CLOSET : An Algorithm for Mining Frequent Closed Itemset

Youssef ELMIR[1]
Master Business Intelligence (MBI)
Faculty of Science and Technology, Sultan Moulay Slimane University
Po. Box 592, 23000 Beni Mellal, Morocco
[1]yssf.elmir@gmail.coms

***Abstract*— Association mining may often derive an undesirably large set of frequent itemsets and association rules. Recent studies have proposed an interesting alternative: mining frequent closed itemsets and their corresponding rules, which has the same power as association mining but substantially reduces the number of rules to be presented. In this paper, we propose an efficient algorithm, CLOSET, for mining closed itemsets, with the development of three techniques: (1) applying a compressed, frequent pattern tree FP-tree structure for mining closed itemsets without candidate generation, (2) developing a single prefix path compression technique to identify frequent closed itemsets quickly, and (3) exploring a partition-based projection mechanism for scalable mining in large databases. Our performance study shows that CLOSET is efficient and scalable over large databases, and is faster than the previously proposed methods.**

***Index Terms*— MEC, computation offloading, single task offloading, single user, offloading policy.**

I. Introduction

Data mining is the extraction of knowledge from large amounts of data, using automatic or semi-automatic methods. It proposes to use a set of algorithms from various scientific disciplines such as statistics, artificial intelligence or computer science, to build models from data, i.e., to find interesting structures or patterns according to previously fixed criteria, and to extract a maximum of knowledge from them. The industrial or operational use of this knowledge in the professional world allows to solve very diverse problems, from customer relationship management to preventive maintenance, from fraud detection to website optimization. It is also the way data journalism works. Data mining follows, in the escalation of the exploitation of company data, the decisional computing. The latter makes it possible to observe a fact, such as the turnover, and to explain it, such as the turnover broken down by product, whereas data mining makes it possible to classify the facts and to predict them to a certain extent, or to shed light on them by revealing, for example, the variables or parameters that could help us understand why the turnover of this point of sale is higher than that of another.

II. Definitions

Definition 1 (Itemset) : Is a non-empty subset of I where I represents the set of items. A transaction $t \in T$, with an identifier commonly denoted TID (Tuple IDentifier), contains a non-empty set of items of I. A subset I of I where $k = |I|$ is called a k-pattern or simply a pattern, and k represents the cardinality of I The number of transactions t in a database D containing a pattern I,$|\{t \in D | I \subset t\}|$, is called the absolute support of I and denoted in the following $Supp(I)$.

Definition 2 (Support of an Itemset) : The support of a pattern i is thus the ratio of the cardinality of the set of transactions that contain all items of I to the cardinality of the set of all transactions. It captures the scope of the pattern, by measuring its frequency of occurrence.

Definition 3 (Frequent Itemset) : An itemset is said to be frequent if its support is greater than or equal to a minimum threshold called minsup fixed apriori.

Types of frequent itemsets : The number of patterns present in a database is generally very large, it is common to obtain several million. Their use is classically based on a selection approach by frequency in order to keep only the most representative ones, but also because it is a pragmatic and efficient way to prune the search space. Nevertheless, this approach proves to be insufficient in many practical cases and shows its limits when the search space is large or the database is very dense and correlated. Condensed pattern representations provide a solution to the problem of frequent pattern extraction by providing a summary of the frequent patterns, while ensuring that all the frequent patterns can be reconstructed if necessary. In general, there are two types:

- **closed frequent itemset** : An itemset is closed frequent in a data set if there exists no superset that has the same support count as this original itemset.
- **maximum frequent itemset** : It is a frequent itemset for which none of its immediate supersets are frequent.

III. CLOSET Algorithm

CLOSET is an FP-tree based database projection method for efficient extraction of frequent closed etemsets in large databases. The algorithm proposes an original and efficient approach, called ”divide-and-regard”, for the discovery of frequent closed itemsets. The CLOSET algorithm proposed to adopt an advanced data structure, where the database can be compressed in order to complete the data mining process (e.g., the FP-tree structure). The idea behind this compact data structure, based on the notion of sorting, is that when several transactions share an item, then they can be merged by taking care of recording the number of occurrences of the items. The authors of the CLOSET algorithm do not present a buffer management technique. However, it is almost unrealistic to assume that all the information necessary for the process of discovering closed itemsets could fit in central memory.

Indeed, for weakly correlated bases, the tree representing the execution trace tends to increase in depth, while for dense bases, this tree tends to increase in width with little depth. This finding only contradicts the claims of the authors of the CLOSET algorithm. Indeed, the latter claim that their method only tests without generating candidates (”restricted testing only”). However, the increase in tree depth implicitly indicates that the CLOSET algorithm generates a large number of candidates, for which it should build subcontexts.

A. The process of the Closet algorithm

The CLOSET algorithm performs the extraction process of closed itemsets in two successive steps :

1) The transaction items are ordered by decreasing support. Then, the FP-Tree is built. This data structure is constructed as follows. First, the root node is created and labeled with ”root”. For each transaction in the context, the items are processed and a branch is created for each transaction. In each node of the FP-tree, there is a counter that keeps track of the number of occurrences of the item in the extraction context. Specifically in the case where a transaction has a common prefix with a branch of the FP-tree, then the counter of each node belonging to this prefix is incremented by 1 and a sub-branch will be created containing the rest of the items of the transaction.
2) Instead of a breadth-first exploration of candidate closed itemsets, it performs a partition of the search space to perform a depth-first exploration. Thus, it starts by considering frequent items and examines only their conditional subcontext. A conditional subcontext contains only those items that co-occur with the 1-itemset in question. The associated conditional FP tree is built and the process continues recursively.

Note that the Closet algorithm is essentially based on the following properties :

- the closed itemset, say p, that is extracted from a conditional subcontext is constituted by the concatenation of 1-itemsets that are as frequent as p.
- there is no need to develop a conditional subcontext of an itemset, say p, which is included in an already discovered closed itemset, say c, such that support(p)= $s$upport(c).

B. The algorithm

C. The algorithm of the MINE-FERME function

D. Example

Transaction ID	Items
10	a,c,d,e,f
20	a,b,e
30	c,e,f
40	a,c,d,f

1) Find frequent articles ($min_sup = 2$) : List of frequent items in support descending order $f_list =< c : 4, e : 4, f : 4, a : 3, d : 2 >$

Algorithm 1 An algorithm with caption

```
Require: $K$ : extraction context, minsup
Ensure: $FC = UkFCk$ : Set of closed itemsets
  $FC = \emptyset$
  ▷ Sort extraction context K with decreasing support value
  $K = TRI(K)$
  for any i 2 Freq1 (in ascending order) do
      $FCi = MINE - FERME(i, K)$
                              ▷ MINE-FERME constructs the
  conditional subcontext of i and performs a walk In depth
  first by recursive calls
  for the extraction of itemsets Frequent closed
      $FC = FC \cup FCi$
  end for
  returnFC
```

Algorithm 2 An algorithm with caption

```
Require: 1-Tree: the condensed conditional tree of item I,
  minSup
Ensure: The updated CFI tree
  if $(support(I)$ ¿ $minSup)$ then
      $Sum \leftarrow \emptyset$                ▷ Set of processed items
      for any item Ii belongs I-Tree and Ii does not belong to
  s  do
          $Sum \leftarrow$ Ii
          Construct Ii Tree         ▷ the condensed conditional
  CATS tree of item Ii
          MINE-FERME (Ii -Tree, minSup)    ▷ Generate all
  frequent closed itemsets
          for any item Ii belongs I-Tree and Ii does not belong
  to s  do
              if $(XisnotincludedinCFI)$ then Update the
  CFI tree
              end if
          end for
      end for
  end if
  return$theupdatedCFItree$
```

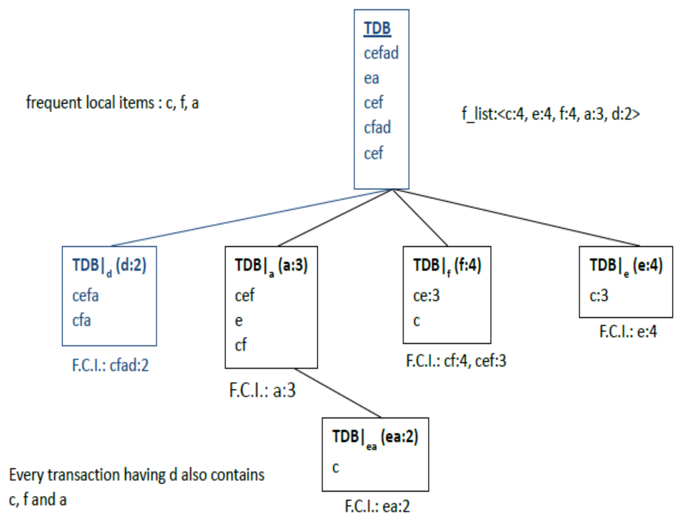
2) Divide Search Space: All frequent closed itemsets can be divided into 5 non-overlap subsets based on f_list

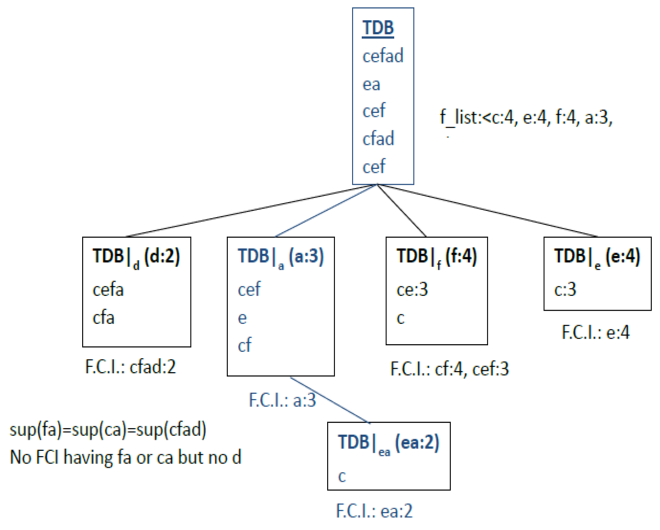
- The ones containing d
- The ones containing a but no d
- The ones containing f but no a nor d
- The ones containing e but no f, a nor d
- The ones containing only c

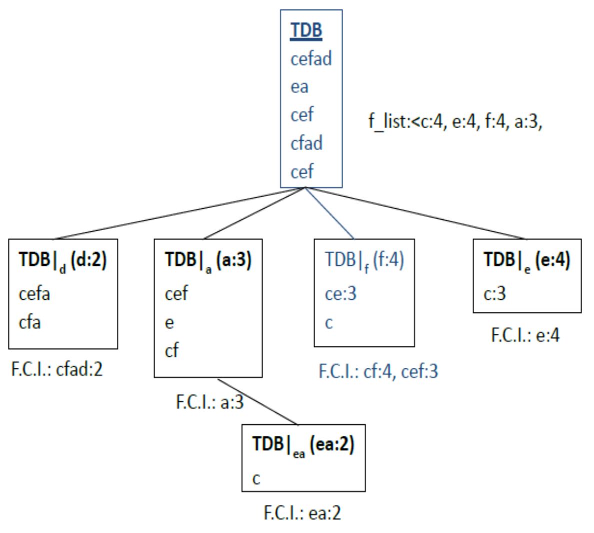
3) Find Subsets of Frequent Closed Itemsets by Constructing Conditional Databases: Let a be a frequent item in TDB. The a-conditional database, denoted as TDB—a, is the subset of transactions in TDB containing a, and all occurrences of infrequent items, item a, and items following a in f_list are omitted.

4) Find Frequent Closed Itemsets Containing d:

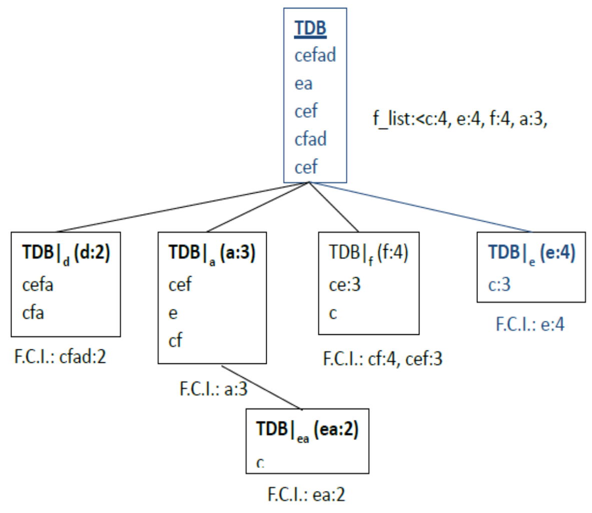
5) Find Frequent Closed Itemsets Containing a but no d:







6) Find Frequent Closed Itemsets Containing f but no a nor d:



7) Find Frequent Closed Itemsets Containing e but no f, a nor d:

8) Find Frequent Closed Itemsets Containing Only c:

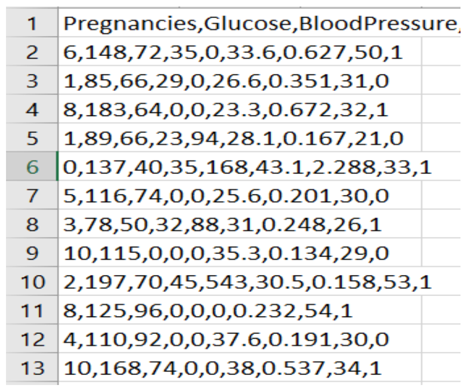
$$sup(c) = sup(cf)$$

c is not a closed itemset In summary, the set of frequent closed itemsets is $acdf : 2, a : 3, ae : 2, cf : 4, cef : 3, e : 4$

E. Extraction of Frequent Closed Itemsets from a Dataset

1) About Dataset : **Context**
This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The objective of the dataset is to diagnostically predict whether or not a patient has diabetes, based on certain diagnostic measurements included in the dataset. Several constraints were placed on the selection of these instances from a larger database. In particular, all patients here are females at least 21 years old of Pima Indian heritage.

Content
The datasets consist of several medical predictor variables and one target variable, Outcome. Predictor variables includes the number of pregnancies the patient has had, their BMI, insulin level, age, and so on.



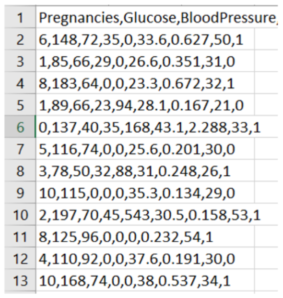
1	Pregnancies,Glucose,BloodPressure,
2	6,148,72,35,0,33.6,0.627,50,1
3	1,85,66,29,0,26.6,0.351,31,0
4	8,183,64,0,0,23.3,0.672,32,1
5	1,89,66,23,94,28.1,0.167,21,0
6	0,137,40,35,168,43.1,2.288,33,1
7	5,116,74,0,0,25.6,0.201,30,0
8	3,78,50,32,88,31,0.248,26,1
9	10,115,0,0,0,35.3,0.134,29,0
10	2,197,70,45,543,30.5,0.158,53,1
11	8,125,96,0,0,0,0.232,54,1
12	4,110,92,0,0,37.6,0.191,30,0
13	10,168,74,0,0,38,0.537,34,1

Acknowledgements
Smith, J.W., Everhart, J.E., Dickson, W.C., Knowler, W.C., & Johannes, R.S. (1988). Using the ADAP learning algorithm to forecast the onset of diabetes mellitus. In Proceedings of the Symposium on Computer Applications and Medical Care (pp. 261–265). IEEE Computer Society Press.

Transformation of the dataset
The extraction of the frequent closed elements with the Closet algorithm, consists in using transactional data, but the diabetes dataset contains only numerical values. To do this we must transform the dataset to transactional data, this transformation consists in dividing each feature into domains that group several individuals:

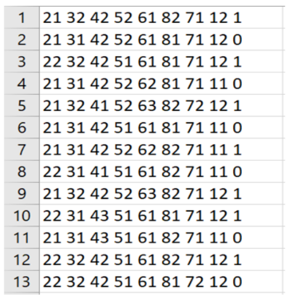
Variables	Domain
1 : Age	11[0,30], 12[30,80]
2 : Pregnacies	21[0,7], 22[7,17]
3 : Glucose	31[0,125], 32[12,200]
4 : Blood pressure	41[0,40], 42[40,90], 43[90,120]
5:Skin Thickeness	51[0,8], 52[8,45], 53[45,60]
6 : Insulin	61[0,30], 62[30,150], 63[150,800]
7 : Diabetes Pedigree Function	71[0,0.8], 72[0.8,125]
8 : BMI	81[0,30], 82[30,60]

After this transformation the dataset will be represented as follows :



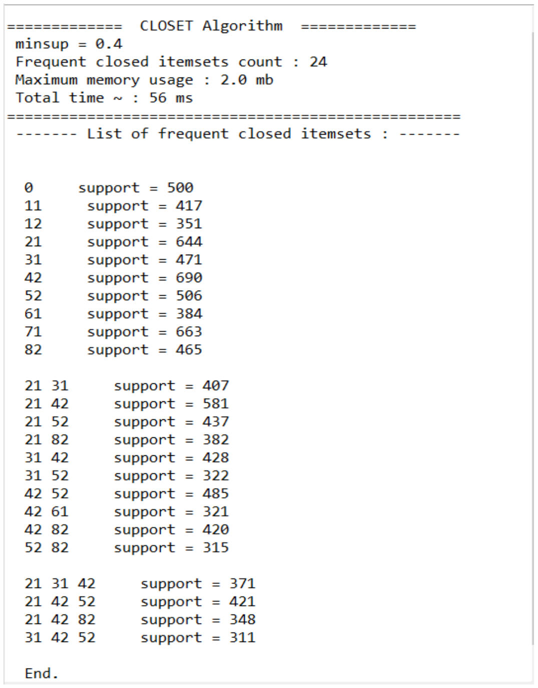
```
Pregnancies,Glucose,BloodPressure,
6,148,72,35,0,33.6,0.627,50,1
1,85,66,29,0,26.6,0.351,31,0
8,183,64,0,0,23.3,0.672,32,1
1,89,66,23,94,28.1,0.167,21,0
0,137,40,35,168,43.1,2.288,33,1
5,116,74,0,0,25.6,0.201,30,0
3,78,50,32,88,31,0.248,26,1
10,115,0,0,0,35.3,0.134,29,0
2,197,70,45,543,30.5,0.158,53,1
8,125,96,0,0,0,0.232,54,1
4,110,92,0,0,37.6,0.191,30,0
10,168,74,0,0,38,0.537,34,1
```

→



```
21 32 42 52 61 82 71 12 1
21 31 42 52 61 81 71 12 0
22 32 42 51 61 81 71 12 1
21 31 42 52 62 81 71 11 0
21 32 41 52 63 82 72 12 1
21 31 42 51 61 81 71 11 0
21 31 42 52 62 82 71 11 1
22 31 41 51 61 82 71 11 0
21 32 42 52 63 82 71 12 1
22 31 43 51 61 81 71 12 1
21 31 43 51 61 82 71 11 0
22 32 42 51 61 82 71 12 1
22 32 42 51 61 81 72 12 0
```

The result Extraction of frequent closed itemsets from the dateset with the Closet algorithm



```
=============  CLOSET Algorithm  =============
 minsup = 0.4
 Frequent closed itemsets count : 24
 Maximum memory usage : 2.0 mb
 Total time ~ : 56 ms
===================================================
 ------- List of frequent closed itemsets : -------


  0     support = 500
  11     support = 417
  12     support = 351
  21     support = 644
  31     support = 471
  42     support = 690
  52     support = 506
  61     support = 384
  71     support = 663
  82     support = 465

  21 31     support = 407
  21 42     support = 581
  21 52     support = 437
  21 82     support = 382
  31 42     support = 428
  31 52     support = 322
  42 52     support = 485
  42 61     support = 321
  42 82     support = 420
  52 82     support = 315

  21 31 42     support = 371
  21 42 52     support = 421
  21 42 82     support = 348
  31 42 52     support = 311

  End.
```

Variation of minsup and comparison

minsup	Execution time	Memory usage	Number of itemsets generated
2%	68 ms	2.0 mb	90
4%	56 ms	2.0 mb	24
6%	42 ms	2.0 mb	9
8%	38 ms	2.0 mb	3

Comparison of the Closet with the Charm algorithm

Algorithm	Output itemsets generate	Technical storage	Execution time	Memory usage	Number of itemsets generated
Closet	frequent closed itemsets	Divide and conquer	68 ms	2.0 mb	24
Charm	frequent closed itemsets	Test and generate	28 ms	2.26 mb	69

IV. Conclusion

Mining complete set of itemsets often suers from generating a very large number of itemsets and association rules. Mining frequent closed itemsets provides an interesting alternative since it inherits the same analytical power as mining the whole set of frequent itemsets but generates a much smaller set of frequent itemsets and leads to less and more interesting association rules than the former. In this paper, we proposed an FP-tree-based database projection method, CLOSET, for ecient mining of frequent closed itemsets in large databases. Our proposed algorithm, CLOSET, for mining closed itemsets adopts three techniques: (1) applying a compressed, frequent pattern tree FP-tree structure for mining closed itemsets without candidate generation, (2) developing a single prex path compression technique to identify frequent closed itemsets quickly, and (3) exploring a partition-based projection mechanism for scalable mining in large databases. The advantage of the algorithm is that it allows to improve the response time by taking into account the nature of the database being explored.