

Data Mining

- Mining Association Rules

University of Queensland, Brisbane Australia
<http://www.itee.uq.edu.au/~dke>
<http://www.itee.uq.edu.au/~xueli>
 Xue Li

INFS4203 / INFS7203 Data Mining 1

Mining Association Rules

- Apriori Algorithm
- Frequent Pattern (FTP) Tree Algorithm

INFS4203 / INFS7203 Data Mining 2

What are the Association Rules?

- Market basket analysis
 - Given a set of transactions, each transaction is expressed by a set of items. The problem is to analyse customer buying habits by finding associations between the different items that customers place in their "shopping baskets"
 - The buying patterns that reflect items that are frequently associated can be represented in a form of association rules.

INFS4203 / INFS7203 Data Mining 3

Mining Association Rules

- Boolean Associate Rule
 - Concerns the associations between presence or absence of items.

Computer \Rightarrow **financial_management_software**
 [support = 2%, confident = 60%]
- Quantitative Association Rule
 - Discretized quantitative values for items or attributes are partitioned into intervals.

Age(X, "30 .. 39") \wedge **income(X, "42K .. 48K")** \Rightarrow **buys(X, high resolution TV)**
 [support = 14%, confident = 40%]

INFS4203 / INFS7203 Data Mining 4

Mining Association Rules (cont)

- Single Dimensional Associate Rule
 - Only one dimension (attribute) is involved

buys(Computer) \Rightarrow **buys(financial_management_software)**
 [support = 2%, confident = 60%]
- Multi-Dimensional Associate Rule
 - Two or more dimension (attributes) are involved in the rule.

age(X, "30 .. 39") \wedge **income(X, "42K .. 48K")** \Rightarrow **buys(X, high resolution TV)**
 [support = 14%, confident = 40%]
- Single or Multi-level association Rule
 - age(X, "30 .. 39")** \Rightarrow **buys(X, "laptop computer")**
 [support = 10%, confident = 60%]
 - age(X, "30 .. 39")** \Rightarrow **buys(X, "computer")**
 [support = 15%, confident = 80%]

Mining on multidimensional association rules is not discussed in this course.

INFS4203 / INFS7203 Data Mining 5

Example of a Transaction Database

This is an sample input of the algorithm to be discussed.

Transaction Database	Items
100	1 2 3 4 5
200	6 2 3 5
300	1 2 4
400	2 9 4 5
500	1 2 6 4
600	6 3 5
...	...

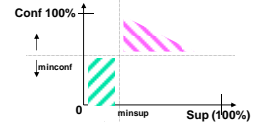
INFS4203 / INFS7203 Data Mining 6

Formal Notations

- A set of items $\mathcal{I} = \{i_1, i_2, \dots, i_n\}$
- A transaction database $\mathcal{D} = \{t_1, t_2, \dots, t_n\}$
 $t_i \subseteq \mathcal{I}$ (a transaction is a set of items)
 TID: $t_i \rightarrow$ transaction number
- An associate rule: $X \Rightarrow Y$,
 $X \subset \mathcal{I}, Y \subset \mathcal{I}$
 $X \cap Y = \emptyset$

Formal Notations (cont')

- Rule confidence: **P for probability.**
 $\text{Confidence}(X \Rightarrow Y) = P(Y | X)$
 transactions in \mathcal{D} contain X has $c\%$ containing Y
- Rule support:
 $\text{support}(X \Rightarrow Y) = P(X \cup Y)$
 $s\%$ of transactions in \mathcal{D} contain $X \cup Y$ (i.e., both X and Y)
- User-specified threshold:
 - min_sup
 - min_conf



Formal Notations (cont')

- Rules that satisfy both a minimum support threshold (min_sup) and a minimum confidence threshold (min_conf) are called **strong**.
- Large itemset (frequent itemset):
 - L_k (itemset, support_count)
 - support_count > min_sup
- Set of candidate k-itemsets
 - C_k (itemset)
- Set of candidate k-itemsets
 - \tilde{C}_k , (T, itemset), when the TIDs (T) of the generating transactions are kept.

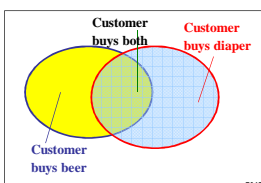
The Problem Statement

- Find all sets of items (itemsets) that have transactions support above min_sup (ie find all large itemsets).
- Use large itemsets to generate association rules.

In this discussion, "large" means "frequent":
 count > min_sup

Example

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



- Itemset $X = \{x_1, \dots, x_k\}$
- Find all the rules $X \Rightarrow Y$ with min confidence and support
 - support, s , probability that a transaction contains $X \cup Y$
 - confidence, c , conditional probability that a transaction having X also contains Y .

Let $\text{min_support} = 50\%$,
 $\text{min_conf} = 50\%$:
 $A \rightarrow C$ (50%, 66.7%)
 $C \rightarrow A$ (50%, 100%)

Support **Confidence**

Example (cont)

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

Min. support 50%
 Min. confidence 50%

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

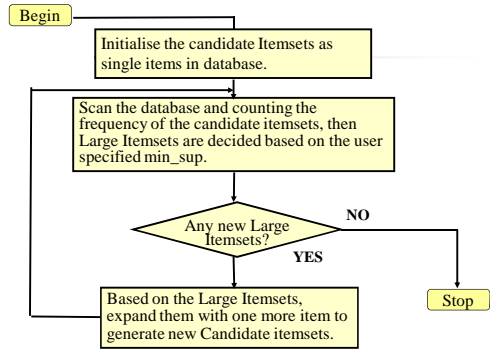
For rule $A \Rightarrow C$:
 support = $\text{support}(\{A\} \cup \{C\}) = 50\%$
 confidence = $\text{support}(\{A\} \cup \{C\}) / \text{support}(\{A\}) = 66.6\%$

What does it mean by "Apriori"?

- Apriori Property:
 - All nonempty subsets of a large (frequent) itemset must also be large (frequent), or
 - If an itemset is not large (frequent), all its supersets cannot be large (frequent). This is called **anti-monotone**.
 - This property is used to create the candidate k -itemsets C_k based on L_{k-1} , then test for L_k .

This is a discovered truth not an assumption.

Apriori Algorithm



Apriori Algorithm

- $L_1 = \{\text{large 1-itemsets}\}$
- for $(k=2; L_{k-1} \neq \emptyset; k++)$ do begin
- $C_k = \text{apriori-gen}(L_{k-1});$ // New candidates
- for each transactions $t \in \mathcal{D}$ do begin
- $C_t = \text{subset}(C_k, t)$ // get the subsets of t that are candidates
- for each candidate $c \in C_t$ do
- $c.\text{count}++;$
- end
- $L_k = \{c \in C_k \mid c.\text{count} \geq \text{min_sup}\}$
- end
- Answer = $\cup_k L_k;$

Apriori-gen Procedure

Input: L_{k-1} // Generate the candidate itemsets C_k from L_{k-1}
Return: C_k

Step 1: Join L_{k-1} with L_{k-1} to create new candidates

```
SELECT INTO C_k
SELECT p.item1, p.item2, ..., p.item_{k-1}, q.item_{k-1}
FROM L_{k-1} p, L_{k-1} q
WHERE p.item1 = q.item2, ..., p.item_{k-2} = q.item_{k-2},
      p.item_{k-1} < q.item_{k-1};
```

Step 2: Prune: Delete itemsets $c \in C_k$ if some $(k-1)$ subsets of c is not in L_{k-1}

```
for each itemsets c in C_k do
for each k-1_subsets s of c do
if (s not in L_{k-1}) then
delete c from C_k;
```

How to justify this deletion?

end;

Apriori Example

Database

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

C_1

Itemset	Support Count
{1}	2
{2}	3
{3}	3
{4}	1
{5}	3

L_1

Itemset	Support
{1}	2
{2}	3
{3}	3
{5}	3

C_2

Itemset	Support
{1 2}	1
{1 3}	2
{1 5}	1
{2 3}	2
{2 5}	3
{3 5}	2

Min_Sup = 2

Apriori Example

L_2

Itemset	Support
{1 3}	2
{2 3}	2
{2 5}	3
{3 5}	2

C_3

Itemset	Support
{2 3 5}	2
{1 2 3}	1

Min_Sup = 2

L_3

Itemset	Support
{2 3 5}	2

Problem in this Algorithm:

When Database is scanned to check C_k for creating L_k , a large number of transactions will be scanned even they do not contain any k -itemset.

Maximum Frequent Itemsets: Max-patterns

- Frequent pattern $\{a_1, \dots, a_{100}\} \rightarrow \binom{100^1}{1} + \binom{100^2}{2} + \dots + \binom{100^0}{0} = 2^{100} - 1 = 1.27 * 10^{30}$ frequent sub-patterns!
- Max-pattern: frequent patterns without proper frequent super pattern
 - BCDE, ACD are max-patterns
 - BCD is not a max-pattern

Min_sup=2

Tid	Items
10	A,B,C,D,E
20	B,C,D,E
30	A,C,D,F

INFS4203 / INFS7203 Data Mining

19

How to Generate Association Rules from Frequent Itemsets?

- Create Rules for Maximum Frequent Itemsets: No trivial rules.
- Confidence $(A \Rightarrow B) = P(B|A) = \text{Support_Count}(X \cup Y) / \text{Support_Count}(A)$
 - For each frequent itemset L , generate all non-empty subsets of L .
 - For every nonempty subset S of L , generate rule: $S \Rightarrow (L - S)$, then calculate its confidence $c = \text{support_count}(L) / \text{Support_Count}(S)$. If $c \geq \text{min_conf}$, the rule is a strong association rule.

INFS4203 / INFS7203 Data Mining

20

Problems with Apriori Algorithms

- It is costly to handle a huge number of candidate sets. For example if there are 10^4 large 1-itemsets, the Apriori algorithm will need to generate more than 10^7 candidate 2-itemsets. Moreover for 100-itemsets, it must generate more than $2^{100} \approx 10^{30}$ candidates in total.
- The candidate generation is the inherent cost of the Apriori Algorithms, no matter what implementation technique is applied.

INFS4203 / INFS7203 Data Mining

21

It is **not** a good idea to use Apriori Algorithms to mine a large database for long patterns.

INFS4203 / INFS7203 Data Mining

22

Frequent Pattern Tree Algorithm ("FP-Growth Algorithm")

- Items in transactions are sorted based on their frequencies in the database (in descending order).
- A tree structure (Prefix-Tree) is used to record the frequent patterns top down. Only frequent (large) item will have nodes in the tree (i.e., database is compressed).
- Each projection on the tree is to mine all frequent patterns associated with the item (in the Header Table) and the procedure is recursive.
- It is not Apriori like restricted generate-and-test but restricted test (divide-and-conquer) only.

INFS4203 / INFS7203 Data Mining

23

Mining Frequent Patterns Without Candidate Generation

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

INFS4203 / INFS7203 Data Mining

24

FP Tree Algorithm

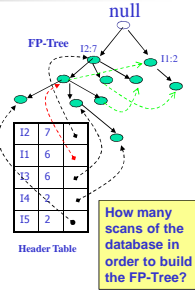
Building the FP-Tree:

Initialise the Frequent (large) Itemsets as single items in database. Sort the 1-itemsets in the descending order.

Create the root of FP-Tree and label it as "null".

Construct the tree by scanning the transactions in the database according to the order of the 1-itemsets. Create a branch in the tree if there is no common prefix in the path of the tree. The counting is performed for the items in the transaction along the path of the tree.

An item header table is used to link all leave nodes in a list.



INFS4203 / INFS7203 Data Mining

25

Construct FP-tree from a Transaction Database

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, w}	{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}

min_support = 3

- Scan DB once, find frequent 1-itemset (single item pattern)
- Sort frequent items in frequency descending order, f-list
- Scan DB again, construct FP-tree

Header Table

Item	frequency head
f	4
c	4
a	3
b	3
m	3
p	3

F-list=f-c-a-b-m-p

INFS4203 / INFS7203 Data Mining

26

FP Tree Algorithm (cont)

Mining Frequent Patterns from the FP-Tree:

Start from the last item header table as the suffix pattern, construct its conditional pattern base. Along the path from the prefix, all the counting (support) will be the same number as the suffix.

Based on the conditional pattern base, construct the item's conditional FP-Tree, and performing recursively on such a tree.

The pattern growth is achieved by the concatenation of the suffix pattern with the frequent patterns generated from a conditional FP-Tree.

The algorithm stops when the traversal of the header table finishes. All the frequent patterns (i.e., large itemsets) are created in the process of the pattern growth.

INFS4203 / INFS7203 Data Mining

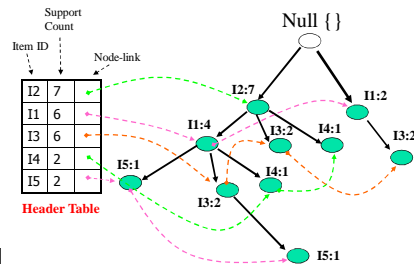
27

Example of FP Tree Algorithm

FP-Tree Construction:

Database

TID	List of item IDs
100	I1, I2, I5
200	I2, I4
300	I2, I3
400	I1, I2, I4
500	I1, I3
600	I2, I3
700	I1, I3
800	I1, I2, I3, I5
900	I1, I2, I3



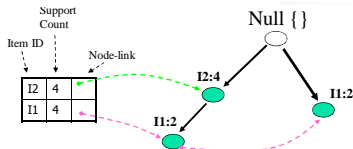
INFS4203 / INFS7203 Data Mining

28

Example of FP Tree Algorithm (cont)

Finding Frequent Patterns:

The conditional FP-Tree associated with the conditional node I3.



item	Conditional pattern base	Conditional FP-Tree	Frequent patterns generated
I5	{ (I2 I1: 1), (I2 I1 I3: 1) }	<I2: 2, I1: 2>	I2 I5: 2, I1 I5: 2, I2 I1 I5: 2
I4	{ (I2 I1: 1), (I2: 1) }	<I2: 2>	I2 I4: 2
I3	{ (I2 I1: 2), (I2: 2), (I1: 2) }	<I2: 4, I1: 2>, <I1: 2>	I2 I3: 4, I1 I3: 4, I2 I1 I3: 2
I1	{ (I2: 4) }	<I2: 4>	I2 I1: 4

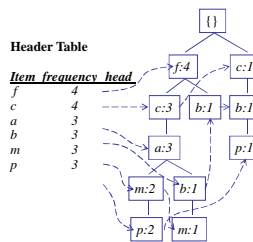
INFS4203 / INFS7203 Data Mining

29

Another example:

Find Patterns Having P From P-conditional Database

- Starting at the frequent item header table in the FP-tree
- Traverse the FP-tree by following the link of each frequent item p
- Accumulate all of transformed prefix paths of item p to form p's conditional pattern base.



Conditional pattern bases

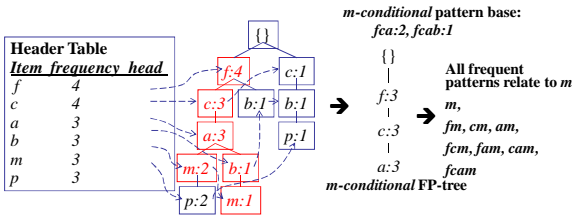
item	cond. pattern base
c	f:3, {}
a	fc:3
b	fca:1, f:1, c:1
m	fca:2, fcab:1
p	fcam:2, cb:1

INFS4203 / INFS7203 Data Mining

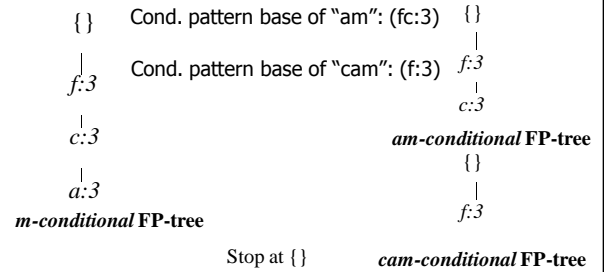
30

Example (cont)

- For each pattern-base
 - Accumulate the count for each item in the base
 - Construct the FP-tree for the frequent items of the pattern base



Recursion: Mining Each Conditional FP-tree



Summary: Mining Frequent Patterns With FP-trees

- Idea: Frequent pattern growth
 - Recursively grow frequent patterns by pattern and database partition
- Method
 - For each frequent item, construct its conditional pattern-base, and then its conditional FP-tree
 - Repeat the process on 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

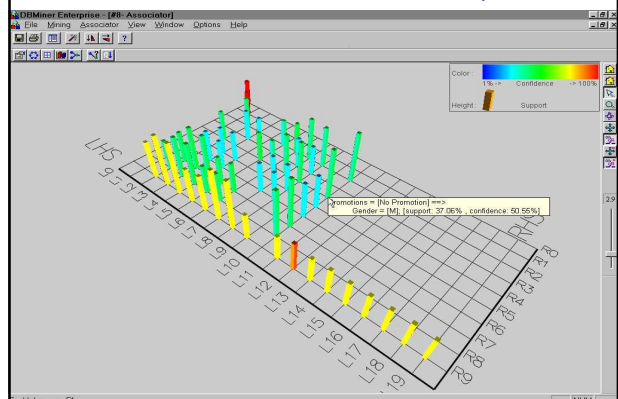
Benefits of the FP-tree Structure

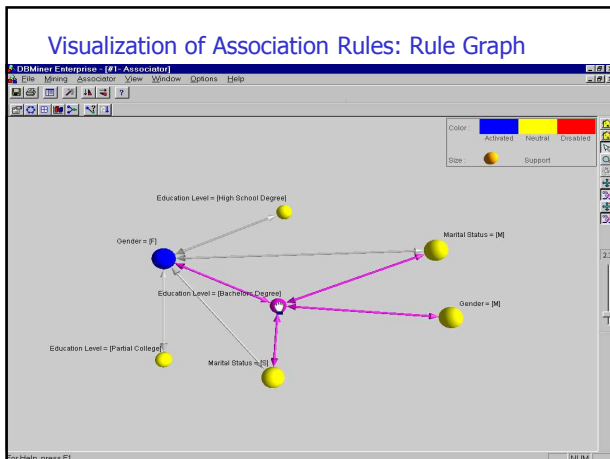
- Completeness
 - Preserve complete information for frequent pattern mining
 - Never break a long pattern of any transaction
- Compactness
 - Reduce irrelevant info—infrequent items are gone
 - Items in frequency descending order: the more frequently occurring, the more likely to be shared
 - Never be larger than the original database (not count node-links and the count field)
 - For Connect-4 DB, compression ratio could be over 100

Why Is FP-Growth the Winner?

- Divide-and-conquer:
 - decompose both the mining task and DB according to the frequent patterns obtained so far
 - leads to focused search of smaller databases
- Other factors
 - no candidate generation, no candidate test
 - compressed database: FP-tree structure
 - no repeated scan of entire database
 - basic ops—counting local freq items and building sub FP-tree, no pattern search and matching

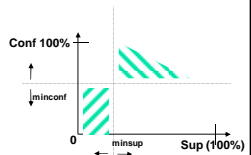
Visualization of Association Rules: Pane Graph





Current Research Issues in Association Rules Mining

- Explanation and Interpretations
 - Causal Relationship/Dependence discovery
- Automatically determining the minimum support and confidence
- High-dimensionality versus Hierarchical mining
- Mining on streaming data
- Mining Negative Rules
 - $A \rightarrow B, \neg A \rightarrow B, \neg A \rightarrow \neg B$



INFS4203 / INFS7203 Data Mining

38

Reading List: Association Rules

- Rakesh Agrawal and Ramakrishnan Srikant, *Fast Algorithms for Mining Association Rules*, Proc of 20th VLDB Conference Santiago, Chile, 1994.*
- Jiawei Han, Jian Pei, and Yiwen Yin, *Mining Frequent Patterns without Candidate Generation* In Proce. Of the 2000 ACM-SIGMOD int'l Conf. On Management of Data, Dallas, Texas, USA, May 2000.*
- Jochen Hipp, Ulrich Guntzer, and Gholamreza, *Algorithms for Association Rule Mining – A general Survey and Comparison* ACM SIGKDD Explorations, Vol 2 Issue 1, July 2000.

INFS4203 / INFS7203 Data Mining

39