Data Mining 2025

Association Analysis: Basic Concepts and Algorithms

Dept. of Computer Science and Information Engineering

National Cheng Kung University

Kun-Ta Chuang

ktchuang@mail.ncku.edu.tw

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} → {Beer},
{Milk, Bread} → {Eggs,Coke},
{Beer, Bread} → {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 → Y, where X and Y are itemsets
  - Example: {Milk, Diaper} → {Beer}
- 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

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:

{Milk, Diaper} ⇒ Beer

s = σ(Milk, Diaper, Beer) / |T| = 2/5 = 0.4

c = σ(Milk, Diaper, Beer) / σ(Milk, Diaper) = 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

⇒ **Computationally prohibitive!**

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:

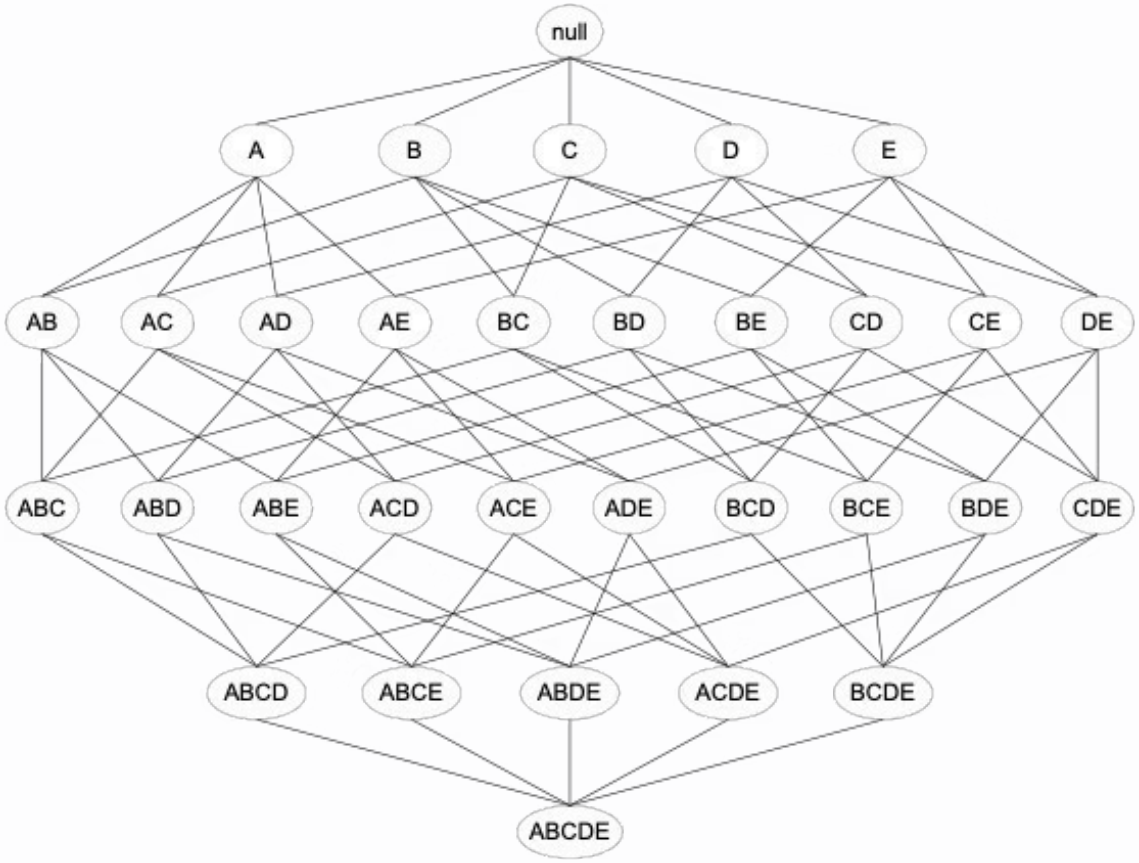
- All the above rules are binary partitions of the same itemset: {Milk, Diaper, Beer}
- Rules originating from the same itemset have identical support but can have different confidence
- Thus, we may decouple the support and confidence requirements

Mining Association Rules

- 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 is still computationally expensive

Frequent Itemset Generation

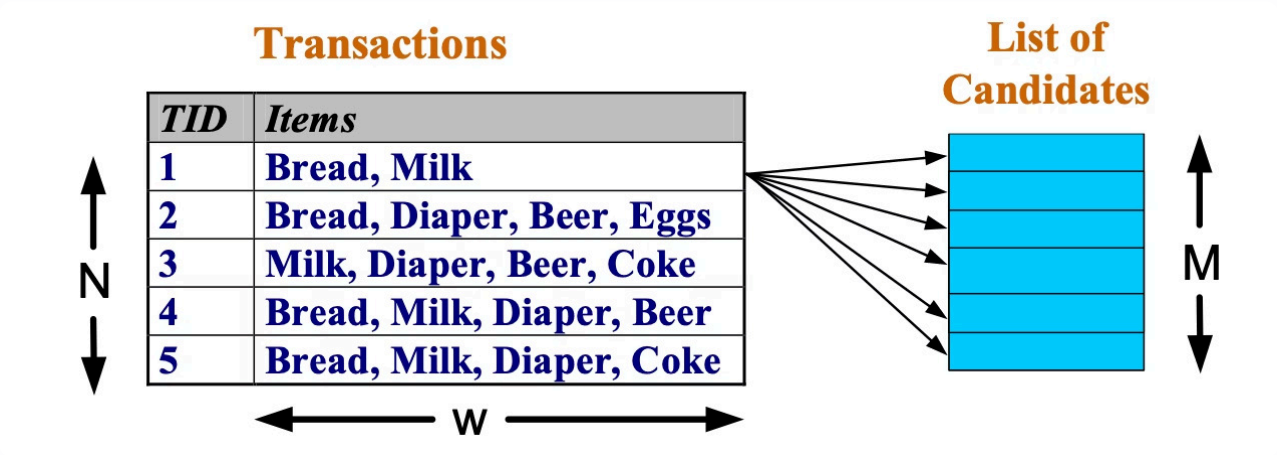


Given d items, there are 2d possible candidate itemsets

Frequent Itemset Generation

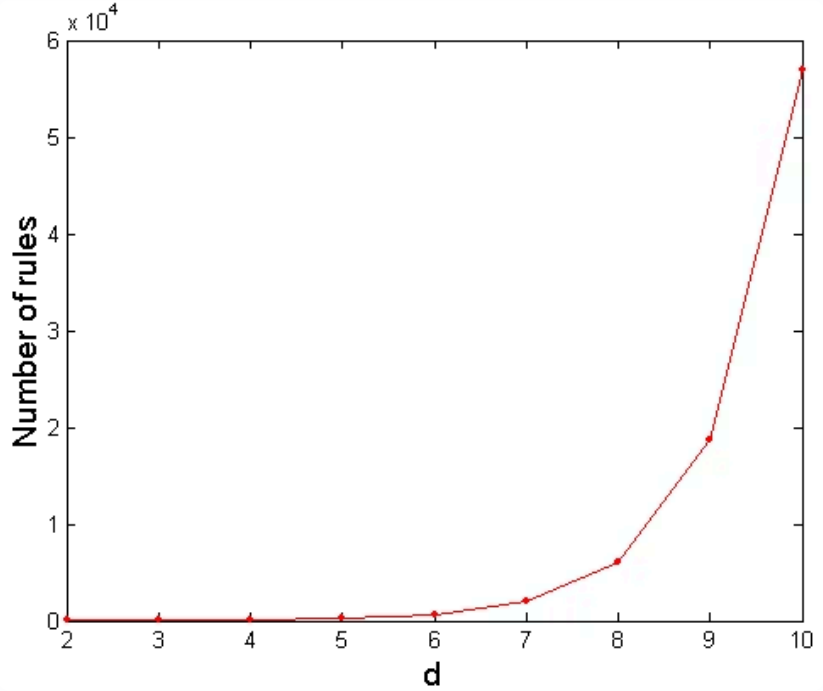
Brute-force approach:

- Each itemset in the lattice is a **candidate** frequent itemset
- Count the support of each candidate by scanning the database
- Match each transaction against every candidate
- Complexity ~ O(NMw) => **Expensive since M = 2d** !!!



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

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

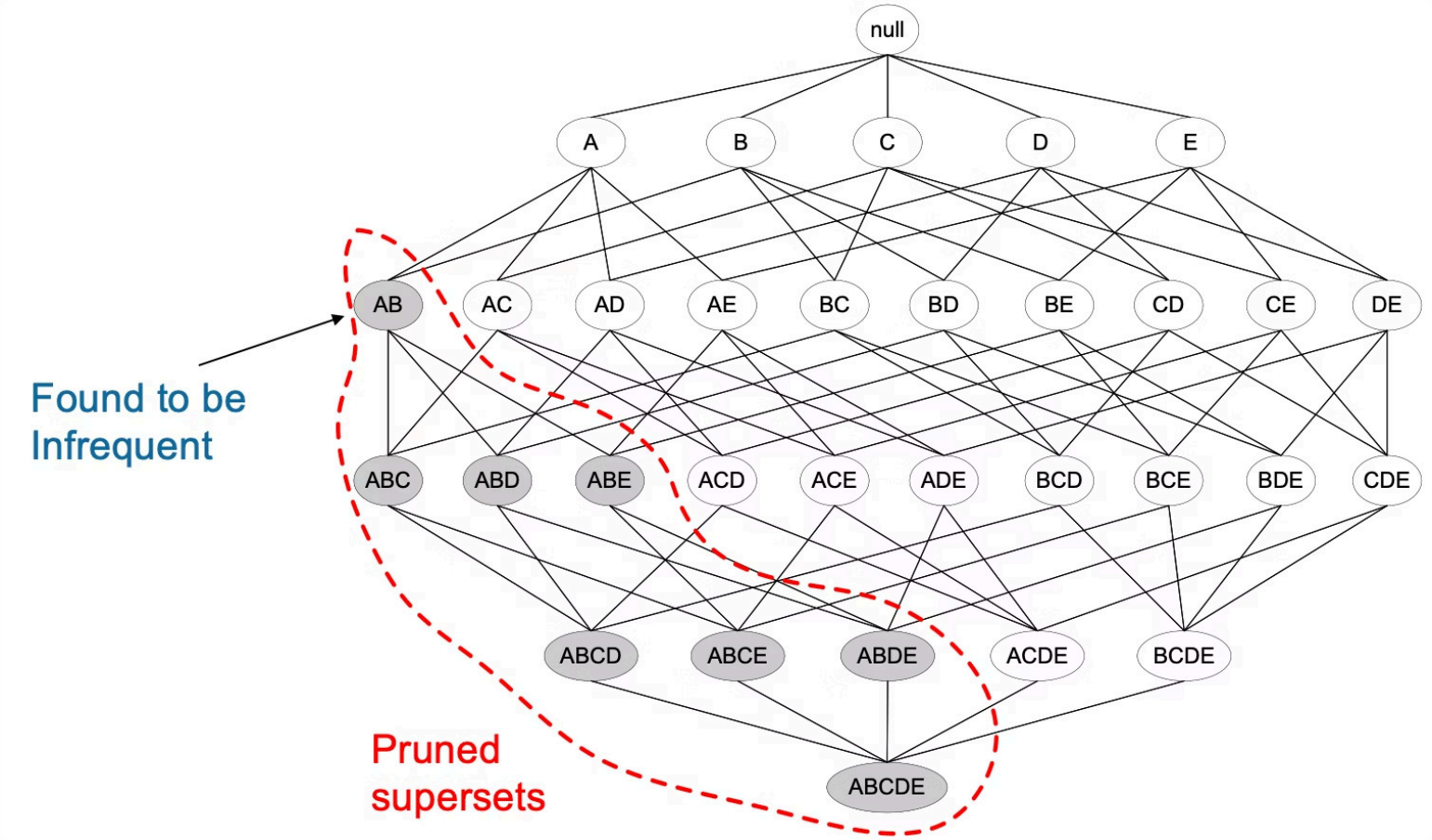
Reducing Number of Candidates

- **Apriori principle**:
  - If an itemset is frequent, then all of its subsets must also be frequent
- Apriori principle holds due to the following property of the support measure:

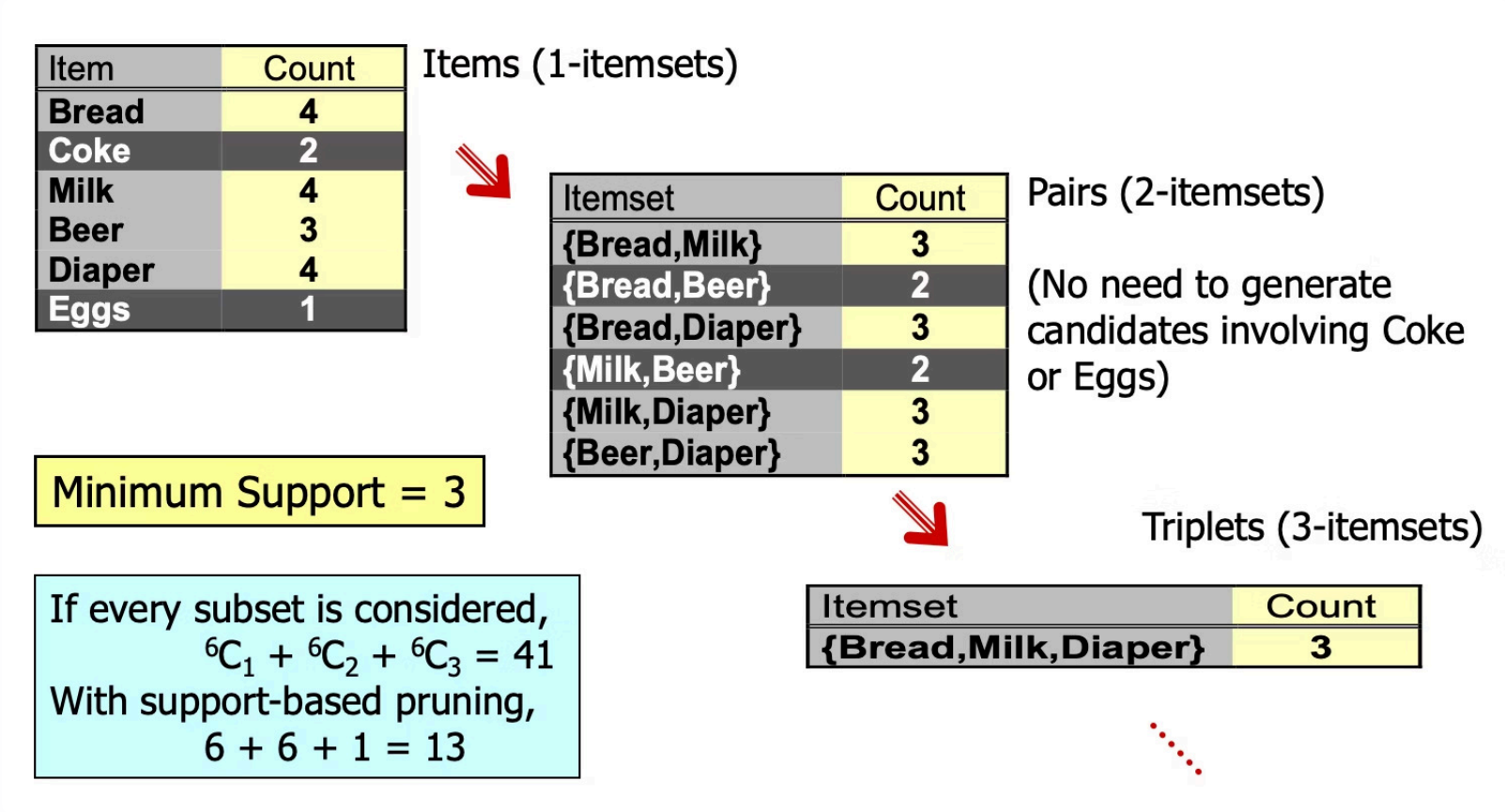
$$\forall X,Y : (X \subseteq Y) \Rightarrow s(X) \geq s(Y)$$

- Support of an itemset never exceeds the support of its subsets
- This is known as the **anti-monotone** property of support

Illustrating Apriori Principle



Illustrating Apriori Principle

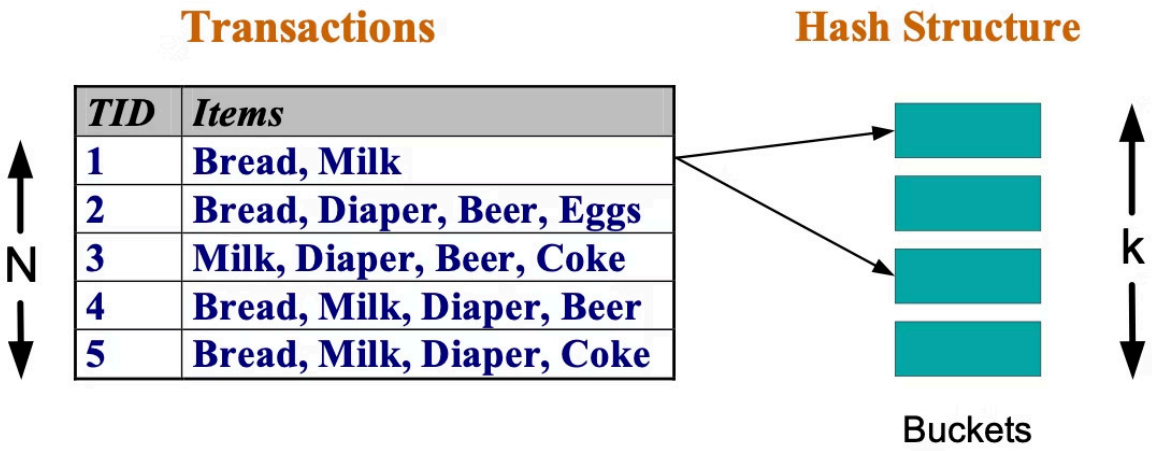


Apriori Algorithm

- 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

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



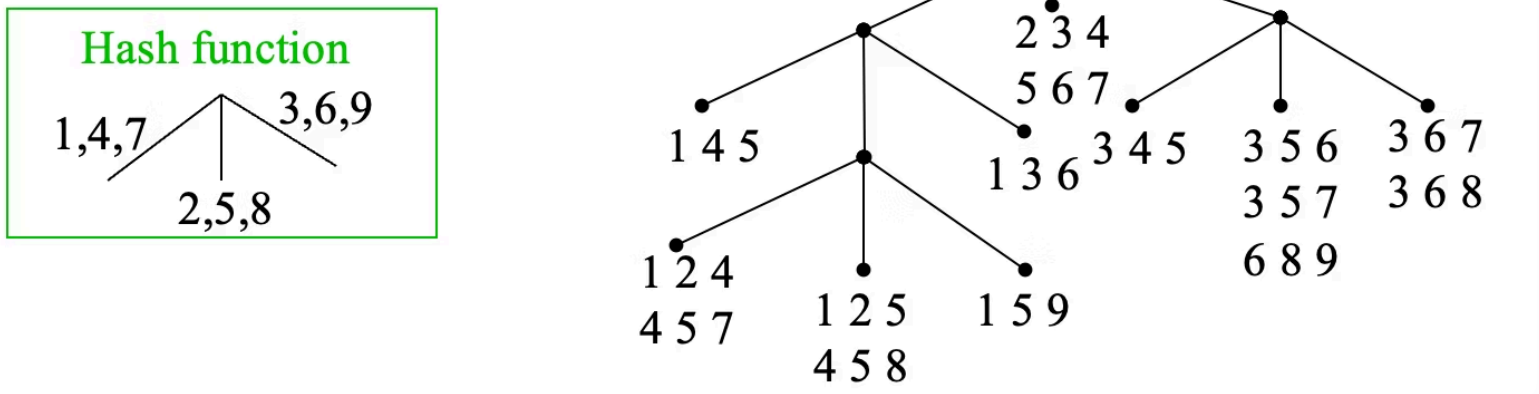
Generate Hash Tree

Suppose you have 15 candidate itemsets of length 3:

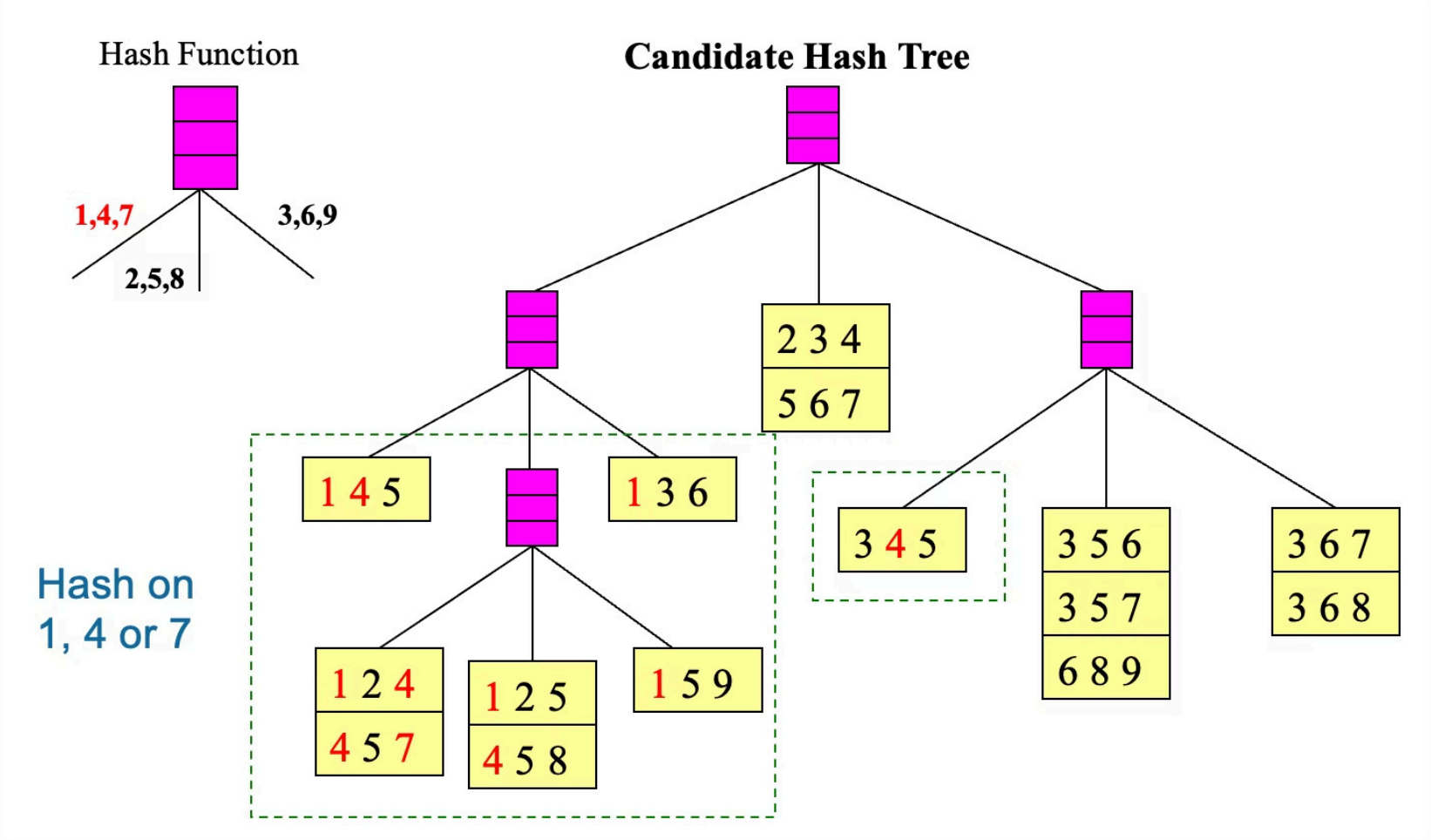
{1 4 5}, {1 2 4}, {4 5 7}, {1 2 5}, {4 5 8}, {1 5 9}, {1 3 6}, {2 3 4}, {5 6 7}, {3 4 5}, {3 5 6}, {3 5 7}, {6 8 9}, {3 6 7}, {3 6 8}

You need:

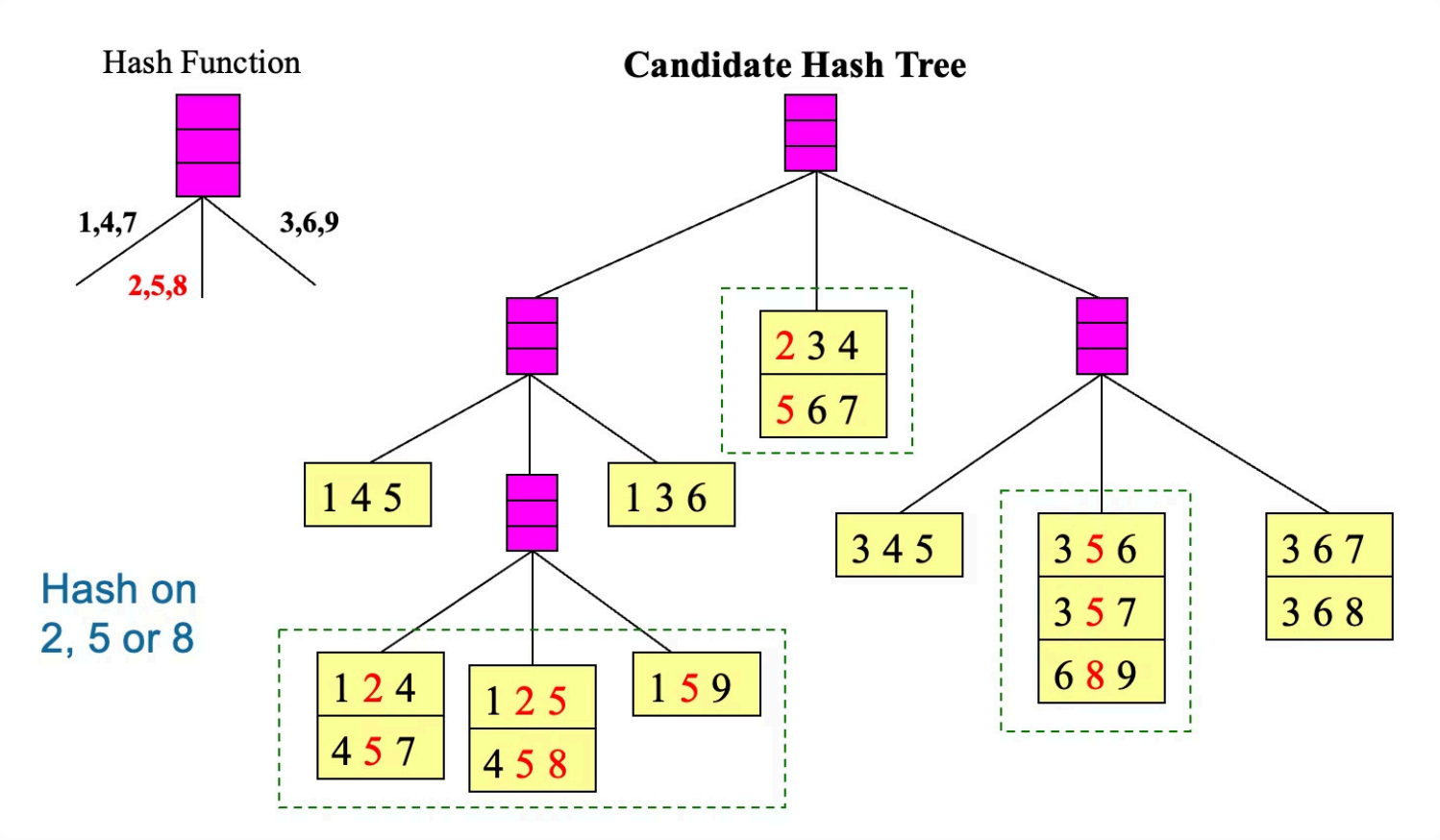
- Hash function
- Max leaf size: max number of itemsets stored in a leaf node (if number of candidate itemsets exceeds max leaf size, split the node)



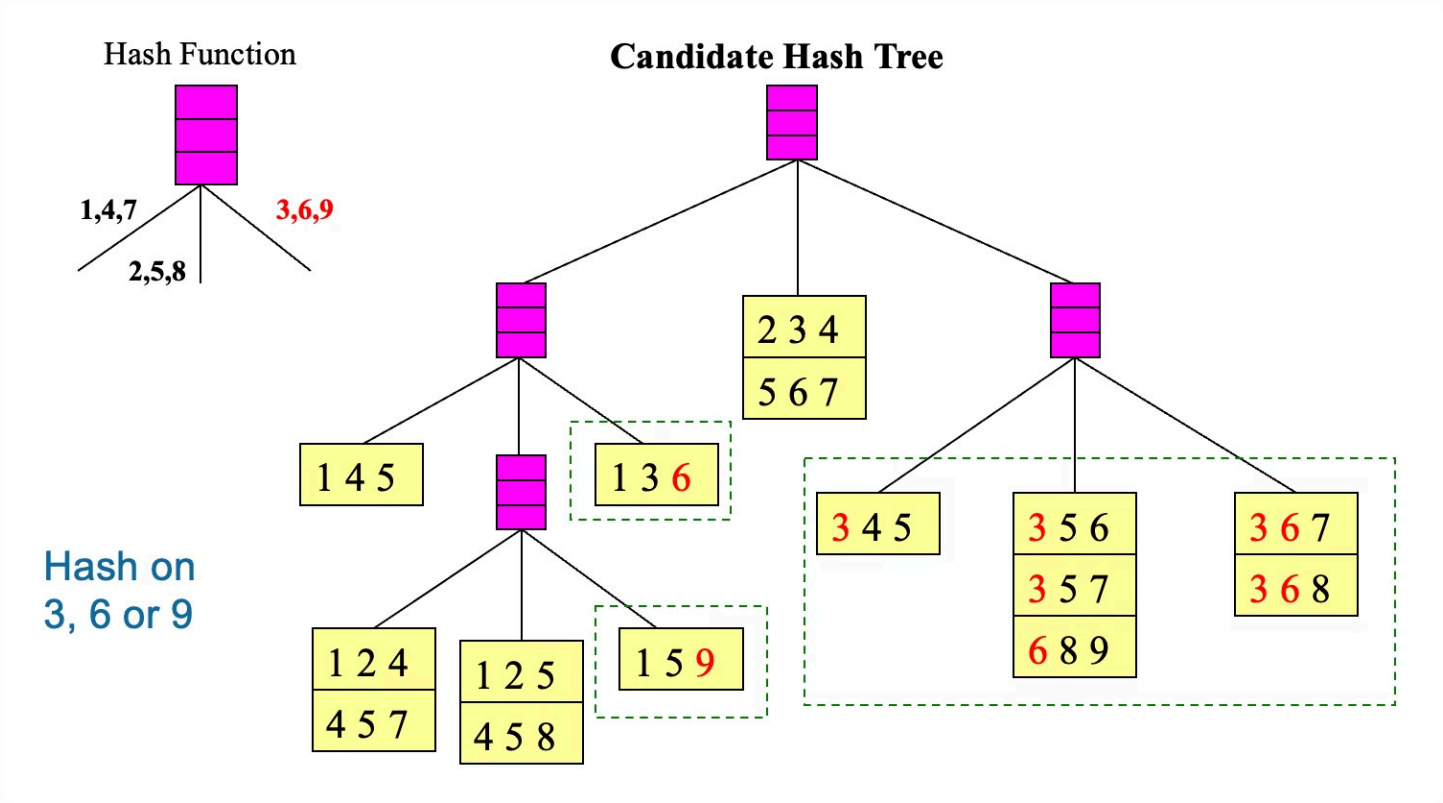
Association Rule Discovery: Hash tree



Association Rule Discovery: Hash tree

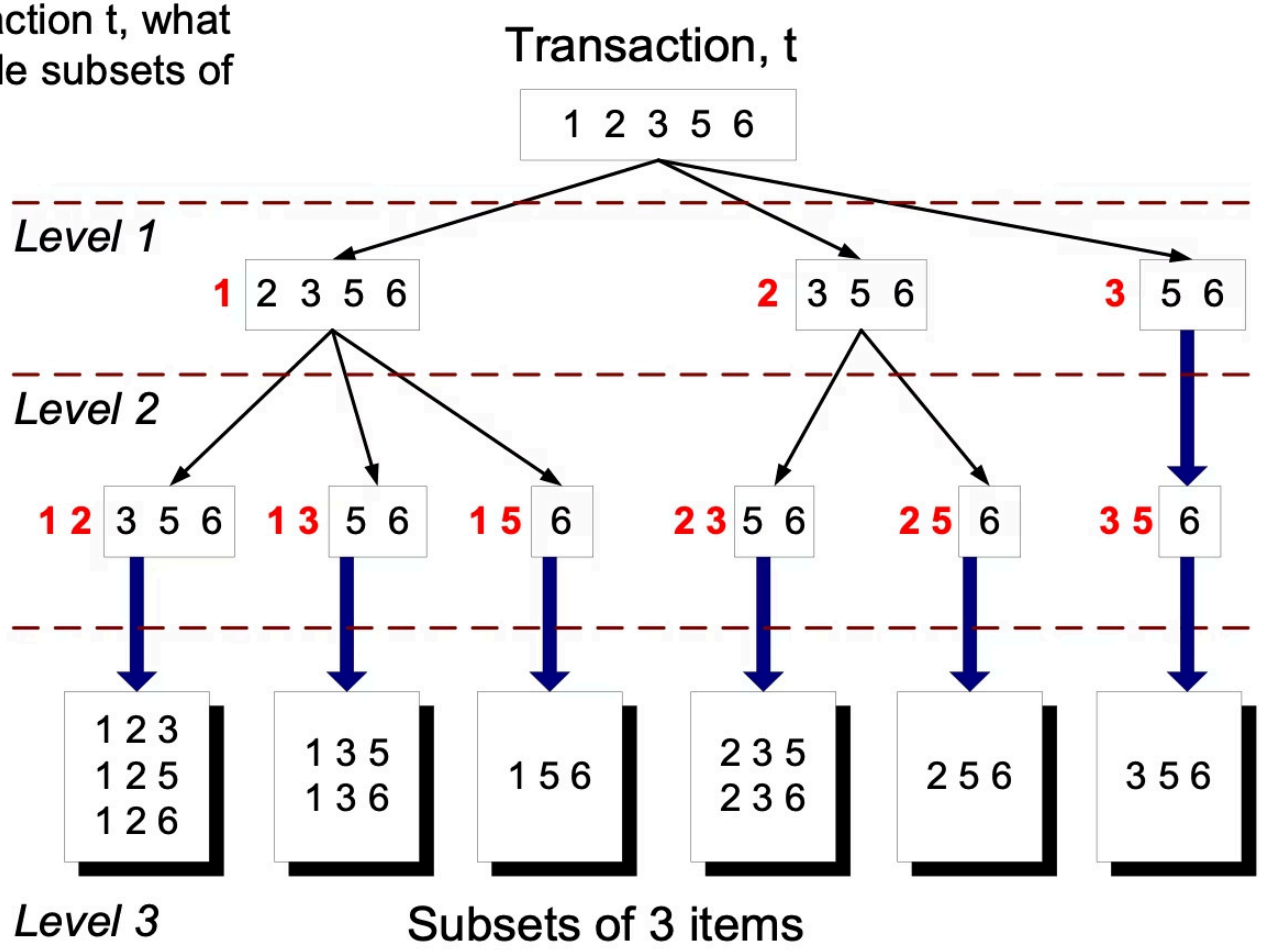


Association Rule Discovery: Hash tree

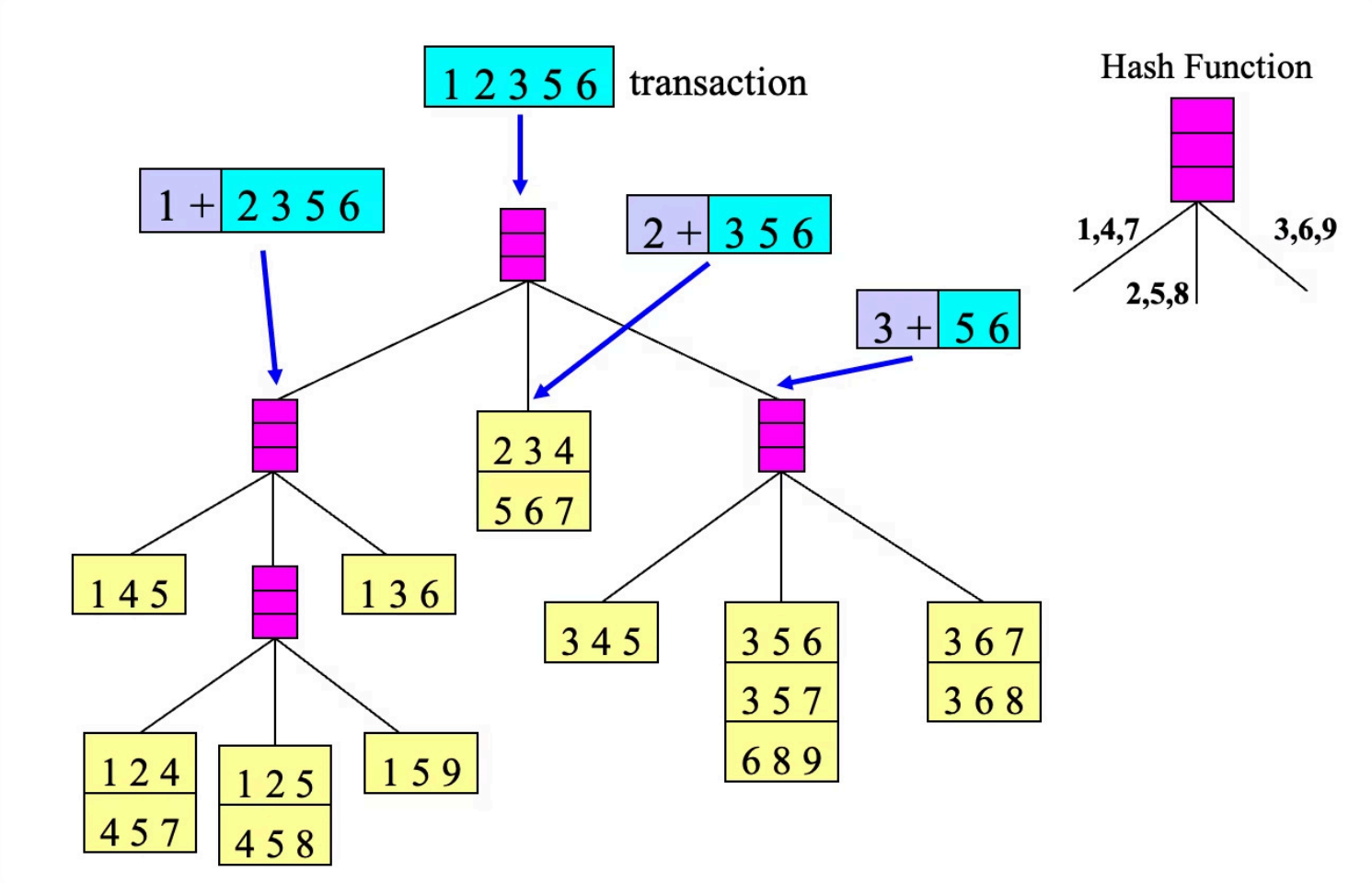


Subset Operation

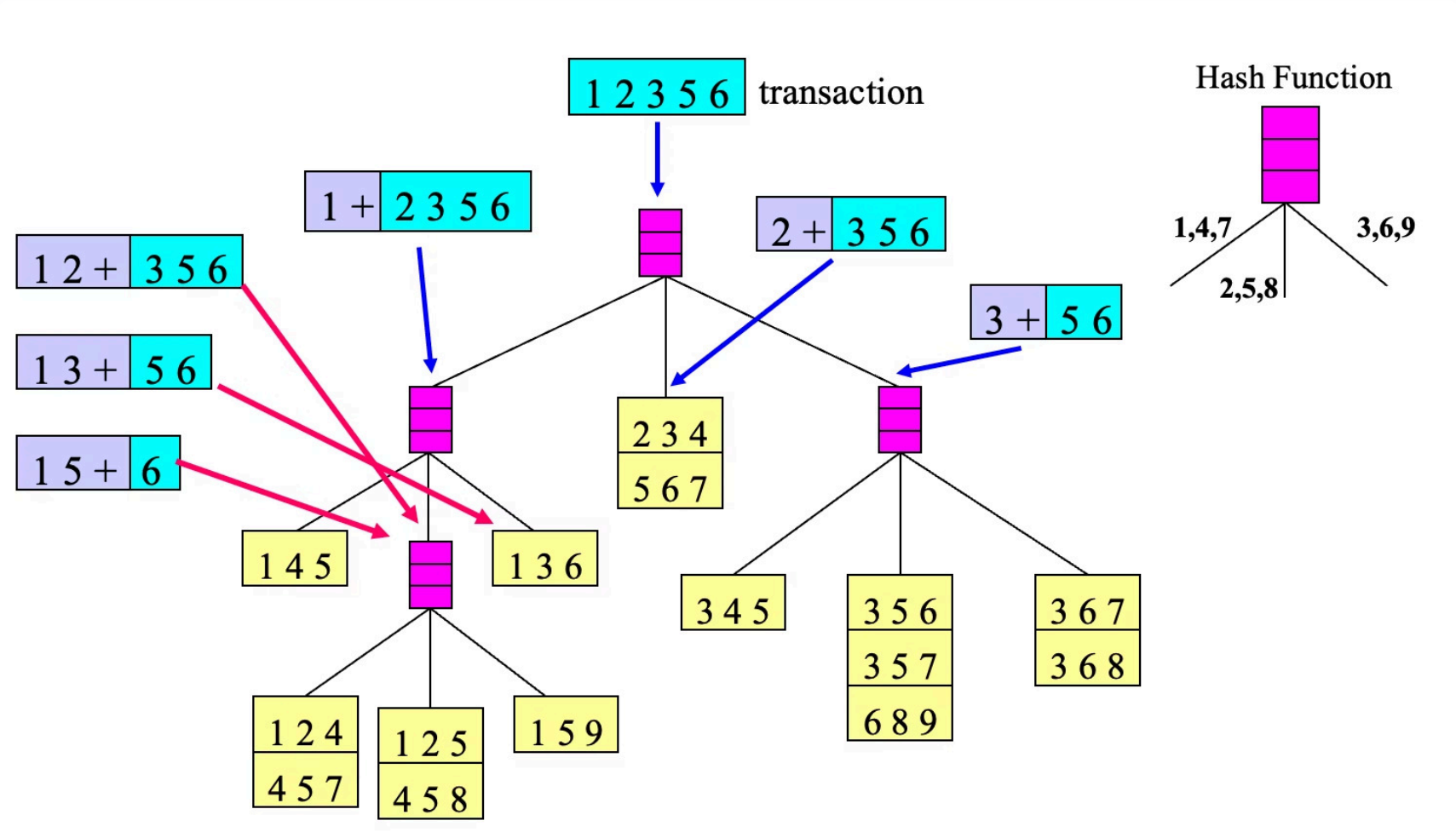
Given a transaction t, what are the possible subsets of size 3?



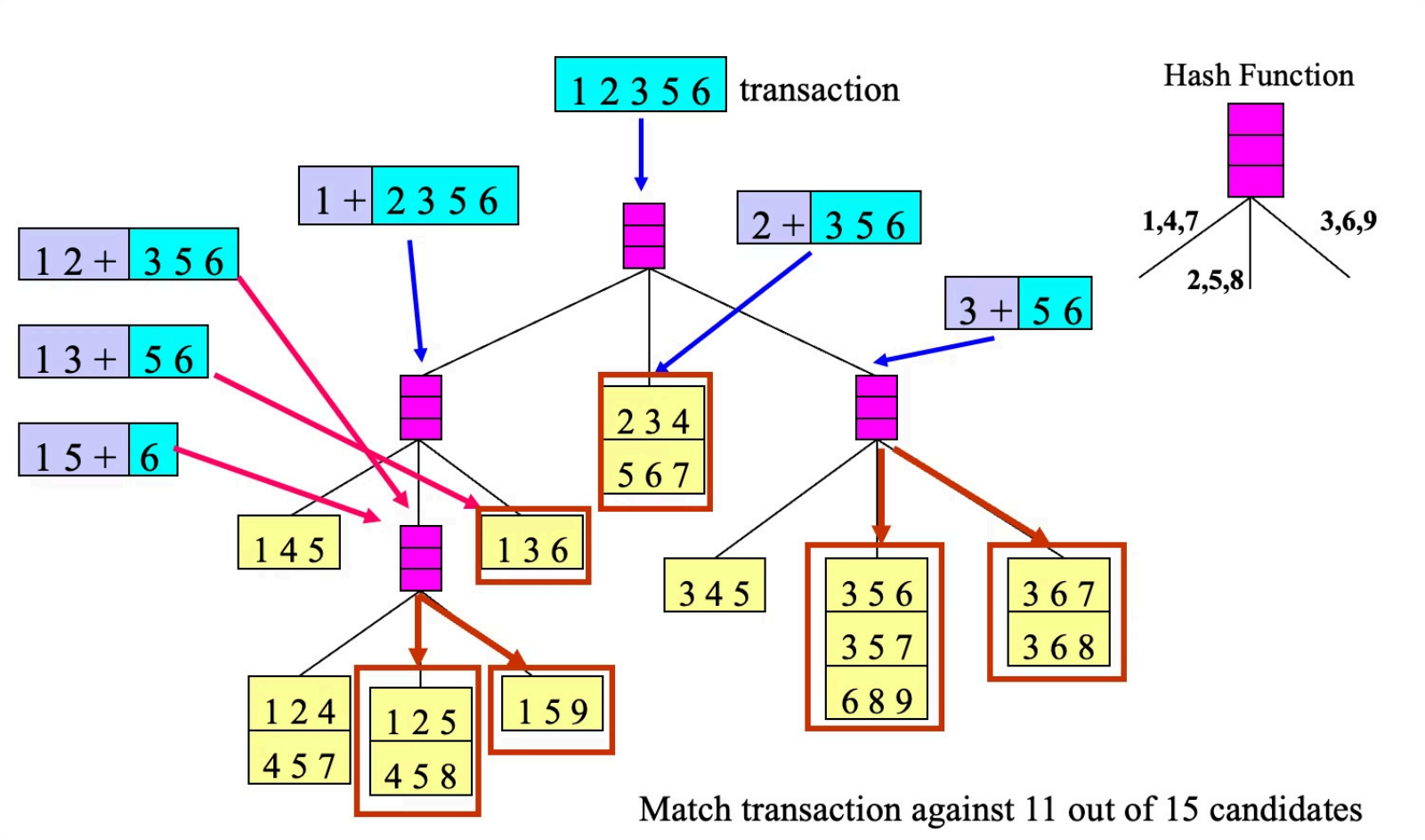
Subset Operation Using Hash Tree



Subset Operation Using Hash Tree



Subset Operation Using Hash Tree



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)