Foundations of Knowledge Management:  
Association Rule Mining

(LV: 707.009)

02.11.2010

Mark Kröll  
mkroell@tugraz.at  
Graz University of Technology, Austria
Today’s Outline

- Association Rule Mining
  - Motivating Example
  - Definitions
  - The Apriori Algorithm
  - Limitations / Improvements
  - Supplemental Material

- Acknowledgements / slides based on:
  - Lecture „Data Management and Exploration“ by Thomas Seidl (RWTH Aachen)
  - Lecture “Association Rules” by Berlin Chen
  - Lecture “PG 402 Wissensmanagement” by Z. Jerroudi
  - Lecture “LS 8 Informatik Computergestützte Statistik“ by Morik and Weihs
  - Association Rules by Prof. Tom Fomby
  - Lecture Association Rules: Advanced Topics / Politecnico Di Milano
Today we learn

- Why Association Rules are useful?
  - history + motivation
- What Association Rules are?
  - definitions (formal / informal)
- How we can mine them?
  - the Apriori algorithm
  - Illustrating example
- Which challenges they face?
  - + means to address them
Why do we need association rule mining?
Motivation for Association Rules (1)

Hmmm, which items are frequently purchased together by my customers?

Association Rule Mining can help to better understand purchase behavior!!
Market Basket Analysis (MBA)(1)

- In retailing, *most purchases are bought on impulse*. Market basket analysis gives clues as to what a customer might have bought *if the idea had occurred to them*.
  - decide the location and promotion of goods inside a store.

Observation: Purchasers of Barbie dolls are more likely to buy candy.

{barbie doll} => {candy}

- place high-margin candy near to the Barbie doll display.

Create Temptation: Customers who would have bought candy with their Barbie dolls *had they thought of it will now be suitably tempted*.

(as it is constantly done in TV commercials)
Market Basket Analysis (MBA)(2)

- Further possibilities:
  - comparing results between different stores, between customers in different demographic groups, between different days of the week, different seasons of the year, etc.
  - If we observe that a rule holds in one store, but not in any other then we know that there is something interesting about that store.
    - different clientele
    - different organization of its displays (in a more lucrative way …)
  → investigating such differences may yield useful insights which will improve company sales.
Objective of Association Rule Mining

- find **associations** and **correlations** between different items (products) that customers place in their shopping basket.

- yet, it also aims to **discover non-trivial, interesting** associations
  - Chips + Beer → we could have guessed that
  - Diapers + Beer → is kind of surprising thus interesting

- to better **predict**, e.g., :
  1. what my customers buy? (→ spectrum of products)
  2. when they buy it? (→ advertising)
  3. which products are bought together? (→ placement)
Today we learn

- Why Association Rules are useful?
  - history + motivation

- What Association Rules are?
  - definitions (formal / informal)

- How we can mine them?
  - the Apriori algorithm
  - Illustrating example

- Which challenges they face?
  - + means to address them
Introduction into AR

- Formalizing the problem a little bit
  - Transaction Database $T$: a set of transactions $T = \{t_1, t_2, \ldots, t_n\}$
  - Each transaction contains a set of items (item set)
  - An item set is a collection of items $I = \{i_1, i_2, \ldots, i_m\}$

- General Objective:
  - Find frequent/interesting patterns, associations, correlations, or causal structures among sets of items or elements in databases or other information repositories.
  - Put these relationships in terms of association rules $X \Rightarrow Y$
    - where $X$, $Y$ represent two itemsets
Examples of AR

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>bread, jelly, peanut-butter</td>
</tr>
<tr>
<td>T2</td>
<td>bread, peanut-butter</td>
</tr>
<tr>
<td>T3</td>
<td>bread, milk, peanut-butter</td>
</tr>
<tr>
<td>T4</td>
<td>beer, bread</td>
</tr>
<tr>
<td>T5</td>
<td>beer, milk</td>
</tr>
</tbody>
</table>

- **Frequent Item Sets:**
  - Items that appear frequently together
    - \( I = \{\text{bread, peanut-butter}\} \)
    - \( I = \{\text{beer, bread}\} \)

**Examples:**

- \( \text{bread} \Rightarrow \text{peanut-butter} \)
- \( \text{beer} \Rightarrow \text{bread} \)

**Reads as:**
If you buy bread, then you will buy peanut-butter as well. (at least there is a high chance)
What is an interesting item set?

- **Support Count ($\sigma$)**
  - Frequency of occurrence of an itemset
    
    \[ \sigma \{\text{bread, peanut-butter}\} = 3 \]
    
    \[ \sigma \{\text{beer, bread}\} = 1 \]

- **Support ($s$)**
  - Fraction of transactions that contain an itemset
    
    \[ s\{\text{bread, peanut-butter}\} = \frac{3}{5} (0.6) \]
    
    \[ s \{\text{beer, bread}\} = \frac{1}{5} (0.2) \]

- **Frequent Itemset**
  - = an itemset whose support is greater than or equal to a minimum support threshold ($\text{mins}up$)

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>bread, jelly, peanut-butter</td>
</tr>
<tr>
<td>T2</td>
<td>bread, peanut-butter</td>
</tr>
<tr>
<td>T3</td>
<td>bread, milk, peanut-butter</td>
</tr>
<tr>
<td>T4</td>
<td>beer, bread</td>
</tr>
<tr>
<td>T5</td>
<td>beer, milk</td>
</tr>
</tbody>
</table>
What is an interesting rule?

- An association rule is an implication of two itemsets
  \[ X \Rightarrow Y \]

- Most common measures:
  - **Support (s)**
    - The occurring frequency of the rule, i.e., the number of transactions that contain both \( X \) and \( Y \)
  - **Confidence (c)**
    - i.e., measures the number of how often items in \( Y \) appear in transactions that contain \( X \) vs. the number of how often items in \( X \) occur in general

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>bread, jelly, peanut-butter</td>
</tr>
<tr>
<td>T2</td>
<td>bread, peanut-butter</td>
</tr>
<tr>
<td>T3</td>
<td>bread, milk, peanut-butter</td>
</tr>
<tr>
<td>T4</td>
<td>beer, bread</td>
</tr>
<tr>
<td>T5</td>
<td>beer, milk</td>
</tr>
</tbody>
</table>

\[
s = \frac{\sigma(X \cup Y)}{\# \text{ of trans.}}
\]

\[
c = \frac{\sigma(X \cup Y)}{\sigma(X)}
\]
Interestingness of Rules

Let's have a look at some associations + the corresponding measures

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
<th>s</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>bread, jelly, peanut-butter</td>
<td>0.60</td>
<td>0.75</td>
</tr>
<tr>
<td>T2</td>
<td>bread, peanut-butter</td>
<td>0.60</td>
<td>1.00</td>
</tr>
<tr>
<td>T3</td>
<td>bread, milk, peanut-butter</td>
<td>0.20</td>
<td>0.50</td>
</tr>
<tr>
<td>T4</td>
<td>beer, bread</td>
<td>0.20</td>
<td>0.33</td>
</tr>
<tr>
<td>T5</td>
<td>beer, milk</td>
<td>0.20</td>
<td>1.00</td>
</tr>
</tbody>
</table>

Let $s$ be the support of a rule $X \Rightarrow Y$ defined as

$$s = \frac{\sigma(X \cup Y)}{\#\text{of trans.}}$$

Let $c$ be the confidence of a rule $X \Rightarrow Y$ defined as

$$c = \frac{\sigma(X \cup Y)}{\sigma(X)}$$
Support vs. Confidence

$S = \frac{\sigma(X \cup Y)}{\# \text{ of trans.}}$

$C = \frac{\sigma(X \cup Y)}{\sigma(X)}$

- Support is symmetric
- Confidence is asymmetric

<table>
<thead>
<tr>
<th>Rule</th>
<th>Support</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>bread $\Rightarrow$ peanut-butter</td>
<td>0.60</td>
<td>0.75</td>
</tr>
<tr>
<td>peanut-butter $\Rightarrow$ bread</td>
<td>0.60</td>
<td>1.00</td>
</tr>
</tbody>
</table>

- Support + Confidence represent different perspectives

<table>
<thead>
<tr>
<th>Rule</th>
<th>Support</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>peanut-butter $\Rightarrow$ jelly</td>
<td>0.20</td>
<td>0.33</td>
</tr>
<tr>
<td>jelly $\Rightarrow$ peanut-butter</td>
<td>0.20</td>
<td>1.00</td>
</tr>
</tbody>
</table>

- Confidence increases if the item occurs exclusively in the corresponding rule.
- Support increases if the items in the rule occur frequently together.
Confidence vs. Conditional Probability

- Recap Confidence ($c$)

\[
\frac{\sigma(X \cup Y)}{\sigma(X)}
\]

= (number of transactions containing all of the items in X and Y) / (number of transactions containing the items in X)

= (support count of X and Y) / (support count of X)

= conditional probability $Pr(Y | X) = \frac{Pr(X \text{ and } Y)}{Pr(X)}$

If X is bought then Y will be bought with a given probability

→ “If jelly is bought then peanut-butter will be bought with a probability of 100%”

(based on the data at hand → transaction database containing 5 entries)
Today we learn

- Why Association Rules are useful?
  - history + motivation
- What Association Rules are?
  - definitions (formal / informal)
- How we can mine them?
  - the Apriori algorithm
  - Illustrating example
- Which challenges they face?
  - + means to address them
Apriori

- Is the most influential AR miner

- It consists of two steps
  1. Generate all frequent itemsets using support $\geq$ minsup
  2. Use frequent itemsets to craft association rules using confidence $\geq$ minconf

- Lets have a look at step one first: Generating Itemsets
Candidate Sets with 5 Items

Graph showing candidate sets with 5 items, from null to ABCDE.
Exponential Complexity

- Given $d$ unique items:
  - Total number of itemsets $= 2^d$
  - Total number of possible association rules $= 3^d - 2^{d+1} + 1$

  $\Rightarrow$ for $d = 5$, there are 32 candidate item sets
  $\Rightarrow$ for $d = 5$, there are 180 rules
Generating Itemsets …

- Brute force approach is computationally expensive
  - = take all possible combinations of items
  - let’s select candidates in a smarter way

- Key idea: Downward closure property
  - any subset of a frequent itemset are also frequent itemsets

→ The algorithm iteratively does:
  - Create itemsets
  - yet, continue exploring only those whose support >= minsup
Example Itemset Generation

- discard infrequent itemsets
  - At the first level B does not meet the required support $\geq$ minsup criterion
  - All potential itemsets that contain B can be discarded (32 → 16)
Let's have a *Frequent Itemset* Example:

Minimum support count = 3

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>bread, jelly, peanut-butter</td>
</tr>
<tr>
<td>T2</td>
<td>bread, peanut-butter</td>
</tr>
<tr>
<td>T3</td>
<td>bread, milk, peanut-butter</td>
</tr>
<tr>
<td>T4</td>
<td>beer, bread</td>
</tr>
<tr>
<td>T5</td>
<td>beer, milk</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Item</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>bread</td>
<td>4</td>
</tr>
<tr>
<td>peanut-b</td>
<td>3</td>
</tr>
<tr>
<td>jelly</td>
<td>1</td>
</tr>
<tr>
<td>milk</td>
<td>2</td>
</tr>
<tr>
<td>beer</td>
<td>2</td>
</tr>
</tbody>
</table>

Frequent Item Sets for min. support count = 3:

\{bread\}, \{peanut-b\} and \{bread, peanut-b\}
@Mining Association Rules

- given the itemset \{bread, peanut-b\} (see last slide)

- corresponding Association Rules:
  - bread → peanut-b. \[support = 0.6, confidence = 0.75\]
  - peanut-b. → bread \[support = 0.6, confidence = 1.0\]

- The above rules are binary partitions of the same itemset

- Recall: Rules originating from the same itemset have identical support but can have different confidences

- Support and confidence are decoupled:
  - Support used during candidate generation
  - Confidence used during rule generation
Example of Apriori Run (1)

**Database TDB**

<table>
<thead>
<tr>
<th>Tid</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>A, C, D</td>
</tr>
<tr>
<td>20</td>
<td>B, C, E</td>
</tr>
<tr>
<td>30</td>
<td>A, B, C, E</td>
</tr>
<tr>
<td>40</td>
<td>B, E</td>
</tr>
</tbody>
</table>

**Itemset**

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A}</td>
<td>2</td>
</tr>
<tr>
<td>{B}</td>
<td>3</td>
</tr>
<tr>
<td>{C}</td>
<td>3</td>
</tr>
<tr>
<td>{D}</td>
<td>1</td>
</tr>
<tr>
<td>{E}</td>
<td>3</td>
</tr>
</tbody>
</table>

**L₁**

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A}</td>
<td>2</td>
</tr>
<tr>
<td>{B}</td>
<td>3</td>
</tr>
<tr>
<td>{C}</td>
<td>3</td>
</tr>
<tr>
<td>{E}</td>
<td>3</td>
</tr>
</tbody>
</table>

**L₂**

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A, C}</td>
<td>2</td>
</tr>
<tr>
<td>{B, C}</td>
<td>3</td>
</tr>
<tr>
<td>{B, E}</td>
<td>3</td>
</tr>
<tr>
<td>{C, E}</td>
<td>3</td>
</tr>
</tbody>
</table>

**Itemset**

<table>
<thead>
<tr>
<th>Itemset</th>
<th>sup</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A, B}</td>
<td>1</td>
</tr>
<tr>
<td>{A, C}</td>
<td>2</td>
</tr>
<tr>
<td>{A, E}</td>
<td>1</td>
</tr>
<tr>
<td>{B, C}</td>
<td>2</td>
</tr>
<tr>
<td>{B, E}</td>
<td>3</td>
</tr>
<tr>
<td>{C, E}</td>
<td>2</td>
</tr>
</tbody>
</table>

**C₁**

**C₂**

**minsup = 2**

**1st scan**

**2nd scan**

**C₂**

<table>
<thead>
<tr>
<th>Itemset</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A, B}</td>
</tr>
<tr>
<td>{A, C}</td>
</tr>
<tr>
<td>{A, E}</td>
</tr>
<tr>
<td>{B, C}</td>
</tr>
<tr>
<td>{B, E}</td>
</tr>
<tr>
<td>{C, E}</td>
</tr>
</tbody>
</table>
Example of Apriori Run(2)

Why not e.g. \{A, B, C\}?

→ only \{A, C\} and \{B, C\} are frequent 2-item sets
  \{A, B\} is not
Apriori – The Second Step

- At this stage, we have all frequent itemsets

\[
\begin{array}{|c|c|}
\hline
\text{Itemset} & \text{sup} \\
\hline
\{A\} & 2 \\
\{B\} & 3 \\
\{C\} & 3 \\
\{E\} & 3 \\
\hline
\end{array}
\]

+ \[
\begin{array}{|c|c|}
\hline
\text{Itemset} & \text{sup} \\
\hline
\{A, C\} & 2 \\
\{B, C\} & 2 \\
\{B, E\} & 3 \\
\{C, E\} & 2 \\
\hline
\end{array}
\]

\[
\begin{array}{|c|c|}
\hline
\text{Itemset} & \text{sup} \\
\hline
\{B, C, E\} & 2 \\
\hline
\end{array}
\]

→ now we use these itemsets to generate association rules
The Apriori – Algorithm(1)

- Let $k = 1$, set $\text{min\_support}$
- Generate frequent itemsets of size 1
- Repeat until no new frequent itemsets are found
  - generate candidate itemsets of size $k+1$ from size $k$ frequent itemsets
  - prune candidate itemsets containing subsets of size $k$ that are infrequent
  - compute the support of each candidate by scanning the transaction DB
  - eliminate candidates that are infrequent, leaving only those that are frequent
The Apriori – Algorithm(2)

\( C_k \): 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 \( C_{k+1} \) with min_support
end
return \( \bigcup_k L_k \)
The Apriori – Algorithm(3)

- Join Step
  - $C_k$ is generated by joining $L_{k-1}$ with itself

- Prune Step
  - Any $(k-1)$ itemset that is not frequent cannot be a subset of a frequent $k$-itemset
Rule Generation in Apriori

- given a frequent itemset \( L \)
  - Find all non-empty subsets \( F \) in \( L \), such that the association rule \( F \Rightarrow \{L-F\} \) satisfies the minimum confidence
  - Create rule \( F \Rightarrow \{L-F\} \)

- if \( L = \{A, B, C\} \)
  - The candidate association rules are:
    - \( AB \Rightarrow C \)
    - \( BC \Rightarrow A \)
    - \( AC \Rightarrow B \)
    - \( C \Rightarrow AB \)
    - \( A \Rightarrow BC \)
    - \( B \Rightarrow AC \)

- In general, there are \( 2^k - 2 \) candidate solutions,
  - where \( k \) is the size of itemset \( L \)
Can we be more efficient?

- can we apply the same heuristic used with itemset generation?
  - Problem: Confidence per se does not have anti-monotone property
  - That is, $c(AB => D) > c(A => D)$?
    - We don’t know …

- but confidence of rules generated from the same itemset does have the anti monotone property
  - $L = \{A, B, C, D\}$
    - $c(ABC => D) \geq c(AB => CD) \geq c(A => BCD)$

  - We can use this property to inform the rule generation
Example of Efficient Rule Generation

\[ c(ABC \Rightarrow D) \geq c(AB \Rightarrow CD) \geq \ldots \]
Apriori / Summary

- Is the most influential AR miner

- It consists of two steps
  1. Generate all frequent itemsets whose *support* $\geq$ *minsup*
  2. Use frequent itemsets to craft association rules (confidence $\geq$ *minconf*)

- Outputs a list of association rules
@Quality of Generated Rules

- Apriori algorithm produces a lot of rules
  - many of them redundant
  - many of them uninteresting
  - many of them uninterpretable

- Strong Rules can be misleading
  - strong = high support and/or high confidence
  - yet, not all strong association rules are interesting enough to be presented and used (see next slide for an example)

→ If a rule is not interpretable or intuitive in the face of domain-specific knowledge, it need not be adopted and used for decision-making purposes.
Strong Rules Are Not Necessarily Interesting (1)

- Example from [Aggarwal & Yu, PODS98]
  - among 5000 students
  - 3000 play basketball (=60%), 3750 eat cereal (=75%),
    2000 both play basketball and eat cereal (=40%)
  - minsup (40%) and minconf (60%)

- Rule play basketball ⇒ eat cereal?
  - [s= 40%, c = 66.7%] ⇒ beyond thresholds …

- Yet, it is misleading because the overall percentage of students eating cereal is 75% which is higher than 66.7%

\[
P(\text{eat cereal}) > P(\text{eat cereal | play basketball})
\]

\[
0.75 > 0.66
\]

⇒ condition of (“play basketball”) leads to a decrease
⇒ negative association (“playing basketball” decreases “eat cereal”)

Professor Horst Cerjak, 19.12.2005
Strong Rules Are Not Necessarily Interesting (2)

- statistical (linear) independence test (e.g. correlation)
  - Heuristics to measure association
    \[ A \Rightarrow B \text{ is interesting if} \]
    \[ \text{support}(A, B) - [\text{support}(A) \cdot \text{support}(B)] > k \]

- example: the association rule in the previous example
  - support(play basketball, eat cereal) -
    [support(play basketball) \cdot support(eat cereal)]
  - = 0.4 - [0.6 \cdot 0.75]
  - = -0.05 < 0 (negatively associated !)
Today we learn

- Why Association Rules are useful?
  - history + motivation
- What Association Rules are?
  - definitions (formal / informal)
- How we can mine them?
  - the Apriori algorithm
  - Illustrating example
- Which challenges they face?
  - + means to address them
Limitations of Apriori

- **Bottlenecks:**
  - Apriori scans the transaction DB several times
  - usually, there is a large number of candidates
  - calculation of candidate’s *support count* can be time-consuming

- **Improvements:**
  - reduce the number of DB scans
  - shrink the number of candidates
  - more efficient support counting for candidates
Revisiting Candidate Generation

- **Remember**
  - Use the previous frequent itemsets \((k-1)\) to generate the \(k\)-itemsets
  - Count itemsets support by iteratively scanning the database

- **Bottleneck in the process: Candidate Generation**
  - Suppose 100 items
  - First level of the tree \(\rightarrow\) 100 nodes
  
  - Second level of the tree \(\rightarrow\) \(\binom{100}{2}\)
  
  - In general, number of \(k\)-itemsets: \(\binom{100}{k}\)
Avoid Candidate Generation
(Jiawei Han, Jian Pei, Yiwen Yin: Mining Frequent Patterns without Candidate Generation)

- build auxiliary structure: Frequent Pattern (FP) Tree
  - to get statistics about the itemsets to avoid candidate generation
  - to avoid multiple scans of the data
  - frequent itemsets are directly extracted from the FP-tree

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>{A,B}</td>
</tr>
<tr>
<td>2</td>
<td>{B,C,D}</td>
</tr>
<tr>
<td>3</td>
<td>{A,C,D,E}</td>
</tr>
<tr>
<td>4</td>
<td>{A,D,E}</td>
</tr>
<tr>
<td>5</td>
<td>{A,B,C}</td>
</tr>
<tr>
<td>6</td>
<td>{A,B,C,D}</td>
</tr>
<tr>
<td>7</td>
<td>{B,C}</td>
</tr>
<tr>
<td>8</td>
<td>{A,B,C}</td>
</tr>
<tr>
<td>9</td>
<td>{A,B,D}</td>
</tr>
<tr>
<td>10</td>
<td>{B,C,E}</td>
</tr>
</tbody>
</table>
Building the FP Tree (1)

- First scan of the database to (a) eliminate infrequent items and (b) sort the items according to their support

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>{A,B}</td>
</tr>
<tr>
<td>2</td>
<td>{B,C,D}</td>
</tr>
<tr>
<td>3</td>
<td>{A,C,D,E}</td>
</tr>
<tr>
<td>4</td>
<td>{A,D,E}</td>
</tr>
<tr>
<td>5</td>
<td>{A,B,C}</td>
</tr>
<tr>
<td>6</td>
<td>{A,B,C,D}</td>
</tr>
<tr>
<td>7</td>
<td>{B,C}</td>
</tr>
<tr>
<td>8</td>
<td>{A,B,C}</td>
</tr>
<tr>
<td>9</td>
<td>{A,B,D}</td>
</tr>
<tr>
<td>10</td>
<td>{B,C,E}</td>
</tr>
</tbody>
</table>

After reading TID=1:
Building the FP Tree (2)

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>{A,B}</td>
</tr>
<tr>
<td>2</td>
<td>{B,C,D}</td>
</tr>
<tr>
<td>3</td>
<td>{A,C,D,E}</td>
</tr>
<tr>
<td>4</td>
<td>{A,D,E}</td>
</tr>
<tr>
<td>5</td>
<td>{A,B,C}</td>
</tr>
<tr>
<td>6</td>
<td>{A,B,C,D}</td>
</tr>
<tr>
<td>7</td>
<td>{B,C}</td>
</tr>
<tr>
<td>8</td>
<td>{A,B,C}</td>
</tr>
<tr>
<td>9</td>
<td>{A,B,D}</td>
</tr>
<tr>
<td>10</td>
<td>{B,C,E}</td>
</tr>
</tbody>
</table>

After reading TID=2:

null

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

Slides from Lecture Association Rules:
Advanced Topics / Politecnico Di Milano
Building the FP Tree (3)

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>{A,B}</td>
</tr>
<tr>
<td>2</td>
<td>{B,C,D}</td>
</tr>
<tr>
<td>3</td>
<td>{A,C,D,E}</td>
</tr>
<tr>
<td>4</td>
<td>{A,D,E}</td>
</tr>
<tr>
<td>5</td>
<td>{A,B,C}</td>
</tr>
<tr>
<td>6</td>
<td>{A,B,C,D}</td>
</tr>
<tr>
<td>7</td>
<td>{B,C}</td>
</tr>
<tr>
<td>8</td>
<td>{A,B,C}</td>
</tr>
<tr>
<td>9</td>
<td>{A,B,D}</td>
</tr>
<tr>
<td>10</td>
<td>{B,C,E}</td>
</tr>
</tbody>
</table>

After reading TID=3:

```
null
A:2
B:1
C:1
D:1
E:1
```

Slides from Lecture Association Rules:
Advanced Topics / Politecnico Di Milano
Points are used to assist frequent itemset generation

The FP-tree is typically smaller than the data
Mining the FP-Tree (1)

- Start from the bottom, from E
- Compute the support count, by adding the counts associated to E
- Since E is frequent we compute the frequent itemsets from the FP-tree

<table>
<thead>
<tr>
<th>Item</th>
<th>Pointer</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
</tr>
</tbody>
</table>

minsupt=2
Mining the FP-Tree (2)

- Consider the path to E, it is frequent
- Extract the conditional pattern base

\[
\text{minsup}=2
\]

\[
\{(ACD:1),(AD:1),(BC:1)\}
\]
Mining the FP-Tree (3)

- From the conditional pattern base
  \{(ACD:1),(AD:1),(BC:1)\}
  build the conditional FP-tree
- Delete unfrequent items

\[
\begin{align*}
A & : 2 \\
B & : 1 \\
C & : 1 \\
D & : 1 \\
\end{align*}
\]

\[
\begin{align*}
A & : 2 \\
C & : 1 \\
D & : 1 \\
\end{align*}
\]
Use the conditional FP-tree to extract the frequent itemsets ending with DE, CE, and AE

Conditional FP-tree for E

Prefix paths ending in DE

\text{minsupt}=2
Mining the FP-Tree (5)

- Consider the suffix DE, it is frequent
- Build the conditional FP-tree for DE
- The last tree contains only A which is frequent
- So far we have obtained three frequent itemsets, \{E, DE, ADE\}

Prefix paths ending in de

Conditional FP-tree for de

C is unfrequent so it is deleted

\texttt{minsuf=2}
Mining the FP-Tree (6)

- After **DE**, the suffix **CE** is considered
- **CE** is frequent and thus added to the frequent itemsets
- Then, we search for the itemsets ending with **AE**
- At the end the frequent itemsets ending with **E** are \{**E**, **DE**, **ADE**, **CE**, **AE**\}

Prefix paths ending in **CE**

Prefix paths ending in **AE**

\[ \text{minsup}=2 \]
Mining Frequent Patterns using FP-Trees

- **General idea (divide-and-conquer)**
  - Recursively grow frequent pattern path using the FP-tree

- **Method**
  - for each item, construct its conditional pattern-base, and then its conditional FP-tree
  - repeat the process for each newly created conditional FP-tree
  - until the resulting FP-tree is empty, or it contains only one path (single path will generate all the combinations of its sub-paths, each of which is a frequent pattern)
- FP-growth is about a magnitude faster than Apriori because of:
  - no candidate generation and testing
  - more compact data structure
  - no iterative database scans
Summary of Today’s Lecture (1)

- **Association Rules** attempt to capture associations between groups of items
  - Interesting / surprising associations

- **Association Rules** are “if-then rules” with two measures
  - support and confidence of the rule

- **Association rule mining** is also known as
  - frequent item set mining
  - market basket analysis
  - affinity analysis
Summary of Today’s Lecture (2)

- **Apriori is most influential rule miner**
  - Consisting of two steps:
    1) Generating Frequent Itemsets
    2) Generating Association Rules from these sets

- **Challenges/ Improvements**
  - exponential runtime / efficient data structures (FP – tree)
  - rule quality / metrics: interestingness of rules

- **Further Directions:**
  - Application to sequences in order to look for patterns that evolve over time
  - Hierarchical Association Rules
  - Quantitative Association Rules
Types of Association Rules

- **Binary Association Rules**
  - Bread => Peanut Butter

- **Quantitative Association Rules**
  - numeric attributes
  - Weight in [70kg – 90kg] => height in [170cm – 190cm]

- **Fuzzy Association Rules**
  - allow different degrees of membership (several categories)
  - to overcome the sharp boundary problem (ex: car’s horsepowers)

- In this lecture, we focused on **Binary Association Rules**
Other Application Areas of AR

- Analysis of credit card purchases.
  - identify the most influential factors common to non-profitable customers, e.g. credit card limit, etc.

- Identification of fraudulent medical insurance claims.
  - analyse claim forms submitted by patients to a medical insurance company
  - find relationships among medical procedures that are often performed together
  - might be indicative for fraud, when common rules are broken

- Recommender Systems
  - E.g. Amazon’s “Customers who bought this item also bought”
  - … is based on association rules
Available Toolkits

- **WEKA**
  - freely available library implemented in Java
  - provides variants of the Apriori Algorithm

- **R**
  - [Link to R-project.org](http://www.r-project.org/)
  - [Link to Rdoc library](http://rss.acs.unt.edu/Rdoc/library/arules/html/apriori.html)

- **DBMiner System**
  - [Han et al. 1996]
Thanks for your attention!

Any Questions …