NCKU DataMining Homework1 P76051072 林嘉源 Report

1. Dataset: ( IBM Dataset Generator )

min\_sup = 0.5

min\_conf = 0.9

Generate top-100 Association Rule

Environment: MSVC 2012, x64, Windows8, C++

The Largest Itemset are listed as following:

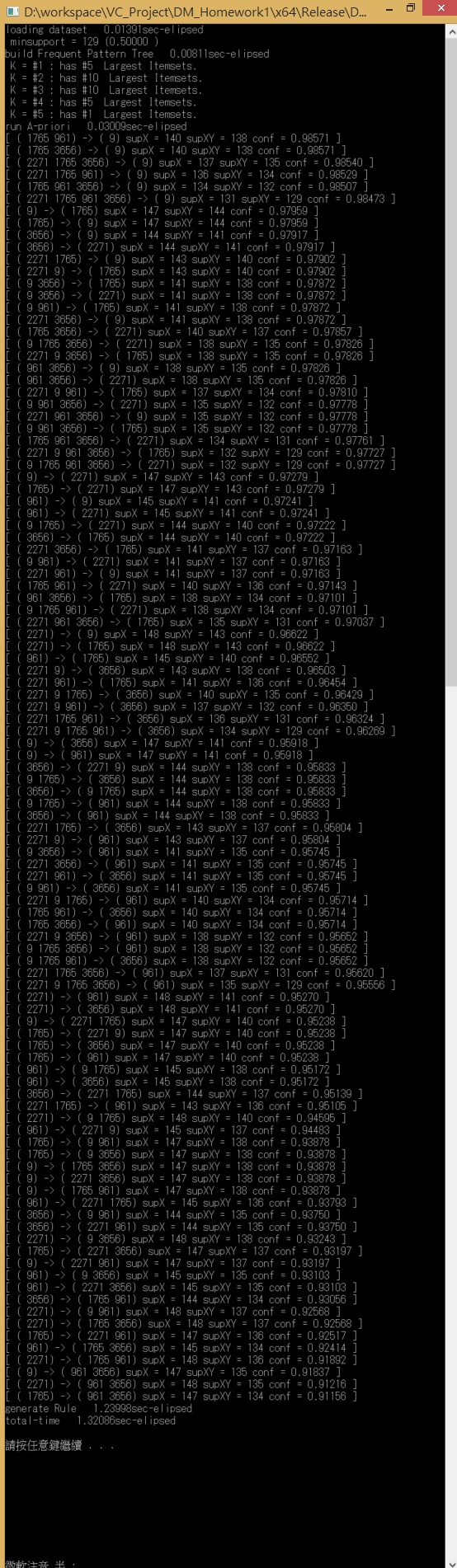
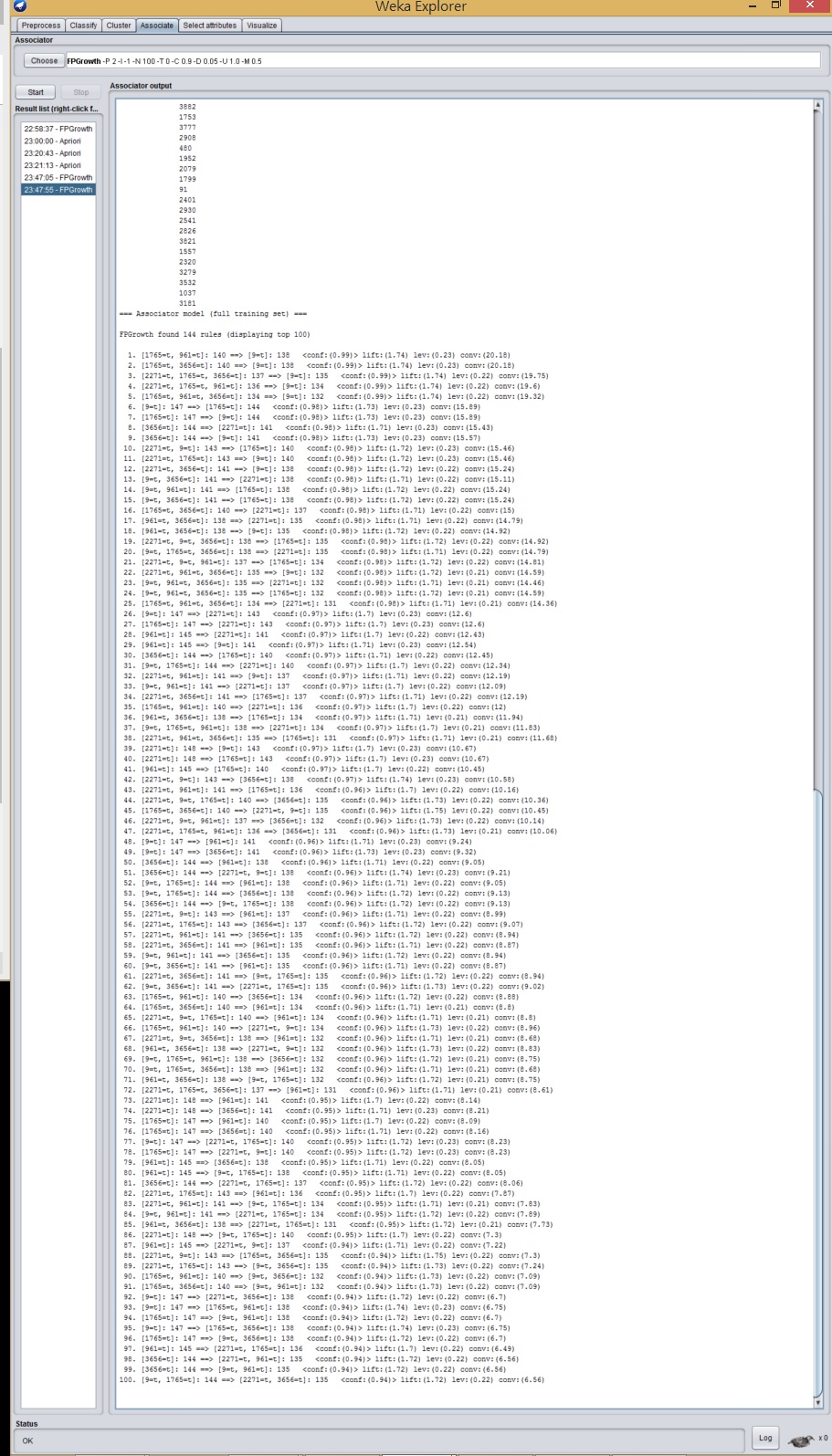
K = 1 has 5 Largest Itemsets

K = 2 has 10 Largest Itemsets

K = 3 has 10 Largest Itemsets

K = 4 has 5 Largest Itemsets

K = 5 has 1 Largest Itemsets



1. Analysis:

The main concept of FPGrowth is harnessing a good strategy to divide the biggest transaction dataset into several minor datasets and then conquer each minor dataset separately to achieve the enhanced speed-up.

The most tricky idea in this FPGrowth method is building the conditional Frequent Pattern Tree (FP Tree). Because for each pair of patterns belonging to two different conditional FP Tree means that they must has different item-ID on the last of the patterns, split the biggest dataset into several partitions based on the last item-ID of each the pattern.

Let’s take a closer look at the algorithm.

1. Improve the Time complexity:

Because the FP Tree is a multi-way tree structure, with an eye to reducing the time complexity in searching the pointer to its children, here this algorithm apply unorder\_map, the standard template library in C++ applying hash to reduce the time complexity to O(1), compared to list O(n) or map O(log(n) ).

1. Reducing the Memory Space:

In my source code, with a view to saving the memory space, each time a conditional FP Tree is constructed, apriori algorithm will be applied immediately to find the frequent patterns and destroy this conditional FP Tree before constructing the next one.

In my experiment, my FPGrowth source code can find the frequent patterns from IBM Dataset Generator with 19578 different items and 10436 itemsets in less than 1.5 second.

1. dd