LINFO2364: Mining Patterns in Data
Project 1: Implementing Apriori

Manuelle Ndamtang, 66732000 Juffern Saskia 24411800

2022 March

Introduction

In the scope of the course LINFO2364: Mining Patterns in Data, we had to realize this first project. The project consist in implementing the Apriori algorithm which aims to find the frequent itemsets in a dataset given a fixed minimum support. Additionally, we also had to implement an alternative frequent itemset miner and to compare the performances of both algorithms. The chosen algorithm for this purpose is Eclat, a version of the DFS algorithm.

1 Apriori

During the apriori algorithm, we are first looking for candidate sets. For each candidate, we count their support and finally we check if their frequencies are higher than our minimum frequency. This process is repeated level-wise, which means it starts with candidate sets that only contain 1 item, then 2 items and so on until there are no candidate sets left. The apriori algorithm relies on anti-monotonicity, which means that any subset of a frequent itemset is also frequent and any superset of an infrequent itemset is also infrequent.

1.1 Generation of candidates

This anti-monotonicity principle is used during the generation of the candidates: we only consider candidates of the previous level and we try to extend these sets. Sets that were already infrequent at the previous step aren't considered anymore. In this way, the anti-monotonicity is used to reduce the number of candidate sets generated.
One thing that has a significant impact on the performance of the developed algorithm is the fact that for each level, the generation of candidates of the next level is based on the selection of the candidates(previously ordered) whose last element differs.

1.2 Counting candidates

After we got the candidates, we count the frequencies of each of them. Through every transaction we check if the candidate is present and we do this at each level.

2 Alternative miner: Eclat

The different steps to proceed to the eclat algorithm:
First we need to generate the vertical representation of the dataset, then we build the projected database for each candidate and determine their frequencies.

2.1 Vertical representation

For the eclat algorithm, we need a vertical representation of our dataset. This means that we generate for each item the set of transactions where this item is included (the cover of the item).

2.2 Projected database

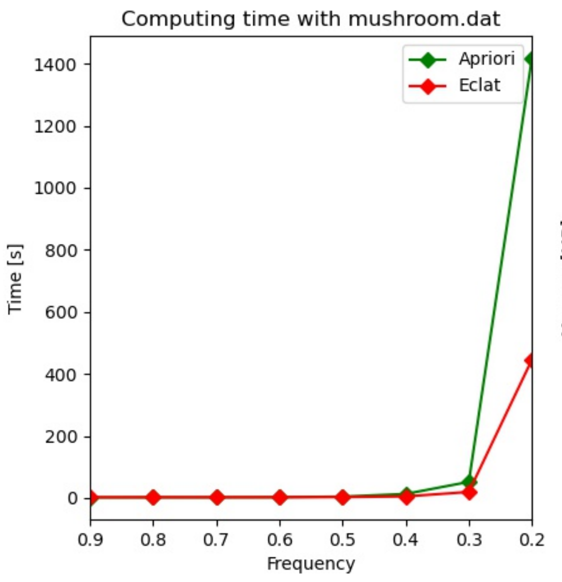
We also use a projected database, which means that we delete every non-frequent item from the database for each specific candidate. This also means that each candidate will have its own version of the projected database. We can predict the excessive use of memory by this algorithm.

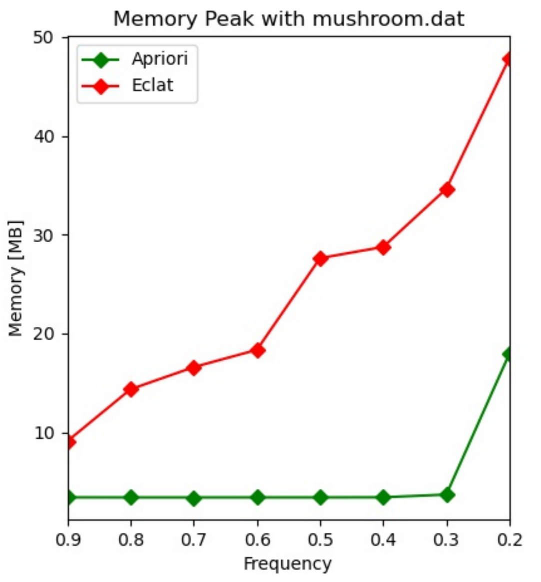
2.3 Main function of the algorithm

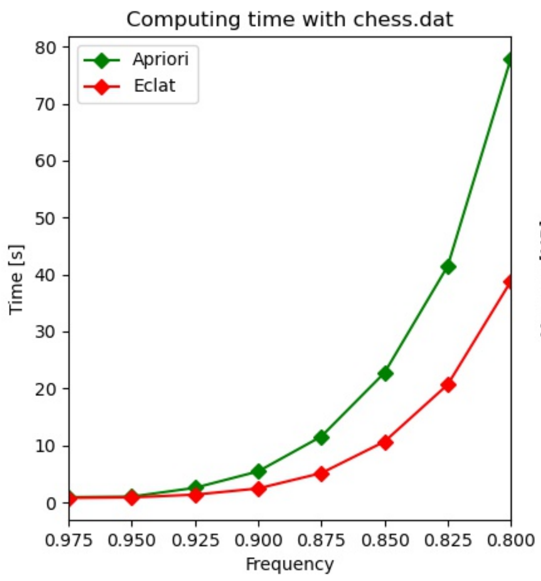
The algorithm consists in generating the candidates in an ordered way(for example one can generate as candidate [12, 15] and not [15, 12] in order to avoid redundancy) and applying recursively the projection of each candidate. Once this step is done, determining the support will consist in finding the length of the union of the different covers of each item. Thus the recursion will stop if the candidate is not frequent.

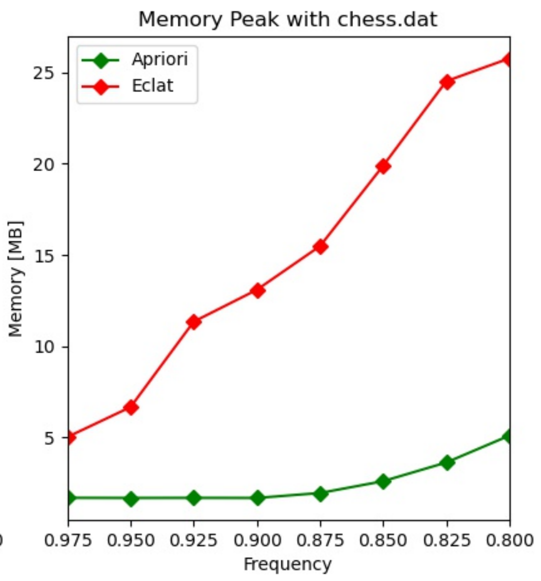
3 Performances of the algorithms

During our performance evaluation, we recorded the time the miner needed to mine all itemsets for different minimum frequencies values as well the peak memory usage during the mining.
We noticed that the apriori algorithm is very fast on the ”mushroom.dat” dataset for high mininum frequencies (from 0.9 to 06). But when it comes to smaller frequencies, Eclat has a very good performance.
However, for datasets like ”chess.dat”, eclat is much faster.
However, in terms of memory management, apriori is more efficient. This could be explained by the fact that apriori does not perform any dataset transformation for each candidate. The algorithm uses the same form of the dataset during its execution for every candidates. While Eclat performs a vertical representation of the dataset and a projection for each candidate (based on a copy of the superset projection).









Conclusion

At the end of our project, we can conclude that the Apriori algorithm is much more efficient for small datasets with large minimum frequency. But when these conditions are not really respected, it is preferable to opt for the Eclat but we must take into account an important disadvantage: the memory consumption. In general, if you want to be more efficient in searching for frequent itemsets, it is better to use Eclat.