Data Mining II

Association Rules

Rita P. Ribeiro

2016/2017

Computer Science Department

FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

(based on slides from Alípio Jorge)

Summary

Association Rules
Basic Concepts

Association Rules: a New Data Mining Task

Data Mining Tasks:

- Prediction
  - Classification
  - Regression
  - ...
- Description
  - Clustering
  - Association Rules
    - find relationships / associations between groups of variables
  - ...

Motivation

Originally stated in the context of Market Basket Analysis

- Data consists of set of items bought by costumers, referred as transactions
- Find unexpected associations between sets of items using the frequency of sets of items
- Discovered sets of items are referred as frequent itemsets or frequent patterns
- Goals
  - Store layout - *Should products A and B be placed together?*
  - Promotions - *If the client is interested in {A,B,C,...}, can we guess other interests?*
  - ...

Market Basket Analysis

Market Baskets data set

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

Products are converted in binary flags

$\rightarrow$

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

Market Basket Analysis: how frequent is an itemset?

- Sugar, Flower and Eggs are sold together

- How important is this set?

- **Support** measures the importance of a set
  - Percentage of transactions *t* containing the set *S*
  - Absolute support: number of transactions *t* containing the set *S*

Market Basket Analysis: how predictive is an itemset?

- Frequent itemsets are used to generate association rules.
- If you buy sugar and flower, you also buy eggs.
- How strong is this rule?
- **Confidence** measures the strength of the rule
  - Percentage of transactions *t* that having sugar and flower also have eggs

$\rightarrow$


Basic Concepts

- Given a data set $D = \{$ transactions $t \mid t$ is a set of items $i \in I\}$
- An association rule is defined as and implication $X \rightarrow Y$, where
  - X and Y are itemsets, i.e. $X, Y \subseteq I$
  - $X \neq \emptyset$, $Y \neq \emptyset$ and $X \cap Y = \emptyset$
- $sup(X)$ is the proportion of transactions in D that include the itemset X, i.e. estimated probability of X, $P(X)$
- $sup(X \rightarrow Y) = sup(X \cup Y)$, i.e. $P(X \cup Y)$
- $conf(X \rightarrow Y) = sup(X \cup Y)/sup(X)$, i.e. $P(Y|X)$

Association Rules: an example

Given the data

Transactions ID	Items Bought
100	A, B, C
200	A, C
150	A, D
500	B, E, F

$\rightarrow$

TID	A	B	C	D	E	F
100	1	1	1	0	0	0
200	1	0	1	0	0	0
150	1	0	0	1	0	0
500	0	1	0	0	1	1

- The itemsets with a minimum support of 50%

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

- Rules with minimum support of 50% and minimum confidence of 50%
  - $A \rightarrow C$
    - $sup(A \rightarrow C) = sup(\{A, C\}) = 50\%$
    - $conf(A \rightarrow C) = sup(\{A, C\})/sup(\{A\}) = 66.6\%$
  - $C \rightarrow A$
    - $sup(C \rightarrow A) = sup(\{A, C\}) = 50\%$
    - $conf(C \rightarrow A) = sup(\{A, C\})/sup(\{C\}) = 100\%$

Given

Cliente	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13
1	1	1	0	0	1	0	0	0	0	0	0	0	0
2	0	0	1	0	0	1	0	0	0	0	0	0	0
3	1	0	1	1	1	0	0	0	0	0	0	0	0
4	1	1	1	0	1	0	0	0	0	0	0	0	0
5	0	0	1	0	0	1	0	1	1	1	0	0	0
6	0	1	0	0	0	0	0	1	0	1	0	0	0
7	1	0	0	0	0	0	1	1	0	1	0	1	1
8	0	1	0	0	0	0	0	1	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	1	0	1	0

Calculate

- Support of
  - $\{A3\}$
  - $\{A3, A5\}$
  - $\{A3, A5, A1\}$
- Confidence of
  - $\{A3\} \rightarrow \{A4\}$
  - $\{A3\} \rightarrow \{A5\}$
  - $\{A3, A5\} \rightarrow \{A1\}$
  - $\{A3, A5\} \rightarrow \{A1, A4\}$

Classification *versus* Association

	Classification	Association
Consequent of rule	1 atom	n atoms
Rule redundancy	little or none	high
Nr. of rules	low	high
Data mining task	supervised	unsupervised
	one target attribute	all attributes are “equal”

Association Rules in Action

Actionable Knowledge: shop layout

- Possible actions from rule $\{A1, A4\} \rightarrow \{A6\}$
  - Sell the $A1$, $A4$, $A6$ together (pack)
  - Place article $A6$ next to articles $A1$, $A4$
  - Offer a discount coupon for $A6$ in articles $A1$, $A4$
  - Place a competitor of A6 next to $A1$, $A4$ (brand protection).
- Note
  - These actions must make sense from the business point of view.

Actionable Knowledge: cross selling

- Steps
  - Client puts article A in basket
  - Shop knows rule $A \rightarrow B$
  - Rule has enough confidence (> 20%)
  - Shop tells client he may be interested in B
  - Client decides whether to buy B or not
- Notes
  - Rules are discovered from business records
  - Discovery (mining) can be made off-line
  - Use of rules can be made on-line

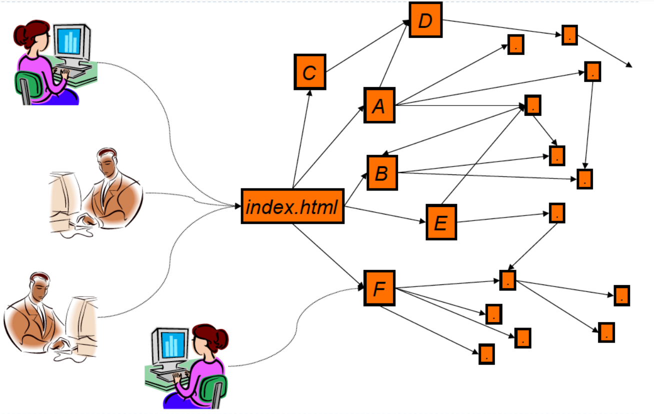
Actionable Knowledge: text mining

- Each document is treated as a "bag" of terms and keywords
  - doc1: Student, Teach, School (Education)
  - doc2: Student, School (Education)
  - doc3: Teach, School, City, Game (Education)
  - doc4: Baseball, Basketball (Sport)
  - doc5: Basketball, Player, Spectator (Sport)
  - doc6: Baseball, Coach, Game, Team (Sport)
  - doc7: Basketball, Team, City, Game (Sport)
- Goal: identify co-occurring terms and keywords
- Example:
  - Student, School $\rightarrow$ Education
  - Game $\rightarrow$ Sport

Actionable Knowledge: health

- Each patient visits a health unit one or more times
  - We record the observations for each visit
    - Symptoms (head ache, temperature)
    - Exam results (blood pressure, sugar level)
  - A set of observations may fire a rule
    {Head ache, blood pressure rise} $\rightarrow$ {stroke, immobilization}
  - Sooner prevention
  - Rules obtained from the patient's records

Actionable Knowledge: web usage analysis



Actionable Knowledge: web usage analysis (cont.)

Usage patterns

- Most visited pages
- Frequent page sets
  - Site structure
- Pages associated to users
  - personalization
- Seasonal effects
  - operations, campaigns
- Cross-preferences
  - cross-selling

Actionable Knowledge: web usage analysis (cont.)

From weblogs to rules

web access logs	$\rightarrow$	user sessions	$\rightarrow$	page "baskets"	$\rightarrow$	association rules

Actionable Knowledge: web usage analysis (cont.)

Web access logs

IP	date	time	url
194.65.227.7	30-12-1997	0:00:02	/verdemo/tema16/tema16.HTM
194.65.227.7	30-12-1997	0:00:02	/verdemo/gifs/infoline.gif
194.65.227.7	30-12-1997	0:00:12	/verdemo/tema16/sb1601/info1601.htm
194.65.227.7	30-12-1997	0:00:13	/verdemo/tema16/tema16.htm
194.65.227.7	30-12-1997	0:00:13	/verdemo/tema16/sb1601/sub1601.htm
194.65.255.18	30-12-1997	0:00:13	/si/apresent/apresent.html
194.65.227.7	30-12-1997	0:00:15	/verdemo/gifs/back3.gif
194.65.255.18	30-12-1997	0:00:15	/si/gifs/bg6.GIF
194.65.255.18	30-12-1997	0:00:17	/si/gifs/botapr.GIF
194.65.255.18	30-12-1997	0:00:17	/si/gifs/bola.GIF
194.65.255.18	30-12-1997	0:00:18	/si/gifs/barr1-2.GIF

Taxonomy of pages

Theme

↓

Sub-theme

↓

Topic

↓

URL

Actionable Knowledge: web usage analysis (cont.)

Sessions / users

IP	date	time	url
194.65.227.7	30-12-1997	0:00:02	/verdemo/tema16/tema16.HTM
194.65.227.7	30-12-1997	0:00:02	/verdemo/gifs/infoline.gif
194.65.227.7	30-12-1997	0:00:12	/verdemo/tema16/sb1601/info1601.htm
194.65.227.7	30-12-1997	0:00:13	/verdemo/tema16/tema16.htm
194.65.227.7	30-12-1997	0:00:13	/verdemo/tema16/sb1601/sub1601.htm
194.65.255.18	30-12-1997	0:00:13	/si/apresent/apresent.html
194.65.227.7	30-12-1997	0:00:15	/verdemo/gifs/back3.gif
194.65.255.18	30-12-1997	0:00:15	/si/gifs/bg6.GIF
194.65.255.18	30-12-1997	0:00:17	/si/gifs/botapr.GIF
194.65.255.18	30-12-1997	0:00:17	/si/gifs/bola.GIF
194.65.255.18	30-12-1997	0:00:18	/si/gifs/barr1-2.GIF

Actionable Knowledge: web usage analysis (cont.)

Processed data (user_id and theme)

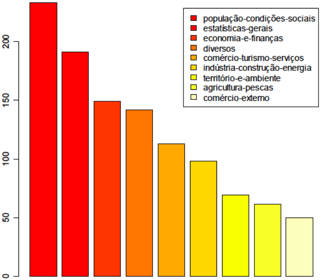
cesto

USER ID	TEMA
acporto	comércio-externo
acporto	comércio-turismo-serviços
agine181	estatísticas-gerais
alggp0157	estatísticas-gerais
alggp0218	economia-e-finanças
aline003	estatísticas-gerais
aline003	território-e-ambiente
aline003	população-condições-sociais
aline003	comércio-turismo-serviços
aline024	comércio-turismo-serviços
aline025	economia-e-finanças
aline025	diversos
aline029	estatísticas-gerais
aline029	economia-e-finanças
aline029	comércio-turismo-serviços
aline032	população-condições-sociais
aline043	economia-e-finanças
aline043	comércio-turismo-serviços
aline065	população-condições-sociais
aline086	agricultura-pescas
...	

Actionable Knowledge: web usage analysis (cont.)

Frequency of visited pages (by theme)

	Frequência
população-condições-sociais	233
estatísticas-gerais	191
economia-e-finanças	149
diversos	142
comércio-turismo-serviços	113
indústria-construção-energia	98
território-e-ambiente	69
agricultura-pescas	61
comércio-externo	50



Actionable Knowledge: web usage analysis (cont.)

Derived association rules

Regras	Suporte	Confiança
diversos & economia-e-finanças & população-condições-sociais -> estatísticas-gerais	0,06	0,97
diversos & economia-e-finanças -> estatísticas-gerais	0,09	0,85
comércio-turismo-serviços & diversos & população-condições-sociais -> estatísticas-gerais	0,05	0,84
comércio-turismo-serviços & estatísticas-gerais & população-condições-sociais -> diversos	0,05	0,84
território-e-ambiente & diversos -> estatísticas-gerais	0,06	0,83
comércio-turismo-serviços & diversos & estatísticas-gerais -> população-condições-sociais	0,05	0,82
indústria-construção-energia & estatísticas-gerais -> diversos	0,06	0,77
indústria-construção-energia & economia-e-finanças -> estatísticas-gerais	0,06	0,77
economia-e-finanças & estatísticas-gerais & população-condições-sociais -> diversos	0,06	0,77
indústria-construção-energia & população-condições-sociais -> diversos	0,05	0,76

- diverse & economy-and-finance $\rightarrow$ general-statistics (sup=9%,conf=85%)
- This means that:
  - “9% of the users visit pages of these 3 themes”
  - “85% of the users interested in diverse and economy-and-finance are also interested in general statistics”

Mining Association Rules

Problem Definition

- Given:
  - data set of transactions *D*
  - minimal support *minsup*
  - minimal confidence *minconf*
- Obtain:
  - **all** association rules

    $$X \rightarrow Y \quad (s = Sup, c = Conf)$$

    such that

    $$Sup \geq minsup \text{ and } Conf \geq minconf$$

The **Apriori Algorithm** [Agrawal and Srikant, 1994] works in two steps:

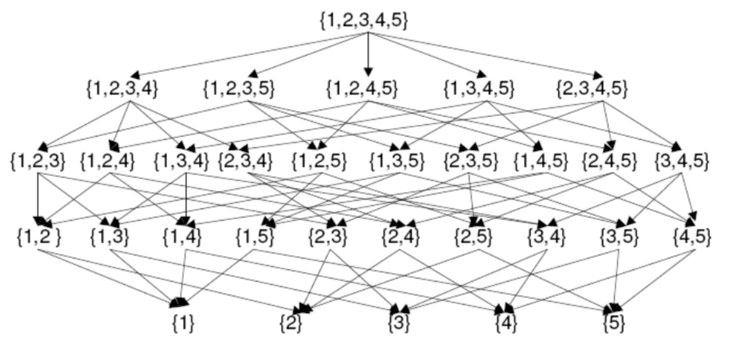
1. **Frequent itemset generation**
   - itemsets with *support* $\geq$ *minsup*

2. **Rule generation**
   - generate all confident association rules from the frequent itemsets, i.e. rules with *confidence* $\geq$ *minconf*

Apriori Algorithm (cont.)

- Problem:
  - there is a very large number of candidate frequent itemsets!
    - for transactions with k items, there are $2^k - 1$ distinct subsets.
- Downward Closure Property
  - every subset of a frequent itemset must also be frequent.
    - ex: if $\{A1, A2, A4\}$ is frequent, so is $\{A1, A2\}$ because every transaction containing $\{A1, A2, A4\}$ also contains $\{A1, A2\}$.
  - thus, every superset of an infrequent itemset is also infrequent.
    - ex: if $\{A1, A2\}$ is infrequent, so is $\{A1, A2, A4\}$.
- Apriori Pruning Principle:
  - if an itemset is below the minimal support, discard all its supersets.

Search Space for 5 items



Example - 1 (cont.)

- Apriori enumerates and counts the support of patterns with increasing length.
- Starts looking for frequent itemsets of size 1 (F_1), assuming *minsup* $= 50\%$ (2 transactions)
- $C_1 = \{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}\}$

TID	ITEM-SET
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

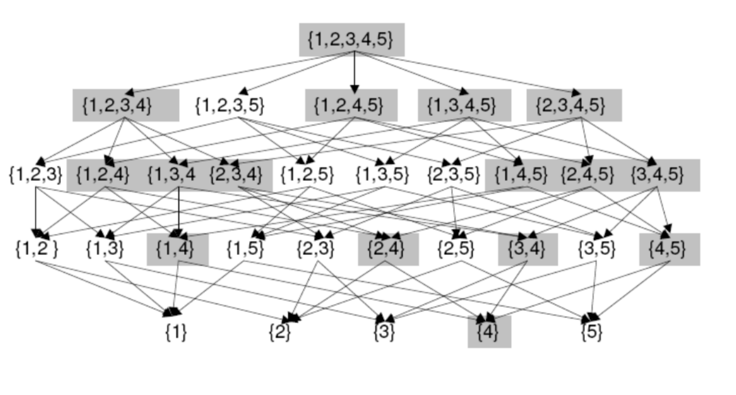
→

ITEM-SET	Support
{1}	2
{2}	3
{3}	3
{4}	1
{5}	3

- $F_1 = \{\{1\}, \{2\}, \{3\}, \{5\}\}$

Example - 1 (cont.)

- Filtered Search Space for 5 items (after removing item "4")



Example - 1 (cont.)

- Looks for frequent itemsets of size 2 (F_2) from frequent itemsets of size 1 (F_1)
- Candidates $C_2 = \{\{a, b\} | \{a\} \in F_1 \wedge \{b\} \in F_1\}$
- $C_2 = \{\{1, 2\}, \{1, 3\}, \{1, 5\}, \{2, 3\}, \{2, 5\}, \{3, 5\}\}$

ITEM-SET	Support
{1,2}	1
{1,3}	2
{1,5}	1
{2,3}	2
{2,5}	3
{3,5}	2

- $F_2 = \{\{1, 3\}, \{2, 3\}, \{2, 5\}, \{3, 5\}\}$

- Looks for frequent itemsets of size 3 (F_3) from frequent itemsets of size 2 (F_2)
- Generation:
  $C0_3 = \{\{a, b, c\} | \{a, b\} \in F_2 \wedge \{a, c\} \in F_2\}$
- Filter:
  $C_3 = \{\{a, b, c\} | \{a, b, c\} \in C0_3 \wedge \forall x \in \{a, b, c\}\ S - \{x\} \in F_2\}$
- $C_3 = \{\{2, 3, 5\}\}$

ITEM-SET	Suporte
{2,3,5}	2

- $F_3 = \{\{2, 3, 5\}\}$
- There are no frequent itemsets of size 4

Example - 2

A	B	C	D
1			
1	1	1	
		1	
1	1	1	1
	1		
1			1
1	1	1	
		1	1
1	1	1	

Pass 1

- $minsup = 0.4$
- $C_1 = \{\{A\}, \{B\}, \{C\}, \{D\}\}$
- $F_1 = \{\{A\}, \{B\}, \{C\}\}$

Example - 2 (cont.)

A	B	C	D
1			
1	1	1	
		1	
1	1	1	1
	1		
1			1
1	1	1	
		1	1
1	1	1	

Pass 2

- $minsup = 0.4$
- $C_2 = \{\{A, B\}, \{A, C\}, \{B, C\}\}$
- $F_2 = \{\{A, B\}, \{A, C\}, \{B, C\}\}$

A	B	C	D
1			
1	1	1	
		1	
1	1	1	1
	1		
1			1
1	1	1	
		1	1
1	1	1	

Pass 3

- $minsup = 0.4$
- $C_3 = \{\{A, B, C\}\}$
- $F_3 = \{\{A, B, C\}\}$

Example - 2 (cont.)

Output

- frequent itemsets (*minsup* $= 0.4$)

$\{A\}$ 66%	$\{B\}$ 55%	$\{C\}$ 66%
$\{A, B\}$ 44%	$\{A, C\}$ 44%	$\{B, C\}$ 44%
$\{A, B, C\}$ 44%		

- rules (*minconf* $= 0.8$)

$\{B\} \rightarrow \{A\}$	(sup = 44%, conf = 80%)
$\{B\} \rightarrow \{C\}$	(sup = 44%, conf = 80%)
$\{B, C\} \rightarrow \{A\}$	(sup = 44%, conf = 100%)
$\{B, A\} \rightarrow \{C\}$	(sup = 44%, conf = 100%)
$\{B\} \rightarrow \{A, C\}$	(sup = 44%, conf = 80%)

Step 1 - identifying frequent itemsets

```
function APRIORI-SETS(D,minsup)
    C_1 ← {{i} | i ∈ I}        // I is the set of items in D
    F_1 ← {{f} | f ∈ C_1 ∧ sup(f) ≥ n × minsup}        // n is the nr. of transactions in D
    k ← 2
    while F_{k−1} ≠ ∅ do
        C_k ← APIORI-GEN(F_{k−1})        // candidate generation
        for all c ∈ C_k do
            sup(c) ← 0
        end for
        for transaction t ∈ D do
            for candidate c ∈ C_k do
                if c ⊆ t then
                    sup(c) ← sup(c) + 1
                end if
            end for
        end for
        F_k ← {c | c ∈ C_k ∧ sup(c) ≥ n × minsup}        // extract frequent k-itemsets
    end while
    return ⋃_k F_k
end function
```

Step 1 - identifying frequent itemsets (cont.)

- It is a **level-wise** algorithm
  - it traverses the itemset lattice one level at a time, from frequent 1-itemsets to the maximum size of frequent itemsets.
- It employs a **generate-and-test** strategy for finding frequent itemsets
  - at each iteration, new candidate itemsets are generated from the frequent itemsets found in the previous iteration; the support for each candidate itemset is then counted and tested against minsup.

Step 1 - identifying frequent itemsets (cont.)

- Candidate generation (Self-Join step)
  - generates new candidate k-itemsets based on the frequent (k-1)-itemsets found in the previous iteration.
- Candidate pruning (Prune step)
  - eliminates some of the candidate k-itemsets using the support-based pruning strategy.

Step 1 - identifying frequent itemsets (cont.)

- Self-Join Example:

  Given the size k candidates
  $\{A, B, C\}$
  $\{A, B, D\}$
  $\{A, C, D\}$
  $\{B, C, D\}$
  $\{A, B, E\}$
  $\{B, C, E\}$
  and assuming that in each itemset the items are lexicographically sorted

- Which are the candidates of size $k + 1$?
- What is the most efficient way of finding them (without repetitions)?

Step 1 - identifying frequent itemsets (cont.)

- Look for pairs of sets with the same prefix of size $k-1$
  $\{A, B, C\}$ and $\{A, B, D\}$
- Combine both, keeping the prefix
  $\{A, B, C, D\}$
- This way
  - No frequent set is unnoticed
  - No candidate is generated more than once

Step 1 - identifying frequent itemsets (cont.)

```
function APRIORI-GEN(F_{k-1})
    C_k ← ∅        // initialize the set of candidates
    for all f_1, f_2 ∈ F_{k-1}        // find all pairs of frequent itemsets
         with f_1 = {i_1, ··· , i_{k-2}, i_{k-1}}        // that differ only in the last item
        and f_2 = {i_1, ··· , i_{k-2}, i'_{k-1}}
        and i_{k-1} < i'_{k-1} do        // according to lexicographic order
        c ← {i_1, ··· , i_{k-1}, i'_{k-1}}        // Self-Join of f_1 and f_2
        C_k ← C_k ∪ c
        for s ⊆ c ∧ |s| = k − 1 do
            if s ∉ F_{k-1} then
                C_k ← C_k \ c        // Prune c from the candidates
            end if
        end for
    end for
    return C_k
end function
```

Step 1 - identifying frequent itemsets (cont.)

- Prune Example:

  $F_3 = \{\{A, B, C\}, \{A, B, D\}, \{A, C, D\}, \{A, C, E\}, \{B, C, D\}\}$

  $C_4 = \{\{A, B, C, D\}, \{A, C, D, E\}\}$

  but $\{A, C, D, E\}$ can be pruned away

  because $\{A, D, E\} \notin F_3$

- Note:
  - Prune maintains the completeness of the process

Step 2 - rule generation

- Given a frequent set $\{A, B, C, D\}$
- Which are the possible rules?
  - $\{A, B, C\} \rightarrow \{D\}$
  - $\{A, B, D\} \rightarrow \{C\}$
  - $\{A, B\} \rightarrow \{C, D\}$
- How to generate them systematically?
- How to reduce the search space?

Step 2 - rule generation (cont.)

- The rules are generated as follows:
  - generates all non-empty subsets s of each frequent itemset l
  - for each subset s computes the confidence of the rule $(l - s) \rightarrow s$
  - selects the rules whose confidence is higher than *minconf*

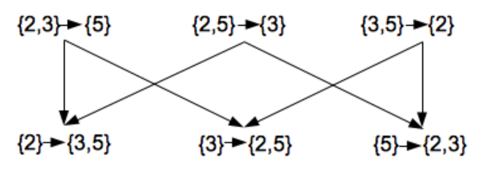
Step 2 - rule generation (cont.)

Consider again

Cliente (TID)	Itens (Item-set)
100	1, 3, 4
200	2, 3, 5,
300	1, 2, 3, 5,
400	2, 5,

and $I = \{2, 3, 5\} (= F_3)$

- Rules generated from the frequent itemset $\{2, 3, 5\}$



- Select rules $(I - a) \rightarrow a$, where $a \subseteq I$, with $minconf = 1$

$$conf((I - a) \rightarrow a) = \frac{sup(I)}{sup(I - a)}$$

Step 2 - rule generation (cont.)

- Rules with 1 consequent

$\{2,3\} \rightarrow \{5\}$ (conf= 2/2)
$\{2,5\} \rightarrow \{3\}$ (conf= 2/3) eliminated because *minconf* $= 1$
$\{3,5\} \rightarrow \{2\}$ (conf= 2/2)

- Rules with 2 consequents

$\{3\} \rightarrow \{2,5\}$ (conf= 2/3) eliminated because *minconf* $= 1$

- we don't need to worry about rules with item 3 in the consequent, because any rule obtained from $\{2,5\} \rightarrow \{3\}$ will have a *conf* $< 2/3$

> Moving items from the antecedent to the consequent never changes support and never increases confidence.

Step 2 - rule generation (cont.)

Rule generation main algorithm

```
function APRIORI-RULES(F_k)
    for f_k ∈ F_k with k ≥ 2 do
        H_1 ← {i | i ∈ f_k}      // generate candidate consequents of size 1
        AP-GENRULES(f_k, H_1)
    end for
end function
```

Step 2 - rule generation (cont.)

```
function AP-GENRULES(f_k, H_m, minconf)
    // k is the size of frequent itemset
    // m is the size of rule consequent
    for each h_m ∈ H_m do
        conf ← sup(f_k)/sup(f_k − h_m)
        if conf ≥ n × minconf then        // n is the nr. of transactions in D
            output rule (f_k − h_m) → h_m
        else
            remove h_m from H_m        // prune
        end if
    end for
    if k > m + 1 then
        H_{m+1} ← APRIORI-GEN(H_m)
        AP-GENRULES(f_k, H_{m+1}, minconf)        // next size of rule consequent
    end if
end function
```

Number of DB scans

- 1 to count frequencies of C_1
- C_2 built in memory
- 2 to count frequencies of C_1
- $\ldots$
- n to count frequencies of C_n
- Rule generation does not need to scan DB
- Number of scans is n
  - if the size of the largest frequent set is n or $n-1$

Complexity factors

- Number of items
- Number of transactions
- Minimal support
- Average size of transactions
- Number of frequent sets
- Average size of a frequent size
- Number of DB scans
    - k or $k + 1$, where k is the size of the largest frequent set

Exercises

1. Consider the following set of transactions:

$$\{\{a, b, c\}, \{a, c\}, \{b, d\}, \{b, c, d\}, \{a\}\}$$

   Using the Apriori algorithm with $minsup = 40\%$ and $minconf = 70\%$

   - find the frequent itemsets
   - find the set of relevant rules

Exercises (cont.)

2. Consider the following set of transactions:

   Using the Apriori algorithm with
   minsup $= 30\%$ and *minconf* $= 80\%$

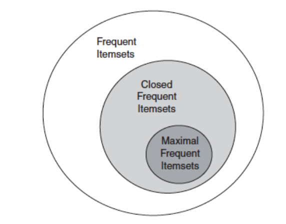
   - find the frequent itemsets
   - find the set of relevant rules

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

Compact Representation of Itemsets

- The number of frequent itemsets produced from a transaction data set can be very large.
- It is useful to identify a small representative set of itemsets from which all other frequent itemsets can be derived.

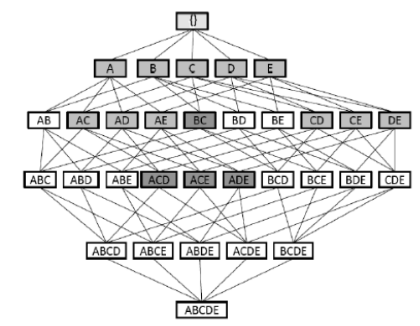
- Two such representations are:
    - maximal
    - closed



Compact Representation of Itemsets (cont.)

- s is a **maximal frequent itemset** if it is a frequent itemset for which none of its supersets is frequent.
- Example: find maximal frequent itemsets with *minsup* $= 0.3$

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



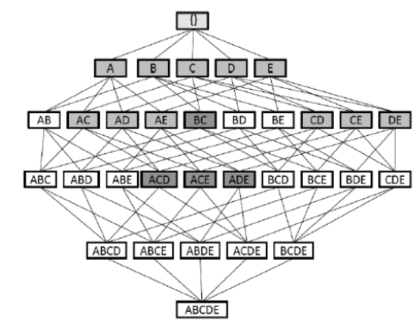
maximal frequent itemsets are:

$\{B, C\}, \{A, C, D\}, \{A, C, E\}, \{A, D, E\}$

Compact Representation of Itemsets (cont.)

- *s* is a **closed frequent itemset** if it is a frequent itemset that has no frequent supersets with the same support.
- Example: find closed frequent itemsets with *minsup* $= 0.3$

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



closed frequent itemsets are:
$\{A\}, \{C\}, \{D\}, \{E\}, \{A, C\}, \{A, D\}, \{A, E\},$
$\{B, C\}, \{C, D\}, \{C, E\}, \{A, C, D\}, \{A, C, E\}, \{A, D, E\}$

Compact Representation of Itemsets (cont.)

- From the maximal itemsets is possible to derive all frequent itemsets (not their support) by computing all non-empty intersections.
  - subsets of the maximal frequent itemset $\{A, C, D\}$ are frequent itemsets
  - $\{A\}$, $\{C\}$, $\{D\}$, $\{A, C\}$, $\{A, D\}$

- The set of all closed itemsets preserves the knowledge about the support values of all frequent itemsets.
  - $\{D, E\}$ is a non closed frequent itemset. What is its support?
  - As it is not closed, its support must be equal to one of its immediate supersets.
  - look for the most frequent closed itemset that contains $\{D, E\}$: $\{A, D, E\}$
  - $sup(\{D, E\}) = sup(\{A, D, E\})$

- There are algorithms that take advantage of this compact representation of frequent itemsets.

Too many rules ...

- The association rule algorithms tend to generate an excessive number of rules (for some problems, there can be thousands).
- Too many rules leads to model's interpretability lack.
- How can we reduce this number?
  - Changing the parameters: *minsup*, *minconf*
  - Restrictions on items: which items are relevant?
  - Summarization techniques: can we represent subsets of rules by a single representative rule?
  - Filter rules: improvement, measures of interest, ...

How to measure the improvement of a rule?

Improvement [Bayardo and Ag, 2000]

- **Improvement** of a rule is the minimum difference between its confidence and the confidence of any of its immediate simplifications.

  $$improv(A \rightarrow C) = min(\{conf(A \rightarrow C) - conf(As \rightarrow C) \mid As \subseteq A\})$$

- Example:
  - $R_1 : \{eggs, flower\} \rightarrow \{sugar\}(conf = 0.5)$
  - $R_2 : \{eggs, flower, bread\} \rightarrow \{sugar\}(conf = 0.505)$
  - $improv(R_2)$ is at most 0.005
  - with a *minimprov* of 0.01, R_2 is excluded.

Are all the rules interesting?

- Are all the discovered patterns interesting?
- In recent years, several measures have been proposed to extract interesting patterns.
- The idea is to select a subset of rules, that somehow are more relevant.
- Interesting rule (Silberschatz & Tuzhilin,95)
  - Unexpected, surprising to the user
    - Measure of interest: deviation from the expected or from the initial belief
  - Useful, actionable
    - Measure of interest: estimated benefit

How to measure the interest of a rule?

- Subjective measures: based on user's belief in the data (ex: unexpectedness, novelty, actionability, confirm hypothesis user wishes to validate)
  - These measures are hard to incorporate in the pattern discovery task.
- Objective measures: based on facts, statistics and structures of patterns (ex: support and confidence), independent of the domain considered.
  - For instance, patterns that involve mutually independent items or cover very few transactions are considered uninteresting.

How to measure the interest of a rule? (cont.)

Typically

- $A \rightarrow B$ is interesting if A and B are not statistically independent
- if A and B are statistically independent, the occurrence of A does not affect the probability of occurrence of B

$$sup(A \cup B) \approx sup(A) * sup(B)$$

$$conf(A \rightarrow B) \approx conf(\emptyset \rightarrow B)$$

- $A \rightarrow B$ may have high support and confidence and still not be interesting.
  - $\{butter\} \rightarrow \{bread\}(sup = 5\%, conf = 95\%)$
  - it is not unexpected
  - it is not useful

How to measure the interest of a rule? (cont.)

- A measure of interest should evaluate the deviation from independence.
- A rule is unexpected as it deviates from independence.
- There are different approaches to measure this deviation:
  - *lift*
  - *conviction*
  - χ^2
  - *correlation*
  - ...

Measures of Interest: limitations of support and confidence

- Assume we are interested in studying the relationship between people who drink tea and coffee.
- We summarize the preferences of 1000 people

	Coffee	$\neg$*Coffee*	
Tea	150	50	200
$\neg$*Tea*	650	150	800
	800	200	1000

- How interesting is the rule $Tea \rightarrow Coffee$?
- $sup = 150/1000 = 15\%$ and $conf = 150/200 = 75\%$
- The confidence of the rule is high, however the likelihood of a person drinking coffee regardless of drinking tea is 80%.
- Knowing that a person drinks tea actually decreases the probability of drinking coffee (from 80% to 75%).
- Thus, the rule is indeed deceitful.
- High confidence rules can be misleading.

Measures of Interest: LIFT

- **lift** is the ratio between confidence of the rule and the support of the itemset appearing in the consequent:

$$lift(A \to B) = \frac{conf(A \to B)}{sup(B)} = \frac{sup(A \cup B)}{sup(A)sup(B)}$$

- Measures the influence of A in the presence of B.
- $lift = 1$: A and B are independent ($sup(A \cup B) = sup(A)sup(B)$).
- $lift < 1$: A and B are negatively correlated.
- $lift > 1$: A and B are positively correlated.

- $lift(Tea \to Coffee) = 0.15/(0.2 * 0.8) = 0.9375$
- negative correlation between tea and coffee drinkers.

Measures of Interest: LIFT (cont.)

- The **lift** is a measure of the deviation from a rule $A \to B$ regarding the statistical independence between the antecedent A and consequent B.

- Takes values between 0 and infinity:
  - a value close to 1 indicates that A and B almost always appear together
    - the occurrence of A has no effect on the occurrence of B.
  - a value smaller than 1 indicates that A and B appear less frequently than expected together
    - the occurrence of A has a negative effect on the occurrence of B, i.e. the occurrence of A is likely to lead to the absence of B.
  - a value greater than 1 indicates that A and B appear more often together than expected
    - the occurrence of A has a positive effect on the occurrence of B, i.e. the occurrence of A increases the likelihood of occurrence of B.

Measures of Interest: Conviction

- **lift** measures co-occurrence only (not implication) and is symmetric with respect to antecedent and consequent, i.e. $lift(A \rightarrow B) = lift(B \rightarrow A)$
- **conviction** is a measure proposed to tackle some of the weaknesses of *confidence* and **lift**.
- Unlike **lift**, **conviction** is sensitive to rule direction. It indicates the departure from independence of A and B taking into account the implication direction.
- Is inspired in the logical definition of implication and attempts to measure the degree of implication of a rule.

Measures of Interest: Conviction (cont.)

- Measures the inverse of the deviation from independence of $A \cup \neg B$ because $A \rightarrow B \equiv \neg A \vee B \equiv \neg(A \wedge \neg B)$

$$conviction(A \rightarrow B) = \frac{1 - sup(B)}{1 - conf(A \rightarrow B)} = \frac{sup(A)sup(\neg B)}{sup(A \cup \neg B)}$$

- Is the inverse **lift** of the rule $R' = A \rightarrow \neg B$.
- **conviction** of a rule $A \rightarrow B$ is the ratio of the expected frequency that A occurs without B (that is to say, the frequency that the rule makes an incorrect prediction) if A and B were independent divided by the observed frequency of incorrect predictions.

Measures of Interest: Conviction (cont.)

- $conviction(A \to B) = 1$ indicates independence between A and B.
- A high value of **conviction** means that the consequent depends strongly on the antecedent.
- **conviction** increases a lot when *confidence* gets closer to 1.
- Example:
  - $sup(female) = 0.5$, $sup(mother) = 0.2$
  - $conf(mother \to female) = 1$
  - $lift(mother \to female) = 0.2/(0.2 * 0.5) = 2$
  - $conviction(mother \to female) = \infty$

Improving Apriori

- Challenges of Frequent Pattern Mining
  - Multiple scans of transaction database
  - Huge number of candidates
  - Tedious workload of support counting for candidates
- Improving Apriori: general ideas
  - Reduce number of transaction database scans
  - Shrink number of candidates (*bottleneck* of Apriori)
  - Facilitate support counting of candidates
- Some methods that improve Apriori's efficiency
  - Partitioning [Savasere et al., 1995]
  - Sampling [Toivonen, 1996]
  - Dynamic Itemset Counting [Brin et al., 1997]
  - **Frequent Pattern Projection and Growth (FP-Growth)** [Han et al., 2004]

FP-Growth

FP-Growth [Han et al., 2004] takes a different approach to discover frequent itemsets.

It proceeds in two phases:

- encodes the transaction data base into a compact structure called FP-Tree;
  - complete for frequent pattern mining and avoids costly database scans
- extracts frequent itemsets directly from this structure.
  - using a divide-and-conquer strategy and avoiding candidate generation

FP-Growth: FP-Tree representation

- FP-Tree is a compressed representation of the input data.
- It is constructed by reading one transaction at a time and mapping it onto to a path in the FP-Tree.
- As different transactions can have several items in common, paths may overlap.
- The more paths overlap, the more compression can be achieved with FP-Tree.

Ultimately, If the size of FP-Tree is small enough to fit into main memory, it will be possible to extract frequent itemsets directly from memory, without making repeated passes over the data stored in disk.

FP-Growth: FP-Tree representation (cont.)

The FP-Tree consists of:

- a root node
- a set of item prefix subtrees
- each node has
  - item name
  - counter for the transactions mapped into the given path (prefix)
  - node-link (pointer to next node with same item name)
- frequent item header table with
  - item name
  - head of node-link (pointer to first node with that name)

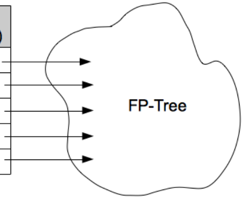
- The data is scanned once to determine the support of each item.
- Infrequent items are discarded.
- Frequent items are sorted in decreasing order of support.

Transaction Data Set

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	{a}
8	{a,b,c}
9	{a,b,d}
10	{b,c,e}

Header Table

Item	Support (minsup=2)
a	8
b	7
c	6
d	5
e	3



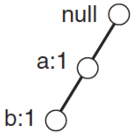
- A second pass over the data is done to construct the FP-Tree.

FP-Growth: Example 1 - FP-Tree (cont.)

- Reads TID 1: $\{a, b\}$
  - forms the path $null \rightarrow a \rightarrow b$ to encode the transaction.
  - both nodes have frequency count of 1

Transaction
Data Set

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	{a}
8	{a,b,c}
9	{a,b,d}
10	{b,c,e}

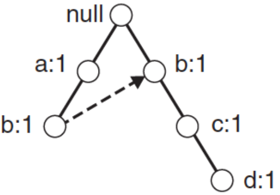


(i) After reading TID=1

- Reads TID 2: $\{b, c, d\}$
  - forms the path $null \rightarrow b \rightarrow c \rightarrow d$ to encode the transaction.
  - all nodes have frequency count of 1
  - although the first two transactions have an item in common, their paths are disjoint because they don't share a common prefix.

Transaction Data Set

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	{a}
8	{a,b,c}
9	{a,b,d}
10	{b,c,e}



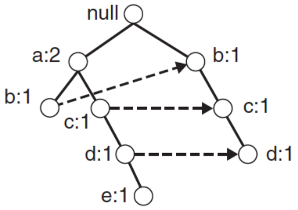
(ii) After reading TID=2

FP-Growth: Example 1 - FP-Tree (cont.)

- Reads TID 3: $\{a, c, d, e\}$
  - forms the path $null \rightarrow a \rightarrow c \rightarrow d \rightarrow e$ to encode the transaction.
  - shares a common prefix with TID 1, thus this path overlaps the path for TID 1
  - frequency count for node a is incremented to 2.
  - frequent counts for nodes c, d and e are equal to 1.

Transaction Data Set

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	{a}
8	{a,b,c}
9	{a,b,d}
10	{b,c,e}

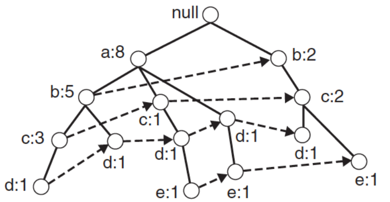


(iii) After reading TID=3

- The process continues until every transaction is mapped onto one of the paths in the FP-Tree.

Transaction Data Set

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	{a}
8	{a,b,c}
9	{a,b,d}
10	{b,c,e}



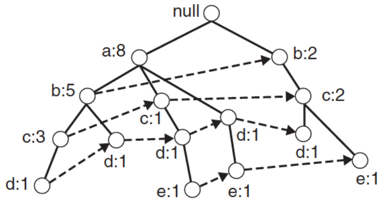
(iv) After reading TID=10

FP-Growth: Example 1 - Frequent Itemset Generation

- To have the frequent patterns ending with A we only need to consider the prefix paths of nodes with A.
- An itemset $B \cup \{A\}$ is frequent iff B is frequent in the conditional pattern base of A.
- These prefix paths can be easily accessed using the pointers associated with node of that item.

FP-Growth: Example 1 - Frequent Itemset Generation (cont.)

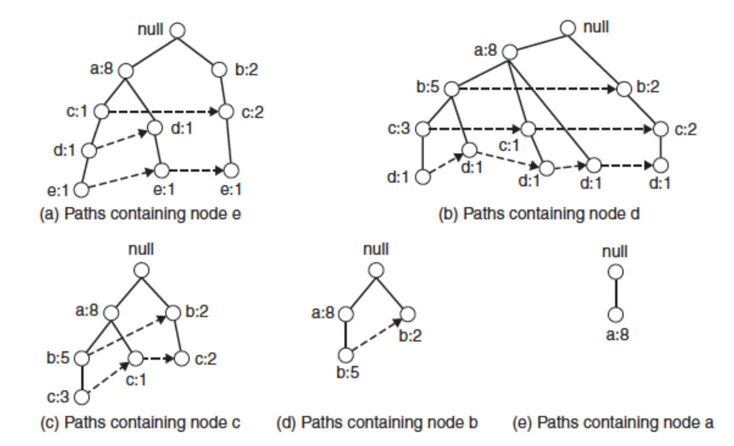
- FP-Growth explores the built FP-Tree in a bottom-up way to generate frequent itemsets.



(iv) After reading TID=10

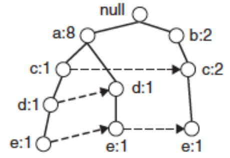
- Looks for frequent itemsets ending with *e* first, followed by *d*, *c*, *b* and *a*.

- Divides the frequent itemset generation problem into multiple subproblems.



FP-Growth: Example 1 - Frequent Itemset Generation (cont.)

- Find all frequent itemsets ending in *e*.

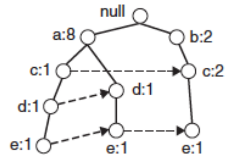


(a) Prefix paths ending in e

- The support count for *e* is the sum of support counts associated with node *e*.
- $\{e\}$ is considered a frequent itemset because its support is 3.
- As $\{e\}$ is frequent, now it has to find frequent itemsets ending in *de*, *ce*, *be* and *ae*.

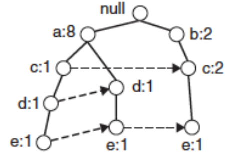
It builds a **conditional FP-tree** to find frequent itemsets ending with a particular suffix (*e*), as follows:

- **conditional FP-Tree - step 1**
  - updates the support counts of the prefix paths (some of them include transactions that do not contain item *e*)
  - $null \rightarrow b : 2 \rightarrow c : 2 \rightarrow e : 1$ includes the transaction $\{b, c\}$ that does not contain *e*.
  - is updated to $null \rightarrow b : 1 \rightarrow c : 1 \rightarrow e : 1$ to reflect the actual number of transactions that contain $\{b, c, e\}$



(a) Prefix paths ending in e

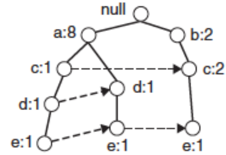
- **conditional FP-Tree - step 2**
  - removes nodes for that suffix (*e*).
  - the support counts along the prefix paths have been updated
  - finding frequent itemsets ending in *de*, *ce*, *be* and *ae* no longer need information about *e*.



(a) Prefix paths ending in e

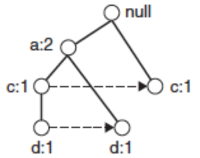
FP-Growth: Example 1 - Frequent Itemset Generation (cont.)

- **conditional FP-Tree - step 3**
  - removes items that are no longer frequent
  - node *b* appears only once and has a count of 1, i.e. there is only one transaction that contains both *b* and *e*
  - all itemsets ending with *be* must be infrequent
  - *b* can be ignored



(a) Prefix paths ending in e

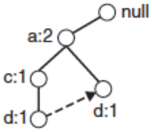
$\rightarrow$



(b) Conditional FP-tree for e

FP-Growth: Example 1 - Frequent Itemset Generation (cont.)

- FP-Growth uses the conditional FP-Tree for *e* to solve the subproblems of finding frequent itemsets ending in *de*, *ce* and *ae*
- frequent itemsets ending in *de*

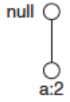


(c) Prefix paths ending in de

- adding frequency counts of node *d*, we obtain that the support count for $\{d, e\}$ is 2, so it is frequent.

FP-Growth: Example 1 - Frequent Itemset Generation (cont.)

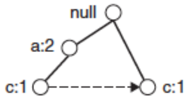
- builds the conditional FP-tree for *de*
- updates counts and removes infrequent item *c*



(d) Conditional FP-tree for de

- as it contains only one item (*a*) and its support is 2, it extracts the frequent itemset $\{a, d, e\}$
- moves on to the next problem

- finding frequent itemsets ending with *ce*
- process the prefix paths for *c*

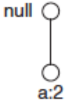


(e) Prefix paths ending in ce

- only $\{c, e\}$ is found to be frequent
- moves on to the next problem

FP-Growth: Example 1 - Frequent Itemset Generation (cont.)

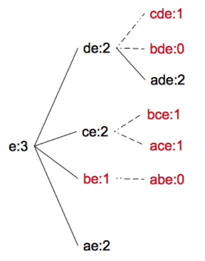
- finding frequent itemsets ending with *ae*
- process the prefix paths for *a*



(f) Prefix paths ending in ae

- only $\{a, e\}$ is found to be frequent

- frequent itemsets with suffix *e*



- at the end, the following frequent items are obtained

Suffix	Frequent Itemsets
e	{e}, {d,e}, {a,d,e}, {c,e},{a,e}
d	{d}, {c,d}, {b,c,d}, {a,c,d}, {b,d}, {a,b,d}, {a,d}
c	{c}, {b,c}, {a,b,c}, {a,c}
b	{b}, {a,b}
a	{a}

FP-Growth: Example 2 - FP-Tree

Consider the following set of transactions with *minsup* $= 3$.

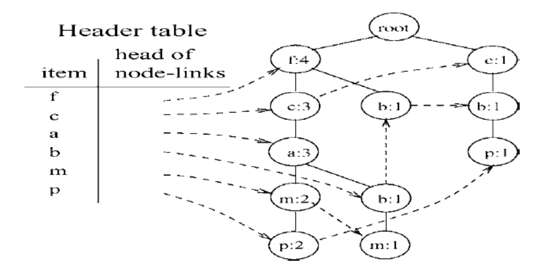
For each transaction we obtain the ordered set of frequent items.

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, b, m
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

For each frequent item, we have the following support

$f : 4, c : 4, a : 3, b : 3, m : 3, p : 3$

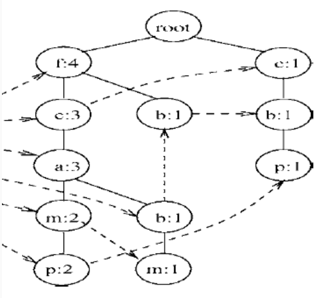
The obtained FP-Tree is



The list of ordered frequent items is $f - c - a - b - m - p$

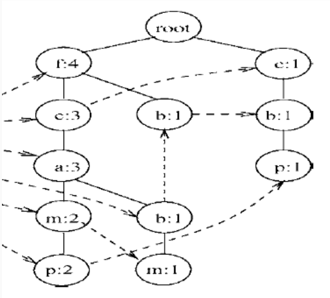
FP-Growth: Example 2 - Frequent Itemset Generation

- patterns containing *p*
- *p*'s conditional patterns
  - *fcam* : 2, *cb* : 1
- conditional FP-Tree
  - only *c* is frequent
  - $null \rightarrow c : 3$
- size 2 patterns
  - *cp* : 3
- patterns with *p*
  - *p* : 3, *cp* : 3



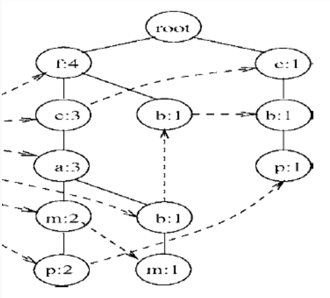
FP-Growth: Example 2 - Frequent Itemset Generation (cont.)

- patterns containing *m* (but no *p*)
- *m*'s conditional patterns
  - *fca* : 2, *fcab* : 1
- conditional FP-Tree
  - *b* is not frequent
  - $null \rightarrow f : 3 \rightarrow c : 3 \rightarrow a : 3$
- size 2 patterns
  - *am* : 3, *cm* : 3, *fm* : 3
- extend *am*
  - ...

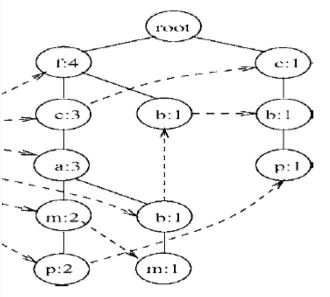


FP-Growth: Example 2 - Frequent Itemset Generation (cont.)

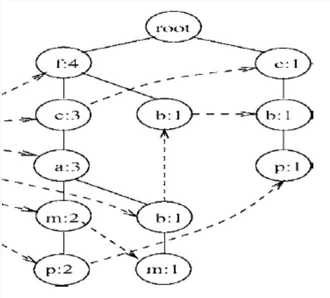
- patterns containing *am*
- FP-Tree is $null \rightarrow f:3 \rightarrow c:3$
- *am*'s conditional patterns
  - $fc:3$
- conditional FP-Tree
  - $null \rightarrow f:3 \rightarrow c:3$
- size 3 patterns
  - $cam:3$, $fam:3$
- extend *cam*
  - *fam* is not extendable



- patterns containing *cam*
- FP-Tree is *null* $\rightarrow f : 3$
- *cam*'s conditional patterns
    - $f : 3$
- conditional FP-Tree
    - *null* $\rightarrow f : 3$
- size 4 patterns
    - *fcam* : 3
- no longer patterns available



- and continues to extend patterns with
  - *cm*
  - *fm*
- and then find patterns with
  - *b*
  - *a*
  - *c*
  - *f*

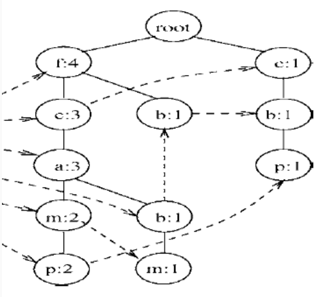


FP-Growth: Example 2 - Frequent Itemset Generation (cont.)

Item	Conditional pattern-base	Conditional FP-Tree
p	$fcam : 2, cb : 1$	$null \rightarrow c : 3$
m	$fca : 2, fcab : 1$	$null \rightarrow f : 3 \rightarrow c : 3 \rightarrow a : 3$
b	$fca : 1, f : 1, c : 1$	$\emptyset$
a	$fc : 3$	$null \rightarrow f : 3 \rightarrow c : 3$
c	$f : 3$	$null \rightarrow f : 3$
f	$\emptyset$	$\emptyset$

Frequent Itemsets ($minsup = 3$)

$$\{p : 3, cp : 3, m : 3, am : 3, cm : 3, fm : 3, cam : 3, fam : 3, fcam : 3, fcm : 3, b : 3, a : 3, fa : 3, ca : 3, fca : 3, c : 4, fc : 3, f : 4\}$$



FP-Growth: Example 2 - Frequent Itemset Generation (cont.)

Single Path FP-Tree special case:

- A FP-Tree with a single path can be mined by enumerating all the combinations of the subpaths.

  Example:
  conditional FP-Tree for *m*:

  $$null \rightarrow\rightarrow f:3 \rightarrow c:3 \rightarrow a:3$$

  leads to the frequent patterns
  $\{m:3, fm:3, cm:3, am:3, fcm:3, fam:3, cam:3, fcam:3\}$

- the count of each subpath is the minimum of the nodes in it.

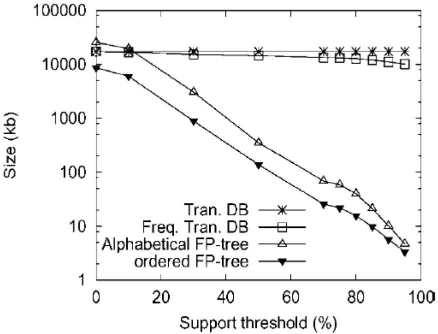
Analysis

It can be shown that FP-Growth:

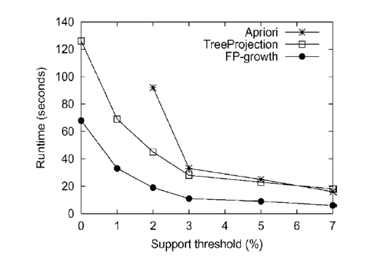
- finds the complete set of frequent patterns in the given transaction DB;
- the FP-Tree is usually much smaller than DB.
- it scans the FP-Tree of DB once foreach frequent item A
  - generates a small pattern involving A
  - pattern mining is done recursively on that small pattern
- mining operation consists of:
  - prefix counts adjustment
  - counting local frequent items
  - pattern fragment concatenation

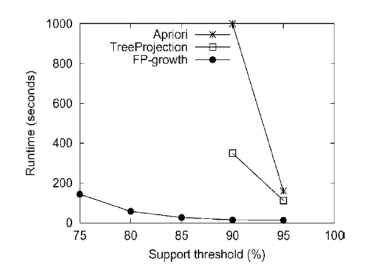
Analysis (cont.)

- Assess compactness of FP-Tree
  - by measuring their size on Connect-4 dataset



- Scalability study
  - by measuring time of competing algorithms on a sparse artificial dataset and (dense) Connect-4 dataset





Exercises

1. Consider the following set of transactions:

$$\{\{a, b, c, d\}, \{a, b, c, d\}, \{a, e\}, \{a, b\}\}$$

   Assuming that *minsup* $= 50\%$, how much space is needed for

   - Apriori (number of candidate sets) ?
   - FP-growth (size of tree and size of header table) ?

Exercises (cont.)

2. Consider the following set of transactions:

$$\{\{a, d, e, f\}, \{b, d, e, f\}, \{c, d, e, f\},$$
$$\{a\}, \{a\}, \{a\}, \{b\}, \{b\}, \{b\}, \{c\}, \{c\}, \{c\}\}$$

   Build the FP-Tree with decreasing frequency order.

Exercises (cont.)

3. Consider the following set of transactions:

TID	Itemset
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

Using the FP-growth algorithm with *minsup* $= 20\%$

- find the frequent itemsets

Association Rules: Conclusions

- GOAL: Finding associations
- Association rule mining:
  - Frequent itemsets (requires min support)
  - Association rules (requires min confidence)
    - Probabilistic implications
- One of the most used data mining tools
  - Problem: generates too much rules
  - Pattern compression and pattern selection
- Several algorithms:
  - Apriori is the most known algorithm
  - There are variants of Apriori that return exactly the same patterns!
  - Completeness: find all rules.

Advanced Topics: Class Association Rules (CAR)

- Normal association rule mining does not have any target.
- It finds all possible rules that exist in data, i.e., any item can appear as a consequent or a condition of a rule.
- However, in some applications, the user is interested in some targets.
  - E.g, the user has a set of text documents from some known topics. He/she wants to find out what words are associated or correlated with each topic.

Advanced Topics: Class Association Rules (CAR) (cont.)

- Let T be a transaction data set consisting of n transactions.
- Each transaction is also labeled with a class y.
- Let I be the set of all items in T, Y be the set of all class labels and $I \cap Y = \emptyset$.
- A class association rule (CAR) is an implication of the form $X \rightarrow y$, where $X \subseteq I$, and $y \in Y$.
- The definitions of support and confidence are the same as those for normal association rules.

Advanced Topics: Class Association Rules (CAR) (cont.)

- Unlike normal association rules, CARs can be mined directly in one step.
- The key operation is to find all ruleitems that have support above *minsup*. A ruleitem is of the form: $(condset, y)$ where *condset* is a set of items from I (i.e., $condset \subseteq I$), and $y \in Y$ is a class label.
- Each ruleitem basically represents a rule: $condset \rightarrow y$,
- The Apriori algorithm can be modified to generate CARs

Advanced Topics: Class Association Rules (CAR) (cont.)

- The user can specify different minimum supports to different classes, which effectively assign a different minimum support to rules of each class.
- For example, we have a data set with two classes, Yes and No. We may want
  - rules of class Yes to have the minimum support of 5% and
  - rules of class No to have the minimum support of 10%.
- By setting minimum class supports to 100% (or more for some classes), we tell the algorithm not to generate rules of those classes.
- This is a very useful trick in applications.

References

References

Aggarwal, C. C. (2015).
Data Mining, The Texbook.
Springer.

Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., and Verkamo, A. I. (1996).
Fast discovery of association rules.
In *Advances in Knowledge Discovery and Data Mining*, pages 307–328. American Association for Artificial Intelligence.

Agrawal, R. and Srikant, R. (1994).
Fast algorithms for mining association rules in large databases.
In *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB '94, pages 487–499. Morgan Kaufmann Publishers Inc.

Brin, S., Motwani, R., Ullman, J. D., and Tsur, S. (1997).
Dynamic itemset counting and implication rules for market basket data.
In *Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data*, volume 26, pages 255–264. ACM.

Domingo, C., Gavalda, R., and Watanabe, O. (1998).
On-line sampling methods for discovering association rules.

References (cont.)

Gama, J. (2016).
Association rules.
Slides.

Gama, J., Oliveira, M., Lorena, A. C., Faceli, K., and de Leon Carvalho, A. P. (2015).
***Extração de Conhecimento de Dados - Data Mining*.**
Edições Sílabo, 2nd edition.

Han, J., Kamber, M., and Pei, J. (2011).
***Data Mining: Concepts and Techniques*.**
Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 3rd edition.

Han, J., Pei, J., Yin, Y., and Mao, R. (2004).
Mining frequent patterns without candidate generation: A frequent-pattern tree approach.
Data Mining and Knowledge Discovery, 8(1):53–87.

Jorge, A. (2016).
Association rules.
Slides.

Liu, B. (2011).
***Web Data Mining. Exploring Hyperlinks, Contents, and Usage Data*.**
Springer, 2nd edition.

References (cont.)

Savasere, A., Omiecinski, E., and Navathe, S. B. (1995).
An efficient algorithm for mining association rules in large databases.
In *Proceedings of the 21th International Conference on Very Large Data Bases*, VLDB '95, pages 432–444. Morgan Kaufmann Publishers Inc.

Tan, P.-N., Steinbach, M., and Kumar, V. (2005).
***Introduction to Data Mining*.**
Addison Wesley.

Toivonen, H. (1996).
Sampling large databases for association rules.
In *Proceedings of the 22th International Conference on Very Large Data Bases*, VLDB '96, pages 134–145. Morgan Kaufmann Publishers Inc.

Torgo, L. (2017).
***Data Mining with R: Learning with Case Studies*.**
Chapman and Hall/CRC, 2nd edition.