# CS 484
# Data Mining

Association Rule Mining 2

# Review: 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
  - This is known as the anti-monotone property of support

# Candidate Generation

- Three basic approaches:
  - Brute-force method
  - $F_{k-1} \times F_1$ method
  - $F_{k-1} \times F_{k-1}$ method
- The next three slides demonstrate how each method generates candidate 3-itemsets

# Brute-Force Method

| *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** |

Min support count = 3
(minsup = 60%)

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} |

Candidate Pruning

| Itemset |
|---|
| {Bread, Diapers, Milk} |

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

# $F_{k-1}$ x $F_1$ method

| *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** |

Min support count = 3
(minsup = 60%)

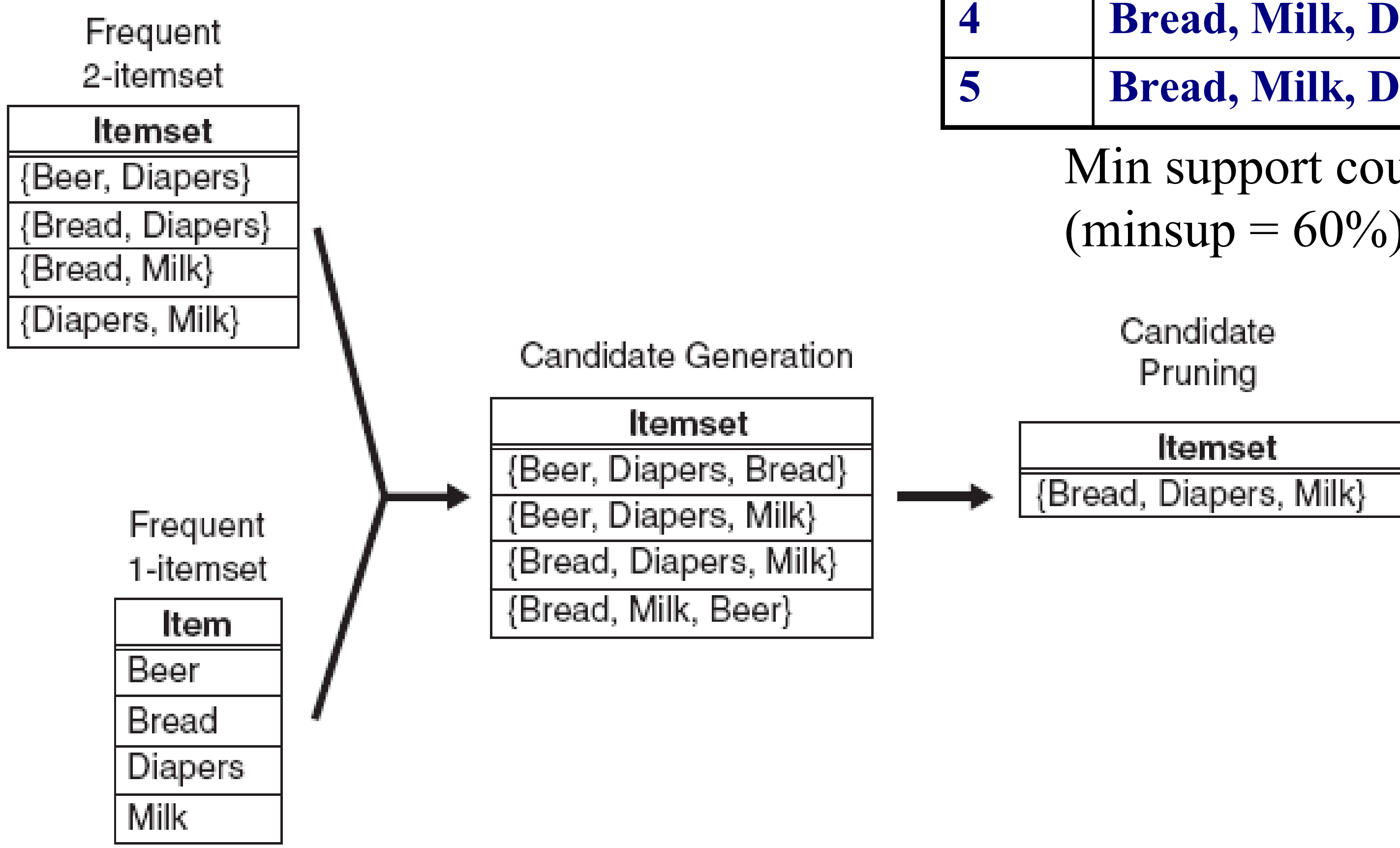



**Figure 6.7.** Generating and pruning candidate $k$-itemsets by merging a frequent $(k-1)$-itemset with a frequent item. Note that some of the candidates are unnecessary because their subsets are infrequent.

# $F_{k-1}$ x $F_{k-1}$ method

| *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** |

Min support count = 3
(minsup = 60%)

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} |

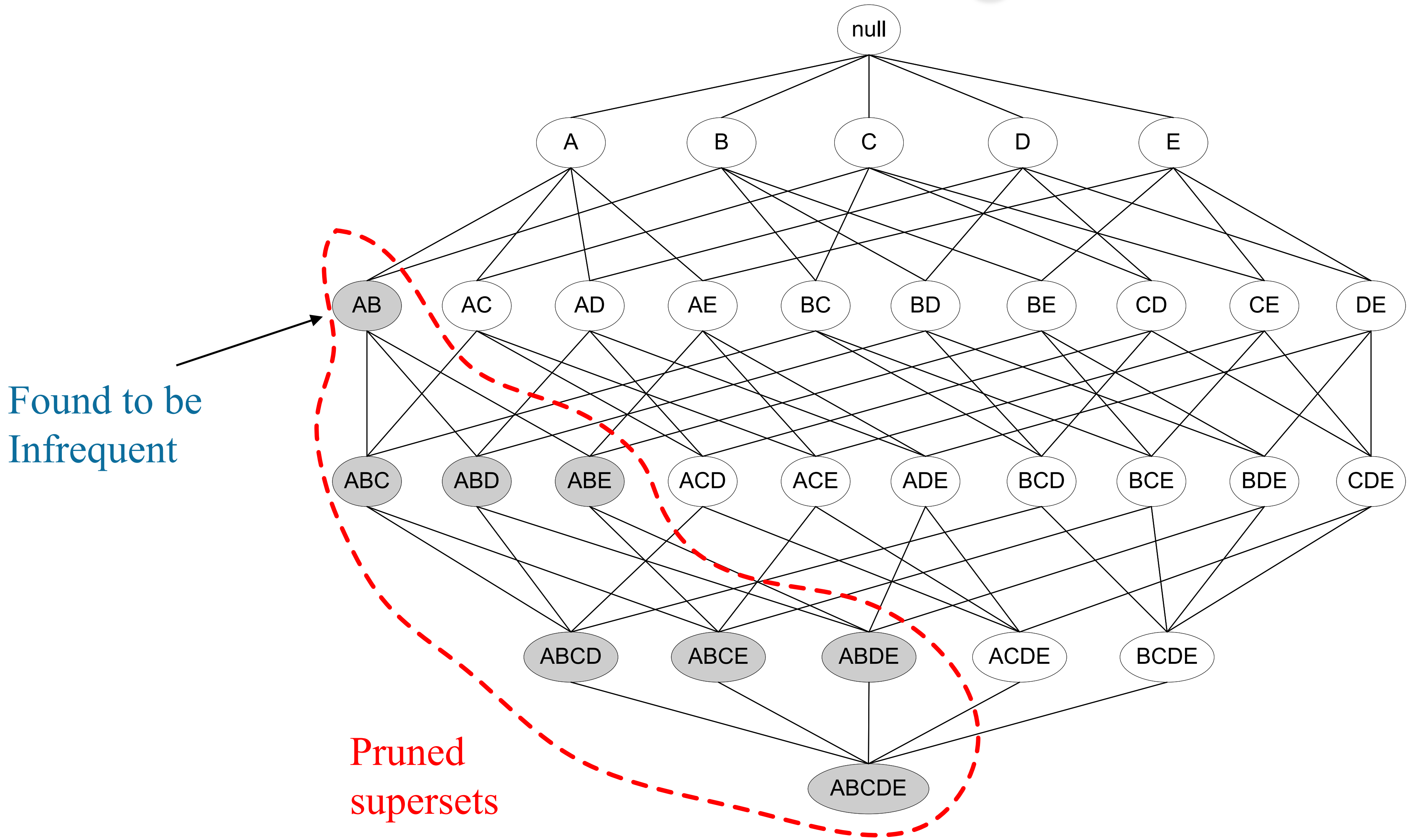
Candidate
Pruning

| Itemset |
|---|
| {Bread, Diapers, Milk} |

Only merge a pair of frequent (k-1)-itemsets if their first k-2 items are identical!

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

# Candidate Pruning



# 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

- How to efficiently generate rules from frequent itemsets?
  - In general, confidence does not have an anti-monotone property

    $c(ABC \rightarrow D)$ can be larger or smaller than $c(AB \rightarrow D)$

  - But confidence of rules generated from the same itemset has an anti-monotone property
  - e.g., $L = \{A,B,C,D\}$:

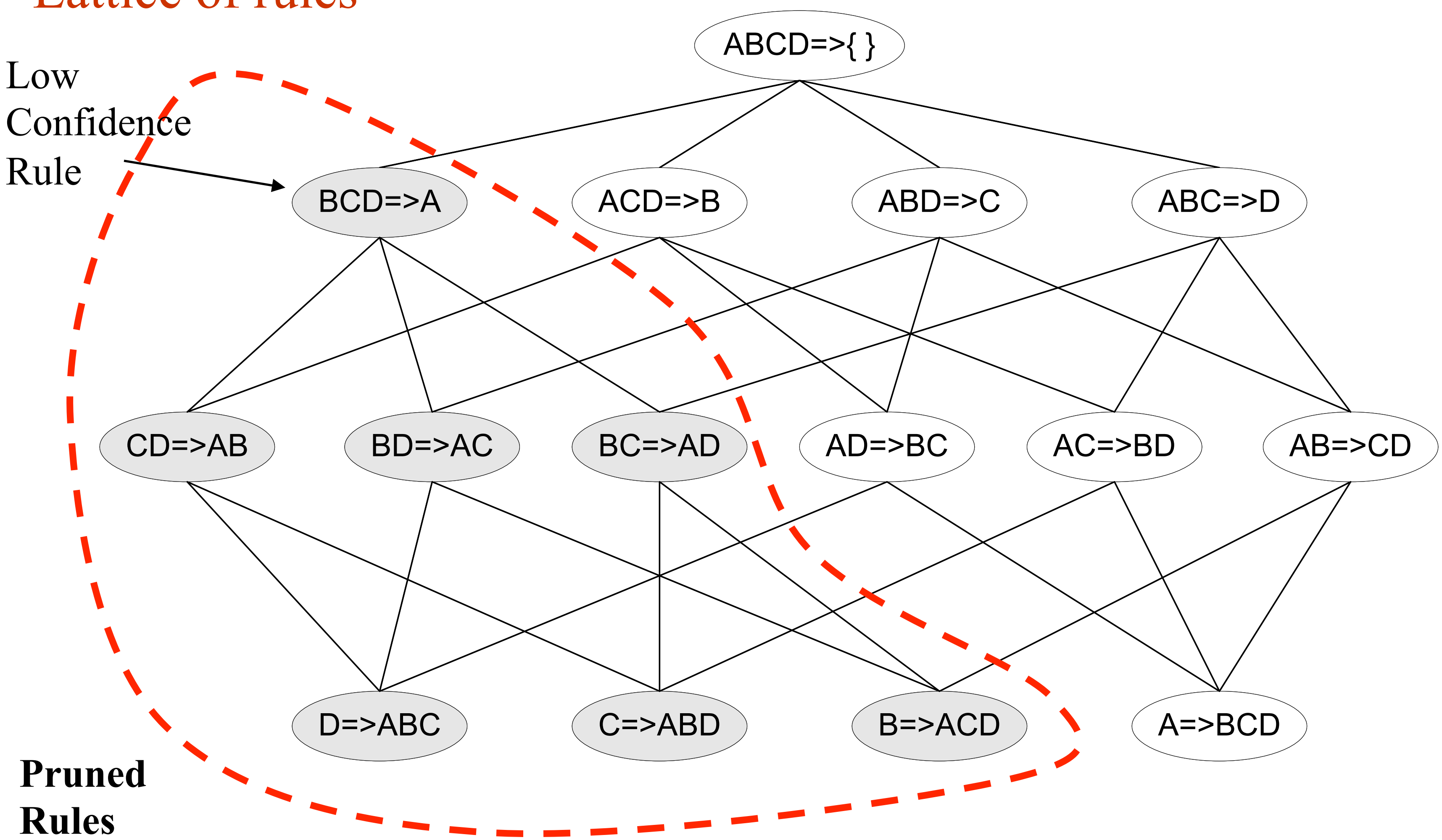
  - $c(ABC \rightarrow D) \geq c(AB \rightarrow CD) \geq c(A \rightarrow BCD)$
    - Confidence is anti-monotone w.r.t. number of items on the RHS of the rule

# Theorem

- If Rule $X \rightarrow Y - X$ does not satisfy the confidence threshold then any rule $X' \rightarrow Y - X'$ where $X'$ is a subset of $X$ does not satisfy the confidence threshold as well.

# Rule Generation for Apriori Algorithm

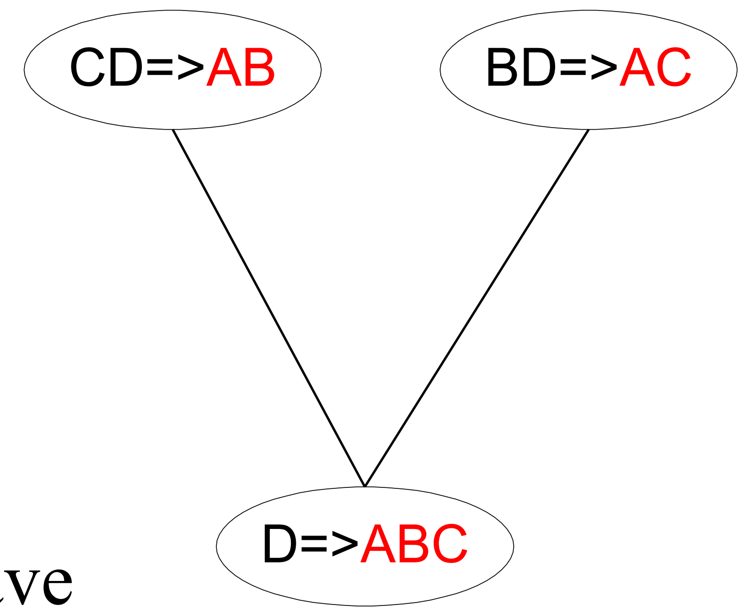
## Lattice of rules



# Rule Generation for Apriori Algorithm

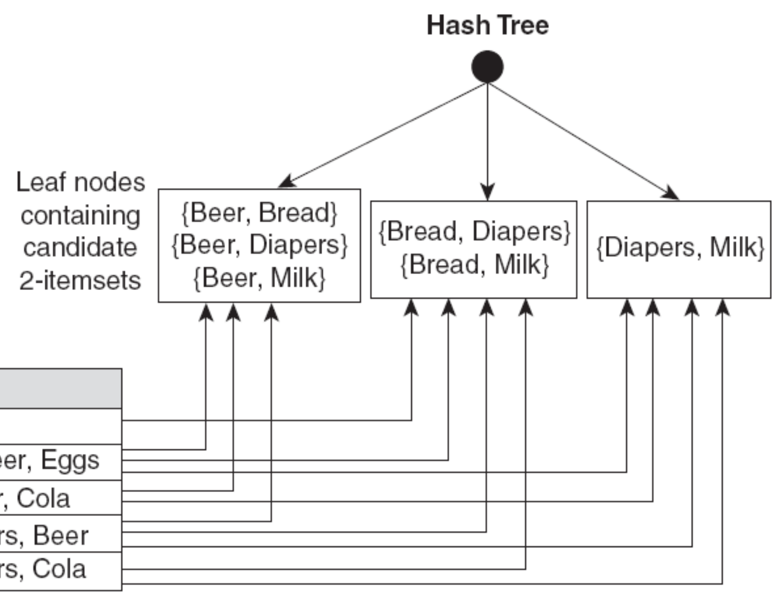
- Candidate rule is generated by merging two rules that share the same prefix
in the rule consequent

- join(CD=>AB,BD=>AC)
would produce the candidate
rule D => ABC
- Prune rule D=>ABC if its
super-set AD=>BC does not have
high confidence



# Reducing Number of Comparisons

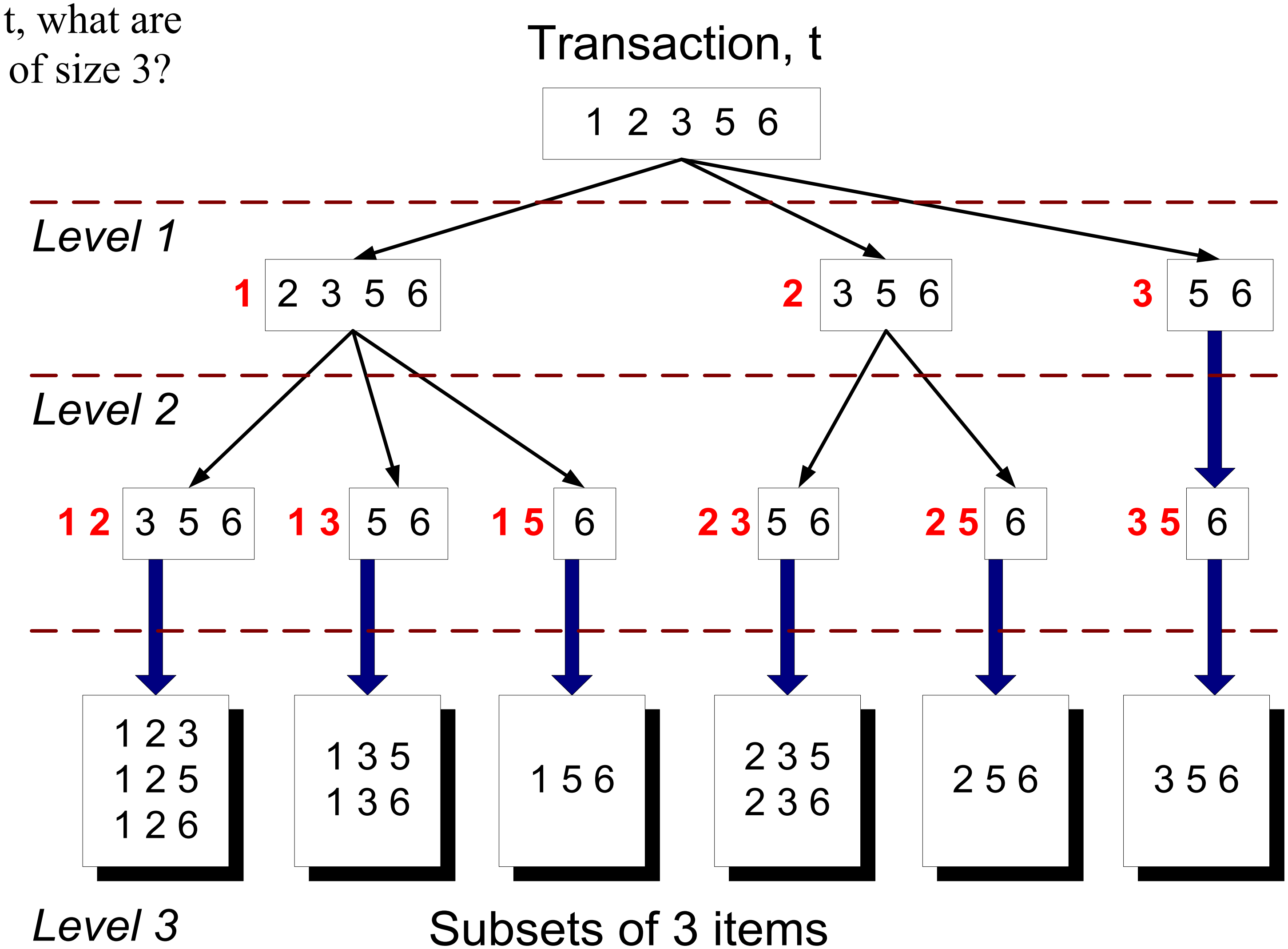
- Candidate counting:
  - Scan the database of transactions to determine the support of each candidate itemset
  - To reduce the number of comparisons, store the candidates in a hash structure
    - Instead of matching each transaction against every candidate, match it against candidates contained in the hashed buckets




| TID | Items |
|---|---|
| 1 | Bread, Milk |
| 2 | Bread, Diapers, Beer, Eggs |
| 3 | Milk, Diapers, Beer, Cola |
| 4 | Bread, Milk, Diapers, Beer |
| 5 | Bread, Milk, Diapers, Cola |

# Subset Operation (Enumeration)

Given a transaction t, what are the possible subsets of size 3?



# Generate Hash Tree

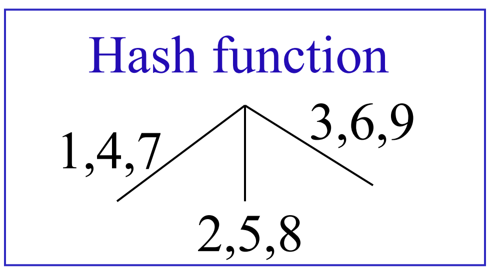
Suppose you have 15 candidate itemsets of length 3:

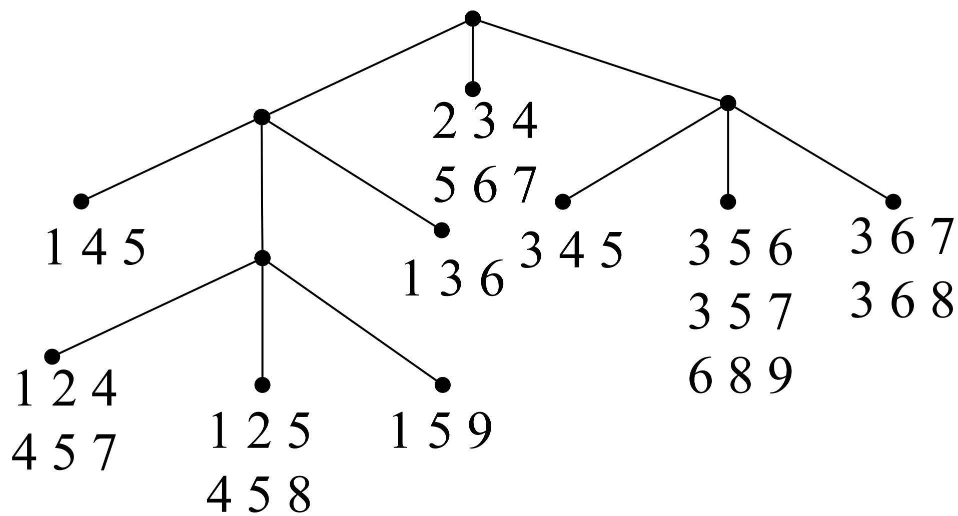
{1 4 5}, {1 2 4}, {4 5 7}, {1 2 5}, {4 5 8}, {1 5 9}, {1 3 6}, {2 3 4}, {5 6 7}, {3 4 5}, {3 5 6}, {3 5 7}, {6 8 9}, {3 6 7}, {3 6 8}

You need:

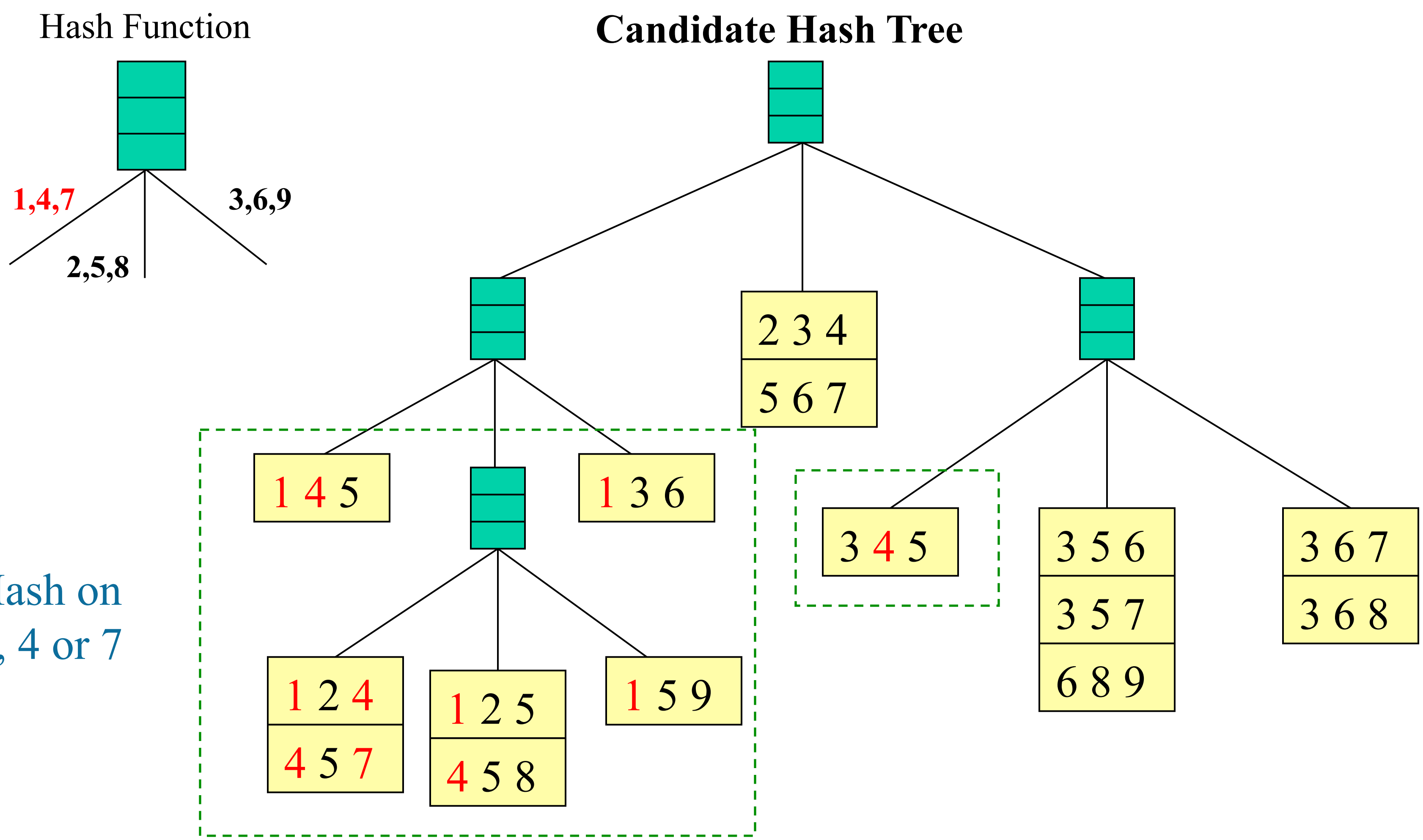
• Hash function

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



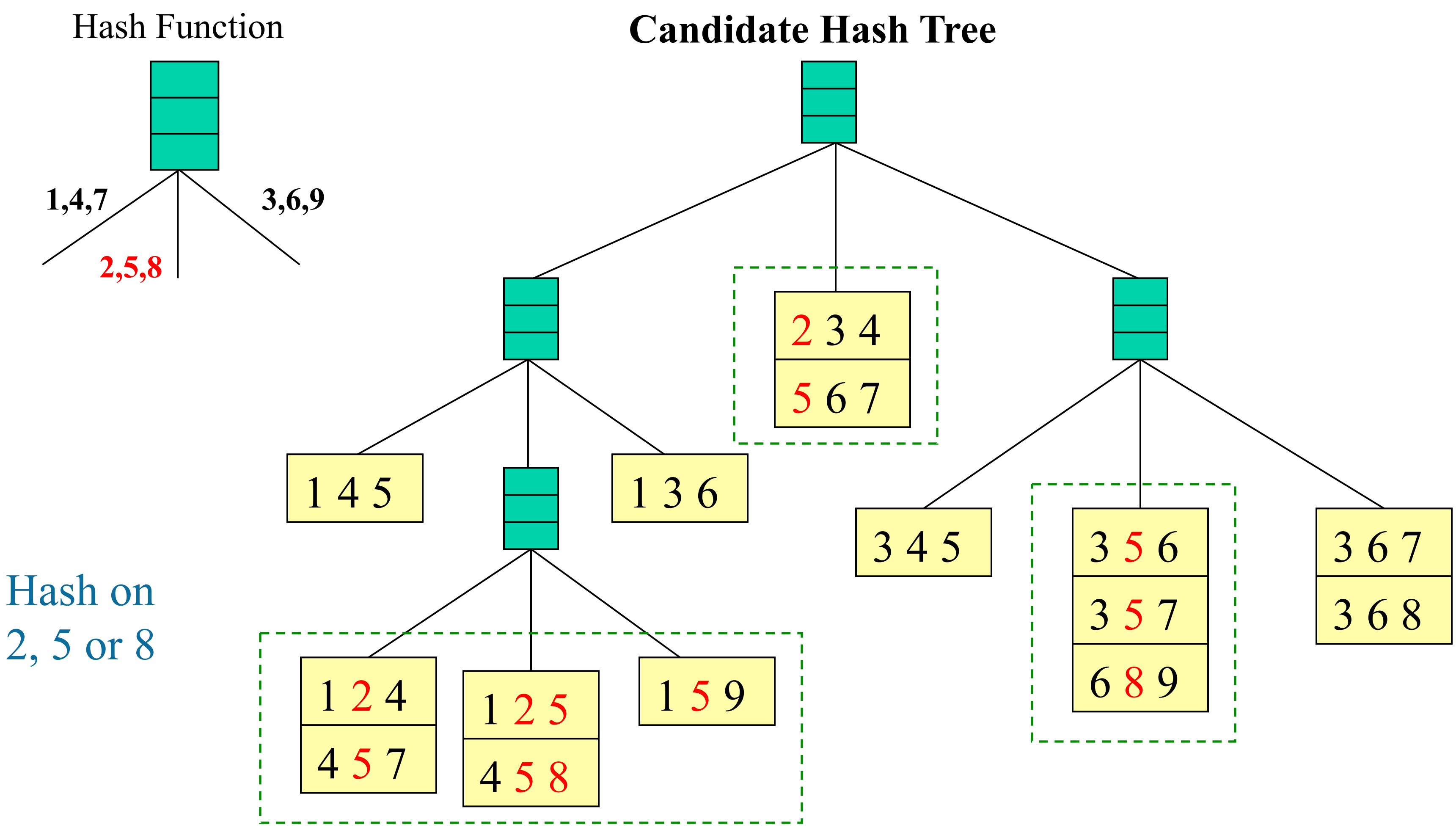



# Association Rule Discovery: Hash tree

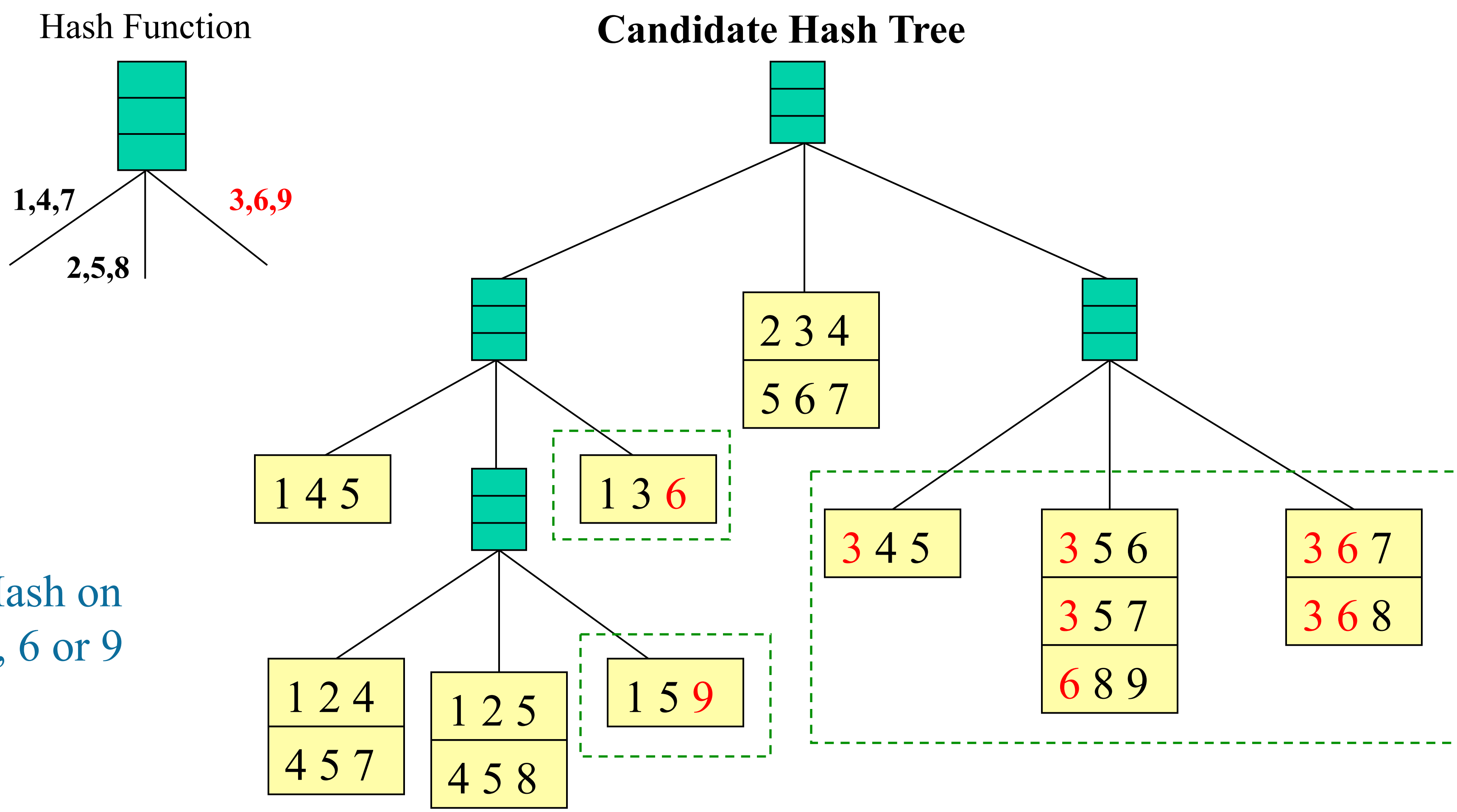



Hash on
1, 4 or 7

# Association Rule Discovery: Hash tree

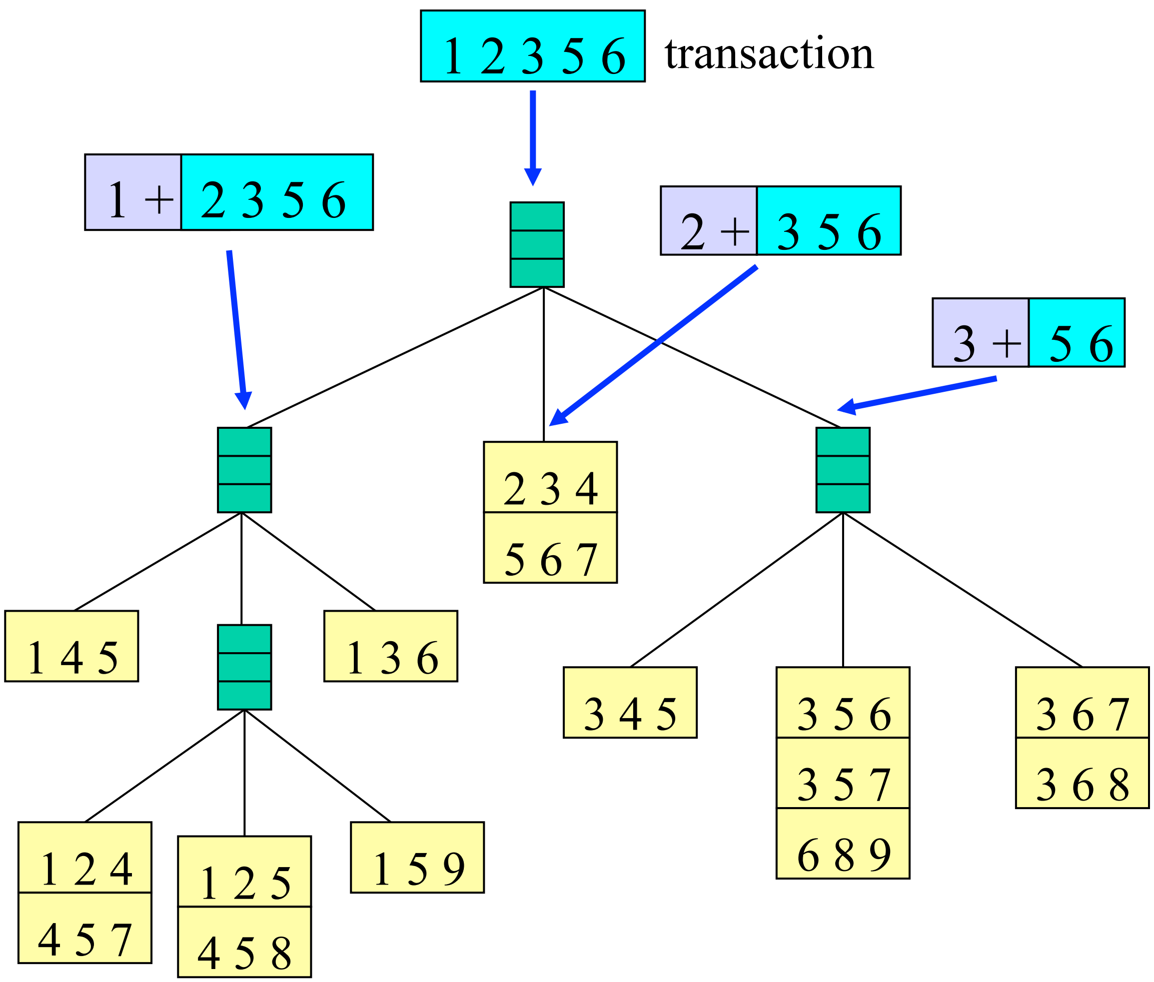


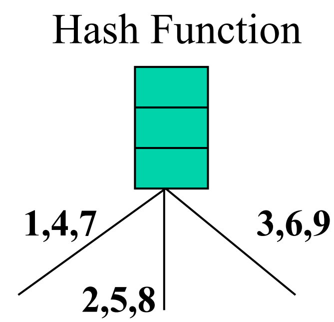
# Association Rule Discovery: Hash tree

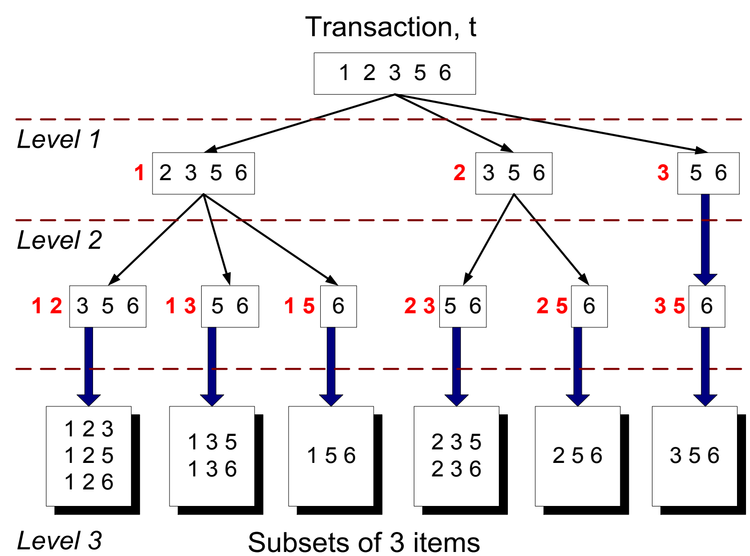



Hash on
3, 6 or 9

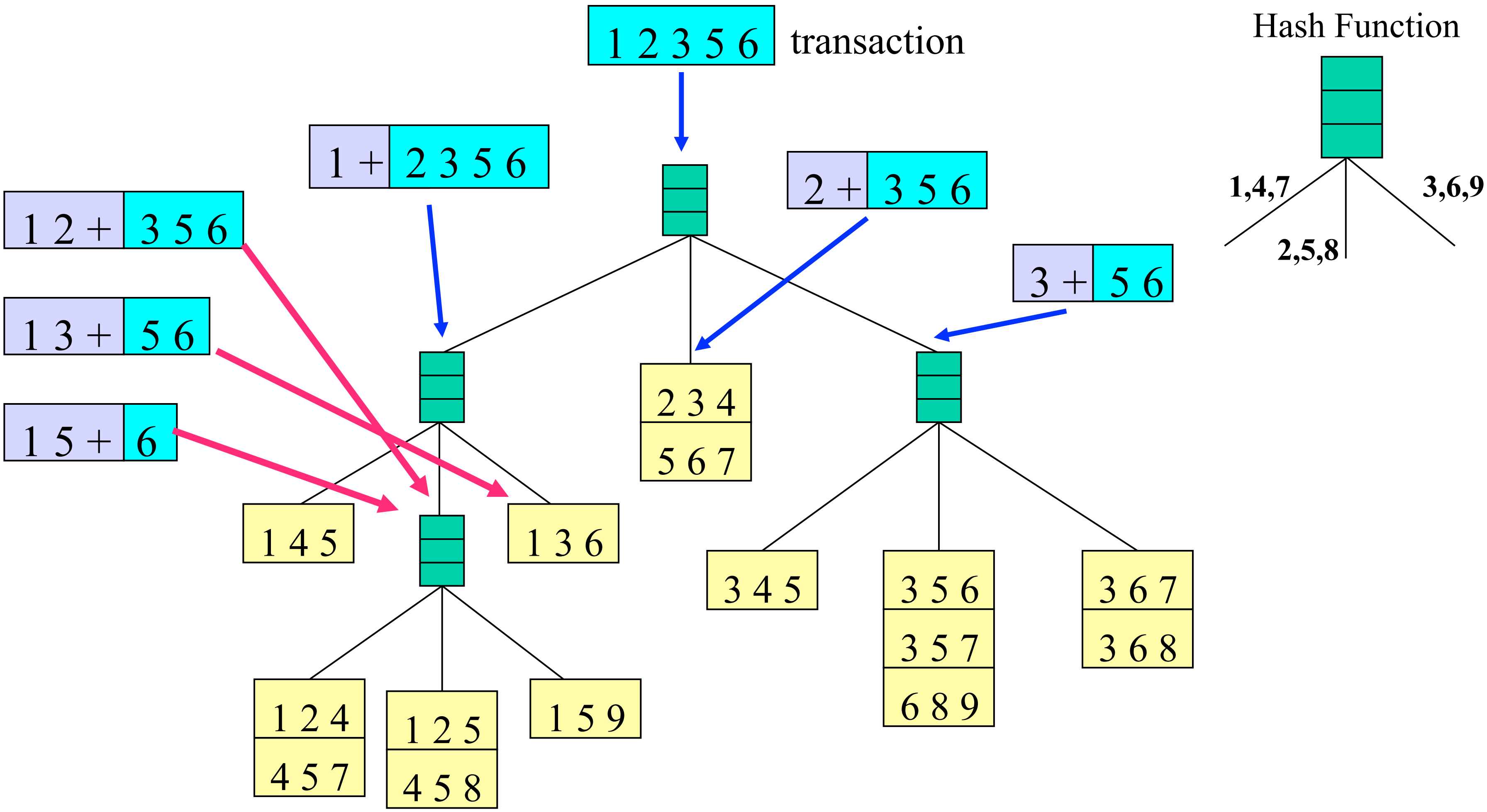
# Subset Operation Using Hash Tree



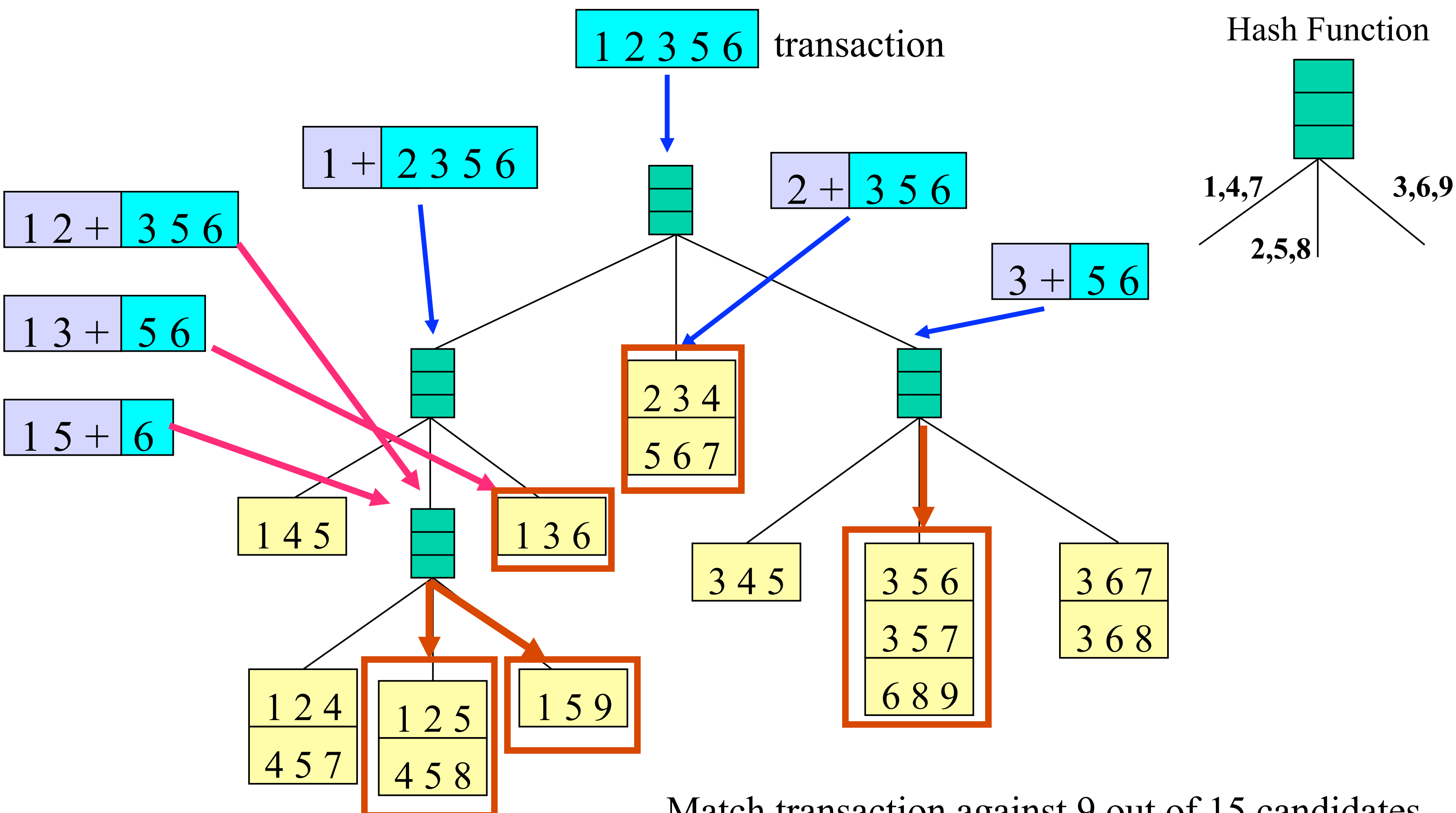



# Subset Operation Using Hash Tree



# Subset Operation Using Hash Tree




Match transaction against 9 out of 15 candidates

# Factors Affecting Complexity

- Choice of minimum support threshold
  - Lowering support threshold results in more frequent itemsets
  - This may increase number of candidates and max length of frequent itemsets
- Dimensionality (number of items) of the data set
  - More space is needed to store support count of each item
  - If number of frequent items also increases, both computation and I/O costs may also increase
- Size of database
  - Since Apriori makes multiple passes, run time of algorithm may increase with number of transactions
- Average transaction width
  - Transaction width increases with denser data sets
  - This may increase max length of frequent itemsets and traversals of hash tree (number of subsets in a transaction increases with its width)