Data Mining 2018

Homework 1 – Frequent Itemset Mining

Due: 23:59, Oct 20, 2018

The frequent itemset mining is a fundamental technique in data mining applications. In previous approaches, there are different ways to represent data. Horizontal data representation records items contained in each transaction (TID). Vertical data representation records TIDs for each item. Examples are shown in Figure 1.

One of the approaches using horizontal data layout is Apriori. It finds frequent itemsets by generating candidate k-itemsets from frequent k-1 itemsets and pruning (counting support of k-itemset candidate) to remove infrequent itemsets.

In Eclat algorithm, for each item, store a list of TIDs with vertical data layout. To determine the support value of any k-itemset, we just intersecting TID lists of two of k-1 subsets. Also, as mentioned in the class, the vertical data layout is very suitable for applying to parallel computing. Thus, the Eclat algorithm may benefit from mining big data by GPGPU to be explored in HW2. An example of eclat is shown in Figure 2.

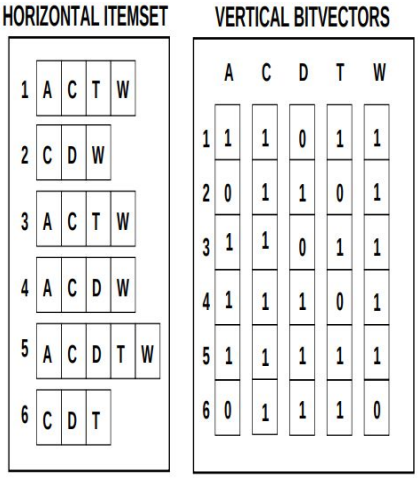


Figure 1. (a) Horizontal data (b) Vertical bitvector data

In this assignment, you are required to implement two basic algorithms to perform frequent pattern mining:

(1) Apriori using horizontal data representation

(2) Eclat using **vertical bitvector** data representation

You can implement in python (version >= 3.5) or C++.

The input dataset and output format are in the following:

- Input:

```
1 6 9 22 23 26 36 39 42 44 56 57 67 71 79 88 90 94 97 104 113 120 128
1 6 9 22 23 26 36 39 42 44 56 57 67 71 79 88 90 94 97 104 108 120 128
1 3 12 21 23 25 36 38 41 53 55 58 67 71 79 88 90 94 97 104 107 119 124
1 3 12 21 23 25 36 38 41 53 55 58 67 71 79 88 90 94 97 104 107 119 122
1 3 12 21 23 25 36 38 41 53 55 58 67 71 79 88 90 94 97 104 107 118 124
```

Each line in the dataset represents one transaction. The numbers separated by space in each line are the items in the transaction.

- Output:

The output format is <itemset, support>. The items in each itemset is in ascending order. For example,

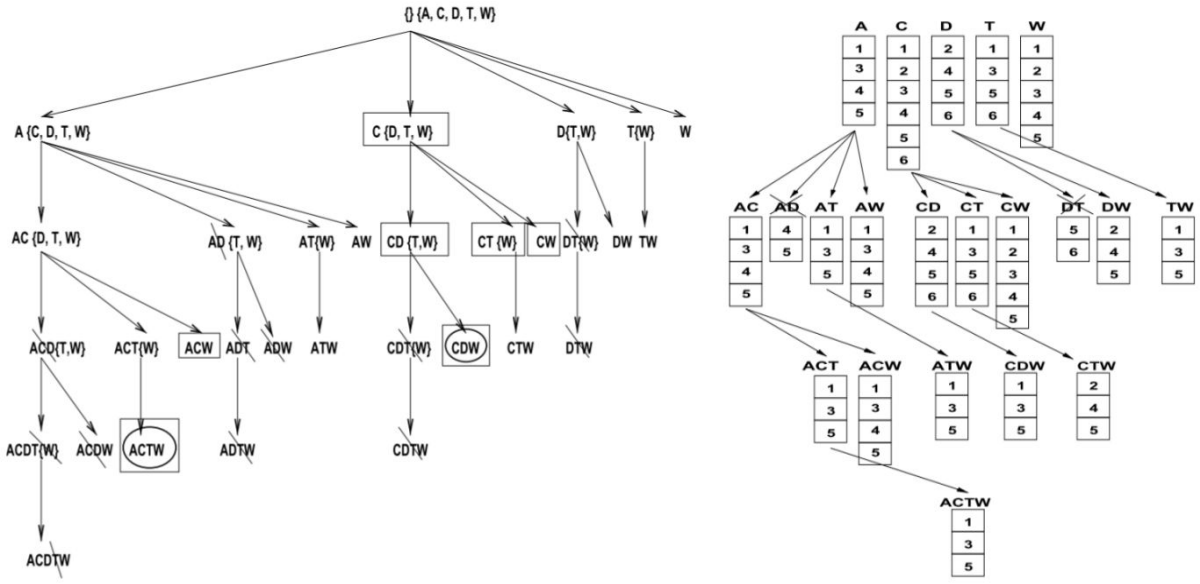
```
1 23 36 38 41 67 71 104 (2272)
1 23 36 38 41 67 90 94 97 104 (2432)
1 23 36 38 41 67 90 94 97 (2432)
```

We will run your code using command:

./apriori.sh $1 $2 $3

./eclat.sh $1 $2 $3

$1: input file, $2: minimum support, $3: output file



(a) (b)

Figure 2. An example of Eclat (a) Search tree and (b) Search tree with vertical TIDSET data presentation

Grading and Report

[30%] Correct implementation and result of Apriori frequent itemset mining.
[40%] Correct implementation and result of Eclat using vertical bitvector.
[30%] Report
(1) Briefly describe how you implement the algorithms, how you improve the performance or what you observed. **(15%)**
(2) Plot the exucution time of different minimum support (x-axis: minimum support, y-axis: execution time). For Apriori, plot support = 0.35, 0.3, 0.25, 0.2. For Eclat, plot support = 0.35, 0.3, 0.25, 0.2, 0.15, 0.1, 0.05. Plot the two lines in the same figure. **(15%)**

Submission

- Submit a zip file containing your code and report.pdf. Name the zip file to studentID.zip
The zip file must contain: apriori.sh, eclat.sh, report.pdf, your code.
- No Plagiarism. You have to implement the code by yourself (do not use other sample code on the Internet).
- Accept late submission for 2 days after the deadline.
- Wrong submitted format will get 10 points penalty.
- Late submission penalty is 15 points per day.
- It is your responsibility to make sure the submission is completed. Showing an unsubmitted set of homework after the due date will not work.

Reference

[1] Agrawal, R., Srikant, R. Fast algorithms for mining association rules. VLDB, 1994.
[2] M.J. Zaki., “Scalable Algorithm for Association Mining,” TKDE, 372-390, 2000.
[3] M.J. Zaki., “Fast Vertical Mining using Diffset,” SIGKDD, .326-335, 2003.