MM-811 Multimedia Data Mining – Winter 2016

Assignment 1

This assignment is about mining for association rules. You are asked to implement in C or C++ (on your own), without copying any code from any on-line sources, the *Apriori* algorithm as published in the paper:

Rakesh Agrawal, Ramakrishnan Srikant
Fast Algorithms for Mining Association Rules
Proc. 20th Int. Conf. Very Large Data Bases, VLDB, 1994
http://citeseer.ist.psu.edu/agrawal94fast.html

The paper is available in the Assignment folder on e-class.

The algorithm for finding frequent itemsets is as follows:

```
Ck: Candidate itemset of size k
Lk: Frequent itemset of size k

INPUT: database and min_support

L1 = {frequent items};
for (k = 1; Lk is not empty; k++) do begin
    Ck+1 = candidates generated from Lk;
    for each transaction t in database do
       increment the count of all candidates in
       Ck+1   that are contained in t
    Lk+1  = candidates in Ck+1 with min_support
    end
return all Lk;
```

To find all strong rules, the following algorithm can be used:

```
For each frequent itemset, f
    generate all non-empty subsets of f.
For every non-empty subset s of f do
     output rule s -> (f-s) if support(f)/support(s) >= min_confidence
end
```

A transactional database with 10,000 transactions each with 12 items on average from a set of 500 distinct items (dataT10K500D12L) is available in the Assignment folder on eclass.

Each line in this file represents a transaction. Items are separated by the space character. For example:
77 100 118 123 145 171 192 225 230 243 261 273 326 349 412 461 470

Assume the items in each transaction are sorted.

Test your implementation on this dataset. You are asked to provide execution time measures as well as numbers of frequent itemsets and number of strong association rules for the following cases:

- 10000 transactions
- 1000 transactions
- 100 transactions

with the minimum support set to 10 transactions (i.e. 0.1%, 1% and 10% respectively) and minimum confidence set to 80%.

For each case, the output should be:

Number of frequent 1_itemsets:	
Number of frequent 2_itemsets:	
Number of frequent 3_itemsets:	
Number of frequent 4_itemsets:	
...	
Number of frequent k_itemsets:	
Number of association rules	

IMPORTANT:

- The algorithm is to be implemented in c/c++ (preferably compiled with gcc). The implementations will be compared with among other criteria the execution time and scalability.
- The implementation will be tested with other datasets.
- Plagiarism will not be tolerated. Write your own code. If you collaborate on modules or part of the code, you need to clearly indicate it.
- Marks will be based on correctness first, then efficiency and cleanliness of code.
- If the execution time takes more than twice the average execution time of the class, the implementation will not be marked for efficiency.
- The number of transactions and the number of distinct items are not given and must be determined by the program while reading the input file.

Deliverables:

This assignment is to be submitted via email (zaiane @ cs.ualberta.ca). Send *only one tar file (or zip file)* that contains all that you are submitting:

1. The executable should get four parameters as input: 1- file name; 2-minimum-support; and 3- minimum confidence. The thresholds should be numbers between 0 and 1. The forth parameter is either "r", "f", "a", or absent. When "r", then all strong association rules are displayed. When "f" then all frequent itemsets are displayed. When "a" then all frequent itemsets and all strong association rules are displayed. When absent, then only the number of frequent itemsets of different sizes and the number of strong rules are displayed.
   The code should be reasonably commented.
   When displaying the frequent itemsets, print their support "(s)". When displaying the strong rules, print their support and confidence "(s, c)". In both cases with 2 decimal point precision.
2. One file containing all strong association rules for the 10,000 transaction case. Each line should contain a rule with the following syntax:
   x, y, z -> a, b (s,c)
   meaning that items are separated by ", " and the body and head of the rule are separated by " -> ". Do not forget the spaces. "s" and "c" are support and confidence respectively. They need to be with 2 decimal point precision.
3. A file containing the time and count measures as explained above.