ASSOCIATION ANALYSIS

Association Rule Mining

association analysis: useful for discovering interesting relationships (Association Rules) hidden in large data sets

- 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

Association Rule Mining

association analysis: useful for discovering interesting relationships (Association Rules) hidden in large data sets

- 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

Market-Basket transactions

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

Association Rule Mining

association analysis: useful for discovering interesting relationships (Association Rules) hidden in large data sets

- 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

Market-Basket transactions

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

$\{\text{Diaper}\} \rightarrow \{\text{Beer}\}$,

Implication means co-occurrence, not causality!

Problem Definition

Binary Representation

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

TID	Bread	Milk	Diapers	Beer	Eggs	Cola
1	1	1	0	0	0	0
2	1	0	1	1	1	0
3	0	1	1	1	0	1
4	1	1	1	1	0	0
5	1	1	1	0	0	1

$$I = \{i_1, i_2, \ldots, i_d\}$$

$$T = \{t_1, t_2, \ldots, t_N\}$$

Definition: Frequent Itemset

- **Itemset**
  - A collection of one or more items
    - Example: {Milk, Bread, Diaper}
  - k-itemset
    - An itemset that contains k items
  - transaction *tj* contains an itemset

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

Definition: Frequent Itemset

- **Itemset**
  - A collection of one or more items
    - Example: {Milk, Bread, Diaper}
  - k-itemset
    - An itemset that contains k items
  - transaction *tj* contains an itemset
- **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

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

Definition: Frequent Itemset

- **Itemset**
  - A collection of one or more items
    - Example: {Milk, Bread, Diaper}
  - k-itemset
    - An itemset that contains k items
  - transaction *tj* contains an itemset
- **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

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**

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

$$\boxed{\text{Support, } s(X \longrightarrow Y) = \frac{\sigma(X \cup Y)}{N}}$$

Definition: Association Rule

- Rule Evaluation Metrics
  - Support (s)
    - Fraction of transactions that contain both X and Y

$$\text{Support, } s(X \longrightarrow Y) = \frac{\sigma(X \cup Y)}{N}.$$

  - Confidence (c)
    - Measures how often items in Y appear in transactions that contain X

$$\text{Confidence, } c(X \longrightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)}.$$

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

Definition: Association Rule

- Rule Evaluation Metrics
  - Support (s)
    - Fraction of transactions that contain both X and Y

$$\text{Support}, s(X \longrightarrow Y) = \frac{\sigma(X \cup Y)}{N}$$

  - Confidence (c)
    - Measures how often items in Y appear in transactions that contain X

$$\text{Confidence}, c(X \longrightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)}.$$

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:

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

Definition: Association Rule

- Rule Evaluation Metrics
  - Support (s)
    - Fraction of transactions that contain both X and Y

$$\text{Support}, s(X \longrightarrow Y) = \frac{\sigma(X \cup Y)}{N}$$

  - Confidence (c)
    - Measures how often items in Y appear in transactions that contain X

$$\text{Confidence}, c(X \longrightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)}.$$

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:

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

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

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

Mining Association Rules

- ✓ a rule that has very low support may occur simply by chance
- ✓ Confidence measures the reliability of the inference made by a rule

Mining Association Rules

- ✓ a rule that has very low support may occur simply by chance
- ✓ Confidence measures the reliability of the inference made by a rule

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)

Mining Association Rules

✓ a rule that has very low support may occur simply by chance
✓ Confidence measures the reliability of the inference made by a rule

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} $\rightarrow$ {Beer} (s=0.4, c=0.67)
{Milk,Beer} $\rightarrow$ {Diaper} (s=0.4, c=1.0)
{Diaper,Beer} $\rightarrow$ {Milk} (s=0.4, c=0.67)
{Beer} $\rightarrow$ {Milk,Diaper} (s=0.4, c=0.67)
{Diaper} $\rightarrow$ {Milk,Beer} (s=0.4, c=0.5)
{Milk} $\rightarrow$ {Diaper,Beer} (s=0.4, c=0.5)

Observations:

- Rules originating from the same itemset have identical support but can have different confidence
- Thus, we may decouple the support and confidence requirements

Association Rule Mining Task

- Given a set of transactions T, the goal of association rule mining is to find all rules having
  - support ≥ *minsup* threshold
  - confidence ≥ *minconf* threshold

Association Rule Mining Task

- Given a set of transactions T, the goal of association rule mining is to find all rules having
  - support $\geq$ *minsup* threshold
  - confidence $\geq$ *minconf* threshold

- Brute-force approach:
  - List all possible association rules
  - Compute the support and confidence for each rule
  - Prune rules that fail the *minsup* and *minconf* thresholds

  $\Rightarrow$ Computationally prohibitive!

Mining Association Rules

If the itemset is infrequent, then all candidate rules can be pruned immediately without compute their confidence values

Mining Association Rules

If the itemset is infrequent, then all candidate rules can be pruned immediately without compute their confidence values

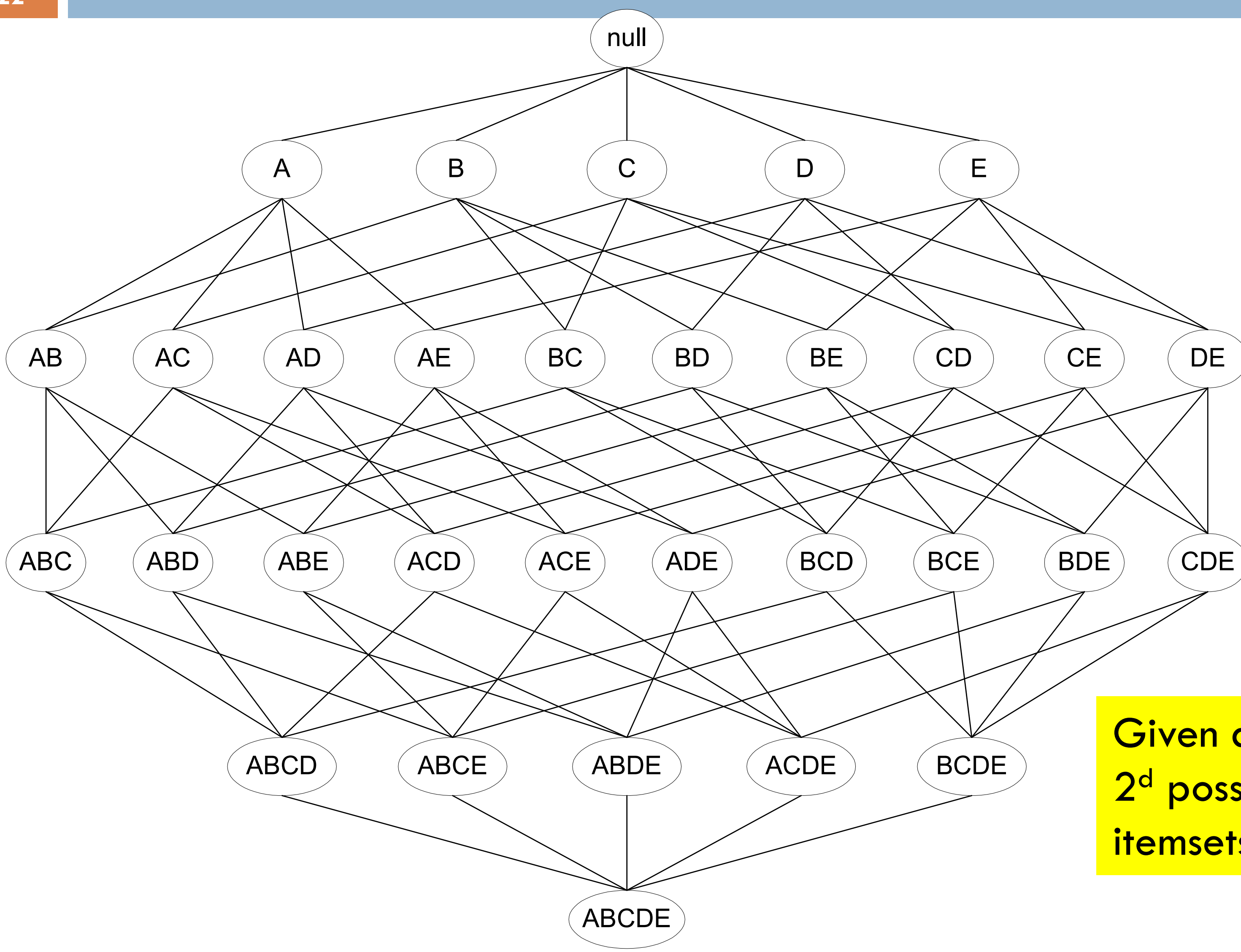
- Two-step approach:
  1. Frequent Itemset Generation
     - Generate all itemsets whose support $\geq$ minsup

Mining Association Rules

If the itemset is infrequent, then all candidate rules can be pruned immediately without compute their confidence values

- Two-step approach:
  1. Frequent Itemset Generation
     - Generate all itemsets whose support ≥ minsup
  2. Rule Generation
     - Generate high confidence rules from each frequent itemset, where each rule is a binary partitioning of a frequent itemset

Frequent Itemset Generation



Given d items, there are 2^d possible candidate itemsets

Frequent Itemset Generation

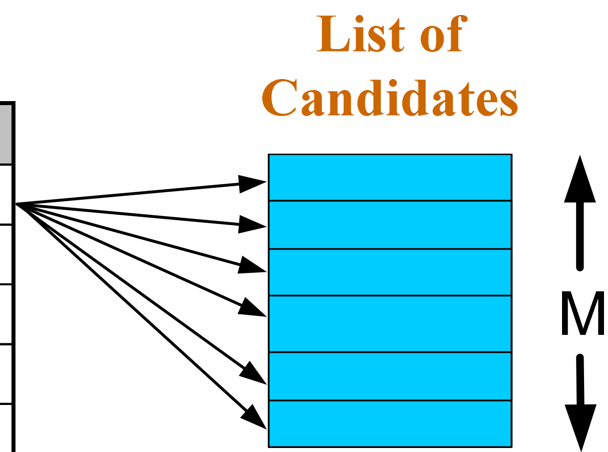
- Brute-force approach:
  - Each itemset in the lattice is a candidate frequent itemset
  - Count the support of each candidate by scanning the database

Transactions

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

N

w



  - Match each transaction against every candidate
  - Expensive!!!

Frequent Itemset Generation Strategies

- Reduce the number of candidates (M)
  - Complete search: $M=2^d$
  - Use pruning techniques to reduce M

Frequent Itemset Generation Strategies

- Reduce the number of candidates (M)
  - Complete search: $M=2^d$
  - Use pruning techniques to reduce M

- Reduce the number of comparisons (NM)
  - Use efficient data structures to store the candidates or transactions
  - No need to match every candidate against every transaction

26

Reducing Number of Candidates

Reducing Number of Candidates

- Apriori principle:
  - If an itemset is frequent, then all of its subsets must also be frequent

Reducing Number of Candidates

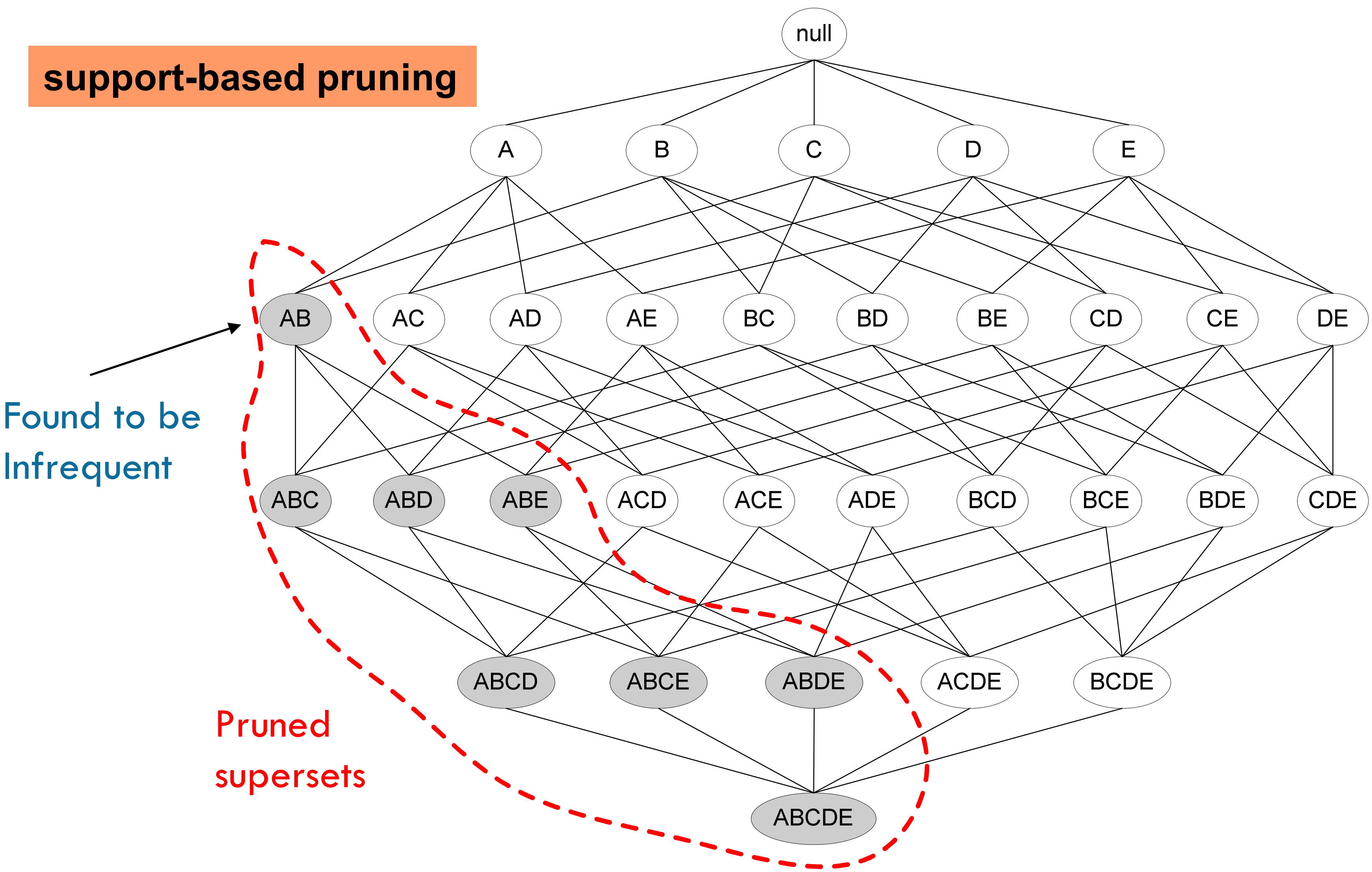
- Apriori principle:
  - If an itemset is frequent, then all of its subsets must also be frequent

- Apriori principle holds due to the following property of the support measure:

$$\forall X, Y : (X \subseteq Y) \Rightarrow s(X) \geq s(Y)$$

  - Support of an itemset never exceeds the support of its subsets

Illustrating Apriori Principle



Illustrating Apriori Principle

Item	Count
Bread	**4**
Coke	**2**
Milk	**4**
Beer	**3**
Diaper	**4**
Eggs	**1**

Items (1-itemsets)

Minimum Support=0.6(3)

Illustrating Apriori Principle

Minimum Support=0.6(3)

Item	Count
Bread	**4**
Coke	**2**
Milk	**4**
Beer	**3**
Diaper	**4**
Eggs	**1**

Items (1-itemsets)

Itemset	Count
{Bread,Milk}	**3**
{Bread,Beer}	**2**
{Bread,Diaper}	**3**
{Milk,Beer}	**2**
{Milk,Diaper}	**3**
{Beer,Diaper}	**3**

Pairs (2-itemsets)

(No need to generate candidates involving Coke or Eggs)

Illustrating Apriori Principle

Minimum Support=0.6(3)

Items (1-itemsets)

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Pairs (2-itemsets)

(No need to generate candidates involving Coke or Eggs)

Itemset	Count
{Bread,Milk}	3
{Bread,Beer}	2
{Bread,Diaper}	3
{Milk,Beer}	2
{Milk,Diaper}	3
{Beer,Diaper}	3

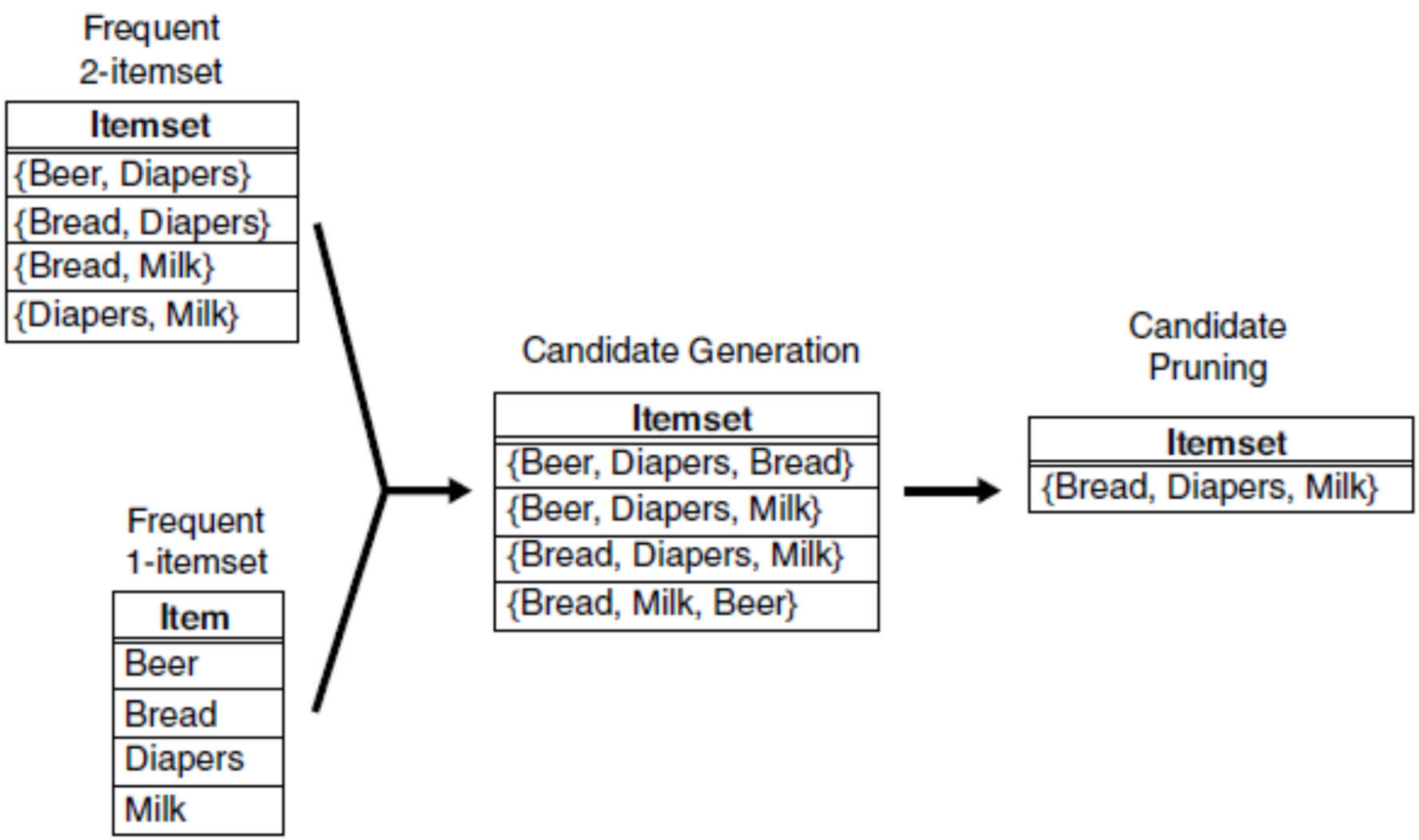
Triplets (3-itemsets)

Itemset	Count
{Bread,Milk,Diaper}	3

Apriori Algorithm

- Method:
  - Let k=1
  - Generate frequent itemsets of length 1
  - Repeat until no new frequent itemsets are identified
    - Generate length (k+1) candidate itemsets from length k frequent itemsets
    - Prune candidate itemsets containing subsets of length k that are infrequent
    - Count the support of each candidate by scanning the DB
    - Eliminate candidates that are infrequent, leaving only those that are frequent
    - k=k+1

$F_{k-1} \times F_1$ Method



$F_{k-1} \times F_{k-1}$ Method

$$A = \{a_1, a_2, \ldots, a_{k-1}\} \text{ and } B = \{b_1, b_2, \ldots, b_{k-1}\}$$

$$a_i = b_i \text{ (for } i = 1, 2, \ldots, k-2) \text{ and } a_{k-1} \neq b_{k-1}$$

$F_{k-1} \times F_{k-1}$ Method

$$A = \{a_1, a_2, \ldots, a_{k-1}\} \text{ and } B = \{b_1, b_2, \ldots, b_{k-1}\}$$

$$a_i = b_i \text{ (for } i = 1, 2, \ldots, k-2) \text{ and } a_{k-1} \neq b_{k-1}$$

Frequent
2-itemset

Itemset
{Beer, Diapers}
{Bread, Diapers}
{Bread, Milk}
{Diapers, Milk}

Frequent
2-itemset

Itemset
{Beer, Diapers}
{Bread, Diapers}
{Bread, Milk}
{Diapers, Milk}

Candidate
Generation

Itemset
{Bread, Diapers, Milk}

Candidate
Pruning

Itemset
{Bread, Diapers, Milk}

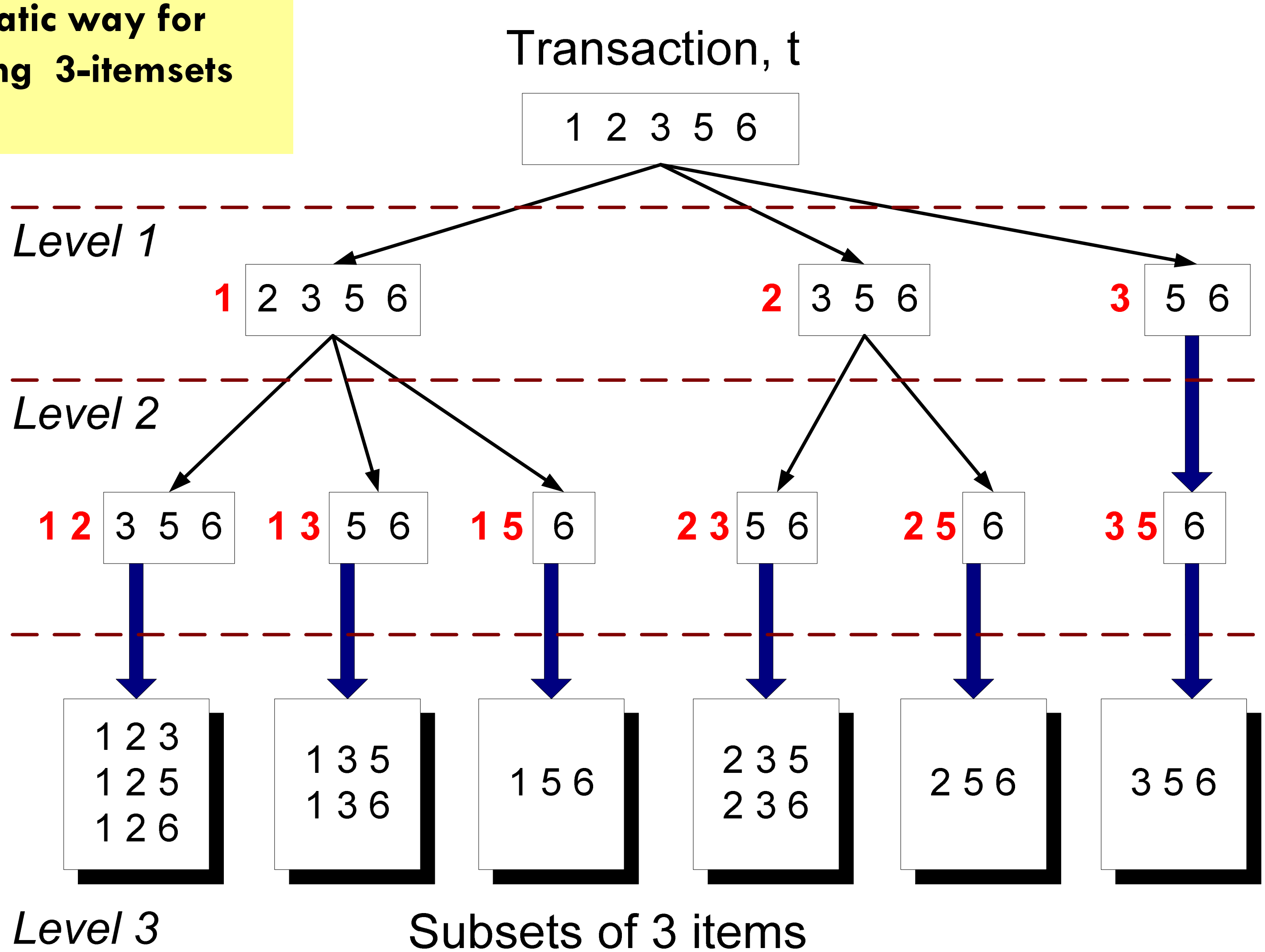
Reducing Number of Comparisons

Reducing Number of Comparisons

- Candidate counting:
  - One approach :compare each transaction against every candidate itemset
  - update the support counts of candidates contained in the transaction
  - An alternative approach :enumerate the itemsets contained in each transaction
  - use them to update the support counts of their respective candidate itemsets

Subset Operation

a systematic way for enumerating 3-itemsets



- We still have to determine whether each enumerated 3-itemset corresponds to an existing candidate itemset
- This matching operation can be performed efficiently using a hash tree structure
- instead of comparing each itemset in the transaction with every candidate itemset, it is matched only against candidate itemsets that belong to the same bucket

Generate Hash Tree

Suppose you have 15 candidate itemsets of length 3:

{1 4 5}, {1 2 4}, {4 5 7}, {1 2 5}, {4 5 8}, {1 5 9}, {1 3 6}, {2 3 4}, {5 6 7}, {3 4 5}, {3 5 6}, {3 5 7}, {6 8 9}, {3 6 7}, {3 6 8}

You need:

- Hash function
- Max leaf size: max number of itemsets stored in a leaf node (if number of candidate itemsets exceeds max leaf size, split the node)

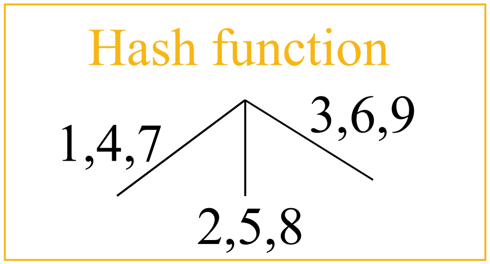
Generate Hash Tree

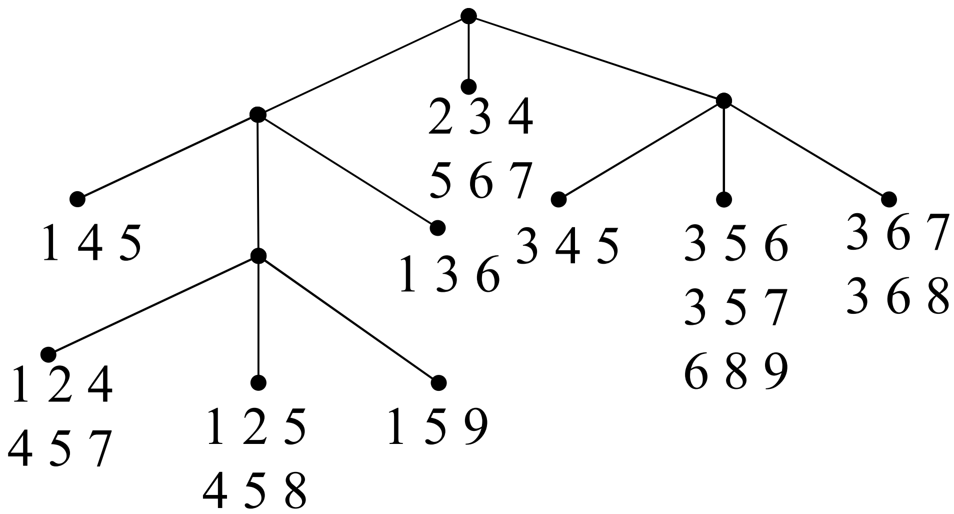
Suppose you have 15 candidate itemsets of length 3:

{1 4 5}, {1 2 4}, {4 5 7}, {1 2 5}, {4 5 8}, {1 5 9}, {1 3 6}, {2 3 4}, {5 6 7}, {3 4 5}, {3 5 6}, {3 5 7}, {6 8 9}, {3 6 7}, {3 6 8}

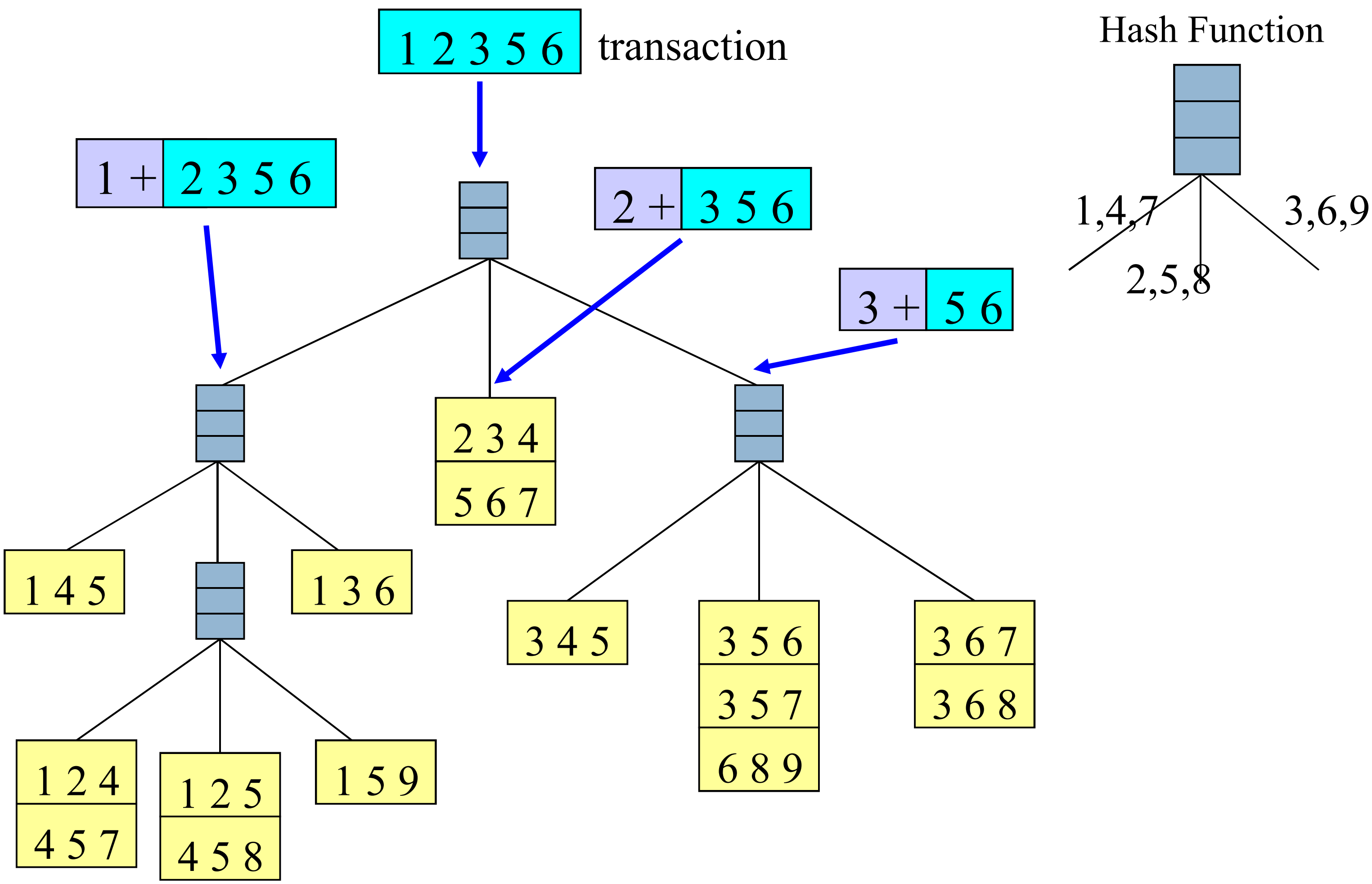
You need:

- Hash function
- Max leaf size: max number of itemsets stored in a leaf node (if number of candidate itemsets exceeds max leaf size, split the node)

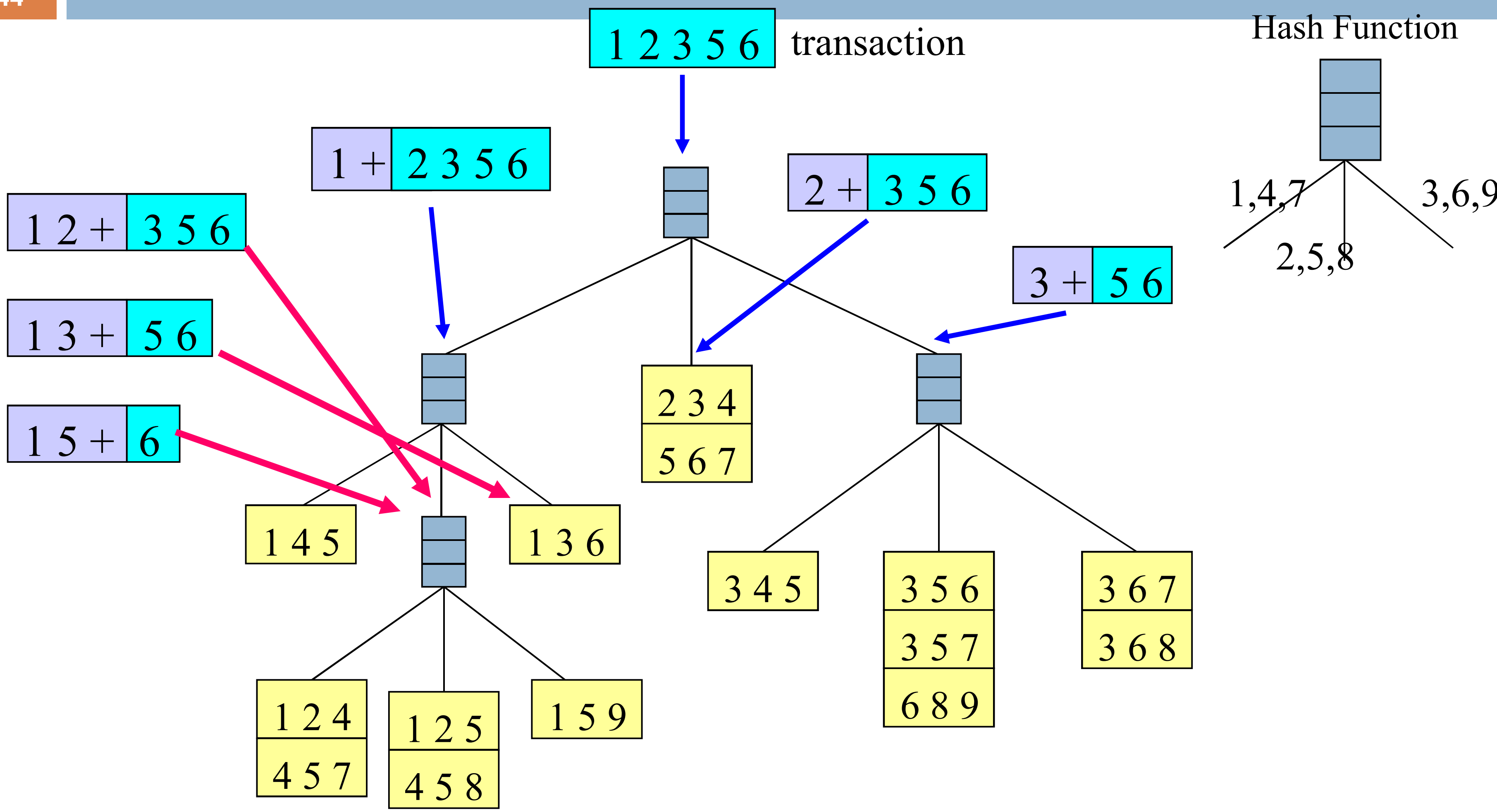




Subset Operation Using Hash Tree



Subset Operation Using Hash Tree



45

Rule Generation

Rule Generation

- Given a frequent itemset L, find all non-empty subsets $f \subset L$ such that $f \rightarrow L - f$ satisfies the minimum confidence requirement

Rule Generation

- Given a frequent itemset L, find all non-empty subsets $f \subset L$ such that $f \rightarrow L - f$ satisfies the minimum confidence requirement
  - If {A,B,C,D} is a frequent itemset, candidate rules:

$ABC \rightarrow D$,	$ABD \rightarrow C$,	$ACD \rightarrow B$,	$BCD \rightarrow A$,
$A \rightarrow BCD$,	$B \rightarrow ACD$,	$C \rightarrow ABD$,	$D \rightarrow ABC$
$AB \rightarrow CD$,	$AC \rightarrow BD$,	$AD \rightarrow BC$,	$BC \rightarrow AD$,
$BD \rightarrow AC$,	$CD \rightarrow AB$,		

- If $|L| = k$, then there are $2^k - 2$ candidate association rules (ignoring $L \rightarrow \emptyset$ and $\emptyset \rightarrow L$)

Rule Generation

- confidence of rules generated from the same itemset has the following property
  - E.g., Suppose {A,B,C,D} is a frequent 4-itemset:

$$c(ABC \rightarrow D) \geq c(AB \rightarrow CD) \geq c(A \rightarrow BCD)$$

Rule Generation

frequent itemset

Theorem 6.2. *If a rule $X \longrightarrow Y - X$ does not satisfy the confidence threshold, then any rule $X' \longrightarrow Y - X'$, where X' is a subset of X, must not satisfy the confidence threshold as well.*

Rule Generation

frequent itemset

Theorem 6.2. *If a rule $X \longrightarrow Y - X$ does not satisfy the confidence threshold, then any rule $X' \longrightarrow Y - X'$, where X' is a subset of X, must not satisfy the confidence threshold as well.*

$$X \longrightarrow Y - X \qquad \sigma(Y)/\sigma(X)$$

$$X' \longrightarrow Y - X' \qquad \sigma(Y)/\sigma(X')$$

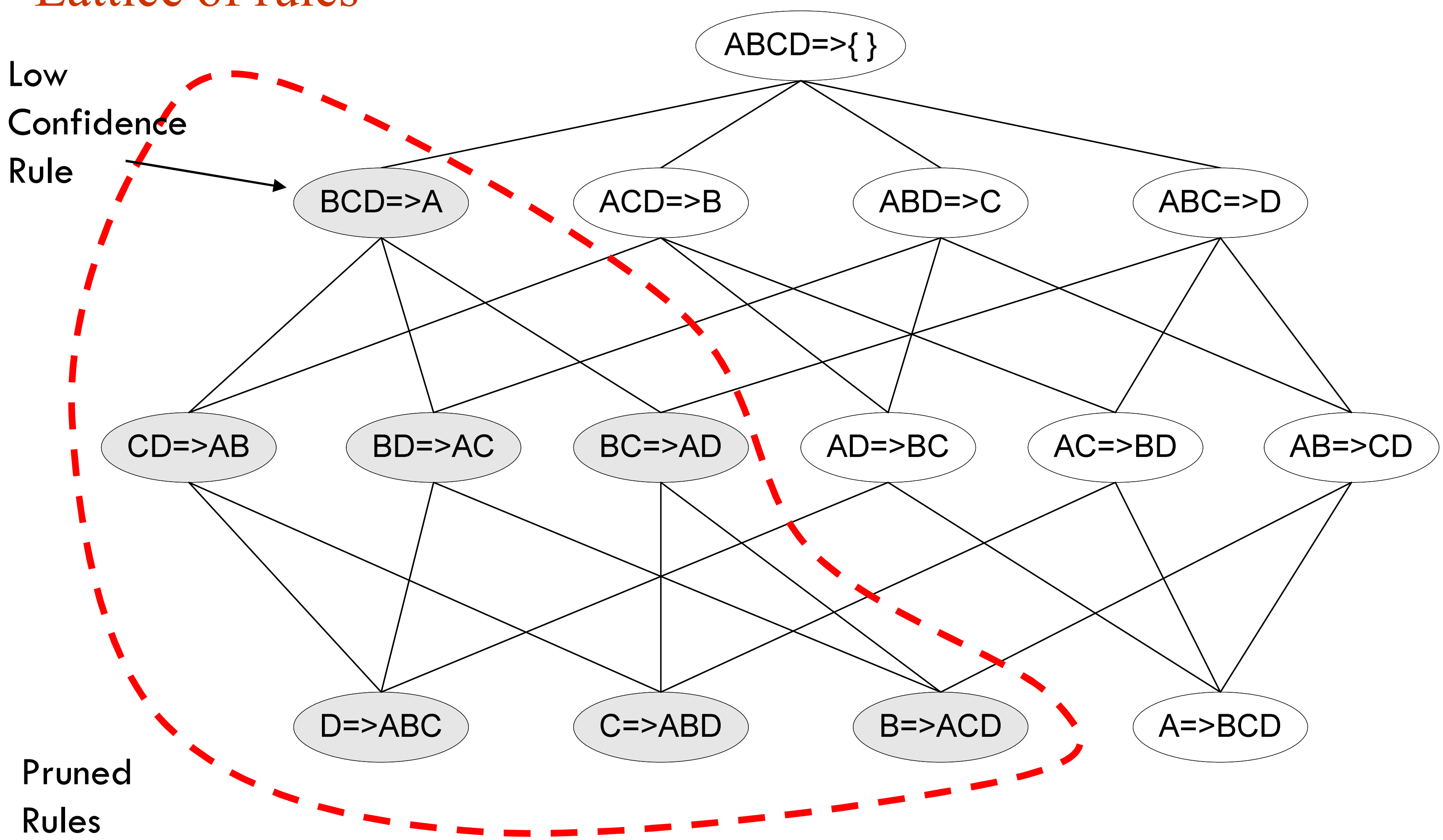
$$\sigma(X') \geq \sigma(X)$$

Rule Generation in *Apriori* Algorithm

- level-wise approach for generating association rules
- each level corresponds to the number of items that belong to the rule consequent
- all the high-confidence rules that have only one item in the rule consequent are extracted
- These rules are then used to generate new candidate rules

Rule Generation for Apriori Algorithm

Lattice of rules



FP-growth Algorithm

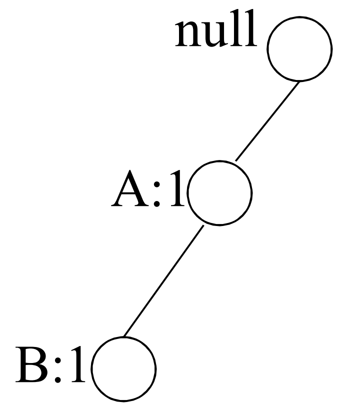
FP-growth Algorithm

- an alternative algorithm **FP-growth** takes a different approach to discovering frequent itemsets
- not subscribe to the generate-and-test
- 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
- determine the support count of each item
- Infrequent items are discarded
- frequent items are sorted in decreasing support counts

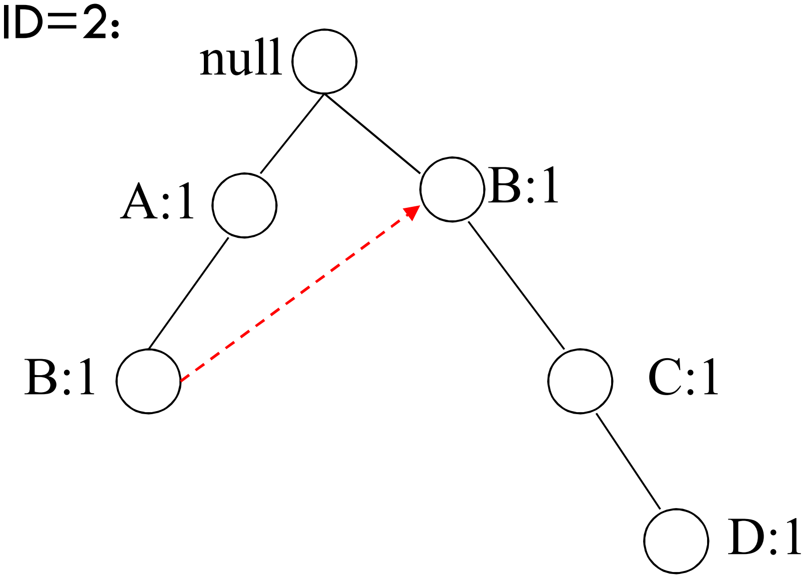
FP-tree construction

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	{B,C}
8	{A,B,C}
9	{A,B,D}
10	{B,C,E}

After reading TID=1:



After reading TID=2:



- data set is scanned once to determine support count of each item. Infrequent items are discarded

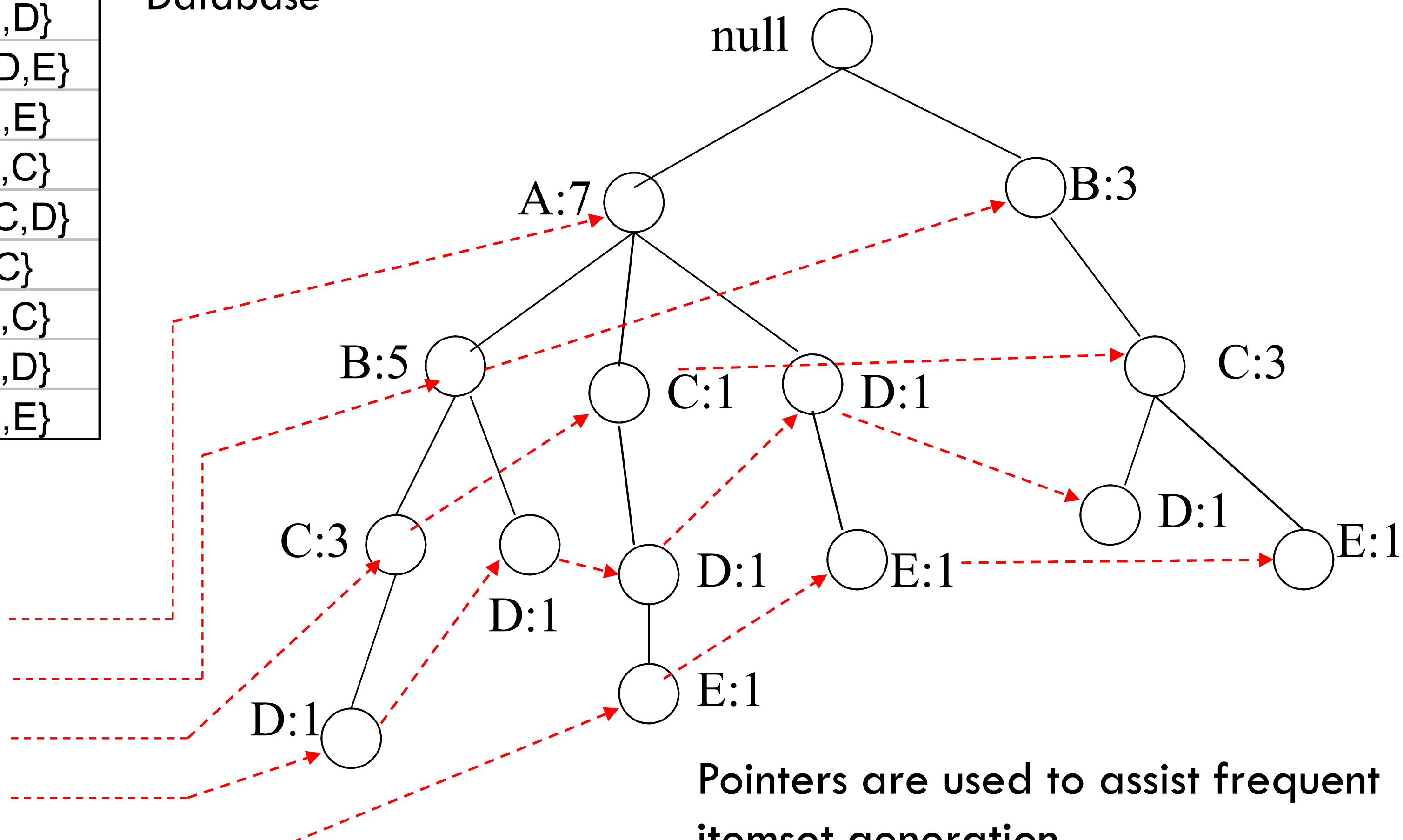
- frequent items are sorted in decreasing support counts.

- *a* is the most frequent item, followed by *b*, *c*, *d*, and *e*.

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	{B,C}
8	{A,B,C}
9	{A,B,D}
10	{B,C,E}

Transaction Database



Pointers are used to assist frequent itemset generation

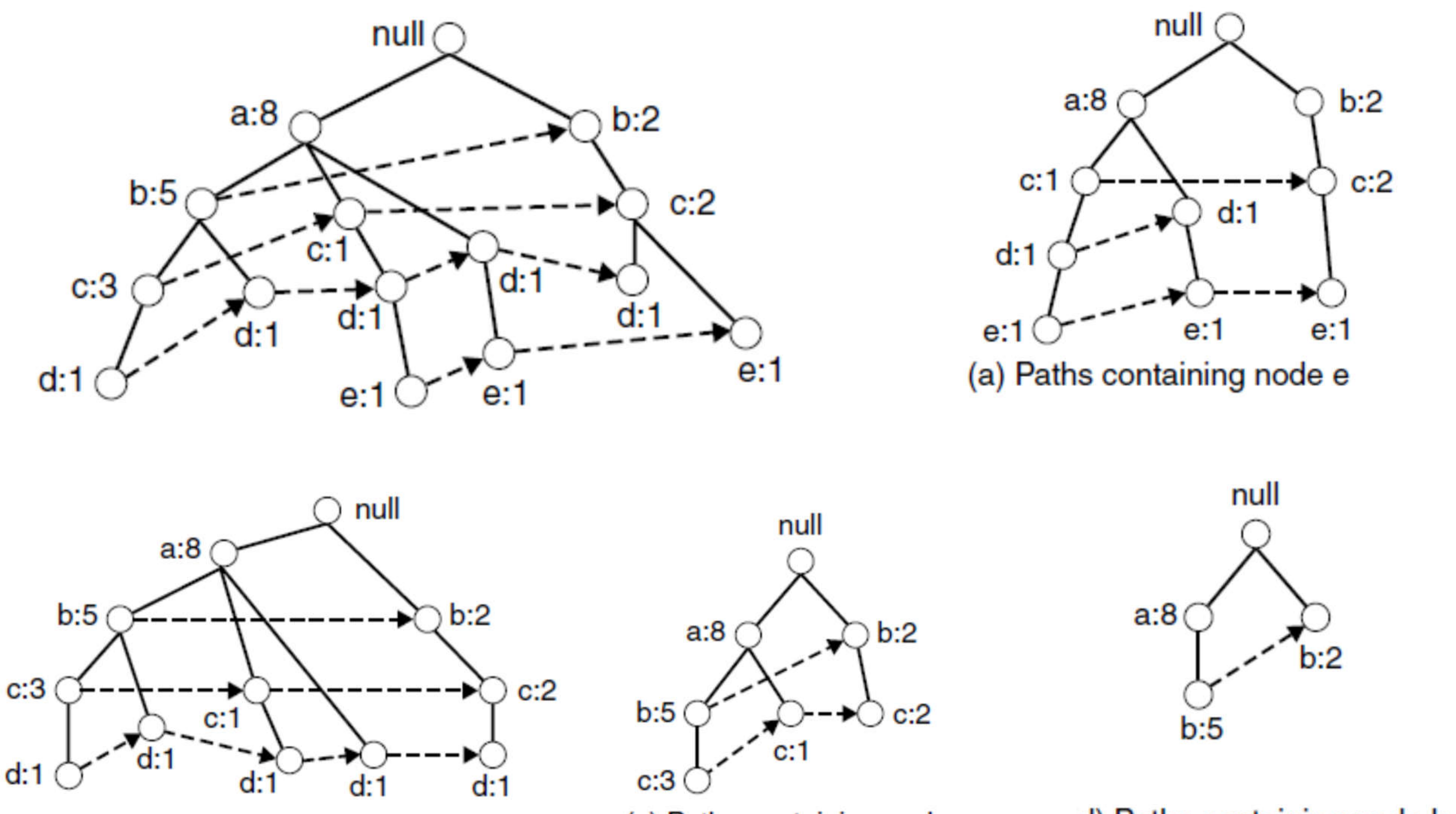
FP-Growth Algorithm

- ✓ FP-growth is an algorithm that generates frequent itemsets from an FP-tree by exploring the tree in a bottom-up fashion
- ✓ algorithm looks for frequent itemsets ending in e first, followed by d, c, b, and finally, a.

Table 6.6. The list of frequent itemsets ordered by their corresponding suffixes.

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}

FP-Growth Algorithm



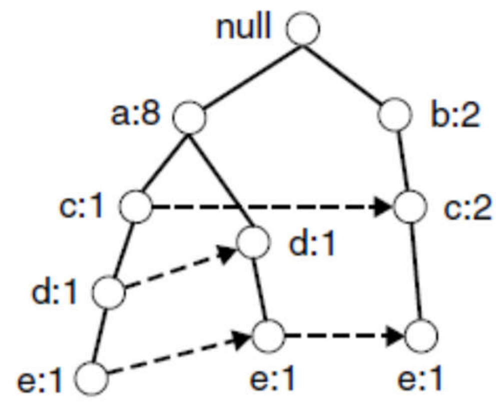
(a) Paths containing node e

(b) Paths containing node d

(c) Paths containing node c

d) Paths containing node b

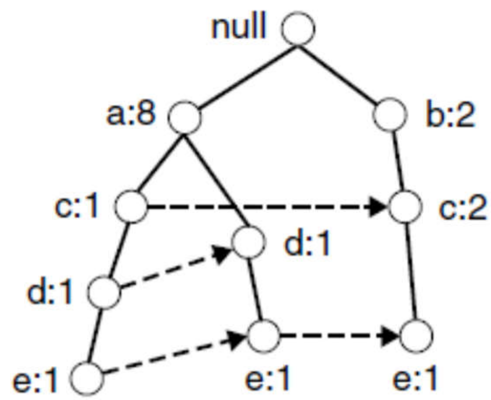
FP-Growth Algorithm



(a) Paths containing node e

- gather all the paths containing node e: **prefix paths**
- From the prefix paths : support count for e : {e} is declared a frequent itemset

FP-Growth Algorithm

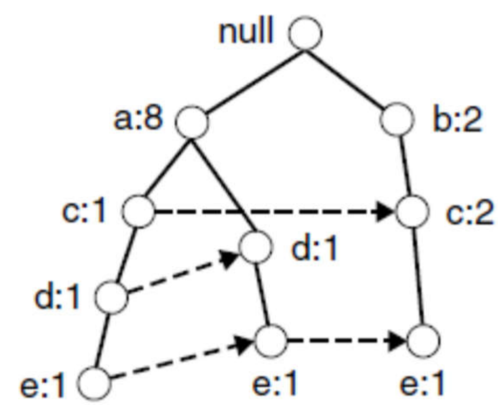


(a) Paths containing node e

- gather all the paths containing node e: **prefix paths**
- From the prefix paths : support count for e : {e} is declared a frequent itemset
- Because {e} is frequent, the algorithm has to solve the subproblems of finding frequent itemsets ending in *de*, *ce*, *be*, and *ae*
- convert the prefix paths into a **conditional FP-tree**

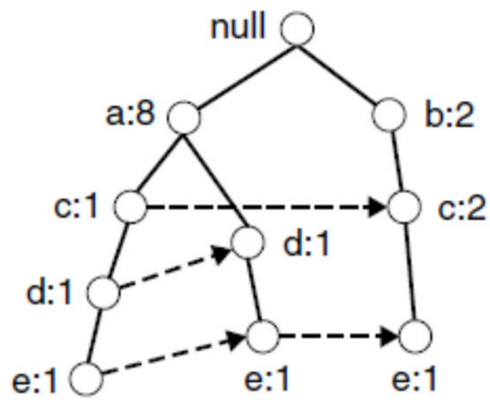
conditional FP-tree :

- support counts along the prefix paths must be updated
- prefix paths are truncated by removing the nodes for e
- reflect only transactions that contain e and the subproblems of finding frequent itemsets ending in *de*, *ce*, *be*, and *ae* no longer need information about node e.
- Some of the items may no longer be frequent

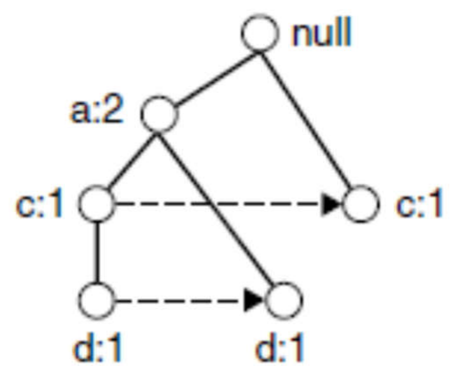


(a) Paths containing node e

FP-Growth Algorithm



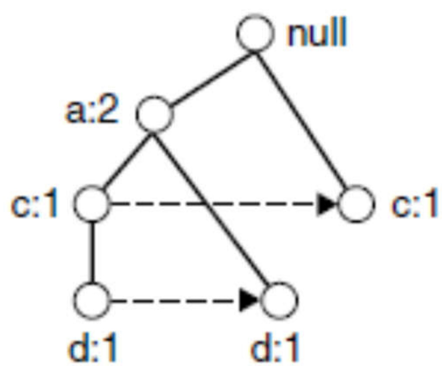
(a) Paths containing node e



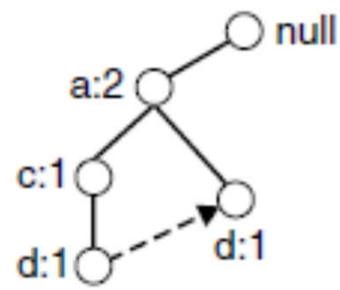
(b) Conditional FP-tree for e

✓ FP-growth uses the conditional FP-tree for finding frequent itemsets ending in *de*, *ce*, and *ae*

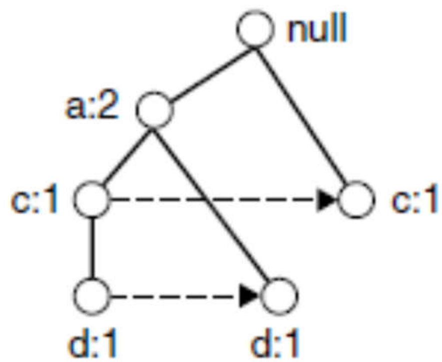
✓ find the frequentitemsets ending in de, the prefix paths for d are gathered from the conditional FP-tree for e



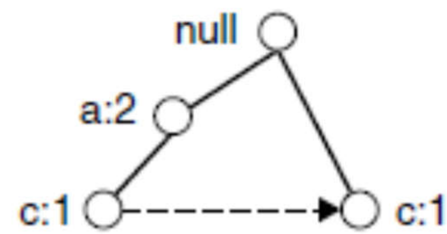
(b) Conditional FP-tree for e



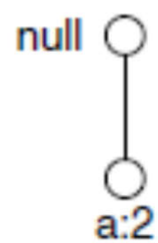
(c) Prefix paths ending in de



(b) Conditional FP-tree for e

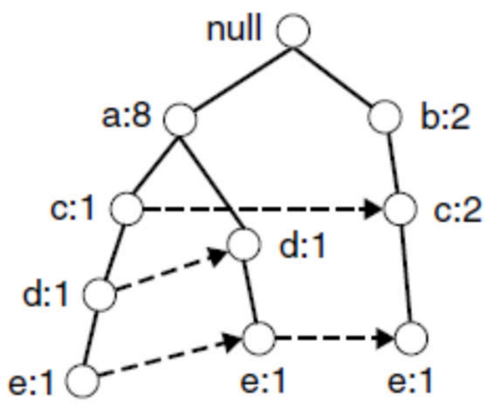


(e) Prefix paths ending in ce

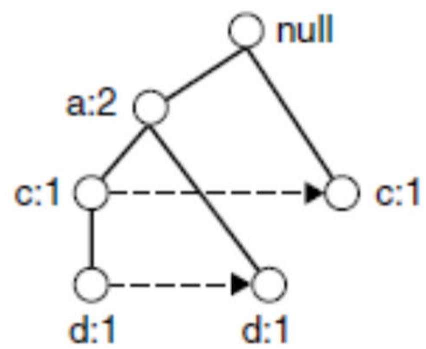


(f) Prefix paths ending in ae

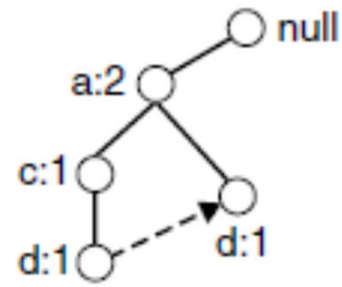
FP-Growth Algorithm



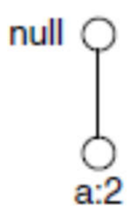
(a) Paths containing node e



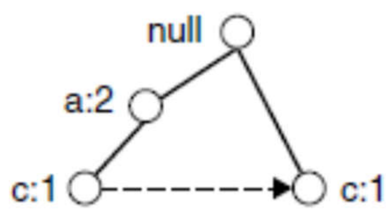
(b) Conditional FP-tree for e



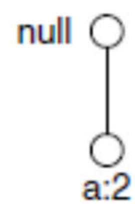
(c) Prefix paths ending in de



(d) Conditional FP-tree for de



(e) Prefix paths ending in ce



(f) Prefix paths ending in ae

Compact Representation of frequent itemsets

Maximal Frequent Itemset

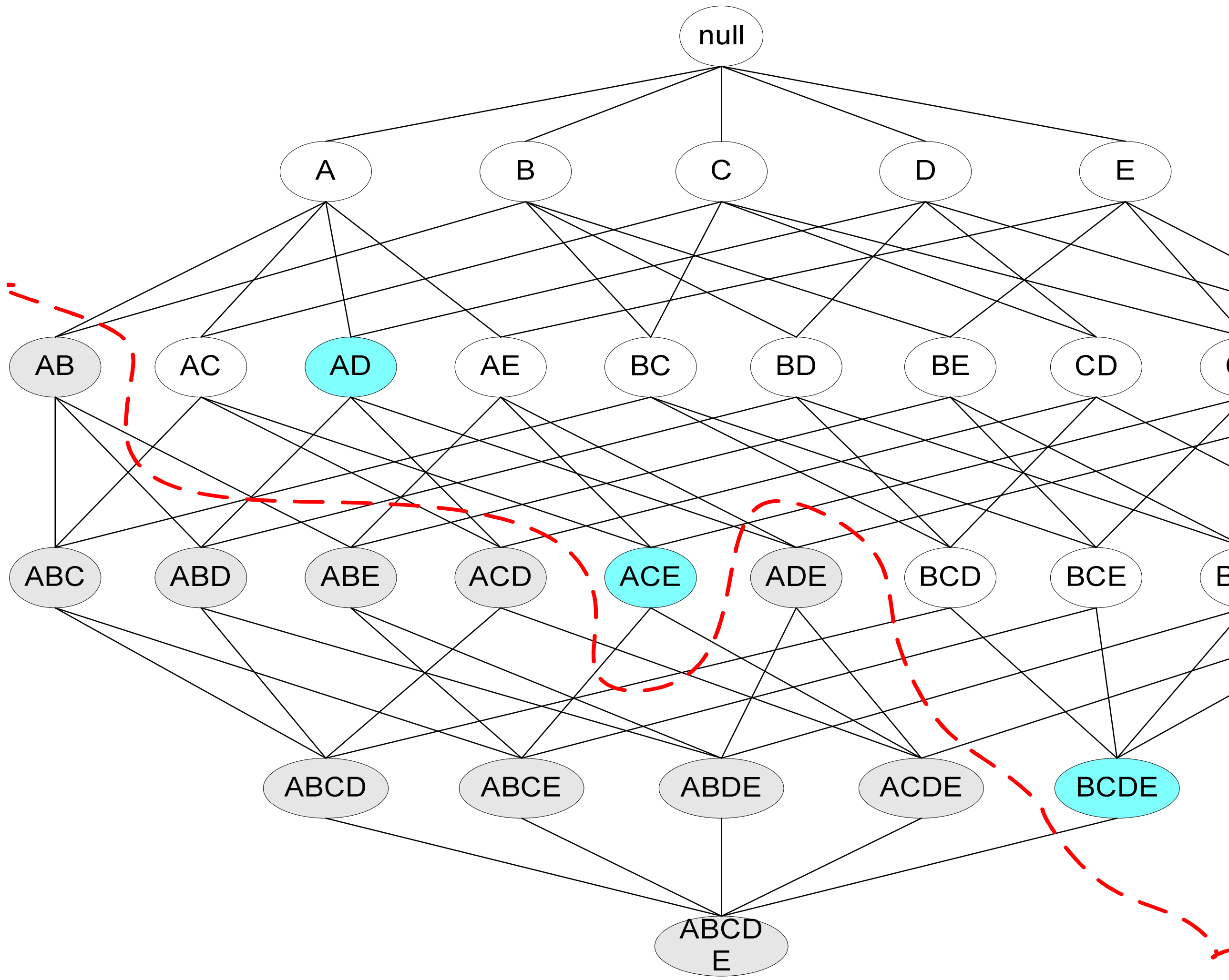
- number of frequent itemsets produced from a transaction data set can be very large
- identify a small representative set of itemsets from which all other frequent itemsets can be derived
- Two such representations
  - maximal frequent itemsets
  - closed frequent itemsets.

Maximal Frequent Itemset

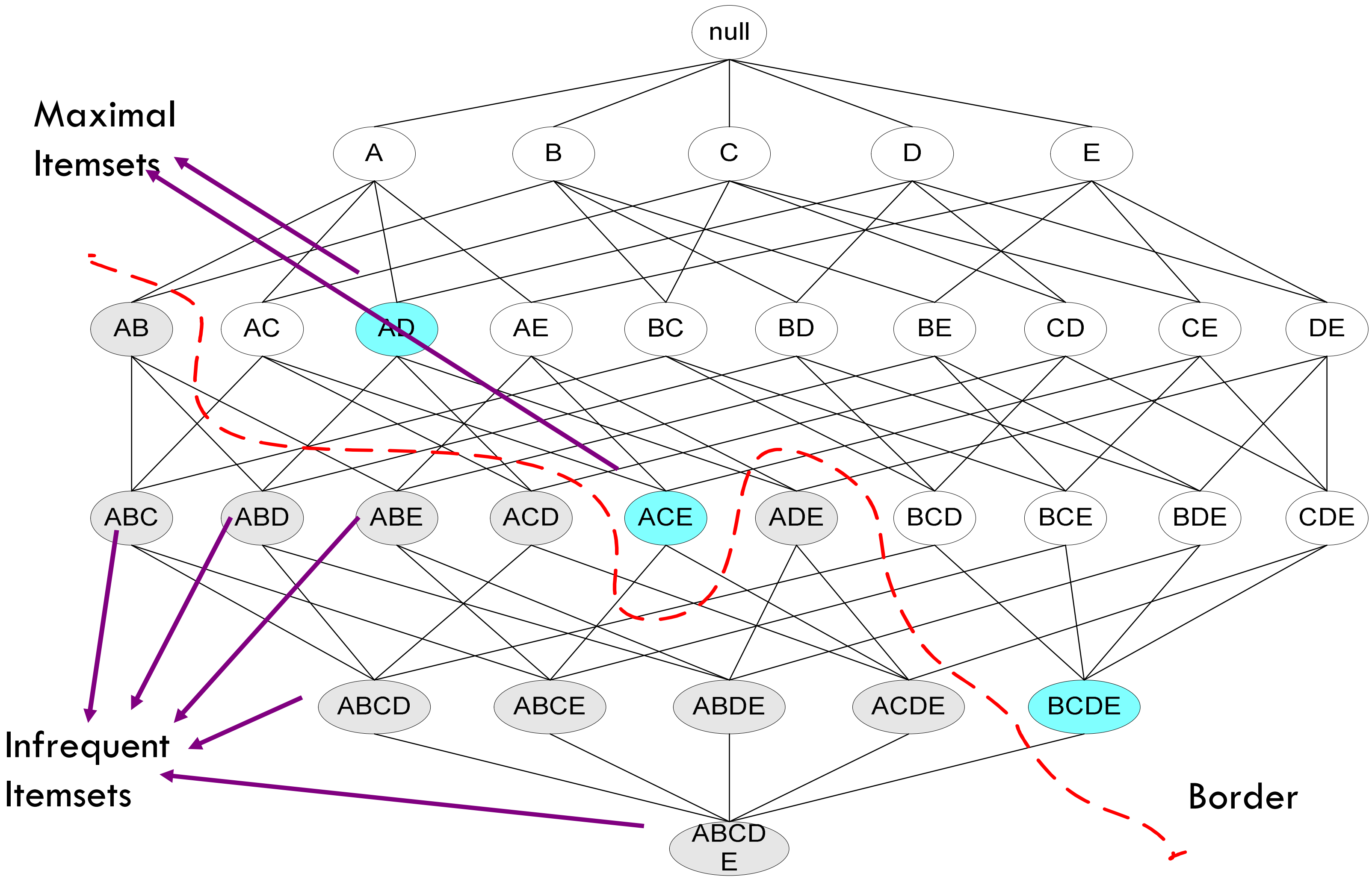
- number of frequent itemsets produced from a transaction data set can be very large
- identify a small representative set of itemsets from which all other frequent itemsets can be derived
- Two such representations
  - maximal frequent itemsets
  - closed frequent itemsets.

An itemset is maximal frequent if none of its immediate supersets is frequent

Maximal Frequent Itemset



Maximal Frequent Itemset



Maximal Frequent Itemset

- ✓ Maximal frequent itemsets effectively provide a compact representation of frequent itemsets
- ✓ they form the smallest set of itemsets from which all frequent itemsets can be derived
- ✓ an efficient algorithm exists to explicitly find the maximal frequent itemsets without having to enumerate all their subsets
- ✓ Despite providing a compact representation, maximal frequent itemsets do not contain the support information of their subsets

Maximal Frequent Itemset

- ✓ Maximal frequent itemsets effectively provide a compact representation of frequent itemsets
- ✓ they form the smallest set of itemsets from which all frequent itemsets can be derived
- ✓ an efficient algorithm exists to explicitly find the maximal frequent itemsets without having to enumerate all their subsets
- ✓ Despite providing a compact representation, maximal frequent itemsets do not contain the support information of their subsets

a minimal representation of frequent itemsets that preserves the support information

Closed Itemset

- An itemset is closed if none of its immediate supersets has the same support as the itemset

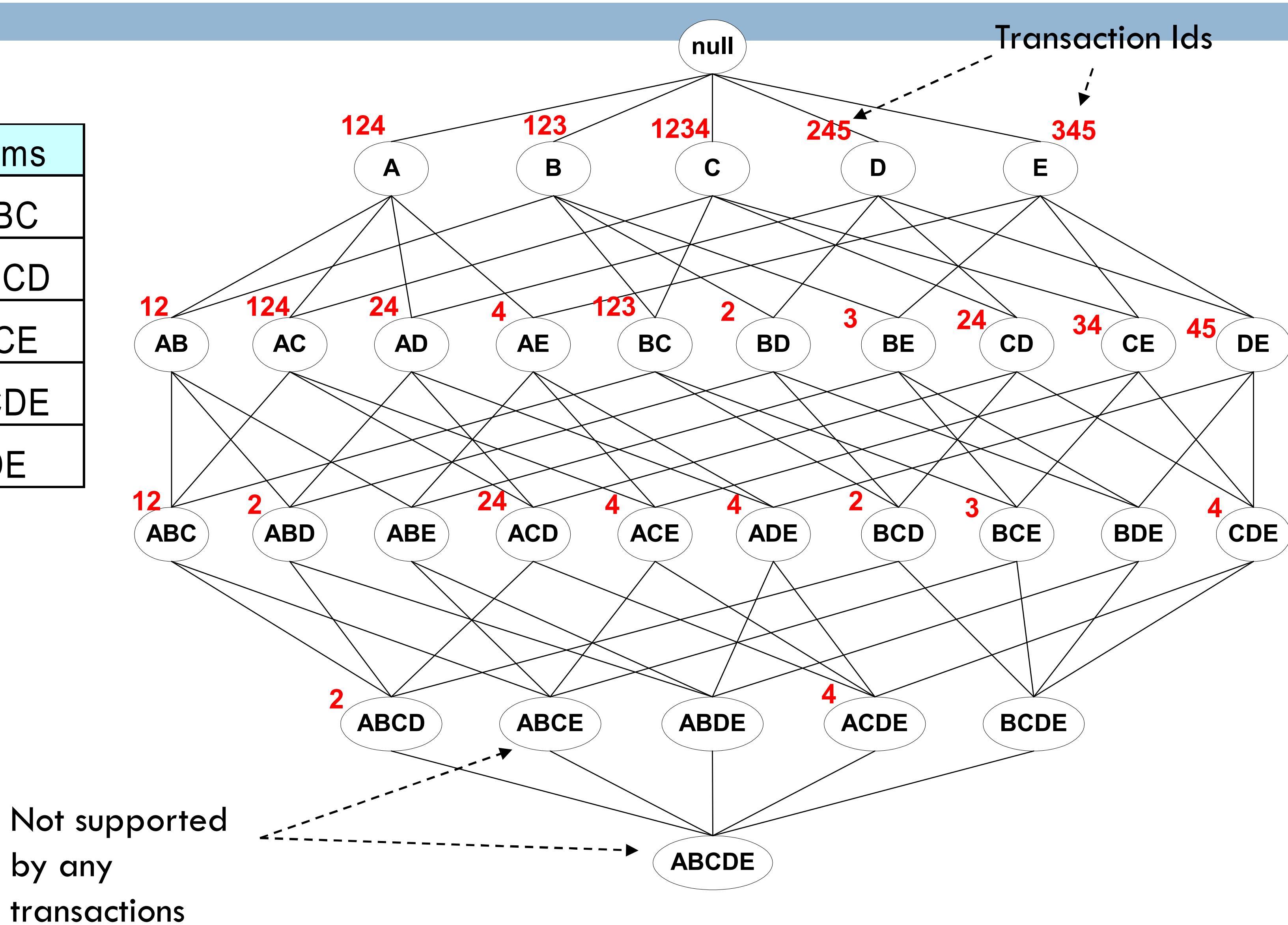
TID	Items
1	{A,B}
2	{B,C,D}
3	{A,B,C,D}
4	{A,B,D}
5	{A,B,C,D}

Itemset	Support
{A}	4
{B}	5
{C}	3
{D}	4
{A,B}	4
{A,C}	2
{A,D}	3
{B,C}	3
{B,D}	4
{C,D}	3

Itemset	Support
{A,B,C}	2
{A,B,D}	3
{A,C,D}	2
{B,C,D}	3
{A,B,C,D}	2

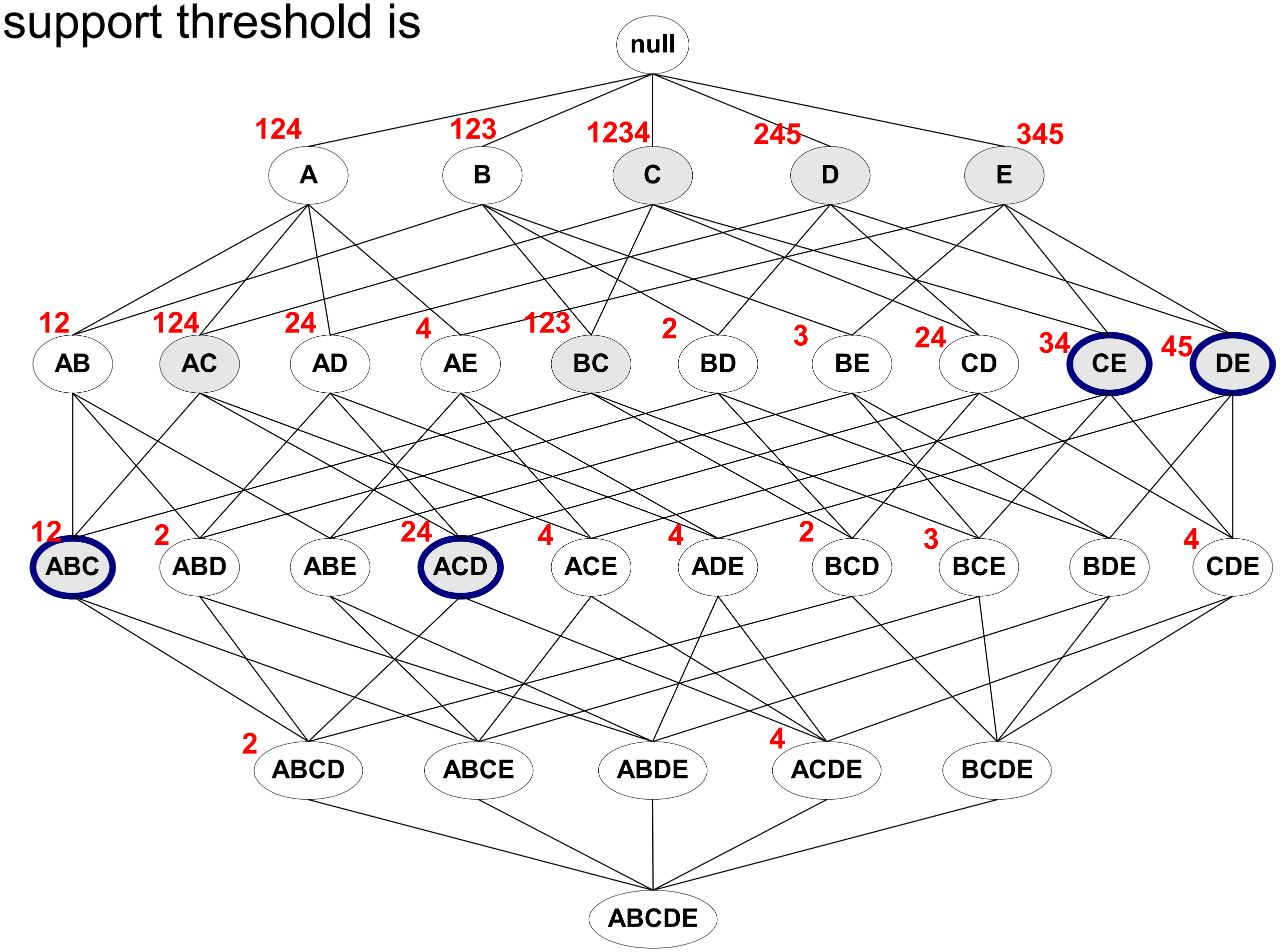
Closed Itemsets

TID	Items
1	ABC
2	ABCD
3	BCE
4	ACDE
5	DE



Frequent Closed Itemsets

assuming that the support threshold is 40%,



Maximal vs Closed Itemsets