Agency for Science, Technology and Research
SINGAPORE

Nanyang Technological University

Evolution and Maintenance of Frequent Pattern Space when Transactions are Removed

Mengling Feng, Guozhu Dong, Jinyan Li, Yap-Peng Tan, Limsoon Wong

Motive

- Update
  - Fundamental activity of data management
- Transaction removal
  - Most common update [1]
- Previous work
  - Mainly focus on incremental updates
  - Few on transaction removal updates
    - E.g. FUP2, Border, ZIGZAG

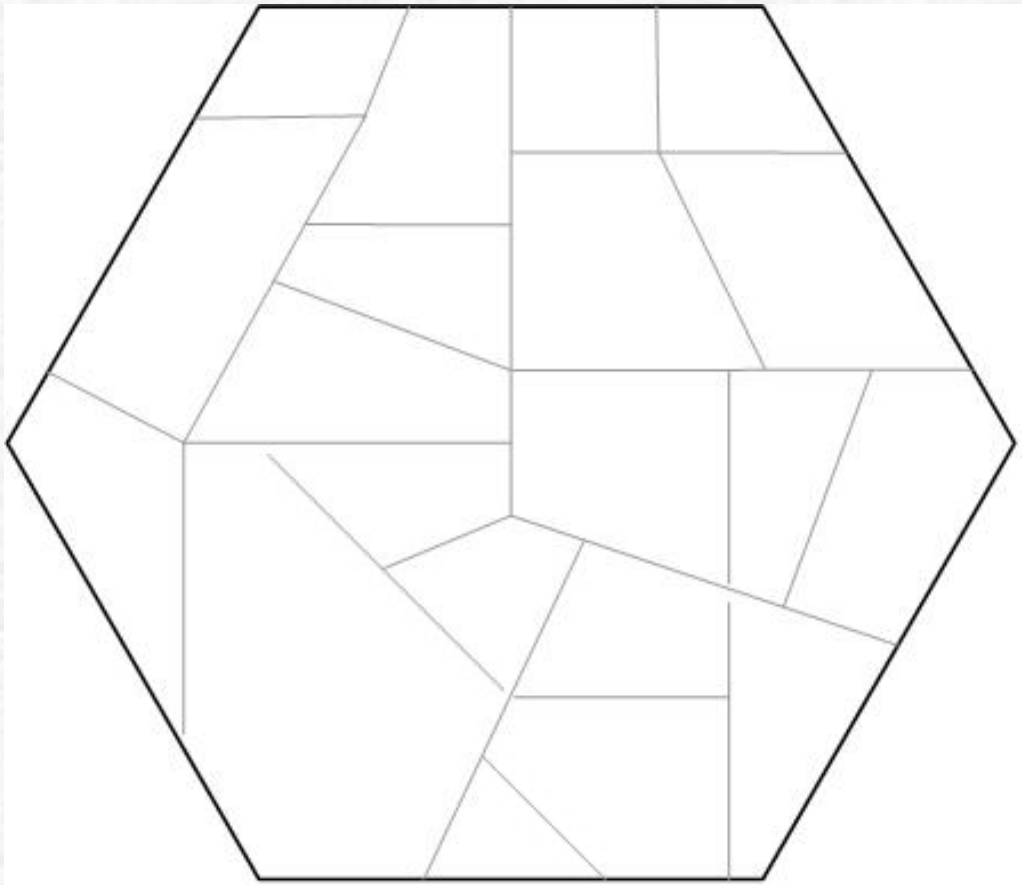
[1] S. Zhang, et al. A decremental algorithm for maintaining frequent itemsets in dynamic databases. In *DaWak*, 2005.

Terminology

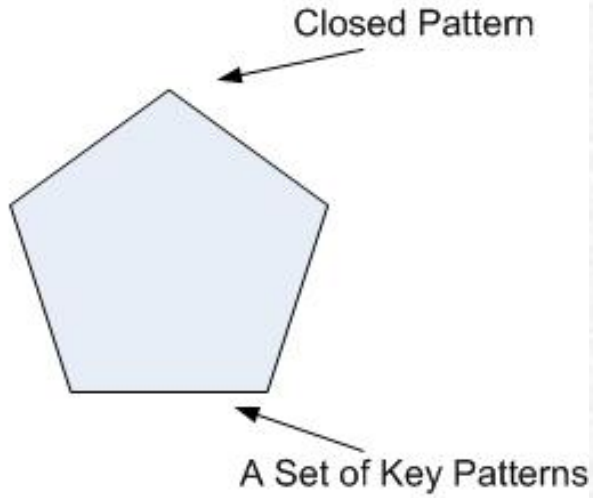
Sample Dataset
a, b, c, d
b, d
a, c, d
a, c
b

- Dataset, transaction & item
- Pattern (itemset) & support
- **Frequent pattern** & Frequent pattern Space
- Support threshold
  - Absolute: sup(P) >= ms
  - Percentage: sup(P)/|D| >= $ms_{\%}$

Frequent Pattern Space

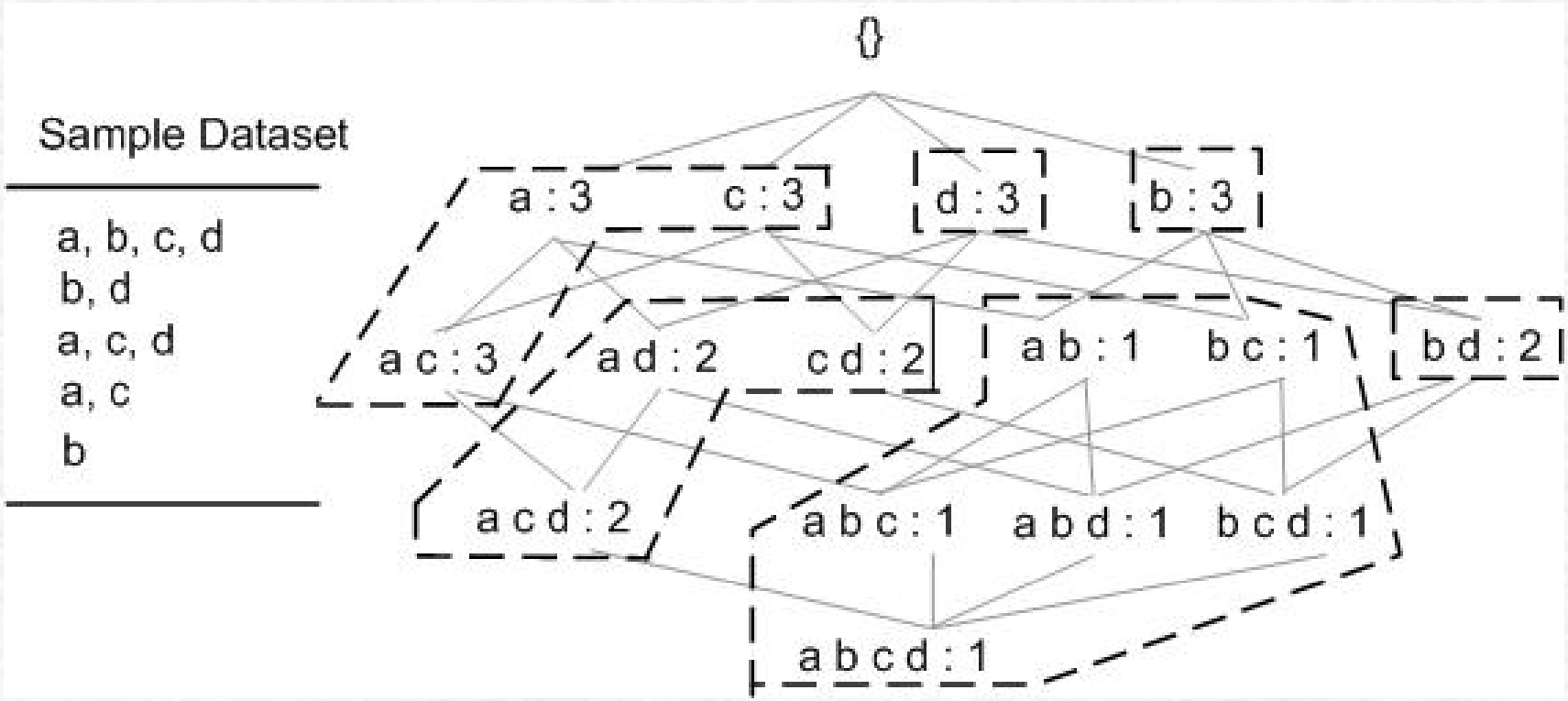


Frequent Pattern Space

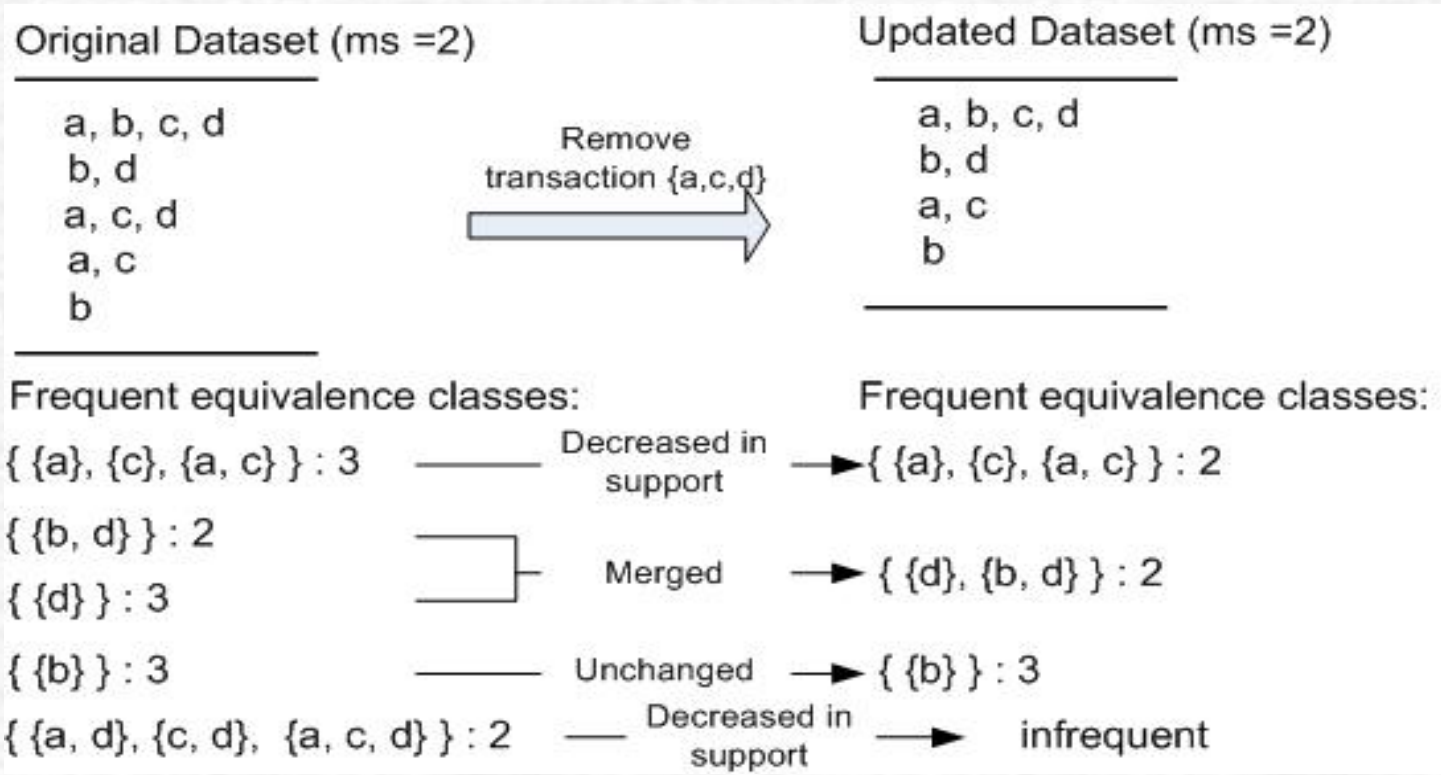


Equivalence Class

Example

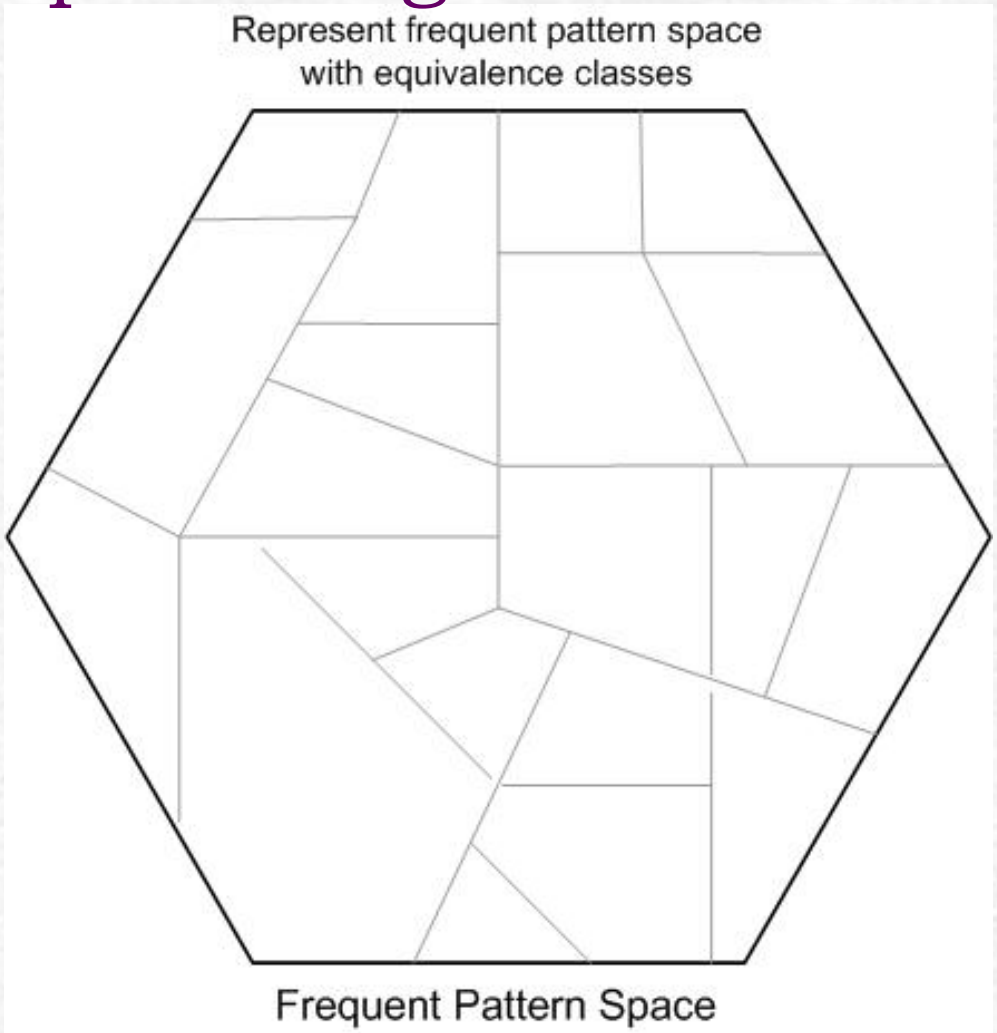

Sample Dataset
a, b, c, d
b, d
a, c, d
a, c
b
{}
a : 3
c : 3
d : 3
b : 3
a c : 3
a d : 2
c d : 2
a b : 1
b c : 1
b d : 2
a c d : 2
a b c : 1
a b d : 1
b c d : 1
a b c d : 1

Evolution of Pattern Space

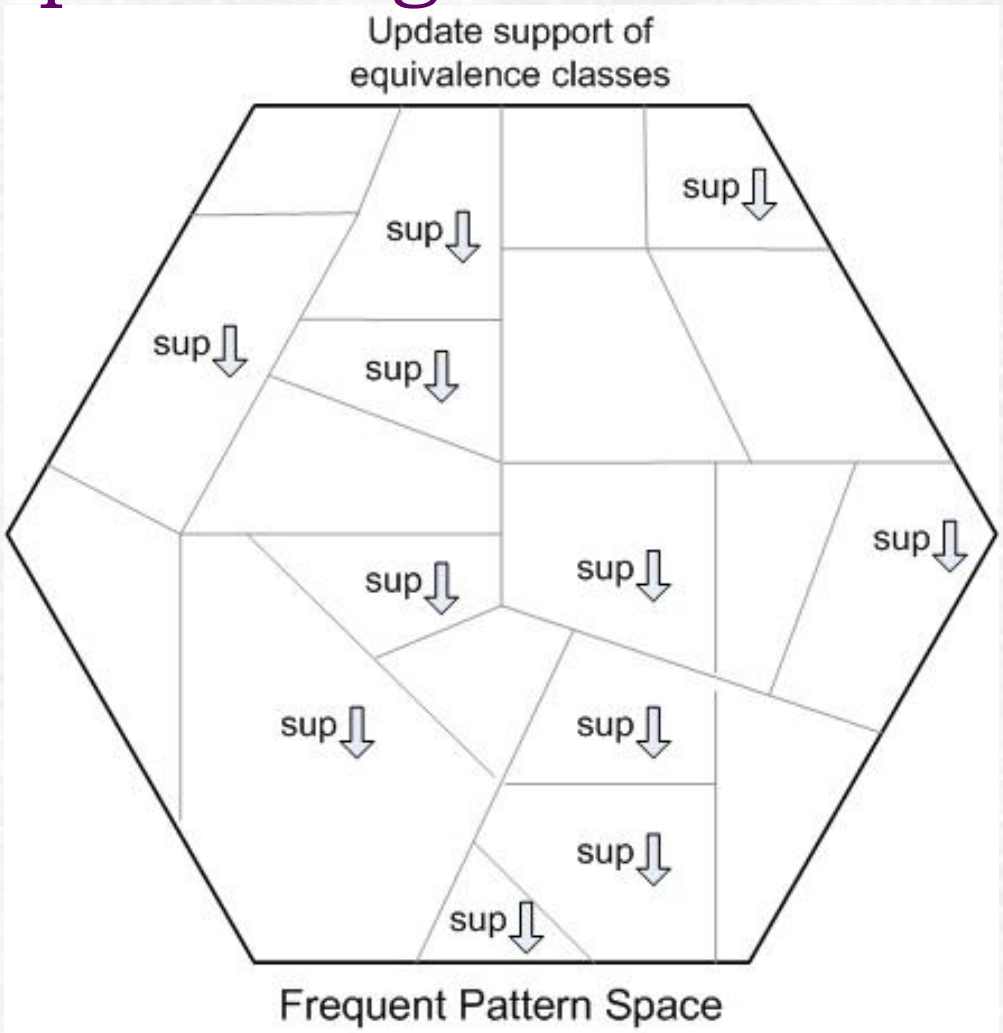


- Merging Criteria: given two equivalence classes P & Q, $P_{upd} = Q_{upd} = P \cup Q$ iff $sup(P_{upd}) = sup(Q_{upd})$ and P.closed $\supseteq$ Q.closed or P.closed $\subseteq$ Q.closed.

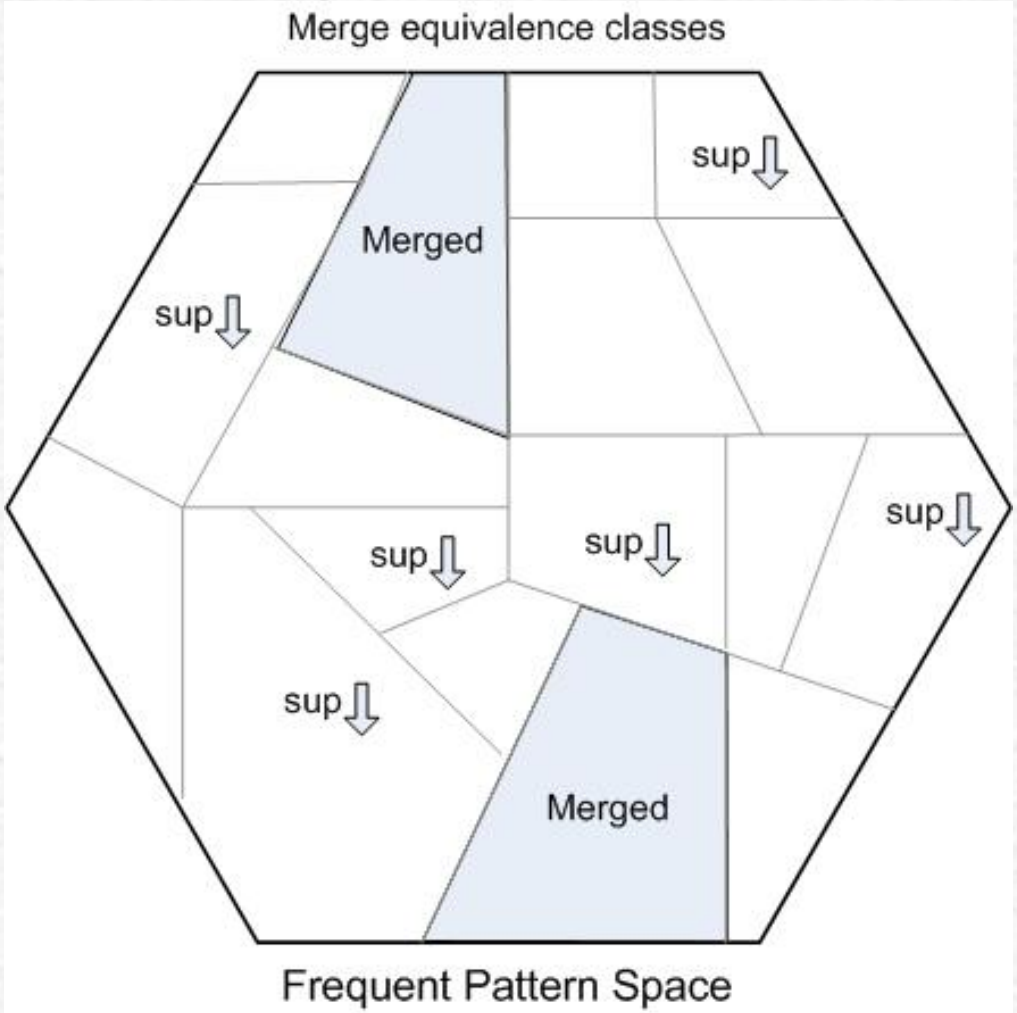
Proposed Algorithm: TRUM



Proposed Algorithm: TRUM



Proposed Algorithm: TRUM



Implementation: Hash

Original Dataset (ms =2)

Tid	Transactions
1	a, b, c, d
2	b, d
3	a, c, d
4	a, c
5	b

Frequent ECs:

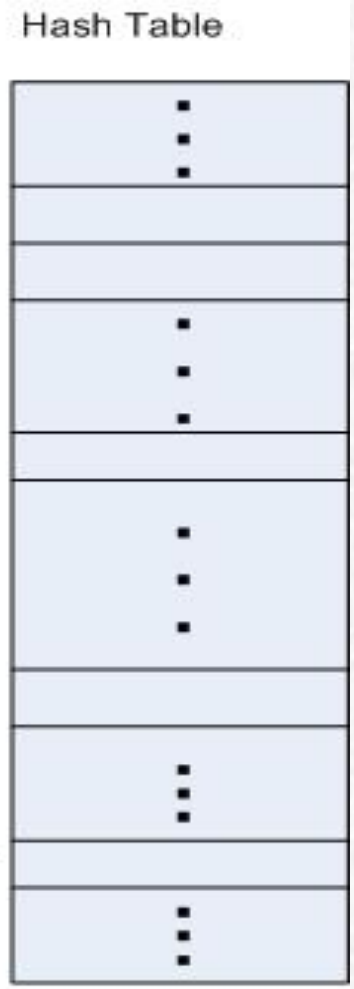
EC_1: { {a}, {c}, {a, c} } : 3
Tid-list <1,3,4>

EC_2: { {b, d} } : 2
Tid-list <1,2>

EC_3: { {d} } : 3
Tid-list <1,2,3>

EC_4: { {b} } : 3
Tid-list <1,2,5>

EC_5: { {a, d}, {c, d}, {a, c, d} } : 2
Tid-list <1,3>



Implementation: Tid-Tree

Frequent ECs (ms =2):

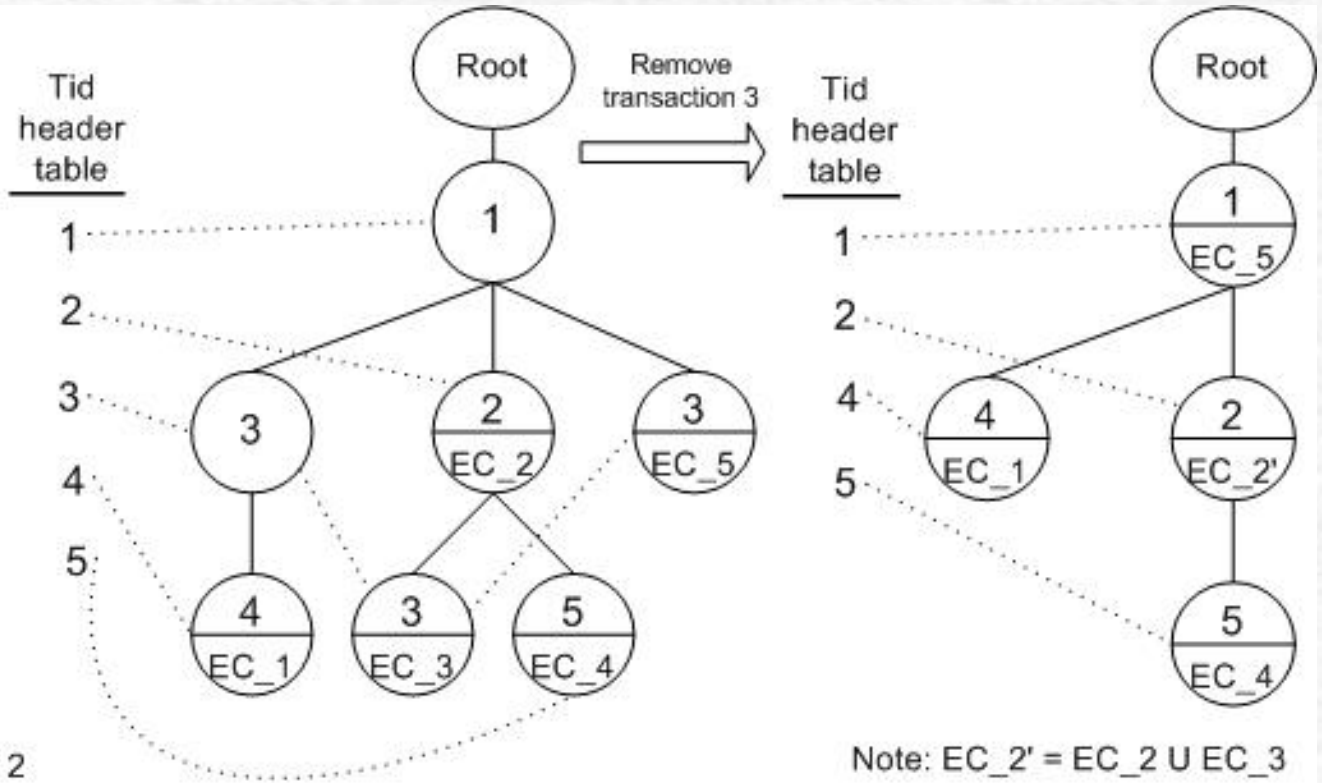
EC_1: { {a}, {c}, {a, c} } : 3
Tid-list <1,3,4>

EC_2: { {b, d} } : 2
Tid-list <1,2>

EC_3: { {d} } : 3
Tid-list <1,2,3>

EC_4: { {b} } : 3
Tid-list <1,2,5>

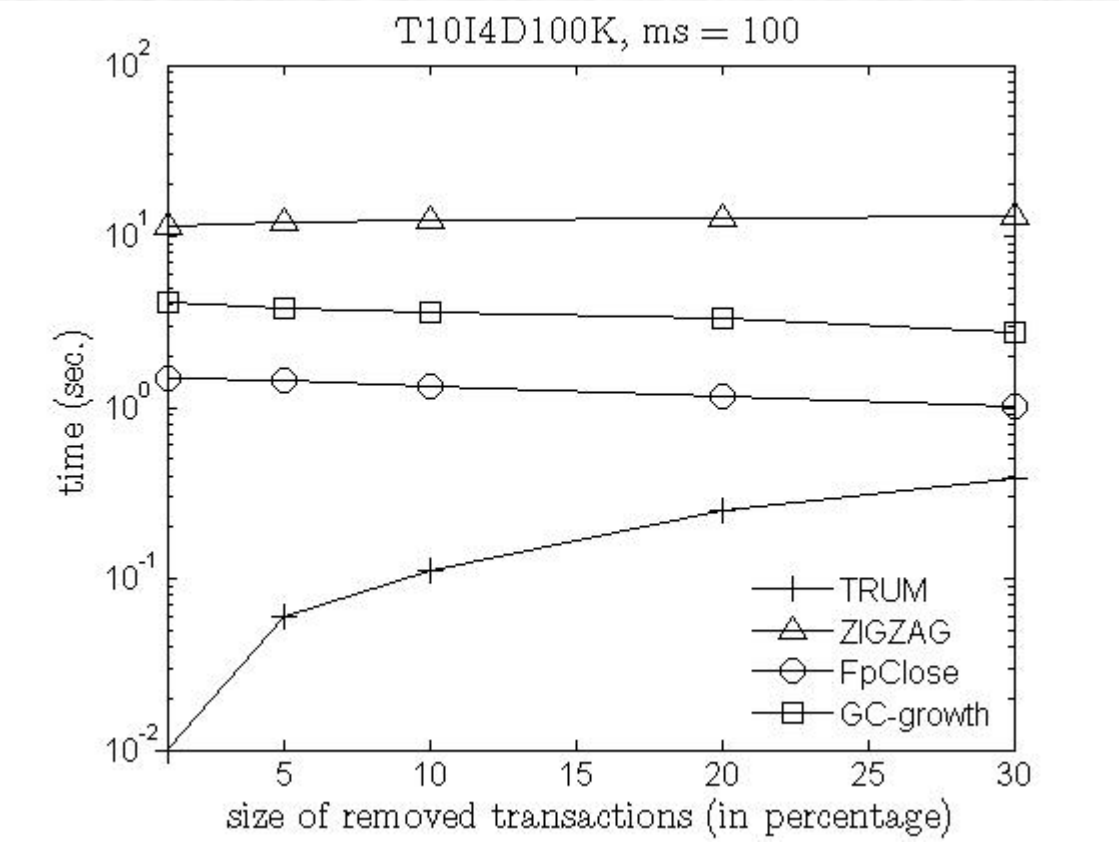
EC_5: { {a, d}, {c, d}, {a, c, d} } : 2
Tid-list <1,3>



Experimental Studies

Algorithm	Output
FpClose	Closed patterns
ZIGZAG	Max patterns
GC-growth	Equivalence classes, closed patterns and key patterns
TRUM	Equivalence classes, closed patterns and key patterns

Performance



Performance

$T_{compare}/T_{TRUm}$	T10D4D100K	Mushroom	Gazelle
ZIGZAG	83	94	400
FpClose	35	13	5
GC-growth	98	29	12

Conclusion

- Contribution
  - Analyze the evolution of frequent pattern space
  - Maintain pattern space in a divide-n-conquer manner
  - Maintain also key patterns
- Future work
  - Different types of data updates
  - Different pattern space