DSC 440-Frequent Itemset Mining Project Report

Ezgi Siir Kibris

1 Introduction

Apriori and FP-growth are two popular frequent itemset mining algorithms. This report compares performances of Apriori and FP-growth algorithms. In general, FP-growth performs better than Apriori. FP-Growth is more scalable and efficient. Apriori has considerably larger memory needs than FP-growth. In addition to that, run time of Apriori is much more than FP-growth.

I cannot run whole adult dataset with my Apriori algorithm because it uses almost all of my computer's RAM memory. Whereas I can run FP-growth with the all adult dataset. It takes around 13-14 seconds to run the whole dataset with FP-growth.

2 Testing Apriori and FP-growth

The graphs below present the tests of Apriori and FP-growth algorithms with different datasets and different minimum support values.

2.1 The effect of different sample sizes on the run time

In Figure 1, I test Apriori and FP-growth with different sample sizes. I try both algorithms with sample size = (10, 20, 30, 40, 50, 60, 70, 80, 90, 100). I keep minimum support value at 2 in both algorithms. The execution times of both algorithms increase linearly as the sample size increases. However, the run times for Apriori grow much more faster than FP-growth.

Because of the fact that run times of FP-growth is very little for sample size values in Figure 1, I also try FP-growth with larger sample sizes. Figure 2 presents run times of FP-growth with different sample sizes. Here in Figure 2, I again keep minimum support level constant at 2. I use sample size values = (5, 20, 100, 500, 1000, 2000, 5000, 10000, 20000 32561). I have 32561 because it is the number of rows in adult dataset. Figure 2 displays a positive linear relationship between the run time and the sample size.

Figure 1: Apriori and FP-growth run times with different sample sizes

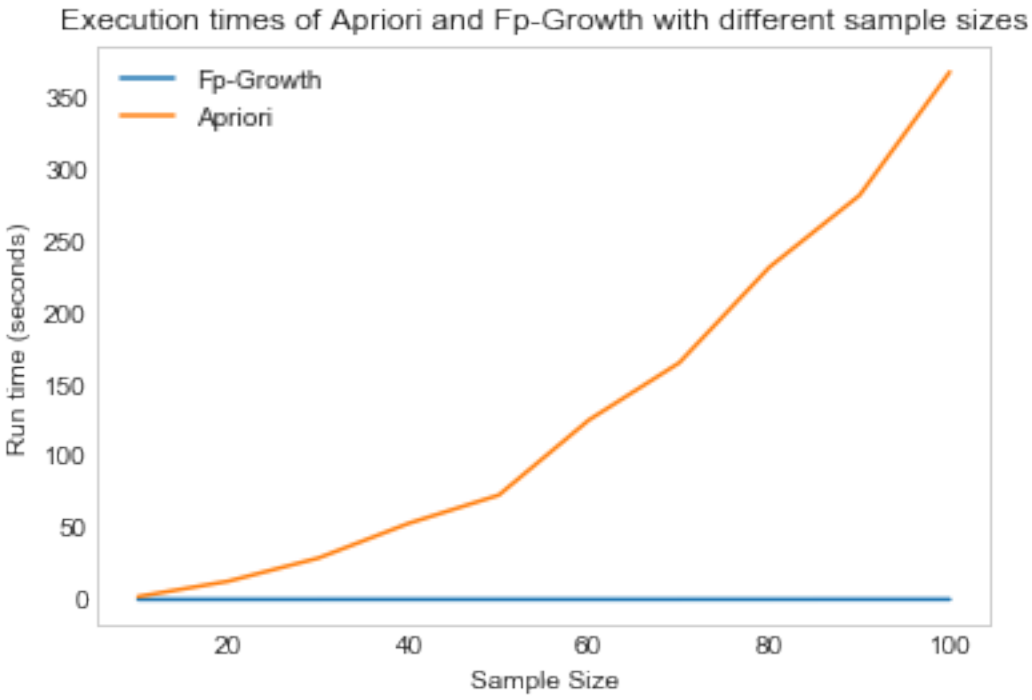
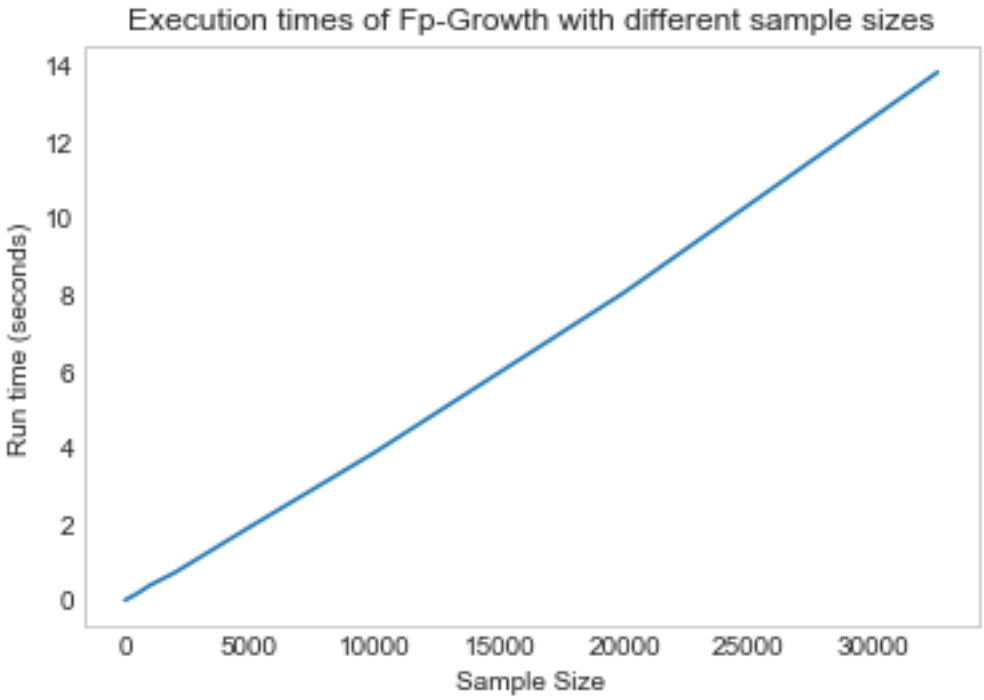


Figure 2: FP-growth run times with different sample sizes



2.2 The effect of minimum support level on the run time

In Figure 3, I test Apriori and FP-growth with different minimum support values. To draw this graph, I keep sample size at 100. I use different minimum support values= (2, 3, 5, 10, 20, 30, 40, 50, 60, 80). As minimum support value increases, the run time of both algorithms decreases. However, I think because of the fact that my sample size is not big enough, the line for FP-growth looks flatter than I expect.

To see the relationship between minimum support value and the run time more clearly, I also use larger N datasets for FP-growth in Figure 4. I try my FP-growth algorithm with N=32561 meaning that I test it for all adult dataset. I use minimum support values (0.004%, 0.01%, 0.02%, 0.50%,1%, 2%, 5%, 10%, 20%, 30%, 50%, 60%, 80%) for Figure 4. Although there are ups and downs in the run times for different minimum support values, the trend line explicitly shows that there is a negative linear relationship between minimum support values and run times. Therefore, I conclude that as the minimum support value increases, the run time will decrease for both algorithms.

Figure 3: Apriori and FP-growth run times with different minimum support values

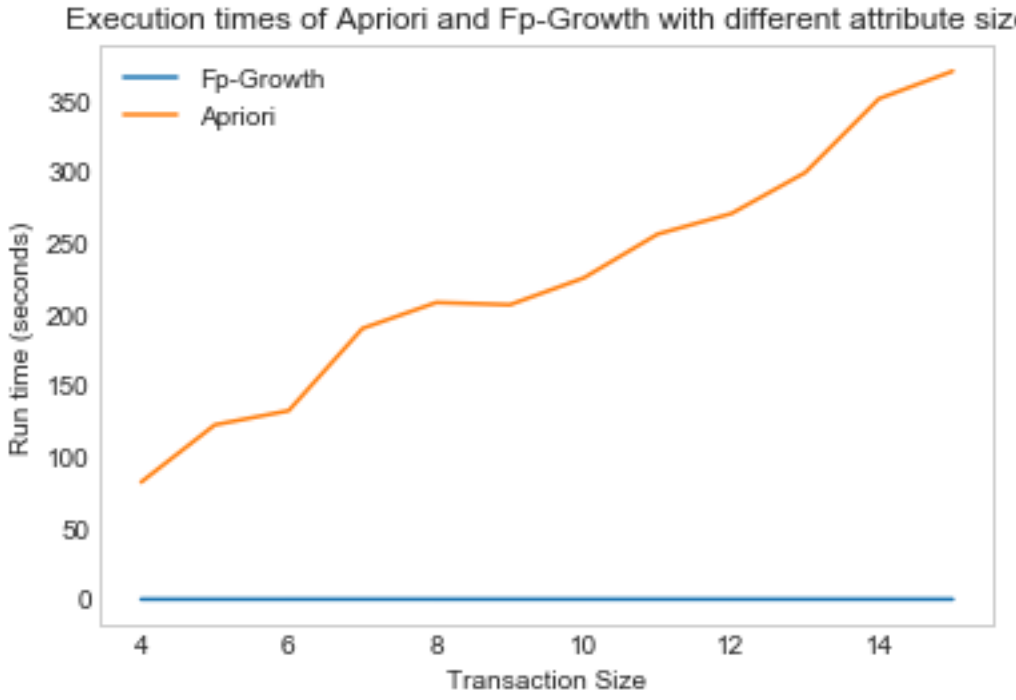
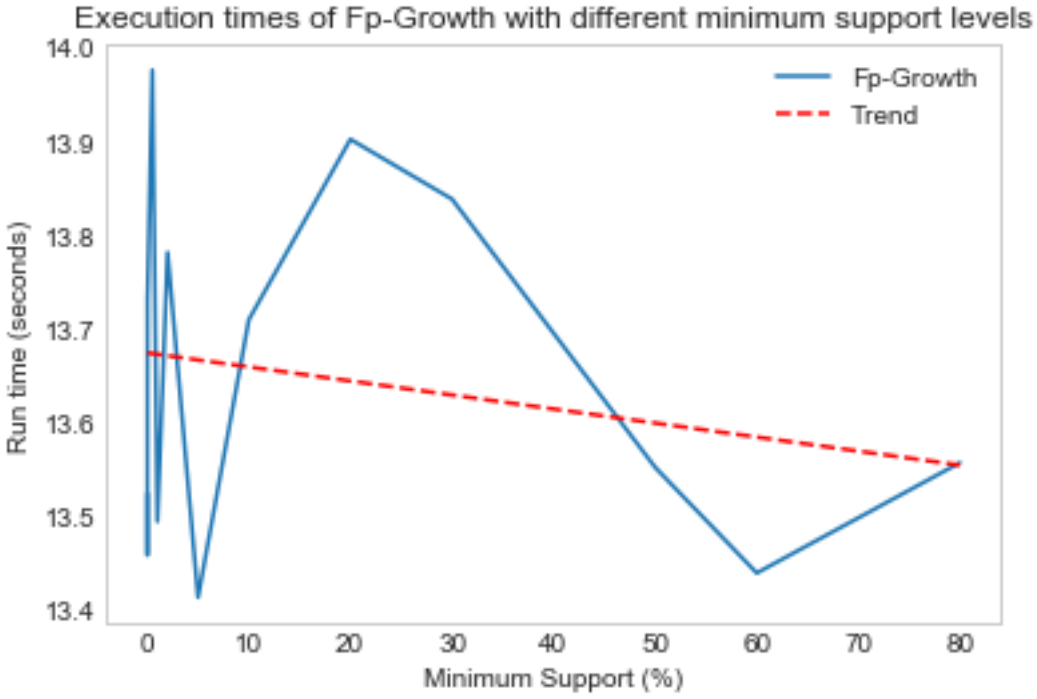


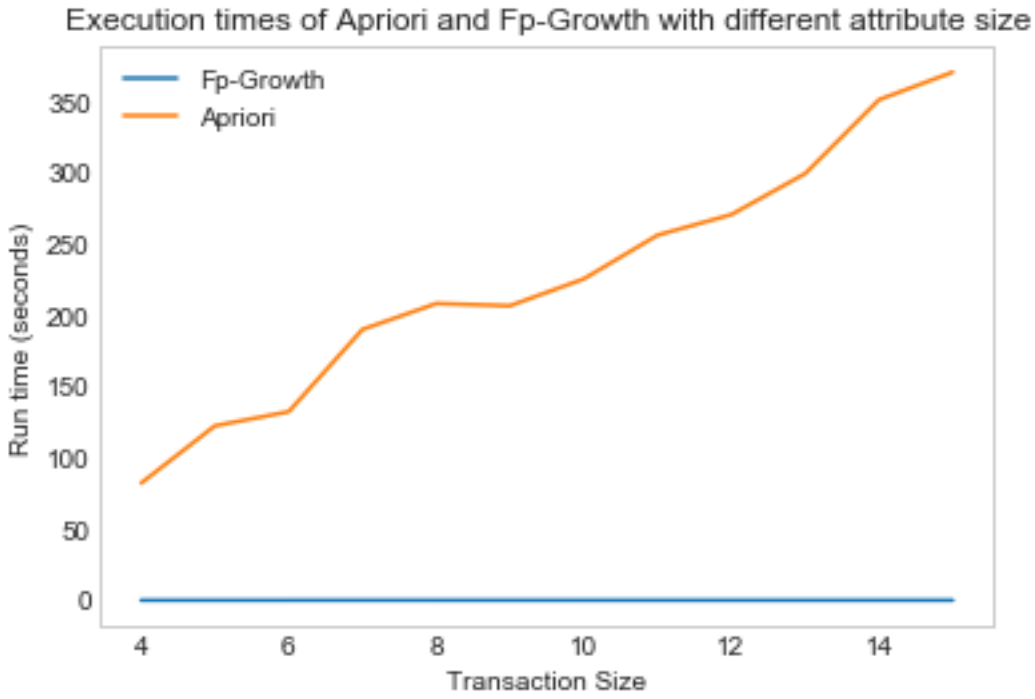
Figure 4: FP-growth run times with different minimum support values



2.3 The effect of transaction size on the run time

Figure 5 shows the effect of transaction size on the run time for Apriori and FP-growth. I test two algorithms with different attribute sizes= (4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15). I keep sample size constant at 100 and minimum support level constant at 2. As attribute size increases, the execution times of algorithms will increase. However, the run times for Apriori grows much more faster than FP-growth.

Figure 5: Apriori and FP-growth run times with different transaction sizes

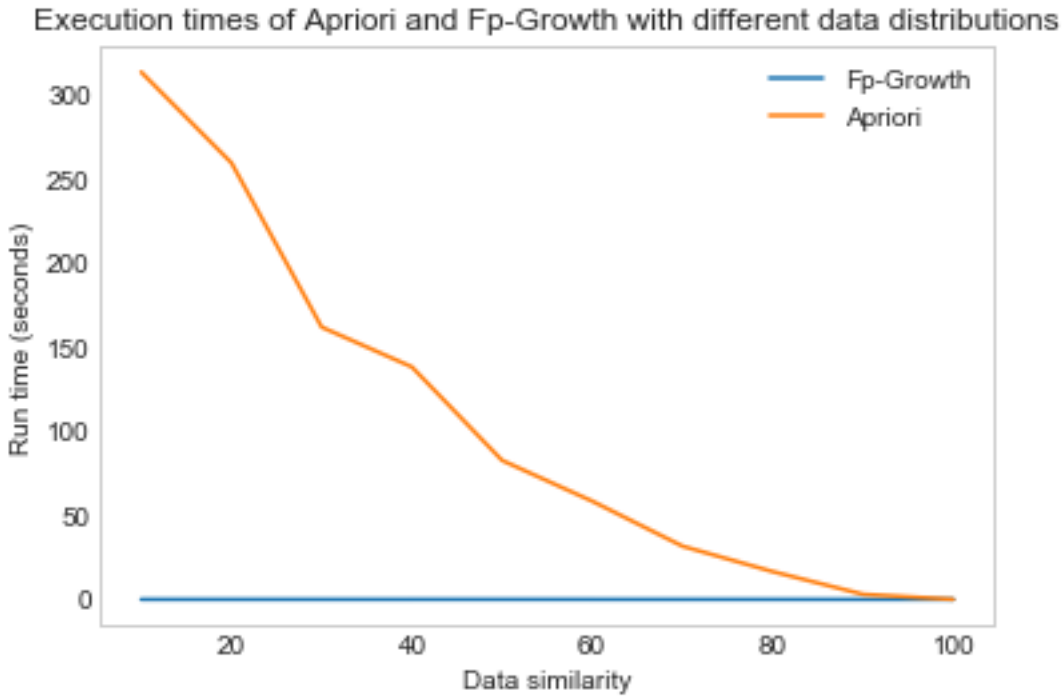


2.4 The effect of data distribution and pattern density on the run time

Figure 6 displays the effect of data distribution on run time of Apriori and FP-growth. To test the relationship, I use sample size N=100 with different similarity levels= (10, 20, 30, 40, 50, 60, 70, 80, 90, 100). Here I keep attribute size at 15 and minimum support level at 2. To create datasets with different similarity levels, I repeat the first line of adult dataset for different times and use the rest of the adult dataset to complete N=100. For example, if

similarity level= 10, I have 10 rows with exactly the same entries and the rest 90 rows are randomly selected from adult dataset. Similary, if similarity level=40, I use 40 rows with exactly same data, and the rest 60 rows are randomly selected from adult dataset. Thus, if similarity=100 it is the repetition of the same row 100 times. As a result, as similarity level increases the variance of the data decreases. At similarity level=100, the data will have uniform or normal distribution. Thus, Figure 6 shows that if the dataset is more uniformly distributed, the run time will decrease.

Figure 6: Apriori and FP-growth run times with different data distributions



Lastly, if pattern density increases I expect the run times will increase. I do not test pattern density with my algorithms.

3 Conclusion

For both Apriori and FP-growth, if sample size, transaction size, and pattern density increase, the run time will increase. Whereas, if minimum support level increases the run time will decrease. In addition, if the data is more uniformly distributed, the run time will decrease.

In conclusion, Apriori algorithm is easier to understand than FP-growth algorithm. Writing code of Apriori is less time-consuming than FP-growth. However, Apriori is computationally inefficient and has scalability issues. Apriori needs much more memory. The run time of Apriori is much more than FP-growth. Because of these large memory usage and slower run time, I think Apriori should not be used for large datasets.