**Efficient Infrequent Pattern Mining Using Negative Itemset Tree**

**Yifeng Lu, Florian Richter and Thomas Seidl**

Problem:

* Existing frequent itemset mining algorithms can be used to mine infrequent pat- terns by setting the minimum support threshold to 1. However, those algorithms have to access all frequent patterns, which is undesired and time consuming. Also, the expensive candidate-generation step.
* Identifying all patterns with low support, has not been addressed efficiently yet.

Related work:

* inverted the idea of Apriori by defining a maximum support threshold.
* ARIMA - uses the pruned itemsets of Apriori in a first mining step to generate rare itemset candidates bottom-up in a second step.
* FRIMA - follows a bottom-up breadth-first traversing based approach.
* Mining patterns bottom-up on this transposed dataset is equivalent to top-down mining on the original dataset. AfRIM and Rarity also traverse the search space in a top-down manner. However, their performances are limited by the breadth-first traversing strategies.

Preliminaries:

* (NegativeItem) GiventhesetofitemsI={i1,i2,...,im}, the corresponding negative item of i ∈ I is denoted as ¬i.
* Neg-RepItemsetandNegativeItemset)GivenapositiveitemsetX= {x1,x2,...,xn} ⊆ I, its neg-rep (negative represented) itemset is the set of items that X does not have.
* (IntersectSupportandJointSupport)Givenanon-emptyitemsetX= {x1,...,xn}: TheintersectsupportofXinatransactiondatabaseDisthenumberoftransactions that contains all items of X
* The joint support of the negative itemset ¬X is defined in the corresponding neg- rep dataset D. It is the number of transactions that contains at least one item of ¬X

Proposed Approach:

Negative Itemset Tree Miner

Diagram

Description automatically generated

Evaluation:

Experiment conducted on four real datasets obtained rom the frequent itemset mining dataset repository (http://fimi.ua.ac.be/data/)

Algorithms compared:

LCMfreq, which represents bottom-up depth-first approach and NIIMiner with different nonexistent pattern skipping methods.

Table

Description automatically generated

* Given a small minimum support value minSup, the NIIMiner will access all patterns with support smaller than minSup. LCMfreq algorithm needs to traverse all patterns with support larger than 1 in D to generate the same set of low support patterns.
* Too slow to compare
* LCMfreq algorithm only traverse patterns with support larger than minSup -c, rather than 1.
* Comparison in terms of:

1. Runtime: NIIMiner is more efficient, since it has access to a huge number of high support patterns.
2. Runtime performance with respect to different dataset sizes: NIIMiner behaviors similar to the bottom- up based approach. The runtime of NIIMiner increased much slower than LCMfreq.