Why Data Mining?—Potential Applications

- Data analysis and decision support
  - Market analysis and management
    - Target marketing, customer relationship management (CRM), market basket analysis, cross selling, market segmentation
  - Risk analysis and management
    - Forecasting, customer retention, improved underwriting, quality control, competitive analysis
  - Fraud detection and detection of unusual patterns (outliers)
- Other Applications
  - Text mining (news group, email, documents) and Web mining
  - Stream data mining
  - Bioinformatics and Bio Data Analysis

Data Mining Tasks

Data mining tasks are generally divided into two major categories:

- Predictive tasks [Use some attributes to predict unknown or future values of other attributes.]
  - Classification
  - Regression
  - Deviation Detection
- Descriptive tasks [Find human-interpretable patterns that describe the data.]
  - Association Discovery
  - Clustering

Major Data Mining Tasks

- Classification: Predicting an item class
- Association Rule Discovery: descriptive
- Clustering: descriptive, finding groups of items
- Sequential Pattern Discovery: descriptive
- Deviation Detection: predictive, finding changes
- Forecasting: predicting a parameter value
- Description: describing a group
- Link analysis: finding relationships and associations

Classification:Definition

- Given a collection of records(training set)
  - Each record contains a set of attributes, one of the attributes is the class.

- Find a model for class attribute as a function of the values of other attributes.

- Goal: previously unseen records should be assigned a class as accurately as possible.

  - A test set is used to determine the accuracy of the model. Usually, the given data set is divided into training and test sets, with training set used to build the model and test set up is used to validated it.

Classification: Application

- Direct Marketing
  - Goal: Reduce cost of mailing by targeting a set of customers likely to buy a new cell-phone product.
  - Approach:
    - Use the data for a similar product introduced before.
    - We know which customers decided to buy and which decided otherwise. This {buy, don’t buy} decision forms the class attribute.
    - Collect various demographic, lifestyle, and company-interaction related information about all such customers.
      - Type of business, where they stay, how much they earn, etc.
    - Use this information as input attributes to learn a classifier model.

Classification

- A sample table

Age	Smoke	Risk
20	No	Low
25	Yes	High
44	Yes	High
18	No	Low
55	No	High
35	No	Low

To identify the risk
of a group of insurance
Applicants.

The class here are:
Risk = Low
Risk = High

Classification

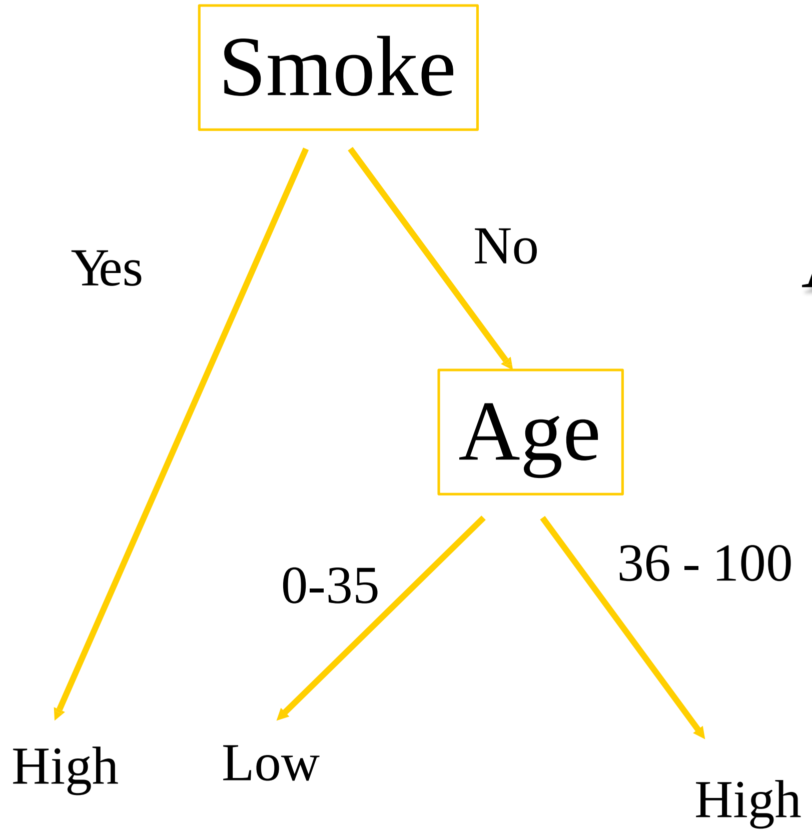
- The following techniques could be used:-
  - Decision Tree
  - Association rule
  - Apriori Algorithm
  - Bayesian classifiers

Decision Tree

- A widely used technique for classification.
- Each leaf node of the tree has an associated class.
- Each internal node has a predicate(or more generally, a function) associated with it.
- To classify a new instance, we start at the root, and traverse the tree to reach a leaf; at an internal node we evaluate the predicate(or function) on the data instance, to find which child to go to.
- A series of nested if/then rules

Decision Tree

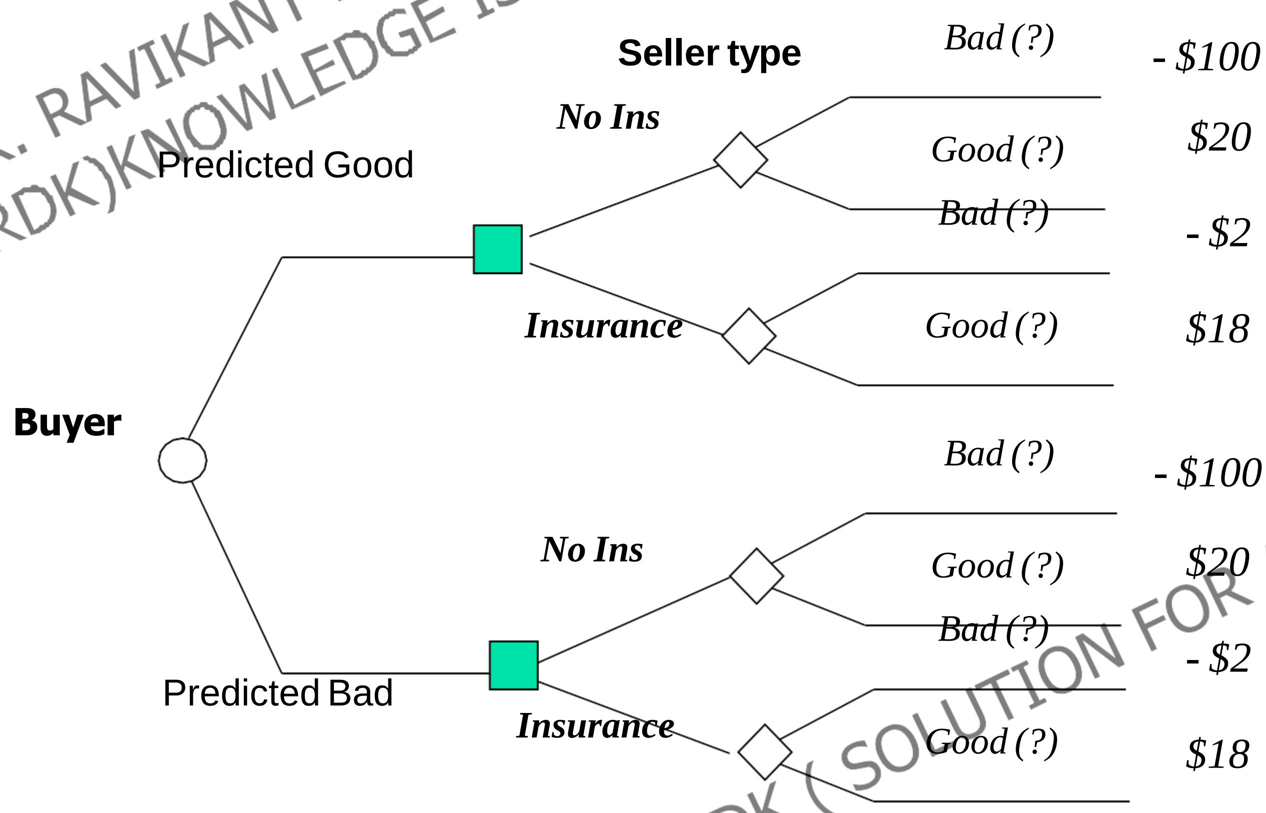
Insurance Risk



Age	Smoke	Risk
20	No	Low
25	Yes	High
44	Yes	High
18	No	Low
55	No	High
35	No	Low

Decision Tree

Extended from the previous online trading question



Questions:

1. Given the suggestion What is your decision?
2. What is the probability wrt the decision you made?
3. How do you estimate The accuracy of a prediction?

Benefits of Decision Tree

- Understandable
- Relatively fast
- Easy to translate to SQL queries

Market Basket Analysis

- Consider shopping cart filled with several items
- Market basket analysis tries to answer the following questions:
  - Who makes purchases?
  - What do customers buy together?
  - In what order do customers purchase items?

Market Basket Analysis

Given:

- A database of customer transactions
- Each transaction is a set of items

- Example: Transaction with TID 111 contains items {Pen, Ink, Milk, Juice}

TID	CID	Date	Item	Qty
111	201	5/1/99	Pen	2
111	201	5/1/99	Ink	1
111	201	5/1/99	Milk	3
111	201	5/1/99	Juice	6
112	105	6/3/99	Pen	1
112	105	6/3/99	Ink	1
112	105	6/3/99	Milk	1
113	106	6/5/99	Pen	1
113	106	6/5/99	Milk	1
114	201	7/1/99	Pen	2
114	201	7/1/99	Ink	2
114	201	7/1/99	Juice	4

Market Basket Analysis (Contd.)

- Coocurrences
  - 80% of all customers purchase items X, Y and Z together.
- Association rules
  - 60% of all customers who purchase X and Y also buy Z.
- Sequential patterns
  - 60% of customers who first buy X also purchase Y within three weeks.

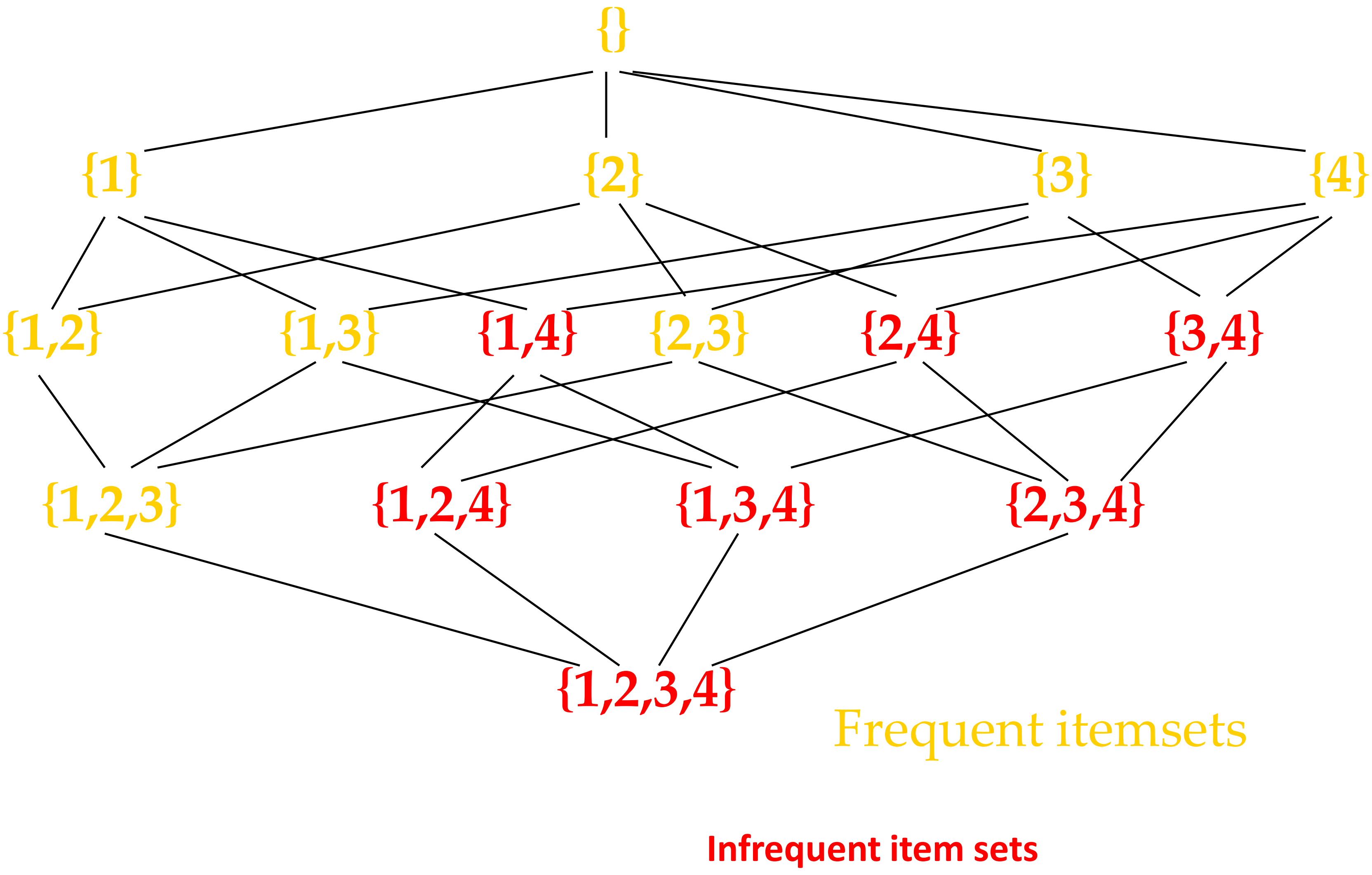
Market Basket Analysis: Applications

- Sample Applications
  - Direct marketing
  - Fraud detection for medical insurance
  - Floor/shelf planning
  - Web site layout
  - Cross-selling

Applications of Frequent Itemsets

- Market Basket Analysis
- Association Rules
- Classification (especially: text, rare classes)
- Seeds for construction of Bayesian Networks
- Web log analysis
- Collaborative filtering

Frequent Itemsets



Association Rules

- If-then rules about the contents of baskets.
- $\{i_1, i_2,\ldots,i_k\} \rightarrow j$ means: "if a basket contains all of $i_1,\ldots,i_k$ then it is likely to contain j.
- Confidence of this association rule is the probability of j given $i_1,\ldots,i_k$.

Example

+ B1 = {m, c, b}
B2 = {m, p, j}

– B3 = {m, b}
B4 = {c, j}

– B5 = {m, p, b}
+ B6 = {m, c, b, j}

B7 = {c, b, j}
B8 = {b, c}

- An association rule: $\{m, b\} \rightarrow c$.
  - Confidence = 2/4 = 50%.

Interest

- The interest of an association rule is the absolute value of the amount by which the confidence differs from what you would expect, were items selected independently of one another.

Example

B1 = {m, c, b}
B2 = {m, p, j}
B3 = {m, b}
B4 = {c, j}
B5 = {m, p, b}
B6 = {m, c, b, j}
B7 = {c, b, j}
B8 = {b, c}

- For association rule {m, b} → c, item c appears in 5/8 of the baskets.
- Interest = | 2/4 - 5/8 | = 1/8 =0.125 --- not very interesting. , |-3|=3=|3|

Associations

- $I = \{i_1, i_2, \ldots i_m\}$: a set of literals, called items.
- Transaction d: a set of items such that $d \subseteq I$
- Database D: a set of transactions
- A transaction d contains X, a set of some items in L, if $X \subseteq d$.
- An association rule is an implication of the form $X \Rightarrow Y$, where $X, Y \subset I$.

Association Rule

- Used to find all rules in a basket data
- Basket data also called transaction data
- analyze how items purchased by customers in a shop are related
- discover all rules that have:-
  - support greater than minsup specified by user
  - confidence greater than minconf specified by user
- Example of transaction data:-
  - CD player, music's CD, music's book
  - CD player, music's CD
  - music's CD, music's book
  - CD player

Association Rule

- Let $I = \{i_1, i_2, \ldots i_m\}$ be a total set of items
  D a set of transactions

  d is one transaction consists of a set of items
  - $d \subseteq \mathbf{I}$
- Association rule:-
  - $X \rightarrow Y$ where $X \subset I$,$Y \subset I$ and $X \cap Y = \varnothing$
  - support = (#of transactions contain $X \cup Y$) / D
  - confidence = (#of transactions contain $X \cup Y$) / #of transactions contain X

Association Rule

- Example of transaction data:-
  - CD player, music's CD, music's book
  - CD player, music's CD
  - music's CD, music's book
  - CD player
- I = {CD player, music's CD, music's book}
- D = 4
- #of transactions contain both CD player, music's CD =2
- #of transactions contain CD player =3
- CD player ➔ music's CD (sup=2/4 , conf =2/3)

Association Rule

- How are association rules mined from large databases ?
- Two-step process:-
  - find all frequent item sets
  - generate strong association rules from frequent item sets

What Is Frequent Pattern Analysis?

- Frequent pattern: a pattern (a set of items, subsequences, substructures, etc.) that occurs frequently in a data set
- Applications
  - Basket data analysis, cross-marketing, catalog design, sale campaign analysis, Web log (click stream) analysis, and DNA sequence analysis.

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

Frequent Itemsets

- C_1 = all items
- L_1 = those counted on first pass to be frequent.
- C_2 = pairs, both chosen from L_1.
- In general, C_k = k –tuples each k –1 of which is in L_{k-1}.
- L_k = those candidates with support $\geq$ s.

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

2nd scan

C_2

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

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 $\cup_k L_k$;

Important Details of Apriori

- How to generate candidates?
  - Step 1: self-joining L_k
  - 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}

The Apriori Algorithm — Example

Database D

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

Scan D $\longrightarrow$ C_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{4}	1
{5}	3

$\longrightarrow$ L_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{5}	3

C_2

itemset
{1 2}
{1 3}
{1 5}
{2 3}
{2 5}
{3 5}

Scan D $\longleftarrow$ C_2

itemset	sup
{1 2}	1
{1 3}	2
{1 5}	1
{2 3}	2
{2 5}	3
{3 5}	2

$\longleftarrow$ L_2

itemset	sup
{1 3}	2
{2 3}	2
{2 5}	3
{3 5}	2

C_3

itemset
{2 3 5}

Scan D $\longrightarrow$ L_3

itemset	sup
{2 3 5}	2

Bayesian Classification: Why?

- A statistical classifier: performs *probabilistic prediction*, *i.e.*, predicts class membership probabilities
- Foundation: Based on Bayes' Theorem.
- Performance: A simple Bayesian classifier, *naïve Bayesian classifier*, has comparable performance with decision tree and selected neural network classifiers
- Incremental: Each training example can incrementally increase/decrease the probability that a hypothesis is correct — prior knowledge can be combined with observed data
- Standard: Even when Bayesian methods are computationally intractable, they can provide a standard of optimal decision making against which other methods can be measured

Bayesian Theorem: Basics

- Let X be a data sample ("evidence"): class label is unknown
- Let H be a hypothesis that X belongs to class C
- Classification is to determine P(H|X), the probability that the hypothesis holds given the observed data sample X
- P(H) (prior probability), the initial probability
  - E.g., X will buy computer, regardless of age, income, …
- P(X): probability that sample data is observed
- P(X|H) (posteriori probability), the probability of observing the sample X, given that the hypothesis holds
  - E.g., Given that X will buy computer, the prob. that X is 31..40, medium income

Bayesian Theorem

- Given training data X, posteriori probability of a hypothesis H, P(H|X), follows the Bayes theorem
- $$P(H \mid \mathbf{X}) = \frac{P(\mathbf{X} \mid H)P(H)}{P(\mathbf{X})}$$
- Informally, this can be written as
  - posteriori = likelihood x prior/evidence
- Predicts X belongs to C2 iff the probability P(Ci|X) is the highest among all the P(Ck|X) for all the k classes
- Practical difficulty: require initial knowledge of many probabilities, significant computational cost

Bayesian Classifier: Training Dataset

Class:
C1:buys_computer = ‘yes’
C2:buys_computer = ‘no’

Data sample
X = (age <=30,
Income = medium,
Student = yes
Credit_rating = Fair)

age	income	studentc r	edit_ratin g	_cor
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Bayesian Classifier: An Example

- P(Ci): P(buys_computer = "yes") = 9/14 = 0.643
  P(buys_computer = "no") = 5/14= 0.357
- Compute P(X|Ci) for each class

  P(age = "<=30" | buys_computer = "yes") = 2/9 = 0.222
  P(age = "<= 30" | buys_computer = "no") = 3/5 = 0.6
  P(income = "medium" | buys_computer = "yes") = 4/9 = 0.444
  P(income = "medium" | buys_computer = "no") = 2/5 = 0.4
  P(student = "yes" | buys_computer = "yes) = 6/9 = 0.667
  P(student = "yes" | buys_computer = "no") = 1/5 = 0.2
  P(credit_rating = "fair" | buys_computer = "yes") = 6/9 = 0.667
  P(credit_rating = "fair" | buys_computer = "no") = 2/5 = 0.4
- X = (age <= 30 , income = medium, student = yes, credit_rating = fair)

P(X|Ci) : P(X|buys_computer = "yes") = 0.222 x 0.444 x 0.667 x 0.667 = 0.044
P(X|buys_computer = "no") = 0.6 x 0.4 x 0.2 x 0.4 = 0.019

P(X|Ci)*P(Ci) : P(X|buys_computer = "yes") * P(buys_computer = "yes") =0.044*0.643
= 0.028
P(X|buys_computer = "no") * P(buys_computer = "no") =0.019 *0.357
= 0.007

Therefore, X belongs to class ("buys_computer = yes")

Introduction

- Terminology
- Apriori-like Algorithms
  - Generate-and-Test
  - Cost Bottleneck
- FP-Tree and FP-Growth Algorithm
  - FP-Tree: Frequent Pattern Tree
  - FP-Growth: Mining frequent patterns with FP-Tree

Terminology

- Item set
  - A set of items: $I = \{a_1, a_2, \ldots\ldots, a_m\}$
- Transaction database
  - $DB = <T_1, T_2, \ldots\ldots, T_n>$
- Pattern
  - A set of items: A
- Support
  - The number of transactions containing A in DB
- Frequent pattern
  - A's support $\geq$ minimum support threshold ξ
- Frequent Pattern Mining Problem
  - The problem of finding the complete set of frequent patterns

FP-Tree and FP-Growth Algorithm

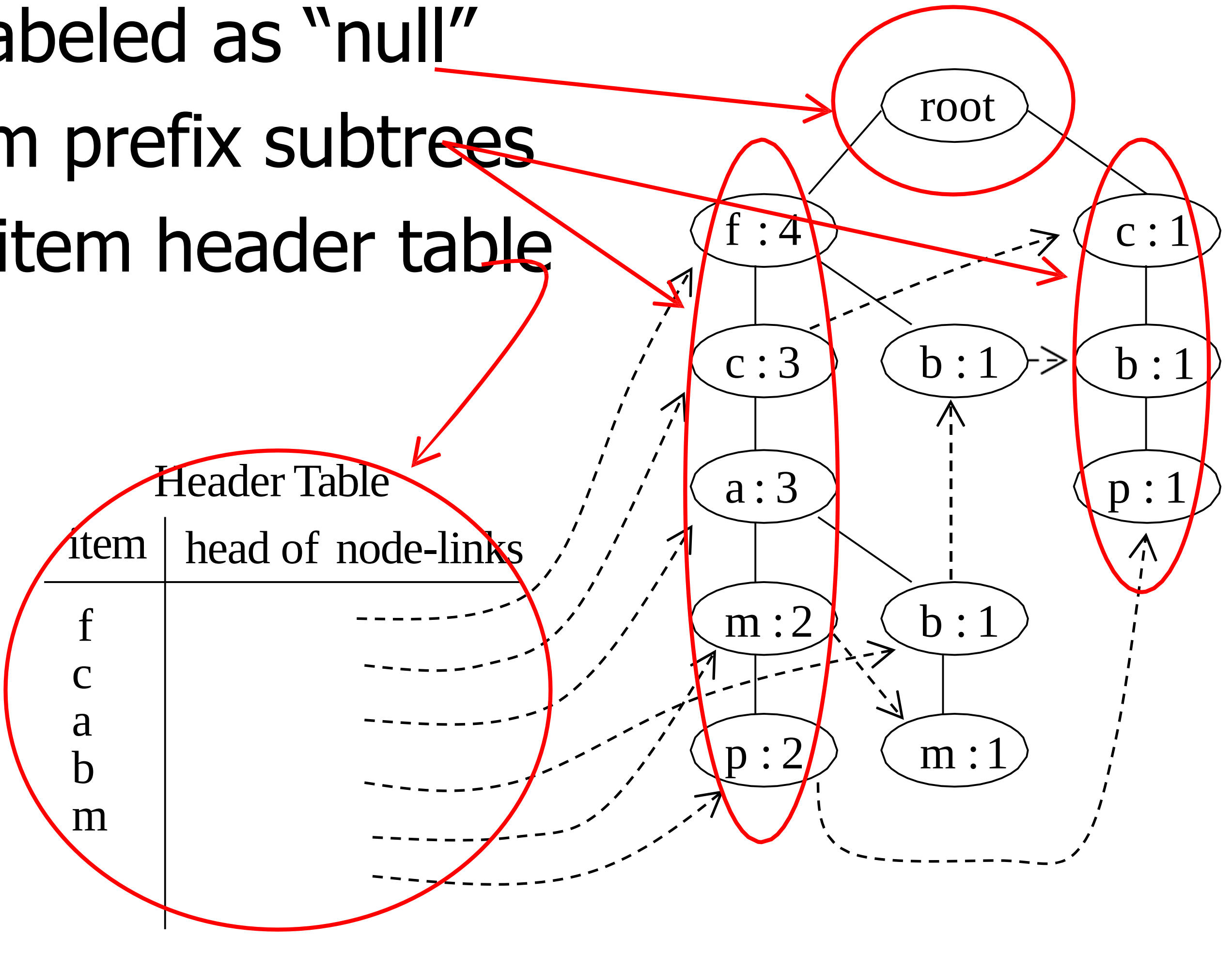
- FP-Tree: Frequent Pattern Tree
  - Compact presentation of the DB without information loss.
  - Easy to traverse, can quickly find out patterns associated with a certain item.
  - Well-ordered by item frequency.
- FP-Growth Algorithm
  - Start mining from length-1 patterns
  - Recursively do the following
    - Constructs its conditional FP-tree
    - Concatenate patterns from conditional FP-tree with suffix
  - Divide-and-Conquer mining technique

Outline

- Introduction
- Constructing FP-Tree
  - Example 1
- Mining Frequent Patterns using FP-Tree
  - Example 2
- Performance Evaluation
- Discussions

FP-Tree Definition

- Three components:
  - One root: labeled as "null"
  - A set of item prefix subtrees
  - A frequent-item header table



FP-Tree Definition (cont.)

- Each node in the item prefix subtree consists of three fields:
  - item-name
  - node-link
  - count
- Each entry in the frequent-item header table consists of two fields:
  - item-name
  - head of node-link

Example 1: FP-Tree Construction

- The transaction database used (fist two column only):

TID	Items Bought	(Ordered) Frequent Items
100	*f,a,c,d,g,i,m,p*	*f,c,a,m,p*
200	*a,b,c,f,l,m,o*	*f,c,a,m,b*
300	*b,f,h,j,o*	*f,b*
400	*b,c,k,s,p*	*c,b,p*
500	*a,f,c,e,l,p,m,n*	*f,c,a,m,p*

f=4,a=3,c=4,d=1,g= 1,i=1,m=3,p=2,b=3,l= 2,o=2,h=1,
J=1,k=1,s=1,p=3

minimum support threshold ξ= 3

f=4,a=3,c=4, m=3, b=3,p=3

443333=> f,c,a,m,b,p

Example 1 (cont.)

- First Scan: //count and sort
  - count the frequencies of each item
  - collect length-1 frequent items, then sort them in support descending order into L, frequent item list.

    L = {(f:4), (c:4), (a:3), (b:3), (m:3), (p:3)}

Example 1 (cont.)

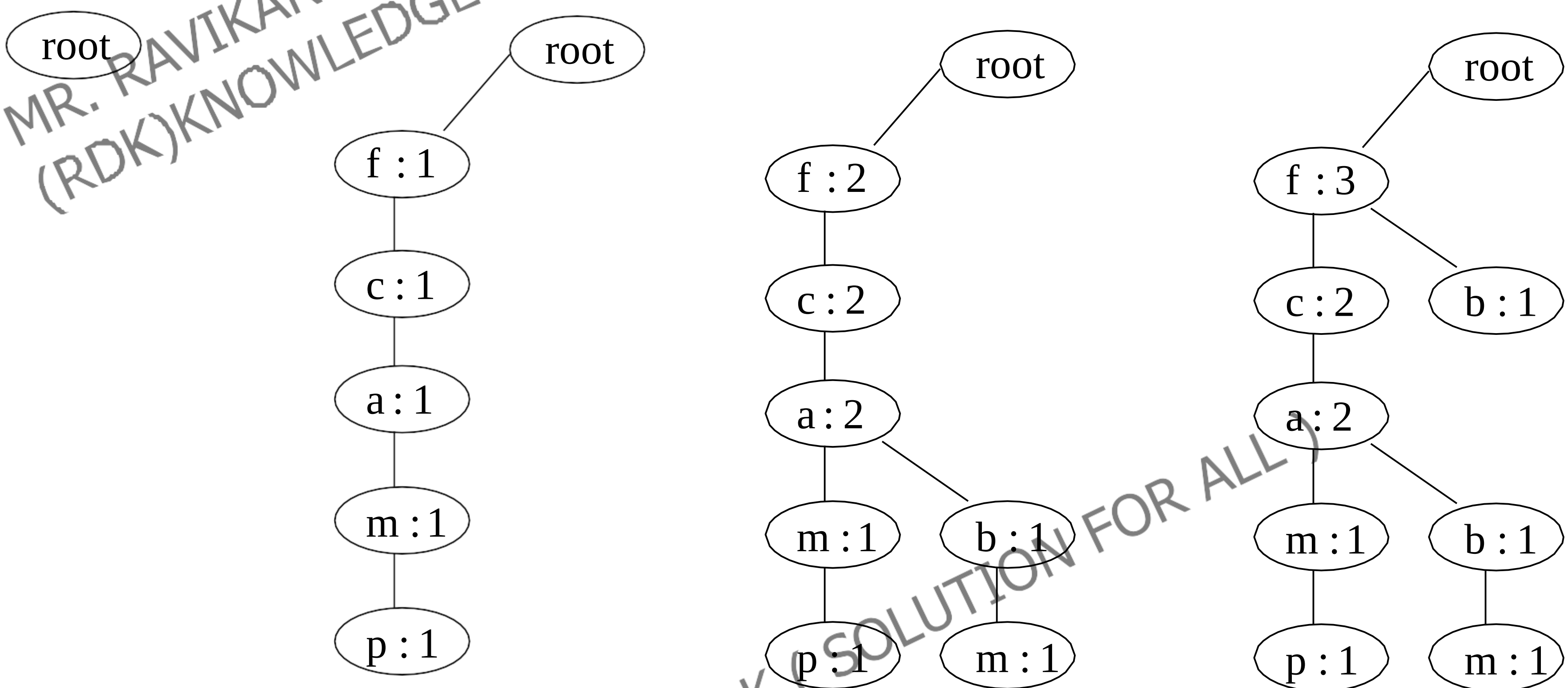
- Second Scan://create the tree and header table
  - create the root, label it as “null”
  - for each transaction Trans, do
    - select and sort the frequent items in Trans
    - increase nodes count or create new nodes

      If prefix nodes already exist, increase their counts by 1;

      If no prefix nodes, create it and set count to 1.
  - build the item header table
    - nodes with the same item-name are linked in sequence via node-links

Example 1 (cont.)

The building process of the tree



Create root

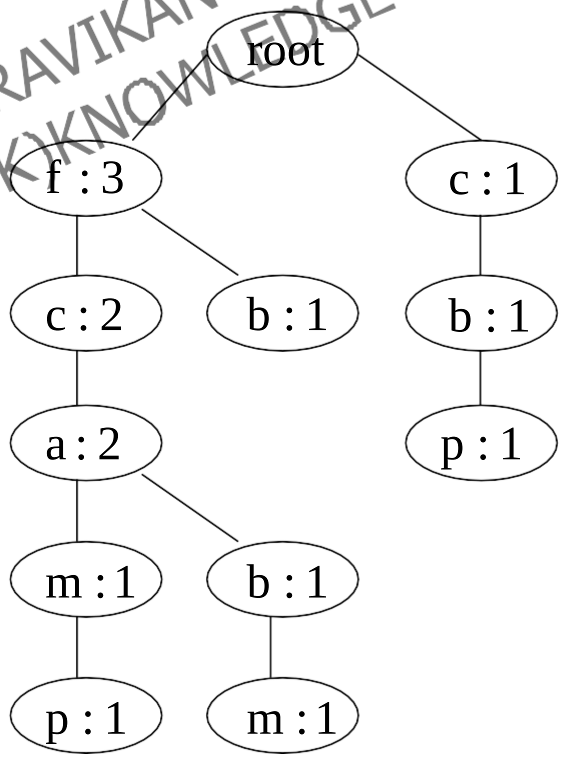
After trans 1 (f,c,a,m,p)

After trans 2 (f,c,a,b,m)

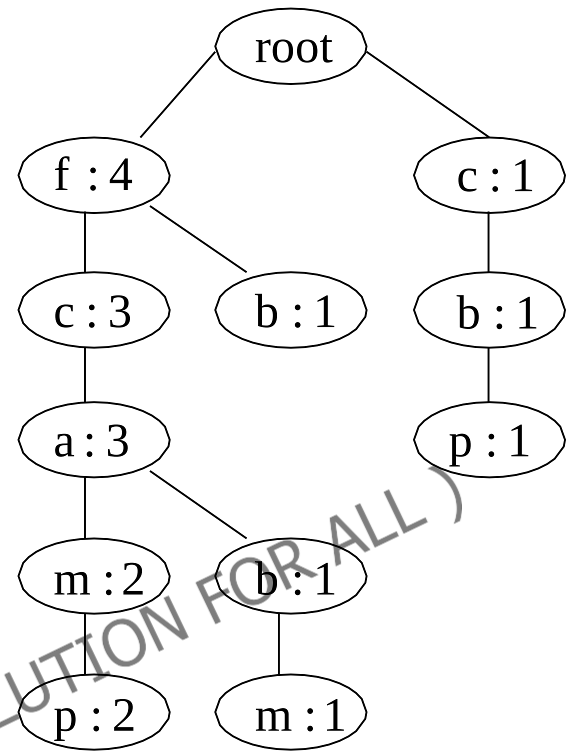
After trans 3 (f,b)

Example 1 (cont.)

The building process of the tree (cont.)



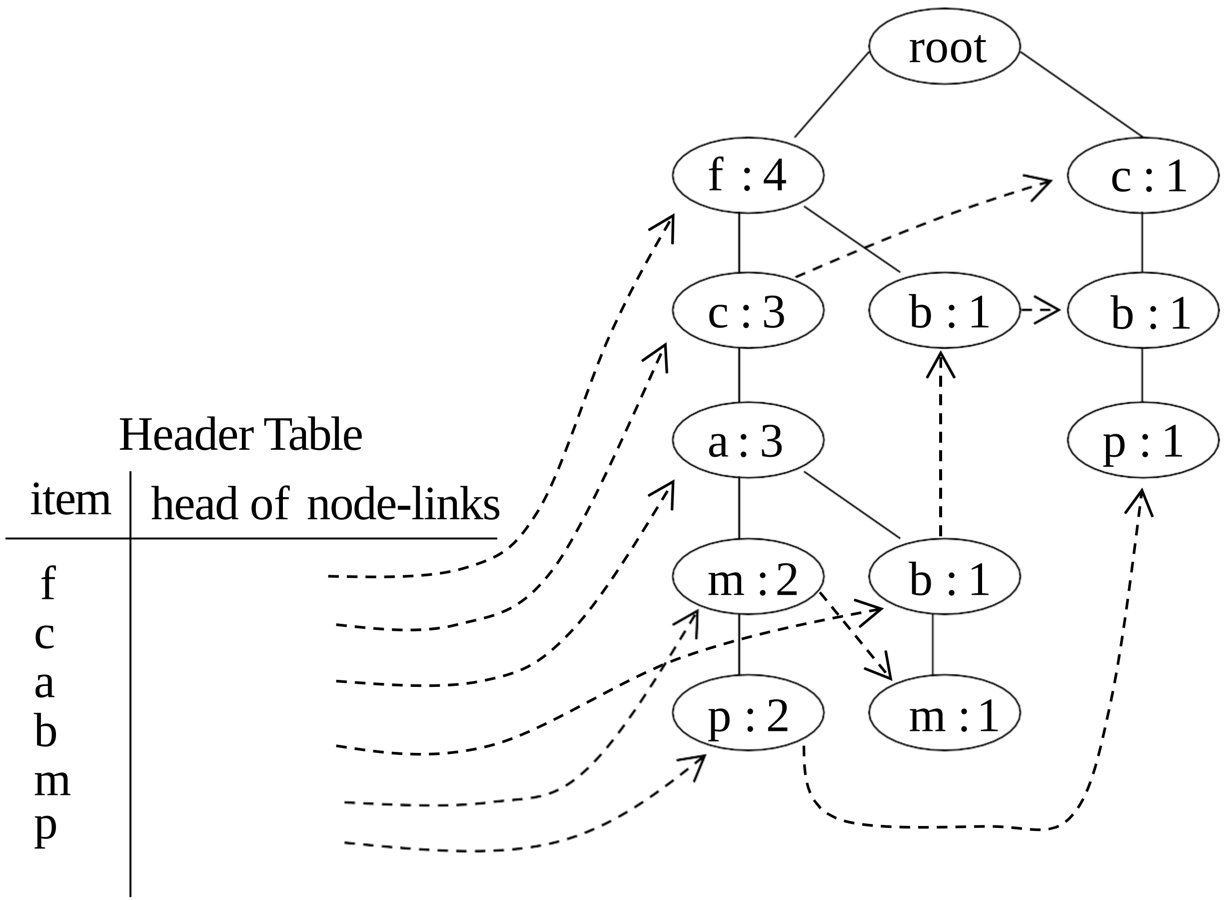
After trans
4 (c,b,p)



After trans
5 (f,c,a,m,p)

Example 1 (cont.)

Build the item header table



FP-Tree Properties

- Completeness
  - Each transaction that contains frequent pattern is mapped to a path.
  - Prefix sharing does not cause path ambiguity, as only path starts from root represents a transaction.
- Compactness
  - Number of nodes bounded by overall occurrence of frequent items.
  - Height of tree bounded by maximal number of frequent items in any transaction.

- Clustering is grouping thing with similar attribute values into the same group. Given a database

Partitioning Algorithms: Basic Concept

- Partitioning method: Construct a partition of a database **D** of **n** objects into a set of **k** clusters, s.t., min sum of squared distance

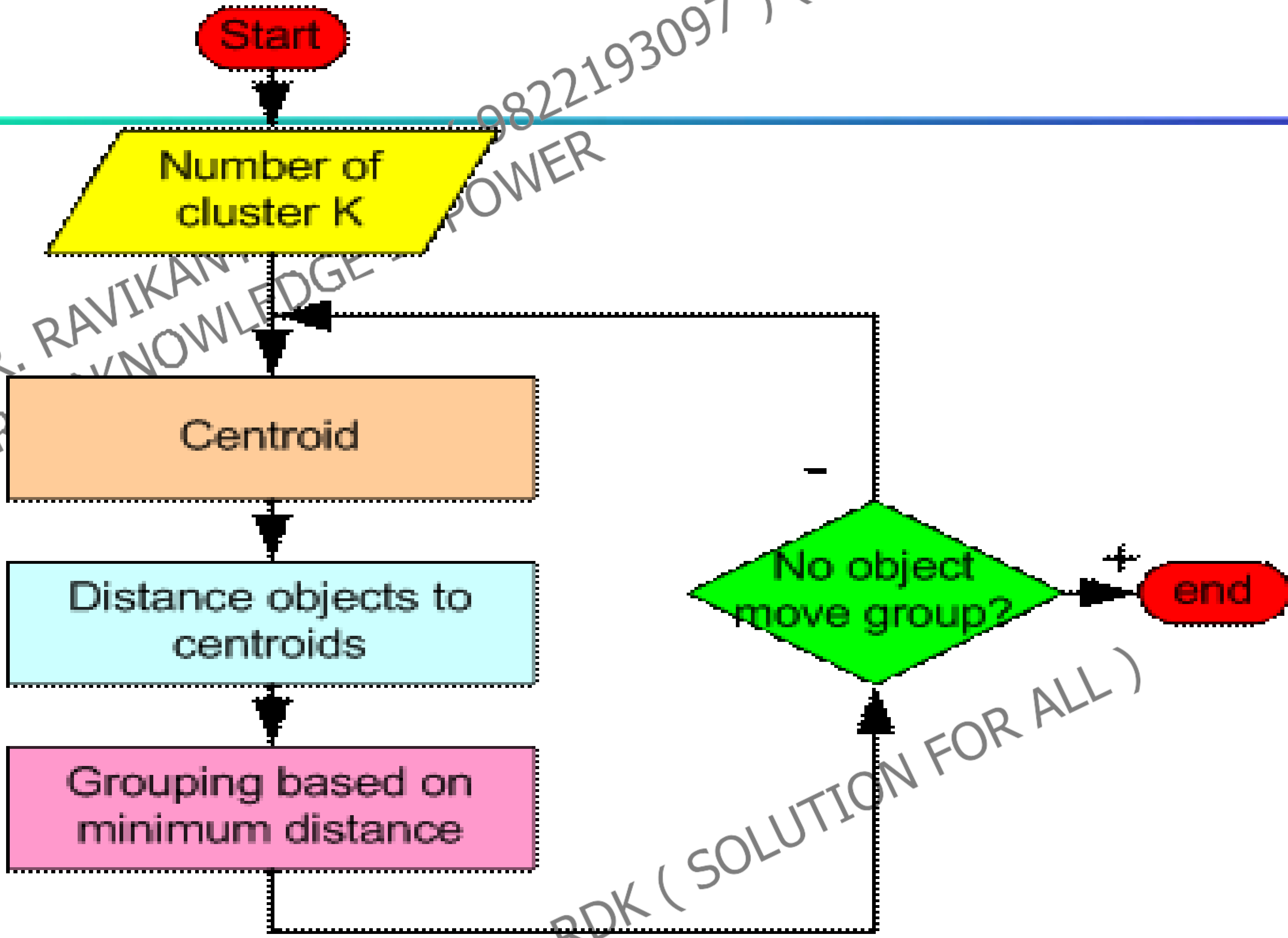
$$\Sigma_{m=1}^{k}\Sigma_{t_{mi}\in Km}(C_m - t_{mi})^2$$

Given a k, find a partition of k clusters that optimizes the chosen partitioning criterion

- k-means: Each cluster is represented by the center of the cluster

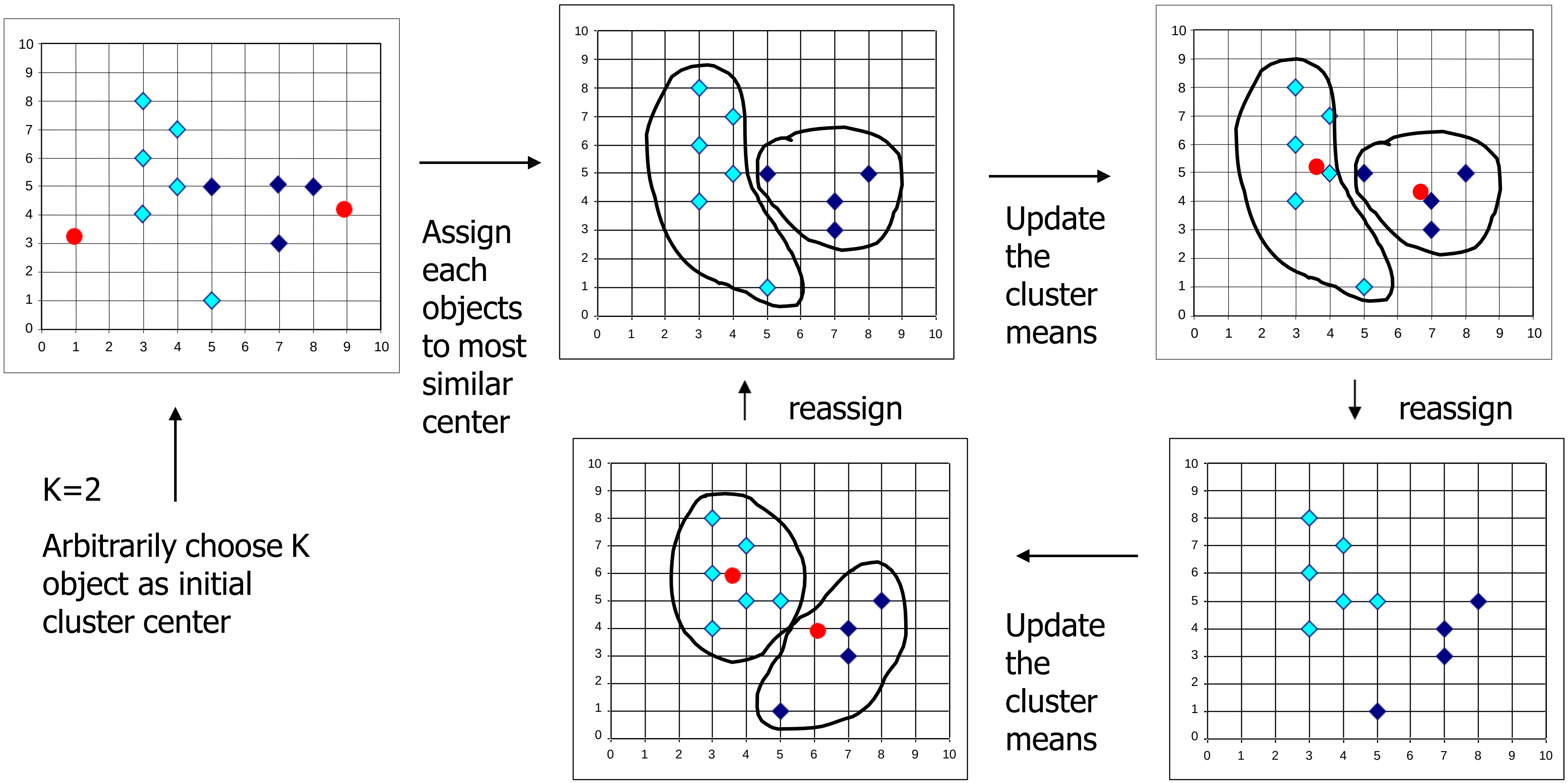
The K-Means Clustering Method

- Given k, the k-means algorithm is implemented in 3 steps:
- **Steps are as follows:**
- Determine the centroid coordinate.
- Determine the distance of each object to the centroids.
- Group the object based on minimum distance.


Start
Number of cluster K
Centroid
Distance objects to centroids
Grouping based on minimum distance
No object move group?
-
+
end

The K-Means Clustering Method

- Example



K-Means clustering

In the K-Means clustering, initially a set of clusters is randomly chosen. Then iteratively, items are moved among sets of clusters until the desired set is reached.

A high degree of similarity among elements in a cluster is obtained by using this algorithm.

For this algorithm a set of clusters $K_i=\{t_{i1},t_{i2},\ldots,t_{im}\}$ is given , the cluster mean is:

$m_i = (1/m)(t_{i1} + \ldots + t_{im}) \ldots$

Where ti represents the tuples and m represents the mean

The K-Means algorithm is as follows:

Input :

D= { t1,t2,…tn} //Set of elements

A //Adjacency matrix showing distance between elements.

k //Number of desired clusters.

Output :

K //Set of Clusters

K-Means Algorithm

Assign initial values for means m1,m2..mk;
Repeat
    Assign each item ti to the cluster which has the closest mean;
    Calculate new mean for each cluster;
Until convergence criteria is met.