Lab Work 01: Comparison of the Performances of the Apriori and FP-Growth Algorithms

Course: CSE-4255: Introduction to Data Mining and Warehousing Lab

Submitted by:
Sree Sowmik Kumar Sarker
Roll: 54

Submitted to:
Dr. Chowdhury Farhan Ahmed
Professor
&
Dr. Md. Samiullah
Assistant Professor
Department of Computer Science and Engineering
University of Dhaka.

Date : 28 January, 2021

Introduction:

In this experiment, I have implemented Apriori algorithm using trie based data structure and FP-Growth algorithm to mine frequent patterns in real-life datasets. The performance of both algorithms is compared in terms of execution time and execution time memory usage by setting the same minimum support threshold for different datasets.

Implementation:

Apriori:

Apriori is an iterative approach known as level-wise search, where k itemsets are used to explore k+1 itemsets. Apriori property(all the non-empty subsets of a frequent itemset must be frequent) is used to reduce search space.

- Dataset is scanned once for a particular level of candidate generation.
- First find the frequent-1 itemset. Then, to find Lk a set of k itemsets is generated by joining Lk-1 with itself.
- Any k-1 itemset that is not frequent can not be a subset of a frequent k itemset. So, if any k-1 subset of a candidate k itemset is not in Lk-1, then the candidate can not be frequent and can be removed from Ck(candidate-itemset).
- A prefix trie is constructed on the itemsets in Ck to compute the support count of candidates at any particular level.

FP-Growth:

FP-Tree data structure is used. After once scanning the dataset find minimum support for each item. Then discarded infrequent items and sorted the itemsets based on their minimum support in a header table. Header table contains information which item is contained in which nodes in the FP-tree through node links.

- Extracts frequent itemsets from the FP-tree from leaves to the root.
- Perform mining recursively on the FP-tree to generate all the frequent patterns. The pattern generation is done by concatenation of suffix patterns with frequent patterns generated from the FP-tree recursively.

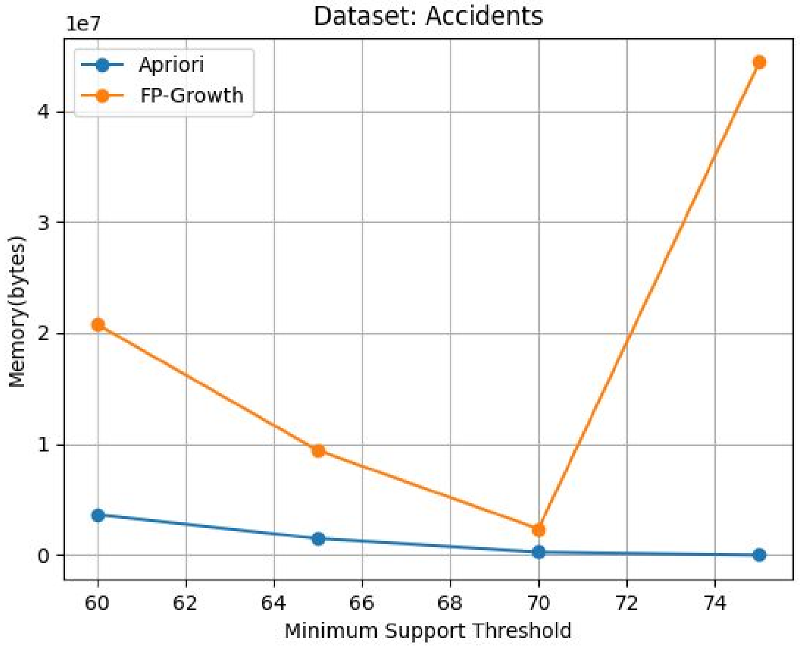
Datasets:

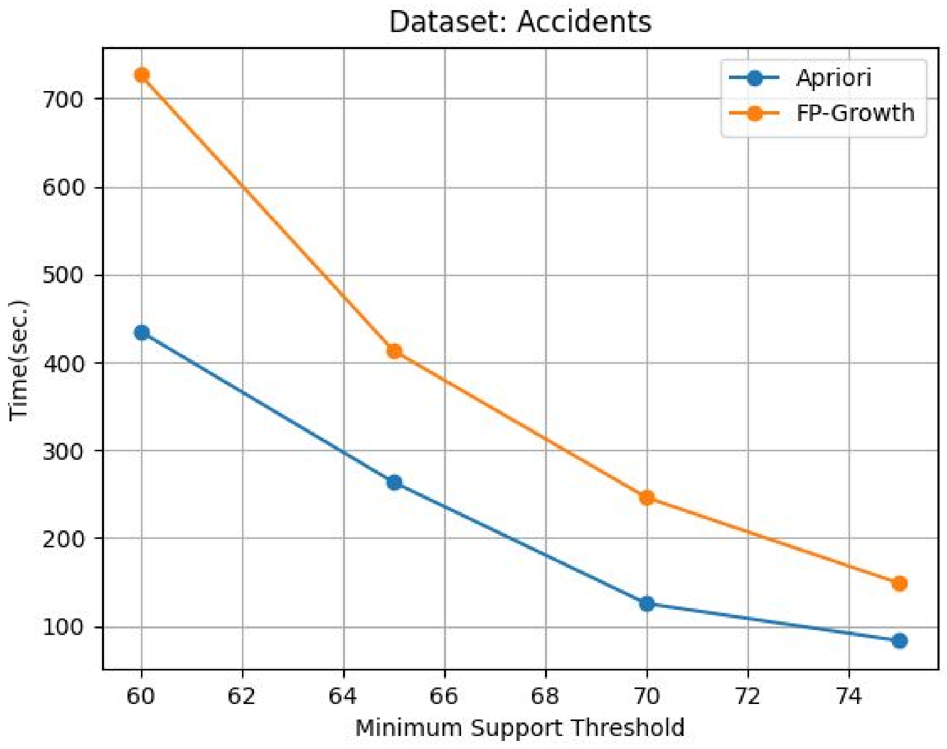
In this experiment, I use five datasets. 2 dense(Mushroom, Chess), 2 sparse(Kosarak, Accidents) and 1 is large(Pumsb_star).

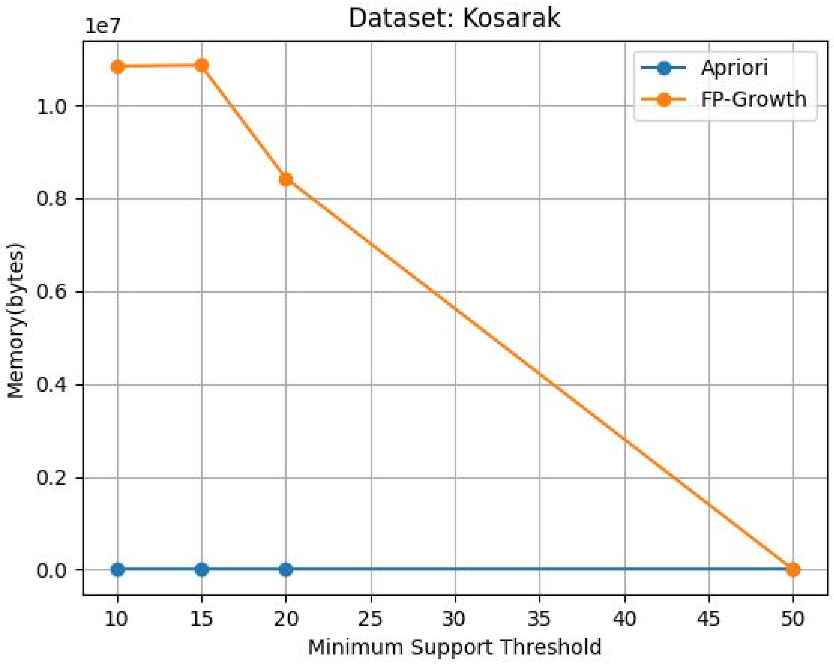
Results:

The obtained results(comparison of runtime and memory of both algorithm are plotted and given below:

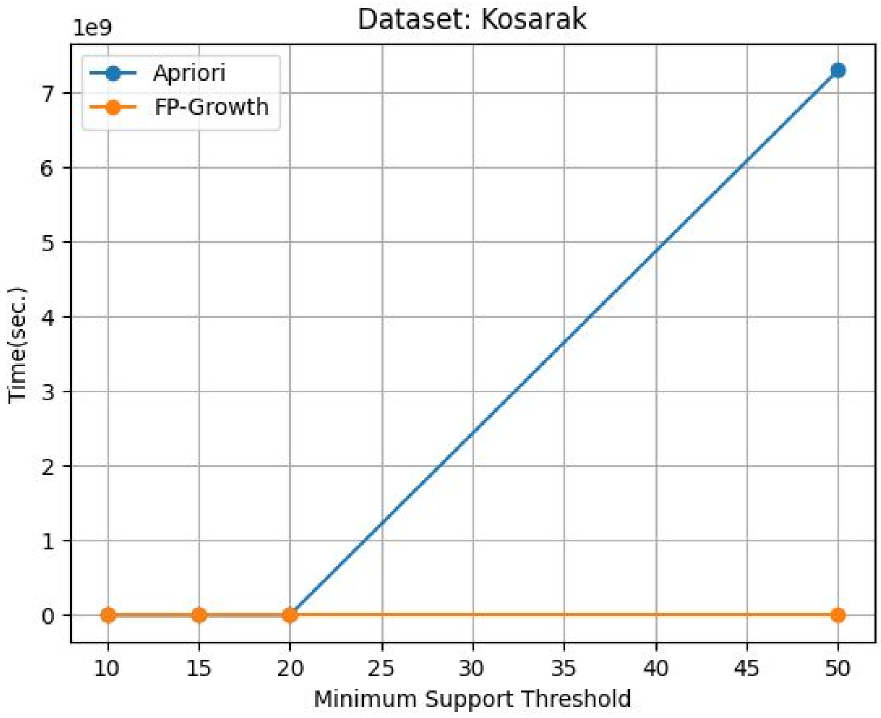
Sparse dataset:(Accidents, Kosarak)





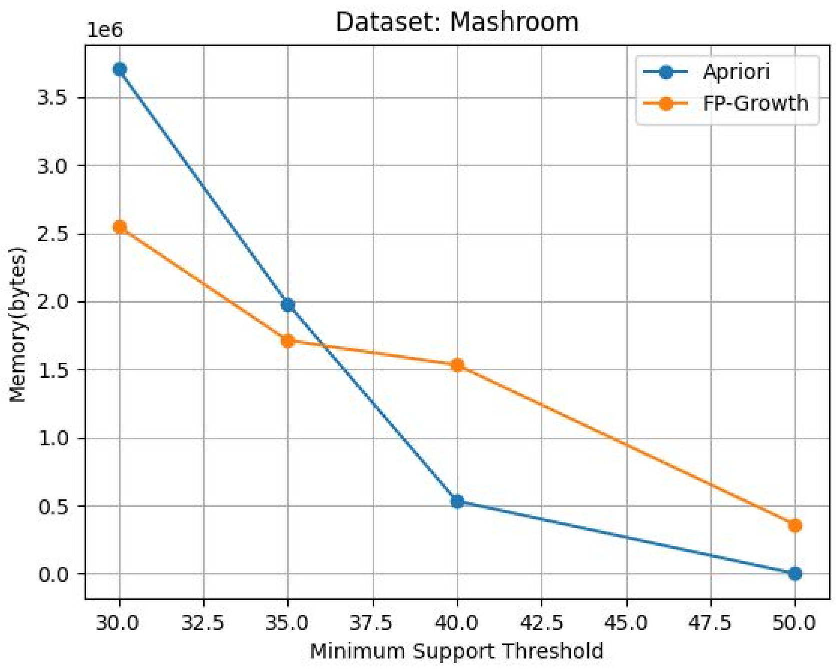


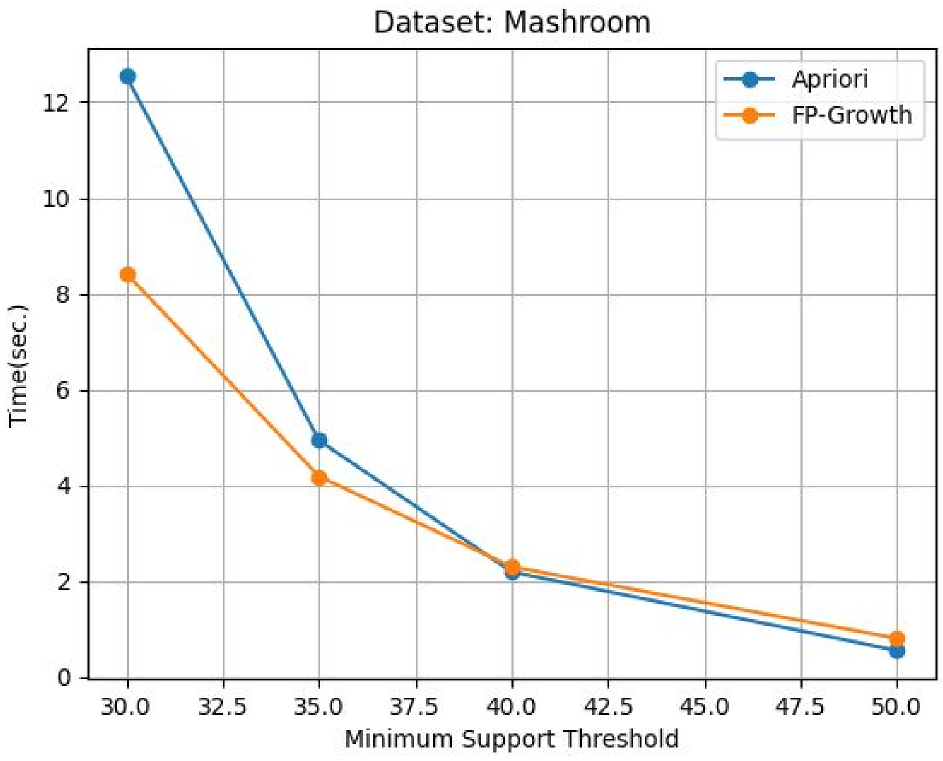
Dataset: Kosarak
1e7
Apriori
FP-Growth
Memory(bytes)
1.0
0.8
0.6
0.4
0.2
0.0
10
15
20
25
30
35
40
45
50
Minimum Support Threshold

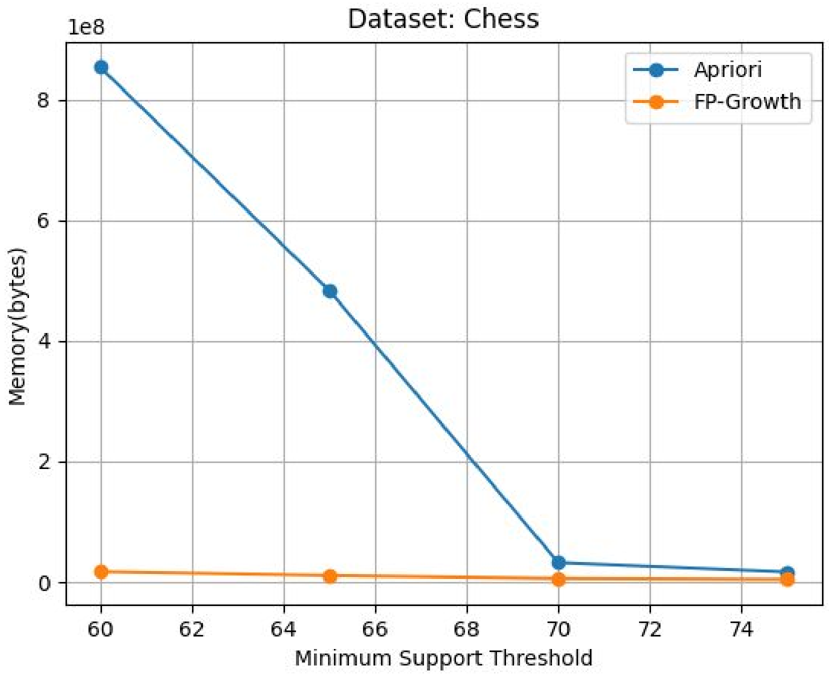


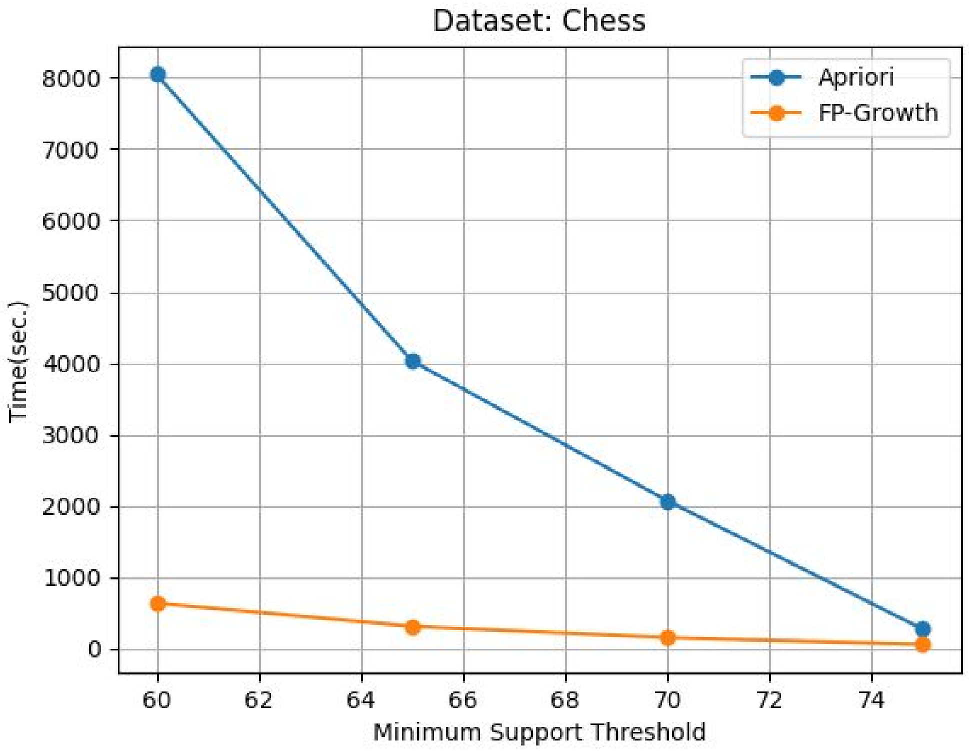
Dataset: Kosarak
1e9
Apriori
FP-Growth
Time(sec.)
7
6
5
4
3
2
1
0
10
15
20
25
30
35
40
45
50
Minimum Support Threshold

Dense dataset(Mushroom, Chess):

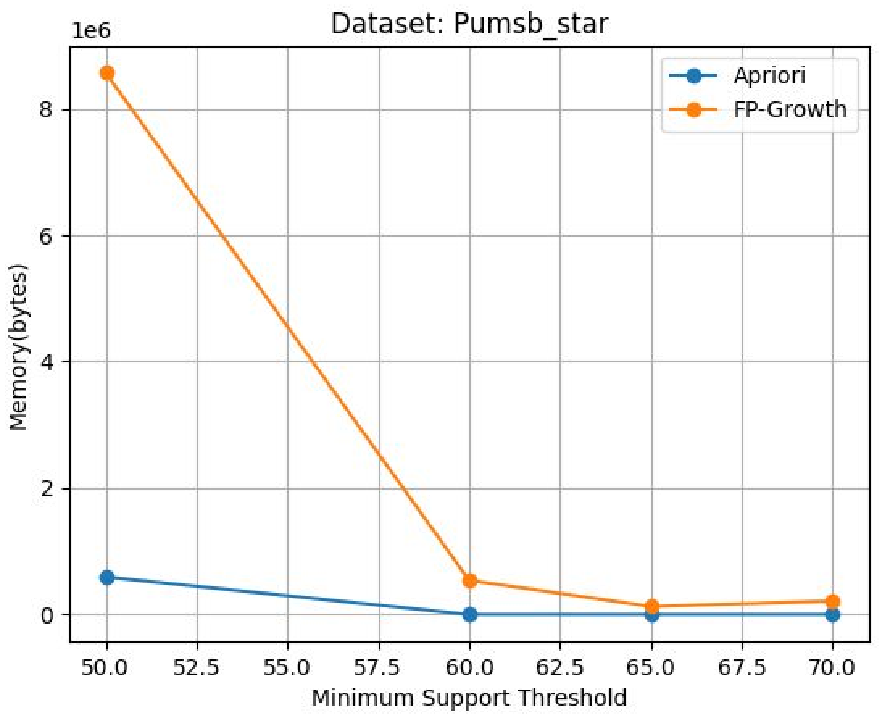


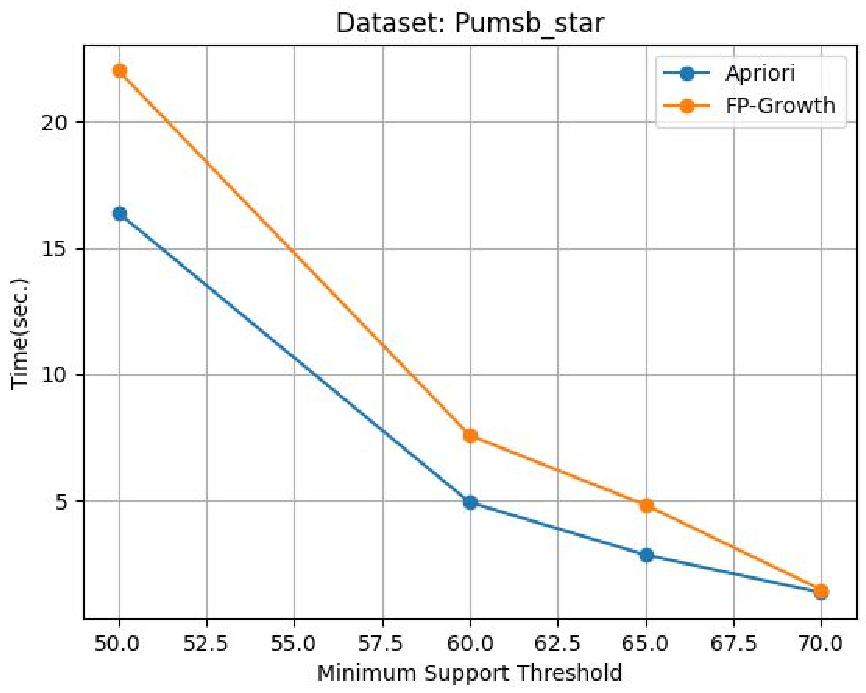



Dataset: Chess
1e8
Memory(bytes)
Minimum Support Threshold
Apriori
FP-Growth
0
2
4
6
8
60
62
64
66
68
70
72
74


Dataset: Chess
Time(sec.)
Minimum Support Threshold
Apriori
FP-Growth
0
1000
2000
3000
4000
5000
6000
7000
8000
60
62
64
66
68
70
72
74

Large dataset(Pubsb_star):


Dataset: Pumsb_star
1e6
Apriori
FP-Growth
Memory(bytes)
Minimum Support Threshold


Dataset: Pumsb_star
Apriori
FP-Growth
Time(sec.)
Minimum Support Threshold