Data Mining

Chapter 5
Association Analysis: Basic Concepts

Introduction to Data Mining, 2nd Edition

by

Tan, Steinbach, Karpatne, Kumar

Association Rule Mining

- 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

$\{Diaper\} \rightarrow \{Beer\}$,
$\{Milk, Bread\} \rightarrow \{Eggs,Coke\}$,
$\{Beer, Bread\} \rightarrow \{Milk\}$,

Implication means co-occurrence, not causality!

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

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:

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

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

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

Association Rule Mining Task

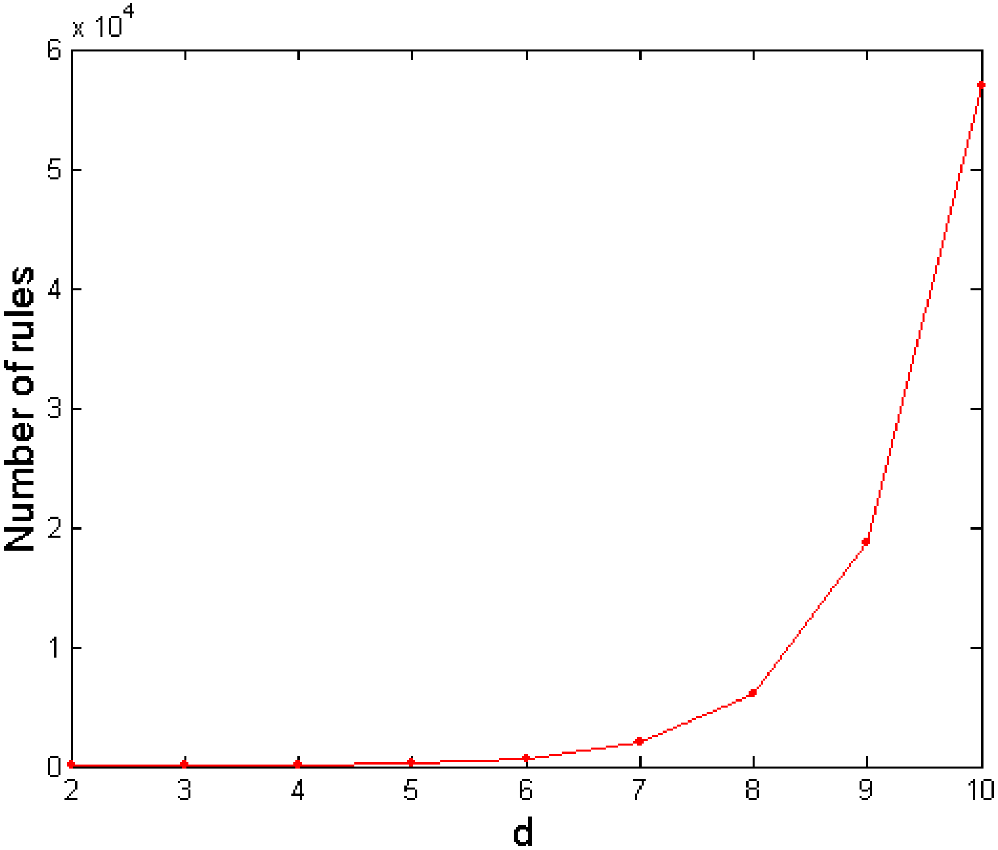
- Given a set of transactions T, the goal of association rule mining is to find all rules having
  – support ≥ *minsup* threshold
  – confidence ≥ *minconf* threshold

- 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!

Computational Complexity

- Given d unique items:
  - Total number of itemsets = 2^d
  - Total number of possible association rules:



$$R = \sum_{k=1}^{d-1}\left[\binom{d}{k} \times \sum_{j=1}^{d-k}\binom{d-k}{j}\right] = 3^d - 2^{d+1} + 1$$

If d=6, R = 602 rules

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} $\rightarrow$ {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:

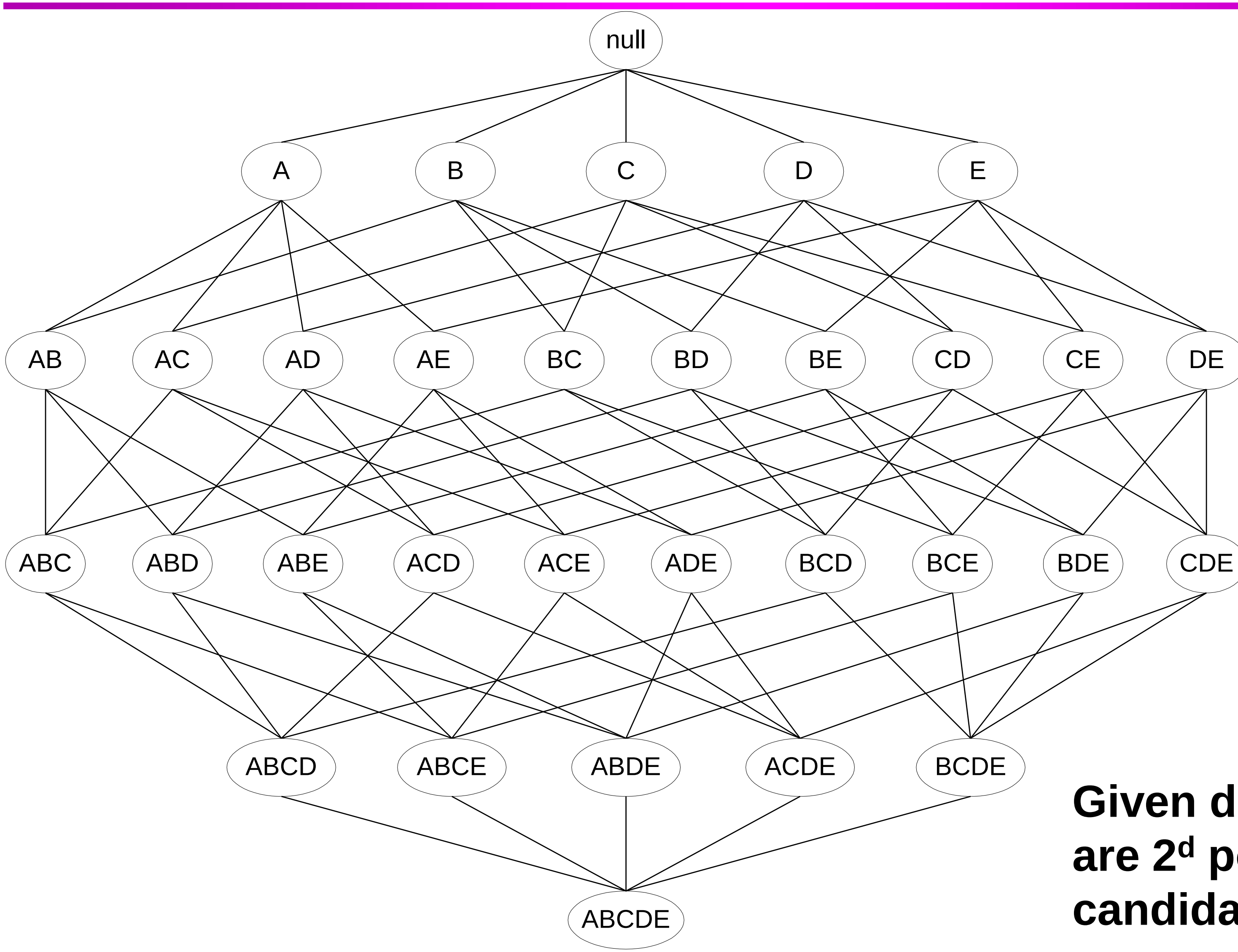
- All the above rules are binary partitions of the same itemset: {Milk, Diaper, Beer}
- Rules originating from the same itemset have identical support but can have different confidence
- Thus, we may decouple the support and confidence requirements

Mining Association Rules

- Two-step approach:
  1. Frequent Itemset Generation
     – Generate all itemsets whose support $\geq$ 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 is still computationally expensive

Frequent Itemset Generation



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

Frequent Itemset Generation

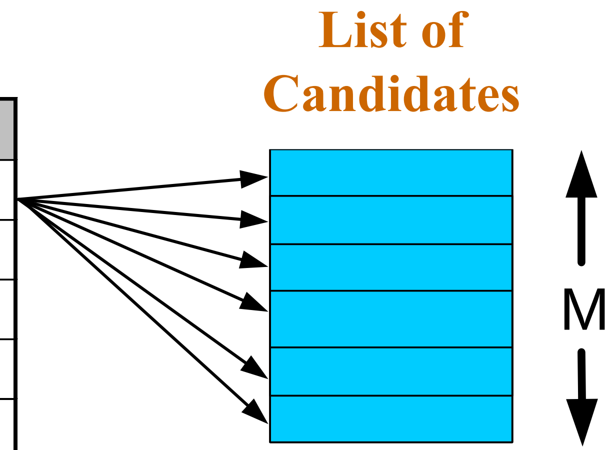
- Brute-force approach:
  - Each itemset in the lattice is a candidate frequent itemset
  - Count the support of each candidate by scanning the database

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
  - Complexity ~ O(NMw) => Expensive since $M = 2^d$!!!

Frequent Itemset Generation Strategies

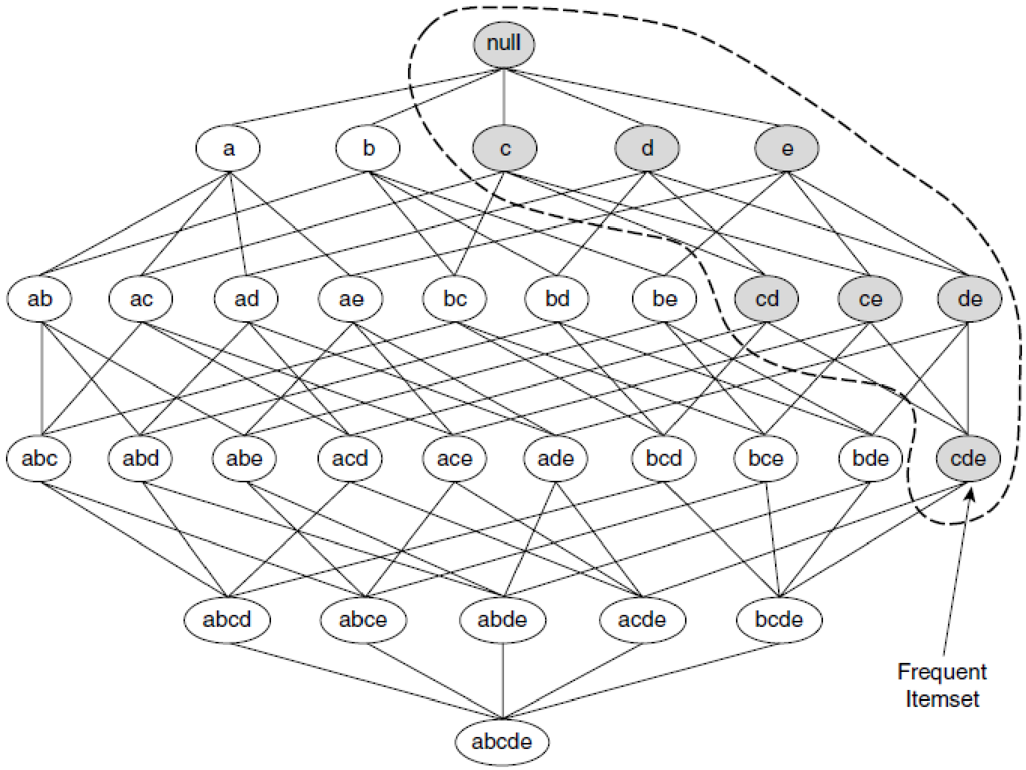
- Reduce the number of candidates (M)
  - Complete search: $M=2^d$
  - Use pruning techniques to reduce M

- Reduce the number of transactions (N)
  - Reduce size of N as the size of itemset increases
  - Used by DHP and vertical-based mining algorithms

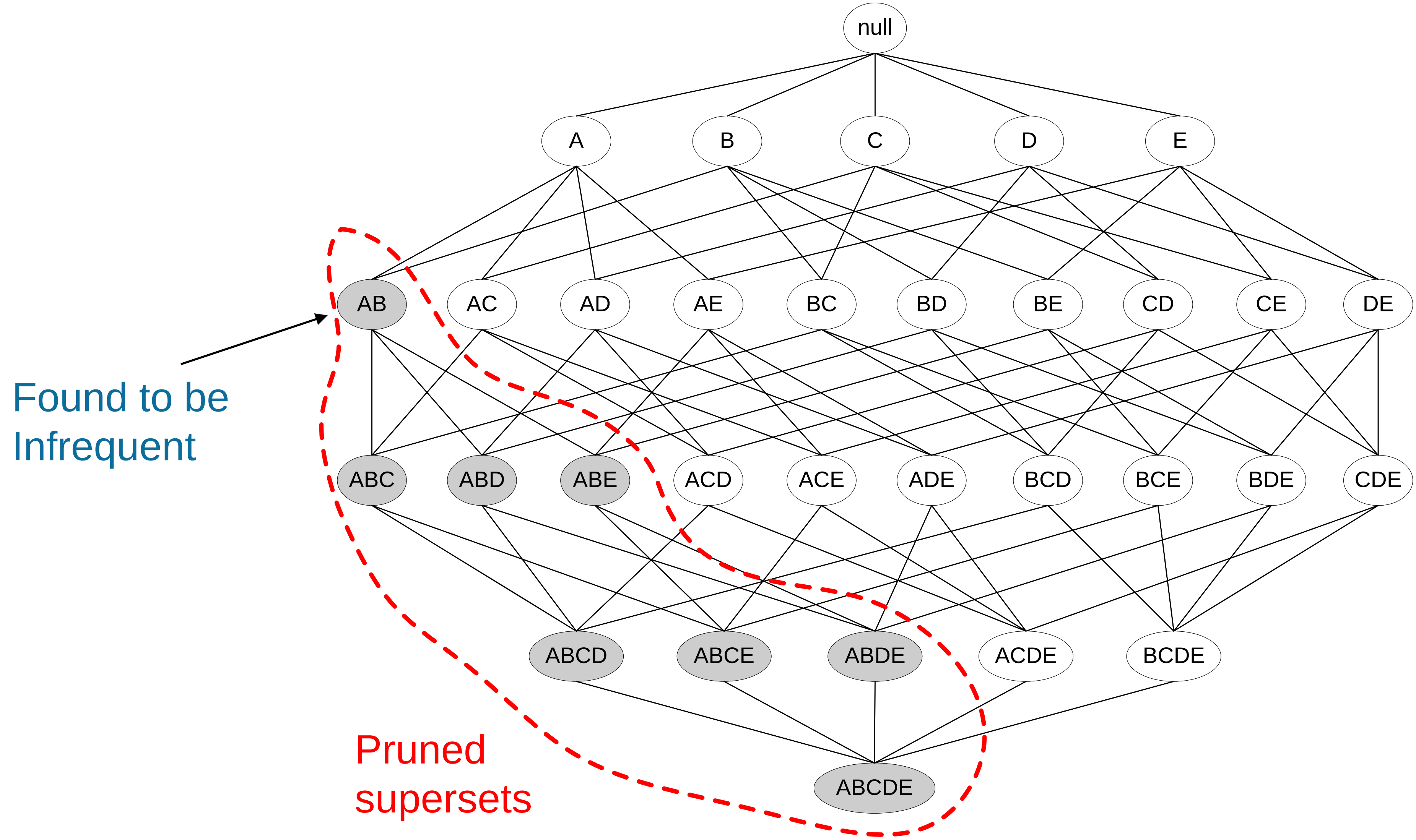
- Reduce the number of comparisons (NM)
  - Use efficient data structures to store the candidates or transactions
  - No need to match every candidate against every transaction

Reducing Number of Candidates

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



Illustrating Apriori Principle



Illustrating Apriori Principle

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

Items (1-itemsets)

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

Minimum Support = 3

If every subset is considered,

$^6C_1 + {}^6C_2 + {}^6C_3$

6 + 15 + 20 = 41

With support-based pruning,

6 + 6 + 4 = 16

Illustrating Apriori Principle

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

Items (1-itemsets)

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

Minimum Support = 3

If every subset is considered,

$$^6C_1 + {}^6C_2 + {}^6C_3$$

$$6 + 15 + 20 = 41$$

With support-based pruning,

$$6 + 6 + 4 = 16$$

Illustrating Apriori Principle

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

Items (1-itemsets)

Itemset
{Bread,Milk}
{Bread, Beer }
{Bread,Diaper}
{Beer, Milk}
{Diaper, Milk}
{Beer,Diaper}

Pairs (2-itemsets)

(No need to generate candidates involving Coke or Eggs)

Minimum Support = 3

If every subset is considered,

$^6C_1 + {}^6C_2 + {}^6C_3$

$6 + 15 + 20 = 41$

With support-based pruning,

$6 + 6 + 4 = 16$

Illustrating Apriori Principle

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

Items (1-itemsets)

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

Pairs (2-itemsets)

(No need to generate candidates involving Coke or Eggs)

Minimum Support = 3

If every subset is considered,

$$^6C_1 + {}^6C_2 + {}^6C_3$$

$$6 + 15 + 20 = 41$$

With support-based pruning,

$$6 + 6 + 4 = 16$$

Illustrating Apriori Principle

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)

Minimum Support = 3

Triplets (3-itemsets)

Itemset
{ Beer, Diaper, Milk}
{ Beer,Bread,Diaper}
{Bread, Diaper, Milk}
{ Beer, Bread, Milk}

If every subset is considered,

$$^6C_1 + {}^6C_2 + {}^6C_3$$

$$6 + 15 + 20 = 41$$

With support-based pruning,

$$6 + 6 + 4 = 16$$

Illustrating Apriori Principle

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)

Minimum Support = 3

Triplets (3-itemsets)

Itemset	Count
{ Beer, Diaper, Milk}	**2**
{ Beer,Bread, Diaper}	**2**
{Bread, Diaper, Milk}	**3**
{Beer, Bread, Milk}	**1**

If every subset is considered,

$^6C_1 + {}^6C_2 + {}^6C_3$

$6 + 15 + 20 = 41$

With support-based pruning,

$6 + 6 + 4 = 16$

$4 + 4 + 1 = 9$

Apriori Algorithm

- F_k: frequent k-itemsets
- L_k: candidate k-itemsets

- Algorithm
  - Let k=1
  - Generate F_1 = {frequent 1-itemsets}
  - Repeat until F_k is empty
    - **Candidate Generation**: Generate L_{k+1} from F_k
    - **Candidate Pruning**: Prune candidate itemsets in L_{k+1} containing subsets of length k that are infrequent
    - **Support Counting**: Count the support of each candidate in L_{k+1} by scanning the DB
    - **Candidate Elimination**: Eliminate candidates in L_{k+1} that are infrequent, leaving only those that are frequent => F_{k+1}

Candidate Generation and Pruning

- In principle, there are many ways to generate candidate itemsets. The following is a list of requirements for an effective candidate generation procedure:

1. It should avoid generating too many unnecessary candidates.
2. It must ensure that the candidate set is complete.
3. It should not generate the same candidate itemset more than once.

Candidate Generation: Brute-force method

Items

Item
Beer
Bread
Cola
Diapers
Milk
Eggs

Candidate Generation

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

Frequent 2-itemset

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

Candidate Pruning

Itemset
{Bread, Diapers, Milk}

Figure 6.6. A brute-force method for generating candidate 3-itemsets.

Candidate Generation: Merge Fk-1 and F1 itemsets

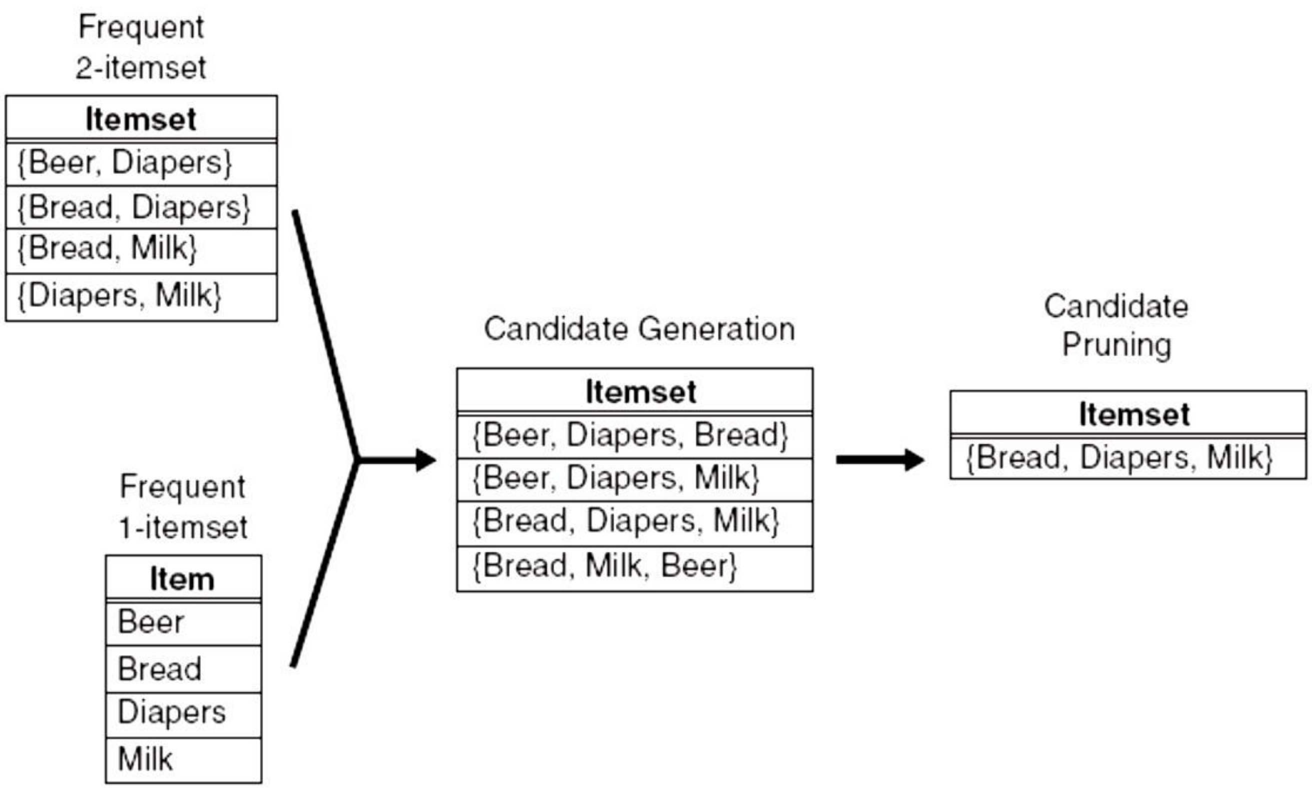


Figure 6.7. Generating and pruning candidate k-itemsets by merging a frequent $(k-1)$-itemset with a frequent item. Note that some of the candidates are unnecessary because their subsets are infrequent.

Candidate Generation: $F_{k-1} \times F_{k-1}$ Method

- Merge two frequent (k-1)-itemsets if their first (k-2) items are identical

- F_3 = {ABC,ABD,ABE,ACD,BCD,BDE,CDE}
  - Merge(**AB**C, **AB**D) = **AB**CD
  - Merge(**AB**C, **AB**E) = **AB**CE
  - Merge(**AB**D, **AB**E) = **AB**DE

  - Do not merge(**A**BD,**A**CD) because they share only prefix of length 1 instead of length 2

Candidate Generation: Fk-1 x Fk-1 Method

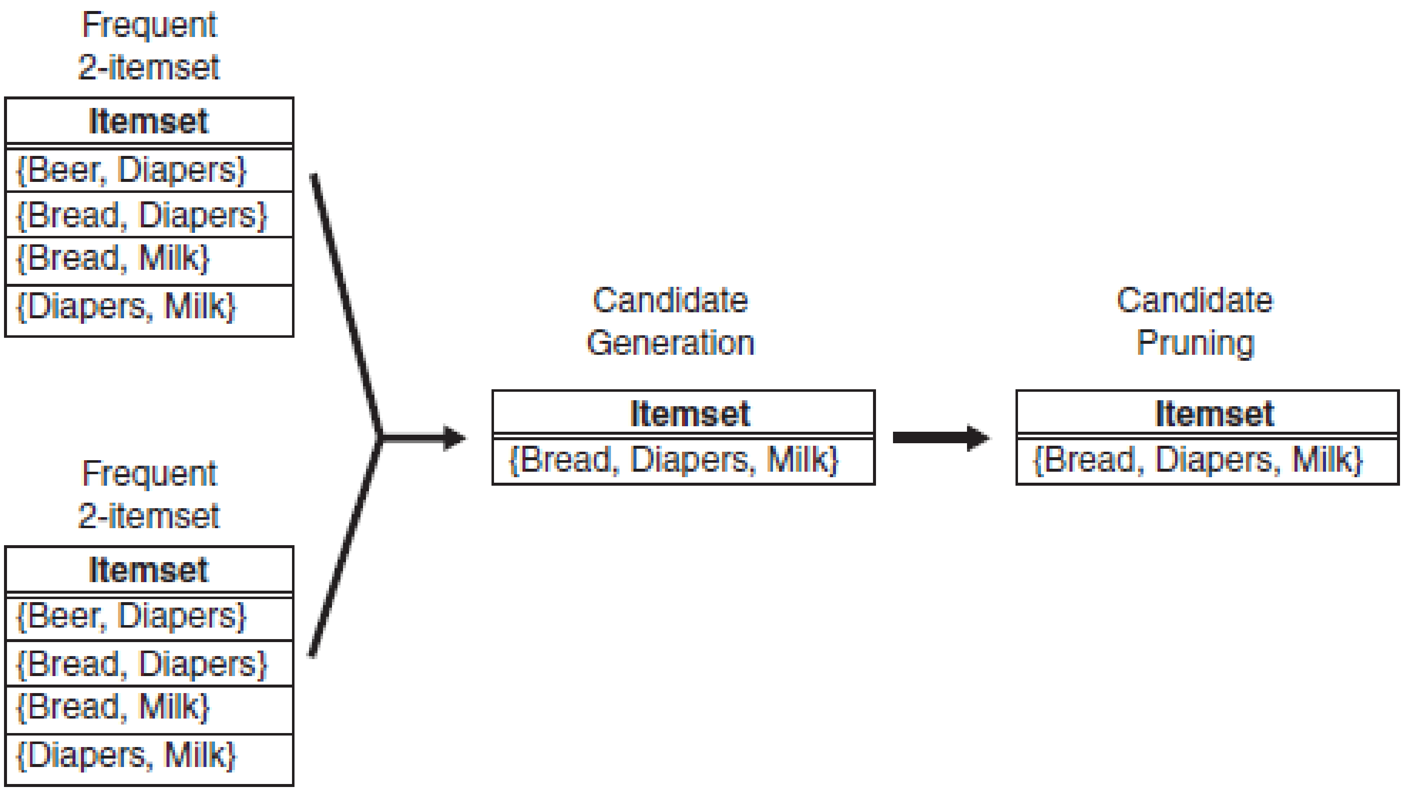


Figure 6.8. Generating and pruning candidate k-itemsets by merging pairs of frequent $(k-1)$-itemsets.

Candidate Pruning

- Let F_3 = {ABC,ABD,ABE,ACD,BCD,BDE,CDE} be the set of frequent 3-itemsets
- L_4 = {ABCD,ABCE,ABDE} is the set of candidate 4-itemsets generated (from previous slide)
- Candidate pruning
  - Prune ABCE because ACE and BCE are infrequent
  - Prune ABDE because ADE is infrequent
- After candidate pruning: L_4 = {ABCD}

Alternate F_{k-1} x F_{k-1} Method

- Merge two frequent (k-1)-itemsets if the last (k-2) items of the first one is identical to the first (k-2) items of the second.

- F_3 = {ABC,ABD,ABE,ACD,BCD,BDE,CDE}
  – Merge(A**<u>BC</u>**, **<u>BC</u>**D) = A**<u>BC</u>**D
  – Merge(A**<u>BD</u>**, **<u>BD</u>**E) = A**<u>BD</u>**E
  – Merge(A**<u>CD</u>**, **<u>CD</u>**E) = A**<u>CD</u>**E
  – Merge(B**<u>CD</u>**, **<u>CD</u>**E) = B**<u>CD</u>**E

Candidate Pruning for Alternate $F_{k-1} \times F_{k-1}$ Method

- Let F_3 = {ABC,ABD,ABE,ACD,BCD,BDE,CDE} be the set of frequent 3-itemsets

- L_4 = {ABCD,ABDE,ACDE,BCDE} is the set of candidate 4-itemsets generated (from previous slide)
- Candidate pruning
  – Prune ABDE because ADE is infrequent
  – Prune ACDE because ACE and ADE are infrequent
  – Prune BCDE because BCE
- After candidate pruning: L_4 = {ABCD}

Support Counting of Candidate Itemsets

- Scan the database of transactions to determine the support of each candidate itemset
  - Must match every candidate itemset against every transaction, which is an expensive operation

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

Itemset
{ Beer, Diaper, Milk}
{ Beer,Bread,Diaper}
{Bread, Diaper, Milk}
{ Beer, Bread, Milk}

Support Counting of Candidate Itemsets

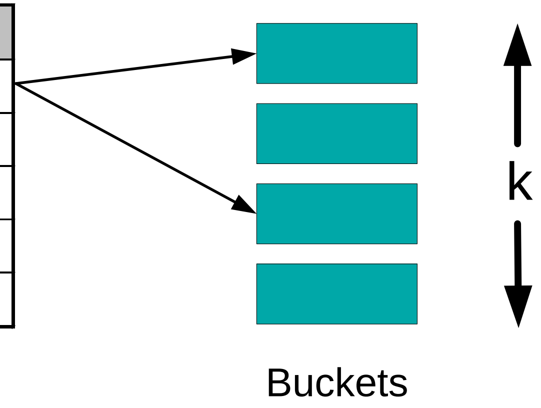
- To reduce number of comparisons, store the candidate itemsets in a hash structure
  - Instead of matching each transaction against every candidate, match it against candidates contained in the hashed buckets

Transactions

N

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

Hash Structure

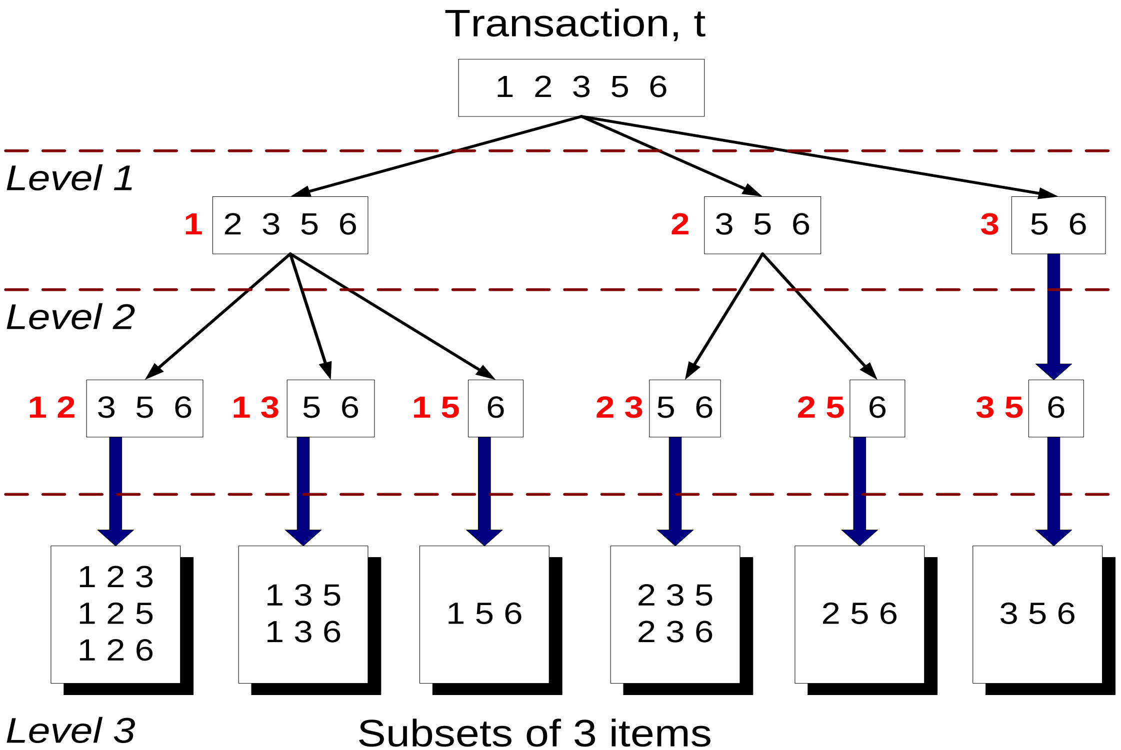


Support Counting: An Example

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}

How many of these itemsets are supported by transaction (1,2,3,5,6)?



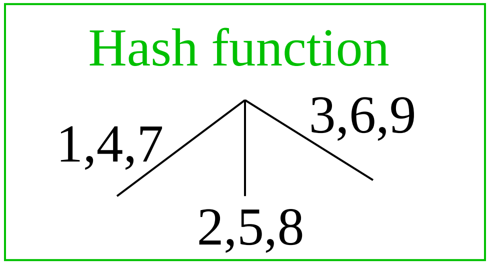
Support Counting Using a Hash Tree

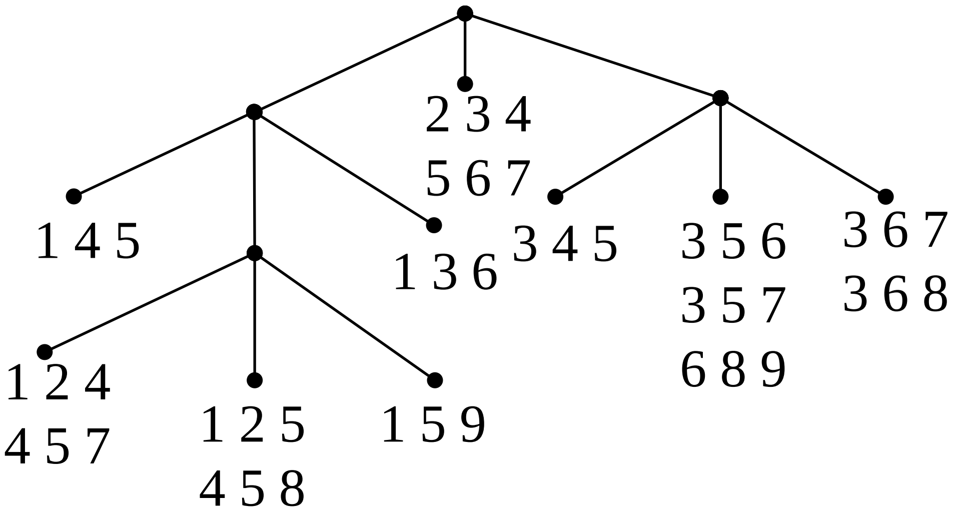
Suppose you have 15 candidate itemsets of length 3:

{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}

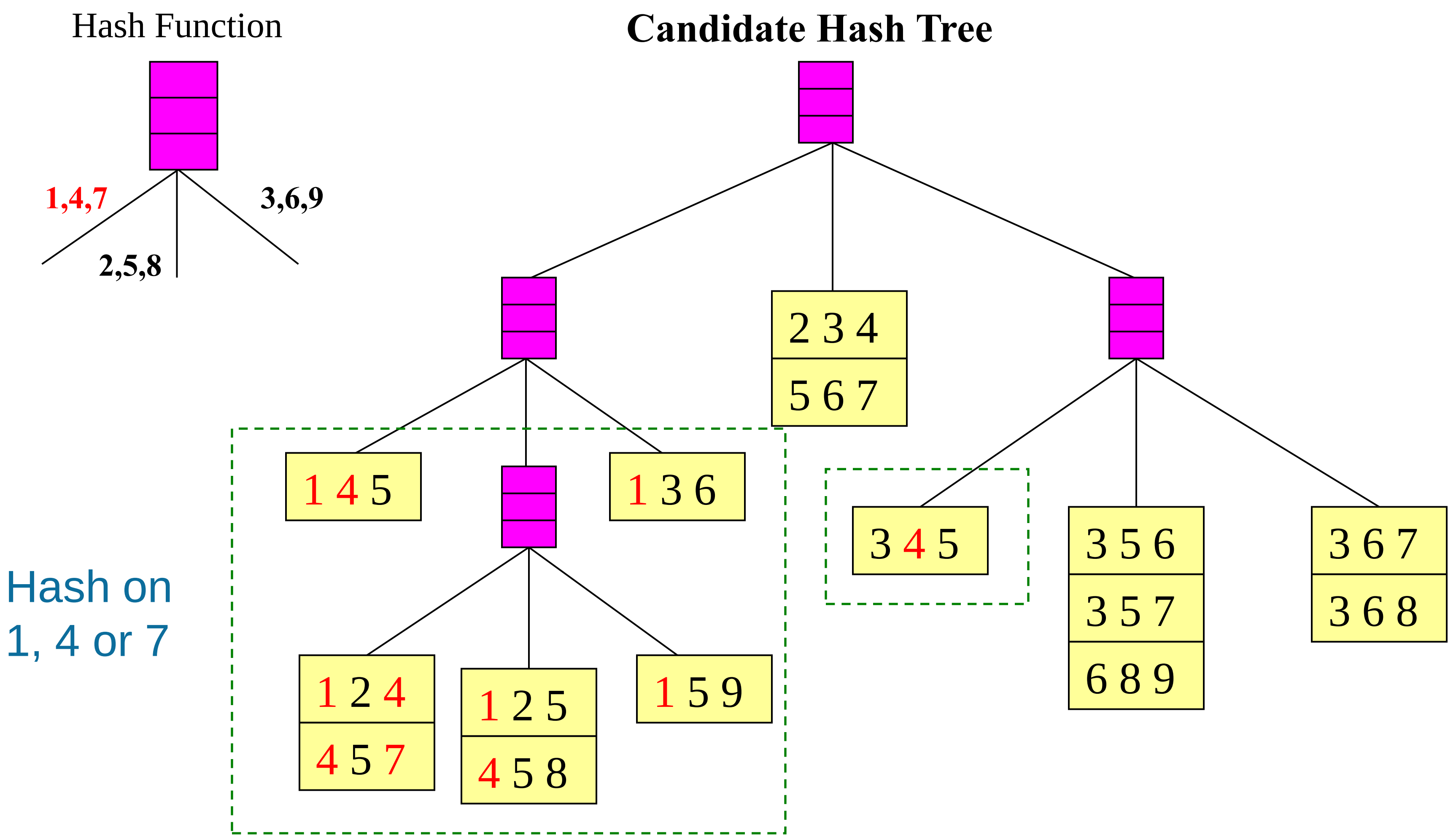
You need:

- **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)**

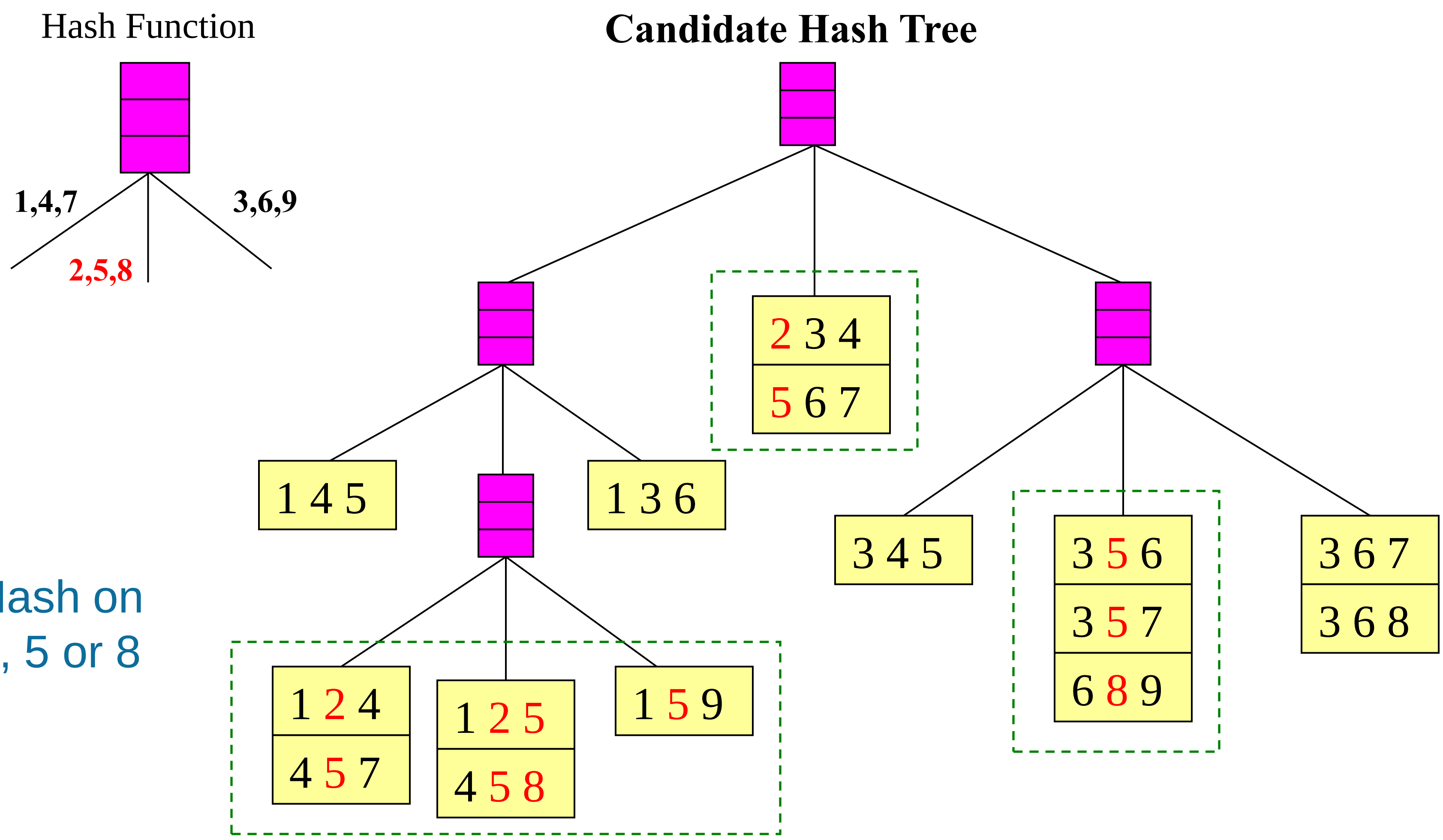




Support Counting Using a Hash Tree

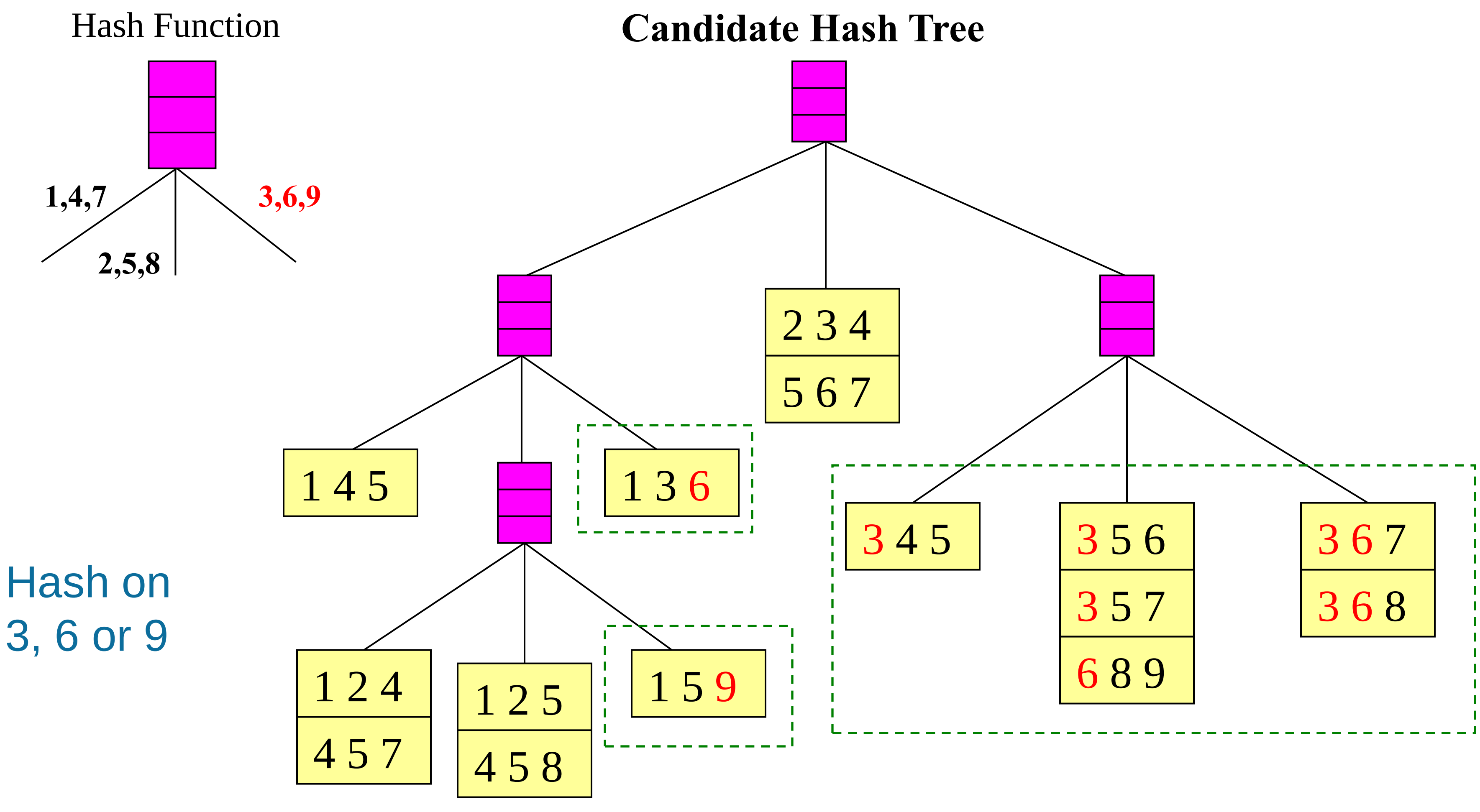


Support Counting Using a Hash Tree

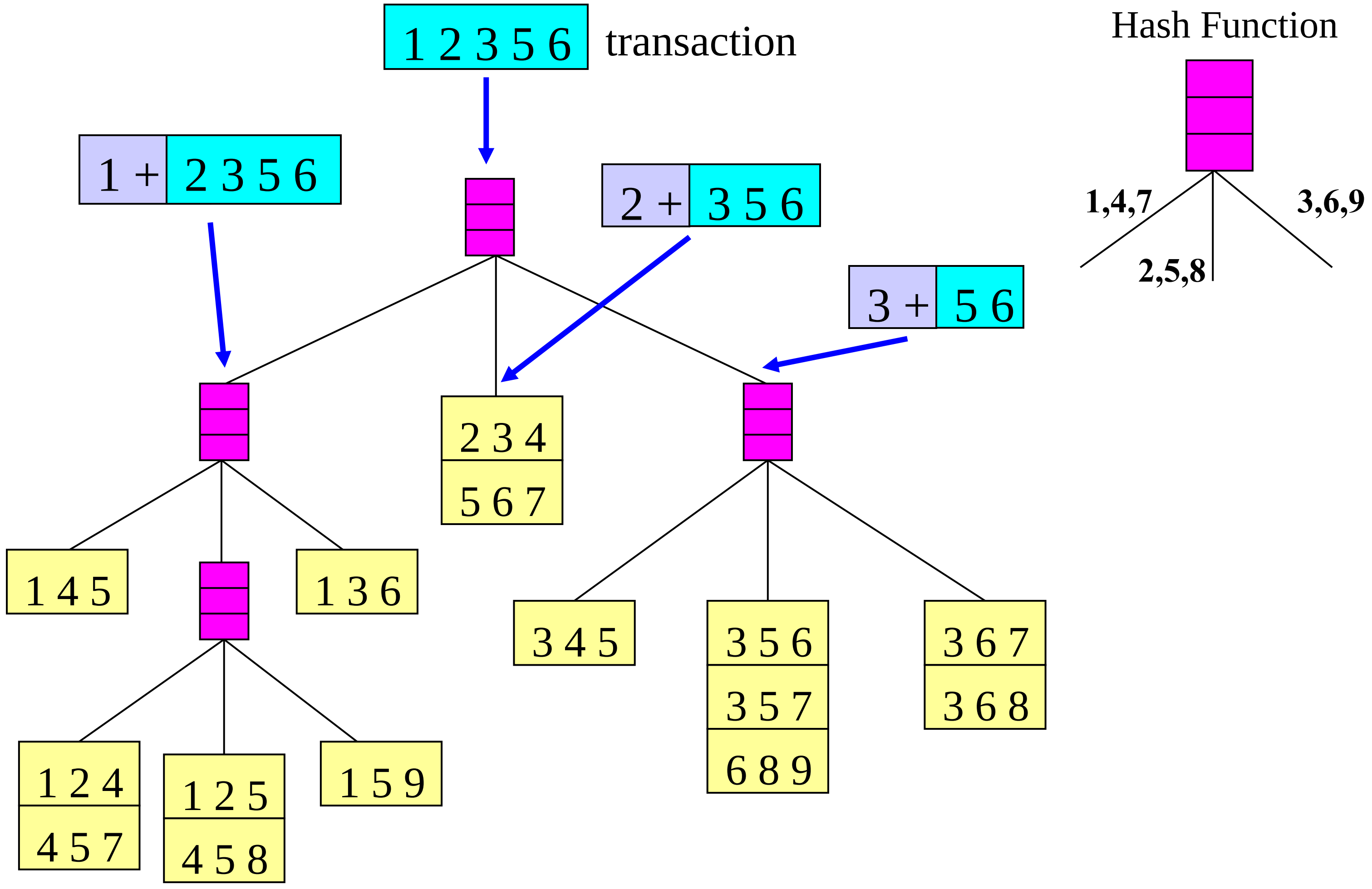


Hash on
2, 5 or 8

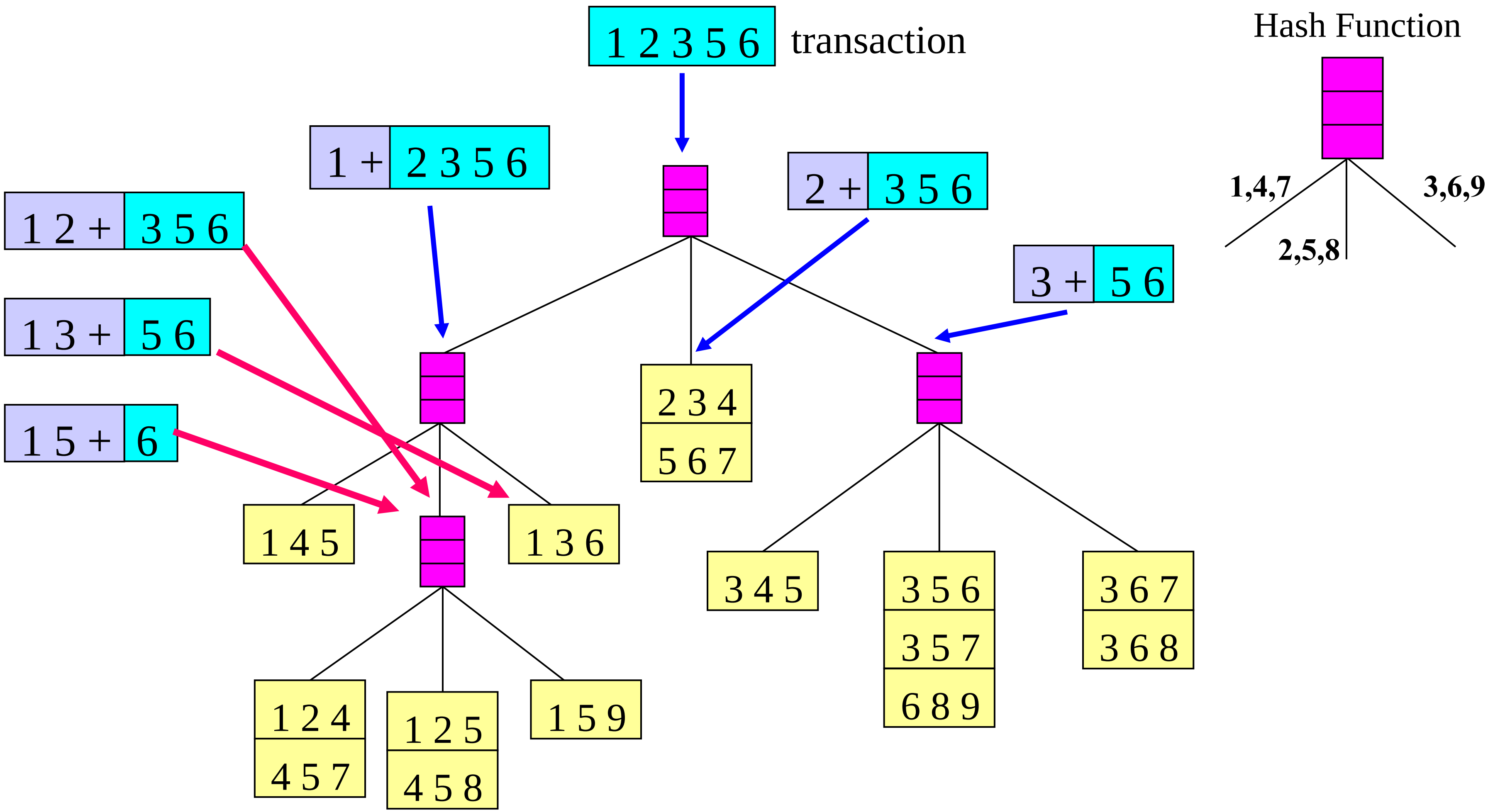
Support Counting Using a Hash Tree



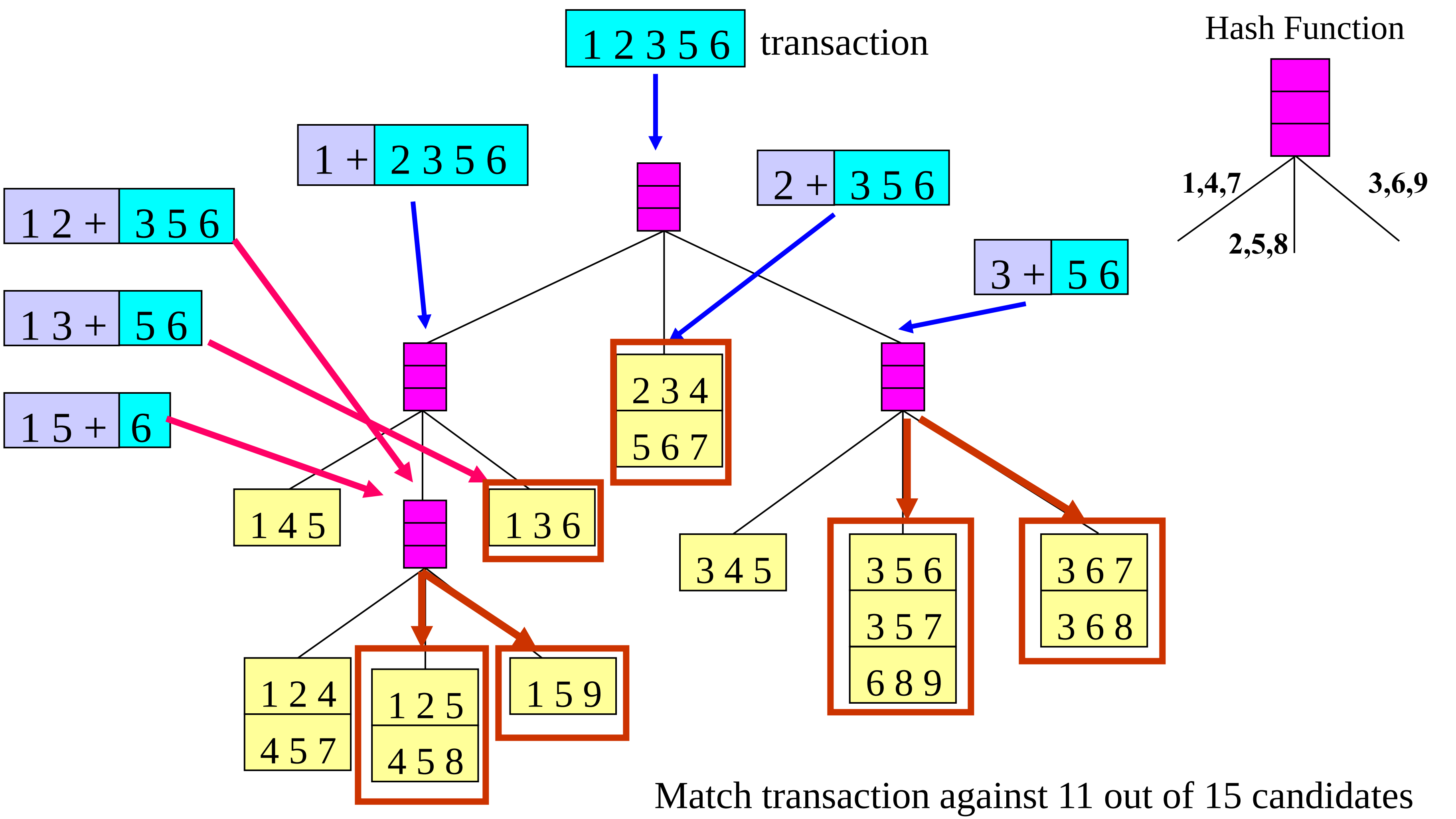
Support Counting Using a Hash Tree



Support Counting Using a Hash Tree



Support Counting Using a Hash Tree



Match transaction against 11 out of 15 candidates

Rule Generation

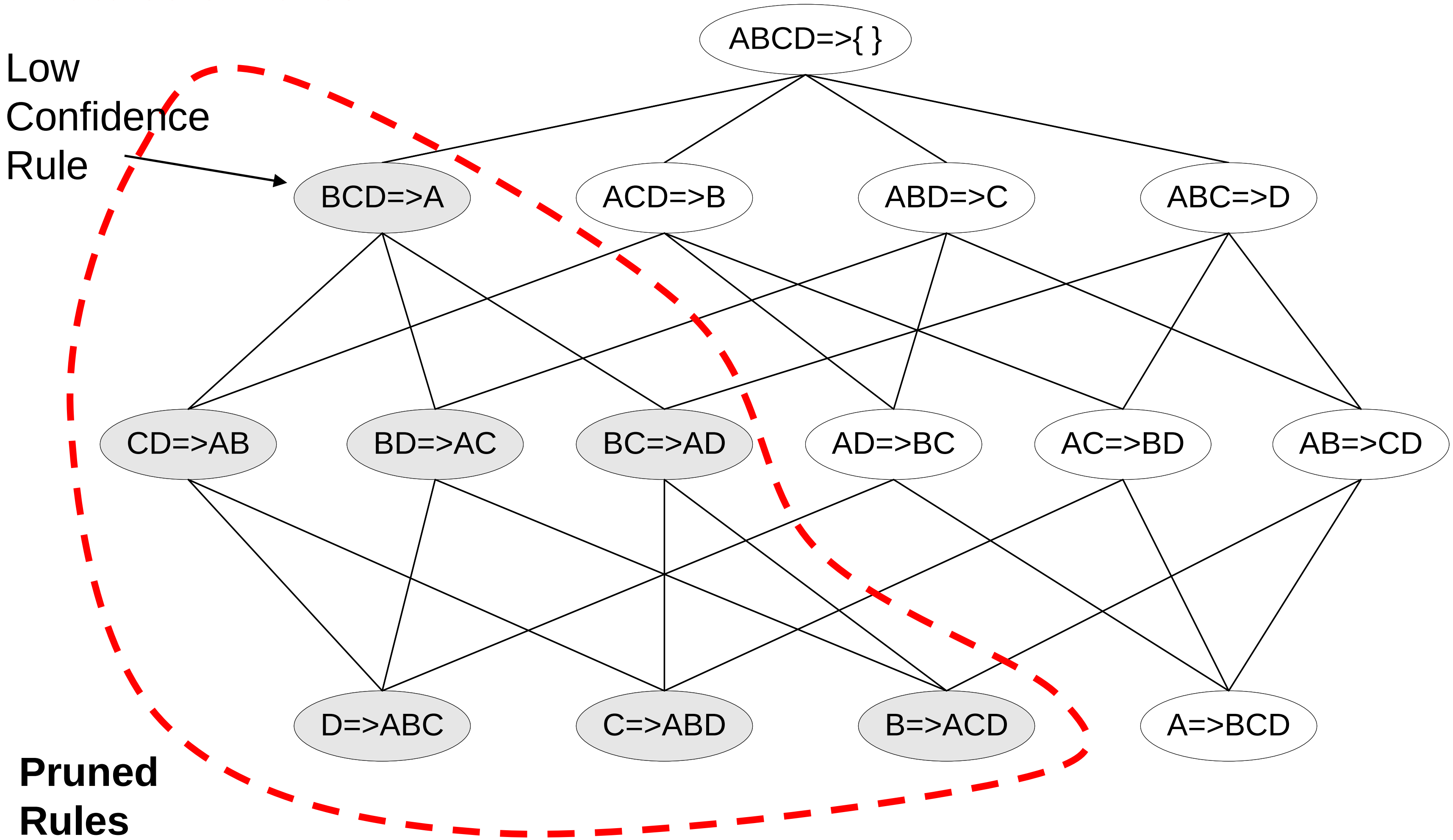
- Given a frequent itemset L, find all non-empty subsets $f \subset L$ such that $f \rightarrow L - f$ satisfies the minimum confidence requirement
  - If {A,B,C,D} is a frequent itemset, candidate rules:

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

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

Rule Generation for Apriori Algorithm

Lattice of rules



Computing Interestingness Measure

- Given $X \rightarrow Y$ or {X,Y}, information needed to compute interestingness can be obtained from a contingency table

Contingency table

	Y	$\overline{Y}$	
X	f_{11}	f_{10}	f_{1+}
$\overline{X}$	f_{01}	f_{00}	f_{o+}
	f_{+1}	f_{+0}	N

f_{11}: support of X and Y
f_{10}: support of X and $\overline{Y}$
f_{01}: support of $\overline{X}$ and Y
f_{00}: support of $\overline{X}$ and $\overline{Y}$

Used to define various measures

- support, confidence, Gini, entropy, etc.

Drawback of Confidence

Customers	Tea	Coffee	…
C1	0	1	…
C2	1	0	…
C3	1	1	…
C4	1	0	…
…			

	Coffee	$\overline{\text{Coffee}}$	
Tea	15	5	20
$\overline{\text{Tea}}$	75	5	80
	90	10	100

Association Rule: Tea $\rightarrow$ Coffee

Confidence $\cong$ P(Coffee|Tea) = 15/20 = 0.75

Confidence > 50%, meaning people who drink tea are more likely to drink coffee than not drink coffee

So rule seems reasonable

Drawback of Confidence

	Coffee	$\overline{\text{Coffee}}$	
Tea	15	5	20
$\overline{\text{Tea}}$	75	5	80
	90	10	100

Association Rule: Tea $\rightarrow$ Coffee

Confidence= P(Coffee|Tea) = 15/20 = 0.75

but P(Coffee) = 0.9, which means knowing that a person drinks tea reduces the probability that the person drinks coffee!

$\Rightarrow$ Note that $P(\text{Coffee}|\overline{\text{Tea}}) = 75/80 = 0.9375$

Measure for Association Rules

- So, what kind of rules do we really want?
  - Confidence($X \rightarrow Y$) should be sufficiently high
    - To ensure that people who buy X will more likely buy Y than not buy Y
  - Confidence($X \rightarrow Y$) > support(Y)
    - Otherwise, rule will be misleading because having item X actually reduces the chance of having item Y in the same transaction
    - Is there any measure that capture this constraint?
      - Answer: Yes. There are many of them.

Statistical Independence

- The criterion

  $$\text{confidence}(X \rightarrow Y) = \text{support}(Y)$$

  is equivalent to:
  - $P(Y|X) = P(Y)$
  - $P(X,Y) = P(X) \times P(Y)$

  If $P(X,Y) > P(X) \times P(Y)$: X & Y are positively correlated

  If $P(X,Y) < P(X) \times P(Y)$: X & Y are negatively correlated

Measures that take into account statistical dependence

$$\left.\begin{aligned} Lift &= \frac{P(Y\,|\,X)}{P(Y)} \\ Interest &= \frac{P(X,Y)}{P(X)P(Y)} \end{aligned}\right\} \text{lift is used for rules while interest is used for itemsets}$$

$$PS = P(X,Y) - P(X)P(Y)$$

$$\phi - coefficient = \frac{P(X,Y) - P(X)P(Y)}{\sqrt{P(X)[1-P(X)]P(Y)[1-P(Y)]}}$$

Example: Lift/Interest

	Coffee	$\overline{\text{Coffee}}$	
Tea	15	5	20
$\overline{\text{Tea}}$	75	5	80
	90	10	100

Association Rule: Tea $\rightarrow$ Coffee

Confidence= P(Coffee|Tea) = 0.75

but P(Coffee) = 0.9

$\Rightarrow$ Lift = 0.75/0.9= 0.8333 (< 1, therefore is negatively associated)

So, is it enough to use confidence/lift for pruning?

Lift or Interest

	Y	$\overline{Y}$	
X	10	0	10
$\overline{X}$	0	90	90
	10	90	100

	Y	$\overline{Y}$	
X	90	0	90
$\overline{X}$	0	10	10
	90	10	100

$$Lift = \frac{0.1}{(0.1)(0.1)} = 10$$

$$Lift = \frac{0.9}{(0.9)(0.9)} = 1.11$$

Statistical independence:

If P(X,Y)=P(X)P(Y) => Lift = 1

Example 1 (Support and Confidence)

TID	date	items_bought
100	10/10/99	{F,A,D,B}
200	15/10/99	{D,A,C,E,B}
300	19/10/99	{C,A,B,E}
400	20/10/99	{B,A,D}

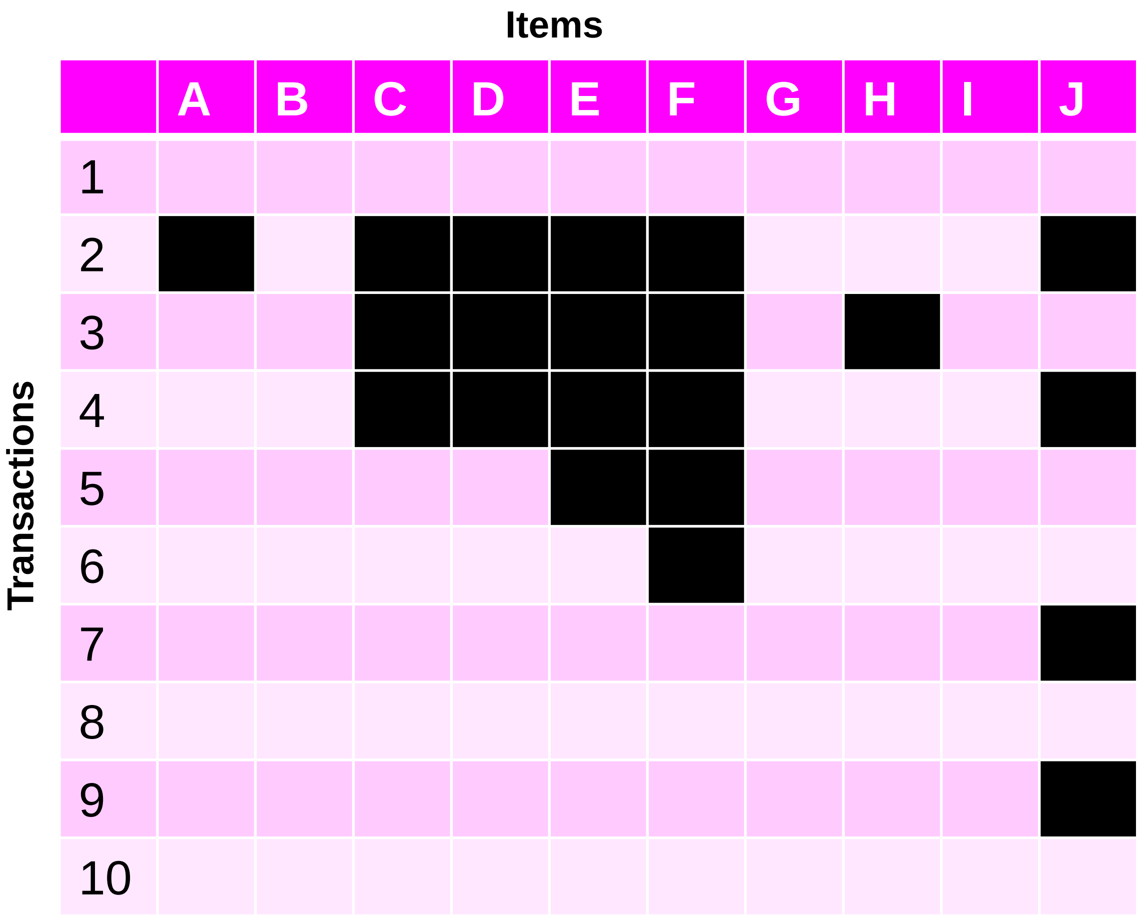
What is the support and confidence of the rule:
$\{B,D\} \rightarrow \{A\}$

Example 2 (Support and Confidence)

- What is the support and confidence of the rule:
- $A \rightarrow C$
- $C \rightarrow A$

Transaction ID	Items Bought
2000	A,B,C
1000	A,C
4000	A,D
5000	B,E,F

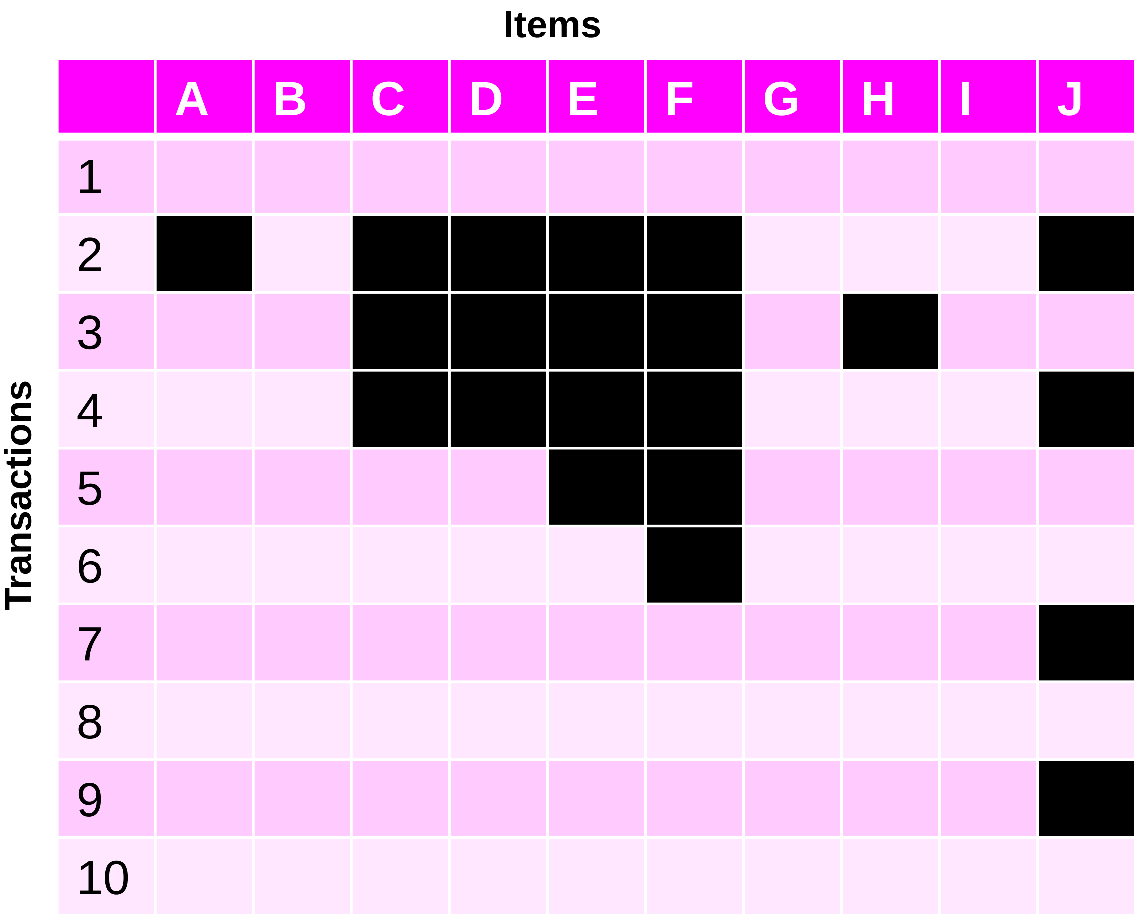
An illustrative example



Support threshold (by count) : 5

Frequent itemsets: ?

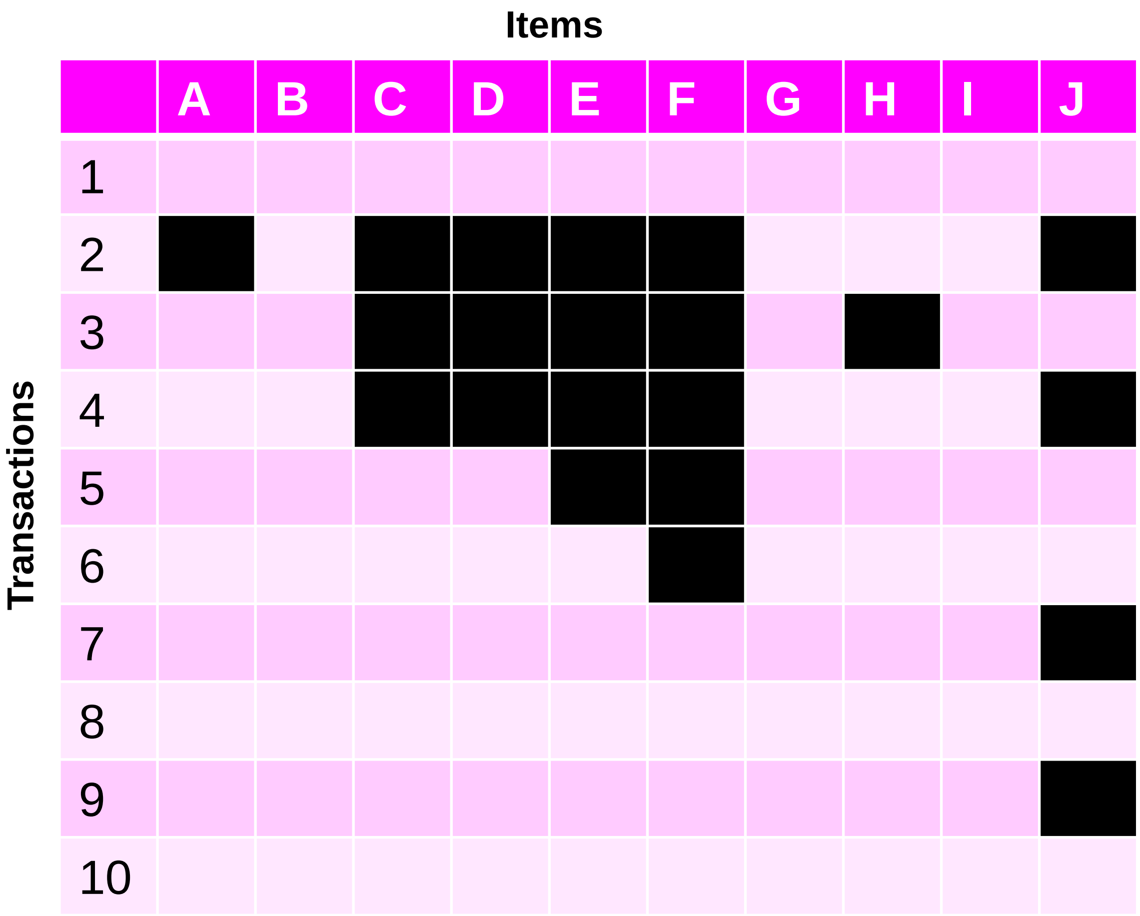
An illustrative example



Support threshold (by count) : 5

Frequent itemsets: {F}

An illustrative example



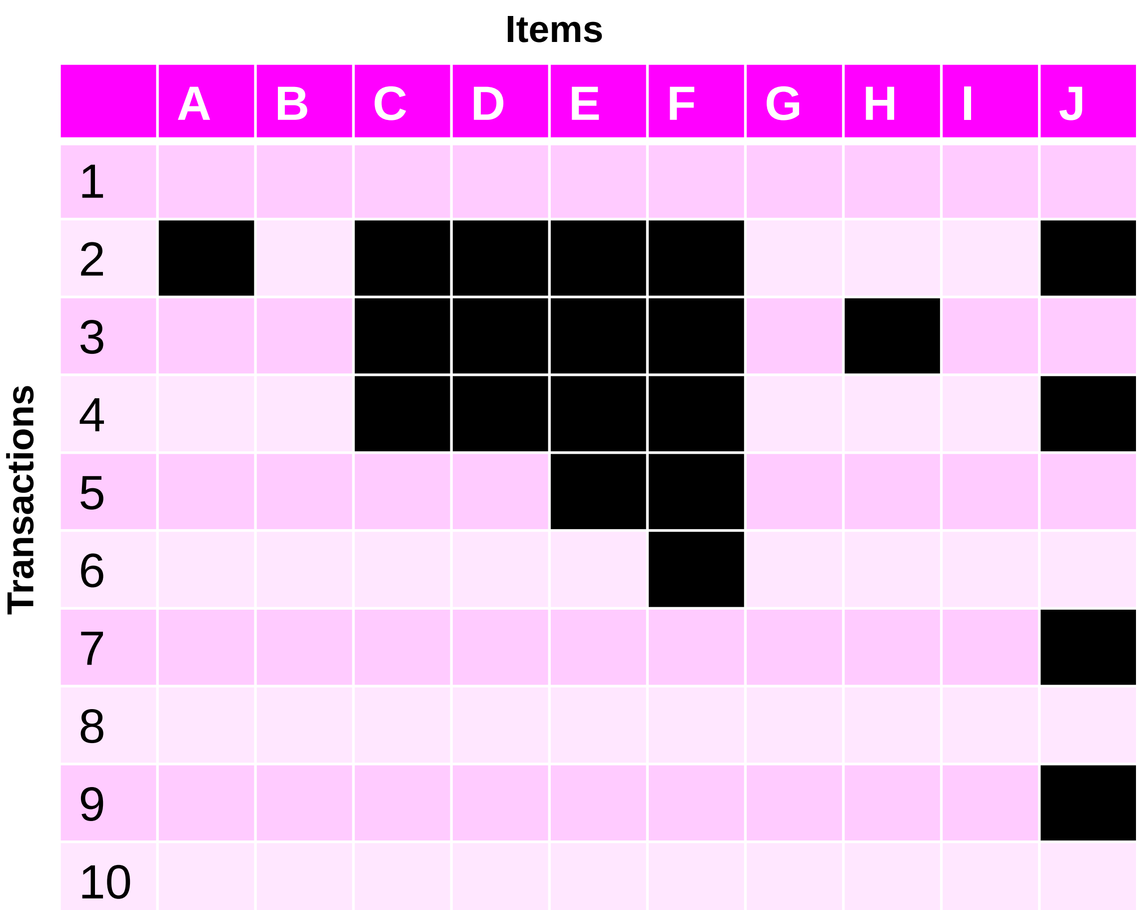
Support threshold (by count) : 5

Frequent itemsets: {F}

Support threshold (by count): 4

Frequent itemsets: ?

An illustrative example



Support threshold (by count) : 5
Frequent itemsets: {F}

Support threshold (by count): 4
Frequent itemsets: {E}, {F}, {E,F}, {J}

An illustrative example

Items

Transactions

	A	B	C	D	E	F	G	H	I	J
1										
2	■		■	■	■	■				■
3			■	■	■	■		■		
4			■	■	■	■				■
5					■	■				
6						■				
7										■
8										
9										■
10										

Support threshold (by count) : 5

Frequent itemsets: {F}

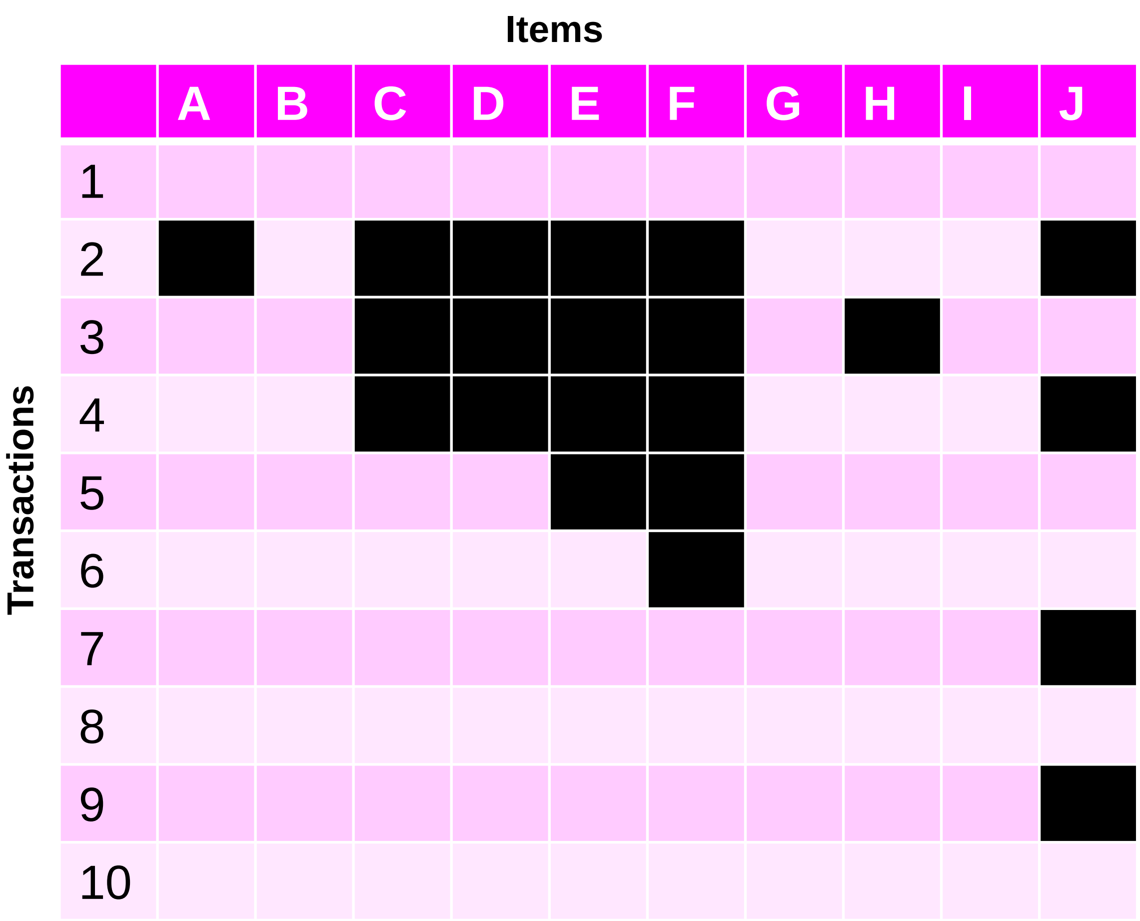
Support threshold (by count): 4

Frequent itemsets: {E}, {F}, {E,F}, {J}

Support threshold (by count): 3

Frequent itemsets: ?

An illustrative example



Support threshold (by count) : 5
Frequent itemsets: {F}

Support threshold (by count): 4
Frequent itemsets: {E}, {F}, {E,F}, {J}

Support threshold (by count): 3
Frequent itemsets:
All subsets of {C,D,E,F} + {J}

Example 1 (Candidate Itemset Generation)

$Sup_{min} = 2$

Database TDB

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

C_1

$\xrightarrow{1^{st}\text{ scan}}$

Example 1 (Candidate Itemset Generation)

$Sup_{min} = 2$

Database TDB

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

1st scan

C_1

Itemset	sup
{A}	2
{B}	3
{C}	3
{D}	1
{E}	3

L_1

Itemset	sup
{A}	2
{B}	3
{C}	3
{E}	3

C_2

Itemset
{A, B}
{A, C}
{A, E}
{B, C}
{B, E}
{C, E}

2nd scan

C_2

Itemset	sup
{A, B}	1
{A, C}	2
{A, E}	1
{B, C}	2
{B, E}	3
{C, E}	2

L_2

Itemset	sup
{A, C}	2
{B, C}	2
{B, E}	3
{C, E}	2

C_3

Itemset
{B, C, E}

3rd scan

L_3

Itemset	sup
{B, C, E}	2

Example 2 (Candidate Itemset Generation)

Database D

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

Sc

min_sup=2=50%

Example 2 (Candidate Itemset Generation)

Database D

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

min_sup=2=50%

Scan D

C_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{4}	1
{5}	3

L_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{5}	3

C_2

itemset
{1 2}
{1 3}
{1 5}
{2 3}
{2 5}
{3 5}

Scan D

C_2

itemset	sup
{1 2}	1
{1 3}	2
{1 5}	1
{2 3}	2
{2 5}	3
{3 5}	2

L_2

itemset	sup
{1 3}	2
{2 3}	2
{2 5}	3
{3 5}	2

C_3

itemset
{2 3 5}

Scan D

L_3

itemset	sup
{2 3 5}	2

Example 1 (Apriori)

- Consider the following transactions for association rules analysis:
- Use minimum support(min_sup) = 2 (2/9 = 22%) and
- Minimum confidence = 70%

TID	*List of item_IDs*
T100	I1, I2, I5
T200	I2, I4
T300	I2, I3
T400	I1, I2, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I1, I2, I3, I5
T900	I1, I2, I3

Step1: Frequent Itemset Generation:

Scan D for count of each candidate

C_1

Itemset	Sup. count
{I1}	6
{I2}	7
{I3}	6
{I4}	2
{I5}	2

Compare candidate support count with minimum support count

L_1

Itemset	Sup. count
{I1}	6
{I2}	7
{I3}	6
{I4}	2
{I5}	2

Generate C_2 candidates from L_1

C_2

Itemset
{I1, I2}
{I1, I3}
{I1, I4}
{I1, I5}
{I2, I3}
{I2, I4}
{I2, I5}
{I3, I4}
{I3, I5}
{I4, I5}

Scan D for count of each candidate

C_2

Itemset	Sup. count
{I1, I2}	4
{I1, I3}	4
{I1, I4}	1
{I1, I5}	2
{I2, I3}	4
{I2, I4}	2
{I2, I5}	2
{I3, I4}	0
{I3, I5}	1
{I4, I5}	0

Compare candidate support count with minimum support count

L_2

Itemset	Sup. count
{I1, I2}	4
{I1, I3}	4
{I1, I5}	2
{I2, I3}	4
{I2, I4}	2
{I2, I5}	2

Generate C_3 candidates from L_2

C_3

Itemset
{I1, I2, I3}
{I1, I2, I5}

Scan D for count of each candidate

C_3

Itemset	Sup. count
{I1, I2, I3}	2
{I1, I2, I5}	2

Compare candidate support count with minimum support count

L_3

Itemset	Sup. count
{I1, I2, I3}	2
{I1, I2, I5}	2

Generation of the candidate itemsets and frequent itemsets, where the minimum support count is 2.

Step2: Generating association rules:

- The data contain frequent itemset *X ={I1,I2,I5}*. What are the association rules that can be generated from *X?*

The nonempty subsets of *X are* {I1, I2}, {I1, I5}, {I2, I5}, {I1}, {I2}, and {I5}. *The resulting association rules are* as shown below, each listed with its confidence:

$$
\begin{array}{ll}
\{I1,I2\} \Rightarrow I5, & \mathit{confidence} = 2/4 = 50\% \\
\{I1,I5\} \Rightarrow I2, & \mathit{confidence} = 2/2 = 100\% \\
\{I2,I5\} \Rightarrow I1, & \mathit{confidence} = 2/2 = 100\% \\
I1 \Rightarrow \{I2,I5\}, & \mathit{confidence} = 2/6 = 33\% \\
I2 \Rightarrow \{I1,I5\}, & \mathit{confidence} = 2/7 = 29\% \\
I5 \Rightarrow \{I1,I2\}, & \mathit{confidence} = 2/2 = 100\%
\end{array}
$$

Here, minimum confidence threshold is 70%, so only the second, third, and last rules are output, because these are the only ones generated that are strong.

Example (Contingency Tables)

- The original association rule mining formulation uses the support and confidence measures to prune uninteresting rules.

a) Draw a contingency table for each of the following rules using the transactions
Rules: $\{b\} \rightarrow \{c\}$, $\{a\} \rightarrow \{d\}$, $\{b\} \rightarrow \{d\}$, $\{e\} \rightarrow \{c\}$, $\{c\} \rightarrow \{a\}$.

b) Use the contingency tables in part (a) to compute and rank the rules in decreasing order according to

- Support
- Confidence
- Lift

Transaction ID	Items Bought
1	$\{a, b, d, e\}$
2	$\{b, c, d\}$
3	$\{a, b, d, e\}$
4	$\{a, c, d, e\}$
5	$\{b, c, d, e\}$
6	$\{b, d, e\}$
7	$\{c, d\}$
8	$\{a, b, c\}$
9	$\{a, d, e\}$
10	$\{b, d\}$

- Contingency tables

	c	$\overline{c}$
b	3	4
$\overline{b}$	2	1

	d	$\overline{d}$
a	4	1
$\overline{a}$	5	0

	d	$\overline{d}$
b	6	1
$\overline{b}$	3	0

	c	$\overline{c}$
e	2	4
$\overline{e}$	3	1

	a	$\overline{a}$
c	2	3
$\overline{c}$	3	2

- Support

Rules	Support	Rank
$b \longrightarrow c$	0.3	3
$a \longrightarrow d$	0.4	2
$b \longrightarrow d$	0.6	1
$e \longrightarrow c$	0.2	4
$c \longrightarrow a$	0.2	4

Confidence

Rules	Confidence	Rank
$b \longrightarrow c$	3/7	3
$a \longrightarrow d$	4/5	2
$b \longrightarrow d$	6/7	1
$e \longrightarrow c$	2/6	5
$c \longrightarrow a$	2/5	4

Lift

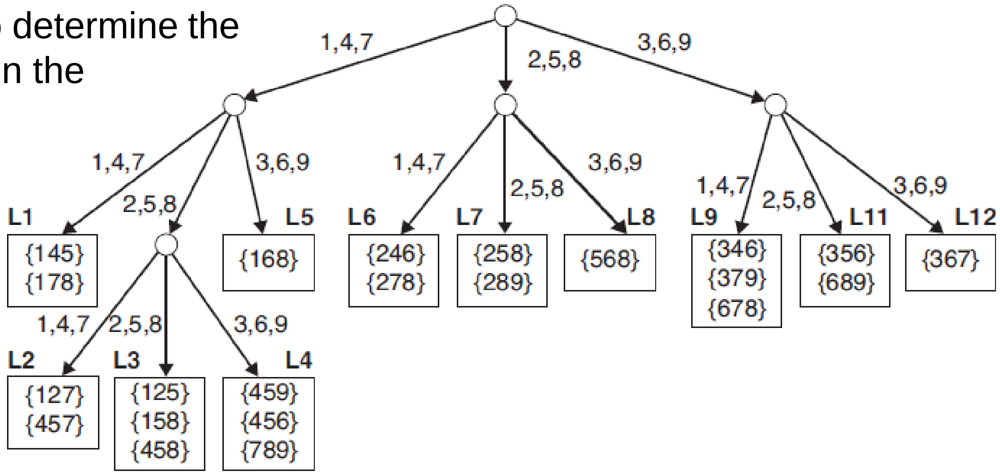
Rules	Interest	Rank
$b \longrightarrow c$	0.214	3
$a \longrightarrow d$	0.72	2
$b \longrightarrow d$	0.771	1
$e \longrightarrow c$	0.167	5
$c \longrightarrow a$	0.2	4

Example (Hash Tree)

- The *Apriori* algorithm uses a hash tree data structure to efficiently count the support of candidate itemsets. Consider the hash tree for candidate 3- itemsets shown in figüre below.

a) Given a transaction that contains items *{*1*,* 3*,* 4*,* 5*,* 8*}*, which of the hash tree leaf nodes will be visited when finding the candidates of the transaction?

b) Use the visited leaf nodes in part (a) to determine the candidate itemsets that are contained in the transaction {1, 3, 4, 5, 8}.



Example (Hash Tree)

a) Given a transaction that contains items {1, 3, 4, 5, 8}, which of the hash tree leaf nodes will be visited when finding the candidates of the transaction?

- The leaf nodes visited are L1, L3, L5, L9, and L11.

b) Use the visited leaf nodes in part (a) to determine the candidate itemsets that are contained in the transaction {1, 3, 4, 5, 8}.

- The candidates contained in the transaction are {1, 4, 5}, {1, 5, 8}, and {4, 5, 8}.