Association rules

Lecturer: JERZY STEFANOWSKI

Institute of Computing Sciences

Poznan University of Technology

Poznan, Poland

Lecture 10

SE Master Course

2008/2009

This lecture is based on the following resources - slides:
G.Piatetsky-Shapiro: Association Rules and Frequent Item Analysis.
and partly on
J.Han: Mining Association Rules in Large Databases
and my other notes.

Outline

- Transactions
- Frequent itemsets
- Subset Property
- Association rules
- Applications

Association rules

- Transaction data
- Market basket analysis

TID	Produce
1	MILK, BREAD, EGGS
2	BREAD, SUGAR
3	BREAD, CEREAL
4	MILK, BREAD, SUGAR
5	MILK, CEREAL
6	BREAD, CEREAL
7	MILK, CEREAL
8	MILK, BREAD, CEREAL, EGGS
9	MILK, BREAD, CEREAL

- {Cheese, Milk} → Bread [sup=5%, conf=80%]
- Association rule:
  „80% of customers who buy *cheese* and *milk* also buy *bread* and 5% of customers buy all these products together”

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?
  - Can we automatically classify web documents?
- Applications
  - Basket data analysis, cross-marketing, catalog design, sale campaign analysis, Web log (click stream) analysis, and DNA sequence analysis.

Why is Frequent Pattern or Association Mining an Essential Task in Data Mining?

- Foundation for many essential data mining tasks
  - Association, correlation, causality
  - Sequential patterns, temporal or cyclic association, partial periodicity, spatial and multimedia association
  - Associative classification, cluster analysis, fascicles (semantic data compression)
- DB approach to efficient mining massive data
- Broad applications
  - Basket data analysis, cross-marketing, catalog design, sale campaign analysis
  - Web log (click stream) analysis, DNA sequence analysis, etc

Transactions Example

TID	Produce
1	MILK, BREAD, EGGS
2	BREAD, SUGAR
3	BREAD, CEREAL
4	MILK, BREAD, SUGAR
5	MILK, CEREAL
6	BREAD, CEREAL
7	MILK, CEREAL
8	MILK, BREAD, CEREAL, EGGS
9	MILK, BREAD, CEREAL

Transaction database: Example, 1

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

Instances = Transactions

ITEMS:

A = milk
B= bread
C= cereal
D= sugar
E= eggs

Transaction database: Example, 2

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

Attributes converted to binary flags

TID	A	B	C	D	E
1	1	1	0	0	1
2	0	1	0	1	0
3	0	1	1	0	0
4	1	1	0	1	0
5	1	0	1	0	0
6	0	1	1	0	0
7	1	0	1	0	0
8	1	1	1	0	1
9	1	1	1	0	0

Definitions

- Item: *attribute*=*value* pair or simply *value*
  - usually attributes are converted to binary *flags* for each value, e.g. **product="A"** is written as **"A"**
- Itemset *I* : a subset of possible items
  - Example: *I* = {A,B,E} (order unimportant)
- Transaction: (TID, itemset)
  - TID is transaction ID

Support and Frequent Itemsets

- Support of an itemset
  - sup(I) = no. of transactions t that support (i.e. contain) I
- In example database:
  - sup ({A,B,E}) = 2, sup ({B,C}) = 4
- Frequent itemset I is one with at least the minimum support count
  - sup(I) >= *minsup*

SUBSET PROPERTY (Agrawal et al..)

- **Every subset of a frequent set is frequent!**
- Q: Why is it so?
- A: Example: Suppose {A,B} is frequent. Since each occurrence of A,B includes both A and B, then both A and B must also be frequent
- Similar argument for larger itemsets
- Almost all association rule algorithms are based on this subset property

Association Rules

- Association rule *R* : *Itemset1* => *Itemset2*
  - *Itemset1, 2* are disjoint and *Itemset2* is non-empty
  - meaning: if transaction includes *Itemset1* then it also has *Itemset2*
- Examples
  - A,B => E,C
  - A => B,C

From Frequent Itemsets to Association Rules

- *Q: Given frequent set {A,B,E}, what are possible association rules?*
  - A => B, E
  - A, B => E
  - A, E => B
  - B => A, E
  - B, E => A
  - E => A, B
  - __ => A,B,E (empty rule), or true => A,B,E

Rule Support and Confidence

- Suppose $R : I => J$ is an association rule
  - sup (R) = sup (I ∪ J) is the *support count*
    - support of itemset I ∪ J (I or J)
  - conf (R) = sup(J) / sup(R) is the *confidence* of R
    - fraction of transactions with I ∪ J that have J
- Association rules with minimum support and count are sometimes called "***strong***" rules

Classification vs Association Rules

Classification Rules

- Focus on one target field
- Specify class in all cases
- Measures: Accuracy

Association Rules

- Many target fields
- Applicable in some cases
- Measures: Support, Confidence, Lift

Association Rules Example, 1

- ***Q: Given frequent set {A,B,E}, what association rules have minsup = 2 and minconf= 50% ?***

A, B => E : conf=2/4 = 50%

TID	List of items
1	**A, B, E**
2	B, D
3	B, C
4	**A, B,** D
5	A, C
6	B, C
7	A, C
8	**A, B, C, E**
9	**A, B**, C

Association Rules Example, 2

- ***Q: Given frequent set {A,B,E}, what association rules have minsup = 2 and minconf= 50% ?***

A, B => E : conf=2/4 = 50%

A, E => B : conf=2/2 = 100%

B, E => A : conf=2/2 = 100%

E => A, B : conf=2/2 = 100%

Don't qualify

A =>B, E : conf=2/6 =33%< 50%

B => A, E : conf=2/7 = 28% < 50%

__ => A,B,E : conf: 2/9 = 22% < 50%

TID	List of items
1	**A, B, E**
2	B, D
3	B, C
4	A, B, D
5	A, C
6	B, C
7	A, C
8	**A, B, C, E**
9	A, B, C

Find Strong Association Rules

- A rule has the parameters *minsup* and *minconf*:
  - sup(R) >= *minsup* and conf (R) >= *minconf*
- **Problem**:
  - Find all association rules with given *minsup* and *minconf*
- First, find all frequent itemsets

Finding Frequent Itemsets

- Start by finding one-item sets (easy)
- *Q: How?*
- A: Simply count the frequencies of all items

Finding itemsets: next level

- Apriori algorithm (Agrawal & Srikant 94)
- Idea: use one-item sets to generate two-item sets, two-item sets to generate three-item sets, …
    - If (A B) is a frequent item set, then (A) and (B) have to be frequent item sets as well!
    - In general: if X is frequent *k*-item set, then all (*k*-1)-item subsets of X are also frequent

  ⇒ Compute *k*-item set by merging (*k*-1)-item sets

Another example

- Given: five three-item sets

  (A B C), (A B D), (A C D), (A C E), (B C D)

- Lexicographic order improves efficiency!
- Candidate four-item sets:

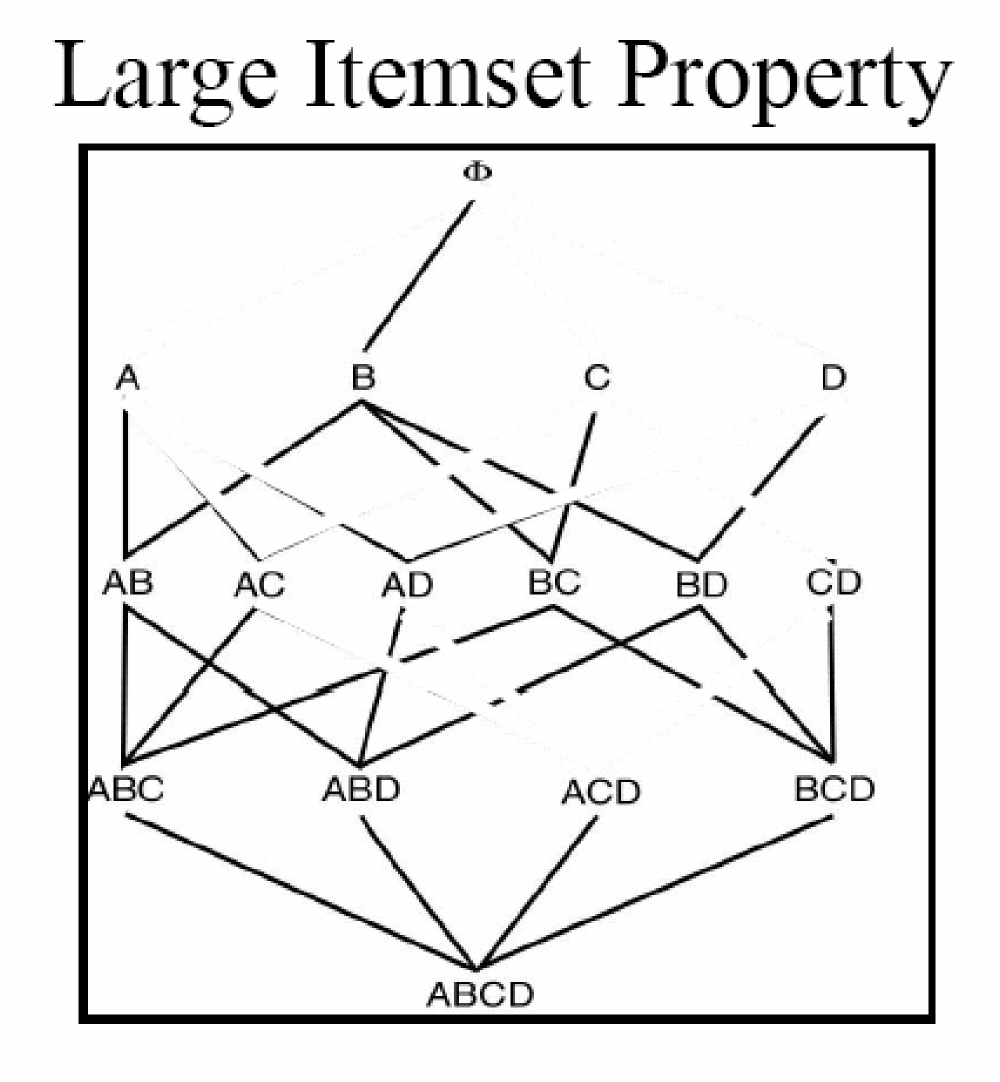
  (A B C D) Q: OK?

A: yes, because all 3-item subsets are frequent

(A C D E) Q: OK?

A: No, because (C D E) is not frequent

Pruning search space



Large Itemset Property

Apriori: A Candidate Generation-and-test Approach - Summary

- Any subset of a frequent itemset must be frequent
  - if **{beer, diaper, nuts}** is frequent, so is **{beer, diaper}**
  - Every transaction having {beer, diaper, nuts} also contains {beer, diaper}
- Apriori pruning principle: If there is any itemset which is infrequent, its superset should not be generated/tested!
- Method:
  - generate length (k+1) candidate itemsets from length k frequent itemsets, and
  - test the candidates against DB
- The performance studies show its efficiency and scalability
- Agrawal & Srikant 1994, Mannila, et al. 1994

Mining Association Rules—an Example

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

Min. support 50%
Min. confidence 50%

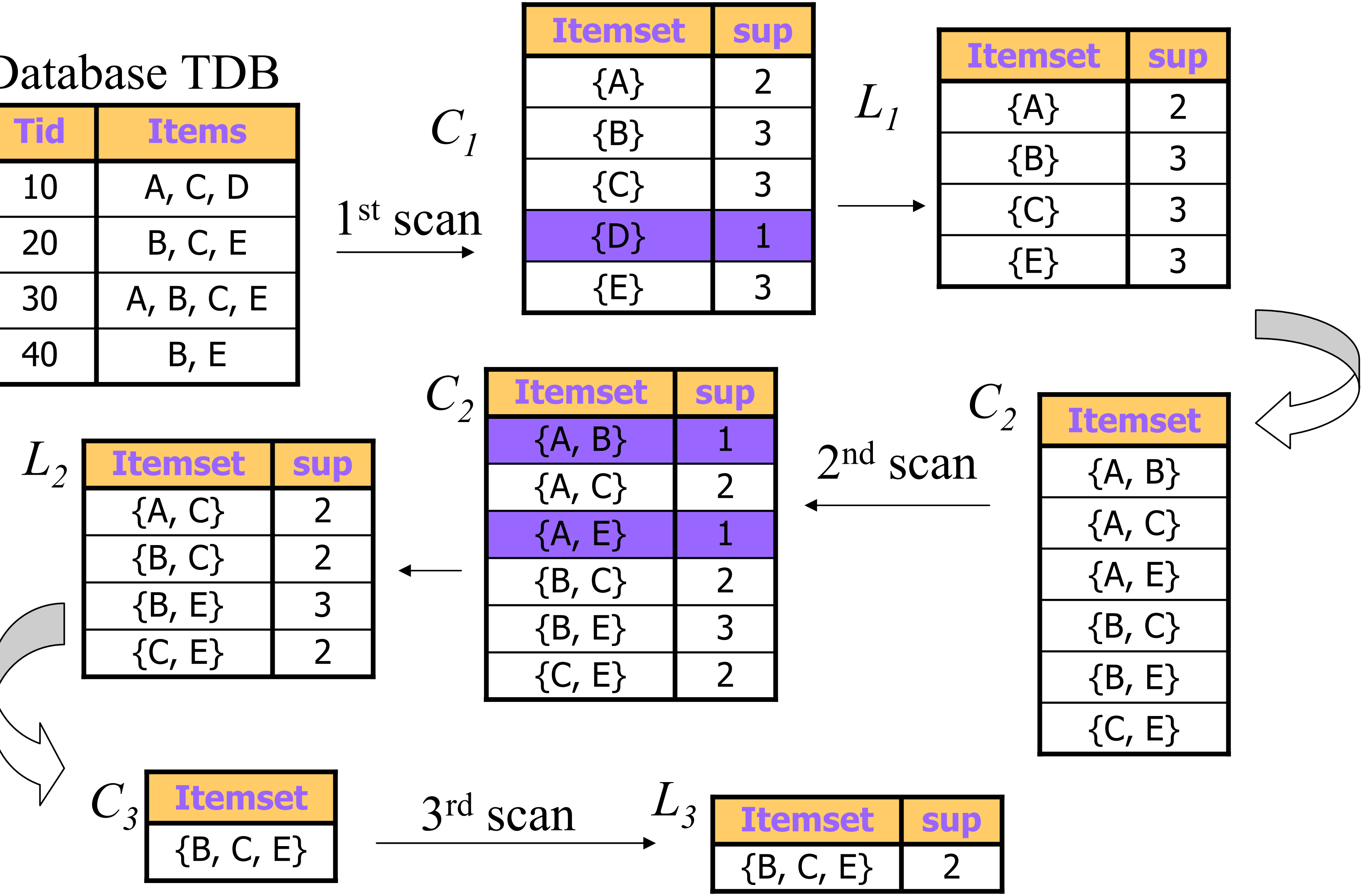
Frequent pattern	Support
{A}	75%
{B}	50%
{C}	50%
{A, C}	50%

For rule $A \Rightarrow C$:

support = support($\{A\}\cup\{C\}$) = 50%

confidence = support($\{A\}\cup\{C\}$)/support($\{A\}$) = 66.6%

The Apriori Algorithm—An Example



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; $L_k$ !=$\emptyset$; $k$++) 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 $\cup_k L_k$;
```

How to Generate Candidates?

- Suppose the items in L_{k-1} are listed in an order
- Step 1: self-joining L_{k-1}

```
insert into $C_k$
select $p.item_1, p.item_2, \ldots, p.item_{k-1}, q.item_{k-1}$
from $L_{k-1}\ p, L_{k-1}\ q$
where $p.item_1=q.item_1, \ldots, p.item_{k-2}=q.item_{k-2}, p.item_{k-1} < q.item_{k-1}$
```

- Step 2: pruning

```
forall itemsets $c$ in $C_k$ do
    forall $(k-1)$-subsets $s$ of $c$ do
        if ($s$ is not in $L_{k-1}$) then delete $c$ from $C_k$
```

How to Count Supports of Candidates?

- Why counting supports of candidates a problem?
  - The total number of candidates can be very huge
  - One transaction may contain many candidates
- Method:
  - Candidate itemsets are stored in a *hash-tree*
  - *Leaf* node of hash-tree contains a list of itemsets and counts
  - *Interior* node contains a hash table
  - *Subset function*: finds all the candidates contained in a transaction

Generating Association Rules

- Two stage process:
    - Determine frequent itemsets e.g. with the Apriori algorithm.
    - For each frequent item set I
        - for each subset J of I
            - determine all association rules of the form: I-J => J
- Main idea used in both stages : subset property

Example: Generating Rules from an Itemset

- Frequent itemset from golf data:

Humidity = Normal, Windy = False, Play = Yes (4)

- Seven potential rules:

If Humidity = Normal and Windy = False then Play = Yes	4/4
If Humidity = Normal and Play = Yes then Windy = False	4/6
If Windy = False and Play = Yes then Humidity = Normal	4/6
If Humidity = Normal then Windy = False and Play = Yes	4/7
If Windy = False then Humidity = Normal and Play = Yes	4/8
If Play = Yes then Humidity = Normal and Windy = False	4/9
If True then Humidity = Normal and Windy = False and Play = Yes	4/12

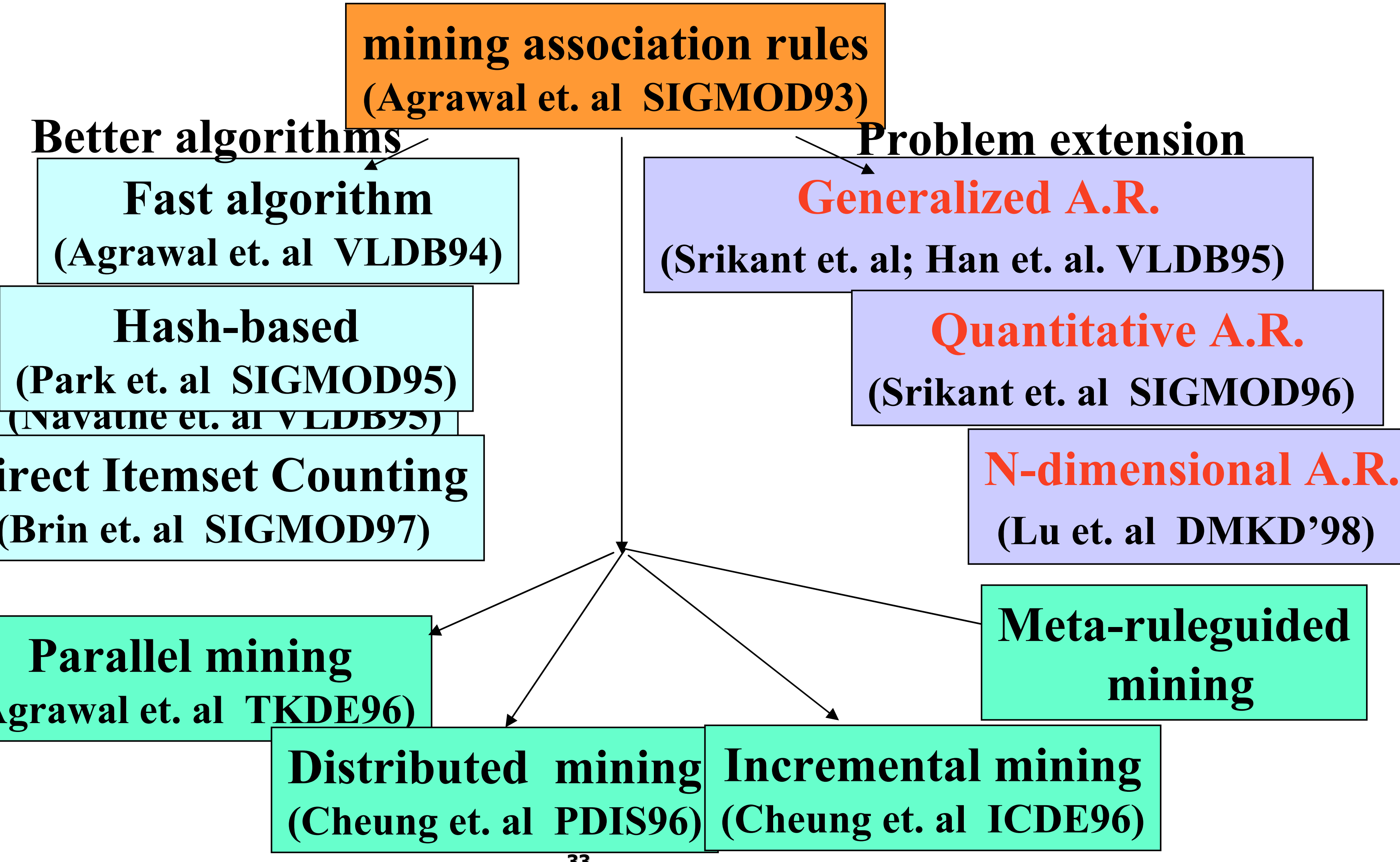
Rules for the weather data

- Rules with support > 1 and confidence = 100%:

	Association rule		Sup.	Conf.
1	Humidity=Normal Windy=False	⇒Play=Yes	4	100%
2	Temperature=Cool	⇒Humidity=Normal	4	100%
3	Outlook=Overcast	⇒Play=Yes	4	100%
4	Temperature=Cold Play=Yes	⇒Humidity=Normal	3	100%
...	...	...	...	...
58	Outlook=Sunny Temperature=Hot	⇒Humidity=High	2	100%

- In total: 3 rules with support four, 5 with support three, and 50 with support two

Association Rule Mining



mining association rules
(Agrawal et. al SIGMOD93)

Better algorithms

Fast algorithm
(Agrawal et. al VLDB94)

Hash-based
(Park et. al SIGMOD95)

(Navathe et. al VLDB95)

irect Itemset Counting
(Brin et. al SIGMOD97)

Problem extension

Generalized A.R.
(Srikant et. al; Han et. al. VLDB95)

Quantitative A.R.
(Srikant et. al SIGMOD96)

N-dimensional A.R.
(Lu et. al DMKD'98)

Meta-ruleguided mining

Parallel mining
grawal et. al TKDE96)

Distributed mining
(Cheung et. al PDIS96)

Incremental mining
(Cheung et. al ICDE96)

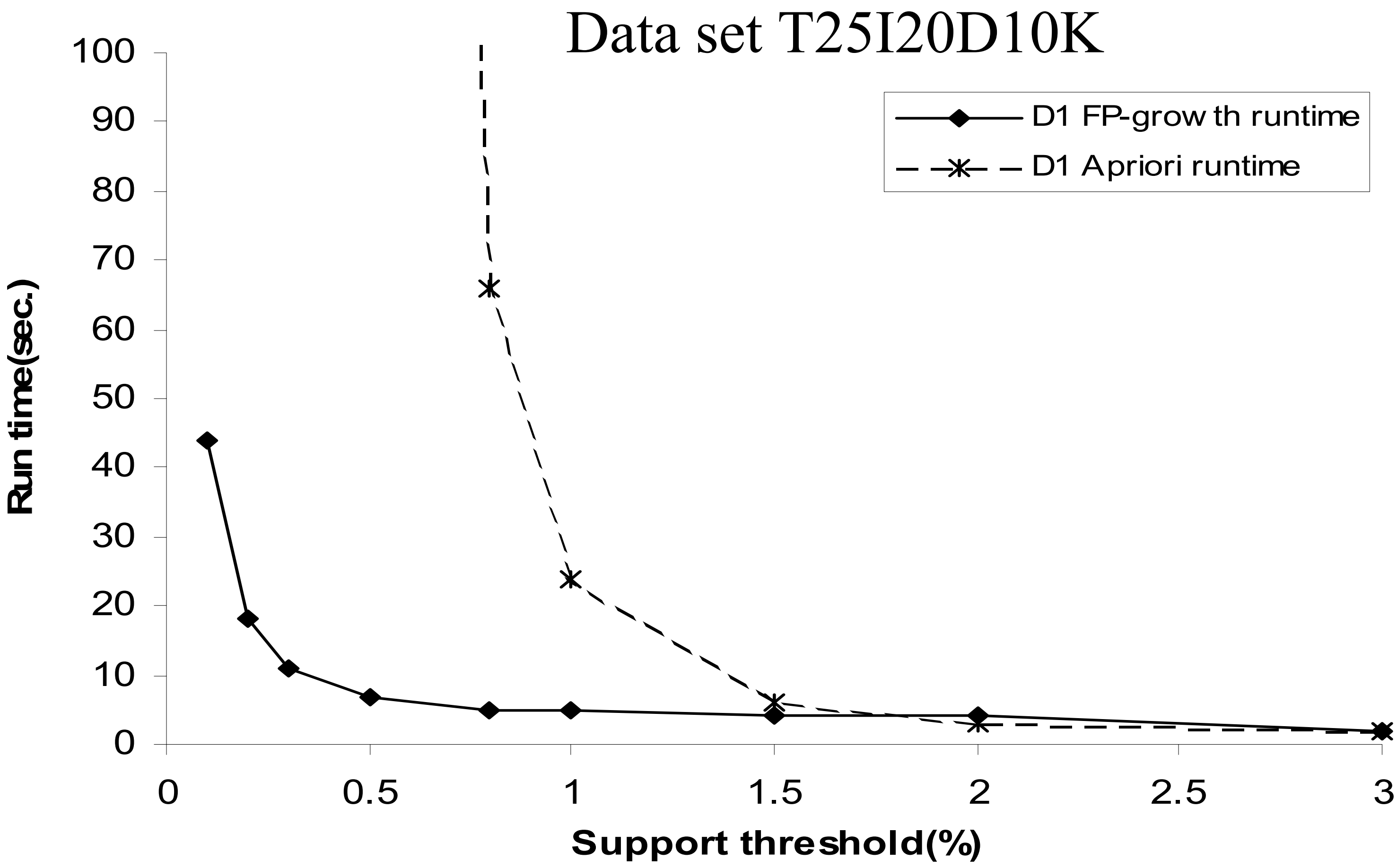
Bottleneck of Frequent-pattern Mining with Apriori

- Multiple database scans are costly
- Mining long patterns needs many passes of scanning and generates lots of candidates
  - To find frequent itemset $i_1 i_2 \ldots i_{100}$
    - # of scans: 100
    - # of Candidates: $({}_{100}{}^{1}) + ({}_{100}{}^{2}) + \ldots + ({}_{1}{}^{1}{}_{0}{}^{0}{}_{0}{}^{0}) = 2^{100}-1 = 1.27*10^{30}$!
- Bottleneck: candidate-generation-and-test
- Can we avoid candidate generation?
- Another algorithms → FP Tree

Mining Frequent Patterns Without Complete Candidate Generation

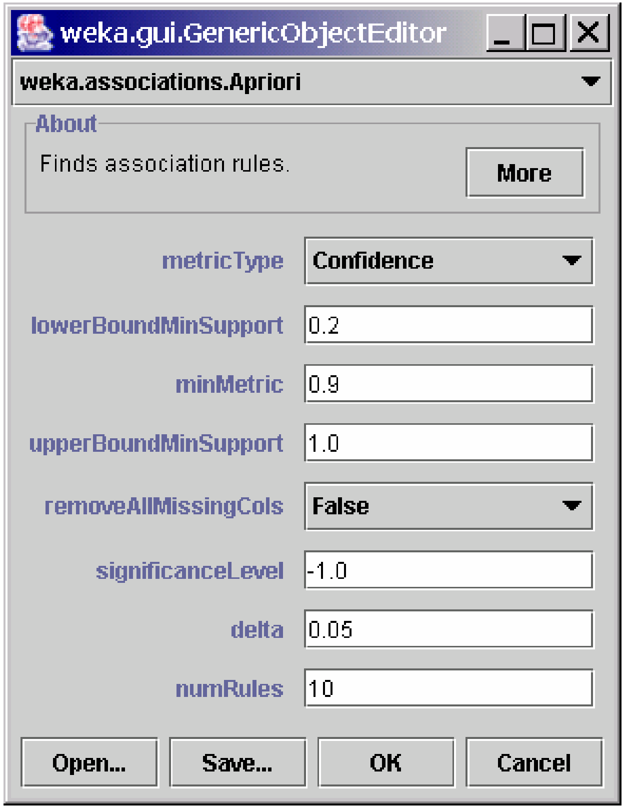
- Grow long patterns from short ones using local frequent items
  - "abc" is a frequent pattern
  - Get all transactions having "abc": DB|abc
  - "d" is a local frequent item in DB|abc → abcd is a frequent pattern

FP-Growth vs. Apriori: Scalability With the Support Threshold



Weka associations

File: weather.nominal.arff
MinSupport: 0.2



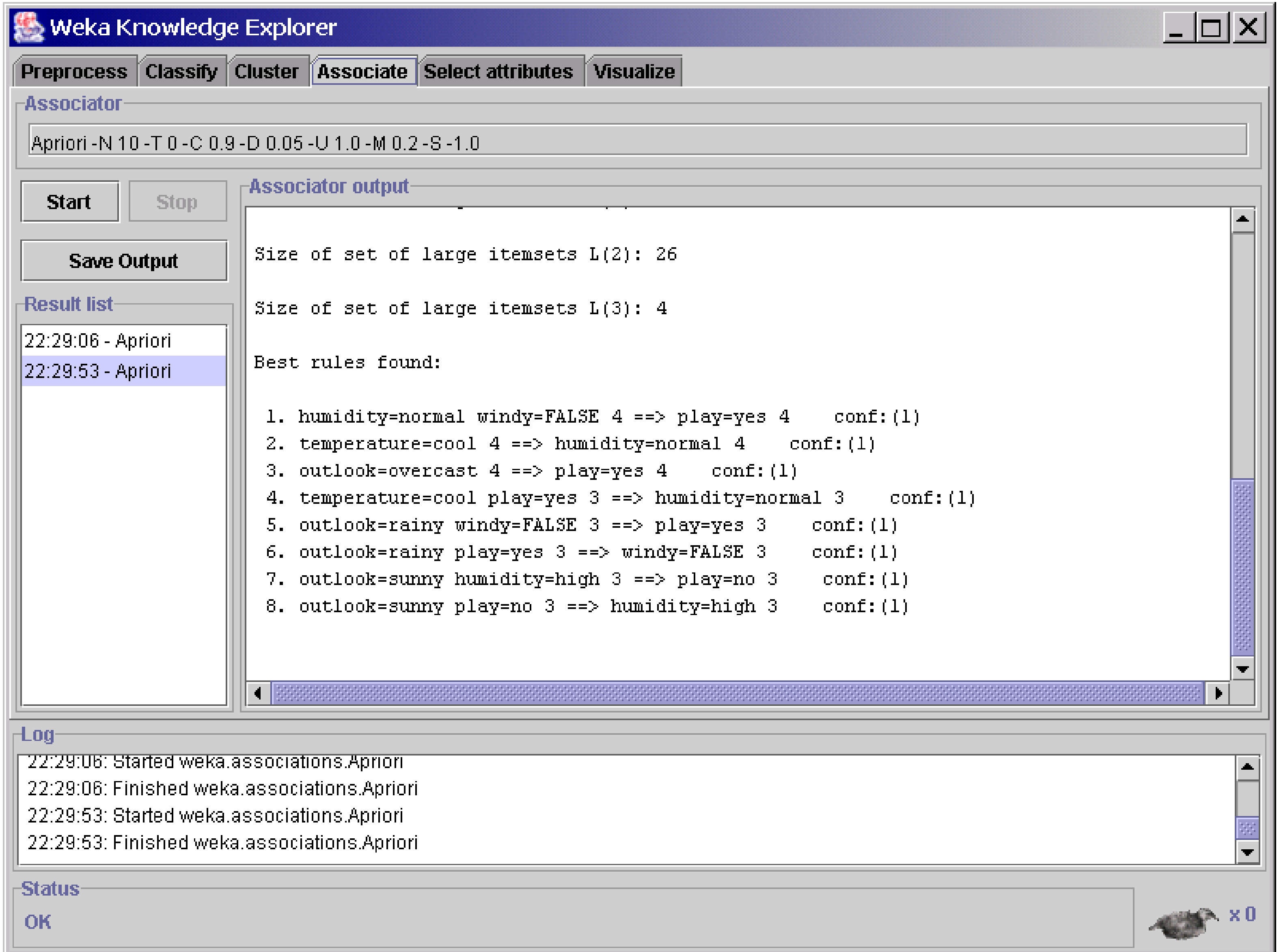
weka.associations.Apriori

About

Finds association rules.

metricType: Confidence
lowerBoundMinSupport: 0.2
minMetric: 0.9
upperBoundMinSupport: 1.0
removeAllMissingCols: False
significanceLevel: -1.0
delta: 0.05
numRules: 10

Weka associations: output



Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.2 -S -1.0

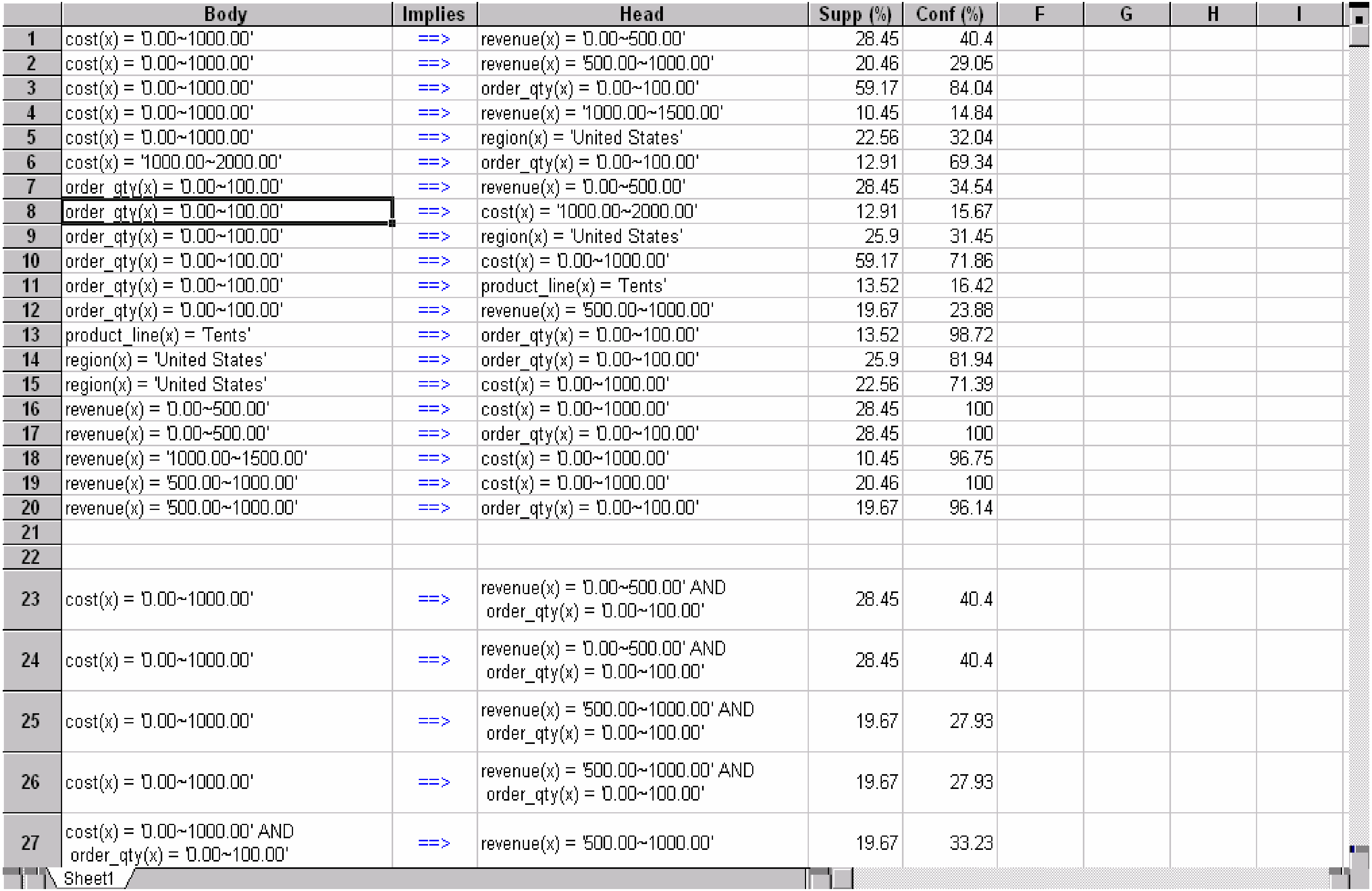
22:29:06 - Apriori
22:29:53 - Apriori

```
Size of set of large itemsets L(2): 26

Size of set of large itemsets L(3): 4

Best rules found:

 1. humidity=normal windy=FALSE 4 ==> play=yes 4    conf:(1)
 2. temperature=cool 4 ==> humidity=normal 4    conf:(1)
 3. outlook=overcast 4 ==> play=yes 4    conf:(1)
 4. temperature=cool play=yes 3 ==> humidity=normal 3    conf:(1)
 5. outlook=rainy windy=FALSE 3 ==> play=yes 3    conf:(1)
 6. outlook=rainy play=yes 3 ==> windy=FALSE 3    conf:(1)
 7. outlook=sunny humidity=high 3 ==> play=no 3    conf:(1)
 8. outlook=sunny play=no 3 ==> humidity=high 3    conf:(1)
```

```
22:29:06: Started weka.associations.Apriori
22:29:06: Finished weka.associations.Apriori
22:29:53: Started weka.associations.Apriori
22:29:53: Finished weka.associations.Apriori
```

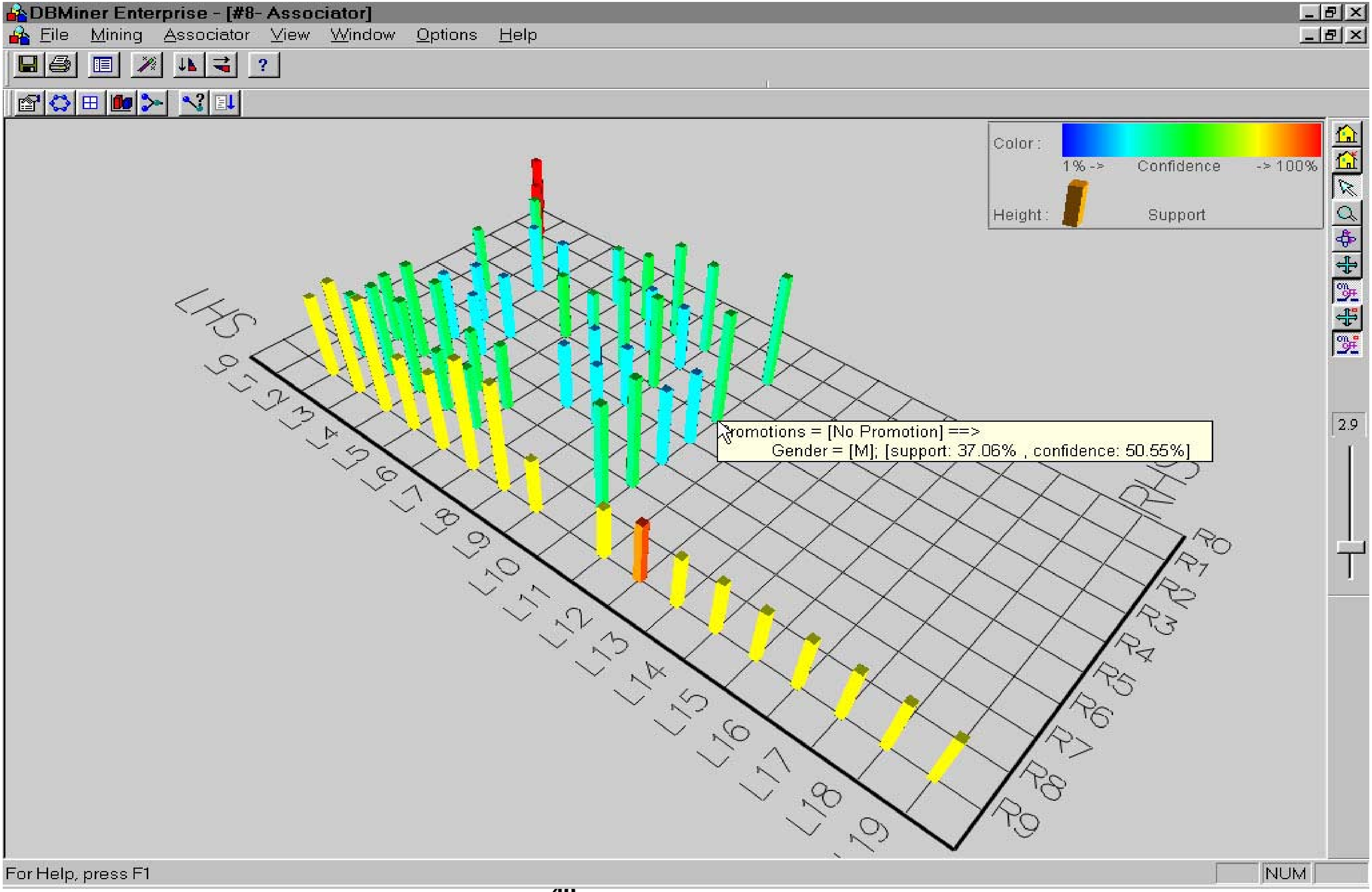
OK

Presentation of Association Rules (Table Form)



	Body	Implies	Head	Supp (%)	Conf (%)	F	G	H	I
1	cost(x) = '0.00~1000.00'	==>	revenue(x) = '0.00~500.00'	28.45	40.4				
2	cost(x) = '0.00~1000.00'	==>	revenue(x) = '500.00~1000.00'	20.46	29.05				
3	cost(x) = '0.00~1000.00'	==>	order_qty(x) = '0.00~100.00'	59.17	84.04				
4	cost(x) = '0.00~1000.00'	==>	revenue(x) = '1000.00~1500.00'	10.45	14.84				
5	cost(x) = '0.00~1000.00'	==>	region(x) = 'United States'	22.56	32.04				
6	cost(x) = '1000.00~2000.00'	==>	order_qty(x) = '0.00~100.00'	12.91	69.34				
7	order_qty(x) = '0.00~100.00'	==>	revenue(x) = '0.00~500.00'	28.45	34.54				
8	order_qty(x) = '0.00~100.00'	==>	cost(x) = '1000.00~2000.00'	12.91	15.67				
9	order_qty(x) = '0.00~100.00'	==>	region(x) = 'United States'	25.9	31.45				
10	order_qty(x) = '0.00~100.00'	==>	cost(x) = '0.00~1000.00'	59.17	71.86				
11	order_qty(x) = '0.00~100.00'	==>	product_line(x) = 'Tents'	13.52	16.42				
12	order_qty(x) = '0.00~100.00'	==>	revenue(x) = '500.00~1000.00'	19.67	23.88				
13	product_line(x) = 'Tents'	==>	order_qty(x) = '0.00~100.00'	13.52	98.72				
14	region(x) = 'United States'	==>	order_qty(x) = '0.00~100.00'	25.9	81.94				
15	region(x) = 'United States'	==>	cost(x) = '0.00~1000.00'	22.56	71.39				
16	revenue(x) = '0.00~500.00'	==>	cost(x) = '0.00~1000.00'	28.45	100				
17	revenue(x) = '0.00~500.00'	==>	order_qty(x) = '0.00~100.00'	28.45	100				
18	revenue(x) = '1000.00~1500.00'	==>	cost(x) = '0.00~1000.00'	10.45	96.75				
19	revenue(x) = '500.00~1000.00'	==>	cost(x) = '0.00~1000.00'	20.46	100				
20	revenue(x) = '500.00~1000.00'	==>	order_qty(x) = '0.00~100.00'	19.67	96.14				
21									
22									
23	cost(x) = '0.00~1000.00'	==>	revenue(x) = '0.00~500.00' AND order_qty(x) = '0.00~100.00'	28.45	40.4				
24	cost(x) = '0.00~1000.00'	==>	revenue(x) = '0.00~500.00' AND order_qty(x) = '0.00~100.00'	28.45	40.4				
25	cost(x) = '0.00~1000.00'	==>	revenue(x) = '500.00~1000.00' AND order_qty(x) = '0.00~100.00'	19.67	27.93				
26	cost(x) = '0.00~1000.00'	==>	revenue(x) = '500.00~1000.00' AND order_qty(x) = '0.00~100.00'	19.67	27.93				
27	cost(x) = '0.00~1000.00' AND order_qty(x) = '0.00~100.00'	==>	revenue(x) = '500.00~1000.00'	19.67	33.23				

Visualization of Association Rules: Plane Graph



Promotions = [No Promotion] ==>
Gender = [M]; [support: 37.06% , confidence: 50.55%]

For Help, press F1

Filtering Association Rules

- Problem: any large dataset can lead to very large number of association rules, even with reasonable Min Confidence and Support
- Confidence by itself is not sufficient
  - e.g. if all transactions include Z, then
  - any rule I => Z will have confidence 100%.
- Other measures to filter rules

Association Rule LIFT

- The *lift* of an association rule *I => J* is defined as:
  - lift = $P(J|I) / P(J)$
  - Note, $P(J)$ = (support of J) / (no. of transactions)
  - ratio of confidence to expected confidence

- Interpretation:
  - if lift > 1, then I and J are positively correlated

    lift < 1, then I are J are negatively correlated.

    lift = 1, then I and J are independent.

Interestingness Measure: Correlations and Lift

- *play basketball* ⇒ *eat cereal* [40%, 66.7%] is misleading
  - The overall percentage of students eating cereal is 75% which is higher than 66.7%.
- *play basketball* ⇒ *not eat cereal* [20%, 33.3%] is more accurate, although with lower support and confidence
- Measure of dependent/correlated events: lift or corr

$$corr_{A,B} = \frac{P(A \cup B)}{P(A)P(B)}$$

	Basketball	Not basketball	Sum (row)
Cereal	2000	1750	3750
Not cereal	1000	250	1250
Sum(col.)	3000	2000	5000

Beyond Binary Data

- Hierarchies
  - drink → milk → low-fat milk → Stop&Shop low-fat milk …
  - find associations on any level
- Sequences over time
- …

Multiple-level Association Rules

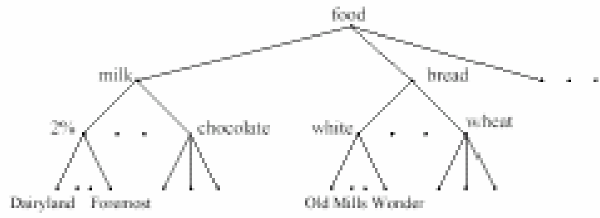


Figure 1: A taxonomy for the relevant data items

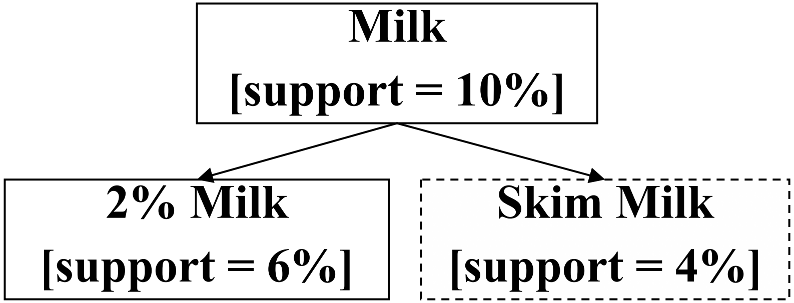
- Items often form hierarchy
- Flexible support settings:
  Items at the lower level are expected to have lower support.
- Transaction database can be encoded based on dimensions and levels
- explore shared multi-level mining

uniform support

reduced support

Level 1
min_sup = 5%

Level 2
min_sup = 5%



Milk
[support = 10%]

2% Milk
[support = 6%]

Skim Milk
[support = 4%]

Level 1
min_sup = 5%

Level 2
min_sup = 3%

Quantitative Association Rules

ID	Age	Salary	Maritial Status	NumCars
100	44	30 000	married	2
200	55	45 000	married	3
300	45	50 000	divorced	1
400	34	44 000	single	0
500	45	38 000	married	2
600	33	44 000	single	2

Sample Rules	Support	Confidence
<age:44..55> and < status: married> ==> <numCars:2>	50%	100%
<NumCars: 0..1> ==> <Married: No>	33%	66,70%

Multi-dimensional Association

- Single-dimensional rules:

  buys(X, “milk”) ⇒ buys(X, “bread”)

- Multi-dimensional rules: ≥ 2 dimensions or predicates
  - Inter-dimension assoc. rules (*no repeated predicates*)

    age(X,”19-25”) ∧ occupation(X,“student”) ⇒ buys(X,“coke”)

  - hybrid-dimension assoc. rules (*repeated predicates*)

    age(X,”19-25”) ∧ buys(X, “popcorn”) ⇒ buys(X, “coke”)

- Categorical Attributes
  - finite number of possible values, no ordering among values
- Quantitative Attributes
  - numeric, implicit ordering among values

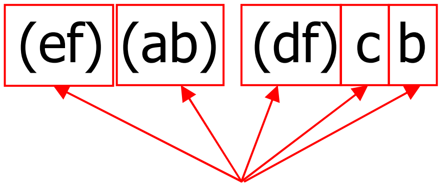
Sequence Databases and Sequential Pattern Analysis

- Transaction databases, time-series databases vs. sequence databases
- Frequent patterns vs. (frequent) sequential patterns
- Applications of sequential pattern mining
  - Customer shopping sequences:
    - First buy computer, then CD-ROM, and then digital camera, within 3 months.
  - Medical treatment, natural disasters (e.g., earthquakes), science & engineering processes, stocks and markets, etc.
  - Telephone calling patterns, Weblog click streams
  - DNA sequences and gene structures

What Is Sequential Pattern Mining?

- Given a set of sequences, find the complete set of *frequent* subsequences

A ***sequence*** : < (ef) (ab) (df) c b >



A ***sequence database***

SID	sequence
10	<a(abc)(ac)d(cf)>
20	<(ad)c(bc)(ae)>
30	<(ef)(ab)(df)cb>
40	<eg(af)cbc>

An element may contain a set of items. Items within an element are unordered and we list them alphabetically.

<a(bc)dc> is a ***subsequence*** of <a(abc)(ac)d(cf)>

Given ***support threshold*** *min_sup* =2, <(ab)c> is a ***sequential pattern***

Challenges on Sequential Pattern Mining

- A huge number of possible sequential patterns are hidden in databases
- A mining algorithm should
  - find the complete set of patterns, when possible, satisfying the minimum support (frequency) threshold
  - be highly efficient, scalable, involving only a small number of database scans
  - be able to incorporate various kinds of user-specific constraints

Applications

- Market basket analysis
  - Store layout, client offers
- This analysis is applicable whenever a customer purchases multiple things in proximity
  - telecommunication (each customer is a transaction containing the set of phone calls)
  - weather analysis (each time interval is a transaction containing the set of observed events)
  - credit cards
  - banking services
  - medical treatments
- Finding unusual events
  - WSARE – What is Strange About Recent Events
- …

Conclusions

- Association rule mining
  - probably the most significant contribution from the database community in KDD
  - A large number of papers have been published
- Many interesting issues have been explored
- An interesting research direction
  - Association analysis in other types of data: sequence data, spatial data, multimedia data, time series data, etc.

Summary

- Frequent itemsets
- Association rules
- Subset property
- Apriori algorithm
- Extensions of this algorithms
- Sequence patterns
- Applications

Any questions, remarks?

?