A Top-Down Row Enumeration Approach of Frequent Patterns from Very High Dimensional Data

Jiaofen Xu

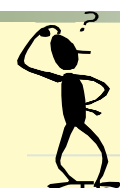
UNIVERSITY OF ALBERTA

The Presentaion-Based Paper

- **Top-Down Mining of Frequent Patterns from Very High Dimensional Data.**
- **Hongyan Liu, Jiawei Han, Dong Xin and Zheng Shao**

Outline

- **Introduction**
- **Preliminaries**
- **Algorithm**
- **Experimental Study**
- **Conclusion**

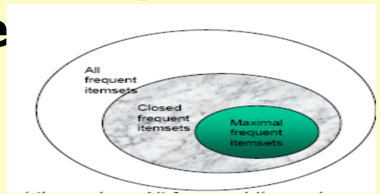


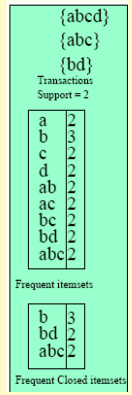
What is high dimensional data?

- The dimension of the data being in the hundreds or thousands. e.g. in text/web mining and bioinformatics.
- A specific kind of high dimensional data set, which contain as and a large number of tuples. many as tens of thousands of columns but only a hundred or a thousand rows, such as microarray data.
- Different from transactional data set, which usually have a small number of columns and a large number of rows.

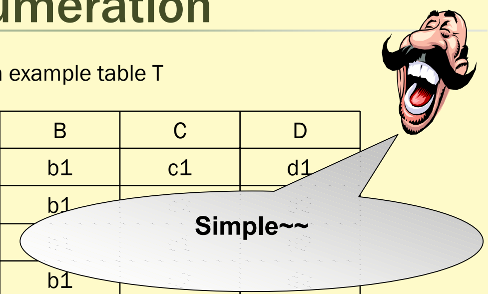
Frequent Pattern Mining
Frequent Close Pattern Mining

For frequent itemset X, if there exists no item y such that every transaction containing X also contains y, then X is a freque... pattern.





Column enumeration & row enumeration



An example table T

	A	B	C	D
1	a1	b1	c1	d1
2	a1	b1		
3	a1			
4	a2	b1		
5	a2	b2	c2	d3

a1, a2, b1, b2, c1, c2, d1, d2, d3
1, 2, 3, 4, 5

a1b1, a1b2, a1c1, a1d1, a1d2, a1d3 , ...
12, 13, 14, 15, 23, 24, 25, ...

a1b1c1, a1b1c2, a1c1d2, a1c1d3, a1c2d3, ...
123, 124, 125, 134, 135, 145, 234, 235, 245, ...

tions

Why are the current

Would row enumeration-based method generate less?

The other reason is that with just a small number Of rows (samples), column-enumeration methods cannot get sufficient support to generate frequent pattern.

For 55555 markers, the number of possible frequent patterns is 2^{55555}.

State of the art

- Bottom-up row enumeration-based method
  F. Pan, G. cong, A.K.H. Tung, J. Yang, and M.J. Zaki. CARPENTER: Finding closed patterns in long biological datasets. In *Proc. 2003 ACM SIGKDD Int. conf.*
  However, the bottom-up search strategy checks row combinations from the smallest to the largest, it cannot make full use of the minimum support threshold to prune search space.
- Top-down row enumeration-based method

Contributions of the paper

- A top-down search method is proposed to take advantage of the pruning power of minimum support threshold, which can cut down the search space dramatically.
- A new method, call closeness-checking, is developed to check ciently and effectively whether a pattern i ed. It does not need to scan the mining the result set, and is down se

This is critical for mining high dimensional
data, becau
an
s

In CARPENTER,
the closeness-checking method
is that before outputting each itemset found currently,
we must check if it is already found before.
If not, output it. Otherwise, discard it.

Outline

- **Introduction**
- **Preliminaries**
- **Algorithm**
- **Experimental Study**
- **Conclusion**

Table and transposed table

Original table T

minsup =2

	A	B	C	D
1	a1	b1	c1	d1
2	a1	b1	c2	d2
3	a1	b1	c1	d2
4	a2	b1	c2	d2
5	a2	b2	c2	d3

Transposed table TT

itemset	rowset
a1	1, 2, 3
a2	4, 5
b1	1, 2, 3, 4
c1	1, 3
c2	2, 4, 5
d2	2, 3, 4

Table TT is already pruned by minsup. For clarity, we

Closed itemset and closed rowset

- Definition 1 (Closure): Given an itemset I and a rowset S, define

$$r(I) = \{r_i \in S \mid \forall i_j \in I, (r_i, i_j) \in \mathcal{R}\}$$

$$i(S) = \{ i_j \in I \mid \forall r_i \in S, (r_i, i_j) \in \mathcal{R} \}$$

- Based on these definitions, we define C(I) as the closure of an itemset I, and C(S) as the closure of a rowset S as follows:

$$C(I) = i(r(I))$$

$$C(S) = r(i(S))$$

- Definition 2 (Closed itemset and closed rowset): An itemset I is called a *closed itemset* iff I=C(I). Likewise, a rowset S is called a *closed rowset* iff S= C(S).
- Definition 3 (Frequent itemset and large rowset): Given *minsup*, an itemset I is called *frequent* if |r(I)| ≥ minsup, where |r(I)| is called the support of itemset I, and a roset S is called large if |S| ≥ minsup, where |S| is called the size of rowset S.
- Further, an itemset I is called frequent closed itemset if it is both closed and frequent. Likewise, a rowset S is called large closed rowset if it is both closed and large.

Example

- For an itemset {b1, c2}, r({b1, c2})= {2, 4}, and i({2, 4})={b1,c2, d2}, so C({b1, c2}= {b1, c2, d2}. Therefore, {b1, c2} is not a closed itemset. If minsup=2, it is a frequent itemset.
- For an rowset {1, 2}, i({1, 2})={a1, b1} and r({a1, b1})={1, 2, 3}, then C({1,2})={1, 2, 3}. So rowset {1, 2} is not a closed rowset, but apparently {1, 2, 3} is.

itemset	rowset
a1	1, 2, 3
a2	4, 5
b1	1, 2, 3, 4
c1	1, 3
c2	2, 4, 5
d2	2, 3, 4

$r(I) = \{r_i \in S \mid \forall i_j \in I, (r_i, i_j) \in \mathcal{R}\}$

$i(S) = \{ i_j \in I \mid \forall r_i \in S, (r_i, i_j) \in \mathcal{R} \}$

$C(I) = i(r(I))$
$C(S) = r(i(S))$

Mining Task

- Originally, we want to find all of the frequent closed itemsets which satisfy the minimum support threshold minsup form the original table T.
- After transposing T to transposed table TT, the mining task becomes finding all of the large closed rowsets which satisfy minimum size threshold minsup from table TT.

Top-down Search Strategy

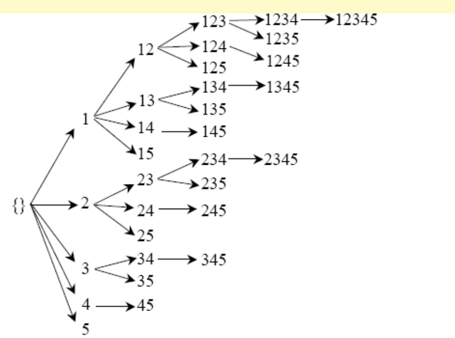


Figure 3.1 Bottom-up row enumeration tree

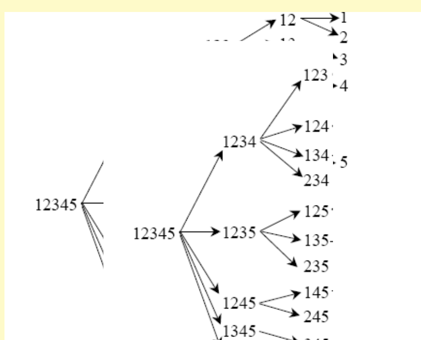


Figure 3 ation tree

Given user specified *minsup*, we can stop further search of the tow-down row enumeration tree at level (n-minsup) for mining frequent itemsets.

minsup =3

X-excluded transposed table

- Each node of the tree in Figure 3.2 corresponds to a sub-table. For example, the root represents the whole table TT, and then it can be divided into 5 sub-tables: table without rid 5, table with 5 but without 4, table with 45 but without 3, table with 345 but without 2, and table with 2345 but without 1.
- Definition (x-excluded transposed table) : Given a rowset x={r^{i1}, r^{i2}, …, r^{ik}} with an order such that r^{i1}> r^{i2}>…> r^{ik}, an minsup and its parent table $TT|_p$, an x-excluded transposed table TT is a table in which each tuple contains rids le than any of rids in x, and at the same time contains all of the rids greater than any of in x. Rowset x is called an exclud

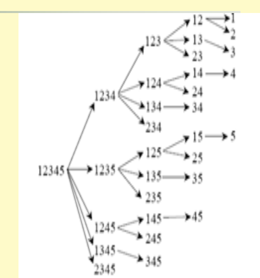


Figure 3.2 Top-down row enumeration t

Tables corresponding to a parent node and a child node are called parent table and child table respectively.

Example

itemset	rowset
a1	1, 2, 3
a2	4, 5
b1	1, 2, 3, 4
c1	1, 3
c2	2, 4, 5
d2	2, 3, 4

minsup =2

Table 3.1 TT|54

itemset	rowset
a_1	1, 2, 3
b_1	1, 2, 3
c_1	1, 3
d_2	2, 3

In $TT|_{54}$, x={5, 4}, each tuple only contains rids which are less than 4, and contains at least two such rids as minsup is 2.

Table 3.2 $TT|_4$

itemset	rowset
c_2	2

In $TT|_4$, each tuple must contain rid 5 as it is greater than 4, and in the meantime must contain at least one rid less than 4 as minsup is 2. As a result, in Table TT, only those tuples containing rid 5 can be a candidate tuple of $TT|_4$.

Excluded row enumeration tree

- Extract form TT or its direct parent table $TT|_p$ each tuple containing all rids greater than r_{ik}.
- For each tuple obtained in the first step, keep only rids less than r_{ik}.
- Get rid of tuples containing less than (minsup-j) number of rids, where j is the number of rids greater than r_{ik} in S.

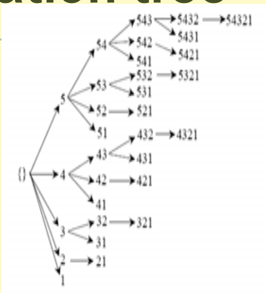


Figure 3.3 Excluded row enumeration tree

Figure 3.2 Top-down row enumeration tree

Closeness-checking

- Lemma 1: In transposed table TT, a rowset S is closed iff it can be represented by an intersection of a set of tuples, that is:

$$\exists I \subseteq I, \text{ s.t. } S = r(I) = \cap_j r(\{i_j\})$$
$$\text{where } i_j \in I, \text{ and } I = \{i_1, i_2, \ldots, i_l\}$$

- Lemma 2: Given a rowset S in transposed table TT, for every tuple i_j containing S, which means $i_j \in i(S)$, if $S \neq \cap r(\{i_j\})$, where $i_j \in i(S)$, then S is not closed.
- Lemma 1 and Lemma 2 are the basis of our closeness-checking method.

Skip-rowset

- The so-called skip-rowset is a set of rids which keeps track of the rids that are excluded from the same tuple of all of its parent tables.
- When two tuples in an x-excluded transposed table have the same rowset, they will be merged to one tuple, and the intersection of corresponding two skip-rowsets will become the current skip-rowset.

Example of skip-rowset and merge of x-excluded transposed table

Table 3.3 TT|5

itemset	rowset
a_1	1, 2, 3
b_1	1, 2, 3, 4
c_1	1, 3
c_2	2, 4
d_2	2, 3, 4

Table 4.1 $TT|_{54}$ with *skip-rowset*

itemset	rowset	*skip-rowset*
a_1	1, 2, 3	
b_1	1, 2, 3	4
c_1	1, 3	
d_2	2, 3	4

Table 4.2 TT|54 after merge

itemset	rowset	*skip-rowset*
a_1 b_1	1, 2, 3	
c_1	1, 3	
d_2	2, 3	4

- When we got TT|54 from its parent TT|5, we excluded rid 4 from tuple b1 and d2 respectively.
- The first 2 tuples have the same rowset {1, 2, 3}. The skip-rowset of this rowset becomes empty because the intersection of an empty set and any other set is still empty.
- If the intersection result is empty, it means that currently this rowset is the result of intersection of two tuples. Therefore, it must be a closed rowset.

Outline

- Introduction
- Preliminaries
- Algorithm
- Experimental Study
- Conclusion

TD-Close

Algorithm *TD-Close*
Input: Table *T*, and minimum support threshold, *minsup*
Output: A complete set of frequent closed patterns, *FCP*
Method:
1. Transform *T* into transposed table *TT*
2. Initialize *FCP* = Φ and *excludedSize* = 0
3. Call TopDownMine($TT|_{\Phi}$, *minsup*, *excludedSize*)

Subroutine TopDownMine($TT|_x$, *cMinsup*, *excludedSize*)
Method:
1. **Pruning** 1: if *excludedSize* >= (*n*—*minsup*) return;
2. **Pruning** 2: If the size of $TT|_x$ is 1, output the corresponding itemset if the rowset is closed, and then return.
3. **Pruning** 3: Derive $TT|_{x\cup y}$ and $TT|_x{}'$, where
   y is the largest rid among rids in tuples of $TT|_x$,
   $TT|_x{}'$ = {tuple t_i | $t_i \in TT|_x$ and t_i contains y},
   $TT|_{x\cup y}$ = {tuple t_i | $t_i \in TT|_x$ and if t_i contains *y*, size of t_i must be greater than *cMinsup*}
   Note, we delete *y* from both $TT|_{x\cup y}$ and $TT|_x{}'$.
4. **Output**: Add to *FCI* itemset corresponding to each rowset in $TT|_{x\cup y}$ with the largest size *k* and ending with rid *k*.
5. Recursive call:
   TopDownMine($TT|_{x\cup y}$, *cMinsup*, *excluedSize*+1)
   TopDownMine($TT|_x{}'$, *cMinsup*—1, *excluedSize*)

Figure 4.1 Algorithm *TD-Close*

Table 3.2 $TT|_4$

itemset	rowset
c_2	2

Table 4.2 TT|54 after merge

itemset	rowset	*skip-rowset*
a_1 b_1	1, 2, 3	
c_1	1, 3	
d_2	2, 3	4

minsup =2

Table TT|543

itemset	rowset	skip-rowset
a1 b1	1, 2	{3}

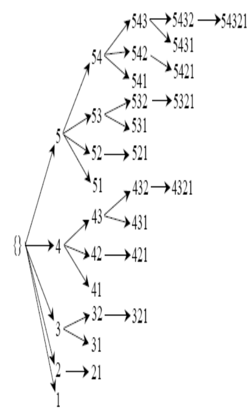


Figure 3.3 Excluded row enumeration tree

Outline

- Introduction
- Preliminaries
- Algorithm
- Experimental Study
- Conclusion

Experimental Study

- Compare the ... and FPclose.
- Using D#T#C ... taset, where D# stands for dimension, the number of attributes of each data set, T# for number of tuples, and C# for cardinality, the number of values per dimension (or attribute).
- In these experiments, D# ranges from 4000 to 10000, T# varies from 100, 150 to 200, and C# varies from 8, 10 to 12.

FPclose is a column enumeration-based algorithm, which won the FIMI' 03 best implementation award.

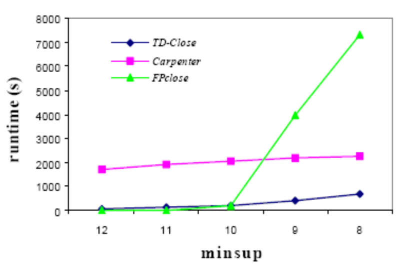


Figure 5.1 Runtime for D4000T100C10

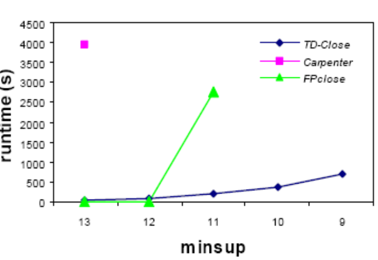


Figure 5.2 Runtime for D6000T100C10

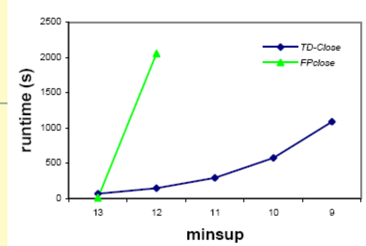


Figure 5.3 Runtime for D8000T100C10

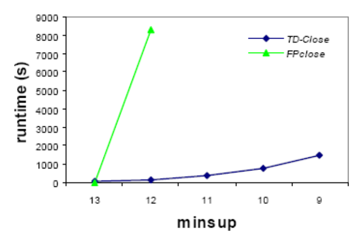


Figure 5.4 Runtime for D10000T100C10

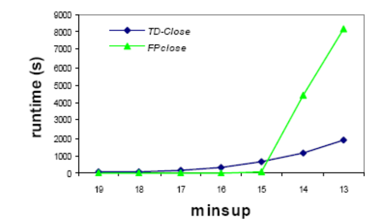


Figure 5.5 Runtime for D4000T150C10

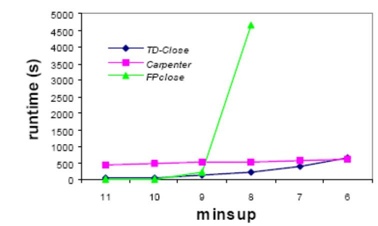


Figure 5.7 Runtime for D4000T100C12

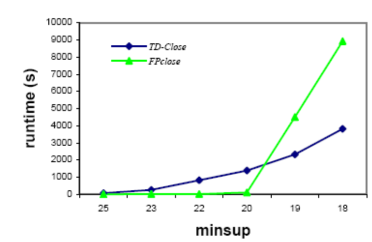


Figure 5.6 Runtime for D4000T200C10

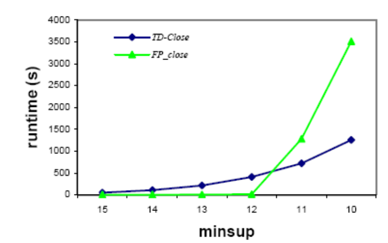


Figure 5.8 Runtime for D4000T100C8

Outline

- Introduction
- Preliminaries
- Algorithm
- Experimental Study
- Conclusion

Conclusion

- Existing algorithms, such as Carpenter, adopt a bottom-up fashion to search the row enumeration space, which makes the pruning power of minimum support threshold (minsup) very weak, and therefore results in long mining process even for high minsup, and much memory cost.
- To solve this problem, a top-down style row enumeration method and an effective closeness-checking method are proposed in this paper. Several pruning strategies are developed to speed up searching.
- Both analysis and experimental study show that this method is effective and useful.

Future work

- Integrating top-down row enumeration method and column row enumeration method for frequent pattern mining from both long and deep large datasets.
- Mining classification rules based on association rules using top-down searching strategy.

Questions

?

Thank you~