**Apriori:**

Time to complete: 1:14:18.724000

Frequent sets found (min\_support – 600): 772

The time complexity of an Apriori algorithm is where d is the number of unique items. Based on the time complexity, the amount of time required to run the algorithm increases significantly as the number of unique items increases.

This can be demonstrated as the code had a very long run time. With a min\_support of 100, the number specified in the assignment, the script took multiple hours to run. The run time was multiple magnitudes larger when compared to FP-Growth or Relim. The script continued to run until the program eventually resulted in a memory error and exited.

In order to avoid reaching a memory error, I increased the minimum support to 0.6% (600). This resulted in a run time of about an hour and 14 minutes.

**FP-Growth:**

Time to complete: 0:04:46.574000

Frequent sets found (min\_support – 100): 27532

The time complexity of FP-Growth algorithm is where d is the number of unique items. As we can see, the run time is significantly faster than the Apriori method. The run time increase is significantly smaller as we increase the number of unique items. This is because the FP-Growth algorithm does not need to search through the entire transaction list every time it checks for frequency. The algorithm only required 5 minutes to find all the frequent sets in 100000 transactions.

**Relim:**

Time to complete: 0:00:49.609000

Frequent sets found (min\_support – 100): 27532

The fasted of all 3 methods, required less than a minute to find all the frequent sets.

Relim recursively creates an FP-Growth Tree, requiring less runs to generate the tree. Thus it is more time efficient compared to a standard FP-Growth algorithm.

**Comparison:**

All of the run times are dependent on the amount of times the algorithm needs to search through the transaction list. The less searching required the lower the run time. FP-Growth and Relim reduce the number of searches by creating a tree thus reducing the search breadth. The Apriori algorithm however, not only searches more often, but also needs to run through the entire transaction list each search. This is why the time complexity of FP-Growth and Relim is much lower.

**Closed/Maximum:**

Frequent Sets: 27532

Closed Sets: 26806

Maximum Sets: 4054

**Notes:**

All of the code written is in Python 2.

All of the python code was written with the assumption that the input dataset “freq\_items\_dataset” is in the same directory as the python script.