

# Association Rules Outline

*Goal:* Provide an overview of basic Association Rule mining techniques

- Association Rules Problem Overview
  - Large/Frequent itemsets
- Association Rules Algorithms
  - Apriori
  - Sampling
  - Partitioning
  - Parallel Algorithms
- Comparing Techniques
- Incremental Algorithms
- Advanced AR Techniques

# Example: Market Basket Data

- Items frequently purchased together:

**Bread  $\Rightarrow$  PeanutButter**

- Uses:
  - Product placement
  - Advertising - Amazon
  - Sales
  - Coupons

# Association Rule Definitions

- ***Set of items:***  $I = \{I_1, I_2, \dots, I_m\}$
- ***Transactions:***  $D = \{t_1, t_2, \dots, t_n\}$ ,  $t_j \subseteq I$
- ***Itemset:***  $\{I_{i1}, I_{i2}, \dots, I_{ik}\} \subseteq I$
- ***Support of an itemset:*** Percentage of transactions which contain that itemset.
- ***Large (Frequent) itemset:*** Itemset whose number of occurrences is above a threshold.

# Association Rules Example

Transaction	Items
$t_1$	Bread,Jelly,PeanutButter
$t_2$	Bread,PeanutButter
$t_3$	Bread,Milk,PeanutButter
$t_4$	Beer,Bread
$t_5$	Beer,Milk

$I = \{ \text{Beer, Bread, Jelly, Milk, PeanutButter} \}$

Support of {Bread,PeanutButter} is 60%

# Association Rule Definitions

- **Association Rule (AR):** implication  $X \Rightarrow Y$  where  $X, Y \subseteq I$  and  $X \cap Y = \emptyset$  ;
- **Support of AR ( $s$ )  $X \Rightarrow Y$ :** Percentage of transactions that contain  $X \cup Y$
- **Confidence of AR ( $\alpha$ )  $X \Rightarrow Y$ :** Ratio of number of transactions that contain  $X \cup Y$  to the number that contain  $X$

# Association Rules Ex (cont'd)

Transaction	Items
$t_1$	Bread,Jelly,PeanutButter
$t_2$	Bread,PeanutButter
$t_3$	Bread,Milk,PeanutButter
$t_4$	Beer,Bread
$t_5$	Beer,Milk

$X \Rightarrow Y$	$s$	$\alpha$
Bread $\Rightarrow$ PeanutButter	60%	75%
PeanutButter $\Rightarrow$ Bread	60%	100%
Beer $\Rightarrow$ Bread	20%	50%
PeanutButter $\Rightarrow$ Jelly	20%	33.3%
Jelly $\Rightarrow$ PeanutButter	20%	100%
Jelly $\Rightarrow$ Milk	0%	0%

# Association Rule Problem

- Given a set of items  $I = \{I_1, I_2, \dots, I_m\}$  and a database of transactions  $D = \{t_1, t_2, \dots, t_n\}$  where  $t_i = \{I_{i1}, I_{i2}, \dots, I_{ik}\}$  and  $I_{ij} \in I$ , the *Association Rule Problem* is to identify all association rules  $X \Rightarrow Y$  with a minimum support and confidence.
- Link Analysis
- **NOTE:** Support of  $X \Rightarrow Y$  is same as support of  $X \cup Y$ .

# Association Rule Techniques

1. Find Large Itemsets.
2. Generate rules from frequent itemsets.

# Algorithm to Generate ARs

## Input:

$D$  // Database of transactions  
 $I$  // Items  
 $L$  // Large itemsets  
 $s$  // Support  
 $\alpha$  // Confidence

## Output:

$R$  // Association Rules satisfying  $s$  and  $\alpha$

## ARGen Algorithm:

$R = \emptyset$ ;

for each  $l \in L$  do

for each  $x \subset l$  such that  $x \neq \emptyset$  and  $x \neq l$  do

if  $\frac{\text{support}(l)}{\text{support}(x)} \geq \alpha$  then

$R = R \cup \{x \Rightarrow (l - x)\}$ ;

# Apriori

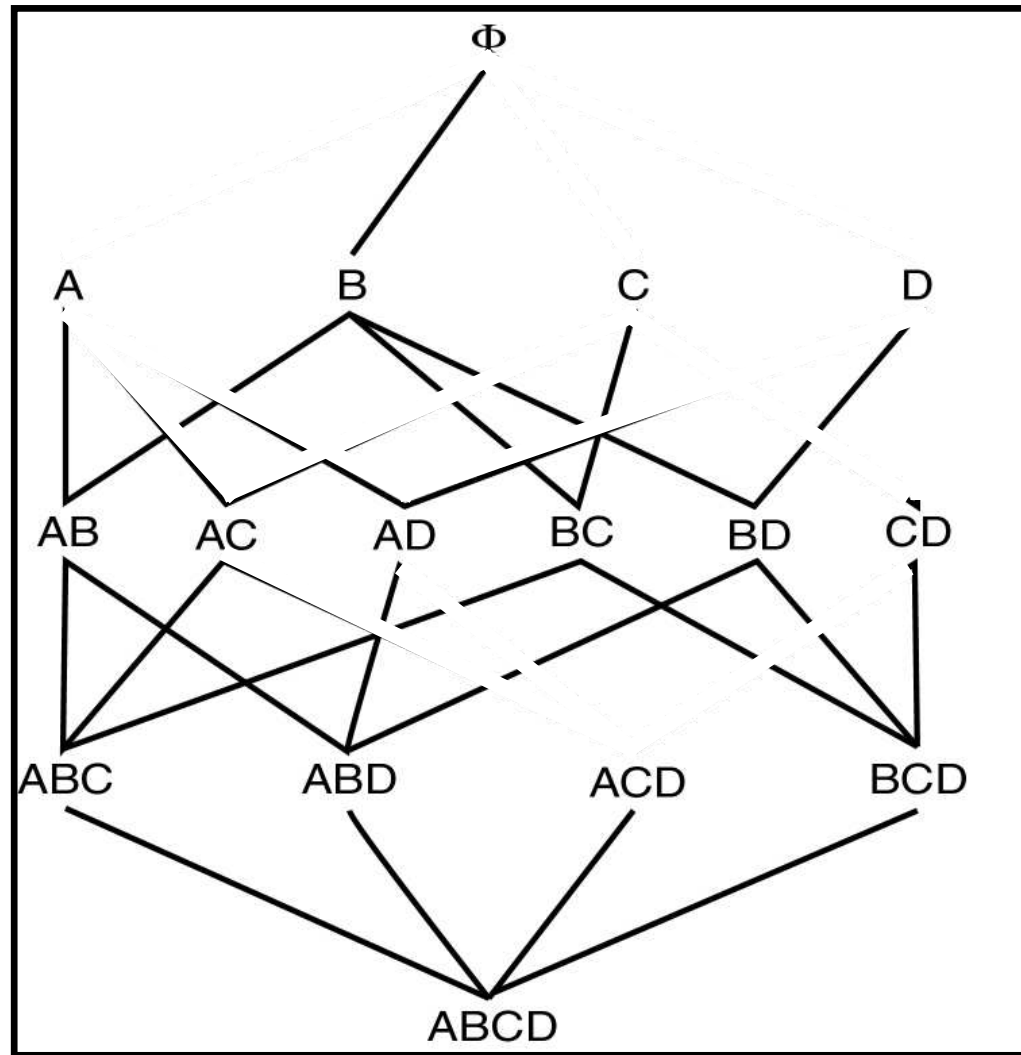
## *Large Itemset Property:*

*Any subset of a large itemset is large.*

## *Contrapositive:*

*If an itemset is not large, none of its supersets are large.*

# Large Itemset Property



# Apriori Ex (cont'd)

Pass	Candidates	Large Itemsets
1	{Beer},{Bread},{Jelly}, {Milk},{PeanutButter}	{Beer},{Bread}, {Milk},{PeanutButter}
2	{Beer,Bread},{Beer,Milk}, {Beer,PeanutButter},{Bread,Milk}, {Bread,PeanutButter},{Milk,PeanutButter}	{Bread,PeanutButter}

$s=30\%$        $\alpha = 50\%$

# Apriori Algorithm

1.  $C_1 =$  Itemsets of size one in  $I$ ;
2. Determine all large itemsets of size 1,  $L_1$ ;
3.  $i = 1$ ;
4. Repeat
5.      $i = i + 1$ ;
6.      $C_i = \text{Apriori-Gen}(L_{i-1})$ ;
7.     Count  $C_i$  to determine  $L_i$ ;
8. until no more large itemsets found;

# Apriori-Gen

- Generate candidates of size  $i+1$  from large itemsets of size  $i$ .
- Approach used: join large itemsets of size  $i$  if they agree on  $i-1$
- May also prune candidates who have subsets that are not large.

# Apriori-Gen Example

<b>Transaction</b>	<b>Items</b>
$t_1$	<b>Blouse</b>
$t_2$	<b>Shoes,Skirt,TShirt</b>
$t_3$	<b>Jeans,TShirt</b>
$t_4$	<b>Jeans,Shoes,TShirt</b>
$t_5$	<b>Jeans,Shorts</b>
$t_6$	<b>Shoes,TShirt</b>
$t_7$	<b>Jeans,Skirt</b>
$t_8$	<b>Jeans,Shoes,Shorts,TShirt</b>
$t_9$	<b>Jeans</b>
$t_{10}$	<b>Jeans,Shoes,TShirt</b>
$t_{11}$	<b>TShirt</b>
$t_{12}$	<b>Blouse,Jeans,Shoes,Skirt,TShirt</b>
$t_{13}$	<b>Jeans,Shoes,Shorts,TShirt</b>
$t_{14}$	<b>Shoes,Skirt,TShirt</b>
$t_{15}$	<b>Jeans,TShirt</b>
$t_{16}$	<b>Skirt,TShirt</b>
$t_{17}$	<b>Blouse,Jeans,Skirt</b>
$t_{18}$	<b>Jeans,Shoes,Shorts,TShirt</b>
$t_{19}$	<b>Jeans</b>
$t_{20}$	<b>Jeans,Shoes,Shorts,TShirt</b>

# Apriori-Gen Example (cont'd)

Scan	Candidates	Large Itemsets
1	{Blouse},{Jeans},{Shoes}, {Shorts},{Skirt},{TShirt}	{Jeans},{Shoes},{Shorts} {Skirt},{Tshirt}
2	{Jeans,Shoes},{Jeans,Shorts},{Jeans,Skirt}, {Jeans,TShirt},{Shoes,Shorts},{Shoes,Skirt}, {Shoes,TShirt},{Shorts,Skirt},{Shorts,TShirt}, {Skirt,TShirt}	{Jeans,Shoes},{Jeans,Shorts}, {Jeans,TShirt},{Shoes,Shorts}, {Shoes,TShirt},{Shorts,TShirt}, {Skirt,TShirt}
3	{Jeans,Shoes,Shorts},{Jeans,Shoes,TShirt}, {Jeans,Shorts,TShirt},{Jeans,Skirt,TShirt}, {Shoes,Shorts,TShirt},{Shoes,Skirt,TShirt}, {Shorts,Skirt,TShirt}	{Jeans,Shoes,Shorts}, {Jeans,Shoes,TShirt}, {Jeans,Shorts,TShirt}, {Shoes,Shorts,TShirt}
4	{Jeans,Shoes,Shorts,TShirt}	{Jeans,Shoes,Shorts,TShirt}
5	$\emptyset$	$\emptyset$

# Apriori Adv/Disadv

- ***Advantages:***
  - Uses large itemset property.
  - Easily parallelized. How?
  - Easy to implement.
- ***Disadvantages:***
  - Assumes transaction database is memory resident.
  - Requires up to  $m$  database scans.

# Partitioning

- Divide database into partitions  $D^1, D^2, \dots, D^p$
- Apply Apriori to each partition
- Any large itemset must be large in at least one partition.

# Partitioning Algorithm

1. Divide  $D$  into partitions  $D^1, D^2, \dots, D^p$ ;
2. For  $I = 1$  to  $p$  do
3.      $L^i = \text{Apriori}(D^i)$ ;
4.  $C = L^1 \cup \dots \cup L^p$ ;
5. Count  $C$  on  $D$  to generate  $L$ ;

# Partitioning Example

	Transaction	Items
D <sup>1</sup>	$t_1$	Bread, Jelly, Peanut Butter
	$t_2$	Bread, Peanut Butter
D <sup>2</sup>	$t_3$	Bread, Milk, Peanut Butter
	$t_4$	Beer, Bread
	$t_5$	Beer, Milk

S=10%

$L^1 = \{ \{ \text{Bread} \}, \{ \text{Jelly} \},$   
 $\{ \text{Peanut Butter} \},$   
 $\{ \text{Bread, Jelly} \},$   
 $\{ \text{Bread, Peanut Butter} \},$   
 $\{ \text{Jelly, Peanut Butter} \},$   
 $\{ \text{Bread, Jelly, Peanut Butter} \} \}$

$L^2 = \{ \{ \text{Bread} \}, \{ \text{Milk} \},$   
 $\{ \text{Peanut Butter} \}, \{ \text{Bread, Milk} \},$   
 $\{ \text{Bread, Peanut Butter} \}, \{ \text{Milk,}$   
 $\text{Peanut Butter} \},$   
 $\{ \text{Bread, Milk, Peanut Butter} \},$   
 $\{ \text{Beer} \}, \{ \text{Beer, Bread} \},$   
 $\{ \text{Beer, Milk} \} \}$

# Partitioning Adv/Disadv

- ***Advantages:***

- Adapts to available main memory
- Easily parallelized
- Maximum number of database scans is two.

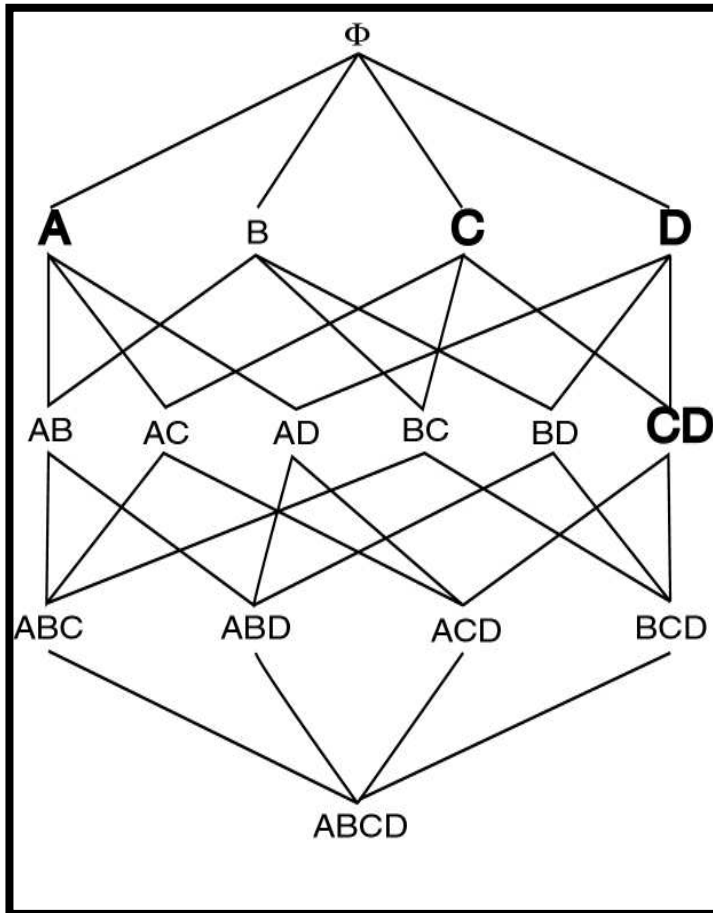
- ***Disadvantages:***

- May have many candidates during second scan.

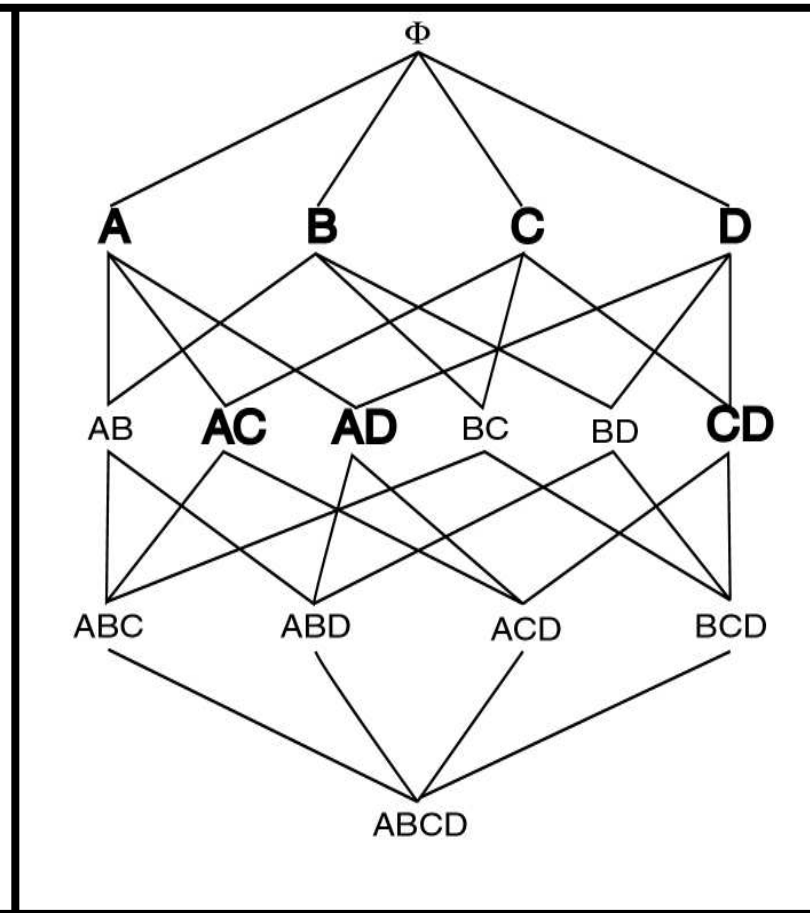
# Sampling

- Large databases
- Sample the database and apply Apriori to the sample.
- *Potentially Large Itemsets (PL)*: Large itemsets from sample
- *Negative Border (BD<sup>-</sup>)*:
  - Generalization of Apriori-Gen applied to itemsets of varying sizes.
  - Minimal set of itemsets which are not in PL, but whose every subset is in PL.

# Negative Border Example



PL



$PL \cup BD^-(PL)$

# Sampling Algorithm

1.  $D_s$  = sample of Database  $D$ ;
2.  $PL$  = Large itemsets in  $D_s$  using  $\alpha MinSup$ ;
3.  $C = PL \cup BD^-(PL)$ ;
4. Count  $C$  in  $D_s$ ;
5.  $ML$  = large itemsets in  $BD^-(PL)$ ;
6. If  $ML = \emptyset$  then done
7.     else  $C$  = repeated application of  $BD^-$ ;
8.     Count  $C$  in Database;

# Sampling Example

- Find AR assuming  $\text{MinSup} = 20\%$
- $D_s = \{ t_1, t_2 \}$
- $\alpha \text{MinSup} = 10\%$
- $\text{PL} = \{ \{ \text{Bread} \}, \{ \text{Jelly} \}, \{ \text{PeanutButter} \}, \{ \text{Bread, Jelly} \}, \{ \text{Bread, PeanutButter} \}, \{ \text{Jelly, PeanutButter} \}, \{ \text{Bread, Jelly, PeanutButter} \} \}$
- $\text{BD}^-(\text{PL}) = \{ \{ \text{Beer} \}, \{ \text{Milk} \} \}$
- $\text{ML} = \{ \{ \text{Beer} \}, \{ \text{Milk} \} \}$
- Repeated application of  $\text{BD}^-$  generates all remaining itemsets

# Sampling Adv/Disadv

- ***Advantages:***
  - Reduces number of database scans to one in the best case and two in worst.
  - Scales better.
- ***Disadvantages:***
  - Potentially large number of candidates in second pass

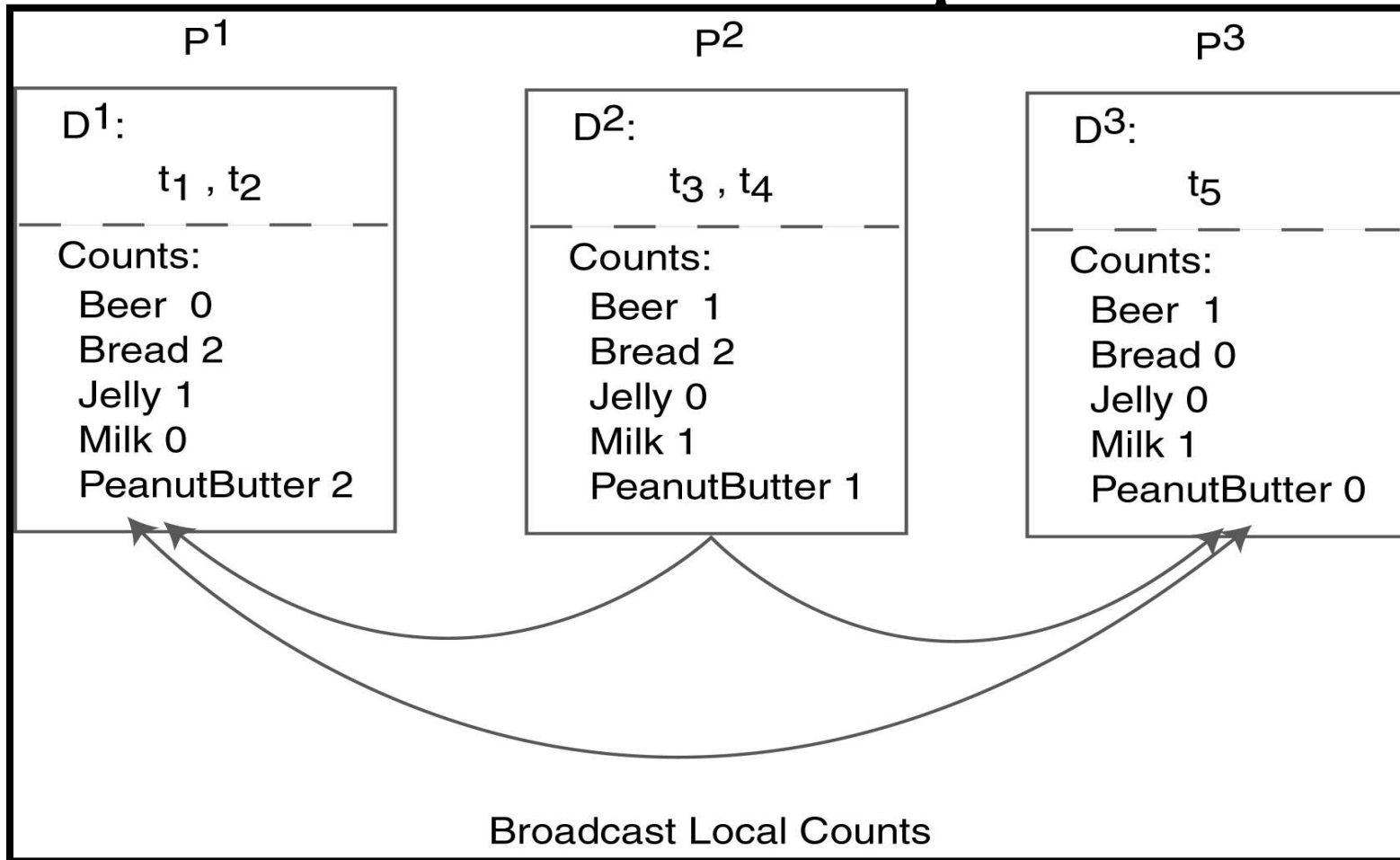
# Parallelizing AR Algorithms

- Based on Apriori
- Techniques differ:
  - What is counted at each site
  - How data (transactions) are distributed
- Data Parallelism
  - Data partitioned
  - Count Distribution Algorithm
- Task Parallelism
  - Data and candidates partitioned
  - Data Distribution Algorithm

# Count Distribution Algorithm(CDA)

1. Place data partition at each site.
2. In Parallel at each site do
3.      $C_1 =$  Itemsets of size one in  $I$ ;
4.     Count  $C_1$ ;
5.     Broadcast counts to all sites;
6.     Determine global large itemsets of size 1,  $L_1$ ;
7.      $i = 1$ ;
8.     Repeat
9.          $i = i + 1$ ;
10.          $C_i =$  Apriori-Gen( $L_{i-1}$ );
11.         Count  $C_i$ ;
12.         Broadcast counts to all sites;
13.         Determine global large itemsets of size  $i$ ,  $L_i$ ;
14.     until no more large itemsets found;

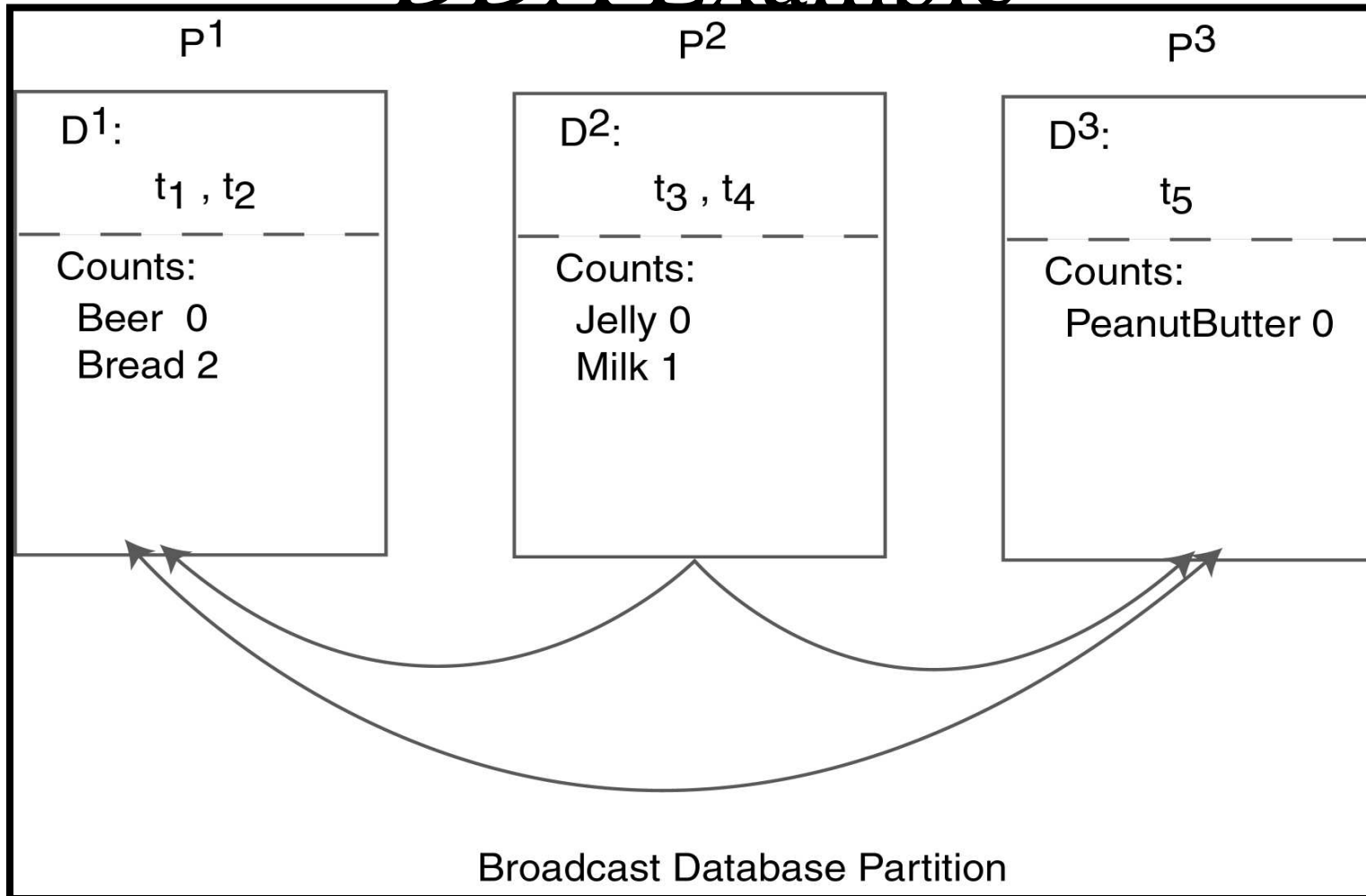
# CDA Example



# Data Distribution Algorithm(DDA)

1. Place data partition at each site.
2. In Parallel at each site do
3.     Determine local candidates of size 1 to count;
4.     Broadcast local transactions to other sites;
5.     Count local candidates of size 1 on all data;
6.     Determine large itemsets of size 1 for local candidates;
7.     Broadcast large itemsets to all sites;
8.     Determine  $L_1$ ;
9.      $i = 1$ ;
10.    Repeat
11.      $i = i + 1$ ;
12.      $C_i = \text{Apriori-Gen}(L_{i-1})$ ;
13.     Determine local candidates of size  $i$  to count;
14.     Count, broadcast, and find  $L_i$ ;
15.    until no more large itemsets found;

# DDA Example



# Comparison of AR Techniques

Partitioning	Scans	Data Structure	Parallelism
Apriori	$m + 1$	hash tree	none
Sampling	2	not specified	none
Partitioning	2	hash table	none
CDA	$m + 1$	hash tree	data
DDA	$m + 1$	hash tree	task

# Incremental Association Rules

- Generate ARs in a dynamic database.
- Problem: algorithms assume static database
- Objective:
  - Know large itemsets for  $D$
  - Find large itemsets for  $D \cup \{\Delta D\}$
- Must be large in either  $D$  or  $\Delta D$
- Save  $L_i$  and counts

# Note on ARs

- Many applications outside market basket data analysis
  - Prediction (telecom switch failure)
  - Web usage mining
- Many different types of association rules
  - Temporal
  - Spatial
  - Causal

# Advanced AR Techniques

- Generalized Association Rules
  - Need is-a hierarchy
- Multiple-Level Association Rules
- Quantitative Association Rules
- Using multiple minimum supports

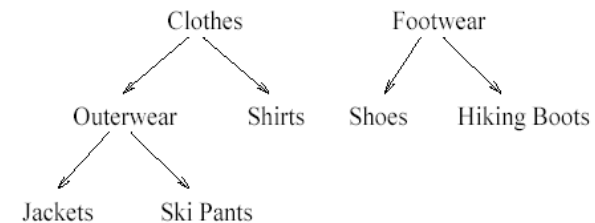


Figure 1: Example of a Taxonomy

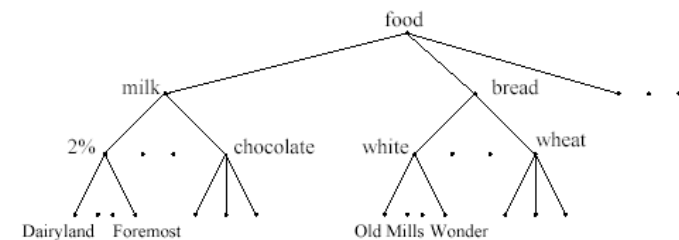


Figure 1: A taxonomy for the relevant data items

# Measuring Quality of Rules

- Support (Joint probability)
- Confidence (Conditional probability)
- Interest (Essentially a measure of independence)
- Conviction (Asymmetrical interest measure)
  - $A \rightarrow B$  rewritten using the implication elimination of P.L?
- Chi Squared Test
  - Create a contingency table, test for independence