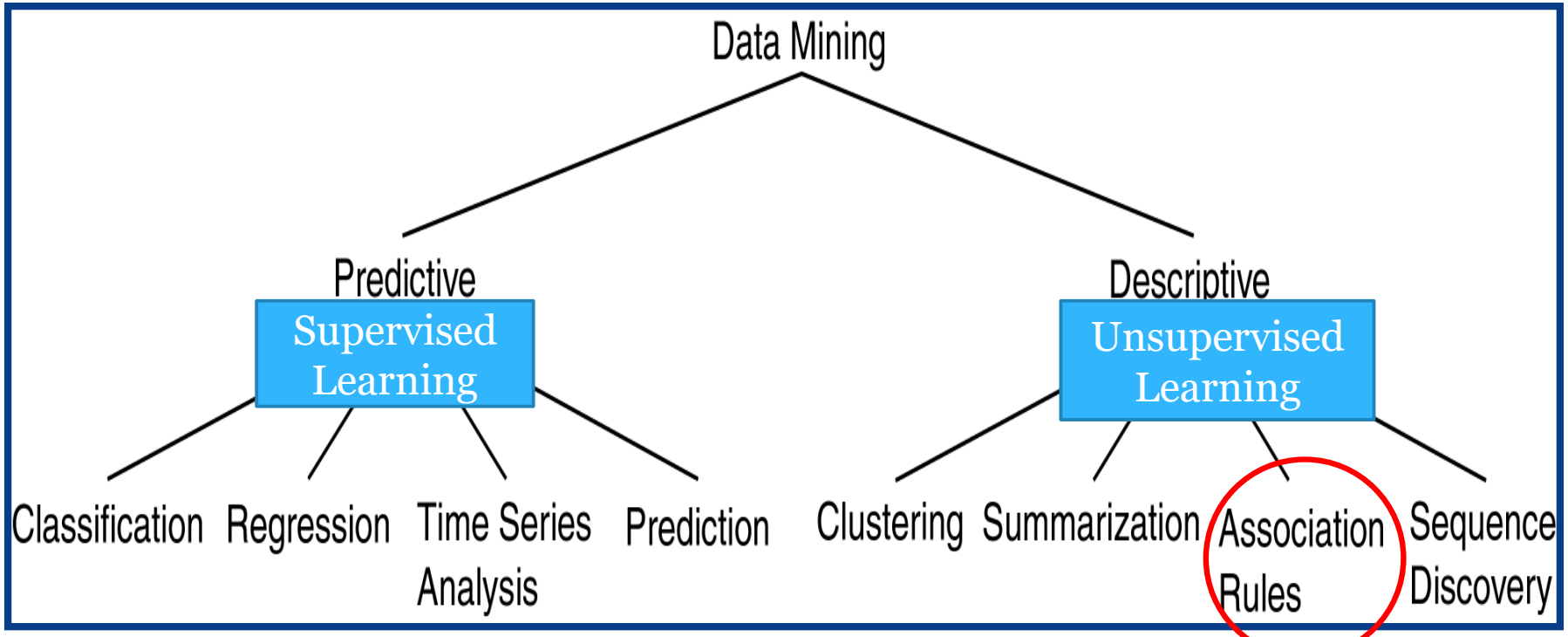
Unsupervised Learning: Association Analysis with FP-Growth Algorithm

Dr. Yuzana Win (Nagasaki University, Japan)
Lecturer
Department of Computer Engineering and Information Technology

Lecture Objectives

- To introduce
  - What is Association Rules Analysis?
  - Association Analysis with FP-Growth Algorithm
  - How does the FP-Growth algorithm work?
  - Why FP-Growth algorithm is used for?

Data Mining Methods and Models



Basket Data (called Transaction Data)

Transaction data => each record represents a transaction between a customer and a shop

CEIT
Think smart, be different.

Association Rules Analysis

- Given a set of transactions, find rules that will predict the occurrence of an item based on the occurrences of other items in the transaction.

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Example of Association Rules

{Diaper} → {Beer},
{Milk, Bread} → {Eggs,Coke},
{Beer, Bread} → {Milk},

more item frequently => to increase profit

The process of identifying an associates between products => association rule mining

Discovering Rules

- According to the transaction data, we find the useful rule such that

  If a basket contains bread and milk, then it also contains diaper

Any such rule has two associated measures:

1. *confidence* – when the `if' part is true, how often is the `then' also true? This is the same as *accuracy*.
2. *coverage* or *support* – how much of the database contains the `if' part?

Definition: Association Rule

- **Association Rule**
  - An implication expression of the form $X \rightarrow Y$, where X and Y are itemsets
  - Example:
    {Milk, Diaper} $\rightarrow$ {Beer}

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

- **Rule Evaluation Metrics**
  - Support (s)
    - Fraction of transactions that contain both X and Y
  - Confidence (c)
    - Measures how often items in Y appear in transactions that contain X

Example:

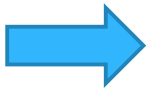
$$\{\text{Milk}, \text{Diaper}\} \Rightarrow \text{Beer}$$

$$s = \frac{\sigma(\text{Milk}, \text{Diaper}, \text{Beer})}{|\text{T}|} = \frac{2}{5} = 0.4$$

$$c = \frac{\sigma(\text{Milk}, \text{Diaper}, \text{Beer})}{\sigma(\text{Milk}, \text{Diaper})} = \frac{2}{3} = 0.67$$

Definition: Frequent Itemset

- **Itemset**
  - A collection of one or more items
    - Example: {Milk, Bread, Diaper}
  - k-itemset
    - An itemset that contains k items
- **Support count (σ)**
  - Frequency of occurrence of an itemset
  - E.g. σ({Milk, Bread,Diaper}) = 2
- **Support**
  - Fraction of transactions that contain an itemset
  - E.g. s({Milk, Bread, Diaper}) = 2/5
- **Frequent Itemset**
  - An itemset whose support is greater than or equal to a *minsup* threshold



TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Algorithm to Generate Association Rule

Input:

D //Database of transactions
I //Items
L //Large itemsets
s //Support
α //Confidence

Output:

R //Association Rules satisfying s and α

ARGen Algorithm:

$R = \emptyset$;
for each $l \in L$ **do**
  for each $x \subset l$ **such that** $x \neq \emptyset$ **and** $x \neq l$ **do**
    if $\frac{support(l)}{support(x)} \geq \alpha$ **then**
      $R = R \cup \{x \Rightarrow (l - x)\}$;

Association Rule Mining Task

1. Find Large Itemsets

1. Generate rules from frequent itemsets and find all rules having
   - support ≥ *minsup* threshold
   - confidence ≥ *minconf* threshold

Mining Association Rules

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Example of Rules:

{Milk,Diaper} → {Beer} (s=0.4, c=0.67)
{Milk,Beer} → {Diaper} (s=0.4, c=1.0)
{Diaper,Beer} → {Milk} (s=0.4, c=0.67)
{Beer} → {Milk,Diaper} (s=0.4, c=0.67)
{Diaper} → {Milk,Beer} (s=0.4, c=0.5)
{Milk} → {Diaper,Beer} (s=0.4, c=0.5)

Observations:

- All the above rules are binary partitions of the same itemset:
  {Milk, Diaper, Beer}
- Rules originating from the same itemset have identical support but can have different confidence

Association Rules Algorithm

- Apriori Algorithm
- FP-growth Algorithm
- ECLAT

What is FP-growth Algorithm?

- FP-growth allows frequent itemset discovery without candidate itemset generation unlike apriori algorithm

- Two step approaches:

(1) Build a compact data structure called an FP-tree

(2) Extracts frequent itemsets directly from the FP-tree

Step 1: FP-Tree Construction

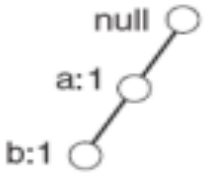
- Constructed using two passes over the data set
- Pass 1:
  - Scan data and find support for each item
  - Discard infrequent items
  - Sort frequent items in decreasing order based on their support

- Pass 2:
  - Nodes corresponds to items and have a counter
  - Reads one transaction at a time and maps it to a path
  - Fixed order is used, so paths can overlap when transactions share items
  - Pointers are used => the more paths that overlap, the higher the compression
  - FP-tree may fit in memory
  - Frequent itemsets extracted from the FP-Tree

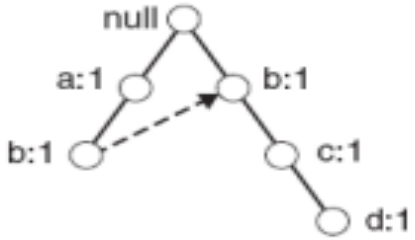
Example: FP-Tree Construction

Transaction Data Set

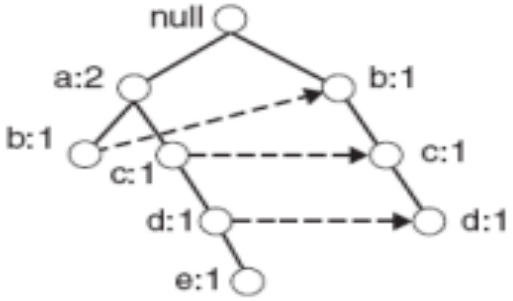
TID	Items
1	{a,b}
2	{b,c,d}
3	{a,c,d,e}
4	{a,d,e}
5	{a,b,c}
6	{a,b,c,d}
7	{a}
8	{a,b,c}
9	{a,b,d}
10	{b,c,e}



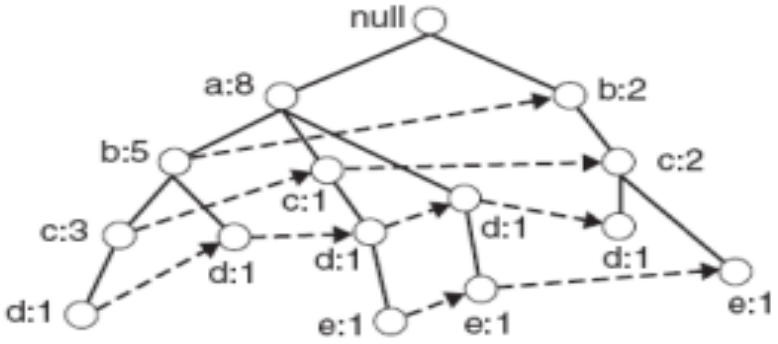
(i) After reading TID=1



(ii) After reading TID=2



(iii) After reading TID=3

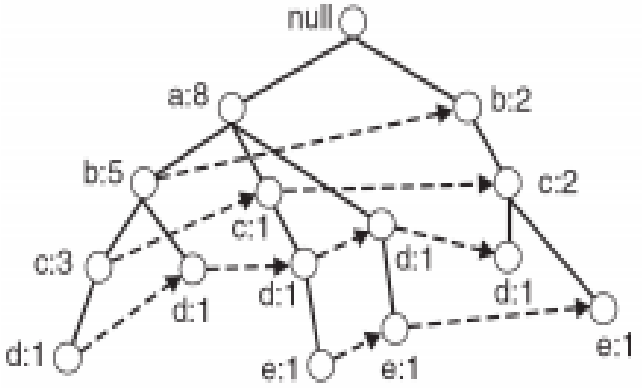


(iv) After reading TID=10

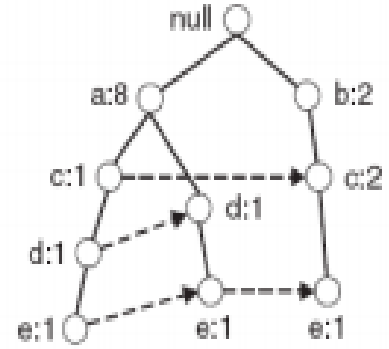
Step 2: Frequent Itemset Generation

- FP-Growth extracts frequent itemsets from the FP-tree
- Bottom-up approach (from the leaves towards the root) and
- Divide the conquer approach
- First, extract prefix path sub trees ending in an itemset
  - Using the linked lists

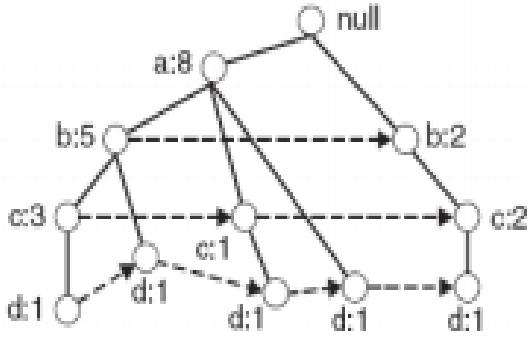
Example: Prefix path sub-trees



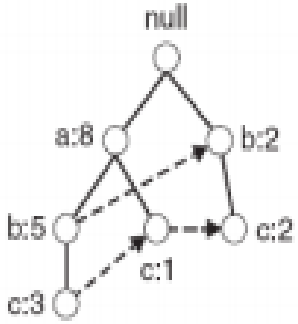
↑ Complete FP-tree



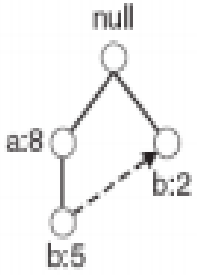
(a) Paths containing node e



(b) Paths containing node d



(c) Paths containing node c

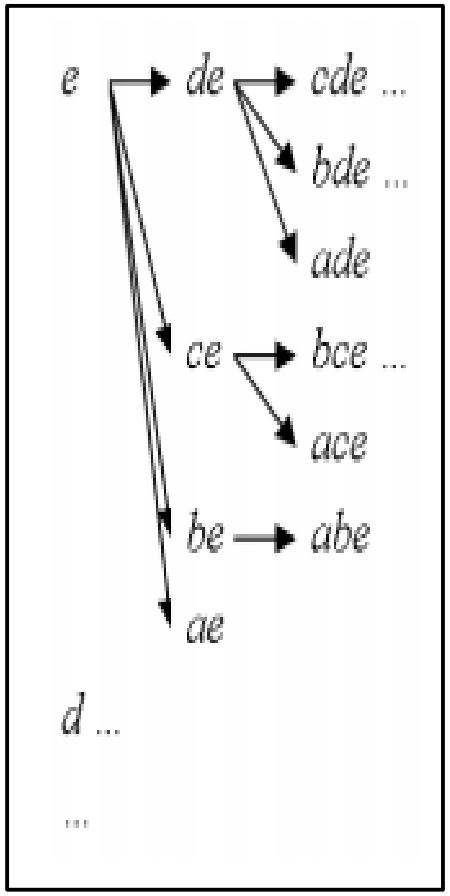


(d) Paths containing node b

(e) Paths containing node a

Step 2: Frequent Itemset Generation

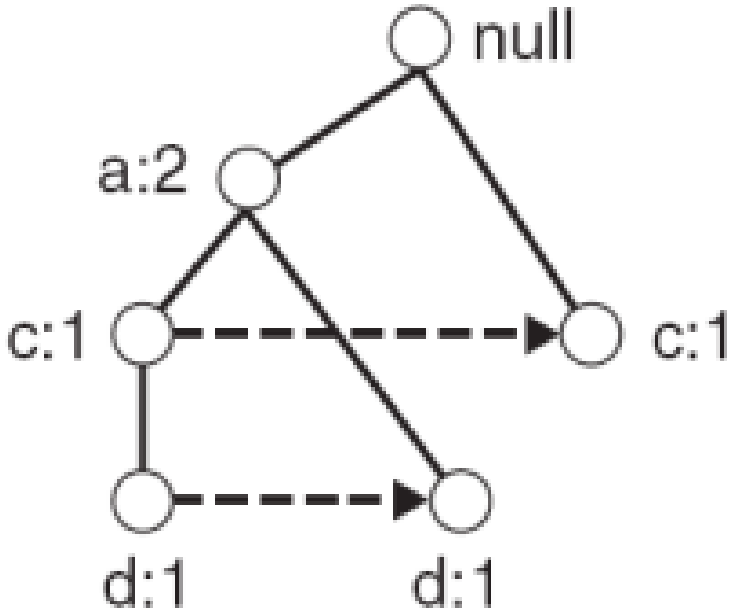
- Each prefix path sub-tree is processed recursively to extract the frequent itemsets. Solutions are then merged.
  - E.g. the prefix path sub-tree for *e* will be used to extract frequent itemsets ending in e, then in de, ce, be and ae, then in cde, bde, cde, etc.
  - Divide and conquer approach



Conditional FP-Tree

- The FP-Tree that would be built if we only consider transactions containing a particular itemset
- Example: FP-Tree conditional on e.

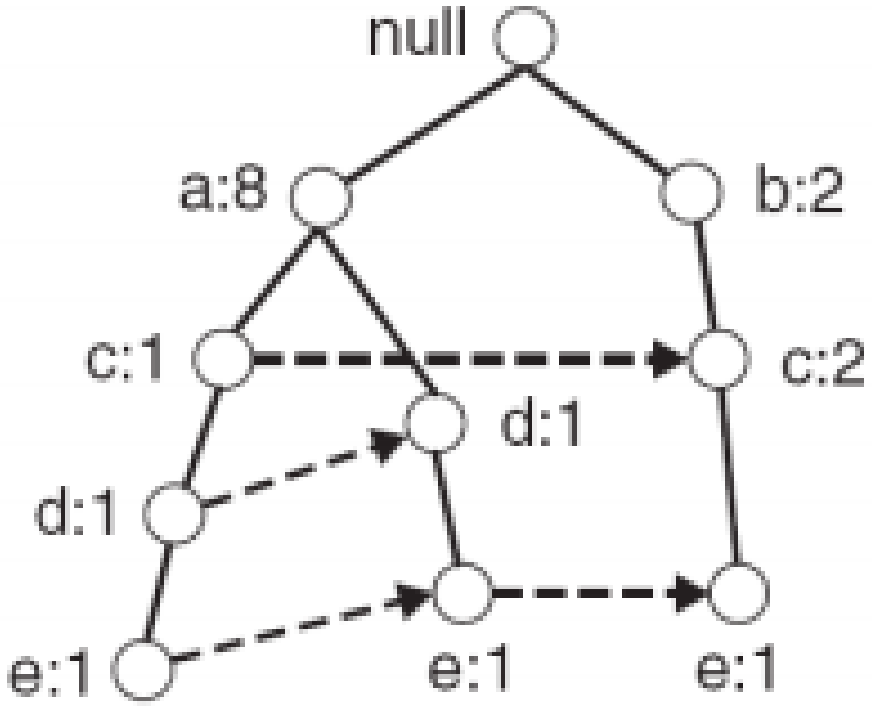
TID	Items
~~1~~	~~{a,b}~~
~~2~~	~~{b,c,d}~~
3	{a,c,d,~~e~~}
4	{a,d,~~e~~}
~~5~~	~~{a,b,e}~~
~~6~~	~~{a,b,e,d}~~
~~7~~	~~{e}~~
~~8~~	~~{a,b,e}~~
~~9~~	~~{a,b,d}~~
10	{b,c,~~e~~}



Example

Let minSup = 2 and extract all frequent itemsets containing e.

- Obtain the prefix path sub-tree for e:



Example

- Check if e is a frequent item by adding the counts along the linked list (dotted line). If so, extract it.
  - Yes, count =3 so {e} is extracted as a frequent itemset.

- As e is frequent, find frequent itemsets ending in e. i.e. de, ce, be and ae.

Example

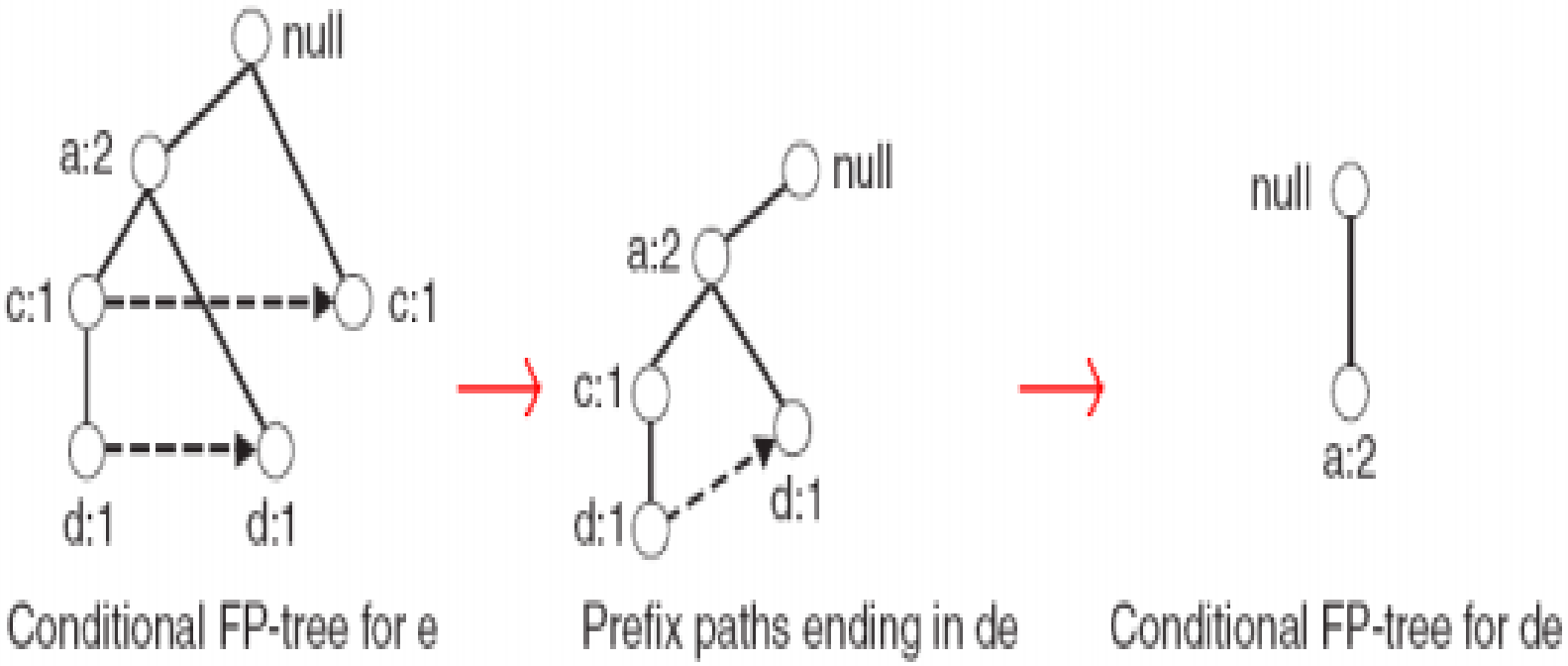
- Use the conditional FP-tree for e to find frequent itemsets ending in de, ce and ae
  - Note that be is not considered as b is not in the conditional FP-tree for e.

For each of them (e.g. de), find the prefix paths from the conditional tree for e, extract frequent itemsets, generate conditional FP-tree, etc... (recursively)

Example

- For the item;

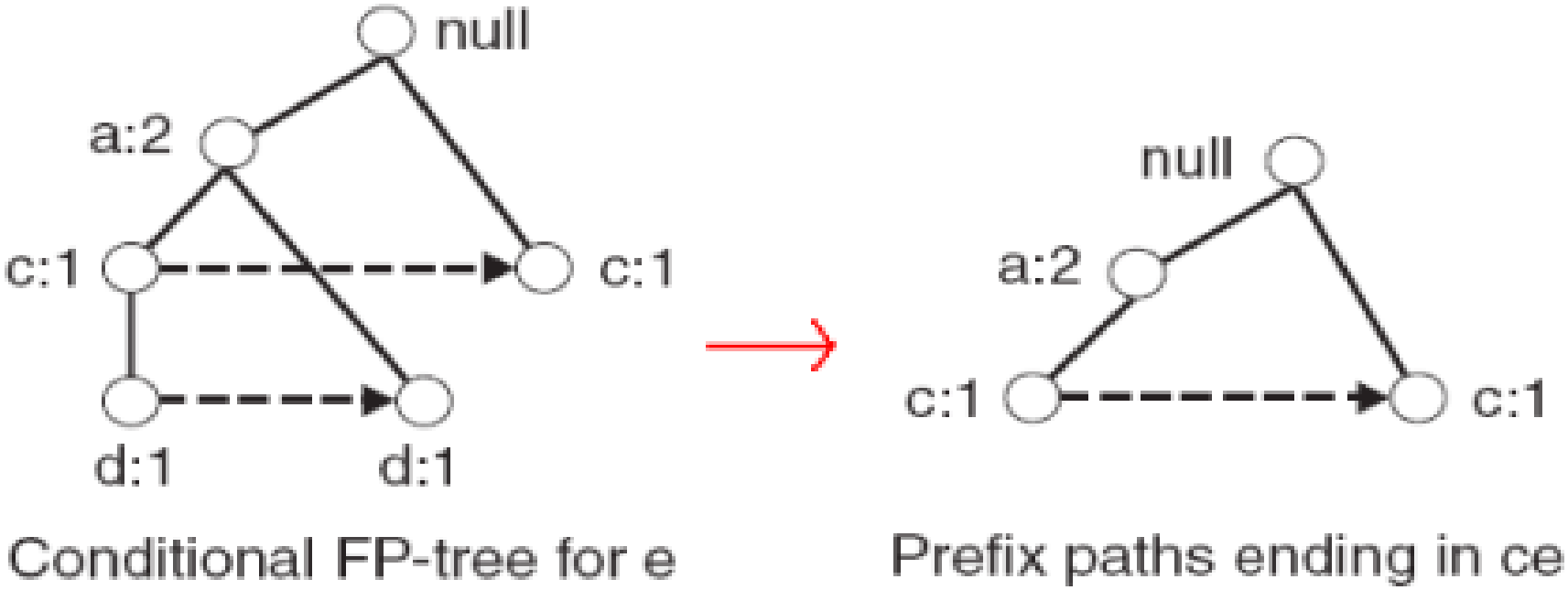
  e -> de -> ade ({d,e}, {a,d,e} are found to be frequent)



Example

- For the item;

  e -> ce ({c,e} is found to be frequent)



Conditional FP-tree for e

Prefix paths ending in ce

Result

- Frequent itemsets are found

Transaction Data Set

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}

Suffix	Frequent Itemsets
e	{e}, {d,e}, {a,d,e}, {c,e},{a,e}
d	{d}, {c,d}, {b,c,d}, {a,c,d}, {b,d}, {a,b,d}, {a,d}
c	{c}, {b,c}, {a,b,c}, {a,c}
b	{b}, {a,b}
a	{a}

Advantages and Disadvantages

- Advantages of FP-Growth
  - only 2 passes over data-set
  - “compresses” data-set
  - no candidate generation
  - much faster than Apriori

- Disadvantages of FP-Growth
  - FP-Tree may not fit in memory!!
  - FP-Tree is expensive to build

References

- Jiawei Han and Micheline Kamber. *Data Mining - Concepts and Techniques*. MorganKaufmann Publishers, 2001
- Pang-Ning Tan, Michael Steinbach, Vipin Kumar:*Introduction to Data Mining*, Addison-Wesley

Next Week Lecture

- Unsupervised Learning: Cluster Analysis with K-means

Thank You