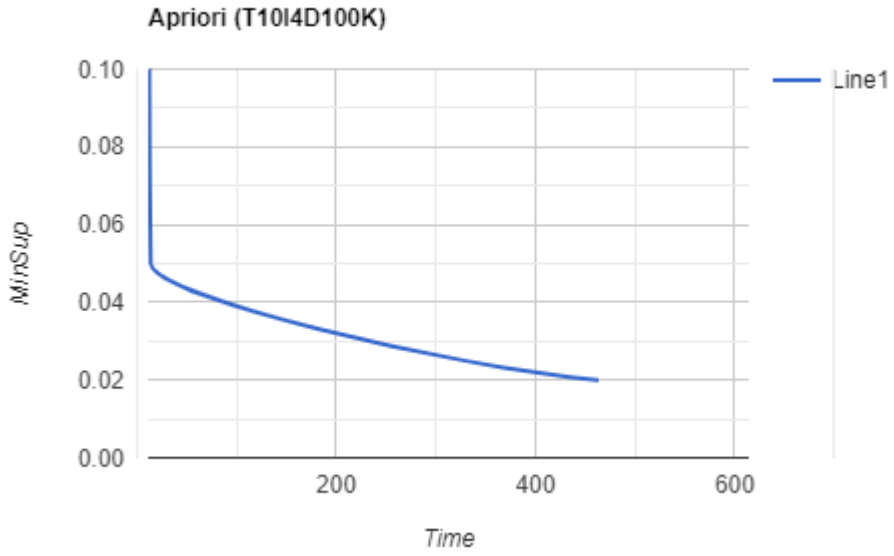
5. groceries.csv : Total number of transactions: 9835
   Average Width of transactions: 32
   Size of maximal frequent itemset for minsup(0.01): 3

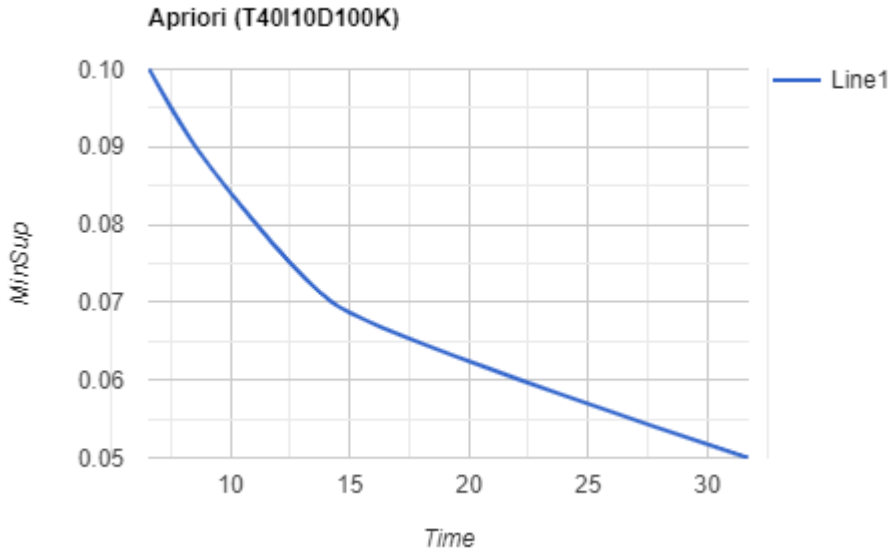
Number of maximal frequent itemsets: 32
Total number of items: 43367

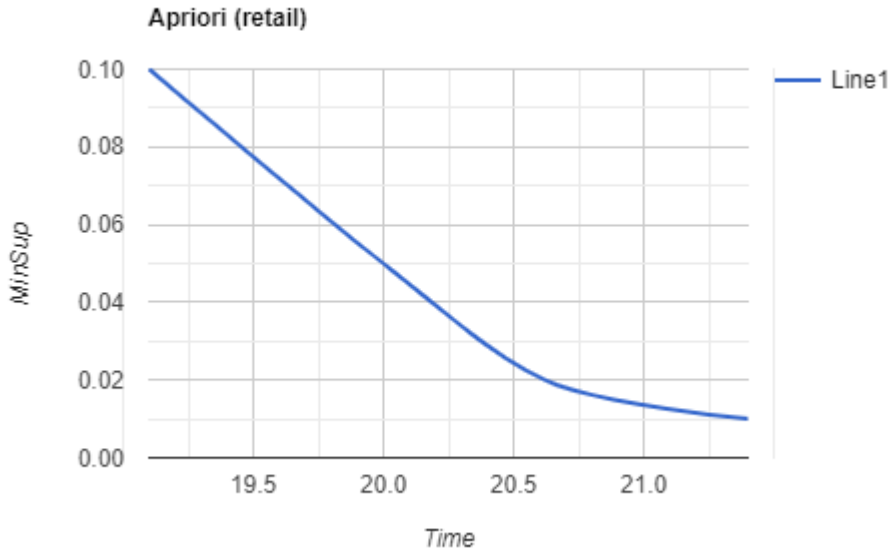
6. dataset (made as per 2nd part of assignment):
   Total number of transactions: 10000
   Average Width of transactions: 32
   Size of maximal frequent itemset for minsup(0.01): 1
   Number of maximal frequent itemsets: 1000
   Total number of items: 314720

NOTE: 6th dataset used in each algorithm is generated with 10000 transactions of 30 average width and 1000 as upper limit for value of item.

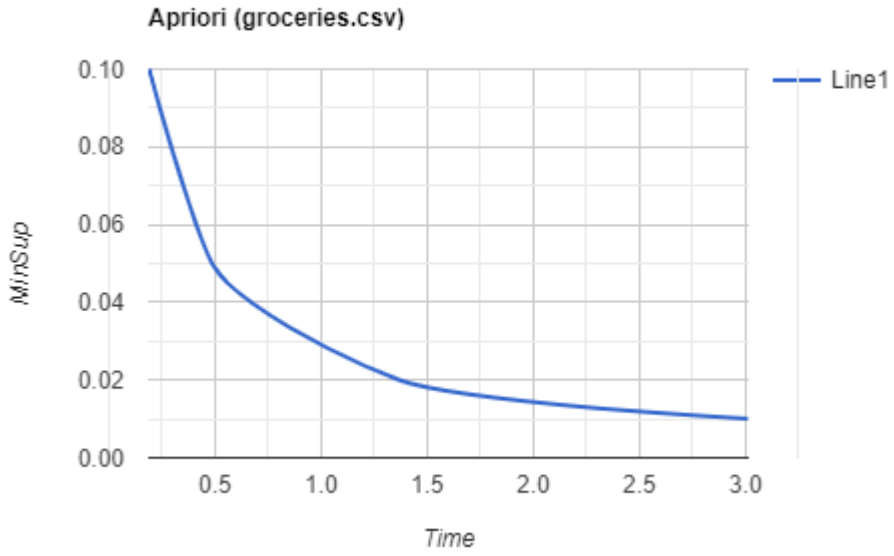
APRIORI Algorithm:

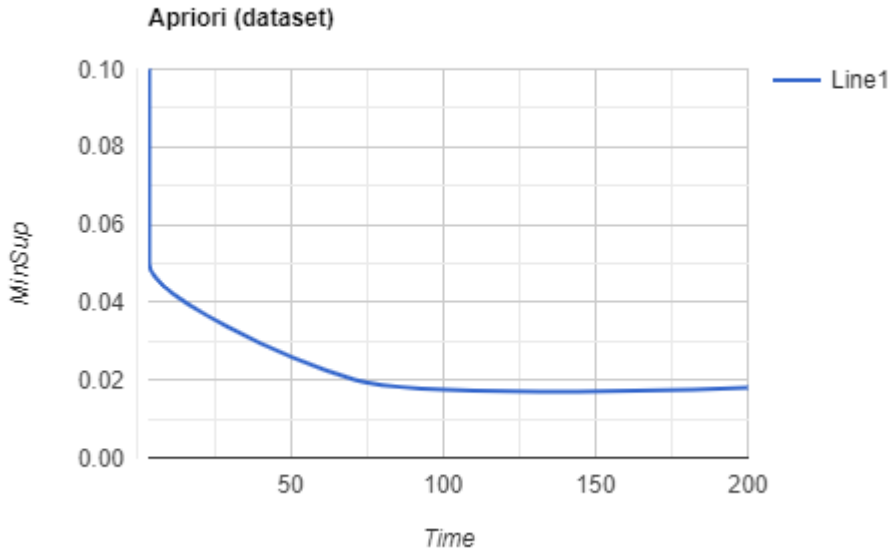

Apriori (T10I4D100K)
0.10
0.08
0.06
0.04
0.02
0.00
MinSup
200
400
600
Time
Line1


Apriori (T40I10D100K)
0.10
0.09
0.08
0.07
0.06
0.05
MinSup
10
15
20
25
30
Time
Line1


Apriori (retail)
0.10
0.08
0.06
0.04
0.02
0.00
MinSup
19.5
20.0
20.5
21.0
Time
Line1

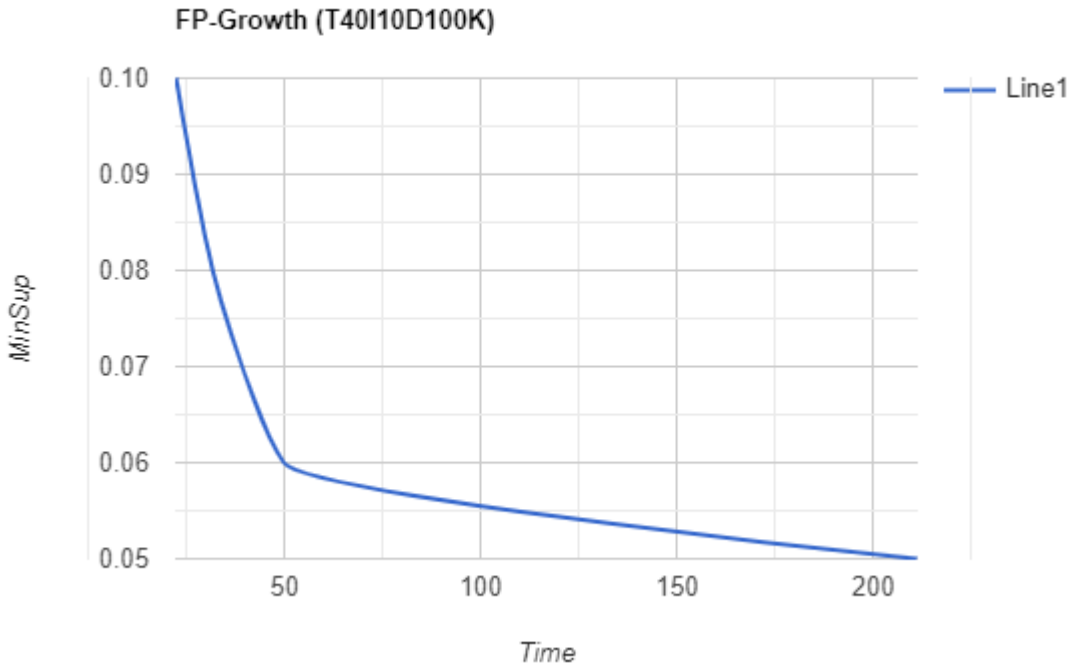

Apriori (groceries - groceries.csv)
0.10
0.08
0.06
0.04
0.02
0.00
MinSup
2
3
4
Time
Line1

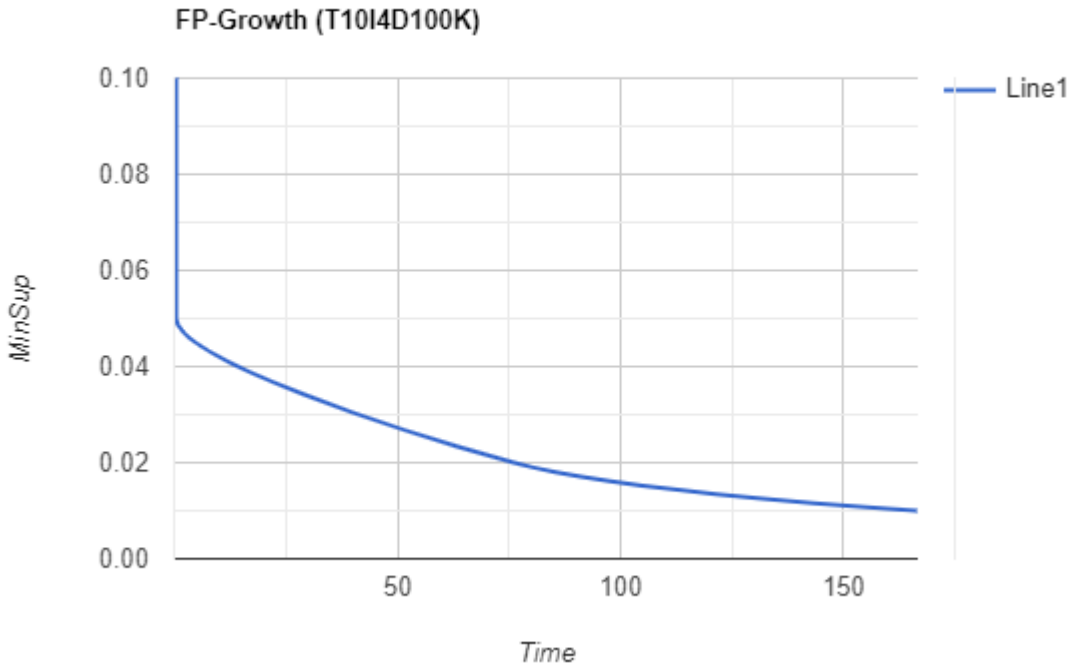


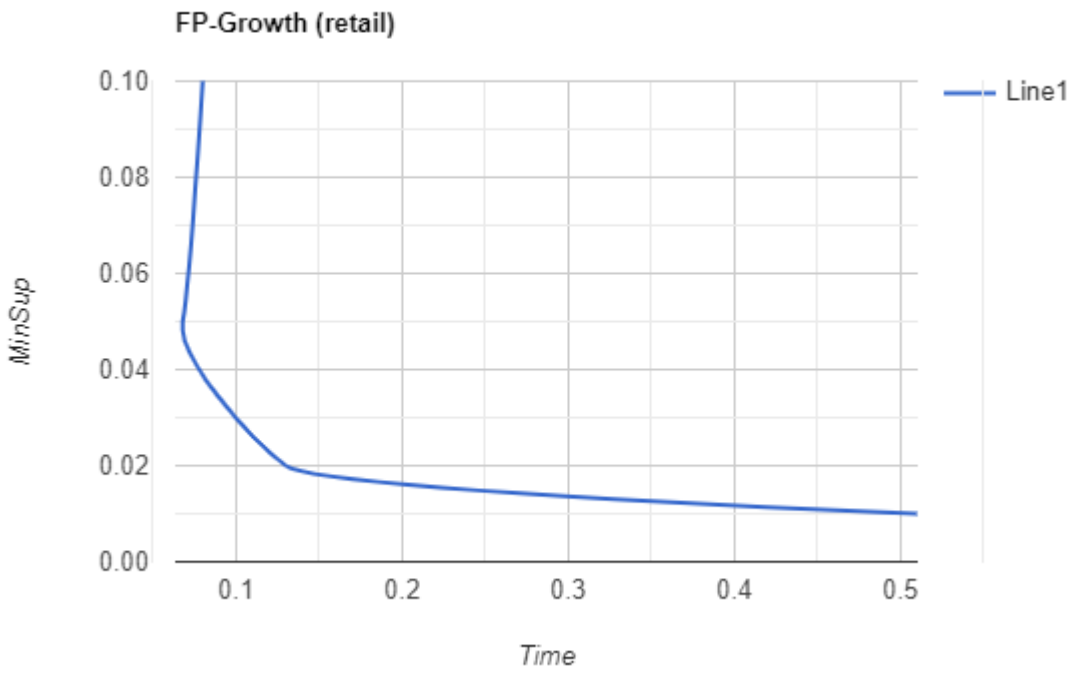


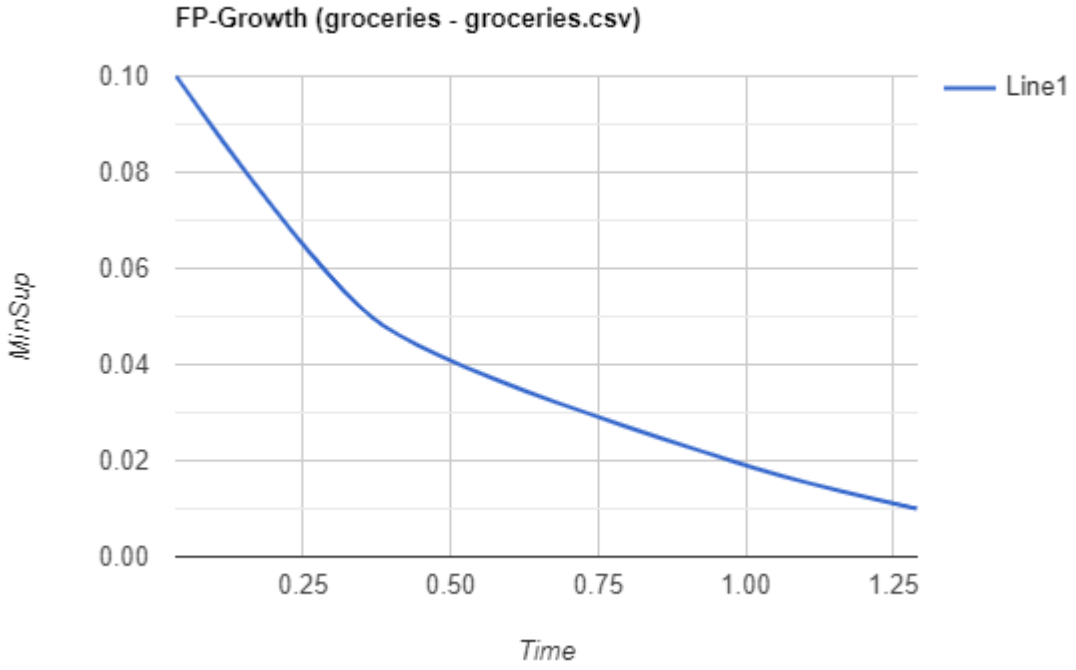
FP Growth Algorithm: at first call createTree function which will further call to a function for creating a header table, after which pruning step is called. We return this tree and call mining function

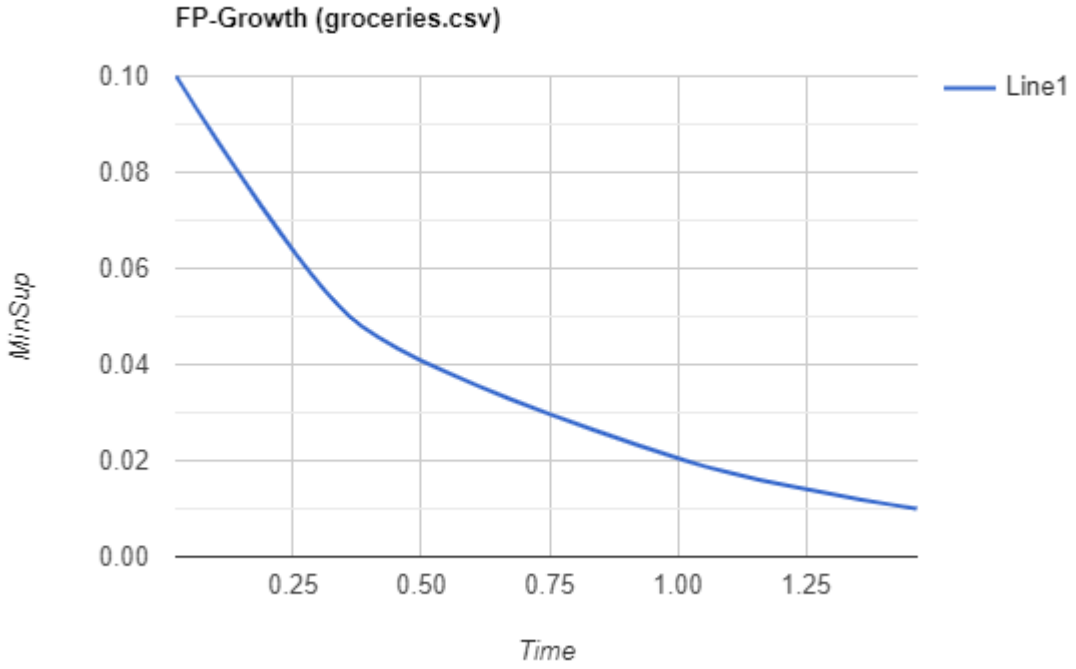
which is most crucial step for finding frequent itemsets, in that we find prefix paths and then conditional trees for each entry in headertable we returned from our initial call to createTree. As long as headertable which is returned from createTree function is not empty it will recursively keep calling mining function and when prefixpath is used the items in headertable will keep reducing until all the elements are covered.

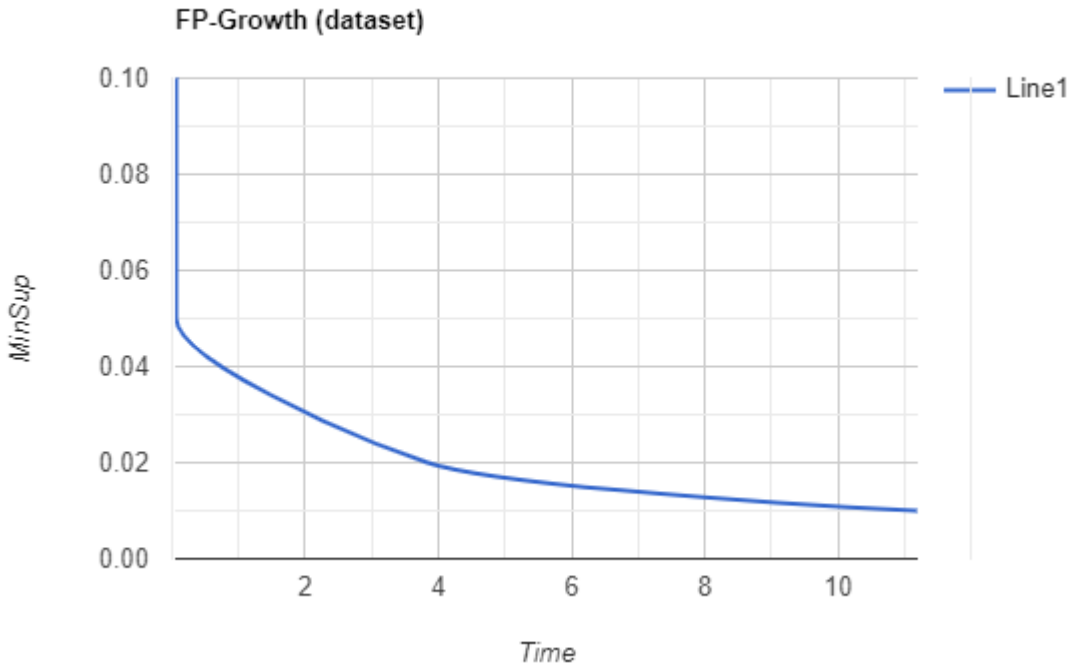



FP-Growth (T10I4D100K)
0.10
0.08
0.06
0.04
0.02
0.00
MinSup
50
100
150
Time
Line1

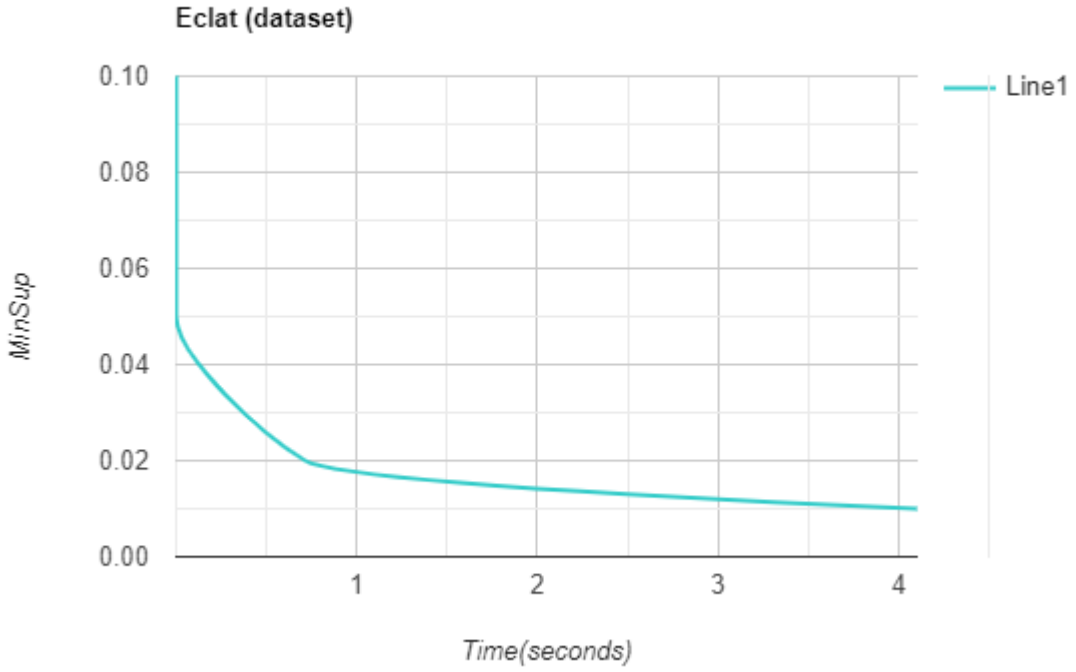

FP-Growth (retail)
0.10
0.08
0.06
0.04
0.02
0.00
MinSup
0.1
0.2
0.3
0.4
0.5
Time
Line1

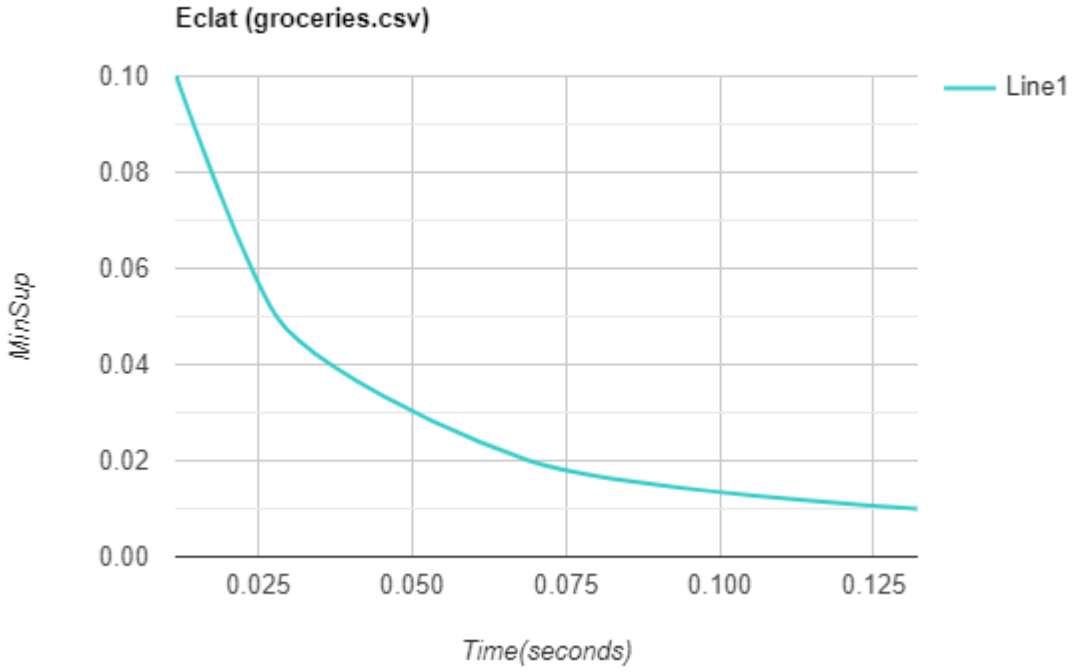

FP-Growth (groceries - groceries.csv)
0.10
0.08
0.06
0.04
0.02
0.00
MinSup
0.25
0.50
0.75
1.00
1.25
Time
Line1

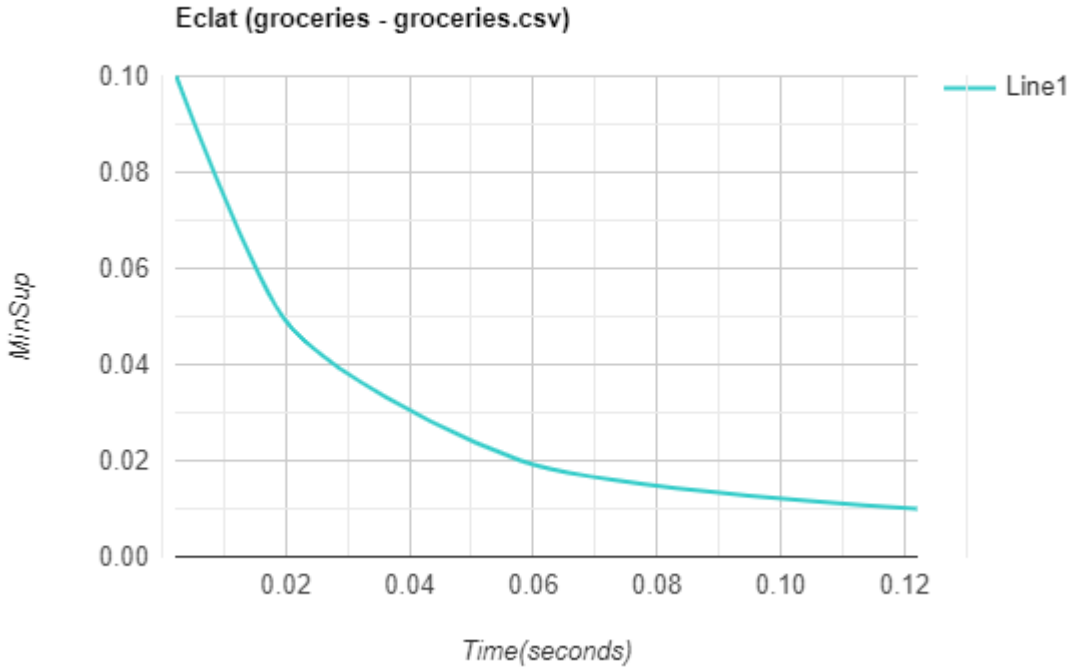

FP-Growth (groceries.csv)
0.10
0.08
0.06
0.04
0.02
0.00
MinSup
0.25
0.50
0.75
1.00
1.25
Time
Line1

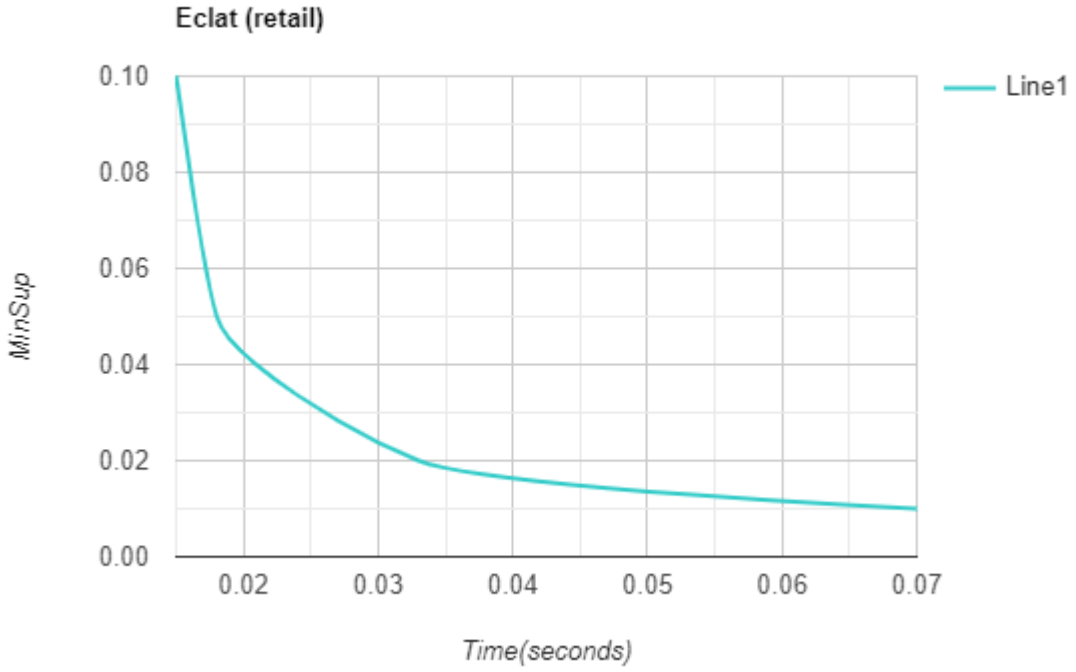


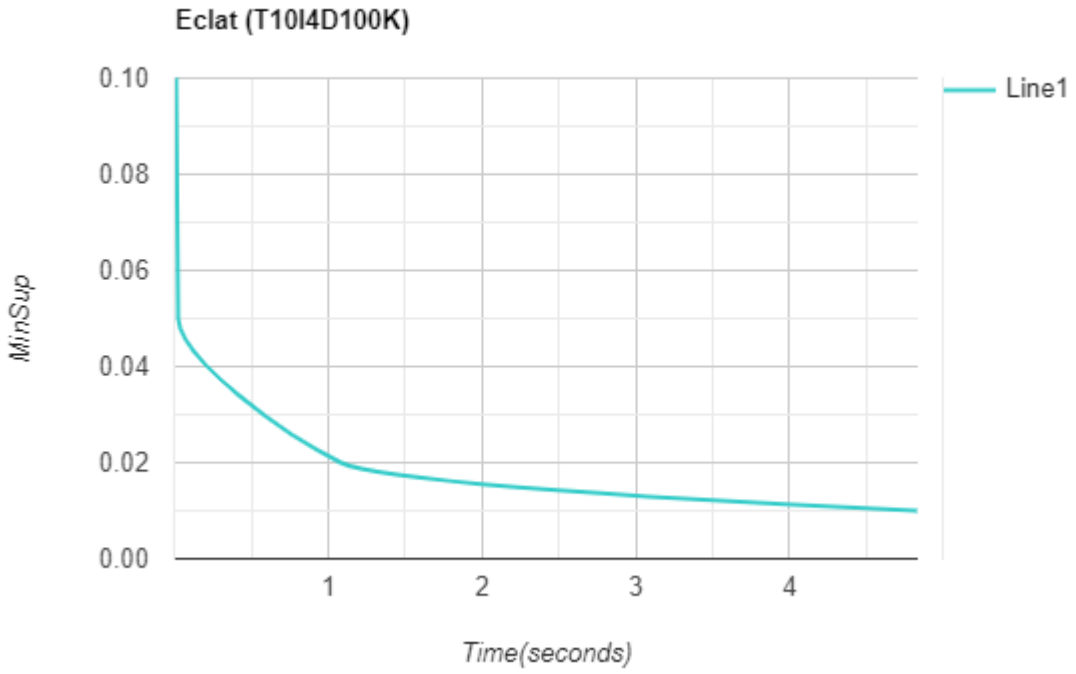
ECLAT Algorithm: In our first call to eclat function all single items will be sent , and in there it will check for which combinations of items their corresponding transaction sets have atleast length equal to or greater than min support. And it will recursively call eclat function for newly generated itemsets.

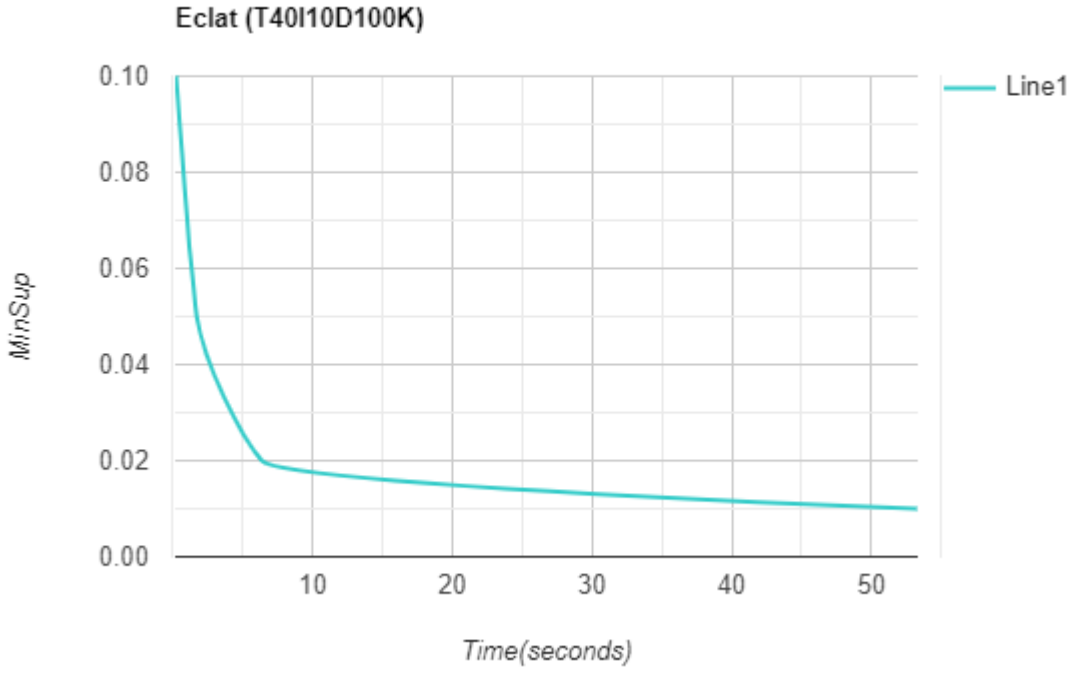

Eclat (dataset)
0.10
0.08
0.06
0.04
0.02
0.00
MinSup
1
2
3
4
Time(seconds)
Line1


Eclat (groceries.csv)
0.10
0.08
0.06
0.04
0.02
0.00
MinSup
0.025
0.050
0.075
0.100
0.125
Time(seconds)
Line1


Eclat (groceries - groceries.csv)
0.10
0.08
0.06
0.04
0.02
0.00
MinSup
0.02
0.04
0.06
0.08
0.10
0.12
Time(seconds)
Line1


Eclat (retail)
0.10
0.08
0.06
0.04
0.02
0.00
MinSup
0.02
0.03
0.04
0.05
0.06
0.07
Time(seconds)
Line1


Eclat (T10I4D100K)
0.10
0.08
0.06
0.04
0.02
0.00
MinSup
1
2
3
4
Time(seconds)
Line1


Eclat (T40I10D100K)
0.10
0.08
0.06
0.04
0.02
0.00
MinSup
10
20
30
40
50
Time(seconds)
Line1

Part 2:

To generate a dataset User have to ender total number of transactions, average width, upper limit for item value(number). Algorithm is very simple, firstly I am generating random number depicting number of items in each transaction and the sum of all these random numbers generated must be exactly number of transaction*average width of transaction. Then for each transaction I generate the number of items corresponding to the number generated in previous step. This way number of items in transactions will be close to average width of transaction.

This is line graph for dataset with 7000 transactions and average width as 10 and 100 as upper limit of each item value. As we are extending upper limit or average width Apriori algorithm takes very long time which is not even able to complete program before crashing.