Data Mining
Association Analysis: Basic Concepts and Algorithms

Lecture Notes for Chapter 6

Introduction to Data Mining

by

Tan, Steinbach, Kumar

This is how it all started…

- Rakesh Agrawal, Tomasz Imielinski, Arun N. Swami: Mining Association Rules between Sets of Items in Large Databases. SIGMOD Conference 1993: 207-216
- Rakesh Agrawal, Ramakrishnan Srikant: Fast Algorithms for Mining Association Rules in Large Databases. VLDB 1994: 487-499

- These two papers are credited with the birth of Data Mining
- For a long time people were fascinated with Association Rules and Frequent Itemsets
  - Some people (in industry and academia) still are.

Market-Basket Data

- A large set of items, e.g., things sold in a supermarket.
- A large set of baskets, each of which is a small set of the items, e.g., the things one customer buys on one day.

- Really, a general many-to-many mapping (association) between two kinds of things, where the one (the baskets) is a set of the other (the items)
  - But we ask about connections among "items," not "baskets."
- The technology focuses on common events, not rare events ("long tail").

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

Applications

- Items = products; baskets = sets of products someone bought in one trip to the store.
  - Example application: given that many people buy beer and diapers together:
  - Run a sale on diapers; raise price of beer.
  - Only useful if many buy diapers & beer.
- Baskets = Web pages; items = words.
  - Example application: Unusual words appearing together in a large number of documents, e.g., “Brad” and “Angelina,” may indicate an interesting relationship.
- Baskets = sentences; items = documents containing those sentences.
  - Example application: Items that appear together too often could represent plagiarism.
  - Notice items do not have to be “in” baskets.

Mining Frequent Itemsets task

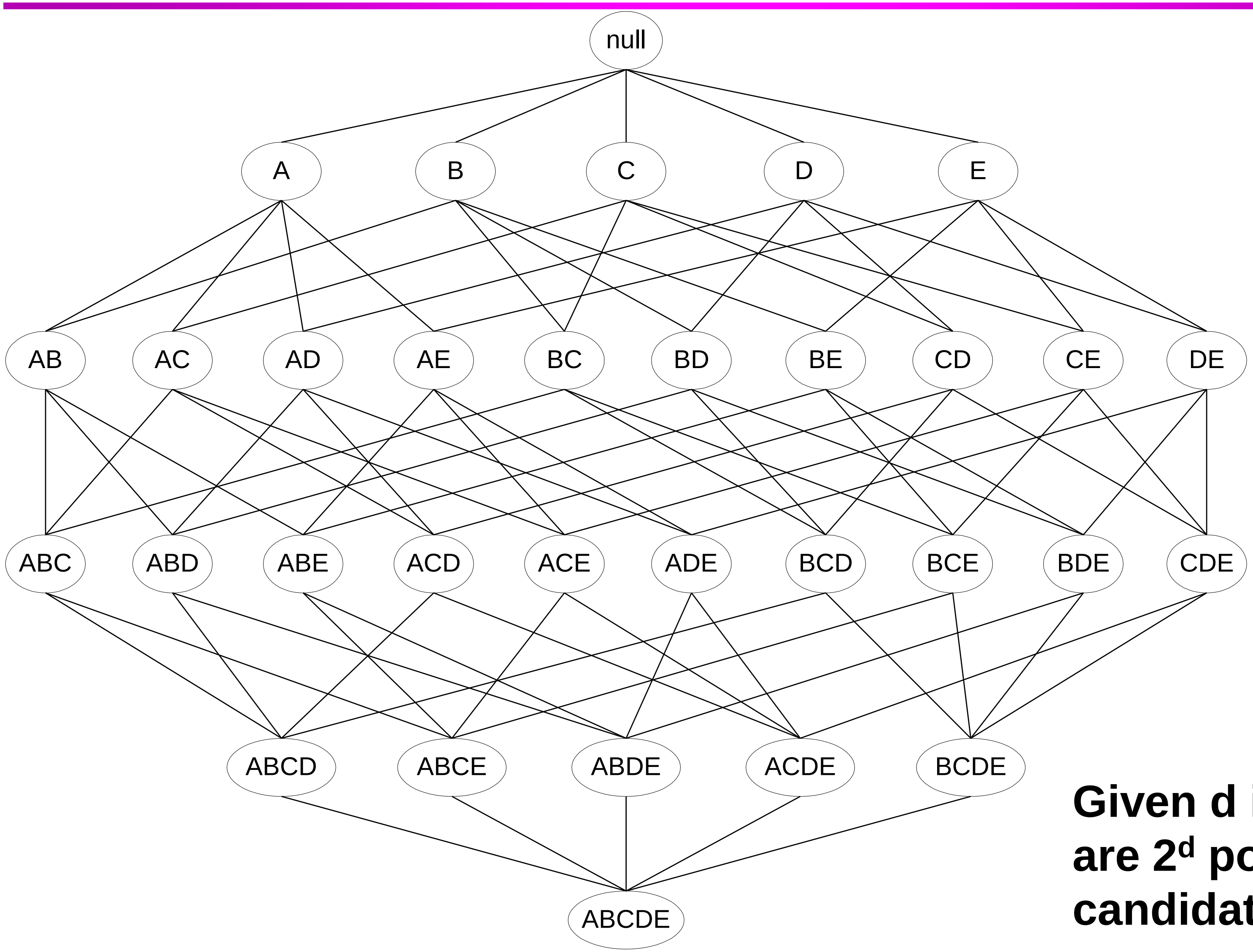
- Input: A set of transactions T, over a set of items I
- Output: All itemsets with items in I having
  - support ≥ *minsup* threshold

- Problem parameters:
  - N = |T|: number of transactions
  - d = |I|: number of (distinct) items
  - w: max width of a transaction
  - Number of possible itemsets?

$$M = 2^d$$

- Scale of the problem:
  - WalMart sells 100,000 items and can store billions of baskets.
  - The Web has billions of words and many billions of pages.

Frequent Itemset Generation



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

Frequent Itemset Generation

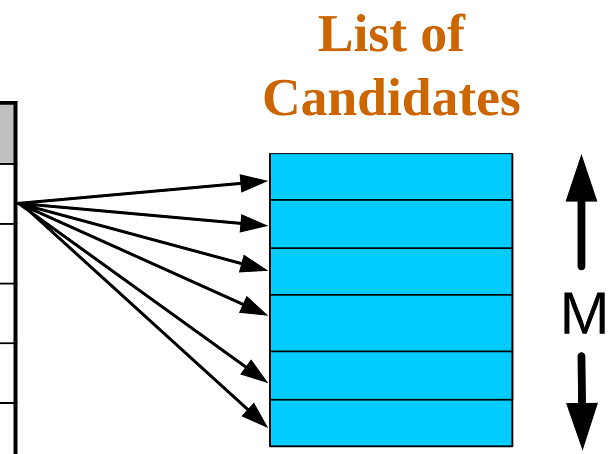
- Brute-force approach:
  - Each itemset in the lattice is a candidate frequent itemset
  - Count the support of each candidate by scanning the database

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

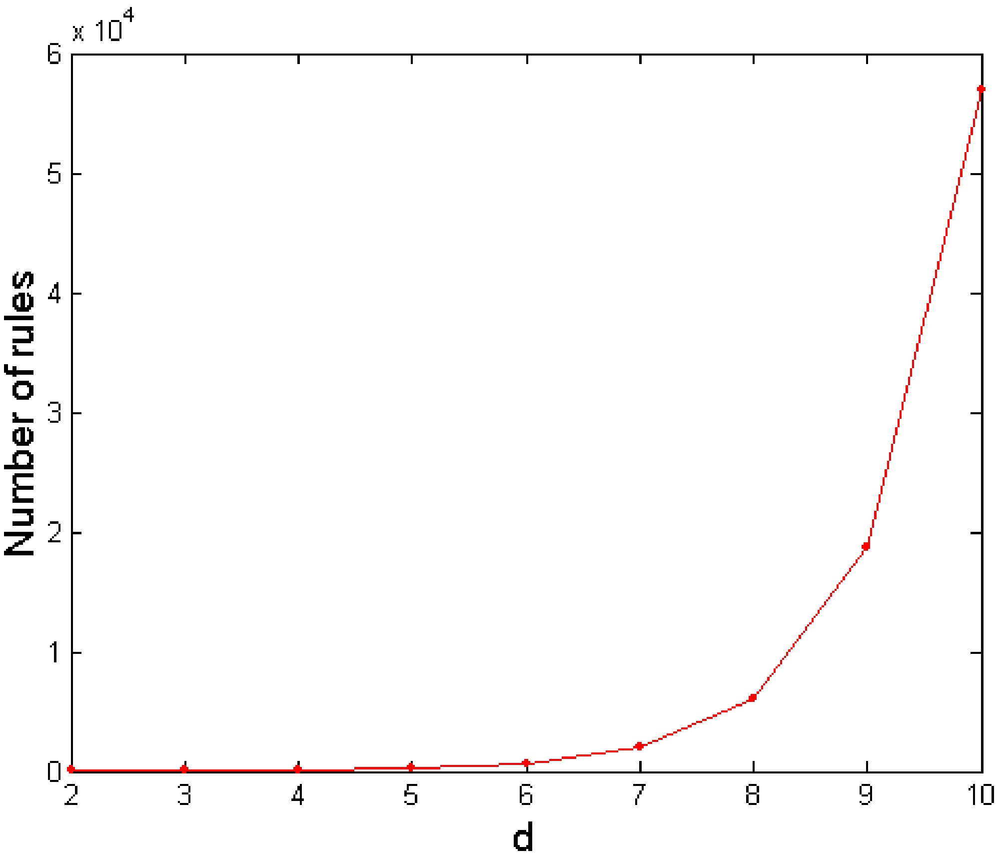
w



  - Match each transaction against every candidate
  - Complexity ~ O(NMw) => Expensive since $M = 2^d$!!!

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

Computation Model

- Typically, data is kept in flat files rather than in a database system.
  - Stored on disk.
  - Stored basket-by-basket.
  - Expand baskets into pairs, triples, etc. as you read baskets.
    - Use k nested loops to generate all sets of size k.

Example file: retail

```
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29
30 31 32
33 34 35
36 37 38 39 40 41 42 43 44 45 46
38 39 47 48
38 39 48 49 50 51 52 53 54 55 56 57 58
32 41 59 60 61 62
3 39 48
63 64 65 66 67 68
32 69
48 70 71 72
39 73 74 75 76 77 78 79
36 38 39 41 48 79 80 81
82 83 84
41 85 86 87 88
39 48 89 90 91 92 93 94 95 96 97 98 99 100 101
36 38 39 48 89
39 41 102 103 104 105 106 107 108
38 39 41 109 110
39 111 112 113 114 115 116 117 118
119 120 121 122 123 124 125 126 127 128 129 130 131 132 133
48 134 135 136
39 48 137 138 139 140 141 142 143 144 145 146 147 148 149
39 150 151 152
38 39 56 153 154 155
```

Example: items are positive integers, and each basket corresponds to a line in the file of space separated integers

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

The Apriori algorithm

Level-wise approach

C_k = candidate itemsets of size k
L_k = frequent itemsets of size k

1. $k = 1$, C_1 = all items
2. While C_k not empty

Frequent itemset generation

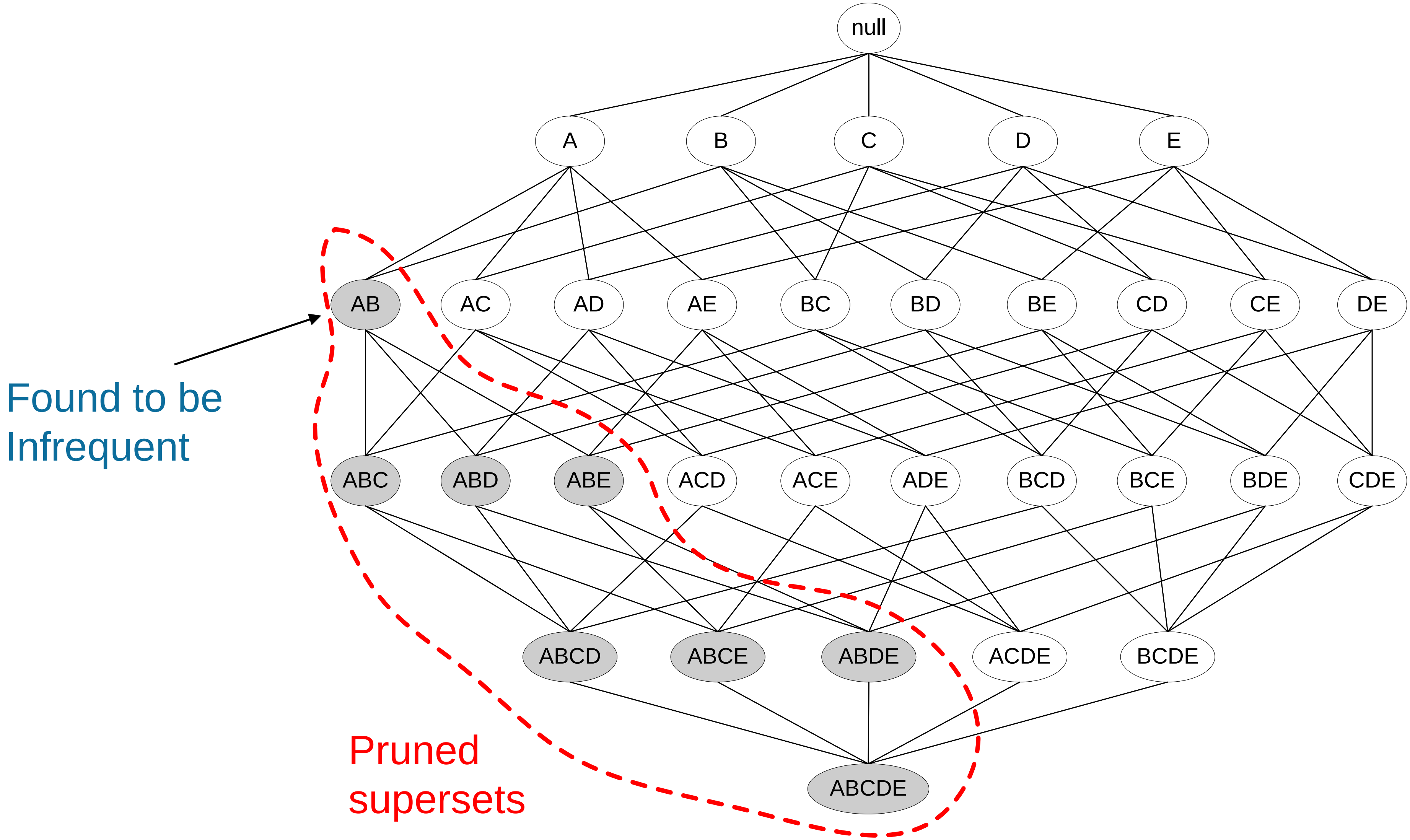
3. Scan the database to find which itemsets in C_k are frequent and put them into L_k

Candidate generation

4. Use L_k to generate a collection of candidate itemsets C_{k+1} of size $k+1$
5. $k = k+1$

R. Agrawal, R. Srikant: "Fast Algorithms for Mining Association Rules", *Proc. of the 20th Int'l Conference on Very Large Databases*, 1994.

Illustrating Apriori Principle



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
{Bread,Milk,Diaper}	**3**

If every subset is considered,

$^6C_1 + {}^6C_2 + {}^6C_3 = 41$

With support-based pruning,

$6 + 6 + 1 = 13$

Candidate Generation

- Basic principle (Apriori):
  – An itemset of size k+1 is candidate to be frequent only if all of its subsets of size k are known to be frequent
- Main idea:
  – Construct a candidate of size k+1 by combining frequent itemsets of size k
    - If k = 1, take the all pairs of frequent items
    - If k > 1, join pairs of itemsets that differ by just one item
    - For each generated candidate itemset ensure that all subsets of size k are frequent.

Generate Candidates C_{k+1}

- **Assumption: The items in an itemset are ordered**
  - **E.g., if integers ordered in increasing order, if strings ordered in lexicographic order**
    - **The order ensures that if item y > x appears before x, then x is not in the itemset**
- **The items in L_k are also listed in an order**

Create a candidate itemset of size k+1, by joining two itemsets of size k, that share the first k-1 items

Item 1	Item 2	Item 3
1	2	3
1	2	5
1	4	5

1	2	4	5

Are we missing something?
What about this candidate?

Generating Candidates C_{k+1} in SQL

- ***self-join*** L_k

  insert into C_{k+1}

  select $p.item_1, p.item_2, \ldots, p.item_k, q.item_k$

  from $L_k\ p, L_k\ q$

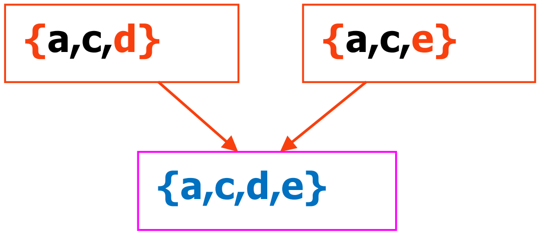
  where $p.item_1=q.item_1, \ldots, p.item_{k-1}=q.item_{k-1}, p.item_k < q.item_k$

Example

- L_3={abc, abd, acd, ace, bcd}
- ***Self-joining***: $L_3 * L_3$
  - abcd from abc and abd
  - acde from acd and ace

item1	item2	item3
a	b	c
a	b	d
a	c	d
a	c	e
b	c	d

item1	item2	item3
a	b	c
a	b	d
a	c	d
a	c	e
b	c	d



$p.item_1 = q.item_1, p.item_2 = q.item_2, p.item_3 < q.item_3$

Generate Candidates C_{k+1}

- Are we done? Are all the candidates valid?

Item 1	Item 2	Item 3
1	2	3
1	2	5
1	4	5

1	2	3	5

Is this a valid candidate?

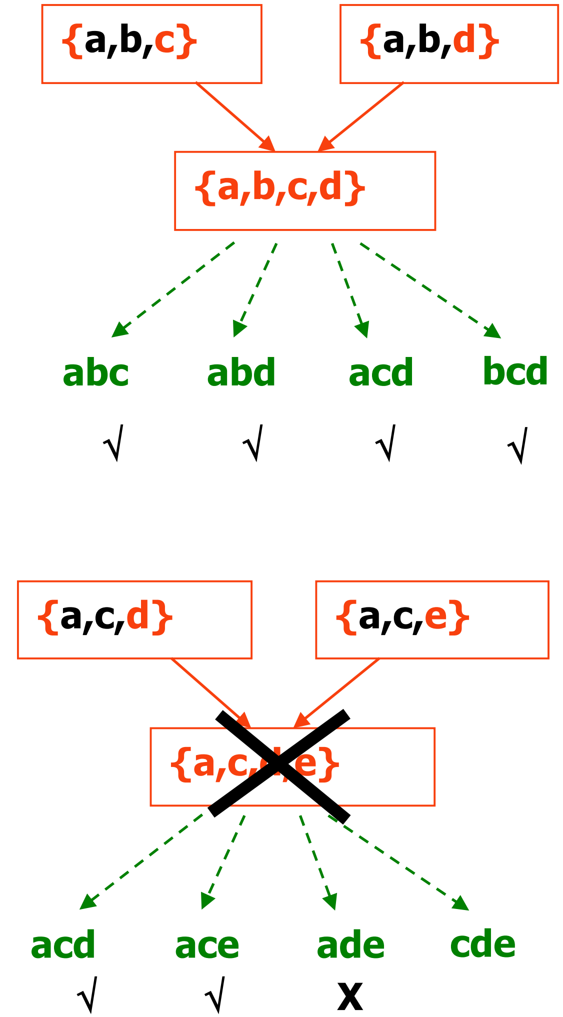
No. Subsets (1,3,5) and (2,3,5) should also be frequent

- Pruning step:
  - For each candidate (k+1)-itemset create all subset k-itemsets
  - Remove a candidate if it contains a subset k-itemset that is not frequent

Apriori principle

Example

- L_3={*abc, abd, acd, ace, bcd*}
- ***Self-joining***: $L_3 * L_3$
  - *abcd* from *abc* and *abd*
  - *acde* from *acd* and *ace*
- ***Pruning:***
  - *abcd* is kept since all subset itemsets are in L_3
  - *acde* is removed because *ade* is not in L_3
- C_4={*abcd*}



Generate Candidates C_{k+1}

- We have all frequent k-itemsets L_k
- Step 1: self-join L_k
  - Create set C_{k+1} by joining frequent k-itemsets that share the first k-1 items
- Step 2: prune
  - Remove from C_{k+1} the itemsets that contain a subset k-itemset that is not frequent

Reducing Number of Comparisons

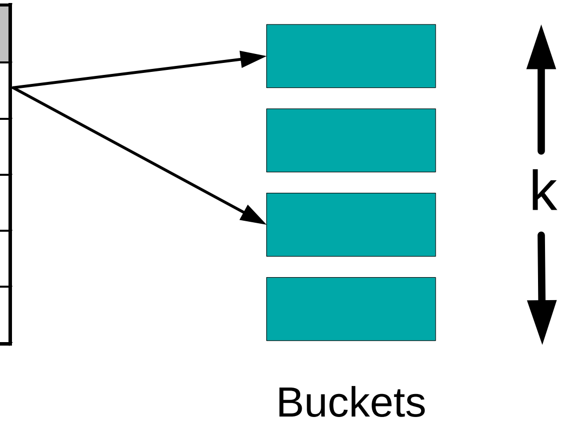
- Candidate counting:
  – Scan the database of transactions to determine the support of each candidate itemset
  – To reduce the number of comparisons, store the candidates in a hash structure
    - ◆ Instead of matching each transaction against every candidate, match it against candidates contained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

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



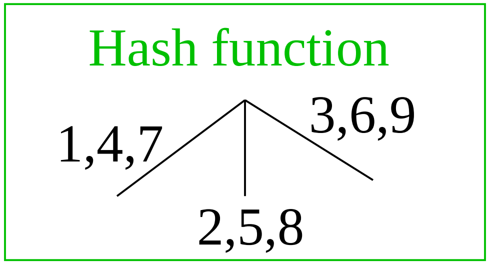
Generate Hash Tree

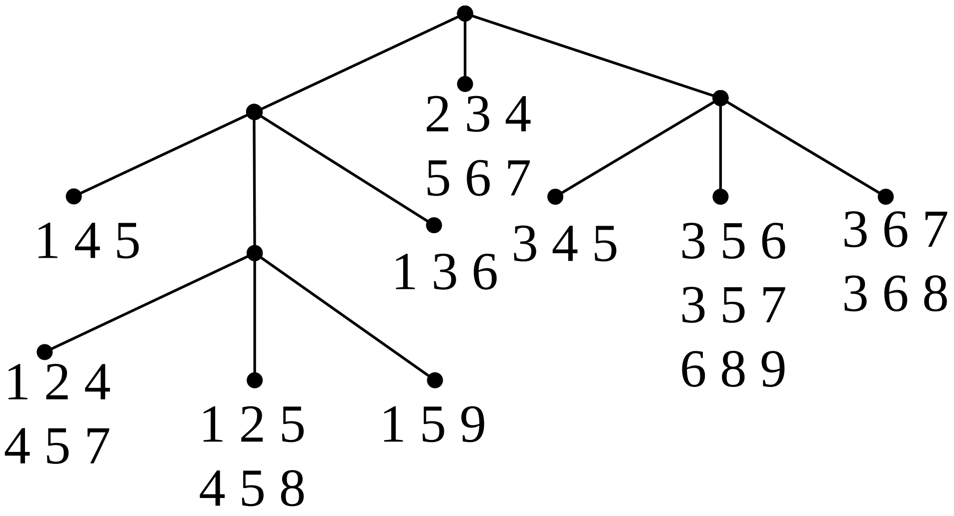
Suppose you have 15 candidate itemsets of length 3:

{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}

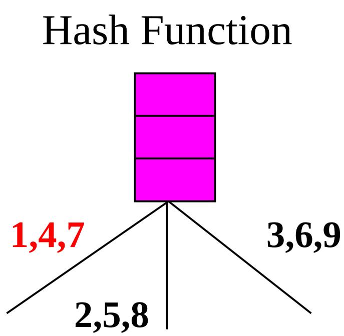
You need:

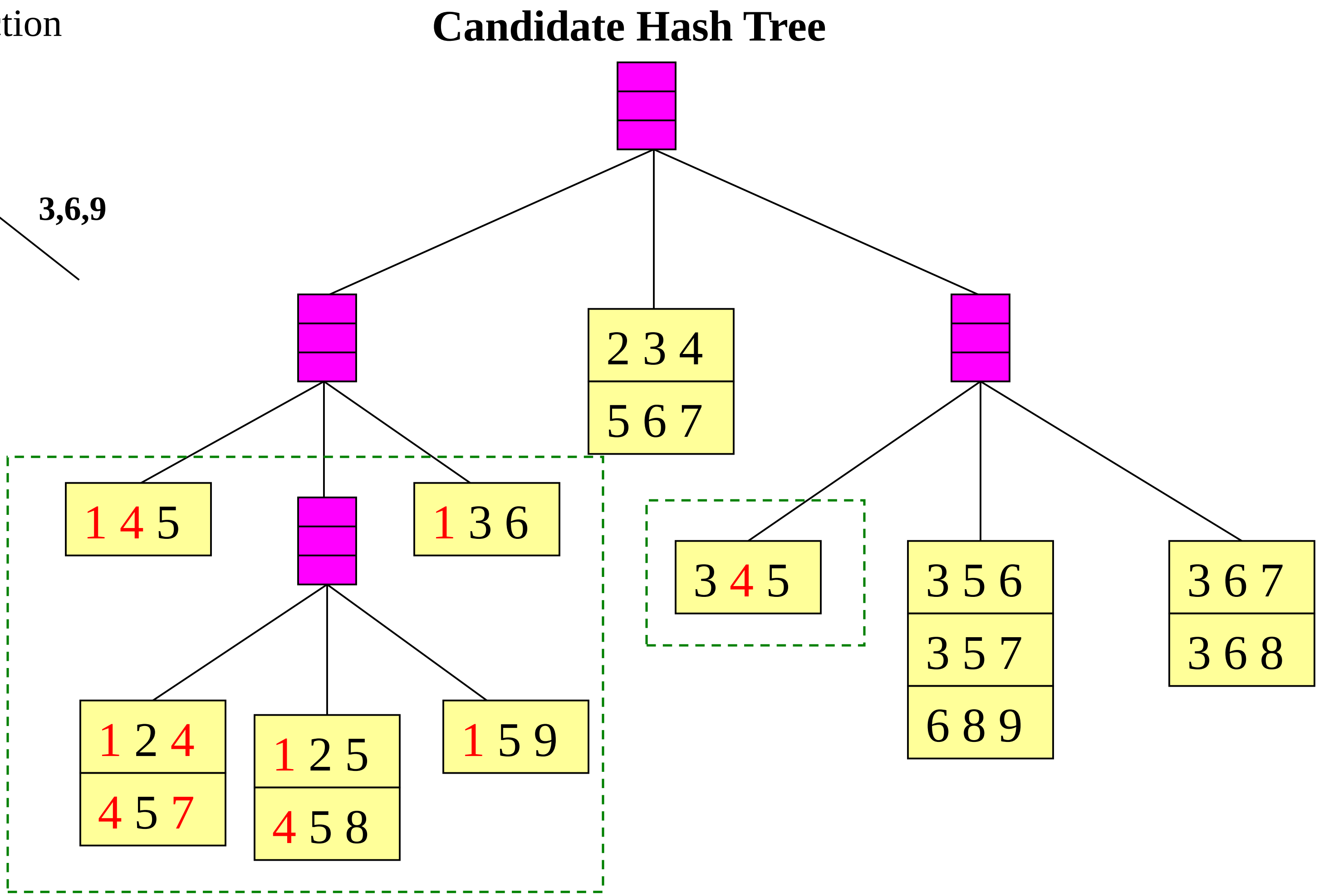
- **Hash function**
- **Max leaf size: max number of itemsets stored in a leaf node (if number of candidate itemsets exceeds max leaf size, split the node)**





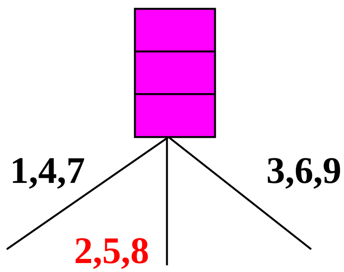
Association Rule Discovery: Hash tree

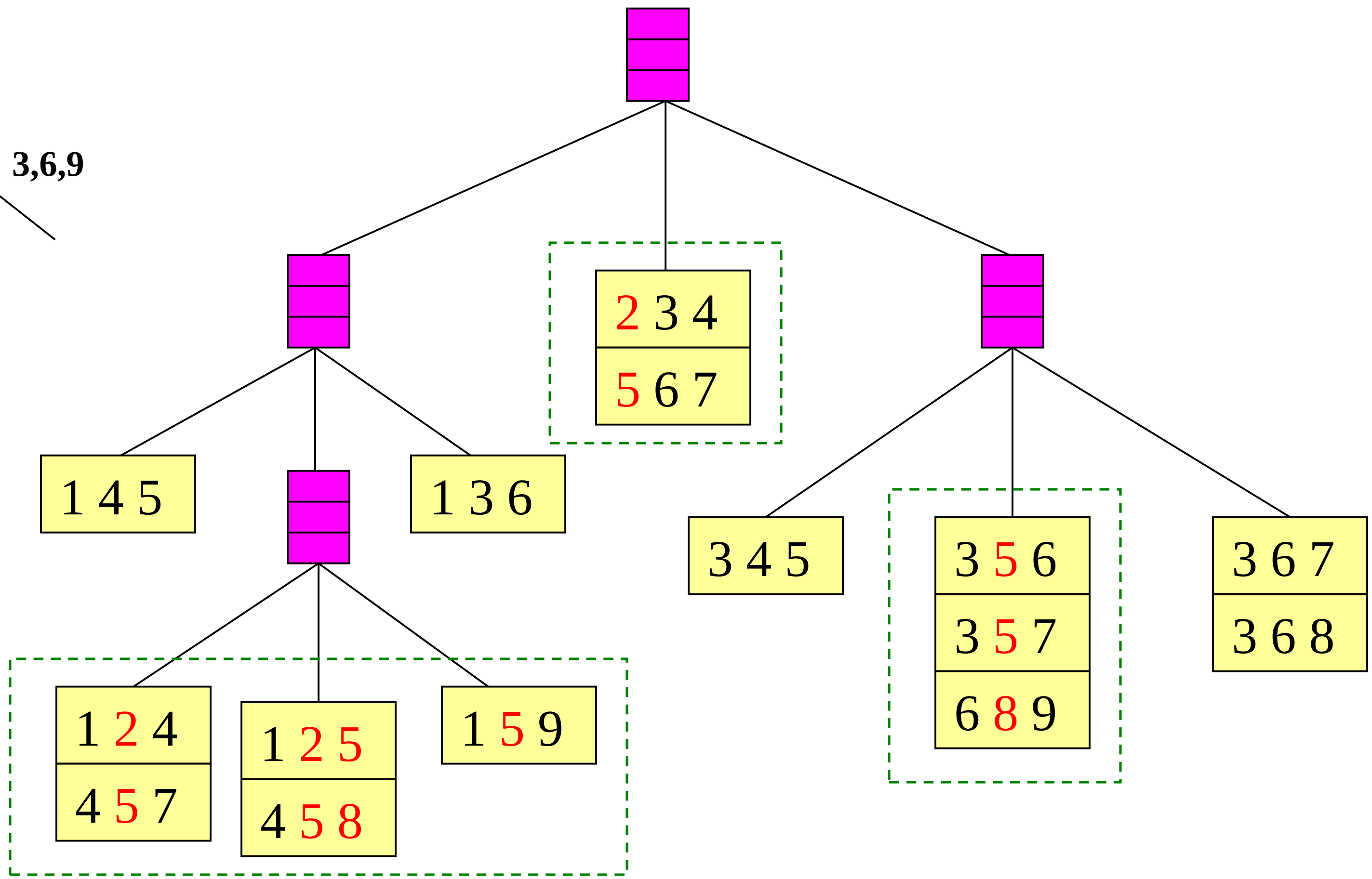




Hash on
1, 4 or 7

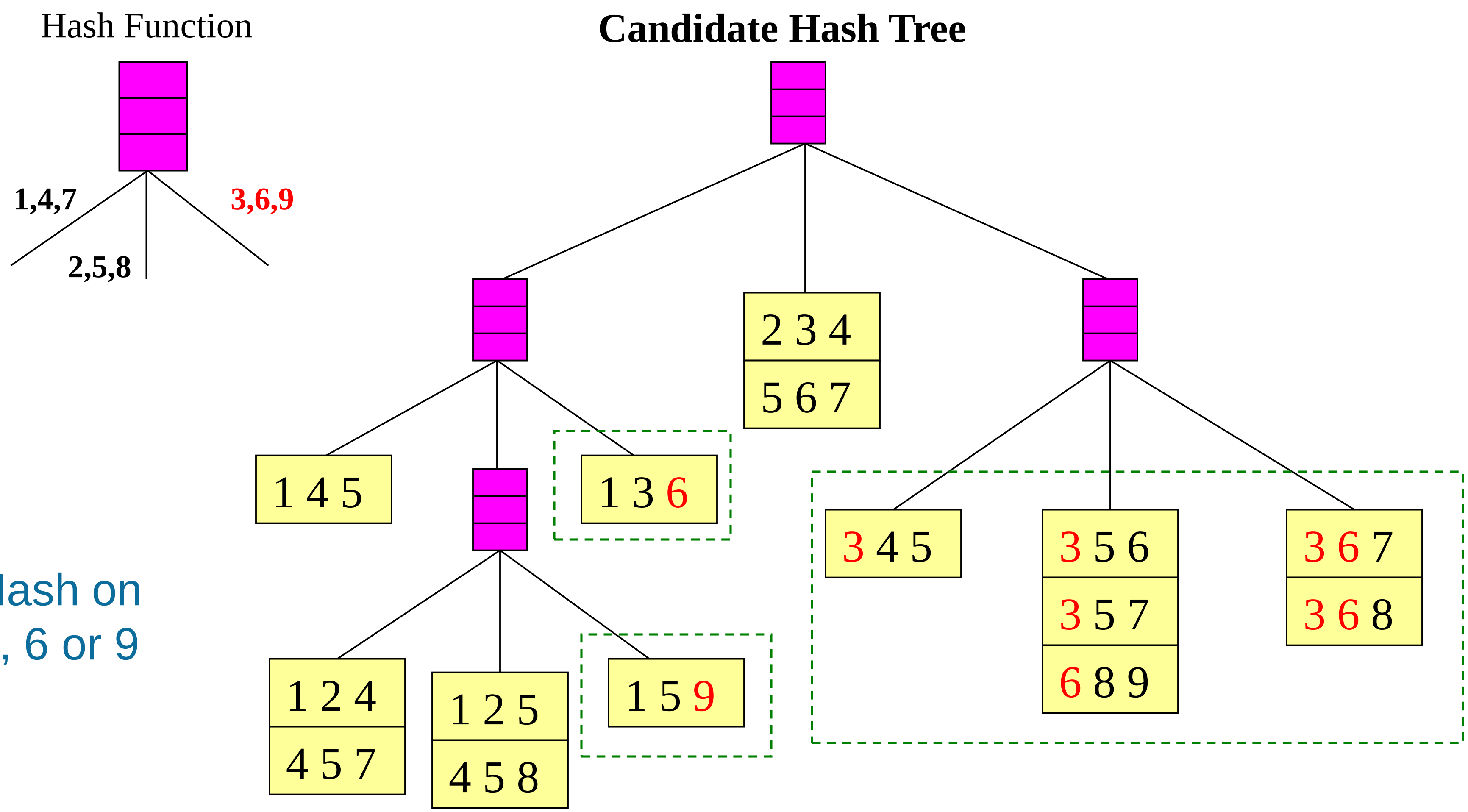
Association Rule Discovery: Hash tree





Hash on
2, 5 or 8

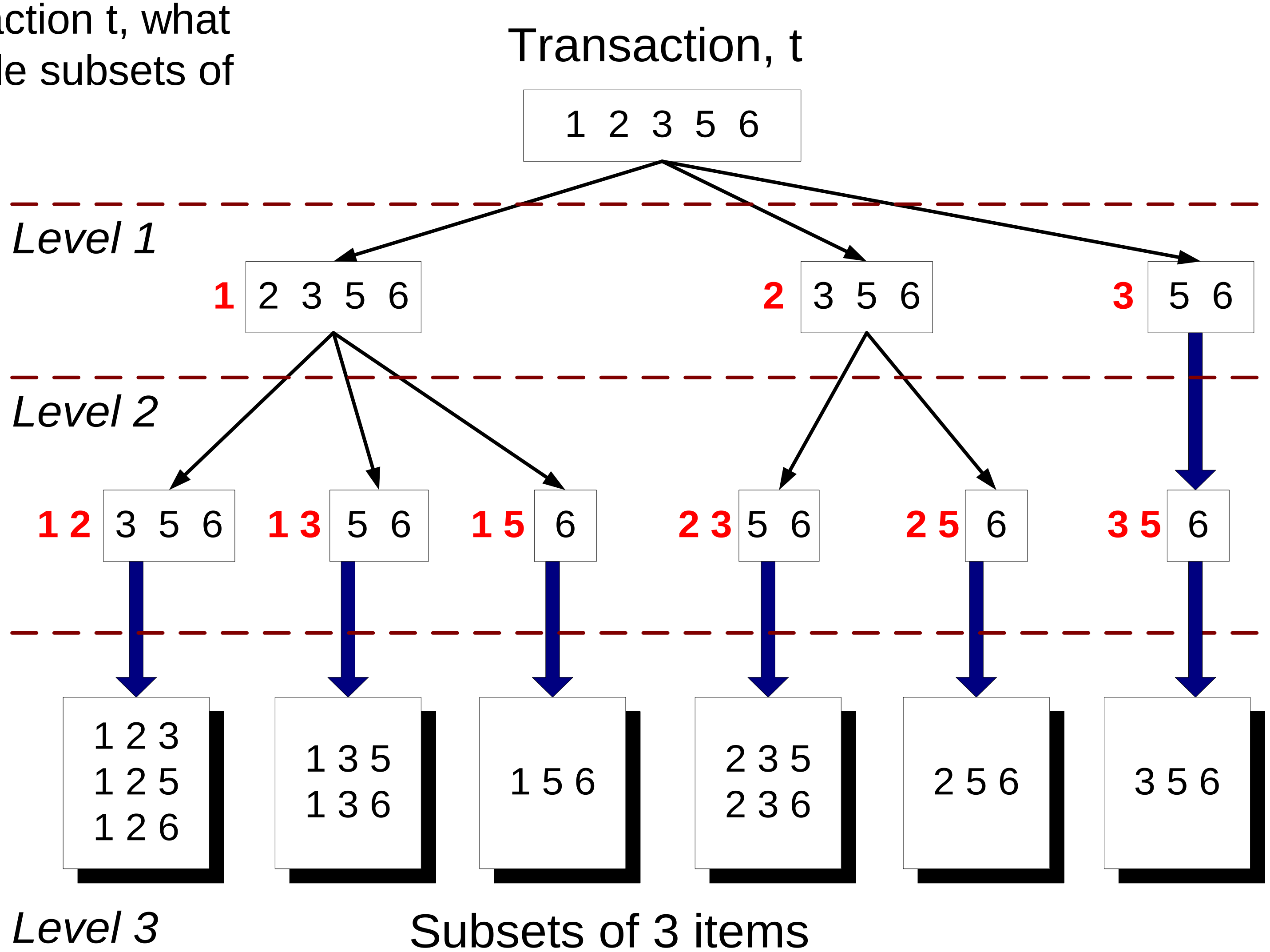
Association Rule Discovery: Hash tree



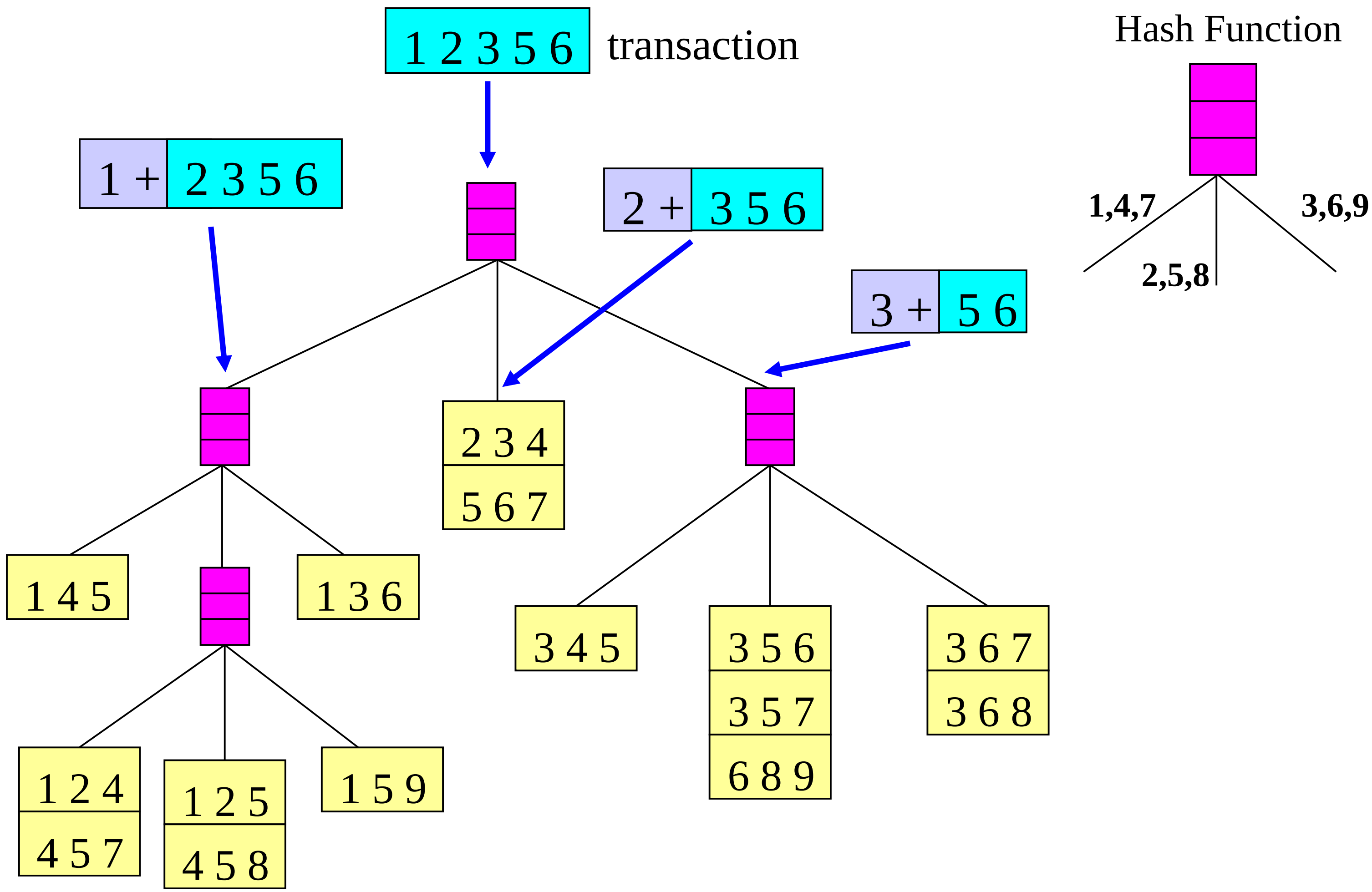
Hash on
3, 6 or 9

Subset Operation

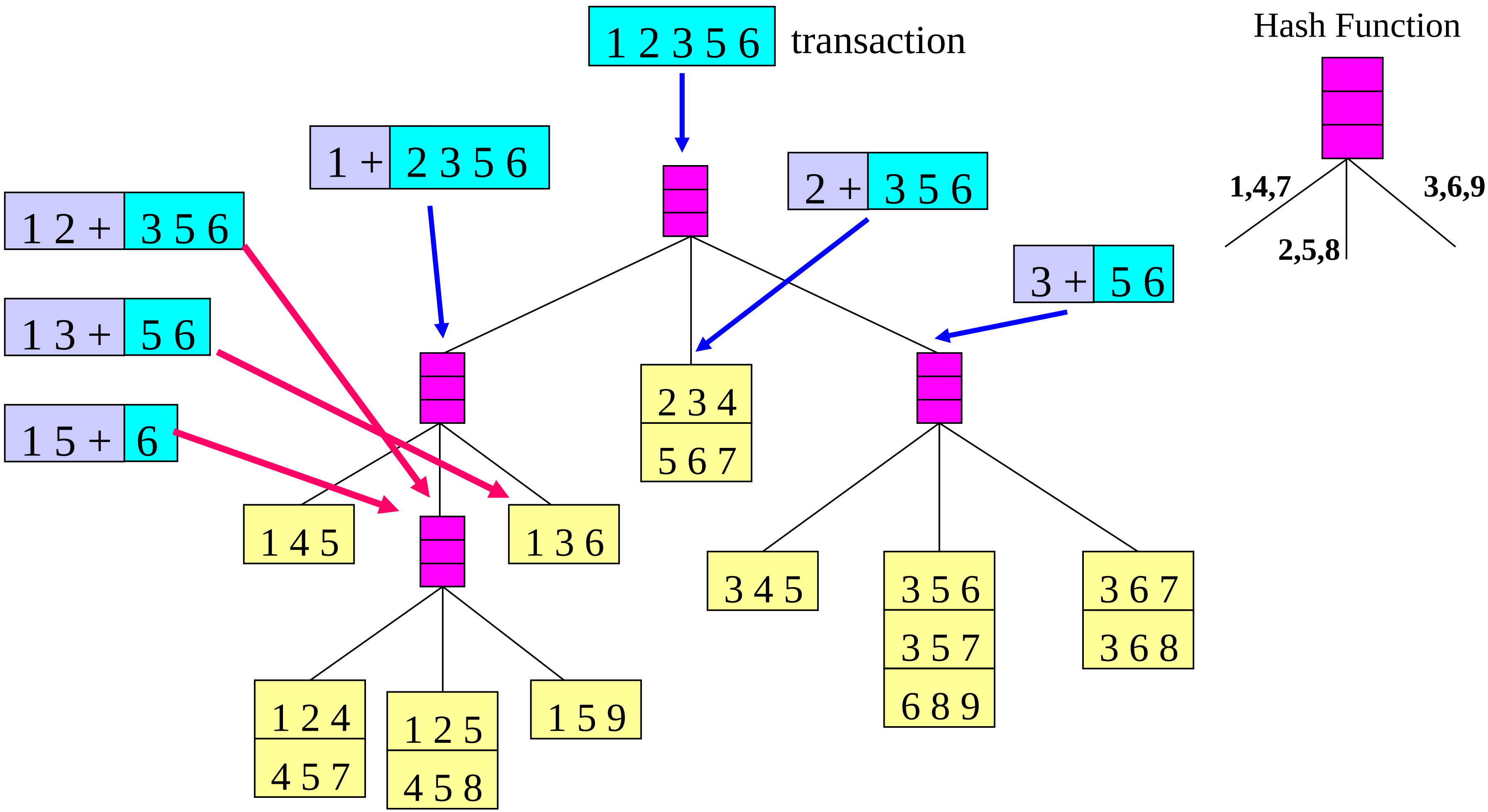
Given a transaction t, what are the possible subsets of size 3?



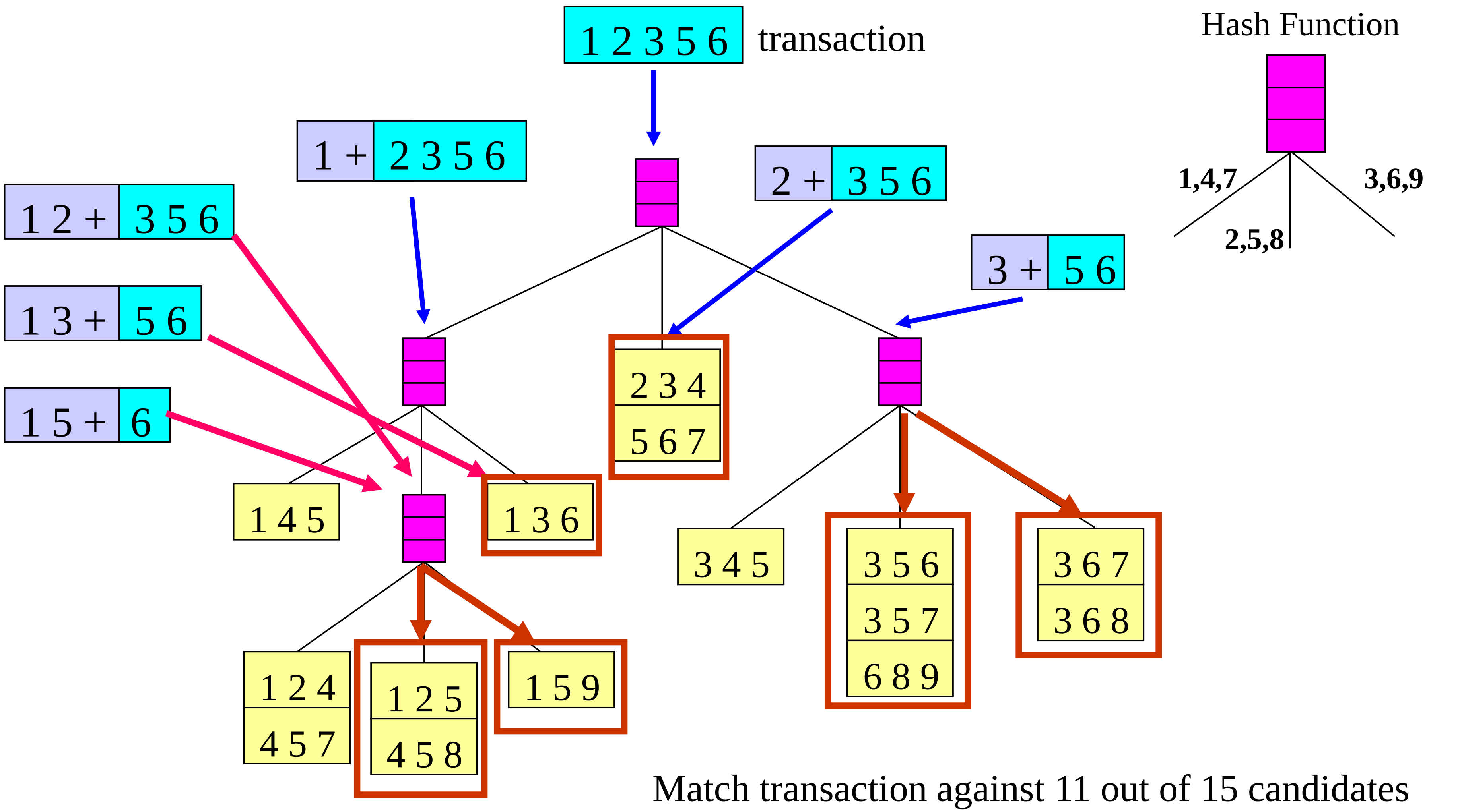
Subset Operation Using Hash Tree



Subset Operation Using Hash Tree



Subset Operation Using Hash Tree



Match transaction against 11 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)

Compact Representation of Frequent Itemsets

- Some itemsets are redundant because they have identical support as their supersets

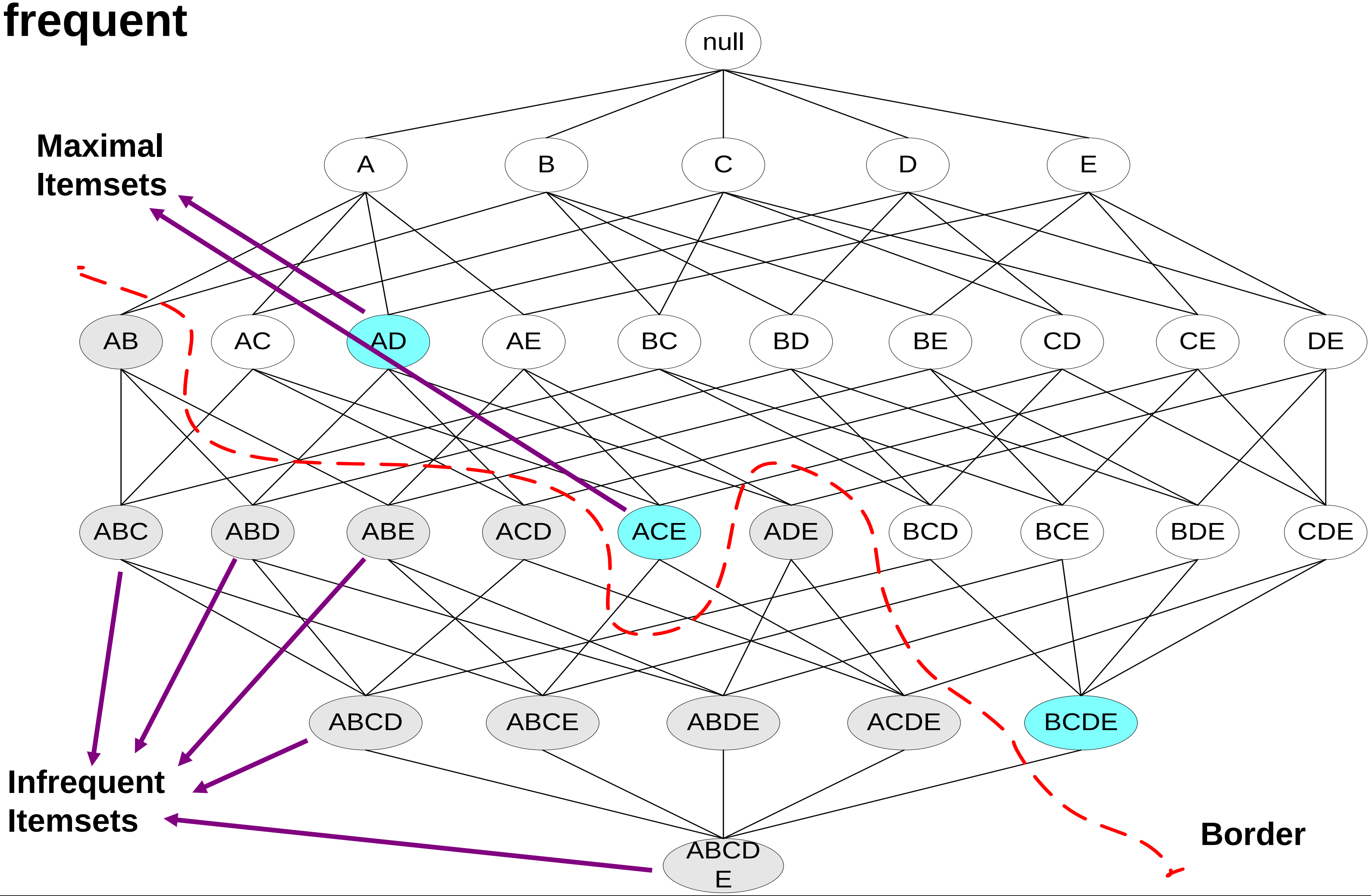
TID	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
1	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

- Number of frequent itemsets $= 3 \times \sum_{k=1}^{10} \binom{10}{k}$

- Need a compact representation

Maximal Frequent Itemset

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



Closed Itemset

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

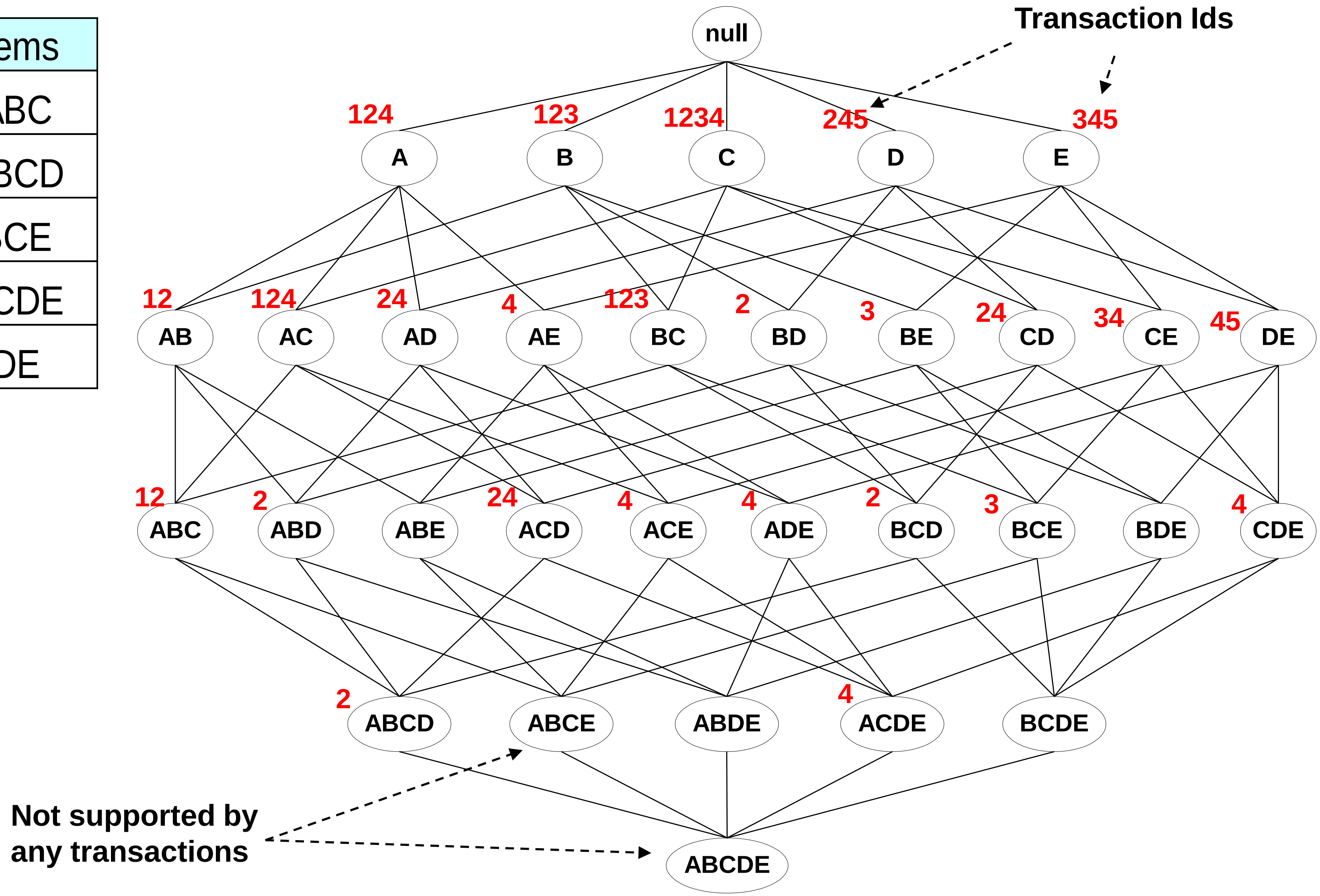
TID	Items
1	{A,B}
2	{B,C,D}
3	{A,B,C,D}
4	{A,B,D}
5	{A,B,C,D}

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

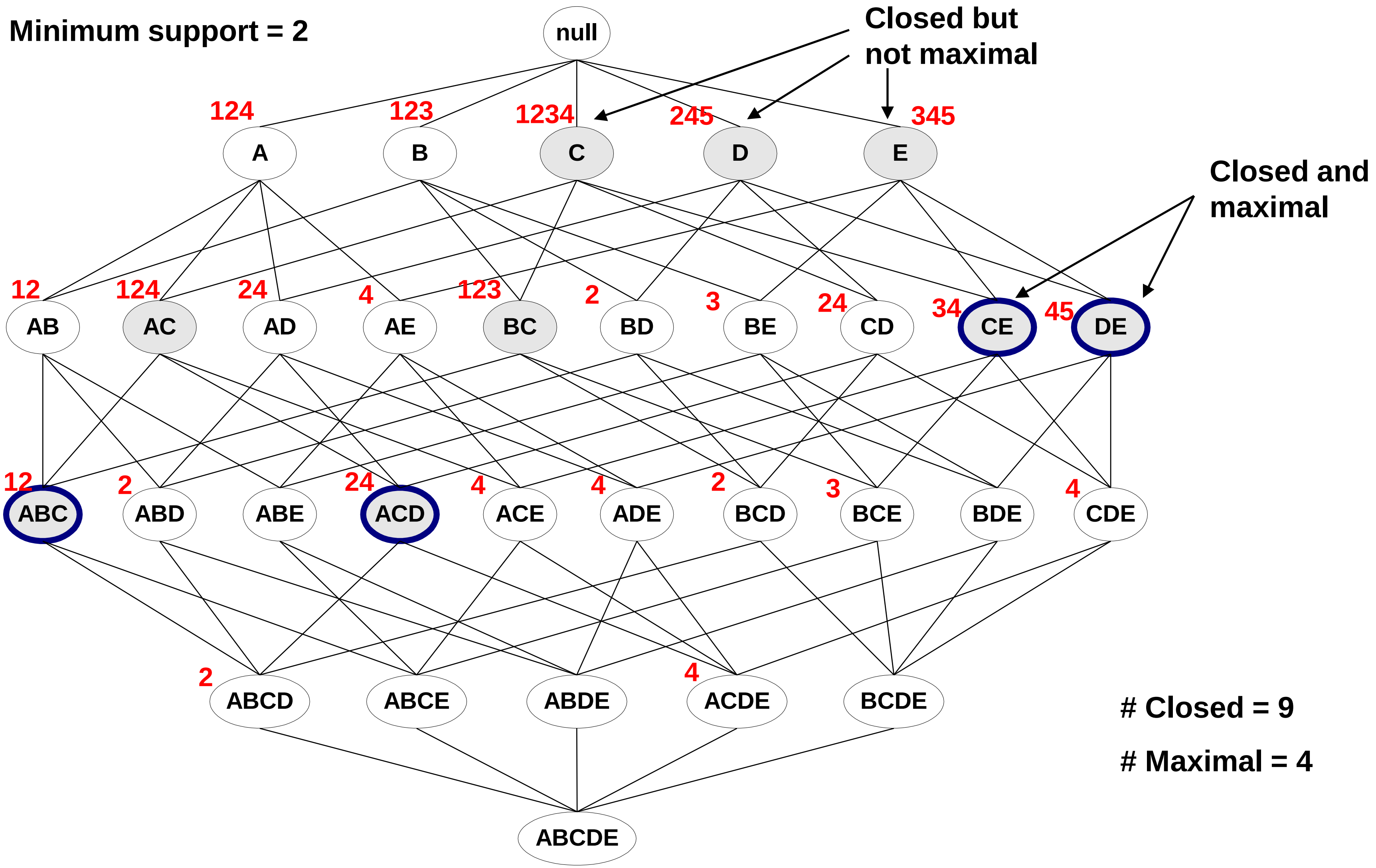
Itemset	Support
{A,B,C}	2
{A,B,D}	3
{A,C,D}	2
{B,C,D}	3
{A,B,C,D}	2

Maximal vs Closed Itemsets

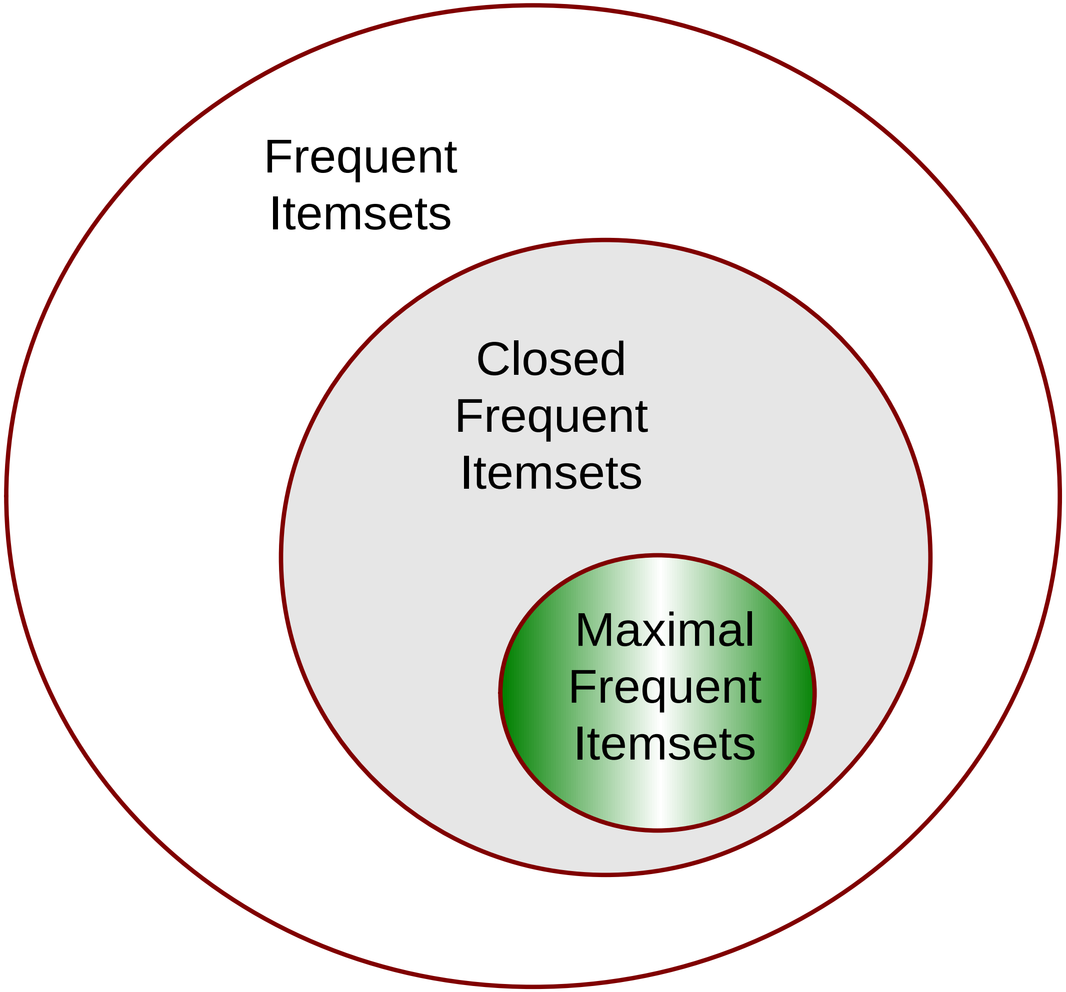
TID	Items
1	ABC
2	ABCD
3	BCE
4	ACDE
5	DE



Maximal vs Closed Frequent Itemsets

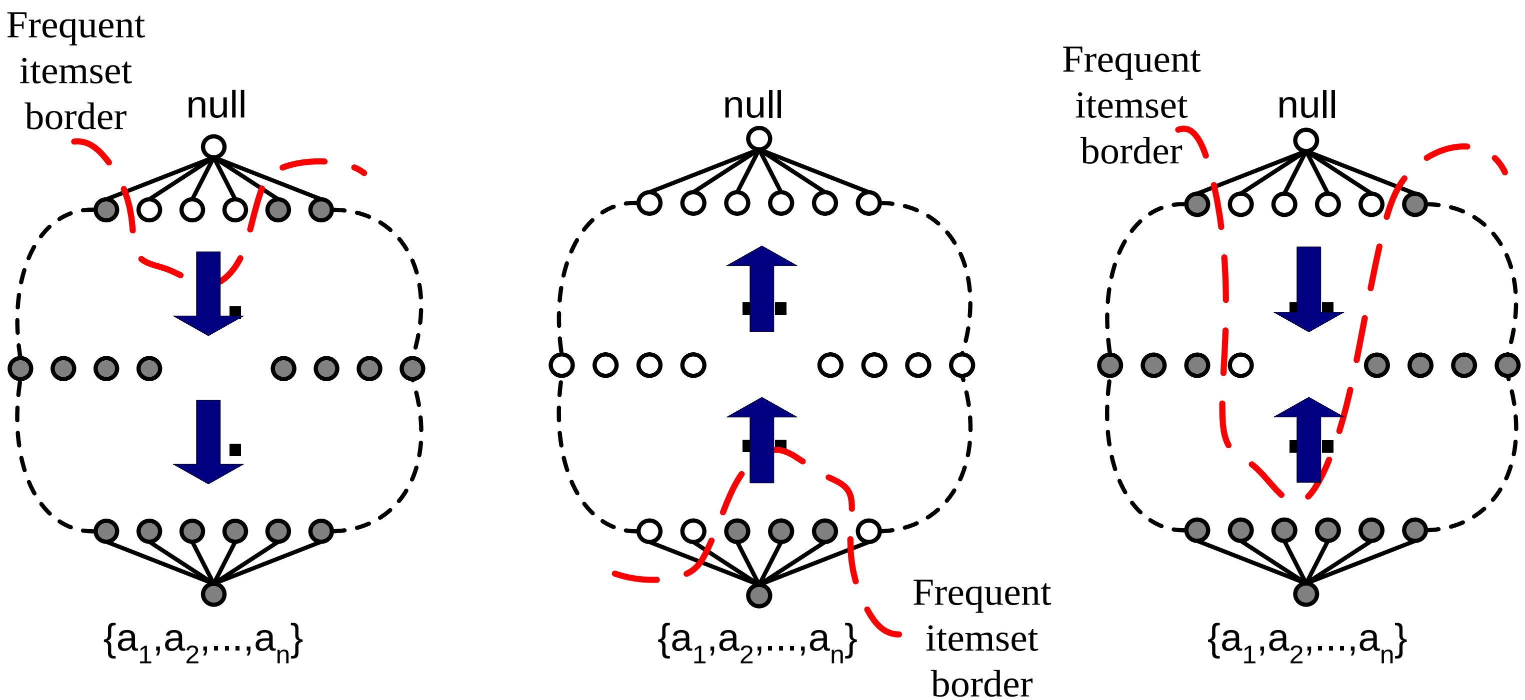


Maximal vs Closed Itemsets



Alternative Methods for Frequent Itemset Generation

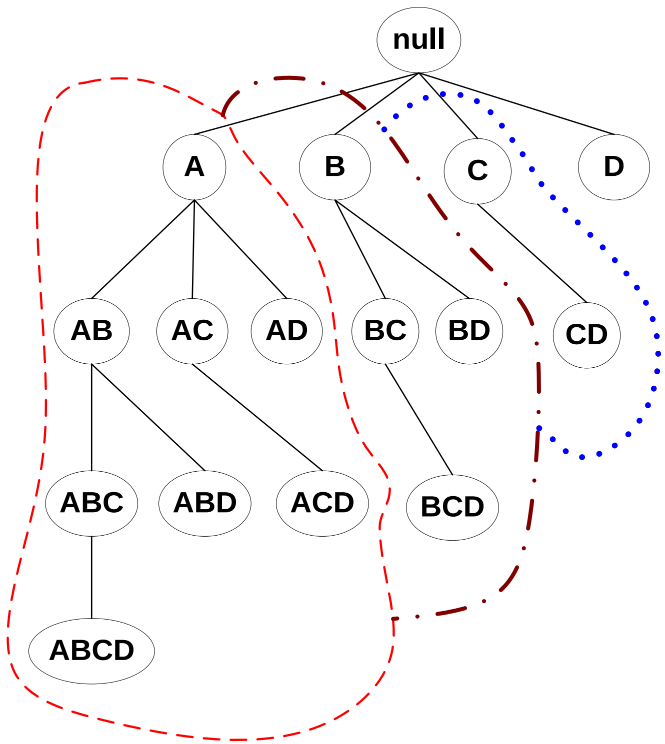
- Traversal of Itemset Lattice
  - General-to-specific vs Specific-to-general



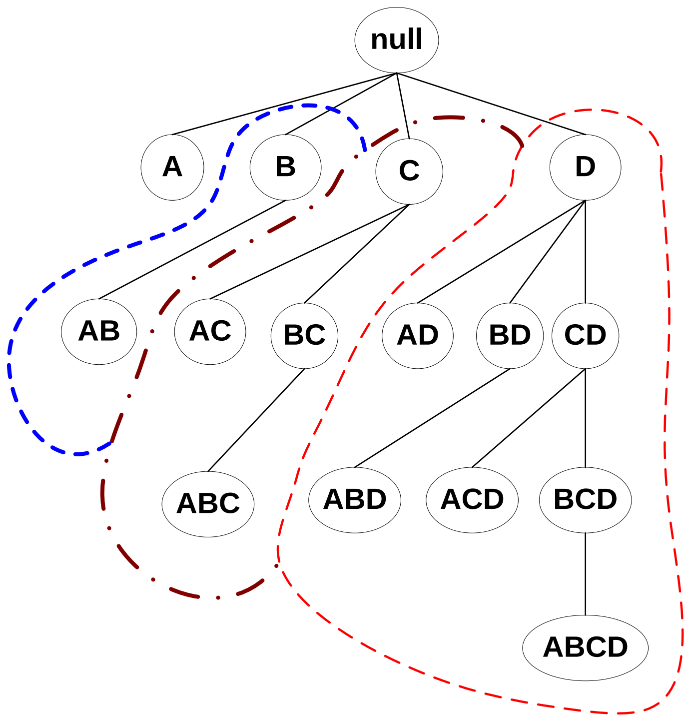
(a) General-to-specific (b) Specific-to-general (c) Bidirectional

Alternative Methods for Frequent Itemset Generation

- Traversal of Itemset Lattice
  - Equivalent Classes



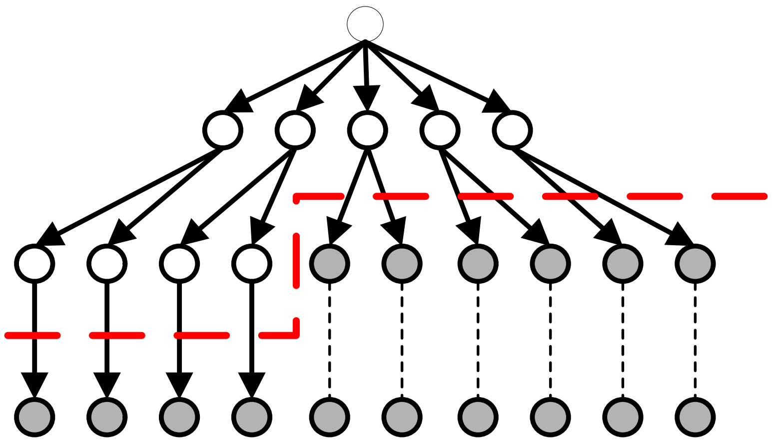
(a) Prefix tree



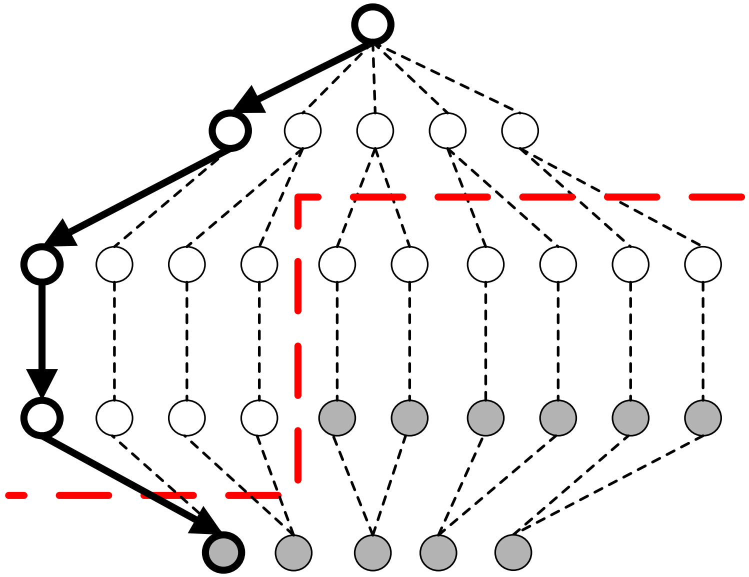
(b) Suffix tree

Alternative Methods for Frequent Itemset Generation

- Traversal of Itemset Lattice
  - Breadth-first vs Depth-first



(a) Breadth first



(b) Depth first

Alternative Methods for Frequent Itemset Generation

- Representation of Database
  - horizontal vs vertical data layout

Horizontal
Data Layout

TID	Items
1	A,B,E
2	B,C,D
3	C,E
4	A,C,D
5	A,B,C,D
6	A,E
7	A,B
8	A,B,C
9	A,C,D
10	B

Vertical Data Layout

A	B	C	D	E
1	1	2	2	1
4	2	3	4	3
5	5	4	5	6
6	7	8	9	
7	8	9		
8	10			
9				

FP-growth Algorithm

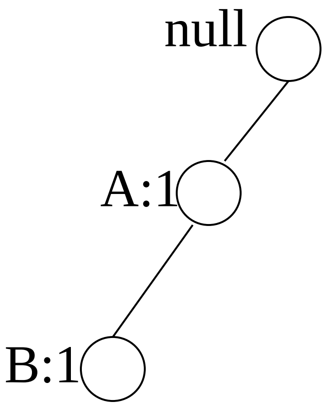
- Use a compressed representation of the database using an FP-tree

- Once an FP-tree has been constructed, it uses a recursive divide-and-conquer approach to mine the frequent itemsets

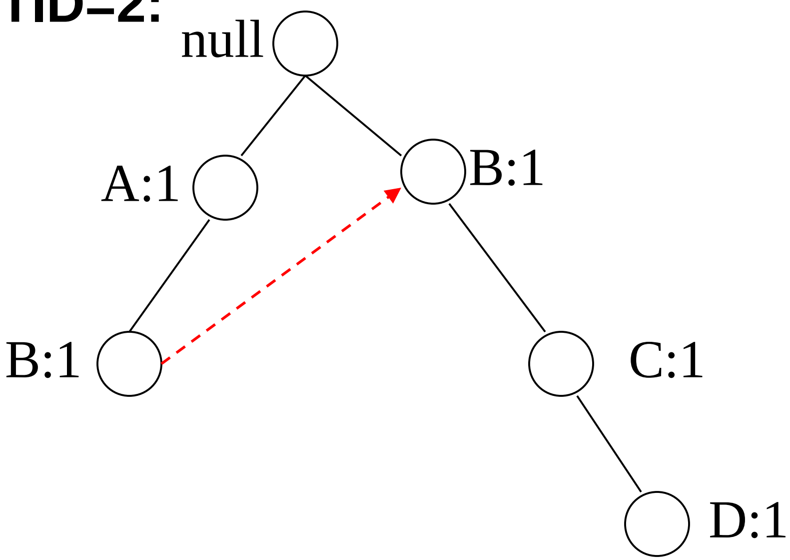
FP-tree construction

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:



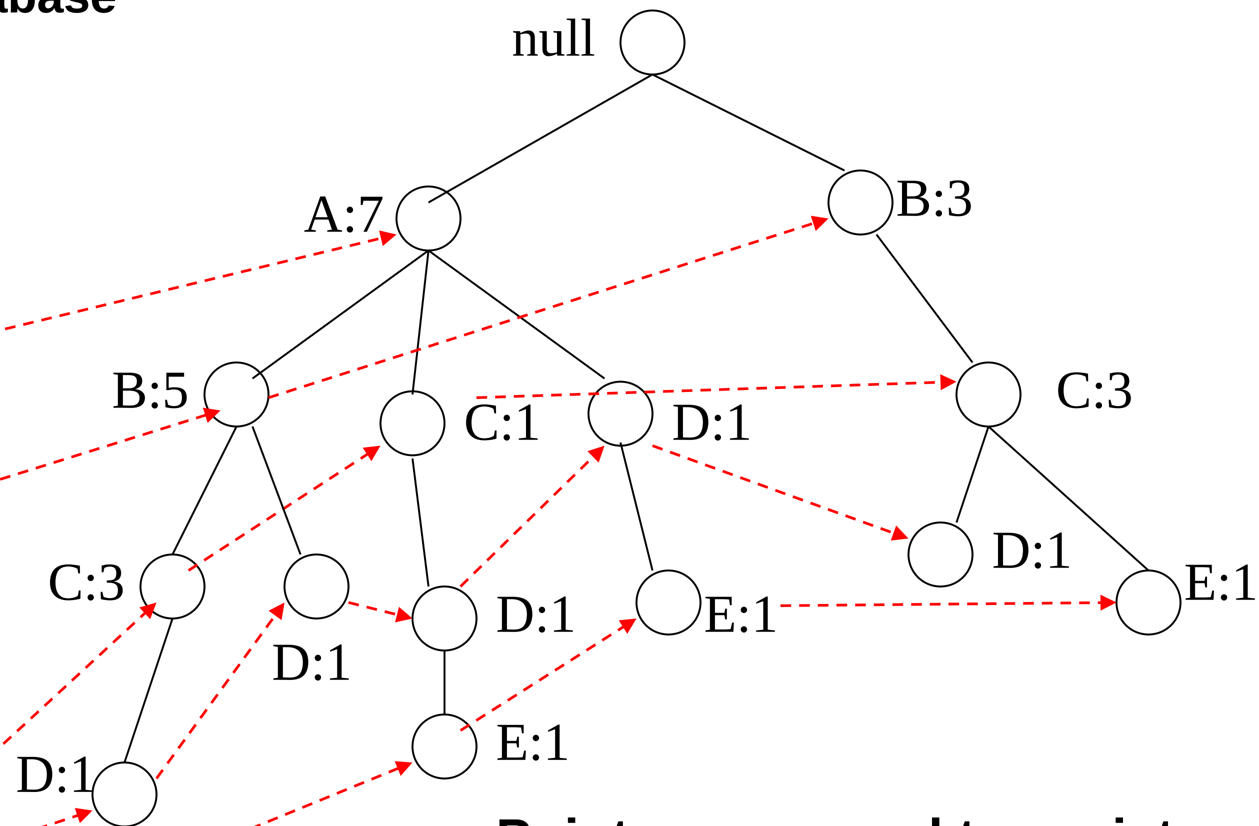
After reading TID=2:



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

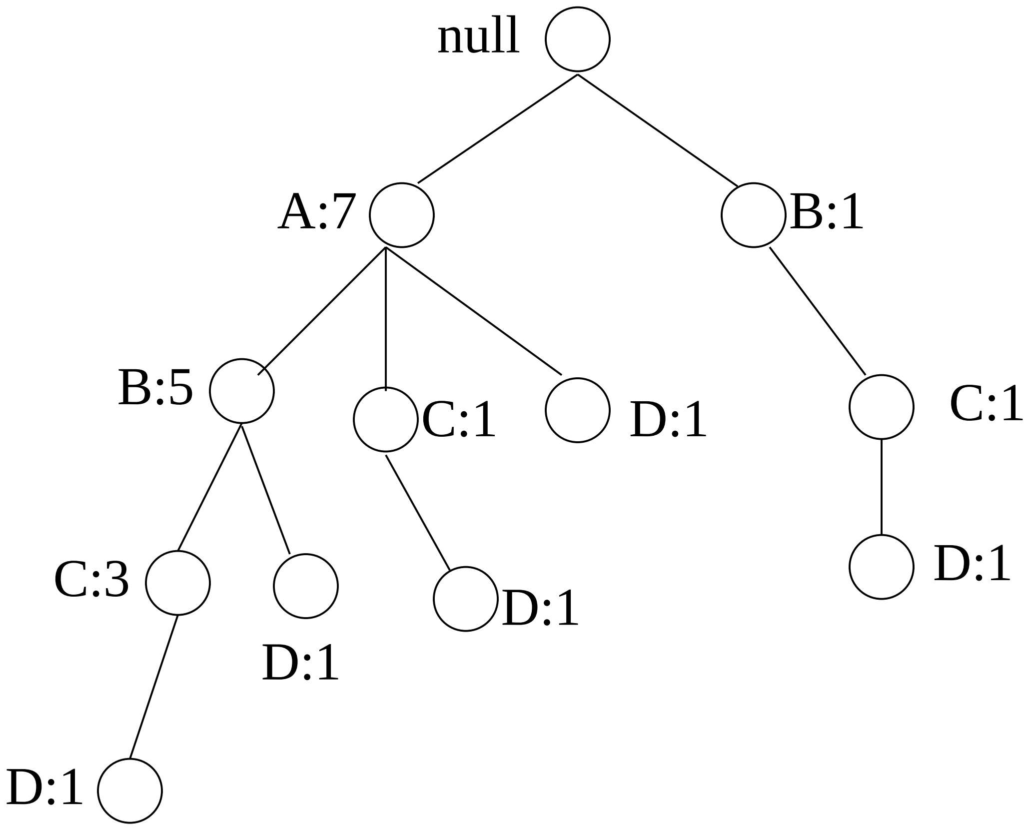


Header table

Item	Pointer
A	
B	
C	
D	
E	

Pointers are used to assist frequent itemset generation

FP-growth



Conditional Pattern base for D:

**P = {(A:1,B:1,C:1),
(A:1,B:1),
(A:1,C:1),
(A:1),
(B:1,C:1)}**

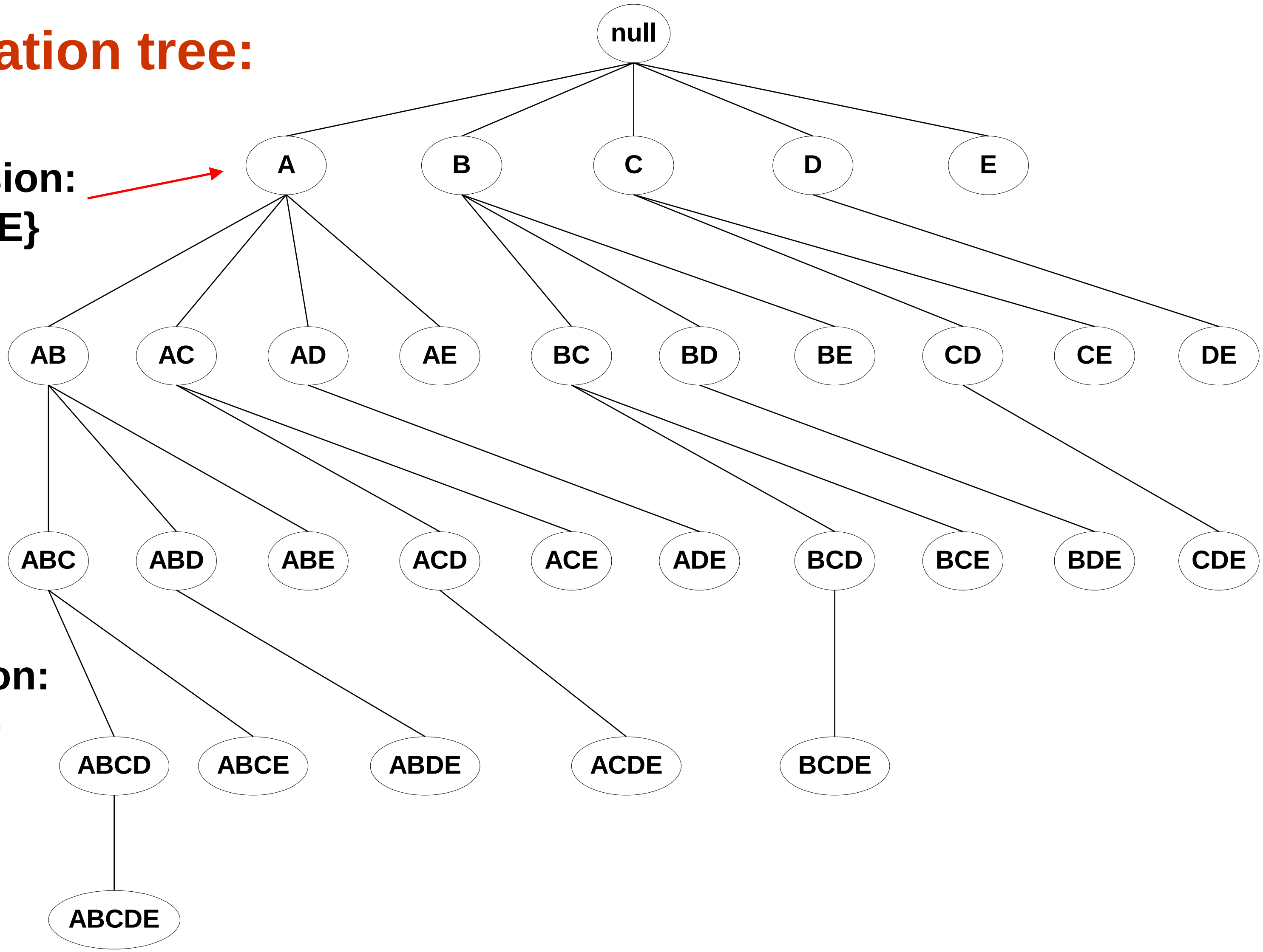
Recursively apply FP-growth on P

Frequent Itemsets found (with sup > 1):
AD, BD, CD, ACD, BCD

Tree Projection

Set enumeration tree:

Possible Extension:
E(A) = {B,C,D,E}



Possible Extension:
E(ABC) = {D,E}

Tree Projection

- Items are listed in lexicographic order
- Each node P stores the following information:
  – Itemset for node P
  – List of possible lexicographic extensions of P: E(P)
  – Pointer to projected database of its ancestor node
  – Bitvector containing information about which transactions in the projected database contain the itemset

Projected Database

Original Database:

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}

Projected Database for node A:

TID	Items
1	{B}
2	{}
3	{C,D,E}
4	{D,E}
5	{B,C}
6	{B,C,D}
7	{}
8	{B,C}
9	{B,D}
10	{}

For each transaction T, projected transaction at node A is $T \cap E(A)$

ECLAT

- For each item, store a list of transaction ids (tids)

Horizontal
Data Layout

TID	Items
1	A,B,E
2	B,C,D
3	C,E
4	A,C,D
5	A,B,C,D
6	A,E
7	A,B
8	A,B,C
9	A,C,D
10	B

Vertical Data Layout

A	B	C	D	E
1	1	2	2	1
4	2	3	4	3
5	5	4	5	6
6	7	8	9	
7	8	9		
8	10			
9				

TID-list

ECLAT

- Determine support of any k-itemset by intersecting tid-lists of two of its (k-1) subsets.

A
1
4
5
6
7
8
9

$\wedge$

B
1
2
5
7
8
10

$\rightarrow$

AB
1
5
7
8

- 3 traversal approaches:
  – top-down, bottom-up and hybrid
- Advantage: very fast support counting
- Disadvantage: intermediate tid-lists may become too large for memory

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: 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

  ⇒ 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} $\rightarrow$ {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

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 $c(AB \rightarrow D)$

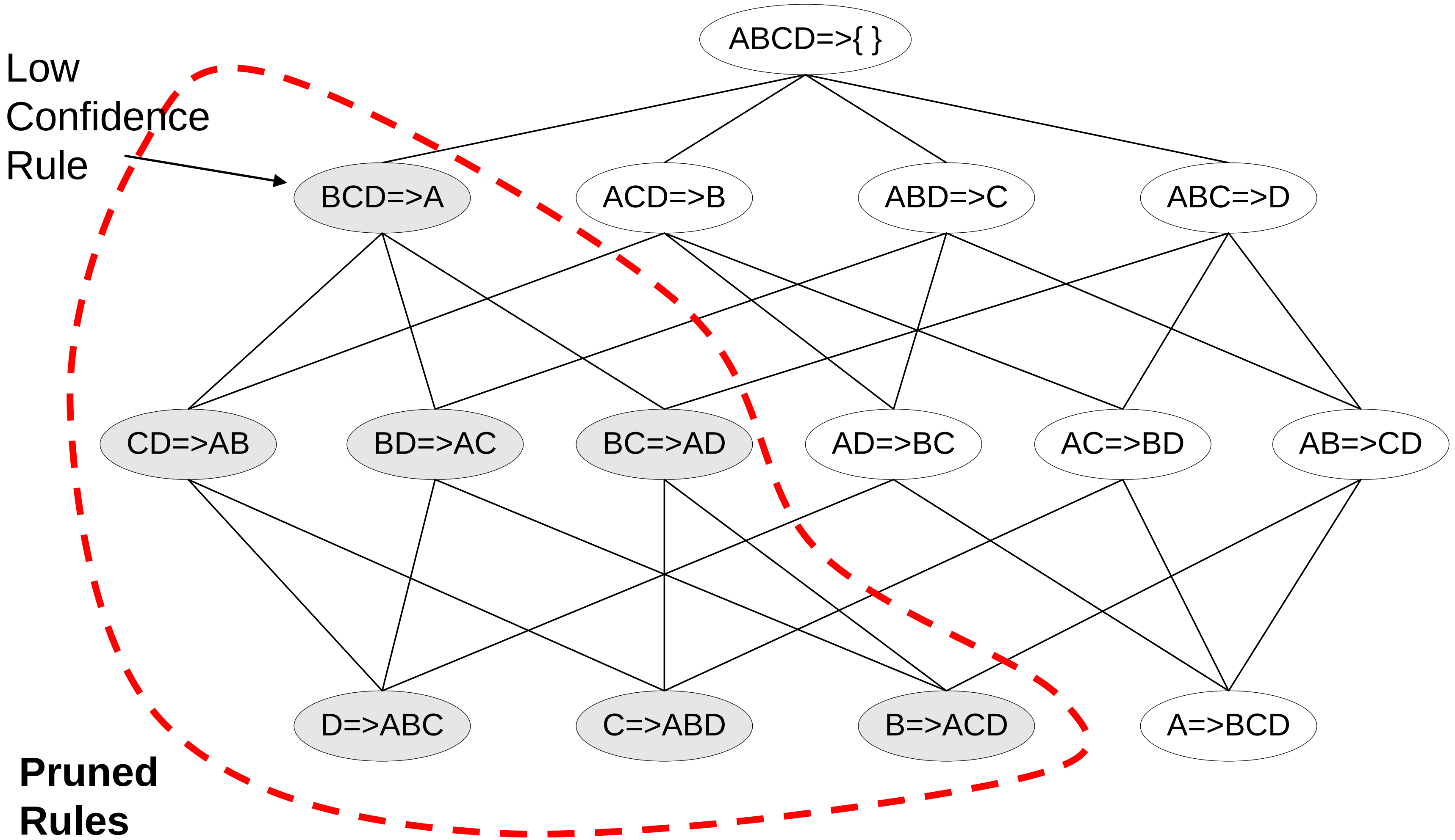
  - But confidence of rules generated from the same itemset has an anti-monotone property
  - e.g., L = {A,B,C,D}:

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

  - Confidence is anti-monotone w.r.t. number of items on the RHS of the rule

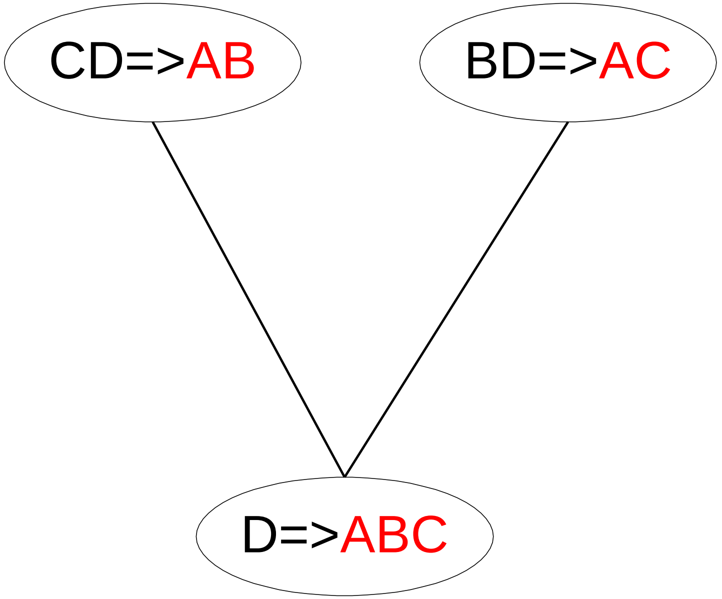
Rule Generation for Apriori Algorithm

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



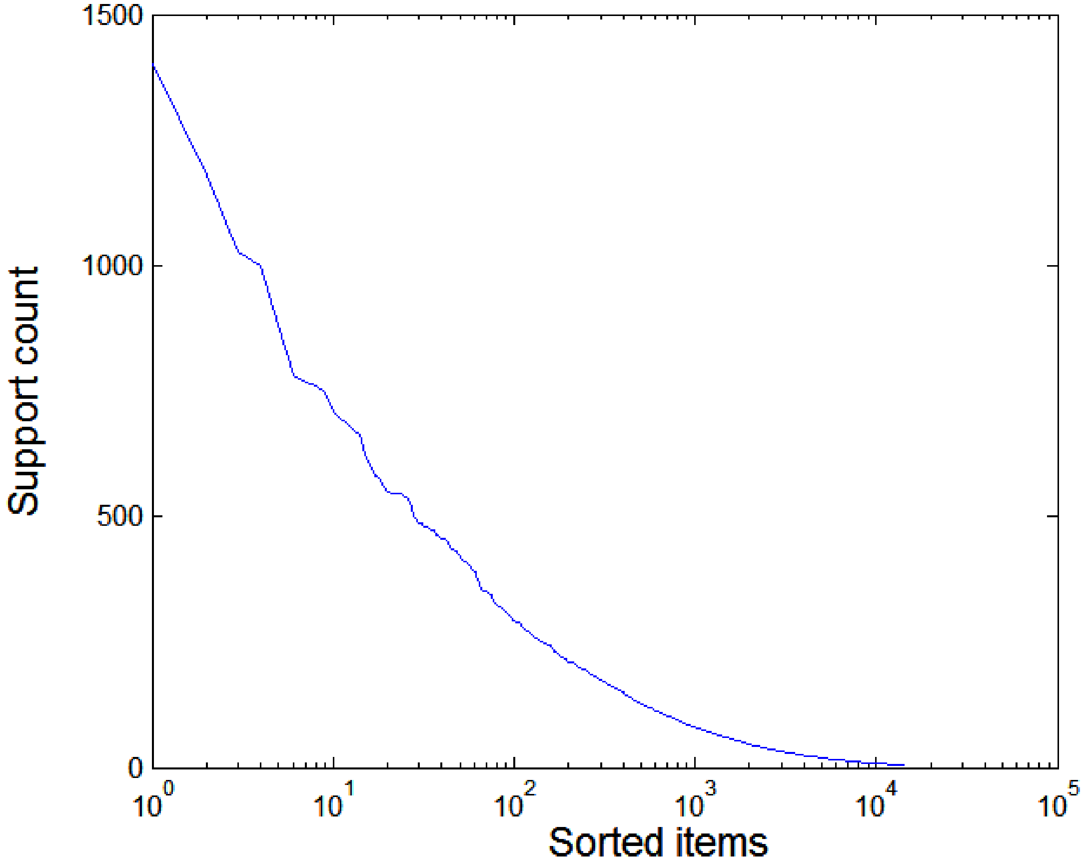
- join(CD=>AB,BD=>AC)
would produce the candidate
rule D => ABC

- Prune rule D=>ABC if its
subset AD=>BC does not have
high confidence

Effect of Support Distribution

- Many real data sets have skewed support distribution

Support distribution of a retail data set



Effect of Support Distribution

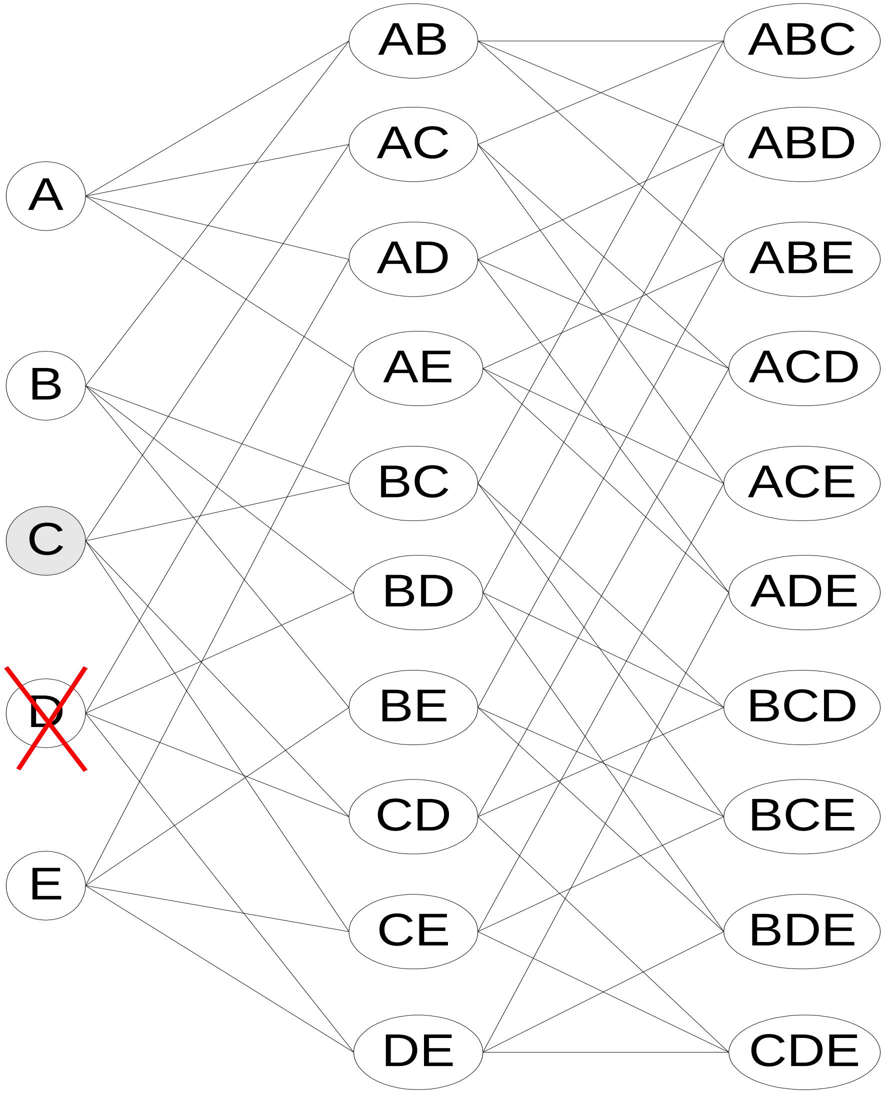
- How to set the appropriate *minsup* threshold?
  - If *minsup* is set too high, we could miss itemsets involving interesting rare items (e.g., expensive products)
  - If *minsup* is set too low, it is computationally expensive and the number of itemsets is very large
- Using a single minimum support threshold may not be effective

Multiple Minimum Support

- How to apply multiple minimum supports?
  - MS(i): minimum support for item i
  - e.g.: MS(Milk)=5%, MS(Coke) = 3%, MS(Broccoli)=0.1%, MS(Salmon)=0.5%
  - MS({Milk, Broccoli}) = min (MS(Milk), MS(Broccoli)) = 0.1%
  - Challenge: Support is no longer anti-monotone
    - Suppose: Support(Milk, Coke) = 1.5% and Support(Milk, Coke, Broccoli) = 0.5%
    - {Milk,Coke} is infrequent but {Milk,Coke,Broccoli} is frequent

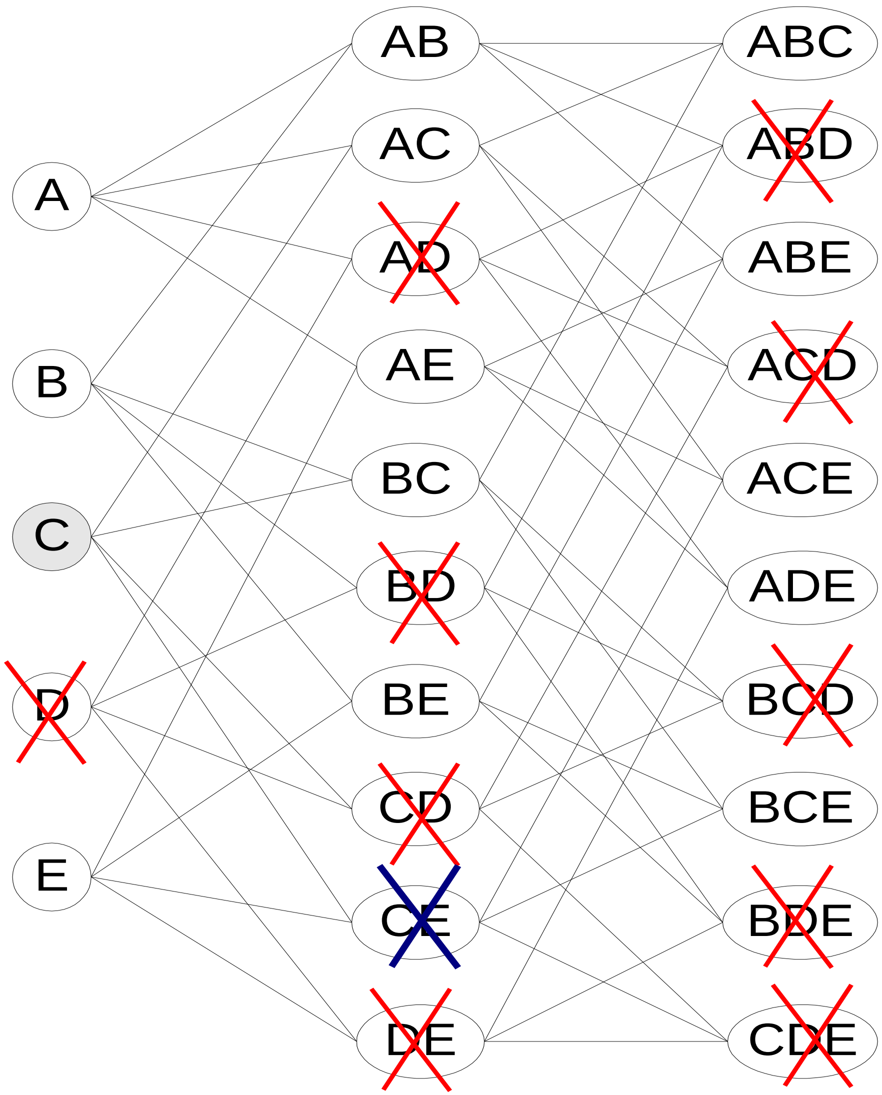
Multiple Minimum Support

Item	MS(I)	Sup(I)
A	0.10%	0.25%
B	0.20%	0.26%
C	0.30%	0.29%
D	0.50%	0.05%
E	3%	4.20%



Multiple Minimum Support

Item	MS(I)	Sup(I)
A	0.10%	0.25%
B	0.20%	0.26%
C	0.30%	0.29%
D	0.50%	0.05%
E	3%	4.20%



Multiple Minimum Support (Liu 1999)

- Order the items according to their minimum support (in ascending order)
  - e.g.: MS(Milk)=5%, MS(Coke) = 3%, MS(Broccoli)=0.1%, MS(Salmon)=0.5%
  - Ordering: Broccoli, Salmon, Coke, Milk

- Need to modify Apriori such that:
  - L_1 : set of frequent items
  - F_1 : set of items whose support is $\geq$ MS(1) where MS(1) is $\min_i$(MS(i))
  - C_2 : candidate itemsets of size 2 is generated from F_1 instead of L_1

Multiple Minimum Support (Liu 1999)

- Modifications to Apriori:
  - In traditional Apriori,
    - A candidate (k+1)-itemset is generated by merging two frequent itemsets of size k
    - The candidate is pruned if it contains any infrequent subsets of size k
  - Pruning step has to be modified:
    - Prune only if subset contains the first item
    - e.g.: Candidate={Broccoli, Coke, Milk} (ordered according to minimum support)
    - {Broccoli, Coke} and {Broccoli, Milk} are frequent but {Coke, Milk} is infrequent
      - Candidate is not pruned because {Coke,Milk} does not contain the first item, i.e., Broccoli.

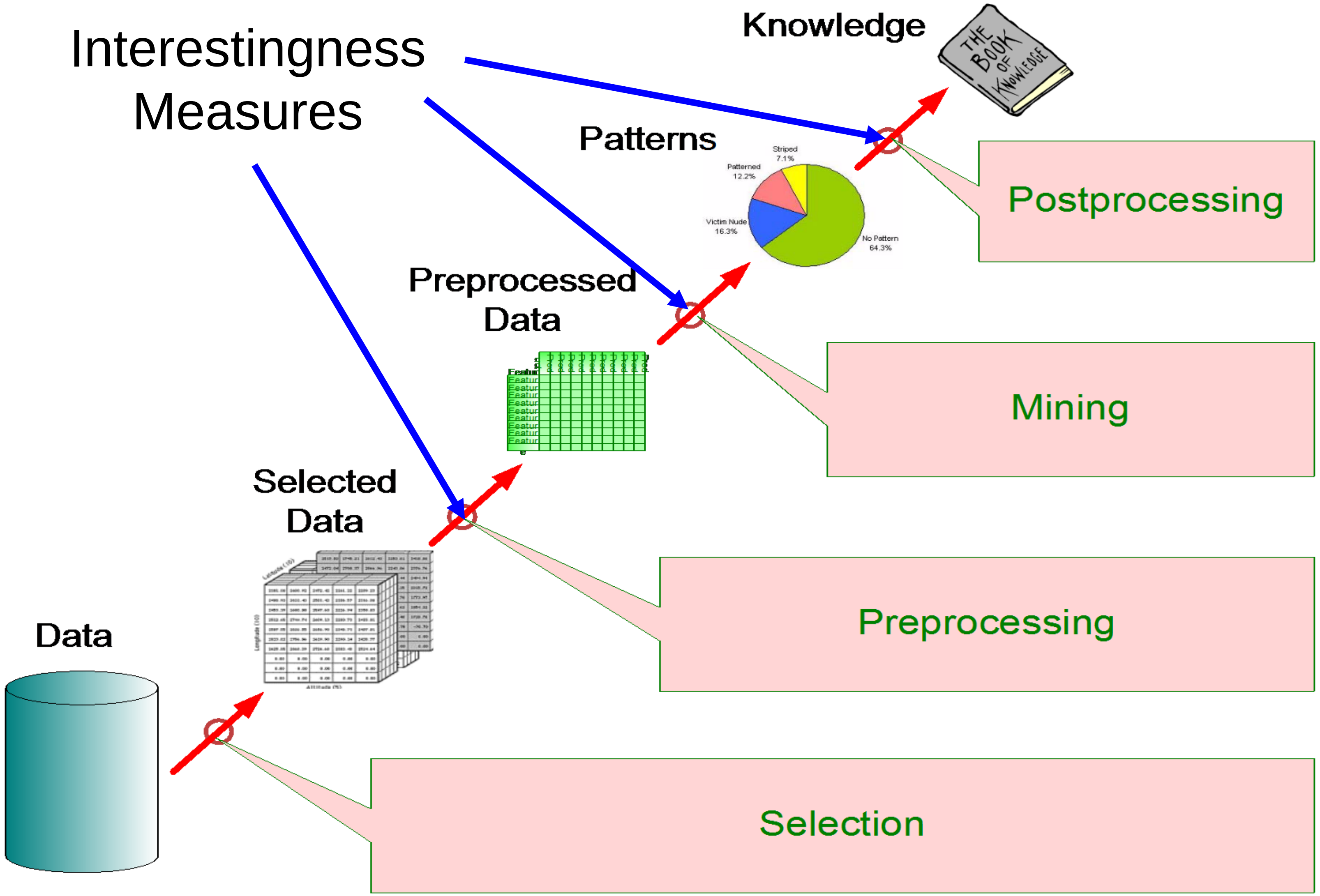
Pattern Evaluation

- Association rule algorithms tend to produce too many rules
  - many of them are uninteresting or redundant
  - Redundant if $\{A,B,C\} \rightarrow \{D\}$ and $\{A,B\} \rightarrow \{D\}$ have same support & confidence

- Interestingness measures can be used to prune/rank the derived patterns

- In the original formulation of association rules, support & confidence are the only measures used

Application of Interestingness Measure



Computing Interestingness Measure

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

Contingency table for $X \rightarrow Y$

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

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, lift, Gini, J-measure, etc.

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) = 0.75

but P(Coffee) = 0.9

$\Rightarrow$ Although confidence is high, rule is misleading

$\Rightarrow$ $P(\text{Coffee}|\overline{\text{Tea}}) = 0.9375$

Statistical Independence

- Population of 1000 students
  - 600 students know how to swim (S)
  - 700 students know how to bike (B)
  - 420 students know how to swim and bike (S,B)

  - $P(S \wedge B) = 420/1000 = 0.42$
  - $P(S) \times P(B) = 0.6 \times 0.7 = 0.42$

  - $P(S \wedge B) = P(S) \times P(B)$ => Statistical independence
  - $P(S \wedge B) > P(S) \times P(B)$ => Positively correlated
  - $P(S \wedge B) < P(S) \times P(B)$ => Negatively correlated

Statistical-based Measures

- Measures that take into account statistical dependence

$$Lift = \frac{P(Y \mid X)}{P(Y)}$$

$$Interest = \frac{P(X,Y)}{P(X)P(Y)}$$

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

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

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)

Drawback of Lift & 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

There are lots of measures proposed in the literature

Some measures are good for certain applications, but not for others

What criteria should we use to determine whether a measure is good or bad?

What about Apriori-style support based pruning? How does it affect these measures?

#	Measure	Formula
1	ϕ-coefficient	$\frac{P(A,B)-P(A)P(B)}{\sqrt{P(A)P(B)(1-P(A))(1-P(B))}}$
2	Goodman-Kruskal's (λ)	$\frac{\sum_j \max_k P(A_j,B_k)+\sum_k \max_j P(A_j,B_k)-\max_j P(A_j)-\max_k P(B_k)}{2-\max_j P(A_j)-\max_k P(B_k)}$
3	Odds ratio (α)	$\frac{P(A,B)P(\overline{A},\overline{B})}{P(A,\overline{B})P(\overline{A},B)}$
4	Yule's Q	$\frac{P(A,B)P(\overline{AB})-P(A,\overline{B})P(\overline{A},B)}{P(A,B)P(\overline{AB})+P(A,\overline{B})P(\overline{A},B)} = \frac{\alpha-1}{\alpha+1}$
5	Yule's Y	$\frac{\sqrt{P(A,B)P(\overline{AB})}-\sqrt{P(A,\overline{B})P(\overline{A},B)}}{\sqrt{P(A,B)P(\overline{AB})}+\sqrt{P(A,\overline{B})P(\overline{A},B)}} = \frac{\sqrt{\alpha}-1}{\sqrt{\alpha}+1}$
6	Kappa (κ)	$\frac{P(A,B)+P(\overline{A},\overline{B})-P(A)P(B)-P(\overline{A})P(\overline{B})}{1-P(A)P(B)-P(\overline{A})P(\overline{B})}$
7	Mutual Information (M)	$\frac{\sum_i \sum_j P(A_i,B_j)\log\frac{P(A_i,B_j)}{P(A_i)P(B_j)}}{\min(-\sum_i P(A_i)\log P(A_i), -\sum_j P(B_j)\log P(B_j))}$
8	J-Measure (J)	$\max\Big(P(A,B)\log(\frac{P(B\mid A)}{P(B)})+P(A\overline{B})\log(\frac{P(\overline{B}\mid A)}{P(\overline{B})}),$ $P(A,B)\log(\frac{P(A\mid B)}{P(A)})+P(\overline{A}B)\log(\frac{P(\overline{A}\mid B)}{P(\overline{A})})\Big)$
9	Gini index (G)	$\max\Big(P(A)[P(B\mid A)^2+P(\overline{B}\mid A)^2]+P(\overline{A})[P(B\mid \overline{A})^2+P(\overline{B}\mid \overline{A})^2]$ $-P(B)^2-P(\overline{B})^2,$ $P(B)[P(A\mid B)^2+P(\overline{A}\mid B)^2]+P(\overline{B})[P(A\mid \overline{B})^2+P(\overline{A}\mid \overline{B})^2]$ $-P(A)^2-P(\overline{A})^2\Big)$
10	Support (s)	$P(A,B)$
11	Confidence (c)	$\max(P(B\mid A), P(A\mid B))$
12	Laplace (L)	$\max\Big(\frac{NP(A,B)+1}{NP(A)+2}, \frac{NP(A,B)+1}{NP(B)+2}\Big)$
13	Conviction (V)	$\max\Big(\frac{P(A)P(\overline{B})}{P(A\overline{B})}, \frac{P(B)P(\overline{A})}{P(B\overline{A})}\Big)$
14	Interest (I)	$\frac{P(A,B)}{P(A)P(B)}$
15	cosine (IS)	$\frac{P(A,B)}{\sqrt{P(A)P(B)}}$
16	Piatetsky-Shapiro's (PS)	$P(A,B)-P(A)P(B)$
17	Certainty factor (F)	$\max\Big(\frac{P(B\mid A)-P(B)}{1-P(B)}, \frac{P(A\mid B)-P(A)}{1-P(A)}\Big)$
18	Added Value (AV)	$\max(P(B\mid A)-P(B), P(A\mid B)-P(A))$
19	Collective strength (S)	$\frac{P(A,B)+P(\overline{AB})}{P(A)P(B)+P(\overline{A})P(\overline{B})} \times \frac{1-P(A)P(B)-P(\overline{A})P(\overline{B})}{1-P(A,B)-P(\overline{AB})}$
20	Jaccard (ζ)	$\frac{P(A,B)}{P(A)+P(B)-P(A,B)}$
21	Klosgen (K)	$\sqrt{P(A,B)}\max(P(B\mid A)-P(B), P(A\mid B)-P(A))$

Properties of A Good Measure

- Piatetsky-Shapiro:
  3 properties a good measure M must satisfy:
  - M(A,B) = 0 if A and B are statistically independent
  - M(A,B) increase monotonically with P(A,B) when P(A) and P(B) remain unchanged
  - M(A,B) decreases monotonically with P(A) [or P(B)] when P(A,B) and P(B) [or P(A)] remain unchanged

Comparing Different Measures

10 examples of contingency tables:

Example	f_{11}	f_{10}	f_{01}	f_{00}
E1	8123	83	424	1370
E2	8330	2	622	1046
E3	9481	94	127	298
E4	3954	3080	5	2961
E5	2886	1363	1320	4431
E6	1500	2000	500	6000
E7	4000	2000	1000	3000
E8	4000	2000	2000	2000
E9	1720	7121	5	1154
E10	61	2483	4	7452

Rankings of contingency tables using various measures:

#	ϕ	λ	α	Q	Y	κ	M	J	G	s	c	L	V	I	IS	PS	F	AV	S	ζ	K
E1	1	1	3	3	3	1	2	2	1	3	5	5	4	6	2	2	4	6	1	2	5
E2	2	2	1	1	1	2	1	3	2	2	1	1	1	8	3	5	1	8	2	3	6
E3	3	3	4	4	4	3	3	8	7	1	4	4	6	10	1	8	6	10	3	1	10
E4	4	7	2	2	2	5	4	1	3	6	2	2	2	4	4	1	2	3	4	5	1
E5	5	4	8	8	8	4	7	5	4	7	9	9	9	3	6	3	9	4	5	6	3
E6	6	6	7	7	7	7	6	4	6	9	8	8	7	2	8	6	7	2	7	8	2
E7	7	5	9	9	9	6	8	6	5	4	7	7	8	5	5	4	8	5	6	4	4
E8	8	9	10	10	10	8	10	10	8	4	10	10	10	9	7	7	10	9	8	7	9
E9	9	9	5	5	5	9	9	7	9	8	3	3	3	7	9	9	3	7	9	9	8
E10	10	8	6	6	6	10	5	9	10	10	6	6	5	1	10	10	5	1	10	10	7

Property under Variable Permutation

	$\mathbf{B}$	$\overline{\mathbf{B}}$
$\mathbf{A}$	p	q
$\overline{\mathbf{A}}$	r	s

	$\mathbf{A}$	$\overline{\mathbf{A}}$
$\mathbf{B}$	p	r
$\overline{\mathbf{B}}$	q	s

Does M(A,B) = M(B,A)?

Symmetric measures:

- support, lift, collective strength, cosine, Jaccard, etc

Asymmetric measures:

- confidence, conviction, Laplace, J-measure, etc

Property under Row/Column Scaling

Grade-Gender Example (Mosteller, 1968):

	Male	Female	
High	2	3	5
Low	1	4	5
	3	7	10

	Male	Female	
High	4	30	34
Low	2	40	42
	6	70	76

2x 10x

Mosteller:

Underlying association should be independent of the relative number of male and female students in the samples

Property under Inversion Operation

	A	B	C	D	E	F
Transaction 1	1	0	0	1	0	0
	0	0	1	1	1	0
	0	0	1	1	1	0
	0	0	1	1	1	0
	0	1	1	0	1	1
	0	0	1	1	1	0
	0	0	1	1	1	0
	0	0	1	1	1	0
	0	0	1	1	1	0
Transaction N	1	0	0	1	0	0

(a) columns A, B; (b) columns C, D; (c) columns E, F

Example: ϕ-Coefficient

- ϕ-coefficient is analogous to correlation coefficient for continuous variables

	Y	$\overline{Y}$	
X	60	10	70
$\overline{X}$	10	20	30
	70	30	100

	Y	$\overline{Y}$	
X	20	10	30
$\overline{X}$	10	60	70
	30	70	100

$$\phi = \frac{0.6 - 0.7 \times 0.7}{\sqrt{0.7 \times 0.3 \times 0.7 \times 0.3}}$$
$$= 0.5238$$

$$\phi = \frac{0.2 - 0.3 \times 0.3}{\sqrt{0.7 \times 0.3 \times 0.7 \times 0.3}}$$
$$= 0.5238$$

ϕ Coefficient is the same for both tables

Property under Null Addition

	B	$\overline{B}$
A	p	q
$\overline{A}$	r	s

	B	$\overline{B}$
A	p	q
$\overline{A}$	r	$s + k$

Invariant measures:

- support, cosine, Jaccard, etc

Non-invariant measures:

- correlation, Gini, mutual information, odds ratio, etc

Different Measures have Different Properties

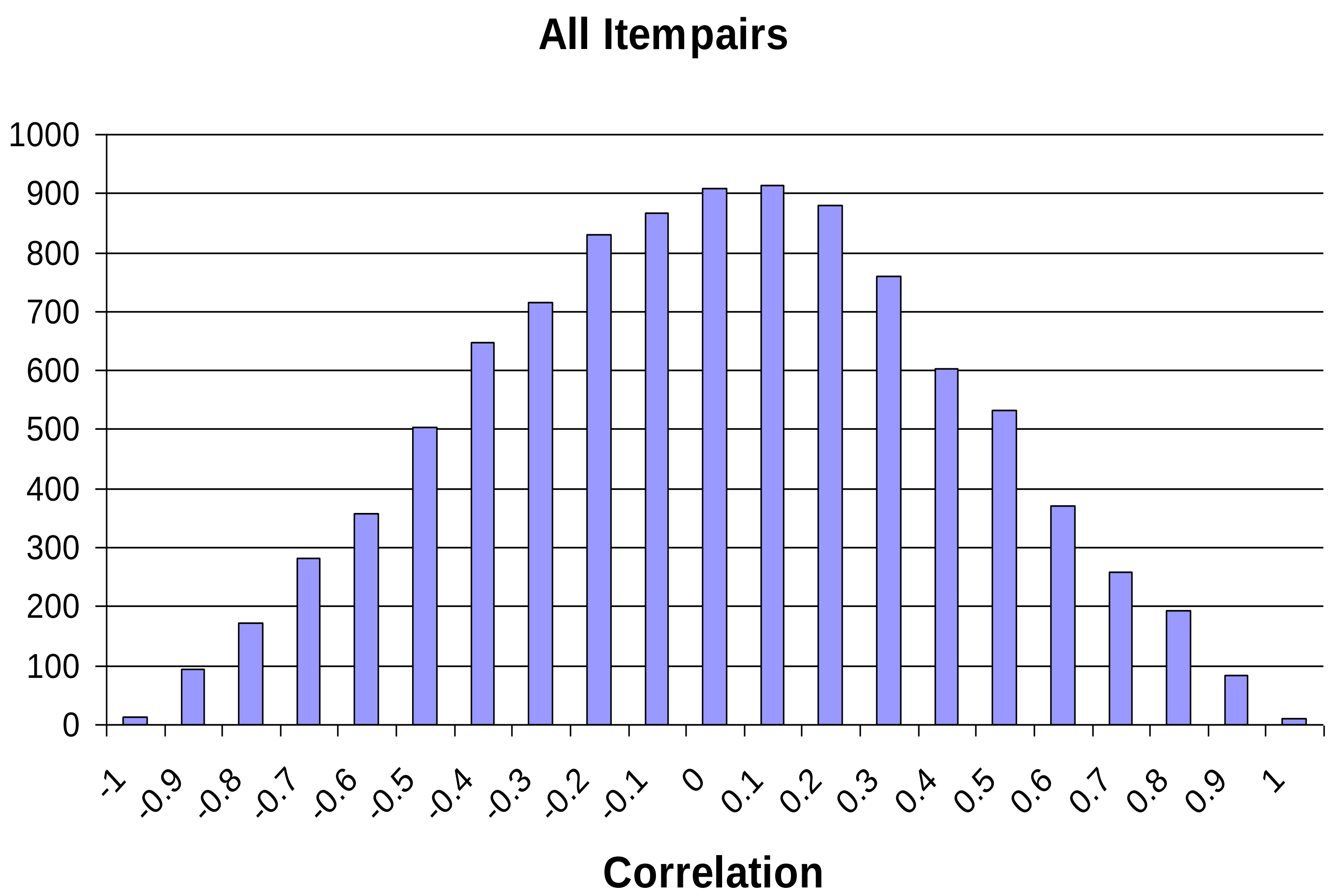
Symbol	Measure	Range	P1	P2	P3	O1	O2	O3	O3'	O4
Φ	Correlation	-1 … 0 … 1	Yes	Yes	Yes	Yes	No	Yes	Yes	No
λ	Lambda	0 … 1	Yes	No	No	Yes	No	No*	Yes	No
α	Odds ratio	0 … 1 … ∞	Yes*	Yes	Yes	Yes	Yes	Yes*	Yes	No
Q	Yule's Q	-1 … 0 … 1	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
Y	Yule's Y	-1 … 0 … 1	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
κ	Cohen's	-1 … 0 … 1	Yes	Yes	Yes	Yes	No	No	Yes	No
M	Mutual Information	0 … 1	Yes	Yes	Yes	Yes	No	No*	Yes	No
J	J-Measure	0 … 1	Yes	No	No	No	No	No	No	No
G	Gini Index	0 … 1	Yes	No	No	No	No	No*	Yes	No
s	Support	0 … 1	No	Yes	No	Yes	No	No	No	No
c	Confidence	0 … 1	No	Yes	No	Yes	No	No	No	Yes
L	Laplace	0 … 1	No	Yes	No	Yes	No	No	No	No
V	Conviction	0.5 … 1 … ∞	No	Yes	No	Yes**	No	No	Yes	No
I	Interest	0 … 1 … ∞	Yes*	Yes	Yes	Yes	No	No	No	No
IS	IS (cosine)	0 .. 1	No	Yes	Yes	Yes	No	No	No	Yes
PS	Piatetsky-Shapiro's	-0.25 … 0 … 0.25	Yes	Yes	Yes	Yes	No	Yes	Yes	No
F	Certainty factor	-1 … 0 … 1	Yes	Yes	Yes	No	No	No	Yes	No
AV	Added value	0.5 … 1 … 1	Yes	Yes	Yes	No	No	No	No	No
S	Collective strength	0 … 1 … ∞	No	Yes	Yes	Yes	No	Yes*	Yes	No
ζ	Jaccard	0 .. 1	No	Yes	Yes	Yes	No	No	No	Yes
K	Klosgen's	$\left(\sqrt{\frac{2}{\sqrt{3}}-1}\right)\left(2-\sqrt{3}-\frac{1}{\sqrt{3}}\right) \ldots 0 \ldots \frac{2}{3\sqrt{3}}$	Yes	Yes	Yes	No	No	No	No	No

Support-based Pruning

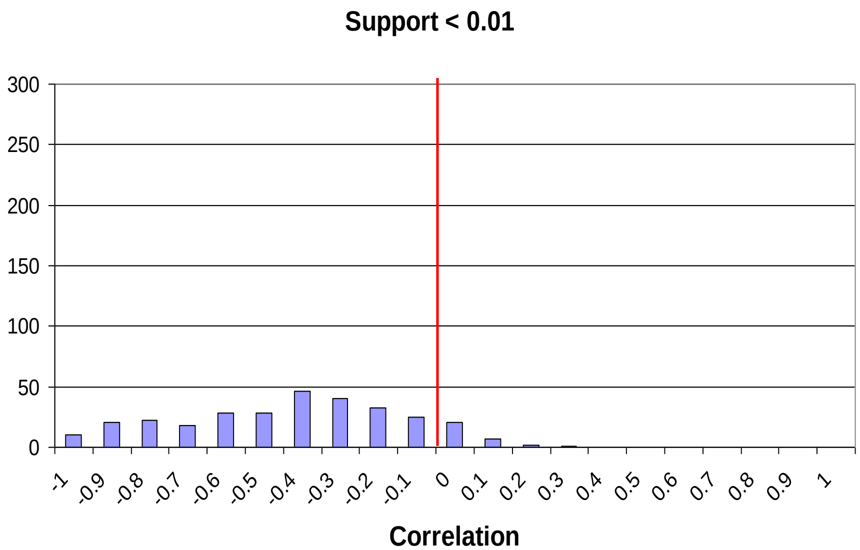
- Most of the association rule mining algorithms use support measure to prune rules and itemsets

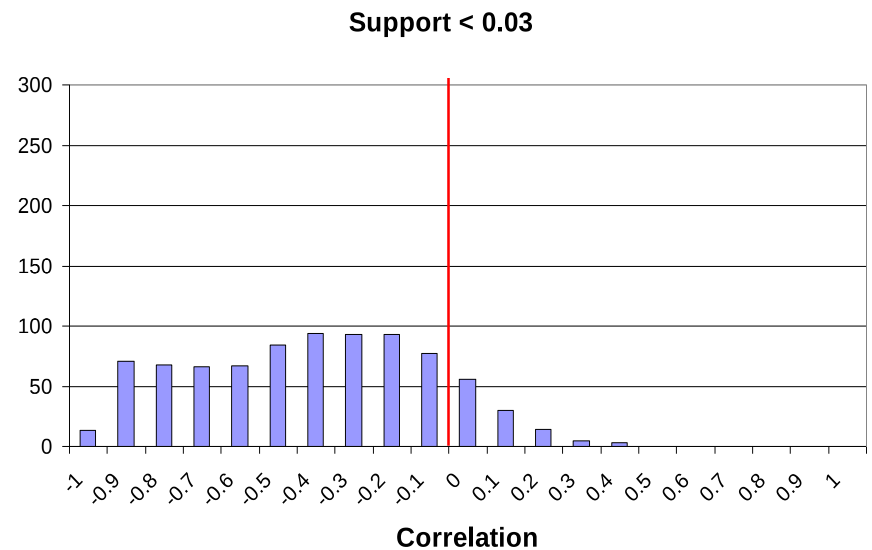
- Study effect of support pruning on correlation of itemsets
  - Generate 10000 random contingency tables
  - Compute support and pairwise correlation for each table
  - Apply support-based pruning and examine the tables that are removed

Effect of Support-based Pruning

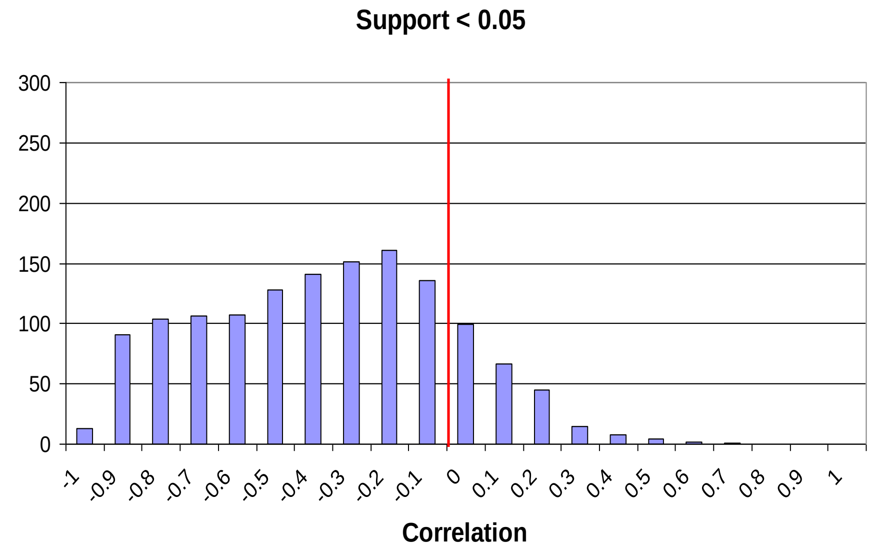


Effect of Support-based Pruning





Support-based pruning eliminates mostly negatively correlated itemsets



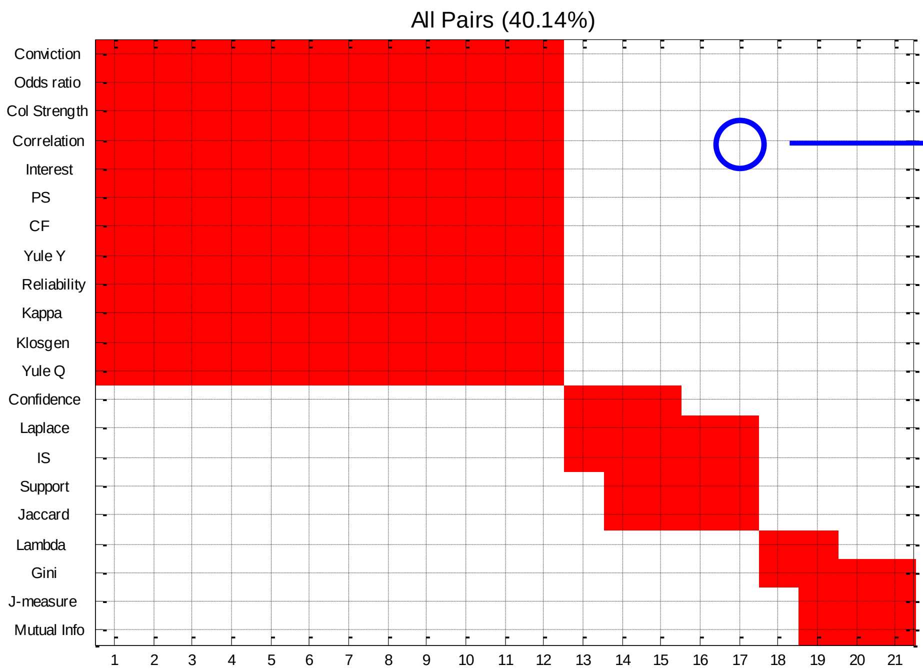
Effect of Support-based Pruning

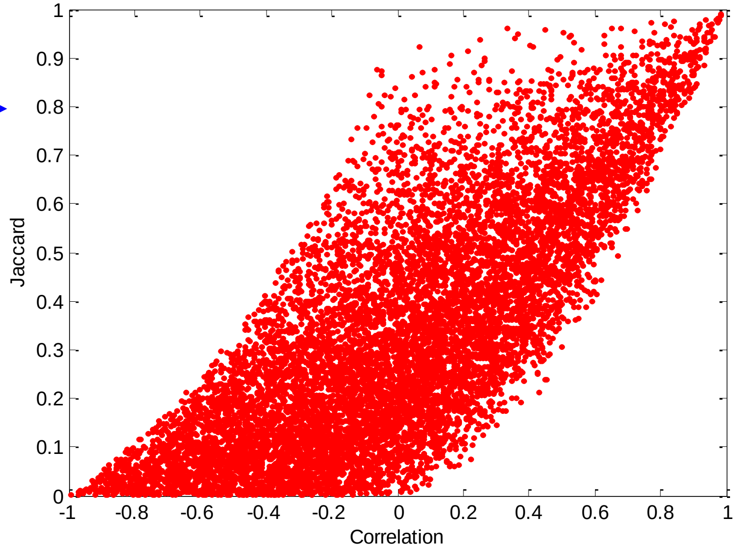
- Investigate how support-based pruning affects other measures

- Steps:
  - Generate 10000 contingency tables
  - Rank each table according to the different measures
  - Compute the pair-wise correlation between the measures

Effect of Support-based Pruning

- Without Support Pruning (All Pairs)



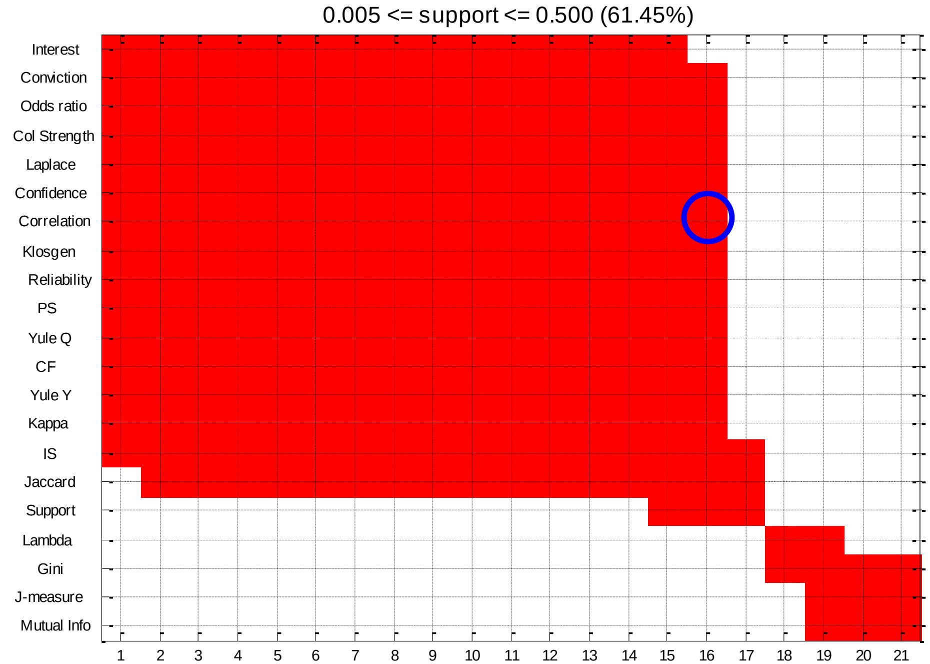


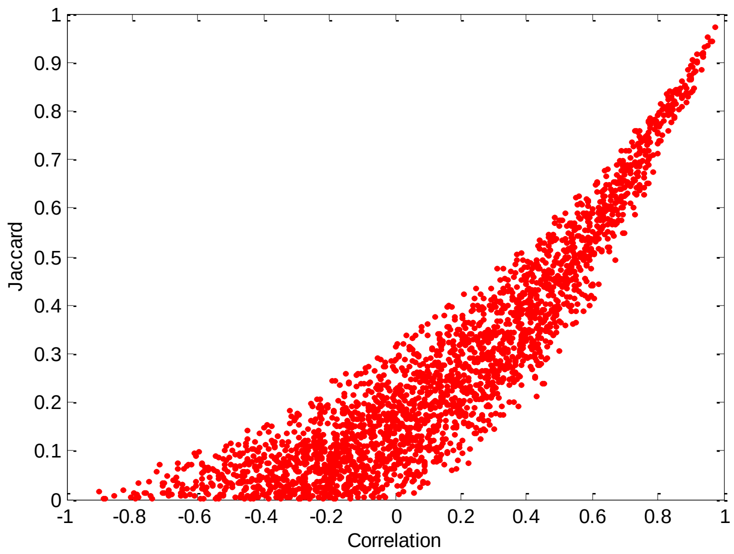
Scatter Plot between Correlation & Jaccard Measure

- Red cells indicate correlation between the pair of measures > 0.85
- 40.14% pairs have correlation > 0.85

Effect of Support-based Pruning

- 0.5% ≤ support ≤ 50%



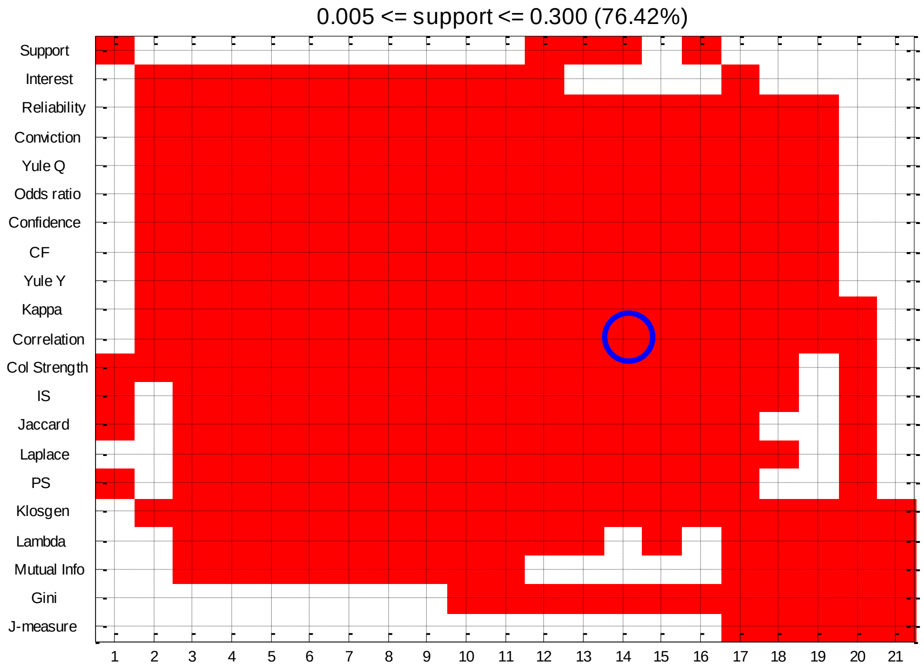


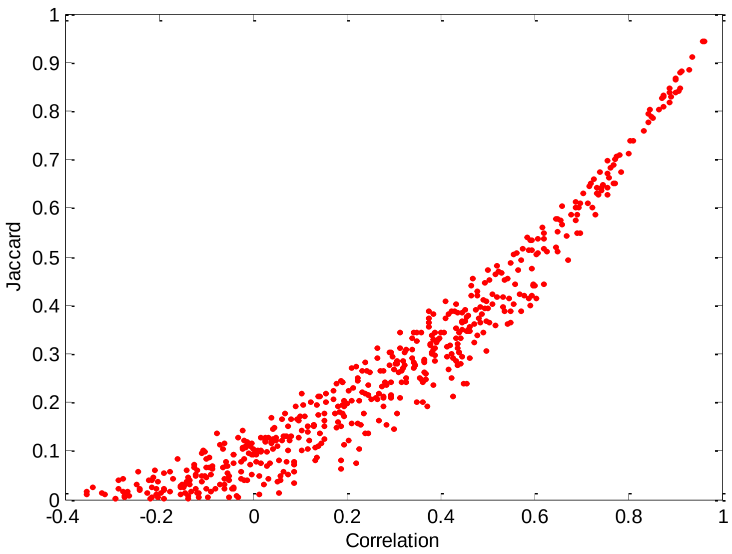
Scatter Plot between Correlation & Jaccard Measure:

- 61.45% pairs have correlation > 0.85

Effect of Support-based Pruning

- 0.5% ≤ support ≤ 30%





Scatter Plot between Correlation & Jaccard Measure

- 76.42% pairs have correlation > 0.85

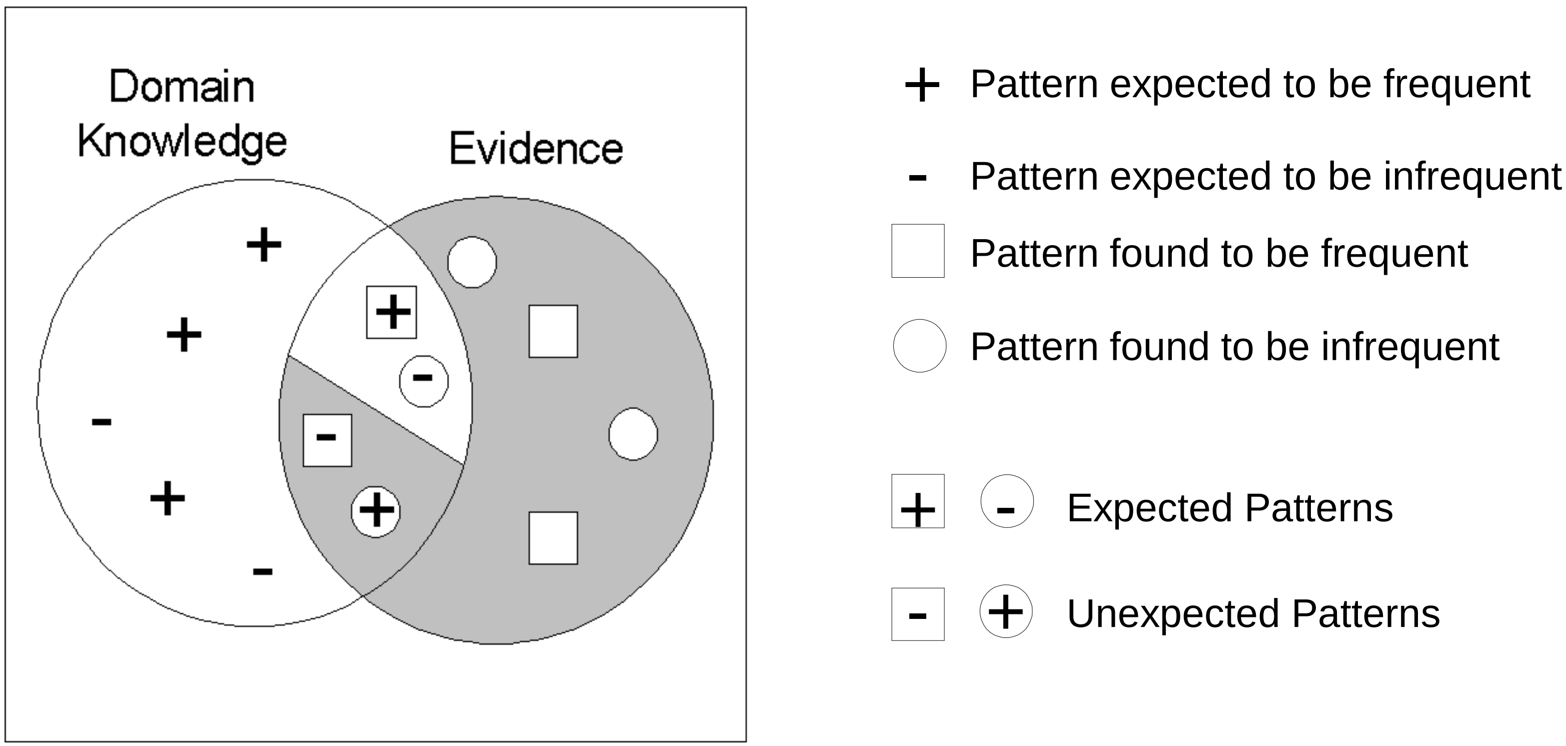
Subjective Interestingness Measure

- Objective measure:
  - Rank patterns based on statistics computed from data
  - e.g., 21 measures of association (support, confidence, Laplace, Gini, mutual information, Jaccard, etc).

- Subjective measure:
  - Rank patterns according to user's interpretation
    - A pattern is subjectively interesting if it contradicts the expectation of a user (Silberschatz & Tuzhilin)
    - A pattern is subjectively interesting if it is actionable (Silberschatz & Tuzhilin)

Interestingness via Unexpectedness

- Need to model expectation of users (domain knowledge)



- Need to combine expectation of users with evidence from data (i.e., extracted patterns)

Interestingness via Unexpectedness

- Web Data (Cooley et al 2001)
  - Domain knowledge in the form of site structure
  - Given an itemset F = $\{X_1, X_2, \ldots, X_k\}$ (X_i : Web pages)
    - L: number of links connecting the pages
    - lfactor = L / (k × k-1)
    - cfactor = 1 (if graph is connected), 0 (disconnected graph)
  - Structure evidence = cfactor × lfactor
  - Usage evidence $= \dfrac{P(X_1 \cap X_2 \cap \ldots \cap X_k)}{P(X_1 \cup X_2 \cup \ldots \cup X_k)}$
  - Use Dempster-Shafer theory to combine domain knowledge and evidence from data