Chapter 5: Mining Frequent Patterns, Association and Correlations

- Basic concepts and a road map

- Efficient and scalable frequent itemset mining methods

- Mining various kinds of association rules

- From association mining to correlation analysis

- Constraint-based association mining

- Summary

What Is Frequent Pattern Analysis?

- Frequent pattern: a pattern (a set of items, subsequences, substructures, etc.) that occurs frequently in a data set

- First proposed by Agrawal, Imielinski, and Swami [AIS93] in the context of frequent itemsets and association rule mining

- Motivation: Finding inherent regularities in data

  - What products were often purchased together?— Beer and diapers?!

  - What are the subsequent purchases after buying a PC?

  - What kinds of DNA are sensitive to this new drug?

- Applications

  - Basket data analysis, cross-marketing, catalog design, sale campaign analysis, Web log (click stream) analysis, and DNA sequence analysis.

Why Is Freq. Pattern Mining Important?

- Discloses an intrinsic and important property of data sets

- Forms the foundation for many essential data mining tasks

  - Association, correlation, and causality analysis

  - Sequential, structural (e.g., sub-graph) patterns

  - Pattern analysis in spatiotemporal, multimedia, time-series, and stream data

  - Classification: associative classification

  - Cluster analysis: frequent pattern-based clustering

  - Semantic data compression

  - Broad applications

Basic Concepts: Frequent Patterns and Association Rules

Transaction-id	Items bought
10	A, B, D
20	A, C, D
30	A, D, E
40	B, E, F
50	B, C, D, E, F

Customer buys both

Customer buys diaper

Customer buys beer

- Itemset X = $\{x_1, ..., x_k\}$

- Find all the association rules $X \rightarrow Y$ with minimum support and confidence

  - **support**, *s*, probability that a transaction contains $X \cup Y$

  - **confidence**, *c*, conditional probability that a transaction having X also contains *Y*
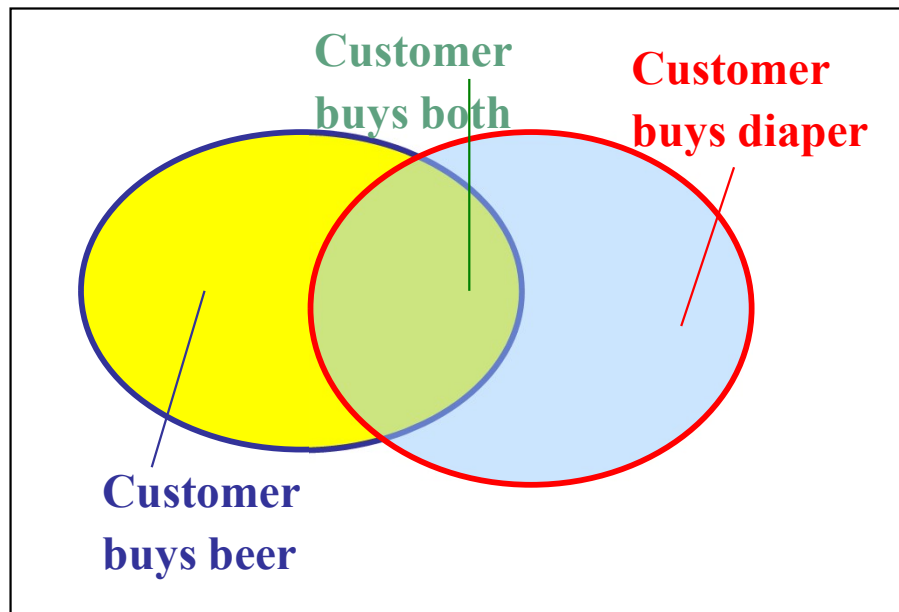
Let sup_{min} = 50%, $conf_{min}$ = 50%
Freq. Pat.: {A:3, B:3, D:4, E:3, AD:3}
Association rules:

  $A \rightarrow D$ (60%, 100%)
  $D \rightarrow A$ (60%, 75%)

Closed Patterns and Max-Patterns

- A long pattern contains a combinatorial number of sub-patterns, e.g., $\{a_1, ..., a_{100}\}$ contains $\binom{100}{1} + \binom{100}{2} + ... + \binom{100}{100} = 2^{100} - 1 = 1.27*10^{30}$ sub-patterns!

- Solution: *Mine closed patterns and max-patterns instead*

- An itemset X is closed if X is *frequent* and there exists *no super-pattern* Y ⊃ X, *with the same support* as X (proposed by Pasquier, et al. @ ICDT'99)

- An itemset X is a max-pattern if X is frequent and there exists no frequent super-pattern Y ⊃ X (proposed by Bayardo @ SIGMOD'98)

- Closed pattern is a lossless compression of freq. patterns
  - Reducing the # of patterns and rules

Closed Patterns and Max-Patterns

- Exercise. DB = $\{<a_1, ..., a_{100}>, <a_1, ..., a_{50}>\}$
  - Min_sup = 1.
- What is the set of all frequent patterns?
  - $2^{100} - 1$
- What is the set of closed itemset?
  - $<a_1, ..., a_{100}>$: 1
  - $<a_1, ..., a_{50}>$: 2
- What is the set of max-pattern?
  - $<a_1, ..., a_{100}>$: 1

Chapter 5: Mining Frequent Patterns, Association and Correlations

- Basic concepts and a road map

- Efficient and scalable frequent itemset mining methods ⬅

- Mining various kinds of association rules

- From association mining to correlation analysis

- Constraint-based association mining

- Summary

Scalable Methods for Mining Frequent Patterns

- The downward closure property of frequent patterns
  - Any subset of a frequent itemset must be frequent
  - If **{beer, diaper, nuts}** is frequent, so is **{beer, diaper}**
  - i.e., every transaction having {beer, diaper, nuts} also contains {beer, diaper}
- Scalable mining methods: Three major approaches
  - Apriori (Agrawal & Srikant@VLDB'94)
  - Freq. pattern growth (FPgrowth—Han, Pei & Yin @SIGMOD'00)
  - Vertical data format approach (Charm—Zaki & Hsiao @SDM'02)

Apriori: A Candidate Generation-and-Test Approach

- **Apriori pruning principle:** If there is any itemset which is infrequent, its superset should not be generated/tested! (Agrawal & Srikant @VLDB'94, Mannila, et al. @ KDD' 94)

- Method:
  - Initially, scan DB once to get frequent 1-itemset
  - Generate candidate itemsets of length (k+1) from frequent itemsets of length k
  - Test the candidates against DB
  - Terminate when no frequent or candidate set can be generated

The Apriori Algorithm—An Example

$Sup_{min} = 2$

Database TDB

Tid	Items
10	A, C, D
20	B, C, E
30	A, B, C, E
40	B, E

1st scan

C_1

Itemset	sup
{A}	2
{B}	3
{C}	3
{D}	1
{E}	3

L_1

Itemset	sup
{A}	2
{B}	3
{C}	3
{E}	3

C_2

Itemset	sup
{A, B}	1
{A, C}	2
{A, E}	1
{B, C}	2
{B, E}	3
{C, E}	2

2nd scan

C_2

Itemset
{A, B}
{A, C}
{A, E}
{B, C}
{B, E}
{C, E}

L_2

Itemset	sup
{A, C}	2
{B, C}	2
{B, E}	3
{C, E}	2

C_3

Itemset
{B, C, E}

3rd scan

L_3

Itemset	sup
{B, C, E}	2
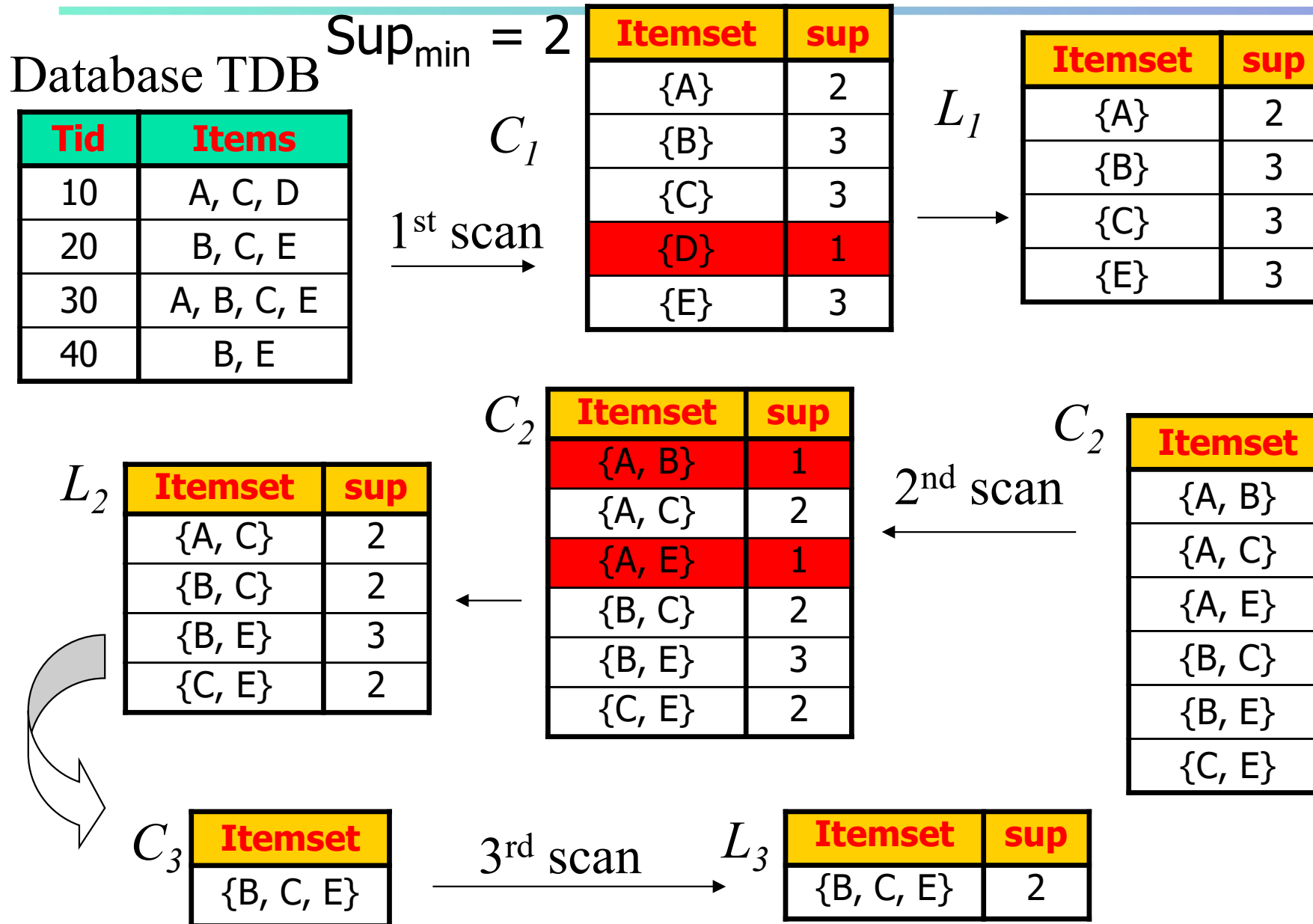
The Apriori Algorithm

- Pseudo-code:

C_k: Candidate itemset of size k

L_k : frequent itemset of size k

L_1 = {frequent items};

for (k = 1; L_k !=$\varnothing$; k++) **do begin**

C_{k+1} = candidates generated from L_k;

for each transaction t in database do

increment the count of all candidates in C_{k+1} that are contained in t

L_{k+1} = candidates in C_{k+1} with min_support

end

return $\cup_k L_k$;

Data Mining: Concepts and Techniques

Important Details of Apriori

- How to generate candidates?
  - Step 1: self-joining L_k (what is a join?)
  - Step 2: pruning
- How to count supports of candidates?
- Example of Candidate-generation
  - L_3={abc, abd, acd, ace, bcd}
  - Self-joining: L_3*L_3
    - abcd from abc and abd
    - acde from acd and ace
  - Pruning:
    - acde is removed because ade is not in L_3
  - C_4={abcd}

Challenges of Frequent Pattern Mining

- Challenges

  - Multiple scans of a transaction database

  - Huge number of candidates

  - Tedious workload of support counting for candidates

- Improving Apriori: general ideas

  - Reduce the number of transaction database scans

  - Shrink the number of candidates

  - Facilitate support counting of candidates