CS 422 Data Mining

Lecture 5

September 20, 2018

- Association Rule Mining

Association Analysis

- Large Data
  - Transactions
  - Market basket transactions

Association Analysis

- Large Data
  - Transactions
  - Market basket transactions

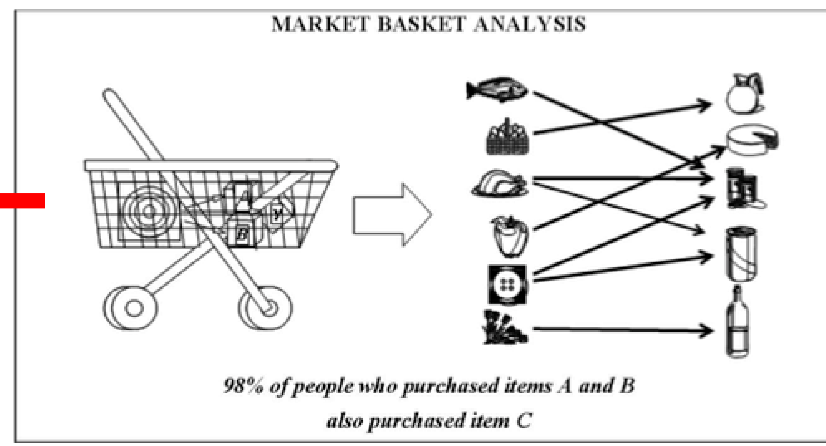
- Association Analysis
  - Discovering of interesting relationships in large data sets

Market Basket Analysis

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

Market Basket Analysis

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



Market Basket Analysis

- Diapers + Beer!

Market Basket Analysis

- Diapers + Beer!
- Diapers ->
  - baby ->
  - don't go out to a bar ->
  - buy more beer for home

Market Basket Analysis

- Hot dog and mustard

+ **Mustard**

Market Basket Analysis

- Hot dog and mustard

+

Association Rules: General Idea

- Given a set of baskets
  - Want to discover association rules
  - People who bought {x,y,z} tend to buy {v,w}
    - Amazon!
- 2 step approach:
  - Find frequent itemsets
  - Generate association rules

Problem Definition

- Itemset $X = \{i | i \subseteq I\}$
  - {Bread, Milk}
  - *k*-itemset has k items
  - {Bread, Milk} is a *2*-itemset
- Transaction t_i contains an itemset 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

Market basket transactions

$t_i \subseteq T$

t1={Bread, Milk}

t_i contains X

$X \subseteq t_i$

$X = \{i_k, i_m, ..\}$,

where $i_k \subseteq I$

X={Bread, Milk}

Problem Definition

- Itemset $X = \{i | i \subseteq I\}$
  - {Bread, Milk}
  - *k*-itemset has k items
  - {Bread, Milk} is a *2*-itemset
- Transaction t_i contains an itemset 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

Market basket transactions

$t_i \subseteq T$

t1={Bread, Milk}

t_i contains X, Y

$X \subseteq t_i$, $Y \subseteq t_i$

X={Bread, Milk}

Y={Bread}

Problem Definition

- Set of items $I=\{i_1,i_2,\ldots,i_d\}$
- Set of transaction $T=\{t_1,t_2,\ldots t_N\}$
- Itemset $X = \{i|i \subseteq I\}$
  - *k*-itemset has k items
- Transaction t_i contains an itemset X

Support Count

- Support Count of an itemset X: $\sigma(X)$
  - $\sigma(X)$ = Number of transactions that contain X

Support Count

- Support Count of an itemset X
  - Number of transactions that contain X
  - Number of transations that support {Bread, Milk}?

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Egg s
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Bee r
5	Bread, Milk, Diaper, Coke

Market basket transactions

Support Count

- Support Count of an itemset X
  - Number of transactions that contain X
  - Number of transations that support {Bread, Milk}?

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

Market basket transactions

- σ(**{Bread, Milk}) = 3**

Support Count

- Support Count of an itemset X
  - Number of transactions that contain X
  - Number of transations that support {Bread}?

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

Market basket transactions

Support Count

- Support Count of an itemset X
  - Number of transactions that contain X
  - Number of transations that support {Bread}?

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

Market basket transactions

- $\sigma(\{\text{Bread}\}) = 4$

Support Count

- Support Count of an itemset X
  - Number of transactions that contain X
  - Number of transations that support {Bread, Milk, Diaper, Coke}?

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

Market basket transactions

Support Count

- Support Count of an itemset X
  - Number of transactions that contain X
  - Number of transations that support {Bread, Milk, Diaper, Coke}?

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

Market basket transactions

- **σ({Bread, Milk Diaper, Coke}) = 1**

Association Rule

- Association rule is an implication expression
  - X -> Y where X and Y are disjoint itemsets

Association Rule

- Association rule is an implication expression
  - X -> Y where X and Y are disjoint itemsets

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

Market basket transactions

Association rules:

{Diapers $\rightarrow$ Beer}
{Beer, Bread} $\rightarrow$ {Milk}

Association Rule

- Association rule:
  - Support
  - Confidence

Association Rule

- **Association rule:**
  - Support X->Y
    - Number of transactions containing $X \cup Y$
    - $S(X\text{->}Y) = \sigma(X \cup Y)/N$

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

Market basket transactions

Association rules:

$\{\text{Diapers} \rightarrow \text{Beer}\}$
$\{\text{Beer, Bread}\} \rightarrow \{\text{Milk}\}$

$S(\text{Diapers} \cup \text{Beer}) =$ **?**

$S(\text{Beer, Bread} \cup \text{Milk}) =$ **?**

Association Rule

- **Association rule:**
  - Support X->Y
    - Number of transactions containing $X \cup Y$
    - $S(X\text{->}Y) = \sigma(X \cup Y)/N$

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

Market basket transactions

Association rules:

{Diapers → Beer}
{Beer, Bread} → {Milk}

$S(\textbf{Diapers} \cup \textbf{Beer}) =$ **3/5**

$S(\textbf{Beer, Bread} \cup \textbf{Milk}) =$ **1/5**

Association Rule

- Association rule:
  - Support
  - Confidence

Association Rule

- **Association rule:**
  - Support
  - Confidence X->Y
    - How often transactions that contain X also contain Y
    - $c(X \rightarrow Y) = \sigma(X \cup Y) / \sigma(X)$

Association Rule

- Association rule:
  - Support
  - Confidence X->Y
    - How often transactions that contain X also contain Y
    - $c(X \to Y) = \sigma(X \cup Y) / \sigma(X)$

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

Market basket transactions

Association rules:

{Diapers → Beer}
{Beer, Bread} → {Milk}

$C(\textbf{Diapers} \to \textbf{Beer}) =$

?

$C(\textbf{Beer, Bread} \to \textbf{Milk}) =$

?

Association Rule

- Association rule:
  - Support
  - Confidence X->Y
    - How often transactions that contain X also contain Y
    - $c(X\text{->}Y) = \sigma(X \cup Y)/ \sigma(X)$

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

Market basket transactions

Association rules:

{Diapers → Beer}
{Beer, Bread} → {Milk}

$$C(\text{Diapers} \rightarrow \text{Beer}) = \mathbf{3/4}$$

$$C(\text{Beer, Bread} \rightarrow \text{Milk}) = \mathbf{1/2}$$

Use of Support and Confidence

- Support
  - Rule with a low support can occur by chance
  - Low support rules are not interesting from the business perspective
  - Eliminate uninteresting rules
- Confidence
  - Reliability of the implication from an association rule X->Y
  - Conditional probability P(Y|X)

Association Rule Mining Problem

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

Association Rule Mining Problem

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

{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)

Minsup=0.4
Minconf=0.6

Association Rule Mining Problem

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

{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)

Minsup=0.4
Minconf=0.6

{Milk,Diaper} → {Beer}
{Diaper,Beer} → {Milk}
{Beer} → {Milk,Diaper}

Computational Challenge

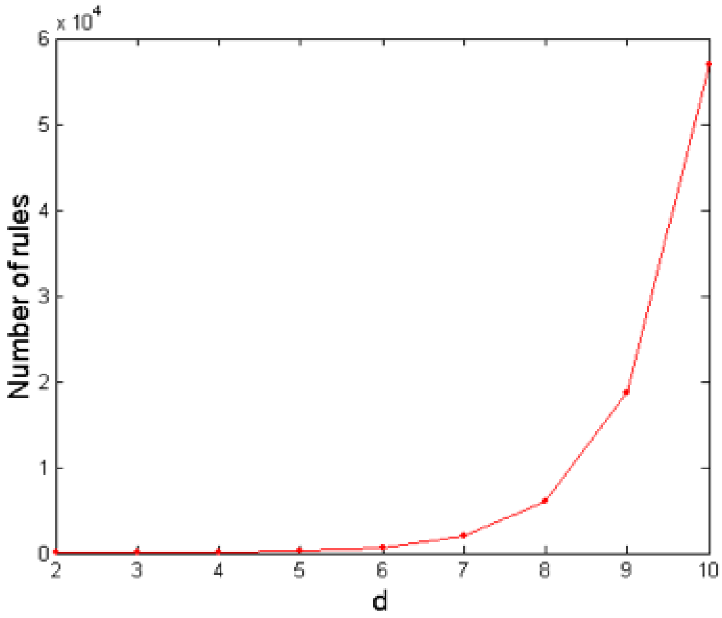
- Brute-force approach
  - Compute support and confidence for every possilbe rule

{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)

  - In our example d=6, there are 602 rules
    - If minsup=20%
    - If minconf=50%, then
    - 80% of rules are discarded

Computational Challenge

- The number of possible rules that contains d items
  - $R = 3^d - 2^{(d+1)} + 1$



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

$$= 3^d - 2^{d+1} + 1$$

Computational Challenge

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)

- 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

Computational Challenge

- Two steps
  - Frequent Itemset Generation
    - Generate all itemsets with support ≥ *minsup*
  - Rule Generation
    - Generate high confidence rules from each frequent itemset

Frequent itemset

- Brute-force approach
  - Support count for every itemset
  - Use lattice structure

Reduce Complexity

- Reduce the number of candidate itemsets M
- Reduce the number of transactions
- Reduce the number of comparisons

- Apriori Principle

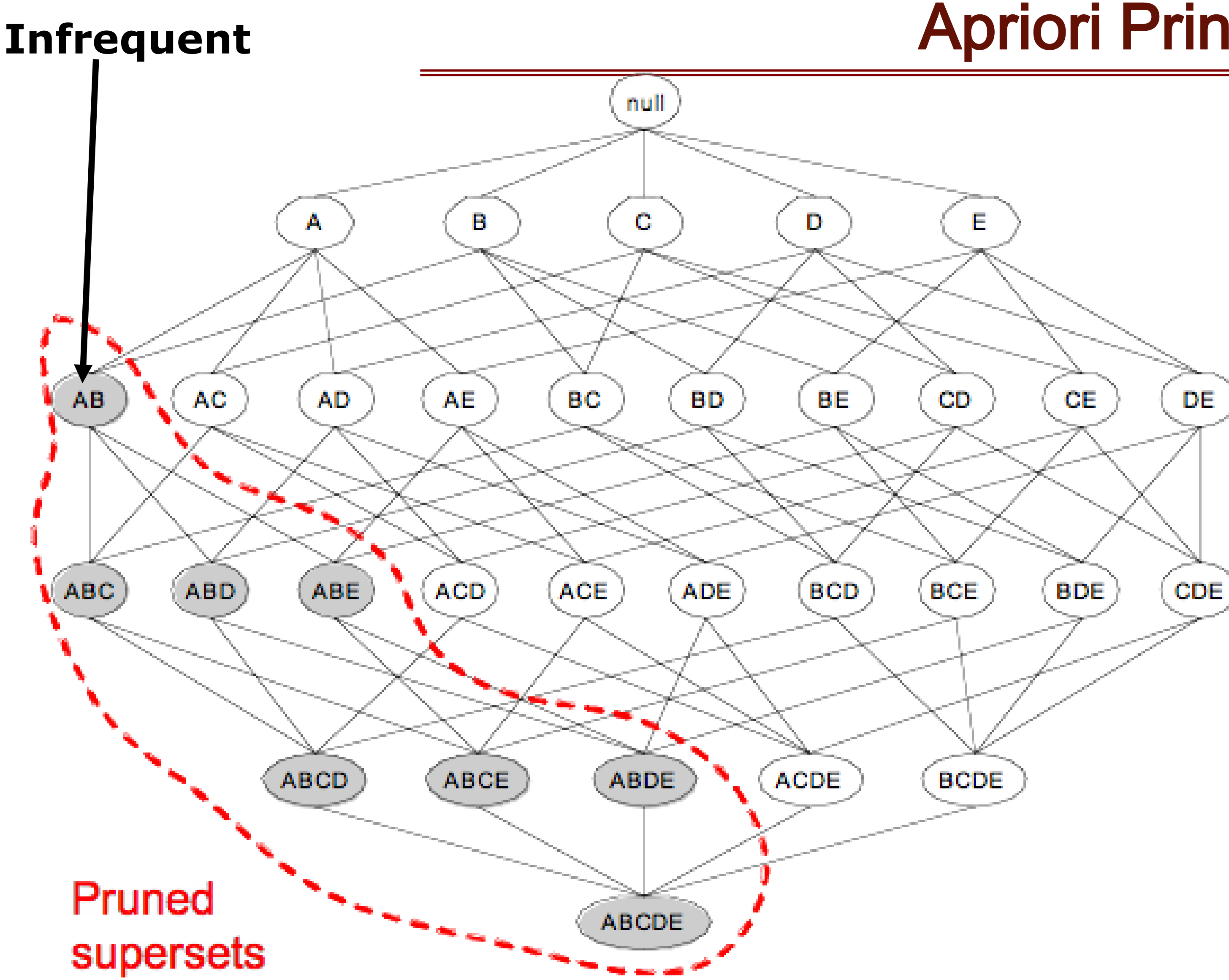
Apriori Principle

- **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

Apriori Principle



Apriori Algorithm

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

Support Counting

- Frequency of each candidate itemset
- Compare each transaction against each candidate, update the counts

Support Counting

K-1 Iteration's itemsets

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

K Iteration's candidate itemsets

{Bread, Milk}
{Bread, Beer}
{Diaper, Bread}
{Diaper, Milk}
…

{Bread, Milk, Beer}
{Diaper, Bread, Milk}
…

Support Counting

K-1 Iteration's itemsets

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

K Iteration's candidate itemsets

{Bread, Milk}
{Bread, Beer}
{Diaper, Bread}
{Diaper, Milk}
…

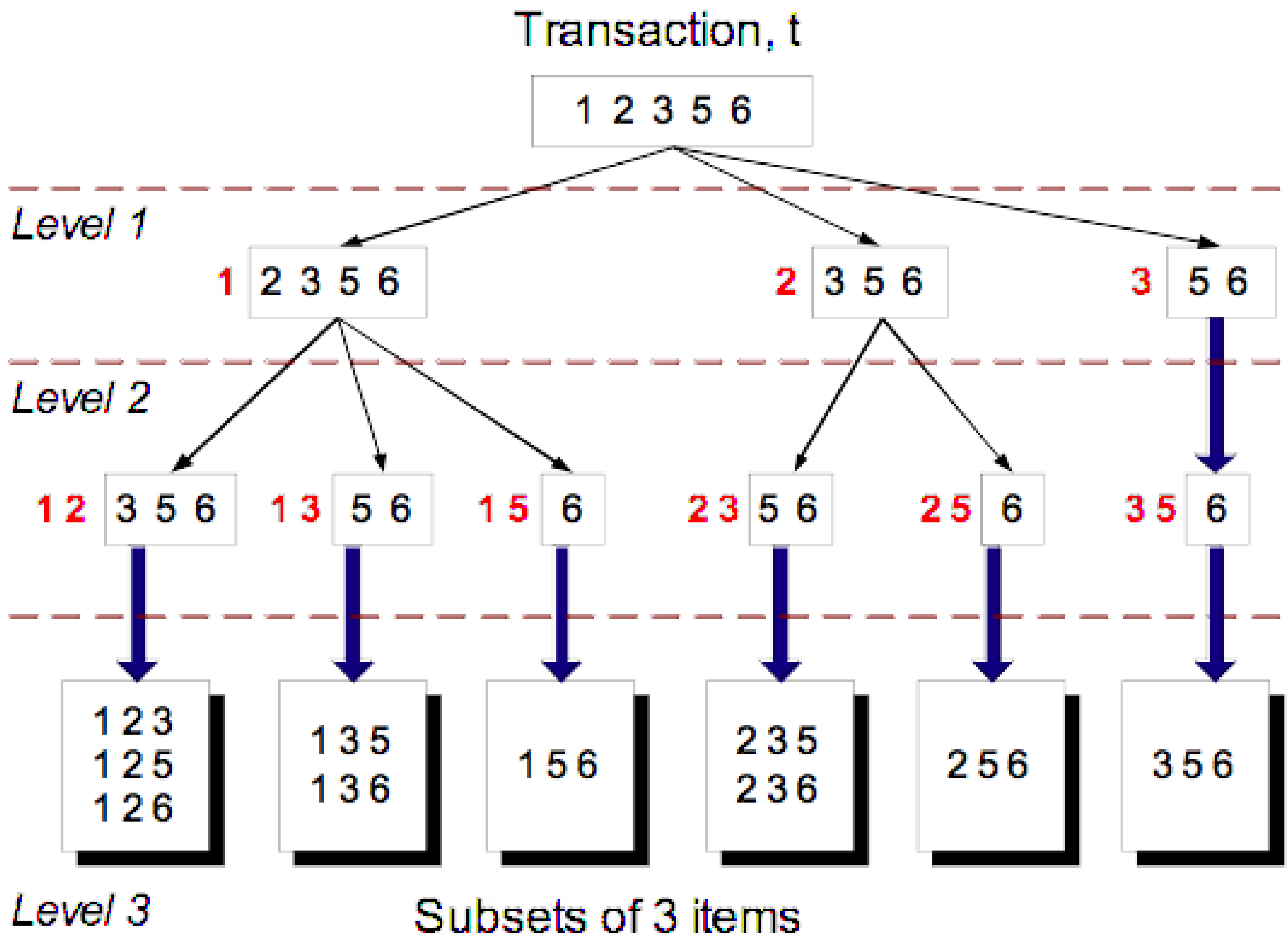
{Bread, Milk, Beer}

{Diaper, Bread, Milk}
…

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

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

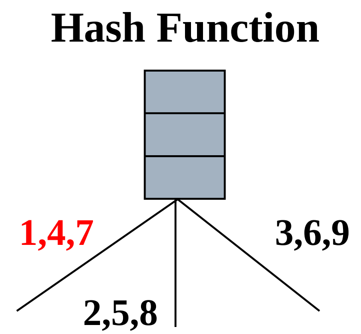


Reducing Number of Comparisons

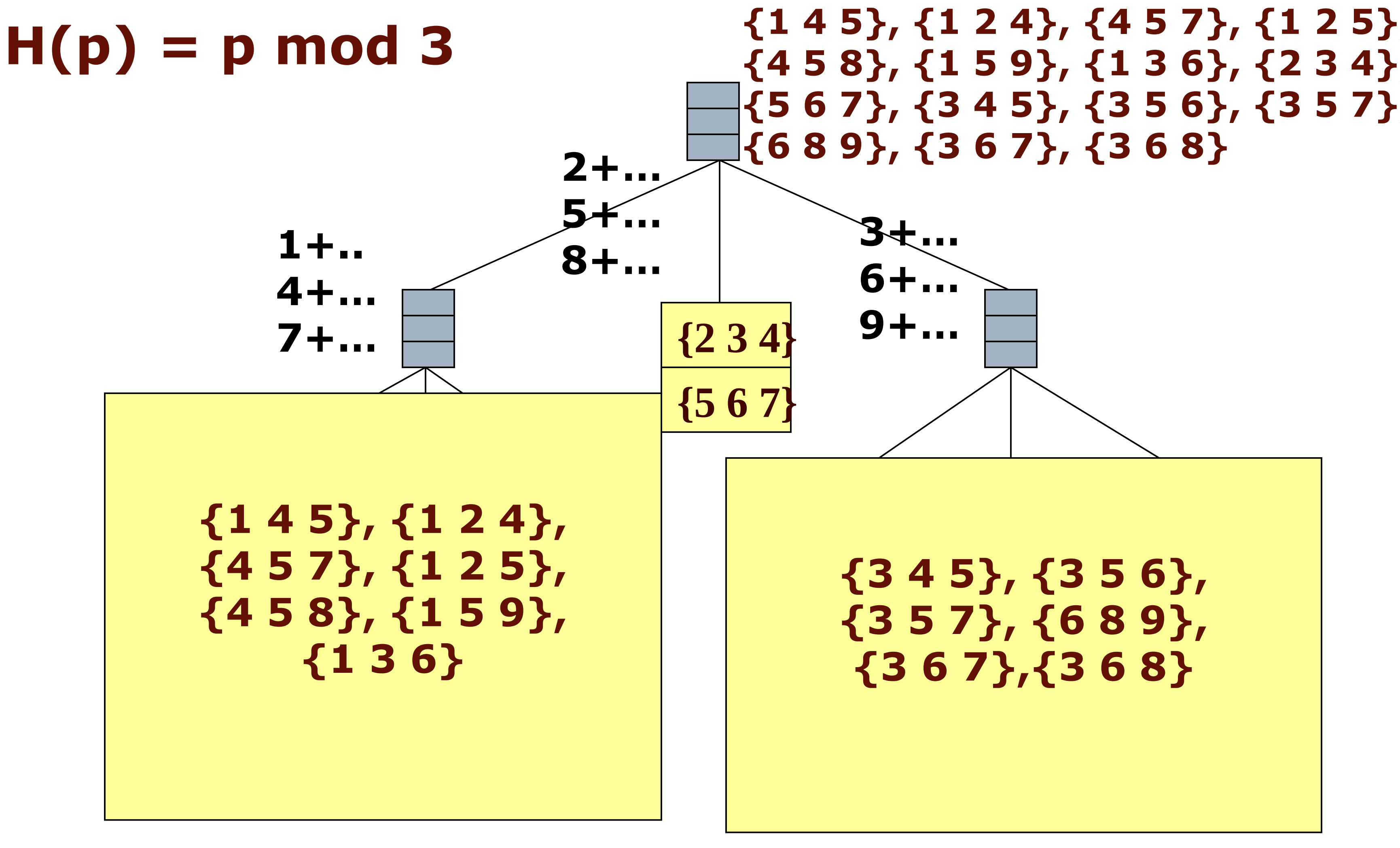
- Candidate counting:
  - Scan the database of transactions to determine the support of each candidate itemset
  - To reduce the number of comparisons, store the candidates in a hash structure
  - Store transactions in the hash as well
  - Instead of matching each transaction against every candidate,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

Candidate Itemsets Hash Tree

- Suppose you have 9 items, 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}
- 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)
  - H(p) = p mod 3
  - Sort items in the itemsets

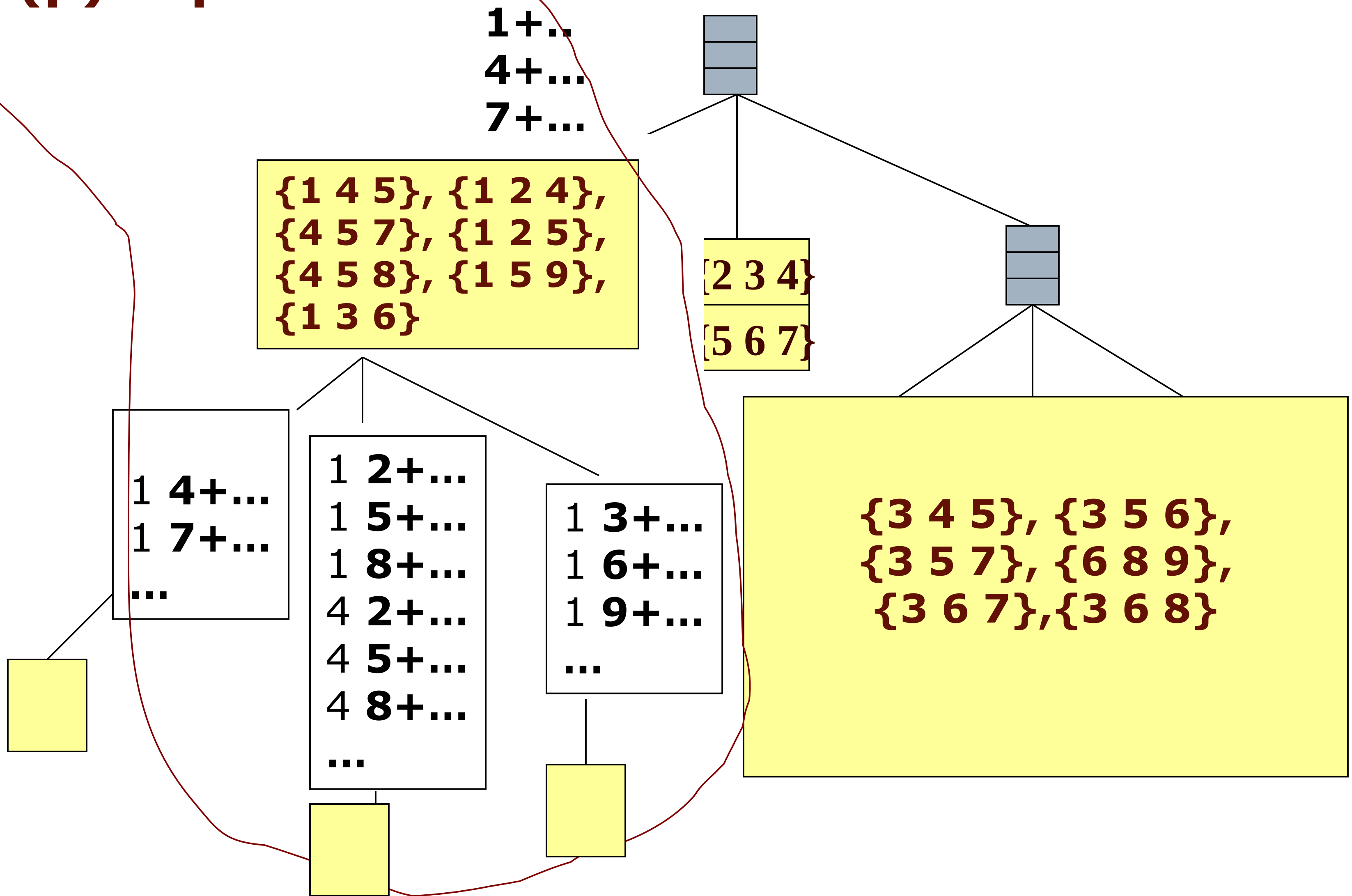


Candidate Itemsets Hash Tree



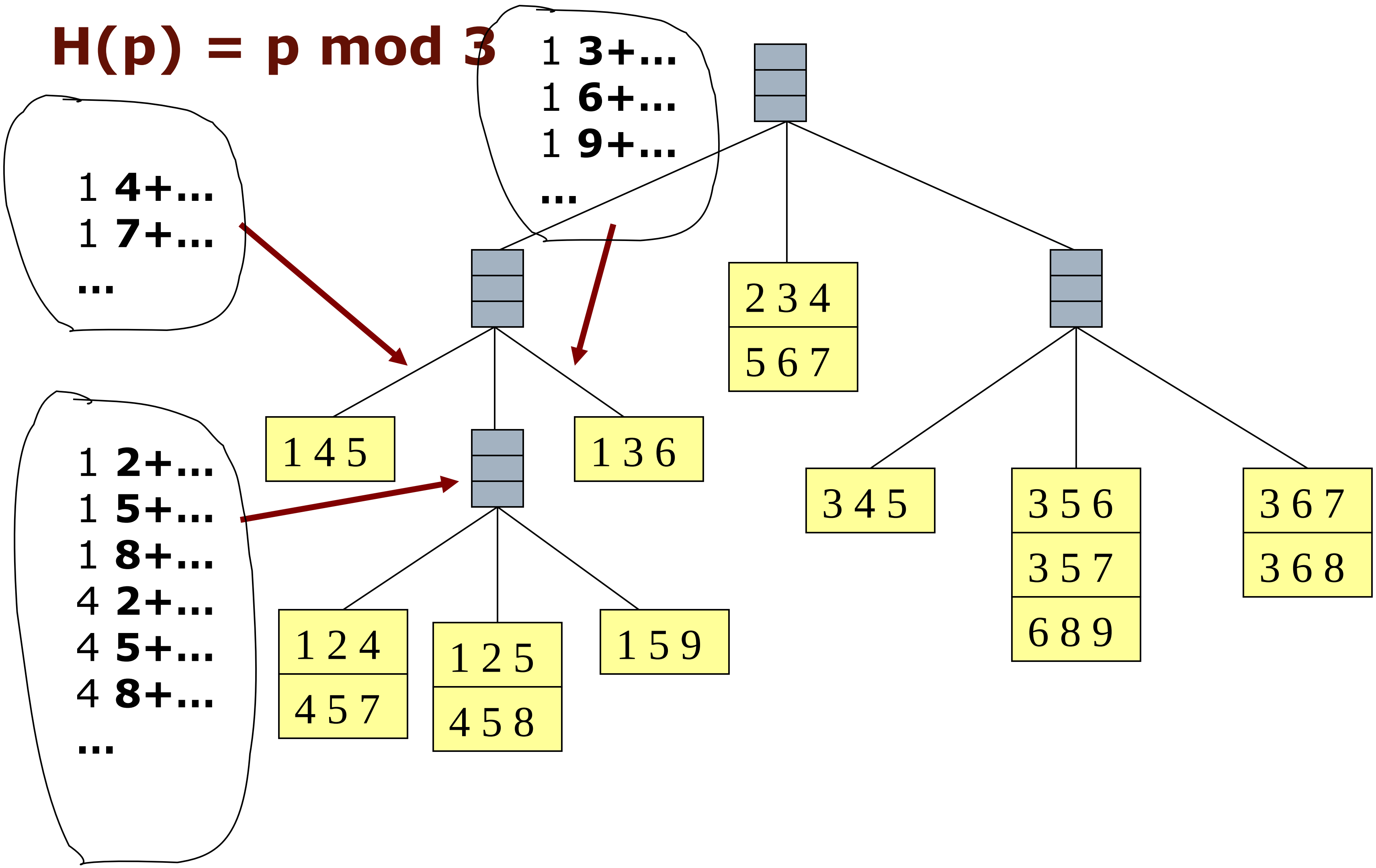
Candidate Itemsets Hash Tree

H(p) = p mod 3

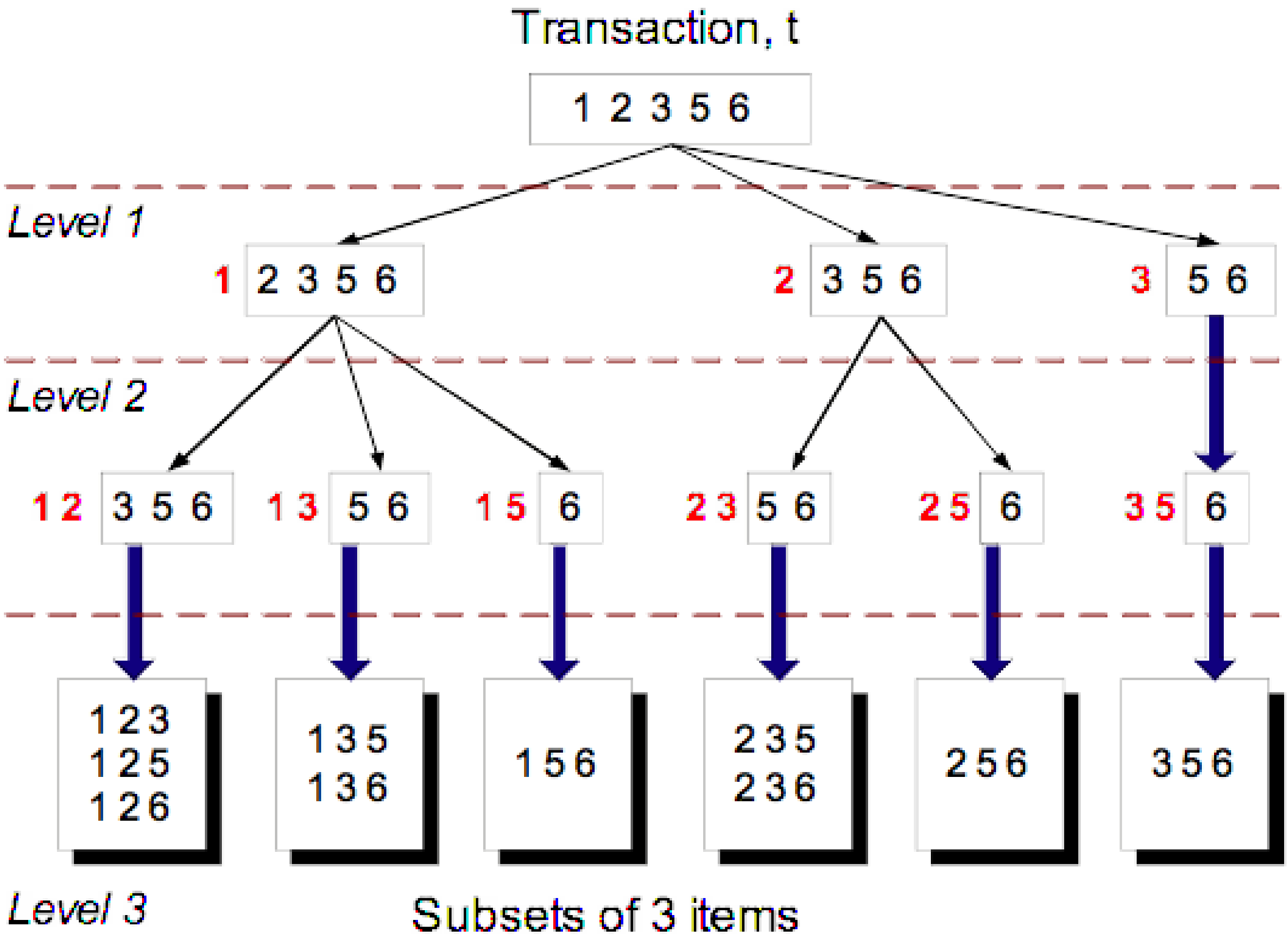


Candidate Itemsets Hash Tree

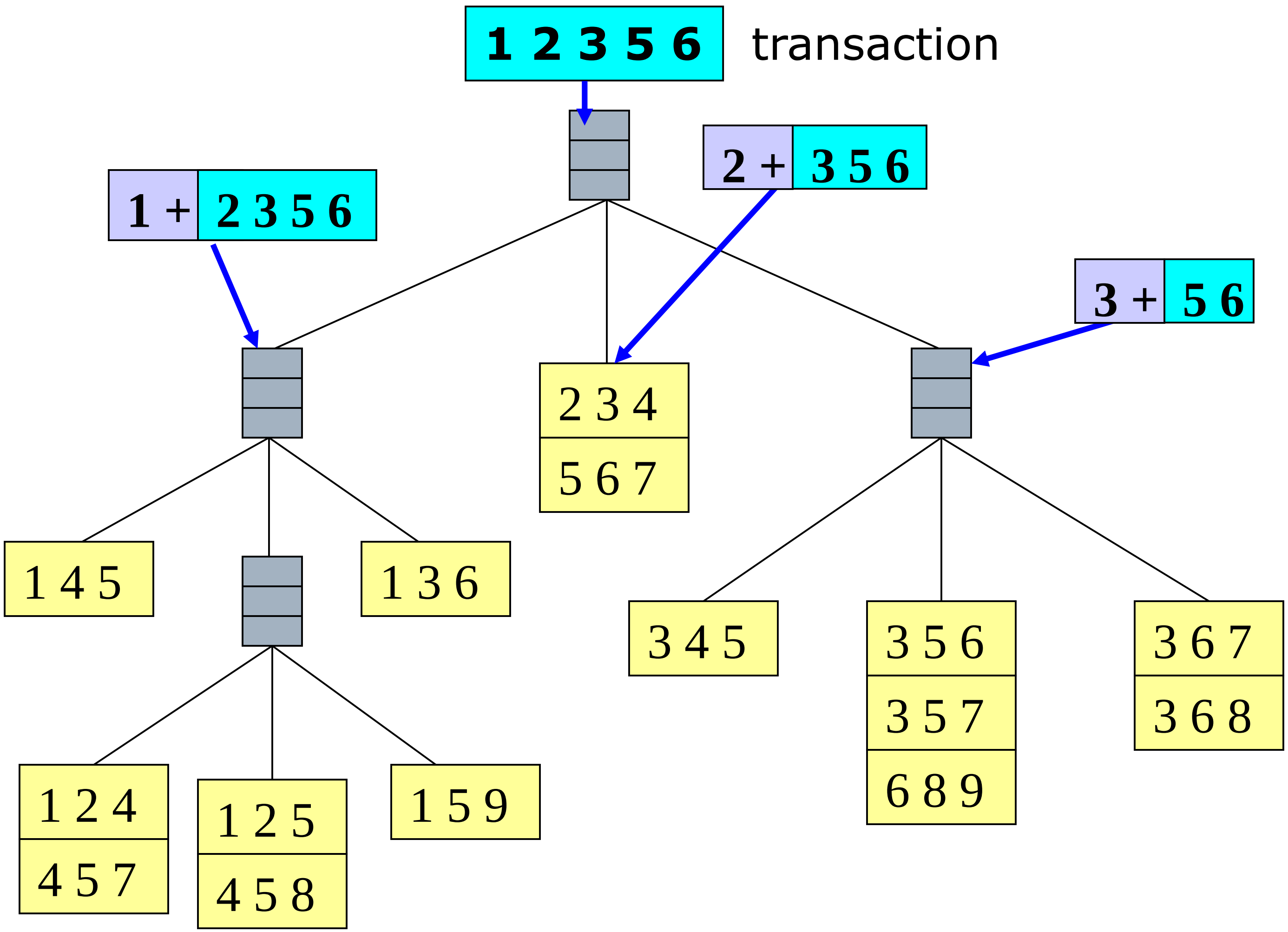
H(p) = p mod 3



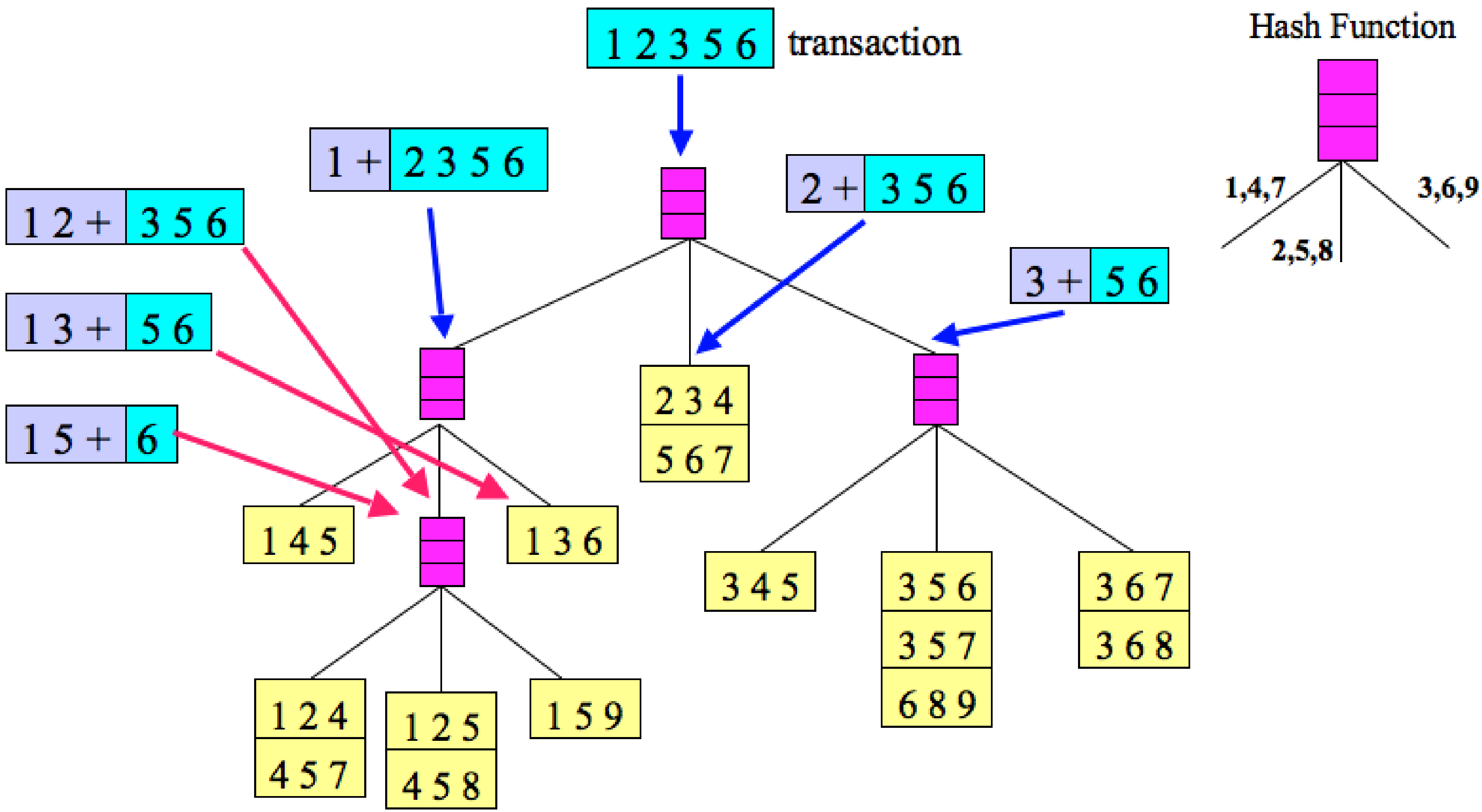
Enumerating Itemsets in Transaction



Itemsets from Transaction in Candidate Hash Tree

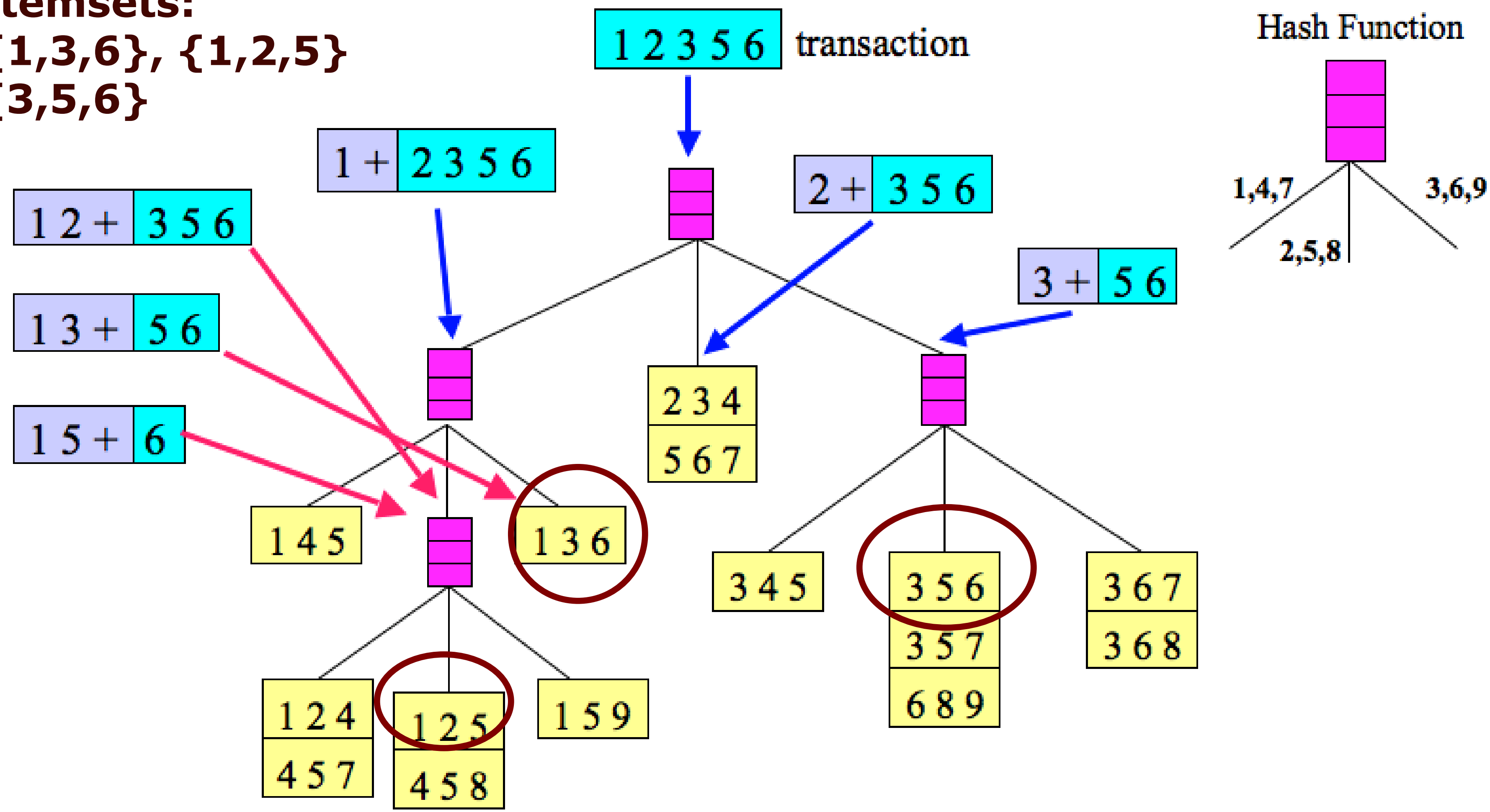


Itemsets from Transaction in Candidate Hash Tree



Count Update

Increment counts for matching candidate Itemsets: {1,3,6}, {1,2,5} {3,5,6}



Complexity Factors

- 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)

- Rule Generation

Rule Generation

- Given a frequent itemset Y, find all non-empty subsets $X \subset Y$ such that
- The rule $X \rightarrow Y - X$

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

Rule Generation

- Given a frequent itemset Y, find all non-empty subsets $X \subset Y$ such that
- The rule $X \rightarrow Y - X$

  satisfies the minimum confidence requirement
- If {A,B,C,D} is a frequent itemset, candidate rules:

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

- Since Y is the frequent itemset, each rules meets the minimum confidence requirement

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., Y = {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 in 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

Support

- 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

- Association Rule Parameters

Effect of Support Distribution

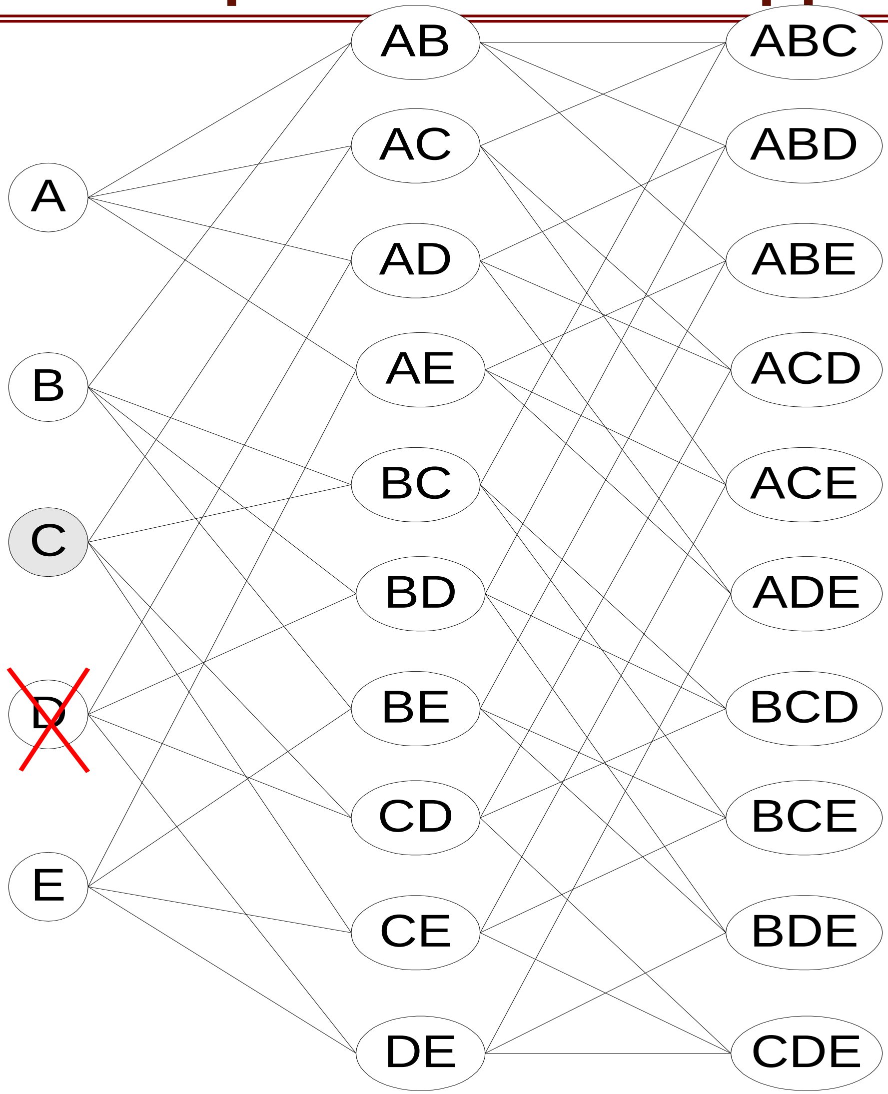
- How to set the appropriate *minsup* threshold?
  - If *minsup* is set too high, we could miss itemsets involving interesting rare items (e.g., expensive products)
  - If *minsup* is set too low, it is computationally expensive and the number of itemsets is very large
- Using a single minimum support 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

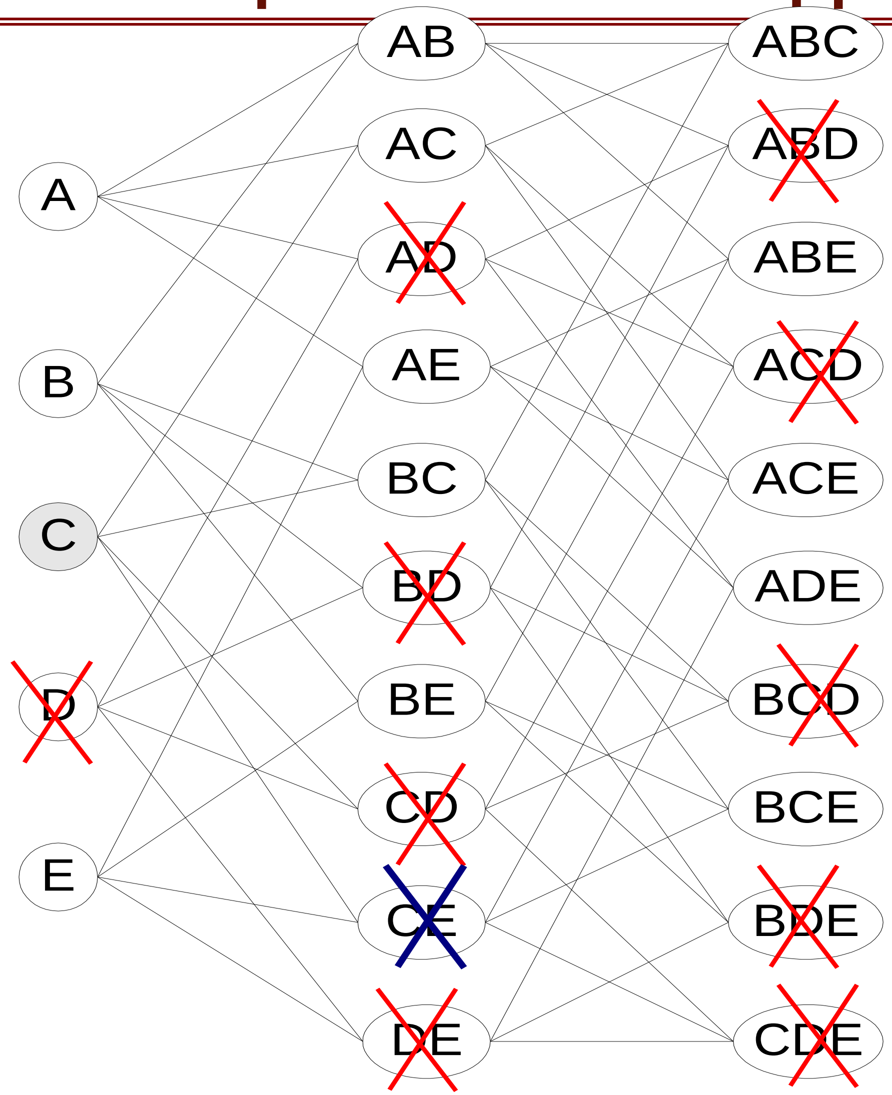
Multiple Minimum Support

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

      subset 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.

- Association Rule Evaluation

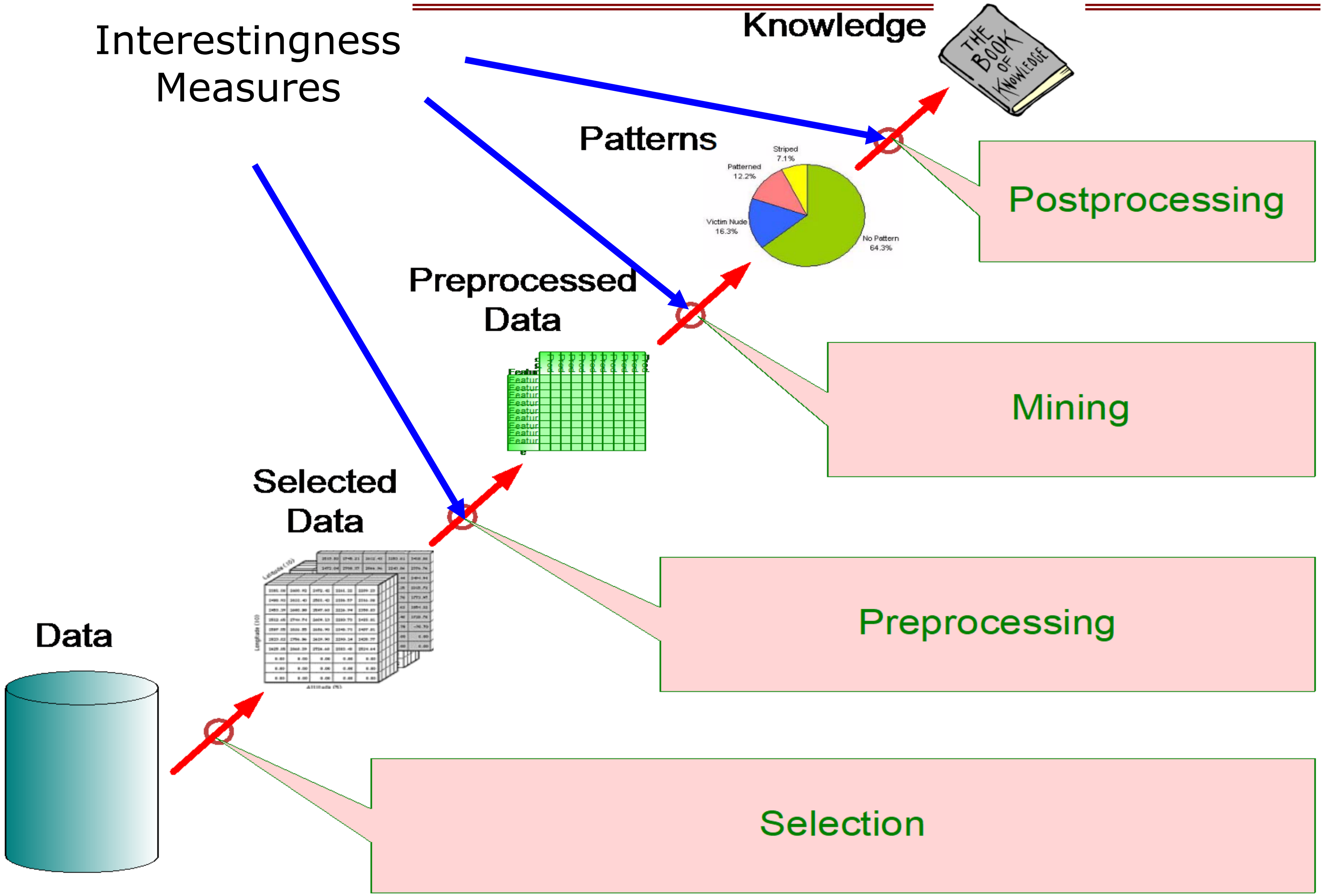
Pattern Evaluation

- 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 and 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

$c(X\text{->}Y) = \sigma(X \cup Y)/\ \sigma(X)$

Association Rule: Tea $\rightarrow$ Coffee

Confidence= P(Coffee|Tea) = 0.75

but P(Coffee) = 0.9

Although confidence is high, rule is misleading

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

Statistical-based Measures

- Confidence does not use the support for B in A->B
- Other measures
  - Lift: ratio between the rule confidence and support of B
    - Lift = c(A->B) / s(B)

Association and Correlation

- As we can see support-confidence framework can be misleading; it can identify a rule (A=>B) as interesting (strong) when, in fact the occurrence of A might not imply the occurrence of B.

- Correlation Analysis provides an alternative framework for finding interesting relationships, or to improve understanding of meaning of some association rules (a lift of an association rule).

Correlation Concepts

- Two item sets A and B are independent (the occurrence of A is independent of the occurrence of item set B) iff
  - $P(A \cup B) = P(A) \cdot P(B)$
- Otherwise A and B are dependent and correlated
- The measure of correlation, or correlation between A and B is given by the formula:
  - Corr(A,B)= P(A U B) / P(A) * P(B)

Correlation Concepts

- corr(A,B) >1 means that A and B are positively correlated i.e. the occurrence of one implies the occurrence of the other.

- corr(A,B) < 1 means that the occurrence of A is negatively correlated with (or discourages) the occurrence of B.

- corr(A,B) =1 means that A and B are independent and there is no correlation between them.

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

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

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

#	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)	$\max\Big(P(A,B)\log(\frac{P(B\|A)}{P(B)})+P(A\overline{B})\log(\frac{P(\overline{B}\|A)}{P(\overline{B})}),$ $P(A,B)\log(\frac{P(A\|B)}{P(A)})+P(\overline{A}B)\log(\frac{P(\overline{A}\|B)}{P(\overline{A})})\Big)$
9	Gini index (G)	$\max\Big(P(A)[P(B\|A)^2+P(\overline{B}\|A)^2]+P(\overline{A})[P(B\|\overline{A})^2+P(\overline{B}\|\overline{A})^2]$ $-P(B)^2-P(\overline{B})^2,$ $P(B)[P(A\|B)^2+P(\overline{A}\|B)^2]+P(\overline{B})[P(A\|\overline{B})^2+P(\overline{A}\|\overline{B})^2]$ $-P(A)^2-P(\overline{A})^2\Big)$
10	Support (s)	$P(A,B)$
11	Confidence (c)	$\max(P(B\|A),P(A\|B))$
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)	$\max\Big(\frac{P(B\|A)-P(B)}{1-P(B)},\frac{P(A\|B)-P(A)}{1-P(A)}\Big)$
18	Added Value (AV)	$\max(P(B\|A)-P(B),P(A\|B)-P(A))$
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)	$\sqrt{P(A,B)}\max(P(B\|A)-P(B),P(A\|B)-P(A))$

Properties of A Good Measure

- 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

Property under Variable Permutation

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

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

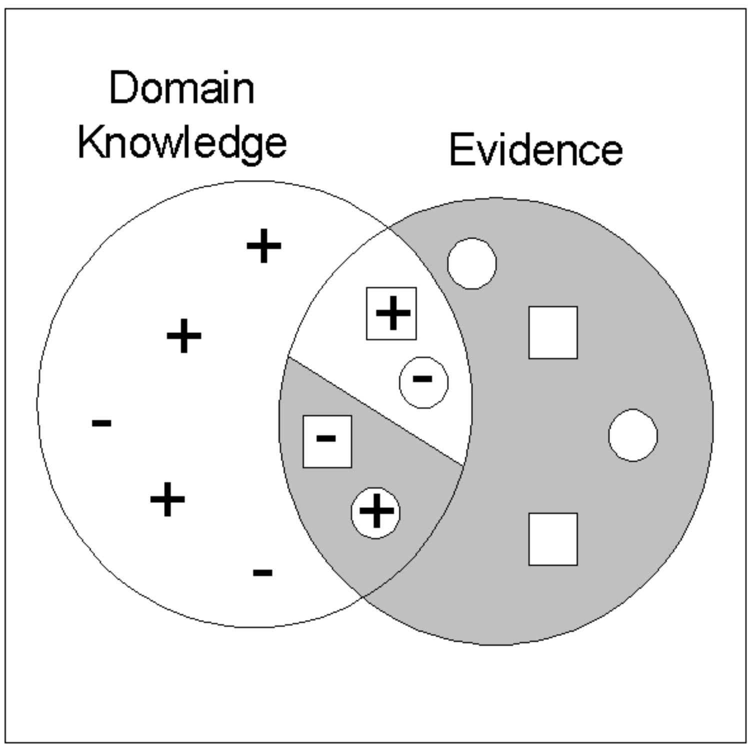
	A	B	C	D	E	F
Transaction 1	1	0	0	1	0	0
■	0	0	1	1	1	0
	0	0	1	1	1	0
■	0	0	1	1	1	0
■	0	1	1	0	1	1
	0	0	1	1	1	0
■	0	0	1	1	1	0
■	0	0	1	1	1	0
	0	0	1	1	1	0
Transaction N	1	0	0	1	0	0
	(a)		(b)		(c)	

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)



+ Pattern expected to be frequent

\- Pattern expected to be infrequent

Pattern found to be frequent

Pattern found to be infrequent

\+ - Expected Patterns

\- + Unexpected Patterns

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

Simpson's Paradox

- Hidden Variables in the data
- Stratification
- Example:
  - College grades for physics
    - Physics major
    - Liberal art major
  - Effect of taking highschool physics

Simpson's Paradox

	HS Physics	None	Improvement
Student	50	5	---
Ave Grade	80	70	10

Table 1. Average college physics grades for students in an engineering program.

	HS Physics	None	Improvement
Student	5	50	---
Ave Grade	95	85	10

Table 2. Average college physics grades for students in a liberal arts program.

Simpson's Paradox

	# Students	Grades	Grade Pts
Engineering	50	80	4000
Lib Arts	5	95	475
Total	55		4475
Average	---	81.4	---

Table 3. Average college physics grades for students who took high school physics.

	# Students	Grades	Grade Pts
Engineering	5	70	350
Lib Arts	50	85	4250
Total			4600
Average		83.6	

Table 4. Average college physics grades for students who didn't take high school physics.

Alternative Frequent Itemsets Algorithm

- FP-Growth Algorithm
- Uses 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 Generation

- Scan the data 1 time to find frequent items
  - Generate 1-itemsets and their support count
  - Discard infrequent items from transactions
  - Sort items in transactions by count, most frequent first
- Second pass
- FP-Tree Generation

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

{A}=2, {B}=2, {C}=2
{D}=2,{E}=1

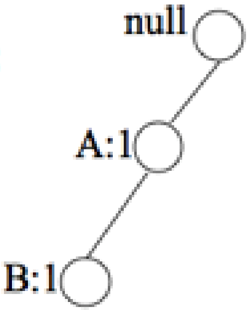
FP-Tree Generation

FP-Tree Generation

- Create a NULL node as the root
- Read one transaction at a time
- Map each transaction into a path in the FP-tree
- Each node corresponds to an item and has a counter field

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

After reading TID=1:



FP-Tree Generation

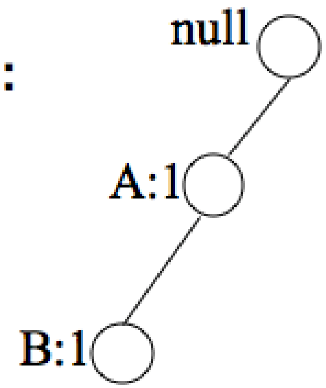
- FP-Tree Generation
  - Create a NULL node as the root
  - Read one transaction at a time
  - Map each transaction into a path in the FP-tree
  - Each node corresponds to an item and has a counter field

  - To facilitate tree traversal, an item header table is built so that each item points to its occurrences in the tree via a chain of node-links
  - If transactions have items in common, their paths can overlap
  - Increment the count for a node (item) if it is shared by many paths

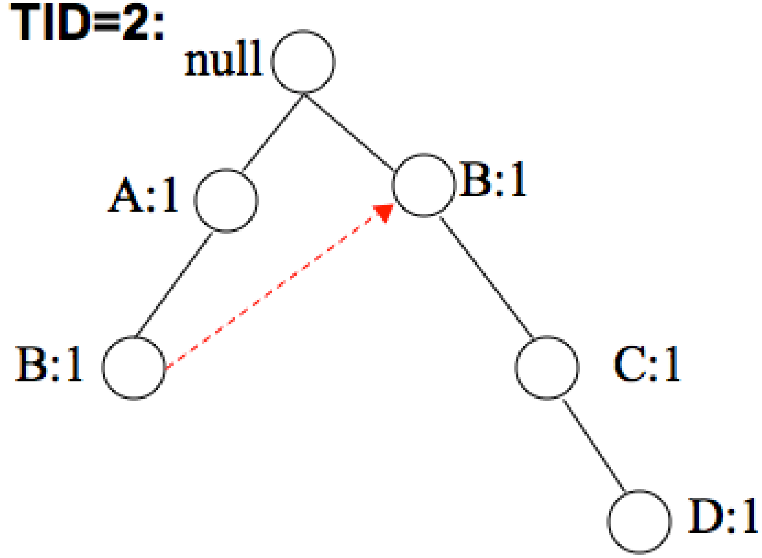
FP-Tree Generation

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:



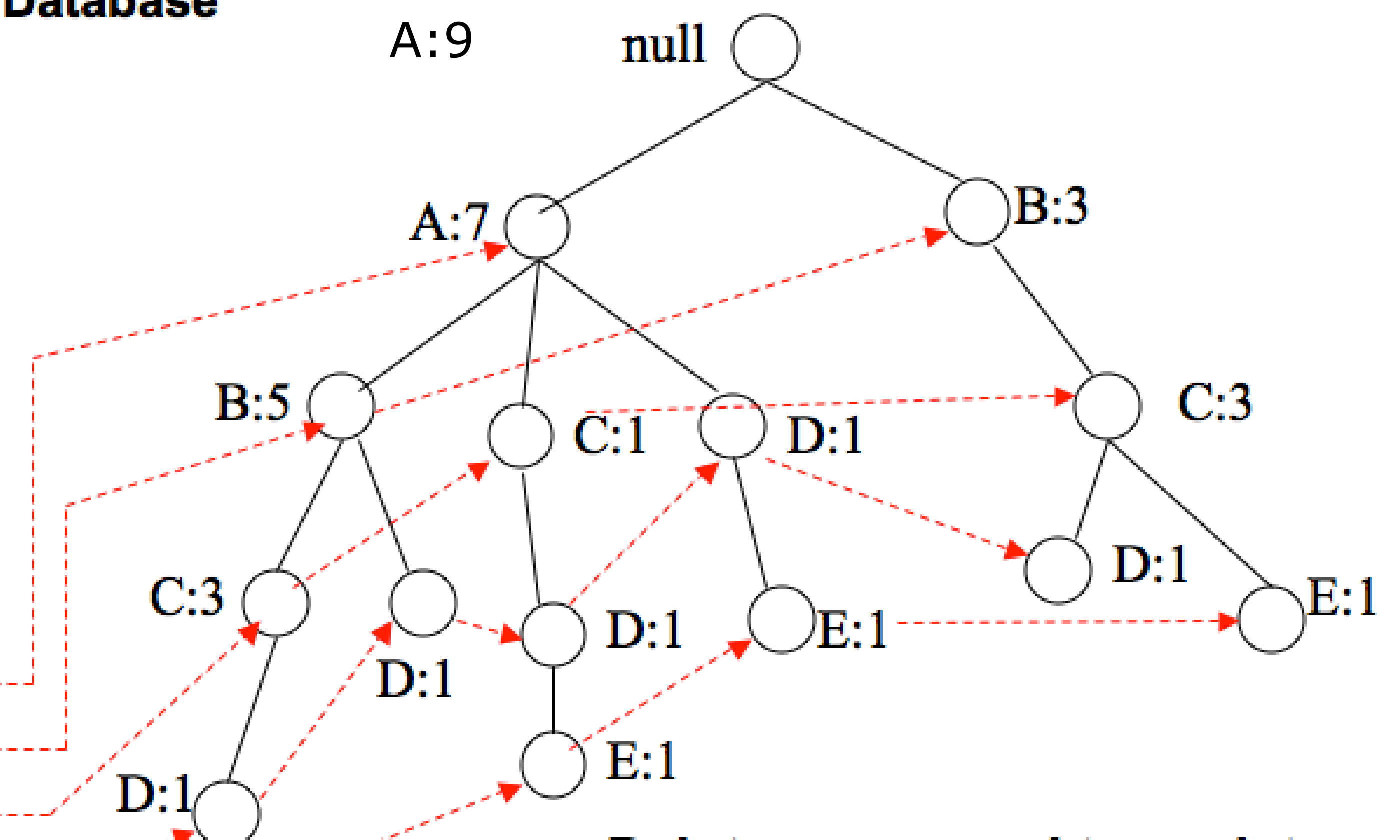
After reading TID=2:



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

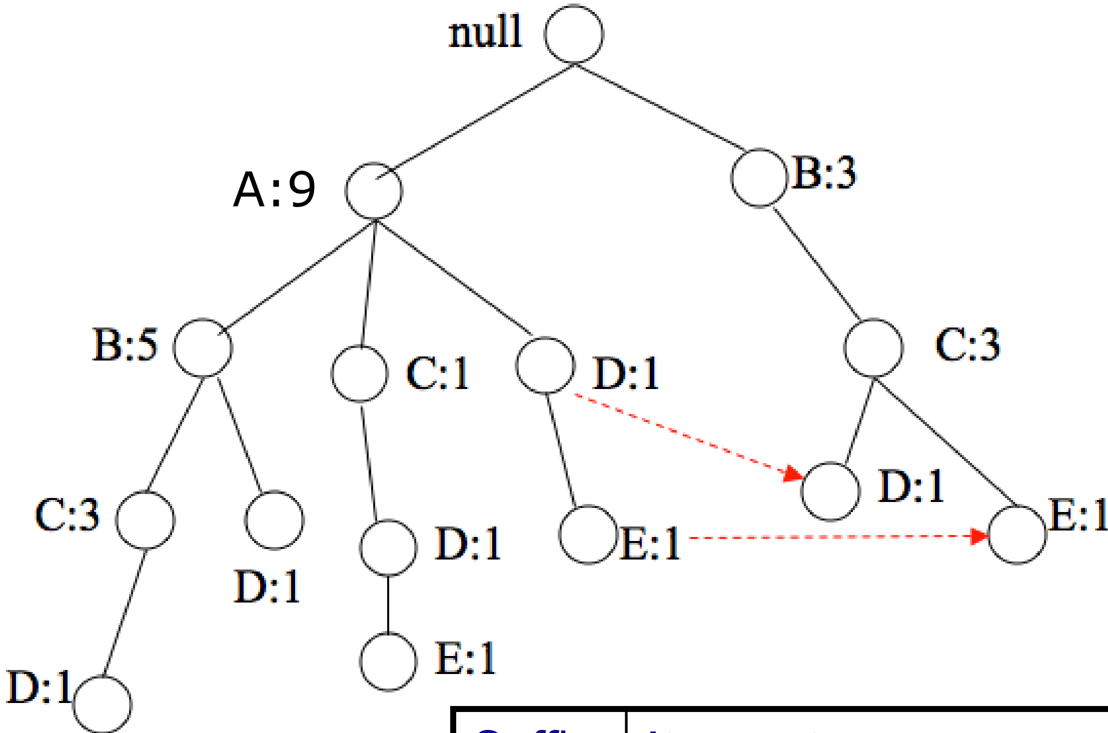
Item	Pointer
A	
B	
C	
D	
E	

Pointers are used to assist frequent itemset generation

FP-Growth

- FP-Growth algorithm generates frequent itemsets from the FP-tree
- Bottom-up
  - Suffix-based approach
- Divide and Conquer

FP-Growth



Use suffixes

Table: Paths in the tree Containing: E,D,C,B,A

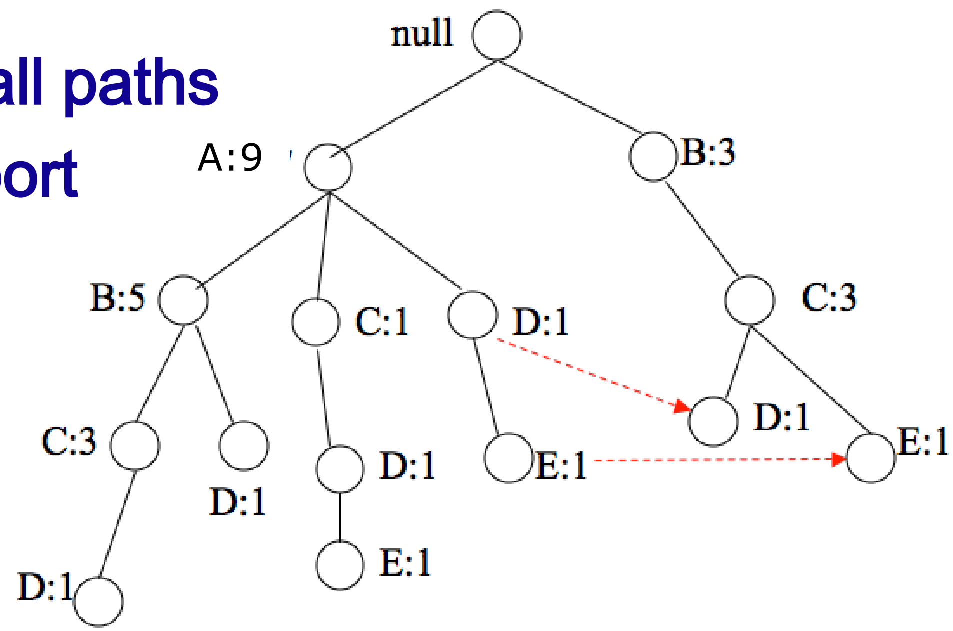
Suffix	Itemsets
e	{e},{de},{a,d,e},{c,e},{b,c,e},{a,c,d},{c,d,e}
d	{d},{c,d},{b,c,d},{a,c,d},{b,d},{a,b,d},{a,d}
c	{c},{b,c},{a,b,c},{a,c}
b	{b},{a,b}
a	{a}

Frequent Itemset Generation

- FP- finds all frequent itemsets ending in a particular suffix
- Divide and conquer
- Find frequent itemsets ending in e
  - Check if e is frequent
  - Subproblems:
    - Frequent itemsets ending in “de”
    - Frequent itemsets ending in “ce”, “be”, “ae”
      - Subproblems:
        - Frequent itemsets ending in “bde”
        - Frequent itemsets ending in “cde”

Frequent Itemset Generation

- Frequent itemsets containing E
- Collect all path containing e
  - Prefix paths
- Add the counts from all paths
- Compare to min support
- Support(e)=3



Frequent Itemset Generation

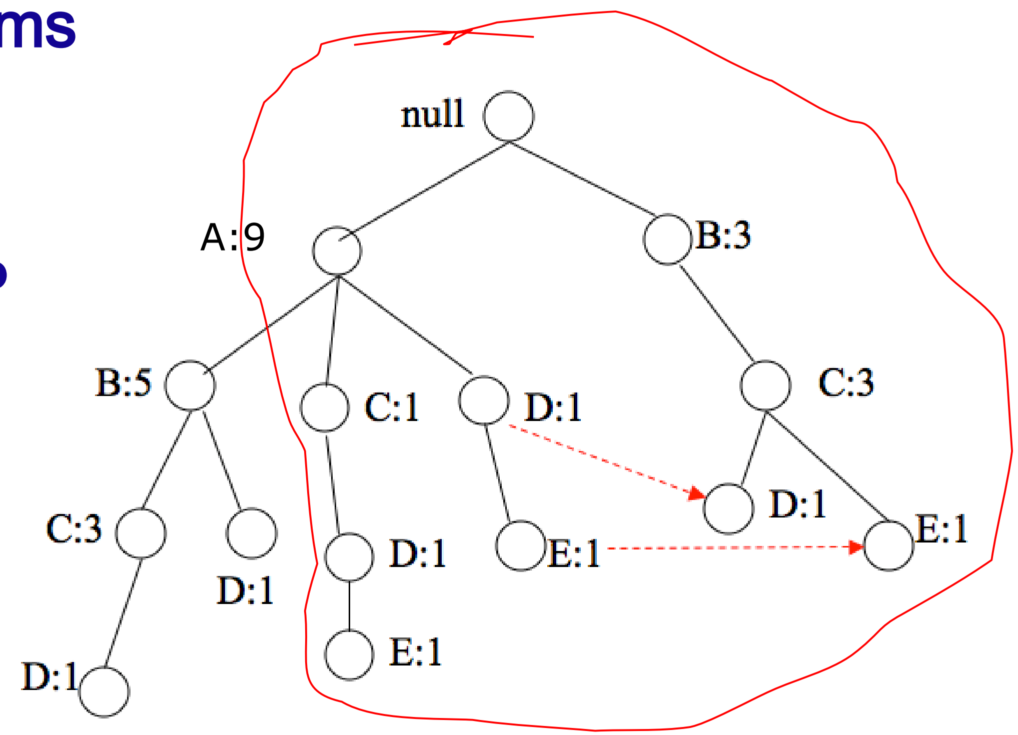
- E is frequent
- Solve the subproblems
  - {de},{be},{ce},{ae}
- Convert the tree

  Into a conditional FP

  tree

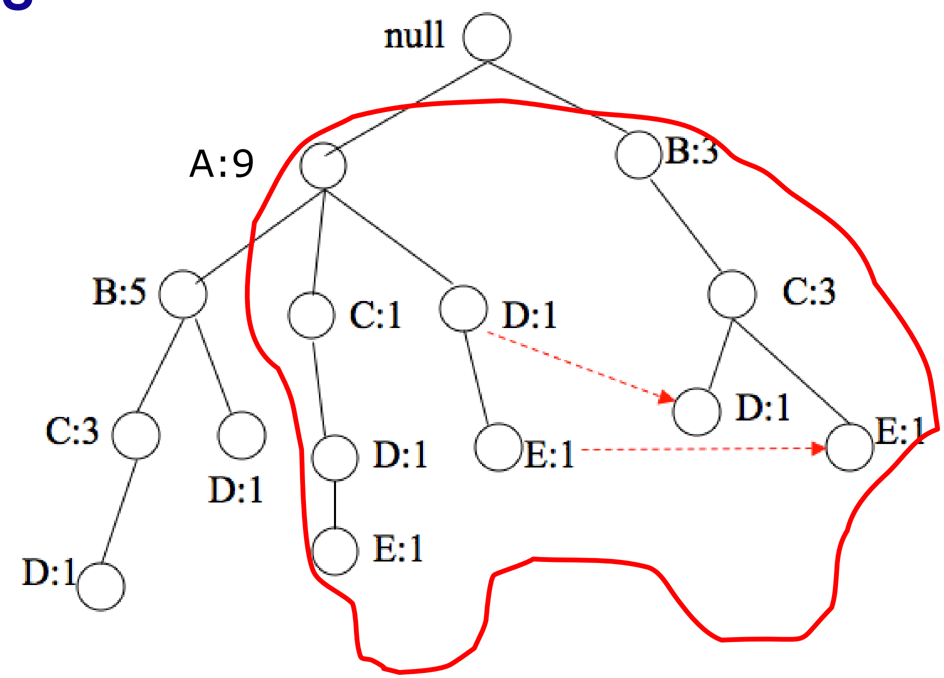
Frequent Itemset Generation

- E is frequent
- Solve the subproblems
  - {de},{be},{ce},{ae}
- Convert the tree
  Into a conditional FP
  tree



Frequent Itemset Generation

- Solve the subproblems
  - {de},{be},{ce},{ae}
- Conditional FP tree
- Update counts
- Remove infrequent items
- b has count of 1
- Eliminate b and {be} subproblem



FP-Growth Performance

- Performance study
  - FP-growth is an order of magnitude faster than Apriori, and is also faster than tree-projection
- Reasoning
  - No candidate generation, no candidate test
  - Use compact data structure
  - Eliminate repeated database scan
  - Basic operation is counting and FP-tree building

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,