Association Rule Mining

CS145

Fall 2015

DHP: Reduce the Number of Candidates

- A hashing bucket count <min_sup → every candidate in the buck is infrequent
  - Candidates: a, b, c, d, e
  - Hash entries: {ab, ad, ae} {bd, be, de} …
  - Large 1-itemset: a, b, d, e
  - The sum of counts of {ab, ad, ae} < min_sup → ab should not be a candidate 2-itemset
- J. Park, M. Chen, and P. Yu, 1995

Partition: Scan Database Only Twice

- Partition the database into n partitions
- Itemset X is frequent → X is frequent in at least one partition
    - Scan 1: partition database and find local frequent patterns
    - Scan 2: consolidate global frequent patterns
- A. Savasere, E. Omiecinski, and S. Navathe, 1995

Sampling for Frequent Patterns

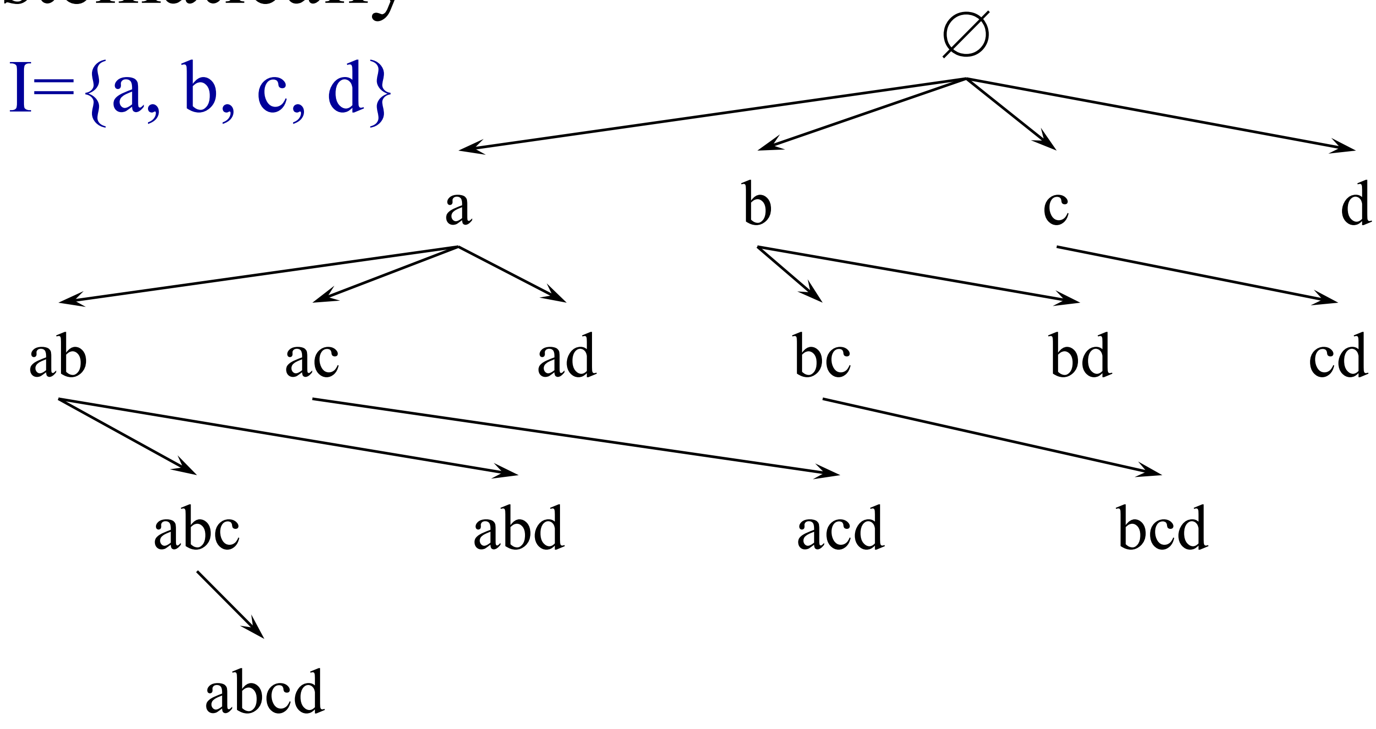
- Select a sample of original database, mine frequent patterns within sample using Apriori
- Scan database once to verify frequent itemsets found in sample, only borders of closure of frequent patterns are checked
  - Example: check abcd instead of ab, ac, …, etc.
- Scan database again to find missed frequent patterns
- H. Toivonen, 1996

Bottleneck of Frequent-pattern Mining

- Multiple database scans are costly
- Mining long patterns needs many passes of scanning and generates lots of candidates
  - To find frequent itemset $i_1 i_2 \ldots i_{100}$
    - # of scans: 100
    - # of Candidates: $\binom{100}{1} + \binom{100}{2} + \cdots + \binom{100}{100} = 2^{100} - 1 \approx 1.27 \times 10^{30}$
  - Bottleneck: candidate-generation-and-test
- Can we avoid candidate generation?

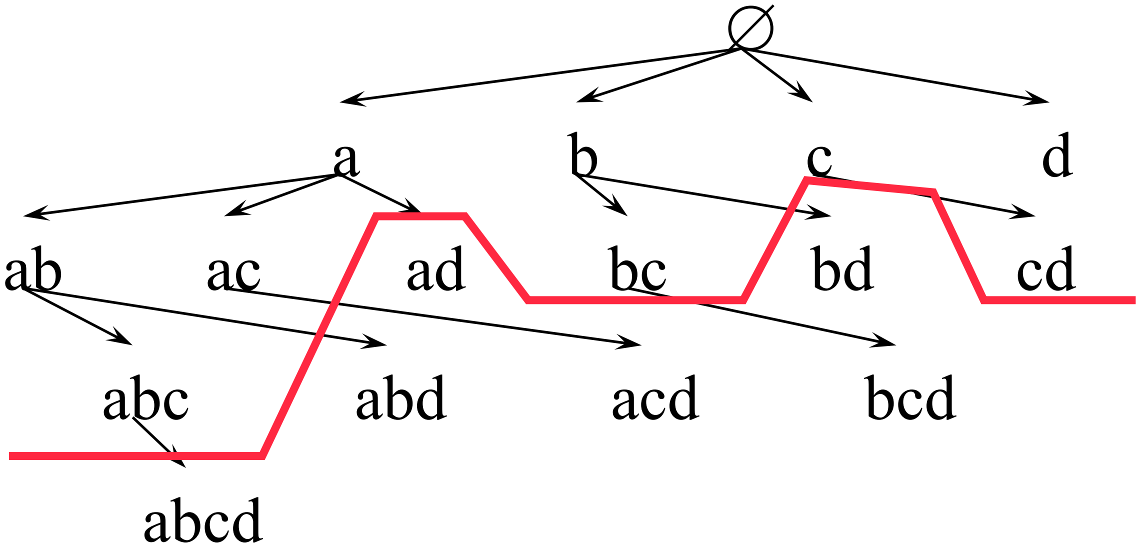
Set Enumeration Tree

- Subsets of *I* can be enumerated systematically
  - I={a, b, c, d}



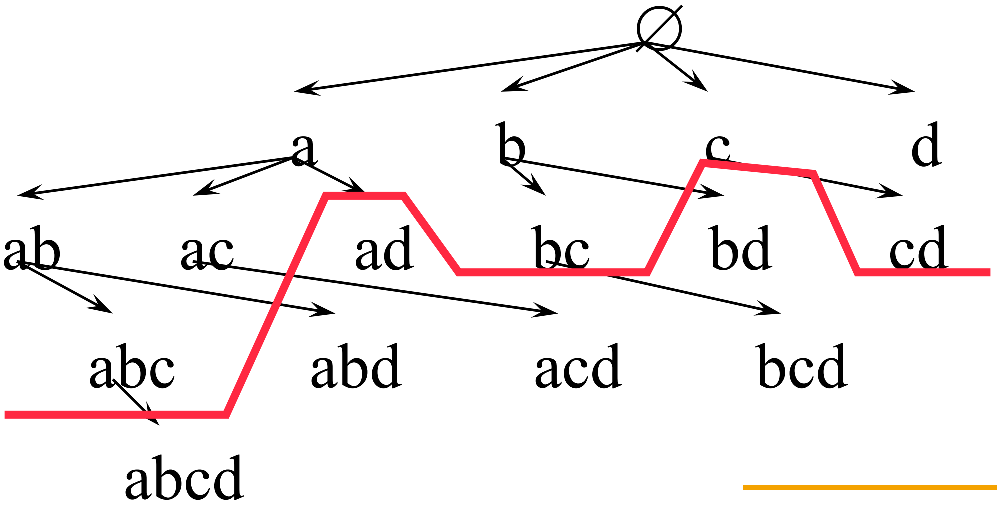
Borders of Frequent Itemsets

- Connected
  - X and Y are frequent and X is an ancestor of Y → all patterns between X and Y are frequent



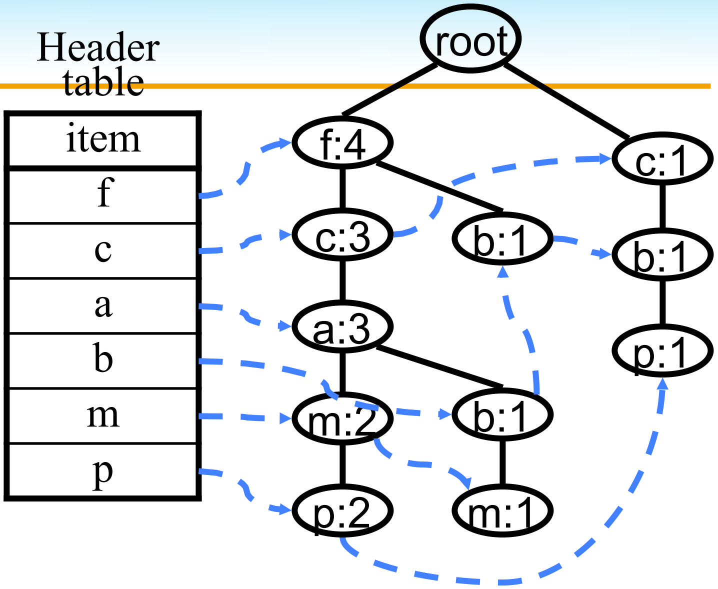
Projected Databases

- To find a child Xy of X, only X-projected database is needed
  - The sub-database of transactions containing X
  - Item y is frequent in X-projected database



Compress Database by FP-tree

- 1st scan: find freq items
  - Only record freq items in FP-tree
  - F-list: f-c-a-b-m-p
- 2nd scan: construct tree
  - Order freq items in each transaction w.r.t. f-list
  - Explore sharing among transactions



TID	Items bought	(ordered) freq items
100	f, a, c, d, g, I, m, p	f, c, a, m, p
200	a, b, c, f, l,m, o	f, c, a, b, m
300	b, f, h, j, o	f, b
400	b, c, k, s, p	c, b, p
500	a, f, c, e, l, p, m, n	f, c, a, m, p

Min_support = 3

Benefits of FP-tree

- Completeness
  - Never break a long pattern in any transaction
  - Preserve complete information for freq pattern mining
    - No need to scan database anymore
- Compactness
  - Reduce irrelevant info — infrequent items are gone
  - Items in frequency descending order (f-list): the more frequently occurring, the more likely to be shared
  - Never be larger than the original database (not counting node-links and the count fields)

Partition Frequent Patterns

- Frequent patterns can be partitioned into subsets according to f-list: f-c-a-b-m-p
  - Patterns containing p
  - Patterns having m but no p
  - …
  - Patterns having c but no a nor b, m, or p
  - Pattern f
- The partitioning is complete and without any overlap

Find Patterns Having Item "p"

- Only transactions containing p are needed
- Form p-projected database
  - Starting at entry p of header table
  - Follow the side-link of frequent item p
  - Accumulate all transformed prefix paths of p

p-projected database $\text{TDB}|_p$

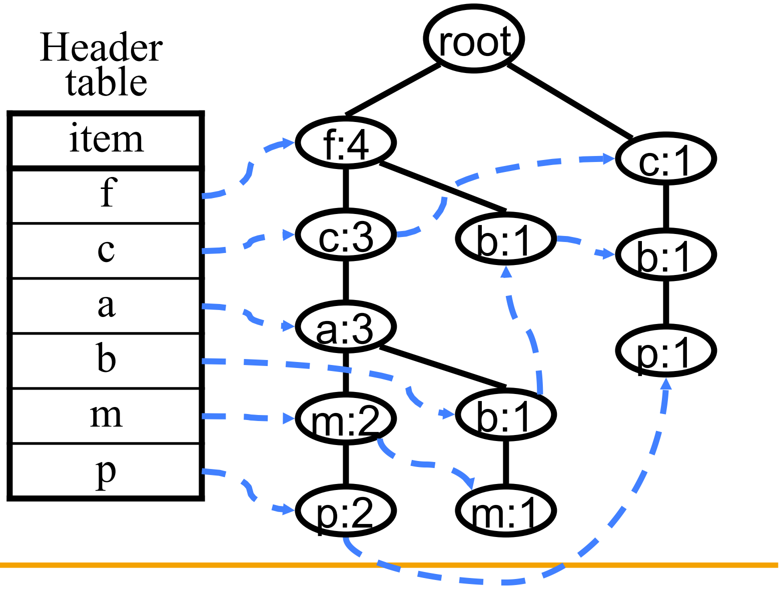
fcam: 2

cb: 1

Local frequent item: c:3

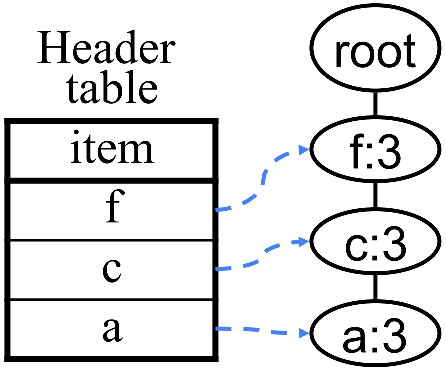
Frequent patterns containing p

p: 3, pc: 3



Find Patterns Having Item m But No p

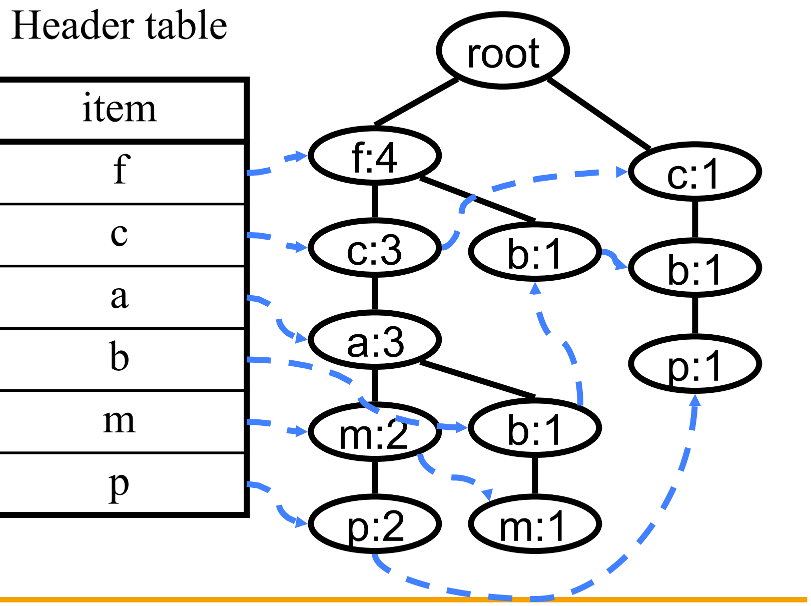
- Form m-projected database TDB|m
  - Item p is excluded
  - Contain fca:2, fcab:1
  - Local frequent items: f, c, a
- Build FP-tree for TDB|m



Header table

item
f
c
a

m-projected FP-tree

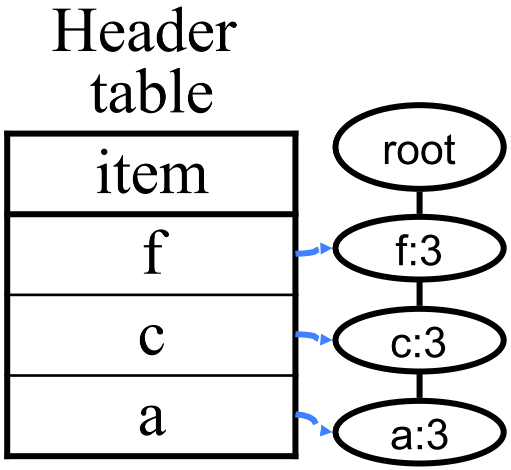


Header table

item
f
c
a
b
m
p

Recursive Mining

- Patterns having m but no p can be mined recursively
- Optimization: enumerate patterns from single-branch FP-tree
  - Enumerate all combination
  - Support = that of the last item
    - m, fm, cm, am
    - fcm, fam, cam
    - fcam



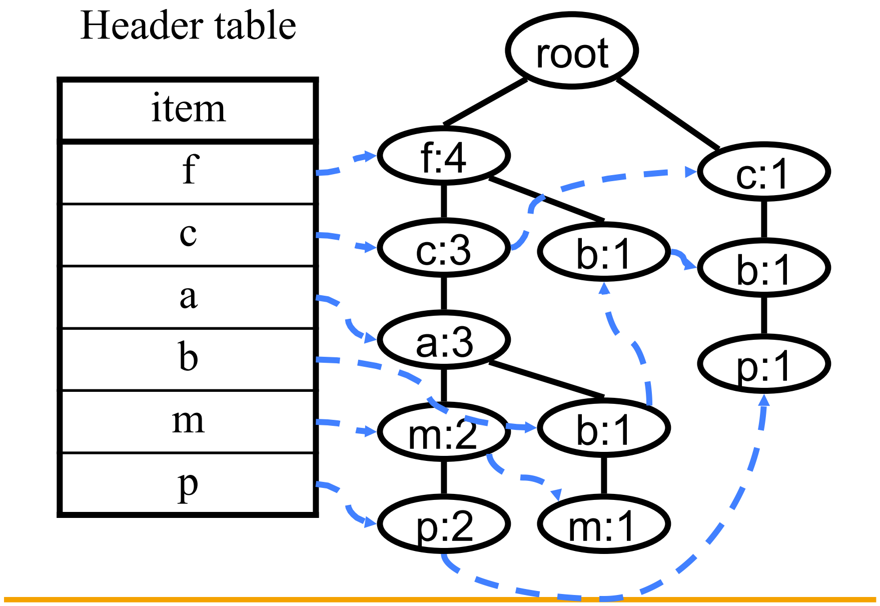
Header table

item
f
c
a

m-projected FP-tree

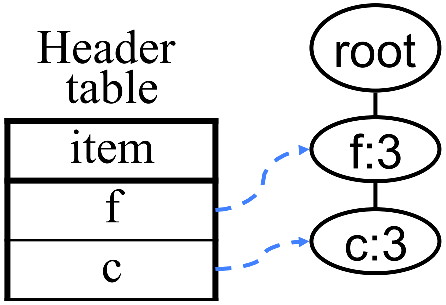
Patterns having b but no p, m

- Form b-projected database TDB|b
  - Items p, m are excluded
  - Contain fca:1, f:1, c:1
  - Local frequent items: none



Patterns having a but no p, m, b

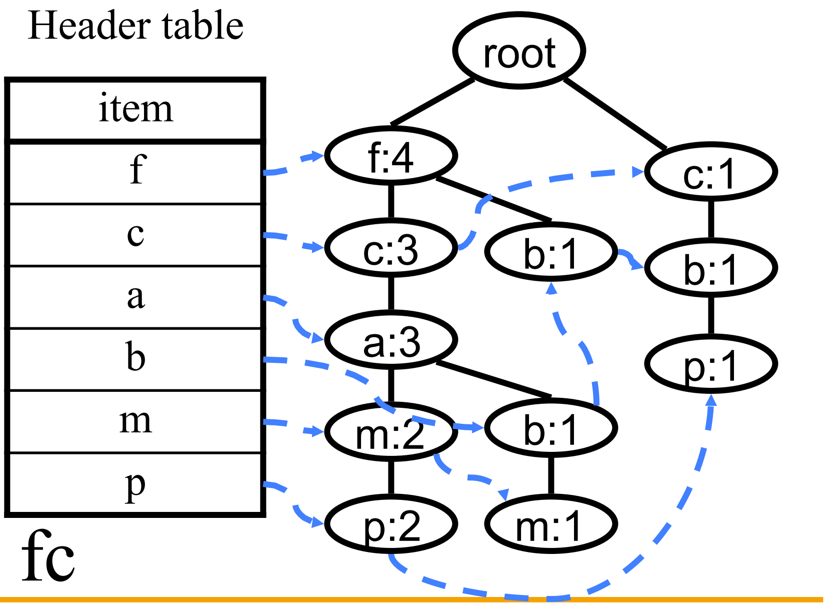
- Form a-projected database TDB|a
  - Items p, m, b are excluded
  - Contain fc:3
  - Local frequent items: f, c
- Build FP-tree for TDB|a



Header table

item
f
c

a-projected FP-tree



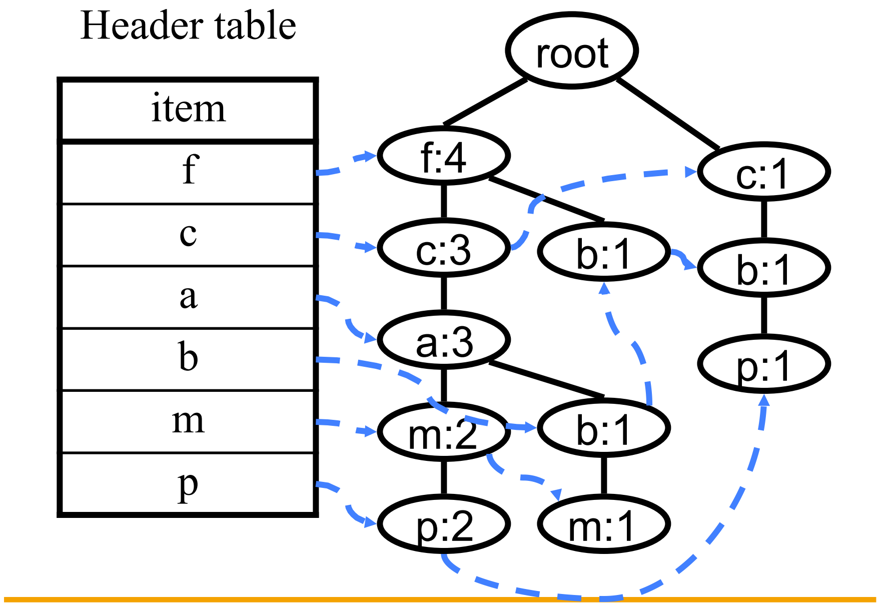
Header table

item
f
c
a
b
m
p

- Local frequent patterns: f, c, fc

Patterns having c but no p, m, b,a

- Form c-projected database TDB|c
  - Items p, m, b, a are excluded
  - Contain f:3
  - Local frequent items: f

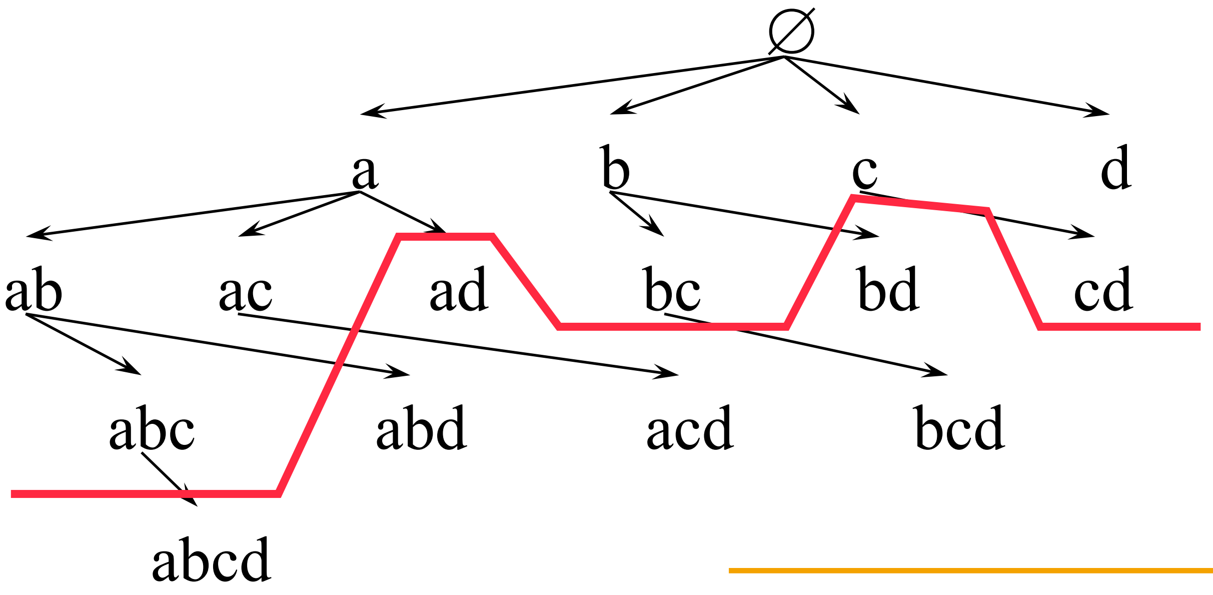


Header table

item
f
c
a
b
m
p

Borders and Max-patterns

- Max-patterns: borders of frequent patterns
  - A subset of max-pattern is frequent
  - A superset of max-pattern is infrequent



MaxMiner: Mining Max-patterns

Tid	Items
10	A,B,C,D,E
20	B,C,D,E,
30	A,C,D,F

Min_sup=2

- 1st scan: find frequent items
  - A, B, C, D, E
- 2nd scan: find support for
  - AB, AC, AD, AE, ABCDE
  - BC, BD, BE, BCDE
  - CD, CE, CDE, DE,

Potential max-patterns

- Since BCDE is a max-pattern, no need to check BCD, BDE, CDE in later scan
- Baya'98

Frequent Closed Patterns

- For frequent itemset X, if there exists no item y s.t. every transaction containing X also contains y, then X is a frequent closed pattern
  - "acdf" is a frequent closed pattern
- Concise rep. of freq pats
- Reduce # of patterns and rules
- N. Pasquier et al. In ICDT'99

Min_sup=2

TID	Items
10	a, c, d, e, f
20	a, b, e
30	c, e, f
40	a, c, d, f
50	c, e, f

CLOSET: Mining Frequent Closed Patterns

- Flist: list of all freq items in support desc. order
  - Flist: c-e-f-a-d
- Divide search space
  - Patterns having d
  - Patterns having a but no d, etc.
- Find frequent closed pattern recursively
  - Every transaction having d also has cfa → cfad may be a frequent closed pattern
- PHM’00

Min_sup=2

TID	Items
10	a, c, d, e, f
20	a, b, e
30	c, e, f
40	a, c, d, f
50	c, e, f

Closed and Max-patterns

- Closed pattern mining algorithms can be adapted to mine max-patterns
  - A max-pattern must be closed
- Depth-first search methods have advantages over breadth-first search ones