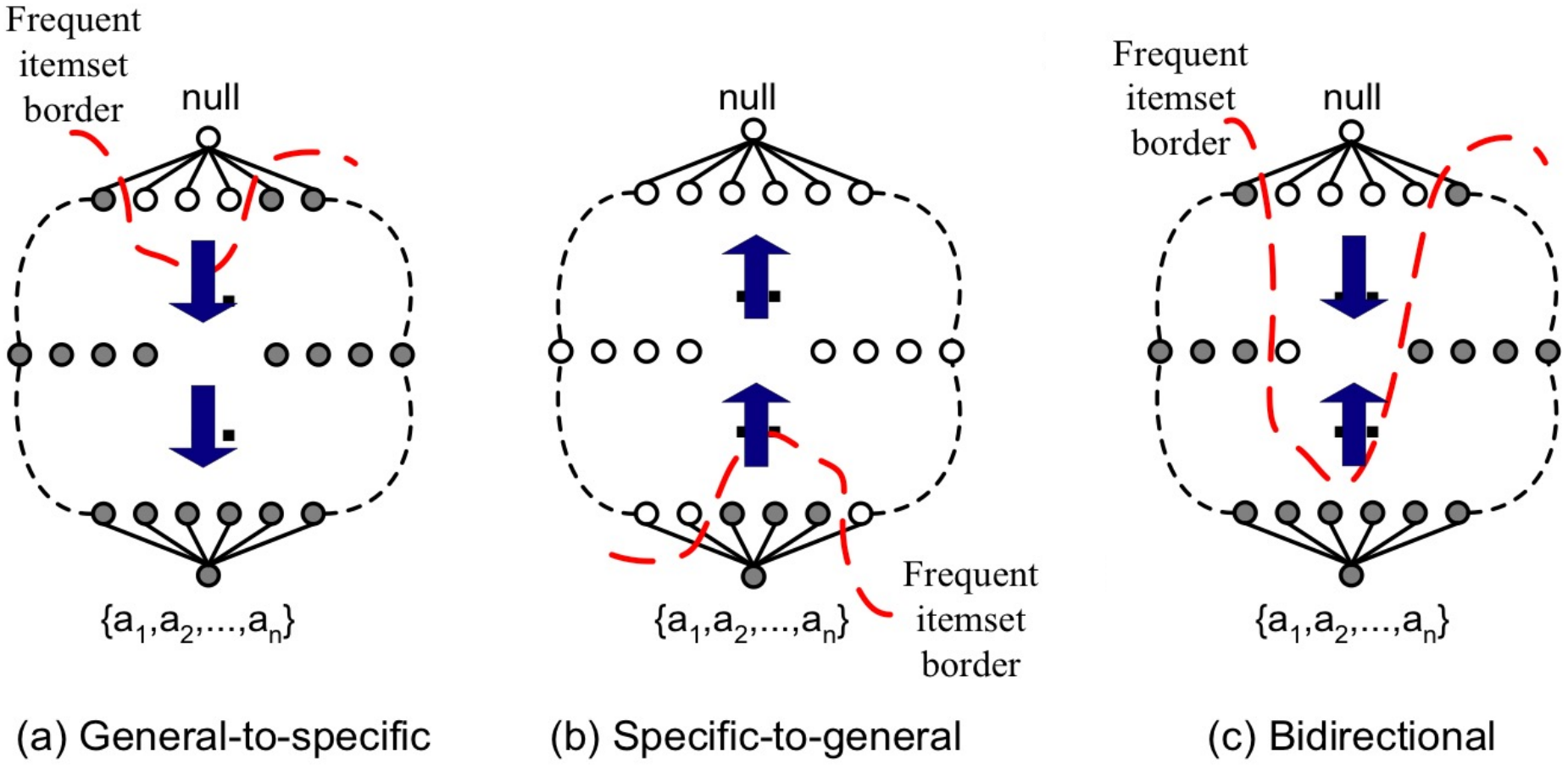
Alternative Methods for Frequent Itemset Generation

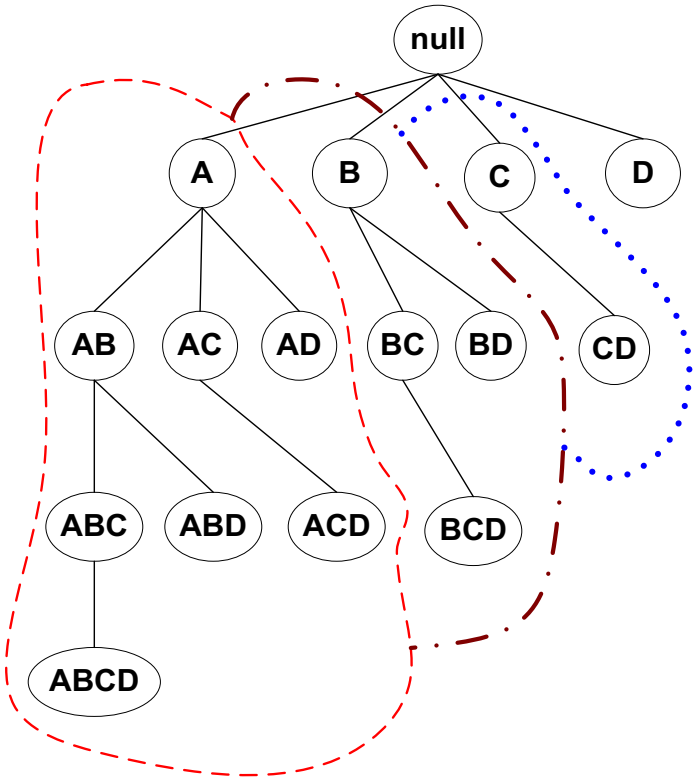
- Traversal of Itemset Lattice
  - General-to-specific vs Specific-to-general



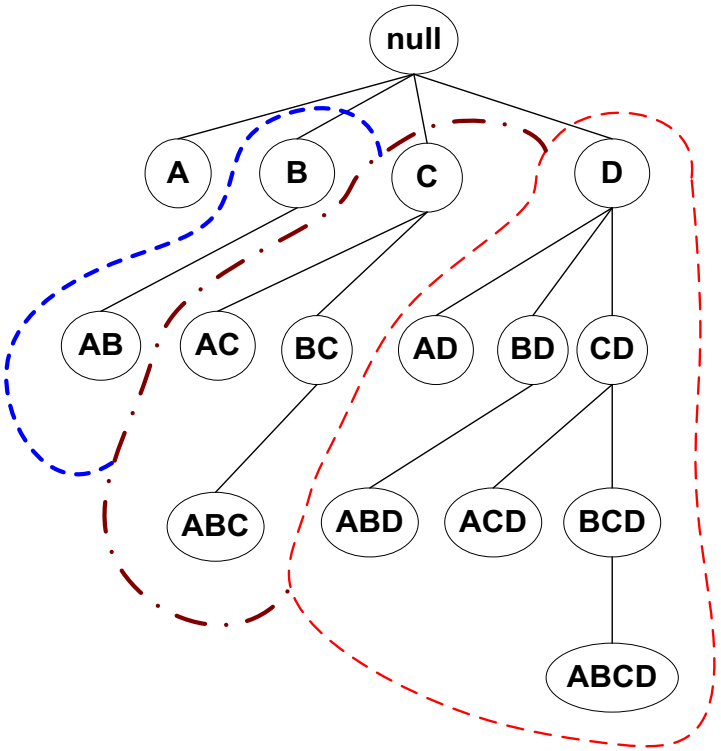
(a) General-to-specific (b) Specific-to-general (c) Bidirectional

Alternative Methods for Frequent Itemset Generation

- Traversal of Itemset Lattice
  - Equivalent Classes



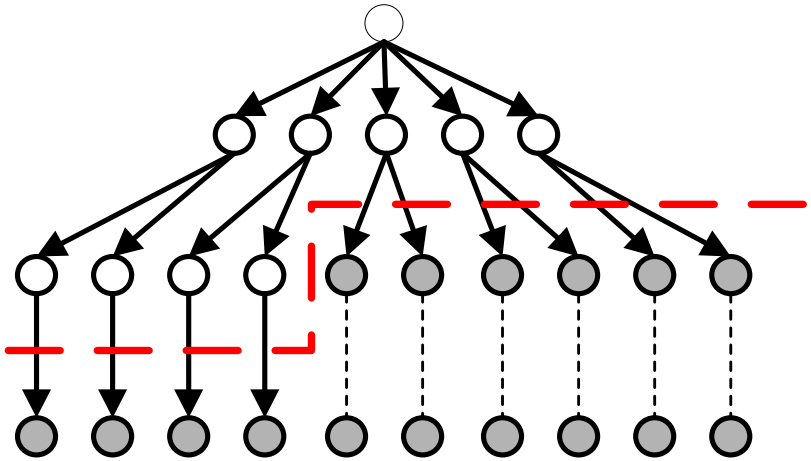
(a) Prefix tree



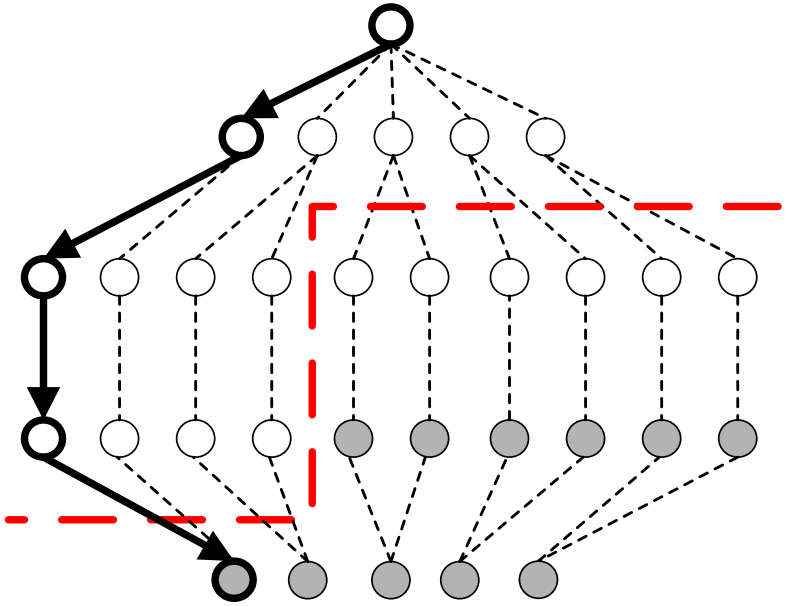
(b) Suffix tree

Alternative Methods for Frequent Itemset Generation

- Traversal of Itemset Lattice
  - Breadth-first vs Depth-first



(a) Breadth first



(b) Depth first

Alternative Methods for Frequent Itemset Generation

- Representation of Database
  - horizontal vs vertical data layout

Horizontal Data Layout

TID	Items
1	A,B,E
2	B,C,D
3	C,E
4	A,C,D
5	A,B,C,D
6	A,E
7	A,B
8	A,B,C
9	A,C,D
10	B

Vertical Data Layout

A	B	C	D	E
1	1	2	2	1
4	2	3	4	3
5	5	4	5	6
6	7	8	9	
7	8	9		
8	10			
9				

FP-growth Algorithm

- Use a compressed representation of the database using an FP-tree

- Once an FP-tree has been constructed, it uses a recursive divide-and-conquer approach to mine the frequent itemsets

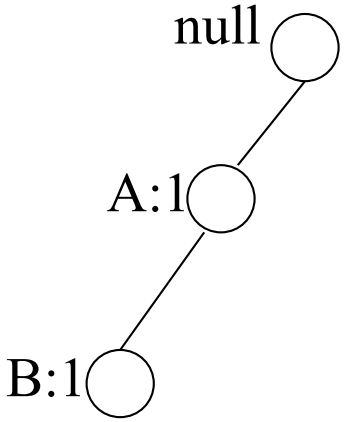
FP-tree construction

Maps each transaction onto a path in the FP-tree

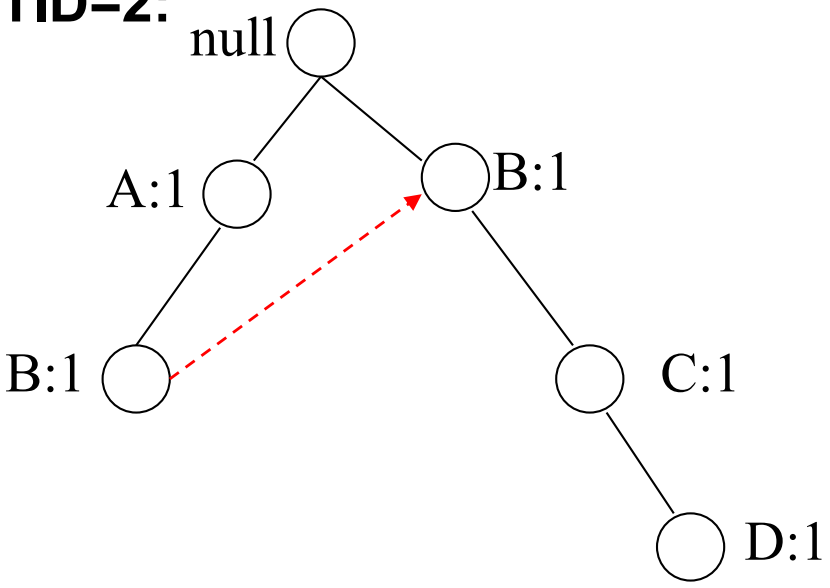
TID	Items
1	{A,B}
2	{B,C,D}
3	{A,C,D,E}
4	{A,D,E}
5	{A,B,C}
6	{A,B,C,D}
7	{B,C}
8	{A,B,C}
9	{A,B,D}
10	{B,C,E}

Frequent 1-itemsets are sorted in decreasing support counts.

After reading TID=1:



After reading TID=2:



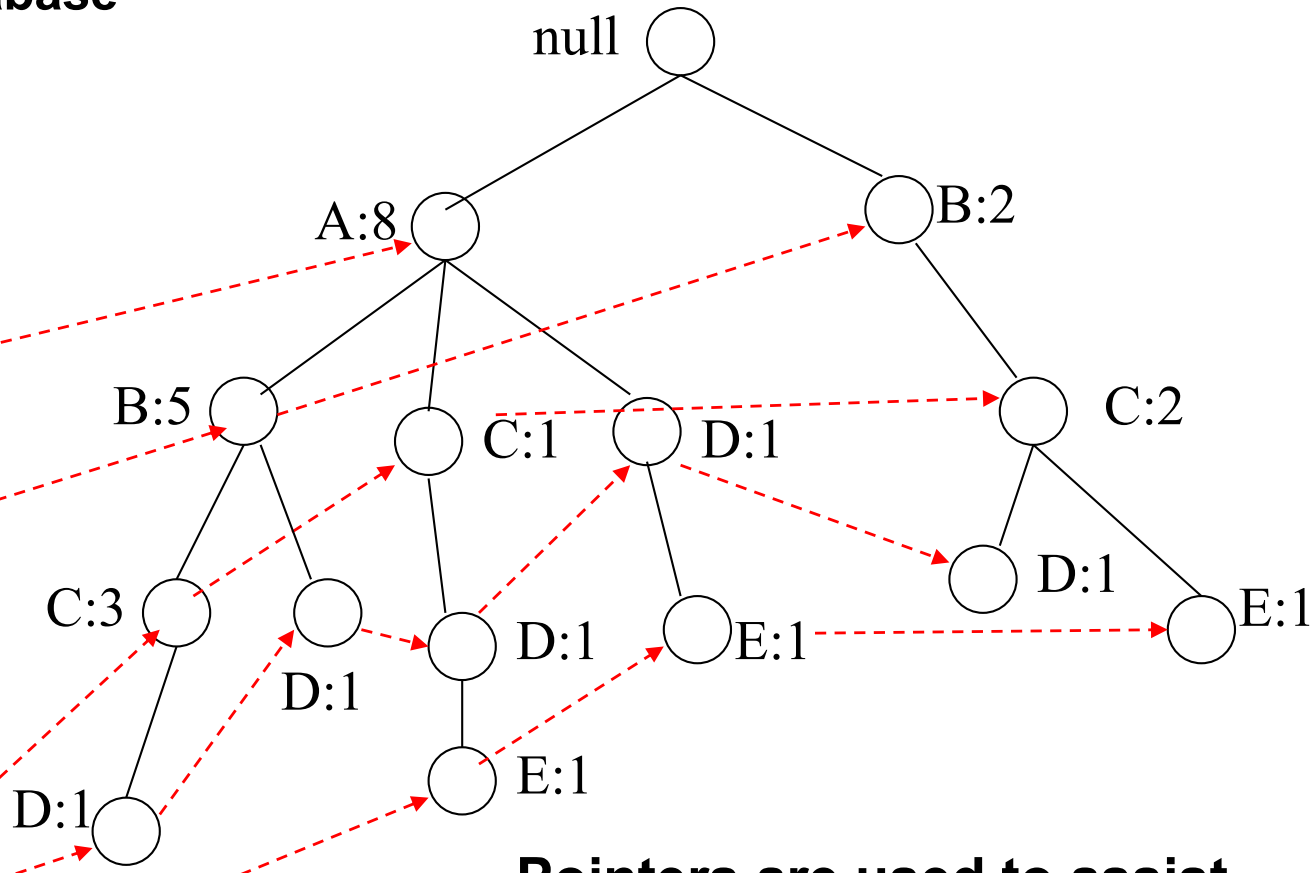
FP-Tree Construction

TID	Items
1	{A,B}
2	{B,C,D}
3	{A,C,D,E}
4	{A,D,E}
5	{A,B,C}
6	{A,B,C,D}
7	{A}
8	{A,B,C}
9	{A,B,D}
10	{B,C,E}

Transaction Database

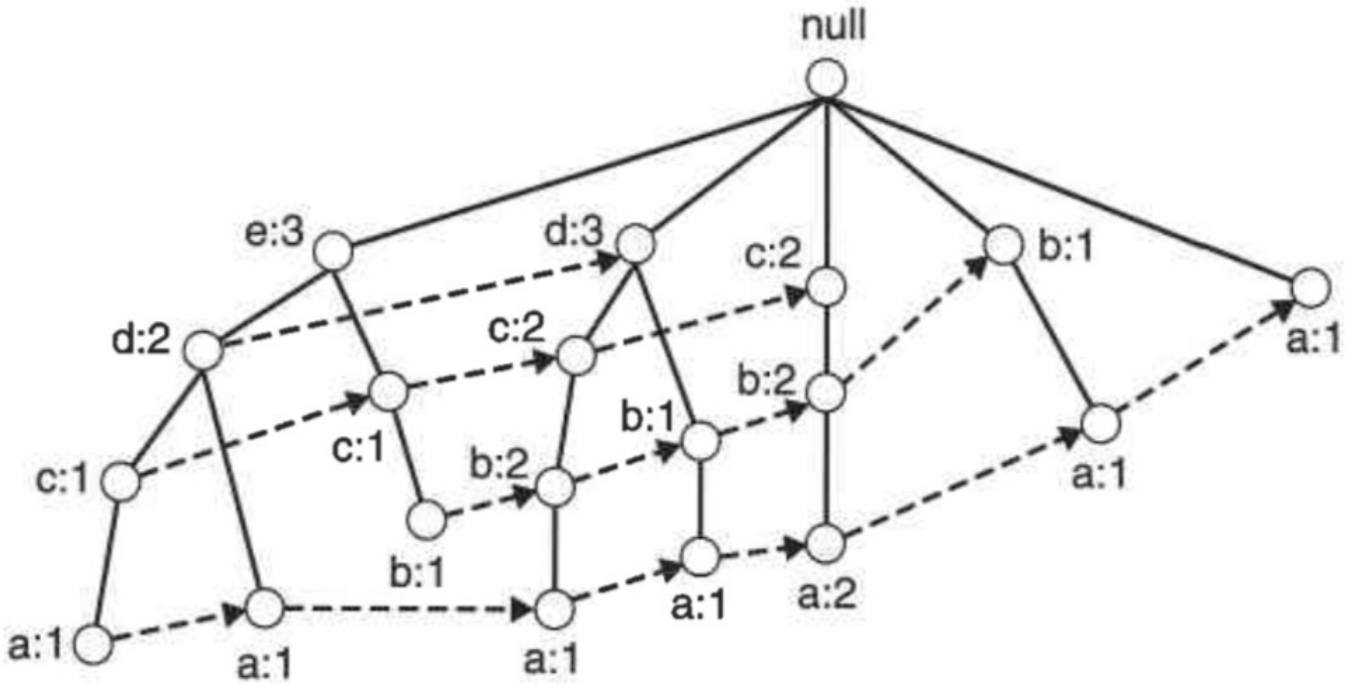
Header table

Item	Pointer
A	
B	
C	
D	
E	



Pointers are used to assist frequent itemset generation

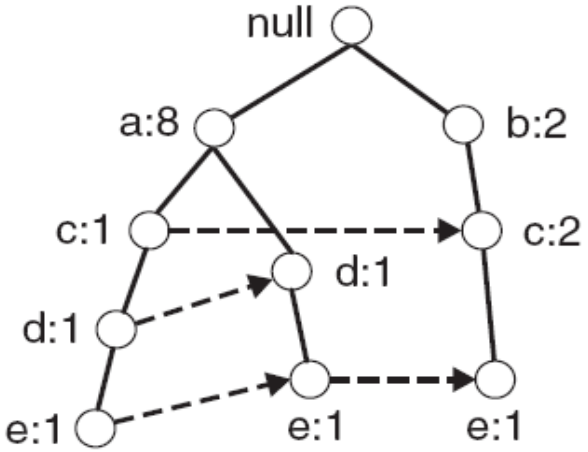
FP-Tree Construction



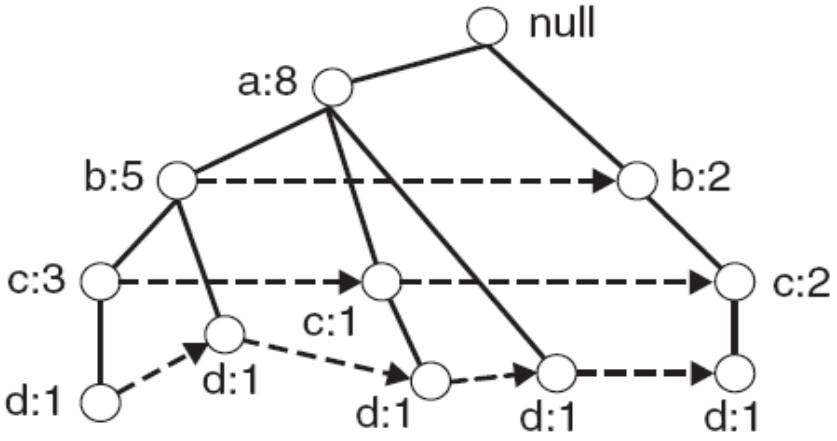
The ordering scheme is from lowest to highest support item.

Frequent Itemset Generation in FP-Growth

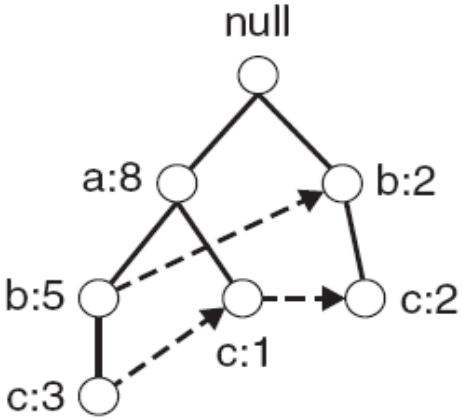
FP-growth generates frequent itemsets from a FP-tree by exploring the tree in a bottom-up fashion.



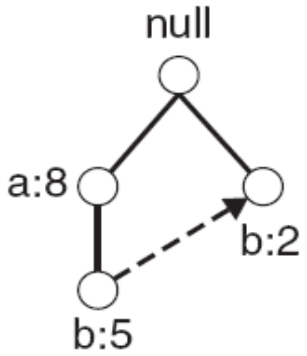
(a) Paths containing node e



(b) Paths containing node d



(c) Paths containing node c

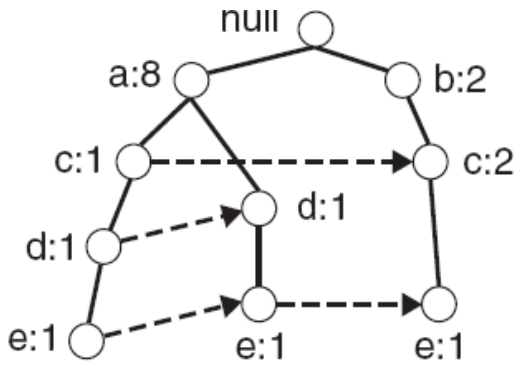


(d) Paths containing node b

(e) Paths containing node a

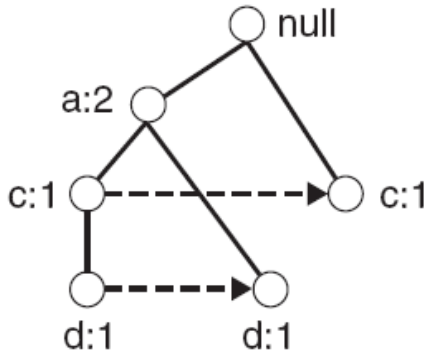
Frequent Itemset Generation in FP-Growth

Finding all the frequent itemsets ending with a particular suffix by employing a divide-and-conquer strategy



(a) Prefix paths ending in e

Minimum support count is 2
{e} is a frequent itemset

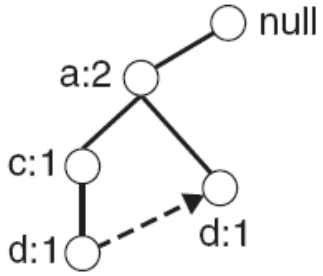


(b) Conditional FP-tree for e

Conditional FP-tree for e
It is used to find frequent itemsets ending in *de,ce,ae*

- Update the support counts
- Remove e
- Remove infrequent items

Frequent Itemset Generation in FP-Growth



(c) Prefix paths ending in de

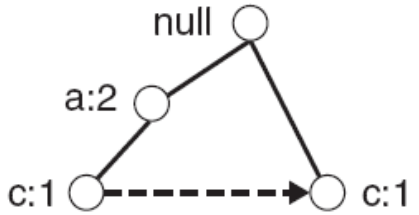
***{d,e}* is a frequent itemset**

(d) Conditional FP-tree for de

Construct a conditional FP-tree for *de*

***{a,d,e}* is a frequent itemset**

Frequent Itemset Generation in FP-Growth



(e) Prefix paths ending in ce

***{c,e}* is a frequent itemset**

(f) Prefix paths ending in ae

***{a,e}* is a frequent itemset**

Summary

- Association rule mining:
  - Frequent itemset generation
    - Evaluation metrics: support, count,…
  - Rule generation
- Frequent itemset generation algorithms:
  - Apriori
    - $F_{k-1}*F_1$, $F_{k-1}*F_{k-1}$
  - FP-growth
- Support count:
  - Hash tree, FP-tree
- Frequent itemsets representations:
  - Maximal/closed frequent itemsets