Q1: Comparison of Apriori and FP-Growth

1 Introduction

Frequent itemset mining is a fundamental task in data mining, with applications in market basket analysis, web usage mining, and recommendation systems. Two classical algorithms for this task are *Apriori* and *FP-Growth*. While both algorithms compute the same set of frequent itemsets, their runtime behavior differs significantly depending on dataset characteristics such as size, density, and item frequency distribution.

In this question, we compare Apriori and FP-Growth on:

- A large real-world dataset (WebDocs), and
- A controlled synthetic dataset generated with a skewed frequency distribution.

We analyze how runtime varies with minimum support and explain the observed trends using algorithmic principles.

2 Experimental Setup

2.1 Algorithms

- **Apriori**: Uses level-wise candidate generation and repeated database scans.
- **FP-Growth**: Avoids candidate generation by constructing an FP-tree and mining conditional FP-trees recursively.

Both algorithms were executed using Christian Borgelt's implementations.

2.2 Support Thresholds

For both datasets, experiments were conducted with the following minimum support thresholds (as percentages of total transactions):

$$\{5\%, 10\%, 25\%, 50\%, 90\%\}$$

The percentage supports were converted to absolute supports internally based on the dataset size.

3 Task 1: Experiments on WebDocs Dataset

3.1 Dataset Description

- **Dataset**: WebDocs
- **Number of transactions**: $\sim$1.69 million
- **Nature**: Large-scale, dense, real-world web transaction data

Due to the size and density of the dataset, unconstrained frequent itemset mining was computationally infeasible at low support thresholds.

3.2 Parameter Choice (`-n5`)

To ensure tractable runtimes while preserving meaningful patterns, we limited the maximum itemset size to 5 for both algorithms using the `-n5` parameter.

This constraint was applied consistently to Apriori and FP-Growth and does not affect the qualitative runtime trends.

3.3 Runtime Results (WebDocs)

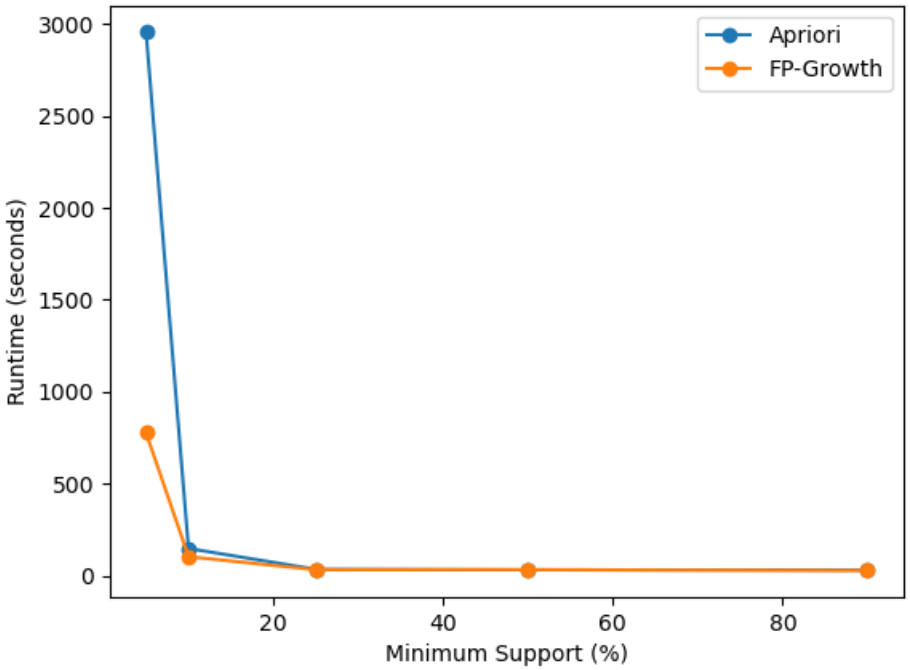

Figure 1: Runtime comparison of Apriori and FP-Growth on WebDocs Dataset

Minimum Support (%)	Apriori (s)	FP-Growth (s)
5	2954.12	777.52
10	149.49	104.78
25	36.15	33.83
50	33.52	34.05
90	31.5	29.77

Table 1: Runtime comparison on WebDocs dataset

(Exact values reported in `runtimes.csv`.)

3.4 Observations

Low support (5%) Apriori exhibits very high runtime due to an explosion in candidate itemsets and multiple database scans. FP-Growth, while still expensive, avoids candidate generation and scales better.

Moderate support (10–25%) As the support threshold increases, the number of frequent items and candidate itemsets decreases, leading to significant runtime reduction for both algorithms. FP-Growth becomes faster than Apriori.

High support (50–90%) Very few itemsets satisfy this support threshold. Both algorithms converge to similar runtimes, as the mining task becomes trivial.

These trends are consistent with the theoretical properties of both algorithms.

4 Task 2: Experiments on Synthetic Dataset

4.1 Dataset Generation

A synthetic dataset was generated using a custom script with the following parameters:

- Number of transactions (N): 15,000
- Universal itemset size (U): 1,000
- Transaction length: Random, between 5 and 30 items
- Item frequency distribution:
  - 30% very frequent items
  - 45% moderately frequent items
  - 25% rare items

Each transaction was guaranteed to be non-empty and there are no duplicates.

4.2 Parameter Choice

Unlike WebDocs, the synthetic dataset is relatively small and controlled. Therefore, no explicit limit on maximum itemset size (`-n`) was imposed, and both algorithms were run fully.

4.3 Runtime Results

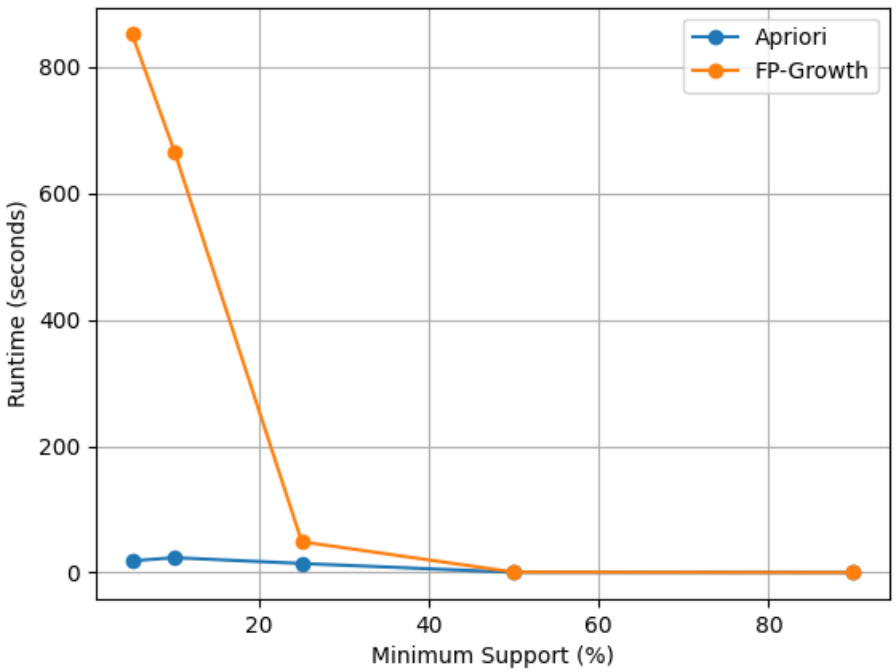

Figure 2: Runtime comparison of Apriori and FP-Growth on Synthetic Dataset

Minimum Support (%)	Apriori (s)	FP-Growth (s)
5	18.63	851.17
10	23.59	665.56
25	14.38	48.7
50	0.47	0.78
90	0.17	0.17

Table 2: Runtime comparison on synthetic dataset

(Exact values reported in `runtimes.csv`.)

4.4 Observations

Low support (5–10%) FP-Growth performs significantly worse than Apriori. The presence of many highly frequent items leads to the construction of dense FP-trees, making recursive conditional mining expensive. Apriori benefits from the relatively small dataset size.

Moderate support (25%) Many items fall below this support threshold, reducing FP-tree complexity. FP-Growth runtime drops sharply and becomes competitive with Apriori.

High support (50–90%) Very few frequent itemsets exist, leading to negligible runtimes for both algorithms.

5 Comparative Analysis: WebDocs vs Synthetic

Aspect	WebDocs	Synthetic
Dataset size	Very large (∼1.69M)	Moderate (15k)
Density	High	Controlled
Max itemset limit	5	None
Low-support winner	FP-Growth	Apriori
Reason	Candidate explosion in Apriori	Dense FP-tree overhead
High-support behavior	Similar runtimes	Similar runtimes

Table 3: Comparison between WebDocs and Synthetic datasets

These results demonstrate that algorithm performance is highly dataset-dependent, and FP-Growth is not universally faster than Apriori.

6 Conclusion

This study highlights the trade-offs between Apriori and FP-Growth in practical settings. Apriori suffers from candidate explosion on large, dense datasets, making FP-Growth more suitable in such cases. However, on smaller or highly skewed datasets, FP-Growth can incur significant overhead due to dense FP-tree construction, allowing Apriori to outperform it.

Overall, the experiments confirm that the choice of frequent itemset mining algorithm should be guided by dataset characteristics rather than theoretical complexity alone.