

# Association Rules Fundamentals




**DBG**  
Data Base and Data Mining Group at Politecnico di Torino

Elena Baralis, Silvia Chiusano  
*Politecnico di Torino*

## Definitions

- Itemset** is a set including one or more items
  - Example: {Beer, Diapers}
- k-itemset** is an itemset that contains k items
- Support count** (#) is the frequency of occurrence of an itemset
  - Example: # {Beer, Diapers} = 2
- Support** is the fraction of transactions that contain an itemset
  - Example:  $sup(\{Beer, Diapers\}) = 2/5$
- Frequent itemset** is an itemset whose support is greater than or equal to a *minsup* threshold

TID	Items
1	Bread, Coke, Milk
2	Beer, Bread
3	Beer, Coke, Diapers, Milk
4	Beer, Bread, Diapers, Milk
5	Coke, Diapers, Milk


4


## Association rules

- Objective**
  - extraction of frequent correlations or pattern from a transactional database

Tickets at a supermarket counter


TID	Items
1	Bread, Coke, Milk
2	Beer, Bread
3	Beer, Coke, Diapers, Milk
4	Beer, Bread, Diapers, Milk
5	Coke, Diapers, Milk
...	...

- Association rule**  
diapers  $\Rightarrow$  beer
  - 2% of transactions contains both items
  - 30% of transactions containing diapers also contains beer


2


## Rule quality metrics

- Given the association rule  $A \Rightarrow B$ 
  - A, B are itemsets
  - Support** is the fraction of transactions containing both A and B
 
$$\frac{\#(A,B)}{|T|}$$
    - |T| is the cardinality of the transactional database
    - a priori probability of itemset AB
    - rule frequency in the database
  - Confidence** is the frequency of B in transactions containing A
 
$$\frac{sup(A,B)}{sup(A)}$$
    - conditional probability of finding B having found A
    - "strength" of the " $\Rightarrow$ "


5

## Association rule mining


- A collection of transactions is given
  - a transaction is a set of items
  - items in a transaction are not ordered
- Association rule**  $A, B \Rightarrow C$ 
  - A, B = items in the *rule body*
  - C = item in the *rule head*
- The  $\Rightarrow$  means co-occurrence
  - not causality
- Examples**
  - cereals, cookies  $\Rightarrow$  milk
  - age < 40, life-insurance = yes  $\Rightarrow$  children = yes
  - customer, relationship  $\Rightarrow$  data, mining


3

## Rule quality metrics: example


- From itemset {Milk, Diapers} the following rules may be derived
- Rule: Milk  $\Rightarrow$  Diapers
  - support  $sup = \# \{Milk, Diapers\} / \#trans. = 3/5 = 60\%$
  - confidence  $conf = \# \{Milk, Diapers\} / \# \{Milk\} = 3/4 = 75\%$
- Rule: Diapers  $\Rightarrow$  Milk
  - same support  $s = 60\%$
  - confidence  $conf = \# \{Milk, Diapers\} / \# \{Diapers\} = 3/3 = 100\%$

TID	Items
1	Bread, Coke, Milk
2	Beer, Bread
3	Beer, Coke, Diapers, Milk
4	Beer, Bread, Diapers, Milk
5	Coke, Diapers, Milk


6

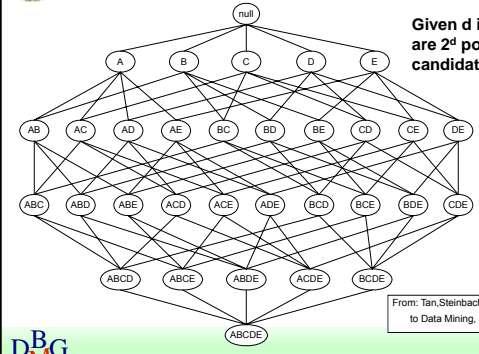
### Association rule extraction

- Given a set of transactions  $T$ , association rule mining is the extraction of the rules satisfying the constraints
  - support  $\geq$  *minsup* threshold
  - confidence  $\geq$  *minconf* threshold
- The result is
  - complete (*all* rules satisfying both constraints)
  - correct (*only* the rules satisfying both constraints)
- May add other more complex constraints




7

### Frequent Itemset Generation



Given  $d$  items, there are  $2^d$  possible candidate itemsets


From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006



10

### Association rule extraction


- Brute-force approach
  - enumerate all possible permutations (i.e., association rules)
  - compute support and confidence for each rule
  - prune the rules that do not satisfy the *minsup* and *minconf* constraints
- Computationally *unfeasible*
- Given an itemset, the extraction process may be split
  - first generate frequent itemsets
  - next generate rules from each frequent itemset
- Example
  - Itemset: {Milk, Diapers} sup=60%
  - Rules: Milk  $\Rightarrow$  Diapers (conf=75%), Diapers  $\Rightarrow$  Milk (conf=100%)



8

### Frequent Itemset Generation


- Brute-force approach
  - each itemset in the lattice is a *candidate* frequent itemset
  - scan the database to count the support of each candidate
    - match each transaction against every candidate
  - Complexity  $\sim O(|T| 2^d w)$ 
    - $|T|$  is number of transactions
    - $d$  is number of items
    - $w$  is transaction length



11

### Association rule extraction


- Extraction of frequent itemsets
  - many different techniques
    - level-wise approaches (Apriori, ...)
    - approaches without candidate generation (FP-growth, ...)
    - other approaches
  - most computationally expensive step
    - limit extraction time by means of support threshold
- Extraction of association rules
  - generation of all possible binary partitioning of each frequent itemset
    - possibly enforcing a confidence threshold



9

### Improving Efficiency

- Reduce the **number of candidates**
  - Prune the search space
    - complete set of candidates is  $2^d$
- Reduce the **number of transactions**
  - Prune transactions as the size of itemsets increases
    - reduce  $|T|$
- Reduce the **number of comparisons**
  - Equal to  $|T| 2^d$
  - Use efficient data structures to store the candidates or transactions




12

### The Apriori Principle

"If an itemset is frequent, then all of its subsets must also be frequent"


- The support of an itemset can never exceed the support of any of its subsets
- It holds due to the antimonotone property of the support measure
  - Given two arbitrary itemsets A and B  
if  $A \subseteq B$  then  $\text{sup}(A) \geq \text{sup}(B)$
- It reduces the number of candidates



13

### Apriori Algorithm [Agr94]

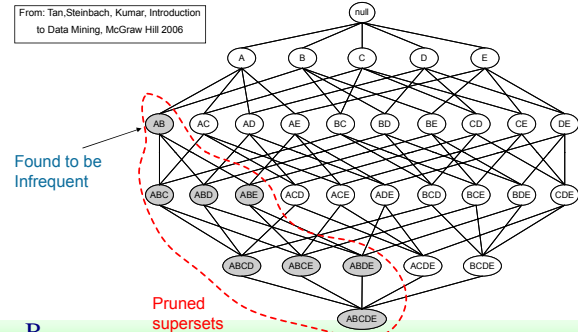

- Pseudo-code
  - $C_k$ : Candidate itemset of size k
  - $L_k$ : frequent itemset of size k
  - $L_1 = \{\text{frequent items}\}$ ;
  - for ( $k = 1$ ;  $L_k \neq \emptyset$ ;  $k++$ ) do
    - begin
    - $C_{k+1}$  = candidates generated from  $L_k$ ;
    - for each transaction  $t$  in database do  
increment the count of all candidates in  $C_{k+1}$   
that are contained in  $t$
    - $L_{k+1}$  = candidates in  $C_{k+1}$  satisfying *minsup*
    - end
  - return  $\cup_k L_k$



16

### The Apriori Principle


From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006

14

### Generating Candidates


- Sort  $L_k$  candidates in lexicographical order
- For each candidate of length k
  - Self-join with each candidate sharing same  $L_{k-1}$  prefix
  - Prune candidates by applying Apriori principle
- Example: given  $L_3 = \{abc, abd, acd, ace, bcd\}$ 
  - Self-join
    - $abcd$  from  $abc$  and  $abd$
    - $acde$  from  $acd$  and  $ace$
  - Prune by applying Apriori principle
    - $acde$  is removed because  $ade$  is not in  $L_3$
    - $C_4 = \{abcd\}$



17

### Apriori Algorithm [Agr94]

- Level-based approach
  - at each iteration extracts itemsets of a given length k
- Two main steps for each level
  - (1) Candidate generation
    - Join Step
      - generate candidates of length k+1 by joining frequent itemsets of length k
    - Prune Step
      - apply Apriori principle: prune length k+1 candidate itemsets that contain at least one k-itemset that is not frequent
  - (2) Frequent itemset generation
    - scan DB to count support for k+1 candidates
    - prune candidates below minsup




15

### Apriori Algorithm: Example

Example DB

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}

minsup > 1



18

### Generate candidate 1-itemsets

Example DB

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}

1st DB scan

$C_1$	
itemsets	sup
{A}	7
{B}	8
{C}	7
{D}	5
{E}	3

minsup > 1

DBG

### Count support for candidates in $C_2$

$L_1$	
itemsets	sup
{A}	7
{B}	8
{C}	7
{D}	5
{E}	3

2nd DB scan

$C_2$	
itemsets	sup
{A,B}	5
{A,C}	4
{A,D}	4
{A,E}	2
{B,C}	6
{B,D}	3
{B,E}	1
{C,D}	3
{C,E}	2
{D,E}	2

DBG

### Prune infrequent candidates in $C_1$

Example DB

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}

1st DB scan

$C_1$	
itemsets	sup
{A}	7
{B}	8
{C}	7
{D}	5
{E}	3

$L_1 \equiv C_1$

- All itemsets in set  $C_1$  are frequent according to minsup > 1

minsup > 1

DBG

### Prune infrequent candidates in $C_2$

$L_1$	
itemsets	sup
{A}	7
{B}	8
{C}	7
{D}	5
{E}	3

2nd DB scan

$C_2$	
itemsets	sup
{A,B}	5
{A,C}	4
{A,D}	4
{A,E}	2
{B,C}	6
{B,D}	3
<del>{B,E}</del>	<del>1</del>
{C,D}	3
{C,E}	2
{D,E}	2

$L_2$	
itemsets	sup
{A,B}	5
{A,C}	4
{A,D}	4
{A,E}	2
{B,C}	6
{B,D}	3
{C,D}	3
{C,E}	2
{D,E}	2

DBG

### Generate candidates from $L_1$

$L_1$	
itemsets	sup
{A}	7
{B}	8
{C}	7
{D}	5
{E}	3

2nd DB scan

$C_2$	
itemsets	sup
{A,B}	
{A,C}	
{A,D}	
{A,E}	
{B,C}	
{B,D}	
{B,E}	
{C,D}	
{C,E}	
{D,E}	

DBG

### Generate candidates from $L_2$

$L_2$	
itemsets	sup
{A,B}	5
{A,C}	4
{A,D}	4
{A,E}	2
{B,C}	6
{B,D}	3
{C,D}	3
{C,E}	2
{D,E}	2

2nd DB scan

$C_3$	
itemsets	sup
{A,B,C}	
{A,B,D}	
{A,B,E}	
{A,C,D}	
{A,C,E}	
{A,D,E}	
{B,C,D}	
{C,D,E}	

DBG

### Apply Apriori principle on $C_3$

$L_2$		$C_3$	
itemsets	sup	itemsets	sup
{A,B}	5	{A,B,C}	3
{A,C}	4	{A,B,D}	2
{A,D}	4	<del>{A,B,E}</del>	
{A,E}	2	{A,C,D}	2
{B,C}	6	{A,C,E}	1
{B,D}	3	{A,D,E}	2
{C,D}	3	{B,C,D}	2
{C,E}	2	{C,D,E}	1
{D,E}	2		

- Prune {A,B,E}
  - Its subset {B,E} is infrequent ({B,E} is not in  $L_2$ )

DBG 25

### Generate candidates from $L_3$

$L_3$		$C_4$	
itemsets	sup	itemsets	sup
{A,B,C}	3	{A,B,C,D}	1
{A,B,D}	2		
{A,C,D}	2		
{A,D,E}	2		
{B,C,D}	2		

DBG 28

### Count support for candidates in $C_3$

$L_2$		$C_3$		$C_3$	
itemsets	sup	itemsets	sup	itemsets	sup
{A,B}	5	{A,B,C}	3	{A,B,C}	3
{A,C}	4	{A,B,D}	2	{A,B,D}	2
{A,D}	4	<del>{A,B,E}</del>		{A,C,D}	2
{A,E}	2	{A,C,D}	2	{A,C,D}	2
{B,C}	6	{A,C,E}	1	{A,C,E}	1
{B,D}	3	{A,D,E}	2	{A,D,E}	2
{C,D}	3	{B,C,D}	2	{B,C,D}	2
{C,E}	2	{C,D,E}	1	{C,D,E}	1
{D,E}	2				

3<sup>rd</sup> DB scan

DBG 26

### Apply Apriori principle on $C_4$

$L_3$		$C_4$	
itemsets	sup	itemsets	sup
{A,B,C}	3	{A,B,C,D}	1
{A,B,D}	2		
{A,C,D}	2		
{A,D,E}	2		
{B,C,D}	2		

- Check if {A,C,D} and {B,C,D} belong to  $L_3$ 
  - $L_3$  contains all 3-itemset subsets of {A,B,C,D}
  - {A,B,C,D} is potentially frequent

DBG 29

### Prune infrequent candidates in $C_3$

$L_2$		$C_3$		$C_3$		$L_3$	
itemsets	sup	itemsets	sup	itemsets	sup	itemsets	sup
{A,B}	5	{A,B,C}	3	{A,B,C}	3	{A,B,C}	3
{A,C}	4	{A,B,D}	2	{A,B,D}	2	{A,B,D}	2
{A,D}	4	<del>{A,B,E}</del>		{A,C,D}	2	{A,C,D}	2
{A,E}	2	{A,C,D}	2	<del>{A,C,E}</del>	1	{A,D,E}	2
{B,C}	6	{A,C,E}	1	{A,D,E}	2	{A,D,E}	2
{B,D}	3	{A,D,E}	2	{B,C,D}	2	{B,C,D}	2
{C,D}	3	{B,C,D}	2	<del>{C,D,E}</del>	1		
{C,E}	2	{C,D,E}	1				
{D,E}	2						

3<sup>rd</sup> DB scan

- {A,C,E} and {C,D,E} are actually infrequent
  - They are discarded from  $C_3$

DBG 27

### Count support for candidates in $C_4$

$L_3$		$C_4$		$C_4$	
itemsets	sup	itemsets	sup	itemsets	sup
{A,B,C}	3	{A,B,C,D}	1	{A,B,C,D}	1
{A,B,D}	2				
{A,C,D}	2				
{A,D,E}	2				
{B,C,D}	2				

4<sup>th</sup> DB scan

DBG 30

### Prune infrequent candidates in $C_4$


$L_3$	
itemsets	sup
{A,B,C}	3
{A,B,D}	2
{A,C,D}	2
{A,D,E}	2
{B,C,D}	2

→  $C_4$  (itemsets: {A,B,C,D})

4<sup>th</sup> DB scan →  $C_4$  (itemsets: {A,B,C,D}, sup: 1)

→  $L_4 = \emptyset$


- {A,B,C,D} is actually infrequent
- {A,B,C,D} is discarded from  $C_4$



31

### Performance Issues in Apriori

- Candidate generation
  - Candidate sets may be huge
    - 2-itemset candidate generation is the most critical step
    - extracting long frequent itemsets requires generating all frequent subsets
- Multiple database scans
  - $n + 1$  scans when longest frequent pattern length is  $n$



34

### Final set of frequent itemsets


Example DB

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}

minsup > 1

$L_1$		$L_2$	
itemsets	sup	itemsets	sup
{A}	7	{A,B}	5
{B}	8	{A,C}	4
{C}	7	{A,D}	4
{D}	5	{A,E}	2
{E}	3	{B,C}	6
		{B,D}	3
		{C,D}	3
		{C,E}	2
		{D,E}	2


$L_3$	
itemsets	sup
{A,B,C}	3
{A,B,D}	2
{A,C,D}	2
{A,D,E}	2
{B,C,D}	2



32

### Factors Affecting Performance


- Minimum support threshold
  - lower support threshold increases number of frequent itemsets
    - larger number of candidates
    - larger (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 in dense data sets
  - may increase max length of frequent itemsets and traversals of hash tree
    - number of subsets in a transaction increases with its width



35

### Counting Support of Candidates


- Scan transaction database to count support of each itemset
  - total number of candidates may be large
  - one transaction may contain many candidates
- Approach [Agr94]
  - 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 candidates contained in a transaction
    - match transaction subsets to candidates in hash tree



33

### Improving Apriori Efficiency

- Hash-based itemset counting [Yu95]
  - A  $k$ -itemset whose corresponding hashing bucket count is below the threshold cannot be frequent
- Transaction reduction [Yu95]
  - A transaction that does not contain any frequent  $k$ -itemset is useless in subsequent scans
- Partitioning [Sav96]
  - Any itemset that is potentially frequent in DB must be frequent in at least one of the partitions of DB



36

### Improving Apriori Efficiency

- Sampling [Toi96]
  - mining on a subset of given data, lower support threshold + a method to determine the completeness
- Dynamic Itemset Counting [Motw98]
  - add new candidate itemsets only when all of their subsets are estimated to be frequent

DBG 37

### FP-tree construction

Example DB

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}

minsup > 1

- (1) Count item support and prune items below minsup threshold
- (2) Build Header Table by sorting items in decreasing support order
- (3) Create FP-tree
  - For each transaction  $t$  in DB
    - order transaction  $t$  items in decreasing support order
      - same order as Header Table
    - insert transaction  $t$  in FP-tree
      - use existing path for common prefix
      - create new branch when path becomes different

DBG 40

### FP-growth Algorithm [Han00]

- Exploits a main memory compressed representation of the database, the FP-tree
  - high compression for dense data distributions
    - less so for sparse data distributions
  - complete representation for frequent pattern mining
    - enforces support constraint
- Frequent pattern mining by means of FP-growth
  - recursive visit of FP-tree
  - applies divide-and-conquer approach
    - decomposes mining task into smaller subtasks
- Only two database scans
  - count item supports + build FP-tree

DBG 38

### FP-tree construction

Transaction

TID	Items
1	{A,B}

Sorted transaction

TID	Items
1	{B,A}

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

FP-tree

DBG 41

### FP-tree construction

Example DB

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}

minsup > 1

- (1) Count item support and prune items below minsup threshold
- (2) Build Header Table by sorting items in decreasing support order

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

DBG 39

### FP-tree construction

Transaction

TID	Items
2	{B,C,D}

Sorted transaction

TID	Items
2	{B,C,D}

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

FP-tree

DBG 42

### FP-tree construction

Transaction

TID	Items
3	{A,C,D,E}

Sorted transaction

TID	Items
3	{A,C,D,E}

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

43

### FP-tree construction

Transaction

TID	Items
6	{A,B,C,D}

Sorted transaction

TID	Items
6	{B,A,C,D}

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

46

### FP-tree construction

Transaction

TID	Items
4	{A,D,E}

Sorted transaction

TID	Items
4	{A,D,E}

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

44

### FP-tree construction

Transaction

TID	Items
7	{B,C}

Sorted transaction

TID	Items
7	{B,C}

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

47

### FP-tree construction

Transaction

TID	Items
5	{A,B,C}

Sorted transaction

TID	Items
5	{B,A,C}

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

45

### FP-tree construction

Transaction

TID	Items
8	{A,B,C}

Sorted transaction

TID	Items
8	{B,A,C}

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

48



### FP-tree construction

Transaction

TID	Items
9	{A,B,D}

Sorted transaction

TID	Items
9	{B,A,D}

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

DBG 49

### FP-growth Algorithm

- Scan Header Table from lowest support item up
- For each item  $i$  in Header Table extract frequent itemsets including item  $i$  and items preceding it in Header Table
  - (1) build Conditional Pattern Base for item  $i$  (i-CPB)
    - Select prefix-paths of item  $i$  from FP-tree
  - (2) recursive invocation of FP-growth on i-CPB

DBG 52

### FP-tree construction

Transaction

TID	Items
10	{B,C,E}

Sorted transaction

TID	Items
10	{B,C,E}

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

DBG 50

### Example

- Consider item D and extract frequent itemsets including D and supported combinations of items A, B, C

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

DBG 53

### Final FP-tree

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

Item pointers are used to assist frequent itemset generation

DBG 51

### Conditional Pattern Base of D

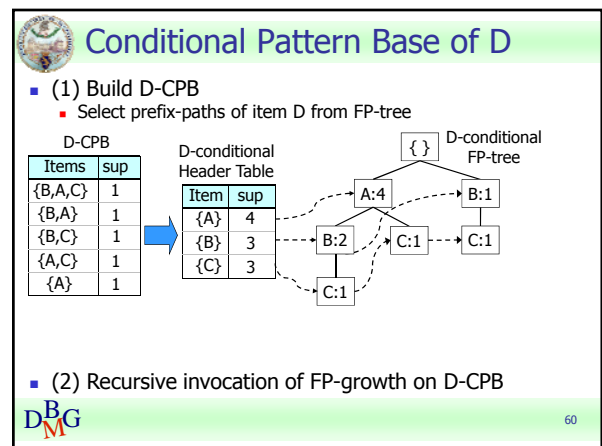
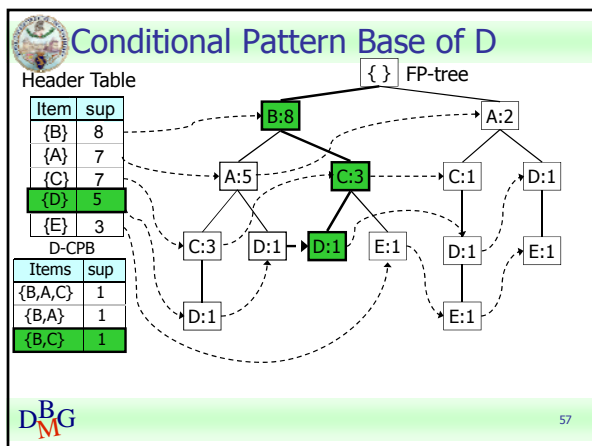
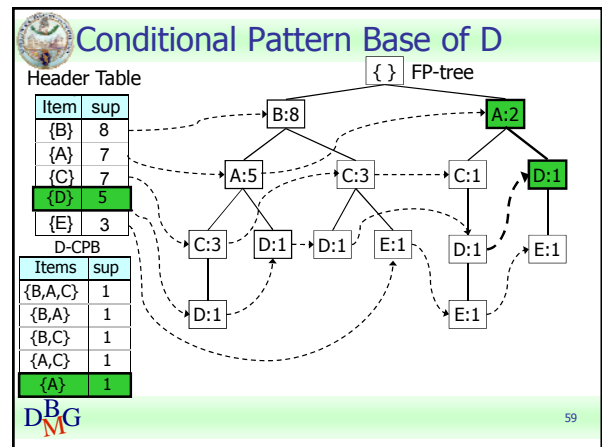
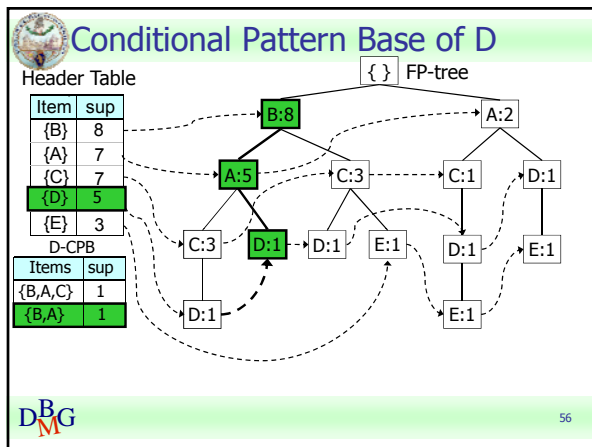
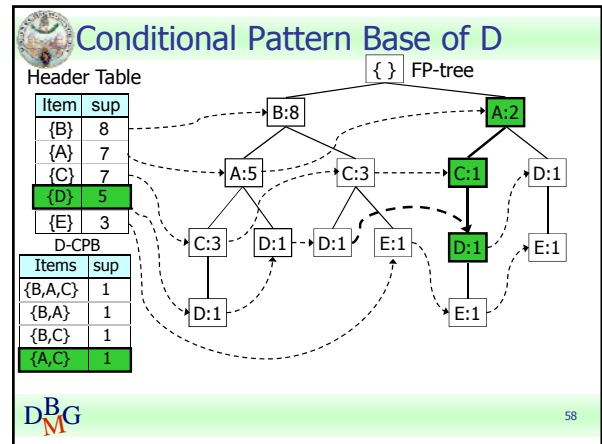
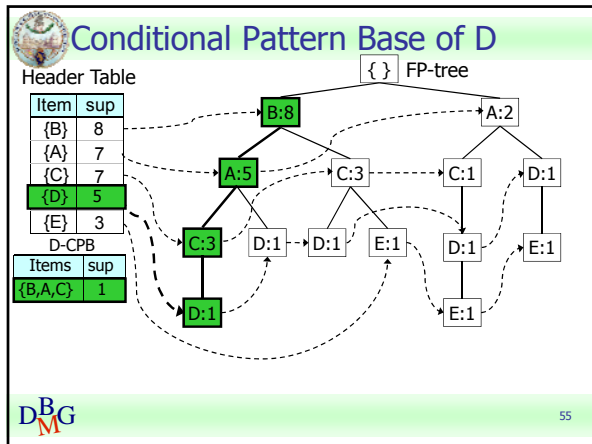
- (1) Build D-CPB
  - Select prefix-paths of item D from FP-tree

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

Frequent itemset:  
D, sup(D) = 5

DBG 54



### Conditional Pattern Base of DC

- (1) Build DC-CPB
  - Select prefix-paths of item C from D-conditional FP-tree

DC-CPB

Items	sup
{B,A,C}	1
{B,A}	1
{B,C}	1
{A,C}	1
{A}	1

D-conditional Header Table

Item	sup
{A}	4
{B}	3
{C}	3

D-conditional FP-tree

```

    {}
   / | \
  A:4 B:1 C:1
 /  \
B:2  C:1
/  \
C:1
    
```

DC-CPB

Items	sup
{A,B}	1
{A}	1
{B}	1

Frequent itemset:  
DC, sup(DC) = 3

### Conditional Pattern Base of DCB

- (1) Build DCB-CPB
  - Select prefix-paths of item B from DC-conditional FP-tree

DCB-CPB

Items	sup
{A}	1

- Item A is infrequent in DCB-CPB
  - A is removed from DCB-CPB
  - DCB-CPB is empty

- (2) The search backtracks to DC-CPB

### Conditional Pattern Base of DC

- (1) Build DC-CPB
  - Select prefix-paths of item C from D-conditional FP-tree

DC-CPB

Items	sup
{A,B}	1
{A}	1
{B}	1

D-conditional Header Table

Item	sup
{A}	2
{B}	2

D-conditional FP-tree

```

    {}
   / | \
  A:2 B:1
 /  \
B:1
    
```

- (2) Recursive invocation of FP-growth on DC-CPB

### Conditional Pattern Base of DCA

- (1) Build DCA-CPB
  - Select prefix-paths of item A from DC-conditional FP-tree

DCA-CPB

Items	sup
{A,B}	1
{A}	1
{B}	1

D-conditional Header Table

Item	sup
{A}	2
{B}	2

D-conditional FP-tree

```

    {}
   / | \
  A:2 B:1
 /  \
B:1
    
```

Frequent itemset:  
DCA, sup(DCA) = 2

DCA-CPB is empty (no transactions)

- (2) The search backtracks to D-CPB

### Conditional Pattern Base of DCB

- (1) Build DCB-CPB
  - Select prefix-paths of item B from DC-conditional FP-tree

DCB-CPB

Items	sup
{A,B}	1
{A}	1
{B}	1

D-conditional Header Table

Item	sup
{A}	2
{B}	2

D-conditional FP-tree

```

    {}
   / | \
  A:2 B:1
 /  \
B:1
    
```

Frequent itemset:  
DCB, sup(DCB) = 2

### Conditional Pattern Base of DB

- (1) Build DB-CPB
  - Select prefix-paths of item B from D-conditional FP-tree

DB-CPB

Items	sup
{B,A,C}	1
{B,A}	1
{B,C}	1
{A,C}	1
{A}	1

D-conditional Header Table

Item	sup
{A}	4
{B}	3
{C}	3

D-conditional FP-tree

```

    {}
   / | \
  A:4 B:1 C:1
 /  \
B:2  C:1
/  \
C:1
    
```

Frequent itemset:  
DB, sup(DB) = 3

### Conditional Pattern Base of DB

- (1) Build DB-CPB
  - Select prefix-paths of item B from D-conditional FP-tree

Items	sup
{A}	2

DB-conditional Header Table

Items	sup
{A}	2

DB-conditional FP-tree

```

graph TD
    Root["{}"] --- Node["A:2"]
            
```

- (2) Recursive invocation of FP-growth on DB-CPB

DBG 67

### Frequent itemsets with prefix D

- Frequent itemsets including D and supported combinations of items B,A,C

Example DB

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}

itemsets	sup
{D}	5
{A,D}	4
{B,D}	3
{C,D}	3
{A,B,D}	2
{A,C,D}	2
{B,C,D}	2

minsup > 1

DBG 70

### Conditional Pattern Base of DBA

- (1) Build DBA-CPB
  - Select prefix-paths of item A from DB-conditional FP-tree

Items	sup
{A}	2

DB-conditional Header Table

Items	sup
{A}	2

DB-conditional FP-tree

```

graph TD
    Root["{}"] --- Node["A:2"]
            
```

Frequent itemset: DBA, sup(DBA) = 2

DBA-CPB is empty (no transactions)

- (2) The search backtracks to D-CPB

DBG 68

### Other approaches

- Many other approaches to frequent itemset extraction
  - some covered later
- May exploit a different database representation
  - represent the tidset of each item [Zak00]

Horizontal Data Layout

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

Vertical Data Layout

	A	B	C	D	E
1	1	1	2	2	1
2	4	2	3	4	3
3	5	5	4	5	6
4	6	7	8	9	
5	7	8	9		
6	8	9			
7		10			
8					
9					
10					

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006

DBG 71

### Conditional Pattern Base of DA

- (1) Build DA-CPB
  - Select prefix-paths of item A from D-conditional FP-tree

Items	sup
{B,A,C}	1
{B,A}	1
{B,C}	1
{A,C}	1
{A}	1

D-conditional Header Table

Item	sup
{A}	4
{B}	3
{C}	3

D-conditional FP-tree

```

graph TD
    Root["{}"] --- NodeA["A:4"]
    NodeA --- NodeB["B:1"]
    NodeA --- NodeC["C:1"]
    NodeB --- NodeC2["C:1"]
    NodeC --- NodeC3["C:1"]
            
```

Frequent itemset: DA, sup(DA) = 4

DA-CPB is empty (no transactions)

The search ends

DBG 69

### Compact Representations

- Some itemsets are redundant because they have identical support as their supersets

TID	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1

- Number of frequent itemsets =  $3 \times \sum_{k=1}^{10} \binom{10}{k}$
- A compact representation is needed

DBG 72

### Maximal Frequent Itemset

An itemset is frequent maximal if none of its immediate supersets is frequent

DBGM 73

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006

### Maximal vs Closed Frequent Itemsets

Minimum support = 2

# Closed = 9  
# Maximal = 4

DBGM 76

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006

### Closed Itemset

An itemset is closed if none of its immediate supersets has the same support as the itemset

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

itemset	sup
{A}	4
{B}	5
{C}	3
{D}	4
{A,B}	4
{A,C}	2
{A,D}	3
{B,C}	3
{B,D}	4
{C,D}	3

itemset	sup
{A,B,C}	2
{A,B,D}	3
{A,C,D}	2
{B,C,D}	3
{A,B,C,D}	2

DBGM 74

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006

### Maximal vs Closed Frequent Itemsets

DBGM 77

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006

### Maximal vs Closed Itemsets

TID	Items
1	ABC
2	ABCD
3	BCE
4	ACDE
5	DE

Not supported by any transactions

DBGM 75

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006

### Effect of Support Threshold

- Selection of the appropriate *minsup* threshold is not obvious
  - If *minsup* is too high
    - itemsets including rare but interesting items may be lost
      - example: pieces of jewellery (or other expensive products)
  - If *minsup* is too low
    - it may become computationally *very expensive*
    - the number of frequent itemsets becomes *very large*

DBGM 78

### Interestingness Measures

- A large number of pattern may be extracted
  - rank patterns by their interestingness
- Objective measures
  - rank patterns based on statistics computed from data
  - initial framework [Agr94] only considered support and confidence
    - other statistical measures available
- Subjective measures
  - rank patterns according to user interpretation [Silb98]
    - interesting if it contradicts the expectation of a user
    - interesting if it is actionable

DBG 79

### Example

- Association rule
  - play basket  $\Rightarrow$  eat cereals
  - has corr = 0.89
    - negative correlation
- but rule
  - play basket  $\Rightarrow$  not (eat cereals)
  - has corr = 1,34

DBG 82

### Confidence measure: always reliable?

- 5000 high school students are given
  - 3750 eat cereals
  - 3000 play basket
  - 2000 eat cereals and play basket
- Rule
  - play basket  $\Rightarrow$  eat cereals
  - sup = 40%, conf = 66,7%

is misleading because eat cereals has sup 75% (>66,7%)

- Problem caused by high frequency of rule head
  - negative correlation

	basket	not basket	total
cereals	2000	1750	3750
not cereals	1000	250	1250
total	3000	2000	5000

DBG 80

#	Measure	Formula
1	$\phi$ -coefficient	$\frac{P(A,B) - P(A)P(B)}{\sqrt{P(A)P(B)(1-P(A))(1-P(B))}}$
2	Goodman-Kruskal's $\lambda$	$\frac{\sum_{i=1}^m \max(P(A_i, B) - \min(P(A_i), P(B)), 0) - \max(P(A), P(B))}{2 - \max(P(A), P(B)) - \max(P(B), P(A))}$
3	Odds ratio ( $\alpha$ )	$\frac{P(A,B)P(\bar{A},\bar{B})}{P(A,\bar{B})P(\bar{A},B)}$
4	Yule's Q	$\frac{P(A,B)P(\bar{A},\bar{B}) - P(A,\bar{B})P(\bar{A},B)}{P(A,B)P(\bar{A},\bar{B}) + P(A,\bar{B})P(\bar{A},B)} = \frac{\alpha - 1}{\alpha + 1}$
5	Yule's Y	$\frac{\sqrt{P(A,B)P(\bar{A},\bar{B})} - \sqrt{P(A,\bar{B})P(\bar{A},B)}}{\sqrt{P(A,B)P(\bar{A},\bar{B})} + \sqrt{P(A,\bar{B})P(\bar{A},B)}} = \frac{\alpha - 1}{\alpha + 1}$
6	Kappa ( $\kappa$ )	$\frac{P(A,B) + P(\bar{A},\bar{B}) - P(A,\bar{B}) - P(\bar{A},B)}{1 - P(A)P(B) - P(\bar{A},\bar{B})}$
7	Mutual Information (MI)	$\frac{P(A,B) \log \frac{P(A,B)}{P(A)P(B)}}{\sum_i \sum_j P(A_i, B_j) \log \frac{P(A_i, B_j)}{P(A_i)P(B_j)}}$
8	J-Measure (J)	$\max(P(A,B) \log \frac{P(A,B)}{P(A)P(B)}, P(\bar{A},\bar{B}) \log \frac{P(\bar{A},\bar{B})}{P(\bar{A})P(\bar{B})})$
9	Gini index (G)	$\max(P(A)[P(B A)^2 + P(\bar{B} A)^2] + P(\bar{A})[P(B \bar{A})^2 + P(\bar{B} \bar{A})^2] - P(B)^2 - P(\bar{B})^2)$
10	Support (s)	$P(A, B)$
11	Confidence (c)	$\frac{P(A, B)}{P(A)}$
12	Laplace (L)	$\max\left(\frac{P(A, B) + 1}{N(A) + 1}, \frac{P(\bar{A}, \bar{B}) + 1}{N(\bar{A}, \bar{B}) + 1}\right)$
13	Conviction (V)	$\max\left(\frac{P(A, \bar{B}) + P(\bar{A}, B)}{P(A, \bar{B})}, \frac{P(\bar{A}, \bar{B})}{P(\bar{A}, \bar{B})}\right)$
14	Interest (I)	$\frac{P(A, B)}{P(A)P(B)}$
15	cosine (CS)	$\frac{P(A, B)}{\sqrt{P(A)P(B)}}$
16	Plattensky-Shapiro's (PS)	$\frac{P(A, B) - P(A)P(B)}{1 - P(A)P(B)}$
17	Certainty factor (CF)	$\frac{P(A, B) - P(A)P(B)}{1 - P(A)}$
18	Added Value (AV)	$\max(P(B A) - P(B), P(A B) - P(A))$
19	Collective strength (S)	$\frac{P(A, B) + P(\bar{A}, \bar{B})}{P(A)P(B) + P(\bar{A}, \bar{B})} \times \frac{1 - P(A)P(B) - P(\bar{A}, \bar{B})}{1 - P(A)P(B) - P(\bar{A}, \bar{B})}$
20	Jaccard (C)	$\frac{P(A, B)}{P(A) + P(B) - P(A, B)}$
21	Klirgen (K)	$\sqrt{\frac{P(A, B)}{P(A)P(B)}} \max(P(B A) - P(B), P(A B) - P(A))$

DBG

### Correlation or lift

$r: A \Rightarrow B$

$$\text{Correlation} = \frac{P(A, B) - P(A)P(B)}{P(A)P(B)} = \frac{\text{conf}(r)}{\text{sup}(B)}$$

- Statistical independence
  - Correlation = 1
- Positive correlation
  - Correlation > 1
- Negative correlation
  - Correlation < 1

DBG 81