A picture containing diagram

Description automatically generated

**Course**: EECS 4415U Big Data Systems

**Topic:** Market Basket Analysis

**Description**

Your project task it to conduct market-basket analysis by developing on your own and comparing experimentally various frequent itemset mining algorithms, including, Apriori, PCY, sampling and SON. The goal is to find frequent pairs of items (since this is the most computationally involving step) and analyze the results.

**Resources**

* Review slides for Lecture 3 posted on the eClass website
* Chapter 6 from MMS book posted on eClass (Association Rules).

**Programming Language**

You can choose your favorite programming language (for example, C++, Java, C#, Python and Go)

**Tasks and Schedule**

1. Setting up environment, implementation of **Apriori** algorithm for finding frequent pairs of elements (Week 1).
2. Finishing implementation of Apriori algorithm and conducting scalability experiments on given datasets (Week 2). Starting implementation of **PCY algorithm**.
3. Finishing implementation of **PCY Algorithm** for finding frequent pairs of elements. Compare experimentally performance of PCY vs Apriori (Week 3).
4. Extend your implementation to the **Random Sampling** and **SON** version of the algorithm (Week 4). What is the efficiency of the method in comparison to algorithms developed in Points 1-3? Quantify the number of false positives.
5. Implement **Multistage OR Multihash version** of the PCY algorithm. Finish the implementation and write the final report (Week 5).

**Datasets**

1. The retail dataset contains the (anonymized) retail market basket data (~88200 baskets) from an anonymous Belgian retail store.

Note that since the dataset was anonymized the preprocessing step to map text labels into integers is done for you. Working with integers is more efficient than textual data as it saves the main memory.

Retails dataset is available to download at:

[Retail](http://fimi.uantwerpen.be/data/retail.dat)

1. Netflix dataset

Conduct additionally experiments over the larger Netflix dataset to test the limitations of your implementations.

Netflix data set is available at:

[Netflix](https://drive.google.com/open?id=1EX_2Pkid6EC4H-4KN0kP_S_89GKaTnXo)

**Experiments**

Provide the runtimes on the entire dataset for both Retail and Netflix for all algorithms listed in Weeks 1-5.

Use a support threshold of 1% and 2%.

Additionally, perform the scalability study on Netflix dataset for only Apriori algorithm for finding frequent pairs of items with a support threshold of 1% by dividing the data into the subsets: 20%, 40%, 60%, 80% and 100% of the entire dataset and measuring the time performance. Provide the results in the diagram figure (with size of the subset of data on X-axis and runtime on the Y-axis).

Note that if the running of the algorithm extends 5h, you can terminate your program, and report it. Save the copy of the set of the frequent itemsets found so far (e.g., say every 30 min) in this case.

Save the found frequent pairs of items. Quantify the number of false positives for the sampling methods.

**Report**

After Week 5 you will have to submit the entire report (for tasks in Weeks 1-5), which will be marked towards your project. Attach separately your code. See FAQ below for details of the submission.

**Award**

The most efficient implementation will be determined and given an extra 1 bonus point.

**Note**

All questions regarding the course projects should be directed to TA (contact e-mail information posted on eClass).

-----------------------------------

FAQ

**Q:**What about citations?

**A:** This is important and you should take it seriously. Everything in your implementation and submission needs to be *well-documented* in your project report and *plagiarism-free*.

**Q**: What to include in my report?

**A**: Please follow the weekly delivery scheme in the instructions to structure the report and provide the details about your implementation and analysis including:

* Description in one paragraph of each of the algorithms
* Experimental performance analysis for each method.
* Performance comparison discussion between various methods. Discuss why one algorithm might be faster than another.
* Discuss how support 1% vs 2% affects the results.
* Discuss the trends you observe in the scalability study by dividing the data into the subsets of 20%, 40%, 60%, 80% and 100% for the Apriori method.
* Choice of hashing function for PCY.
* Discussion about false positives for sampling. What is the good sampling % and strategy? What is the good chunk size?
* Conclude the results in one paragraph that you found in your project.

Both logic and correctness will be taken into account.

**Q:**What to include in my submission?

**A:** Please include the following in a zipped folder: (1) your implementation with each algorithm in a separate file, for example **apriori.py**), (2) the frequent itemsets you find in a txt file (where each line of the **freq-itemsets.txt** file consists of a frequent item pair separated by a space or a comma or some other sensible delimiter), (3) your report in PDF format, (4) a ReadMe txt with the instructions how to run your code.

**Q:**Running my code takes too long and I need to terminate the program since it has been running for more than 5 hours already. But I cannot seem to stop the program from running so it can give me the frequent itemsets. What should I do?

**A:**When your program runs for longer than five hours, please terminate it and document this in your project report. We recommend that when running your program to save a copy of the set of the frequent itemsets found so far in a txt file; and then update this txt file every so often (for example every 30 mins) with the new frequent itemsets your program has found. By doing this, you are sure to obtain at least a partial subset of the total frequent itemsets should it happen that the program exceeds 5 hours of runtime.

**Q:**Could I make use of the frequent itemsets libraries?

**A:** If such library is directly involved with frequent itemset mining, then you are not allowed to use them. You are supposed to code it from scratch on your own.

**Q:**Which hash function should I use?

**A:** There is no one correct hash function to use. Ideally, we would want to try to minimize (or even avoid) collisions, but at the same time trying not to overkill with overly sophisticated and complex hash functions. One can do a quick research on some good hash functions and then explain briefly in the report the choice of such function(s). Please ensure that resource materials are cited properly and accordingly.

**Q:**What format should I use for the project report?

**A:** We encourage everyone to typeset their project reports using LaTeX. However, other forms are definitely also accepted depending, which tools you are comfortable with. You can chosoe some of your other favourite editors, such as MS Word, Open Office etc., as long as your report is well organized and elegant.

In case of LaTeX we recommend using [Overleaf](https://www.overleaf.com/). Outputs in LaTeX generate some high level quality that is clear, easy-to-read and presentable -- qualities that we are looking for in an exceptional report. Here are some sample templates that you can use (feel free to use other templates that you can find):

* <https://www.overleaf.com/latex/templates/acm-conference-proceedings-primary-article-template/wbvnghjbzwpc>

Ultimately, the project report should still be in PDF format when you submit them.