Association Rule Mining: Basic Concepts and Methods

CDS6314 LECTURE 4

MULTIMEDIA UNIVERSITY

Outline

- **Basic Concepts**
- Frequent Itemset Mining Methods
- Pattern Interestingness Evaluation Methods
- Summary

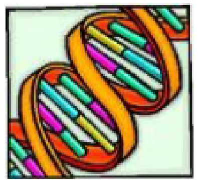
Did you notice this?

What Is Association Mining?

- Association rule mining:
  - Other terms: **Affinity Analysis**; **Market Basket Analysis**
  - Finding frequent patterns, associations, correlations, or causal structures among sets of items or objects in transaction databases, relational databases, and other information repositories.
  - *Frequent pattern*: pattern (set of items, sequence, etc.) that occurs frequently in a database

Motivation of Association Mining

- Finding regularities in data
  - What products were often purchased together?
    - Bread and Milk? Bread and Butter? Bread and Rice?
  - What are the subsequent purchases after buying a PC?
  - What kinds of DNA are sensitive to this new drug?
  - Can we automatically classify web documents?



Why Association Mining?

- Broad applications
  - Basket data analysis, cross-marketing, catalog design, sale campaign analysis
  - Web log (click stream) analysis, DNA sequence analysis, etc.
- Example:
  - $buys(x, \text{"Bread"}) \rightarrow buys(x, \text{"Milk"})[0.5\%, 50\%]$
  - $major(x, \text{"CS"}) \wedge takes(x, \text{"DB"}) \rightarrow grade(x, \text{"A"})[1\%, 75\%]$

Basic Concepts

- Association rules show attribute value conditions that occur frequently together in a given dataset.
- More specific applications:
  - $* \Rightarrow \text{Chicken Rice}$ (What the shop should do to boost Chicken Rice sales)
  - $\text{Laptop} \Rightarrow *$ (What other products related to laptop should the store stocks up?)

Basic Concepts

- Given:
  1) database of transactions
  2) each transaction is a list of items (purchased by a customer in a visit)
- Find: ALL rules that correlate the presence of one set of items with that of another set of items
  - E.g., 98% of people who purchase tires and auto accessories also get automotive services done

$\underbrace{\text{IF a person buys bread,}}_{\text{Antecedent}}\ \underbrace{\text{THEN they will buy chocolates}}_{\text{Consequent}}$

Basic Concept: Frequent Itemsets (Patterns)

- Itemset: A set of one or more items
- k-itemset: $X = \{x_1, \dots, x_k\}$
- (*absolute*) *support (count)* of X: Number of occurrences (frequency) of an itemset X (i.e., number of transactions that contain X)
- (*relative*) *support*, s: The fraction of transactions that contains X (i.e., the probability that a transaction contains X)
- An itemset X is *frequent* if the support of X is no less than a *min_sup* threshold (denoted as σ)

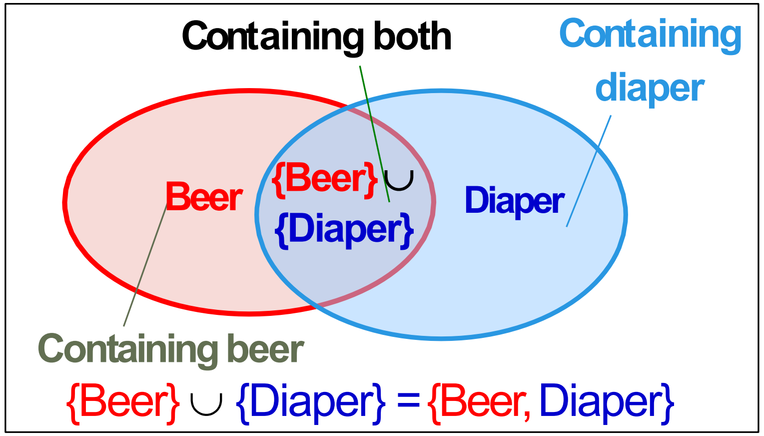
TID	Items bought
10	Beer, Nuts, Diaper
20	Beer, Coffee, Diaper
30	Beer, Diaper, Eggs
40	Nuts, Eggs, Milk
50	Nuts, Coffee, Diaper, Eggs, Milk

- Let *min_sup = 50%*
- Freq. 1-itemsets:
  - Beer: 3 (60%); Nuts: 3 (60%)
  - Diaper: 4 (80%); Eggs: 3 (60%)
- Freq. 2-itemsets:
  - {Beer, Diaper}: 3 (60%)

Basic Concept: Association Rules

- Association rules: $X \rightarrow Y(s, c)$
  - Support, s: The probability that a transaction contains $X \cup Y$
  - Confidence, c: The conditional probability that a transaction containing X also contains Y
  - $c = \sup(X \cup Y) / \sup(X)$
- **Association rule mining**: Find all of the rules, $X \rightarrow Y$, with *minimum support and confidence*
- Frequent itemsets: Let *min_sup = 50%*
  - Freq. 1-itemsets: Beer: 3, Nuts: 3, Diaper: 4, Eggs: 3
  - Freq. 2-itemsets: {Beer, Diaper}: 3
- Association rules: Let *min_conf = 50%*
  - *Beer* → *Diaper* (60%, 100%)(Q: Are these all rules?)
  - *Diaper* → *Beer* (60%, 75%)

TID	Items bought
10	Beer, Nuts, Diaper
20	Beer, Coffee, Diaper
30	Beer, Diaper, Eggs
40	Nuts, Eggs, Milk
50	Nuts, Coffee, Diaper, Eggs, Milk



Mining Association Rules: An Example

Transaction ID	Items Bought
2000	A, B, C
1000	A, C
4000	A, D
5000	B, E, F

Min. support = 50%
Min. confidence = 50%

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

- For rule $A \Rightarrow C$:
  - Support $=$ $\text{support}(\{A \cup C\}) = \frac{1}{2} = 50\%$
  - Confidence $= \frac{\text{support}(\{A \cup C\})}{\text{support}(\{A\})} = \frac{2}{3} = 66.6\%$

Interestingness Measurements

- Pattern-mining will generate a large set of patterns/rules
  - Not all the generated patterns/rules are interesting
- Interestingness measures: Objective vs. subjective
  - Objective interestingness measures: Based on threshold values controlled by the user.
    - Support, confidence, correlation, …
  - Subjective interestingness measures: Often based on earlier user experiences and beliefs
    - Query-based: Relevant to a user’s particular request
    - Actionable: User can do something with the patterns
    - Against one’s knowledge-base: unexpected, freshness, timeliness

The Challenge

- A long pattern contains a combinatorial number of sub-patterns
- For example, a frequent itemset of length 100, such as $\{a_1, a_2, \ldots, a_{100}\}$, will contain:
  - $\binom{100}{1}$ = 100 frequent 1-itemsets: $\{a_1\}, \{a_2\}, \ldots, \{a_{100}\}$,
  - $\binom{100}{2}$ frequent 2-itemsets: $\{a_1, a_2\}, \{a_1, a_3\}, \ldots, \{a_{99}, a_{100}\}, \ldots, \ldots, \ldots, \ldots$
  - and so on till $\binom{100}{100}$.
  - The total number of frequent itemset in total:

$$\binom{100}{1} + \binom{100}{2} + \cdots + \binom{100}{100} = 2^{100} - 1 \approx 1.27 \times 10^{30} \text{ sub-patterns}$$

- There are too many frequent patterns from a large dataset
  - Especially if the minimum support is set low
  - Sets of sub-patterns too huge for any computer to compute or store

Outline

- Basic Concepts
- **Frequent Itemset Mining Methods**
- Pattern Interestingness Evaluation Methods
- Summary

Pattern Mining Methods

- The Downward Closure Property of Frequent Patterns (Apriori Algorithm)
  - Extensions or Improvements of Apriori
- Mining Frequent Patterns by Exploring Vertical Data Format
- FPGrowth: A Frequent Pattern-Growth Approach

Mining Frequent Itemsets: The Key Step

- Find the **frequent** itemsets: the sets of items that must have minimum support
- Frequent patterns have a downward closure property (Apriori)
  - If {beer, diaper, nuts} is frequent, so is {beer, diaper}
  - Every transaction containing {beer, diaper, nuts} also contains {beer, diaper}
- **A subset of a frequent itemset must also be a frequent itemset**
- Iteratively find frequent itemsets with cardinality from 1 to k (k-itemset)
- Use the frequent itemsets to generate association rules
  - If any subset of an itemset S is infrequent, then there is no chance for S to be frequent (no need to consider S rules, more efficient mining)

Apriori: A Candidate Generation-and-test Approach

- Apriori property: Any subset of a frequent itemset must be frequent
- Apriori pruning principle: If there is ANY itemset that is infrequent, its superset should not be generated/tested
- Method: level-wise, candidate generation and test (Apriori outline)
  - Initially, scan DB once to get frequent 1-itemset
  - Repeat
    - Generate length-(k+1) candidate itemsets from length-k frequent itemsets
    - Test the candidates against DB to find frequent (k+1)-itemsets
    - Set k := k +1
  - Until no frequent or candidate set can be generated
  - Return all the frequent itemsets derived

The Apriori Algorithm

- **Join Step**: C_k is generated by joining F_{k-1} with itself
- **Prune Step**: Any $(k-1)$-itemset that is not frequent cannot be a subset of a frequent k-itemset
- Pseudo-code:

C_k : Candidate itemset of size k
F_k : Frequent itemset of size k

$k = 1$;
F_k ={frequent items}; //frequent 1-itemset
while ($F_k! = \emptyset$) do{ //when F_k is non-empty
    C_{k+1} = candidates generated from F_k; //candidate generation
    F_{k+1} = candidatest C_{k+1} above min _sup; //candidate pruning
    $k = k + 1$
    }
Return $\cup_k F_k$ //return F_k generated at each level

The Apriori Algorithm: An example

minsup = 2

Database TDB

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

1st scan

C_1

Itemset	sup
{A}	2
{B}	3
{C}	3
{D}	1
{E}	3

F_1

Itemset	sup
{A}	2
{B}	3
{C}	3
{E}	3

C_2

Itemset
{A, B}
{A, C}
{A, E}
{B, C}
{B, E}
{C, E}

To generate C2, F1 join F1.
To generate C3, F2 join F2.

2nd scan

C_2

Itemset	sup
{A, B}	1
{A, C}	2
{A, E}	1
{B, C}	2
{B, E}	3
{C, E}	2

F_2

Itemset	sup
{A, C}	2
{B, C}	2
{B, E}	3
{C, E}	2

C_3

Itemset
{B, C, E}

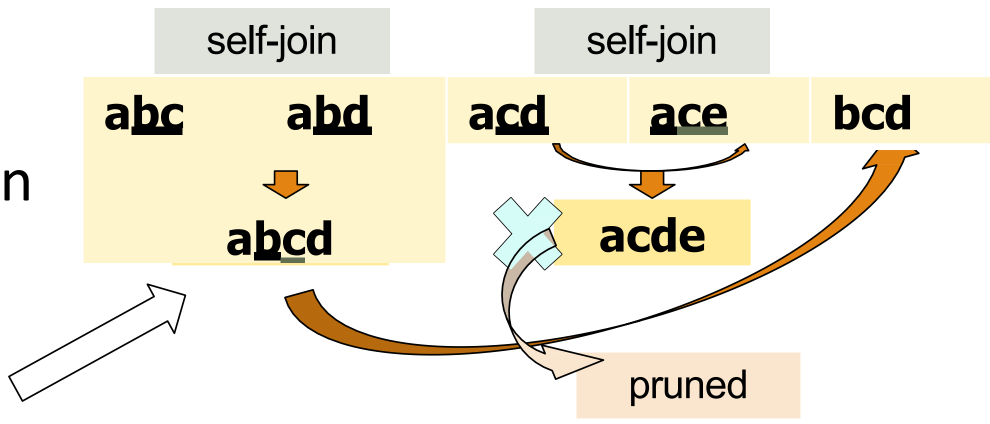
3rd scan

F_3

Itemset	sup
{B, C, E}	2

Important Details of Apriori

- How to generate candidates?
  - Step 1: self-joining F_k
  - Step 2: pruning
- Example of candidate-generation
  - $F_3 = \{abc, abd, acd, ace, bcd\}$
  - Self-joining: $F_3 * F_3$
    - $abcd$ from abc and abd
    - $acde$ from acd and ace
  - Pruning:
    - $acde$ is removed because ade is not in F_3
    - $C_4 = \{abcd\}$



Challenges in Frequent Pattern Mining

- Challenges
  - Multiple scans of transaction database
  - Huge number of candidates
  - Tedious workload of support counting for candidates
- Improving Apriori: general ideas
  - Reduce passes of transaction database scans
  - Shrink number of candidates
  - Facilitate support counting of candidates

Equivalence Class Transformation (ECLAT)

- Vertical data format to improve database scanning
- ECLAT is a depth-first search algorithm using set intersection
- For each item, store a list of transaction ids (*tids*)

Horizontal Data Layout

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

Vertical Data Layout

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

↓

TID-list

ECLAT

- Determine the support of any k-itemset by intersecting tid-lists of two of its $(k-1)$ subsets
- Advantage: very fast support counting
- Disadvantage: intermediate *tid*-lists may become too large for memory

A
1
4
5
6
7
8
9

$\wedge$

B
1
2
5
7
8
10

$\rightarrow$

AB
1
5
7
8

Frequent Pattern Tree (FP-Tree)

- Method to mine frequent pattern without candidate generation
- Compress a large database into a compact structure
  - highly condensed, but complete for frequent pattern mining
  - avoid costly database scans
- FP-tree-based frequent pattern mining method
  - A divide-and-conquer methodology: decompose mining tasks into smaller ones
  - Avoid candidate generation: sub-database test only

FP-Growth Method: Construction of FP-Tree

1. Create the root of the tree, labeled with "null"
2. Scan database to create 1-itemset, and then list in *L* order according to support counts.
3. Remove items lower than minimum support
4. Scan the database D a second time and sort the items in each transaction in *L* order (i.e. sorted order).
5. A branch is created for each transaction with items having their support count separated by colon.
6. Whenever the same node is encountered in another transaction, just increment the support count of the common node or Prefix.
7. To facilitate tree traversal, an item header table is built so that each item points to its occurrences in the tree via a chain of node-links.
8. Now, The problem of mining frequent patterns in database is transformed to that of mining the FP-Tree.

FP-tree Construction: An example

TID	Items in the Transaction
100	$\{f, a, c, d, g, i, m, p\}$
200	$\{a, b, c, f, l, m, o\}$
300	$\{b, f, h, j, o, w\}$
400	$\{b, c, k, s, p\}$
500	$\{a, f, c, e, l, p, m, n\}$

1. Scan DB once, find single item frequent pattern: Let $\text{min\_sup} = 3$

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

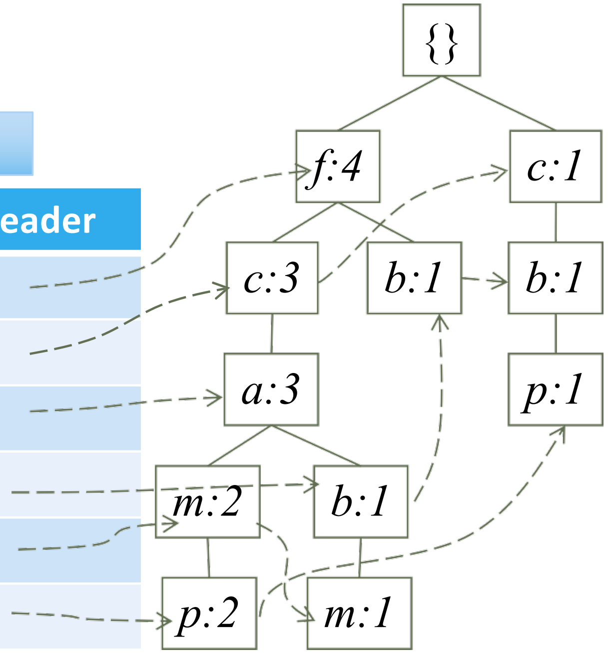
2. Sort frequent items in frequency descending order, f-list

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

3. Scan DB again, construct FP-tree

Header Table

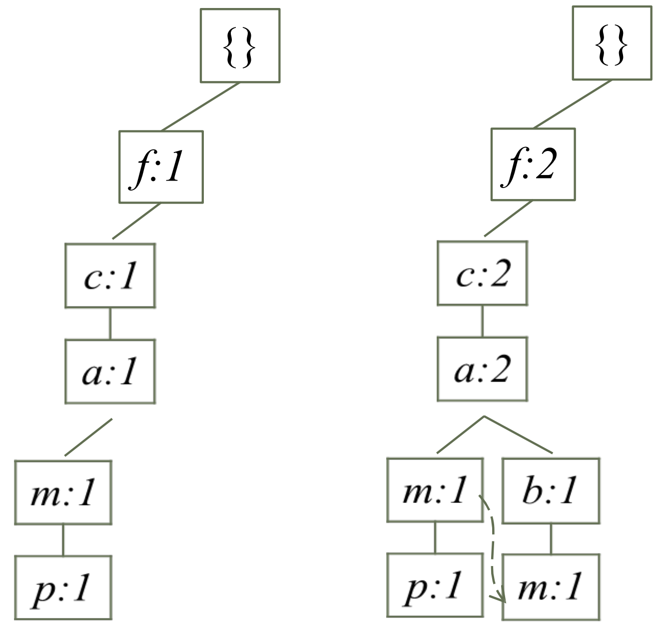
Item	Frequency	Header
f	4	
c	4	
a	3	
b	3	
m	3	
p	3	



Constructing the FP-tree

TID	Items in the Transaction	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	$\{a, f, c, e, l, p, m, n\}$	$\{f, c, a, m, p\}$

Item	Frequency	Header
f	4	
c	4	
a	3	
b	3	
m	3	
p	3	

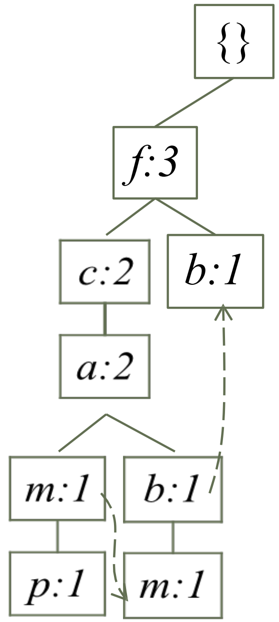


{f, c, a, m, p} **{f, c, a, b, m}**

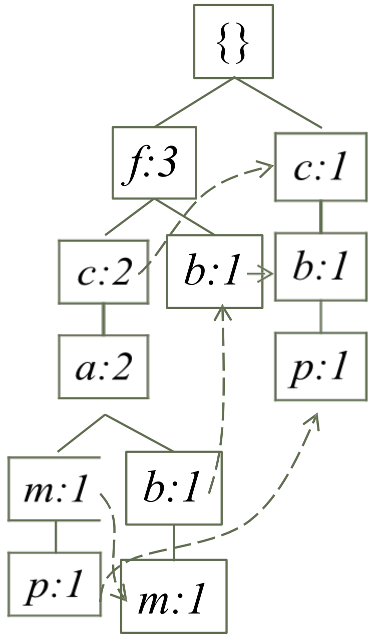
Constructing the FP-tree

TID	Items in the Transaction	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	$\{a,f,c,e,l,p,m,n\}$	$\{f,c,a,m,p\}$

Item	Frequency	Header
f	4	
c	4	
a	3	
b	3	
m	3	
p	3	



{f, b}

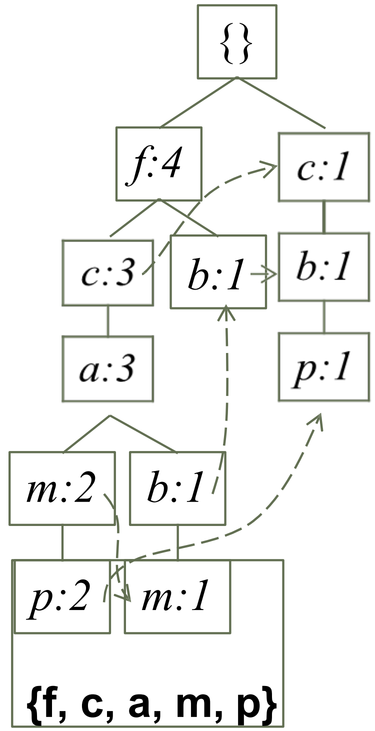


{c, b, p}

Constructing the FP-tree

TID	Items in the Transaction	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	$\{a,f,c,e,l,p,m,n\}$	$\{f,c,a,m,p\}$

Item	Frequency	Header
f	4	
c	4	
a	3	
b	3	
m	3	
p	3	

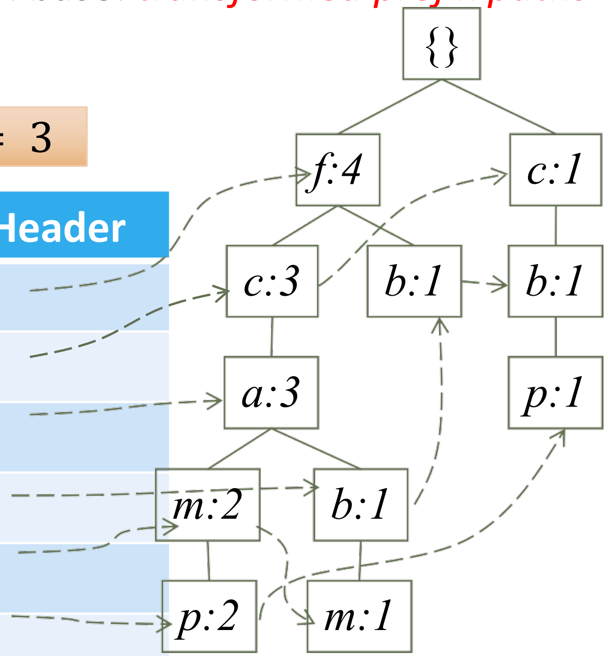


Conditional Pattern-Base

- Divide and conquer based on patterns and data
- Pattern mining can be partitioned according to current patterns
  - Patterns containing p: p's conditional database: $fcam: 2, cb: 1$
  - Patterns having m but no p : m's conditional database: $fca: 2, fcab: 1$
  - …… ……
- p's conditional pattern base: *transformed prefix paths* of item p

min_support = 3

Item	Frequency	Header
f	4	
c	4	
a	3	
b	3	
m	3	
p	3	



Conditional pattern bases

Item	Conditional pattern base
c	$f: 3$
a	$fc: 3$
b	$fca: 1, f: 1, c: 1$
m	$fca: 2, fcab: 1$
p	$fcam: 2, cb: 1$

Mining Conditional Pattern-Base

Conditional pattern bases

$\text{min\_support} = 3$

Item	Conditional pattern base
c	$f: 3$
a	$fc: 3$
b	$fca: 1, f: 1, c: 1$
m	$fca: 2, fcab: 1$
p	$fcam: 2, cb: 1$

- For each conditional pattern-base
  - Mine single-item patterns
  - Construct its conditional FP-tree & mine it

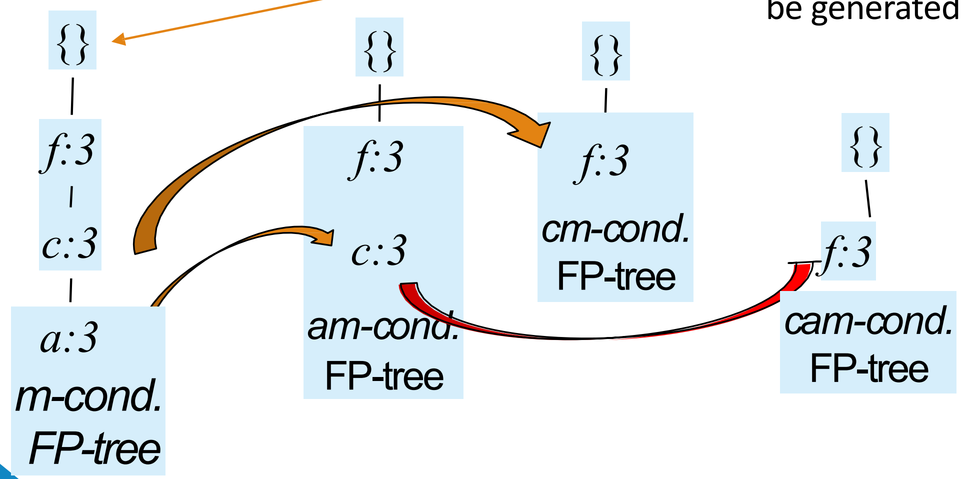
p-conditional PB: $fcam: 2, cb: 1 \rightarrow c: 3$

Conditional FP-tree (CFT) is only $\{c: 3\}$, because you see it in both prefix paths.
The rest are not included as their support count is less than 3.

m-conditional PB: $fca: 2, fcab: 1 \rightarrow fca: 3$

b-conditional PB: $fca: 1, f: 1, c: 1 \rightarrow \emptyset$

Actually, for single branch FP-tree, all frequent patterns can be generated in one shot



$m: 3$
$fm: , cm: 3, am: 3$
$fcm: 3, fam: 3, cam: 3$
$fcam: 3$

Improving Apriori Efficiency

- Hash-based itemset counting: A *k*-itemset whose corresponding hashing bucket count is below the threshold cannot be frequent
- Transaction reduction: A transaction that does not contain any frequent k-itemset is useless in subsequent scans
- Partitioning: Any itemset that is potentially frequent in DB must be frequent in at least one of the partitions of DB
- Sampling: mining on a subset of given data, lower support threshold + a method to determine the completeness
- Dynamic itemset counting: add new candidate itemsets only when all of their subsets are estimated to be frequent

Outline

- Basic Concepts
- Frequent Itemset Mining Methods
- **Pattern Interestingness Evaluation Methods**
- Summary

Support and Confidence

- If confidence gets a value of 100 % the rule is an **exact rule**
- Even if confidence reaches high values the rule is not useful unless the support value is high as well
- Rules that have both high confidence and support are called **strong rules**
- But strong rules are not necessarily interesting

Limitation of Support-Confidence

- Are s and c interesting in association rules: "$A \Longrightarrow B$" $[s, c]$?
- Example: Suppose one school may have the following statistics on # of students who may play basketball and/or eat cereal:

	play-basketball	not play-basketball	sum (row)
eat-cereal	400	350	750
not eat-cereal	200	50	250
sum (col.)	600	400	1000

2-way contingency table

- Association rule mining may generate the following:
  - *play-basketball* $\Longrightarrow$ *eat-cereal* [40%, 66.7%] (higher s & c)
    - Looks good. But if you generate another rule
  - ¬ *play-basketball* $\Longrightarrow$ *eat-cereal* [35%, 87.5%] (high s & c)
- These two rules are confusing

Interestingness Measure: Lift

- Measure of dependent/correlated events: **lift**

$$\text{lift}(B,C) = \frac{c(B \rightarrow C)}{s(C)} = \frac{s(B \cup C)}{s(B) \times s(C)}$$

- $\text{lift}(B,C)$ may tell how B and C are correlated
  - $\text{lift}(B,C) = 1$: B and C are independent
  - > 1: positively correlated
  - < 1: negatively correlated

Lift is more telling than s & c

	B	$\neg B$	Σ_{row}
C	400	350	750
$\neg C$	200	50	250
Σ_{col}	600	400	1000

- For our example, $\text{lift}(B,C) = \frac{400/1000}{600/1000 \times 750/1000} = 0.89$

$$\text{lift}(B,\neg C) = \frac{200/1000}{600/1000 \times 250/1000} = 1.33$$

- Thus, B and C are negatively correlated since $\text{lift}(B,C) < 1$;
  - B and $\neg C$ are positively correlated since $\text{lift}(B,\neg C) > 1$

Lift Ratio

$$\text{Benchmark Confidence} = P(\text{Consequent}) = \frac{\text{no. transactions with consequent itemset}}{\text{no. transactions in database}}$$

$$\text{Lift ratio} = \frac{\text{Confidence}}{\text{Benchmark Confidence}}$$

- A lift ratio is greater than 1.0 suggest that there is some usefulness to the rule
- The level of association between the antecedent and consequent item sets is higher than would be expected if they are independent
- The larger the lift ratio, the greater the strength of the association

Interestingness Measure: χ^2

- Another measure to test correlated events: χ^2

$$\chi^2 = \sum \frac{(\text{Observed} - \text{Expected})^2}{\text{Expected}}$$

Observed value

Expected value

	B	$\neg B$	Σ_{row}
C	400 (450)	350 (300)	750
$\neg C$	200 (150)	50 (100)	250
Σ_{col}	600	400	1000

- General rules
  - $\chi^2 = 0$: independent
  - $\chi^2 > 0$: correlated, either positive or negative, so it needs additional test
- Now, $\chi^2 = \frac{(400-450)^2}{450} + \frac{(350-300)^2}{300} + \frac{(200-150)^2}{150} + \frac{(50-100)^2}{100} = 55.56$
- χ^2 shows B and C are negatively correlated since the expected value is 450 but the observed is only 400
- χ^2 is also more telling than the support-confidence framework

Lift and χ^2: Limitations

- Null transactions: Transactions that contain neither B nor C
- Let's examine the dataset D
  - BC (100) is much rarer than $B\neg C$ (1000) and $\neg BC$ (1000), but there are many $\neg B\neg C$ (100000)
  - Unlikely B & C will happen together
- $\text{Lift}(B, C) = 8.44 >> 1$
  - Lift shows B and C are strongly positively correlated
- $\chi^2 = 670$: Observed(B, C) >> expected value (11.85)
- Too many null transactions may spoil interestingness indication

	B	$\neg B$	Σ_{row}
C	100	1000	1100
$\neg C$	1000	100000	101000
Σ_{col}	1100	101000	102100

null transactions

Contingency table with expected values added

	B	$\neg B$	Σ_{row}
C	100 (11.85)	1000	1100
$\neg C$	1000 (988.15)	100000	101000
Σ_{col}	1100	101000	102100

Interestingness Measures & Null-Invariance

- *Null invariance*: Value does not change with the # of null-transactions
- A few interestingness measures: Some are null invariant

Measure	Definition	Range	Null-Invariant
$\chi^2(A,B)$	$\sum_{i,j=0,1}\frac{(e(a_ib_j)-o(a_ib_j))^2}{e(a_ib_j)}$	$[0,\infty]$	No
$Lift(A,B)$	$\frac{s(A\cup B)}{s(A)\times s(B)}$	$[0,\infty]$	No
$AllConf(A,B)$	$\frac{s(A\cup B)}{max\{s(A),s(B)\}}$	$[0,1]$	Yes
$Jaccard(A,B)$	$\frac{s(A\cup B)}{s(A)+s(B)-s(A\cup B)}$	$[0,1]$	Yes
$Cosine(A,B)$	$\frac{s(A\cup B)}{\sqrt{s(A)\times s(B)}}$	$[0,1]$	Yes
$Kulczynski(A,B)$	$\frac{1}{2}(\frac{s(A\cup B)}{s(A)}+\frac{s(A\cup B)}{s(B)})$	$[0,1]$	Yes
$MaxConf(A,B)$	$max\{\frac{s(A)}{s(A\cup B)},\frac{s(B)}{s(A\cup B)}\}$	$[0,1]$	Yes

χ^2 and *lift* are not *null-invariant*

Jaccard, cosine, AllConf, MaxConf, and *Kulczynski* are *null-invariant measures*

ExKulc: 0- negatively correlated, 0.5- neutral, 1- positively correlated

Importance of Null Invariance

Dataset	mc	$\neg mc$	$m\neg c$	$\neg m\neg c$
D_1	10,000	1,000	1,000	100,000
D_2	10,000	1,000	1,000	100
D_3	100	1,000	1,000	100,000
D_4	1,000	1,000	1,000	100,000
D_5	1,000	100	10,000	100,000
D_6	1,000	10	100,000	100,000

- Let's look at another ex. Check the first 4 data sets.
  - m and c are positively associated in D1 and D2
    - mc(10,000) >> $\bar{m}c$(1000) and $m\bar{c}$(1000)
  - m and c are negatively associated in D3
    - mc(100) << $\bar{m}c$(1000) and $m\bar{c}$(1000)
  - m and c are neutral in D4
    - mc(1000) = $\bar{m}c$(1000) and $m\bar{c}$(1000)

milk vs. coffee contingency table

	milk	¬milk	Σ_{row}
coffee	mc	$\neg mc$	c
¬coffee	$m\neg c$	$\neg m\neg c$	$\neg c$
Σ_{col}	m	$\neg m$	Σ

Importance of Null Invariance

- Why is null invariance crucial for the analysis of massive transaction data?
  - Many transactions may contain neither milk nor coffee
- Lift and χ^2 are not null-invariant: not good to evaluate data that contain too many (D1) or too few (D2) null transactions
- Many measures are not null-invariant

milk vs. coffee contingency table

	milk	¬milk	Σ_{row}
coffee	mc	$\neg mc$	c
¬coffee	$m\neg c$	$\neg m\neg c$	$\neg c$
Σ_{col}	m	$\neg m$	Σ

Dataset	mc	$\neg mc$	$m\neg c$	$\neg m\neg c$	χ^2	$Lift$
D_1	10,000	1,000	1,000	100,000	90557	9.26
D_2	10,000	1,000	1,000	100	0	1
D_3	100	1,000	1,000	100,000	670	8.44
D_4	1,000	1,000	1,000	100,000	24740	25.75
D_5	1,000	100	10,000	100,000	8173	9.18
D_6	1,000	10	100,000	100,000	965	1.97

Null-transactions w.r.t. m and c

Comparison of Null-Invariant Measures

- Not all null-invariant measures are created equal
- $D_4 - D_6$ differentiate the null-invariant measures
- So, which one is better?
  - We use Imbalance Ratio (IR) to measure.

2-variable contingency table

	milk	¬milk	Σ_{row}
coffee	mc	$\neg mc$	c
¬coffee	$m\neg c$	$\neg m\neg c$	$\neg c$
Σ_{col}	m	$\neg m$	Σ

All 5 are null-invariant

Dataset	mc	$\neg mc$	$m\neg c$	$\neg m\neg c$	$AllConf$	$Jaccard$	$Cosine$	$Kulc$	$MaxConf$
D_1	10,000	1,000	1,000	100,000	0.91	0.83	0.91	0.91	0.91
D_2	10,000	1,000	1,000	100	0.91	0.83	0.91	0.91	0.91
D_3	100	1,000	1,000	100,000	0.09	0.05	0.09	0.09	0.09
D_4	1,000	1,000	1,000	100,000	0.5	0.33	0.5	0.5	0.5
D_5	1,000	100	10,000	100,000	0.09	0.09	0.29	0.5	0.91
D_6	1,000	10	100,000	100,000	0.01	0.01	0.10	0.5	0.99

Subtle: They disagree on most cases

Imbalance Ratio with Kulczynski Measure

Dataset	mc	$\neg mc$	$m\neg c$	$\neg m\neg c$	$AllConf$	$Jaccard$	$Cosine$	$Kulc$	$MaxConf$
D_1	10,000	1,000	1,000	100,000	0.91	0.83	0.91	0.91	0.91
D_2	10,000	1,000	1,000	100	0.91	0.83	0.91	0.91	0.91
D_3	100	1,000	1,000	100,000	0.09	0.05	0.09	0.09	0.09
D_4	1,000	1,000	1,000	100,000	0.5	0.33	0.5	0.5	0.5
D_5	1,000	100	10,000	100,000	0.09	0.09	0.29	0.5	0.91
D_6	1,000	10	100,000	100,000	0.01	0.01	0.10	0.5	0.99

- D_5 (and D_6) presents a "balanced" skewness:
  - The ratio of mc to c is greater than 0.9 (1000/1,100)
    - c occurs strongly suggest that m occurs
  - The ratio of mc to m is less than 0.1 (1000/11,000)
    - c is quite unlikely to occur due to the occurrence of m
- Diverse results:
  - All confidence & cosine measures view both cases as negatively associated.
  - The max confidence measure claims strong positive associations for these cases.
  - But the Kulc measure views both as neutral

Imbalance Ratio with Kulczynski Measure

- IR (Imbalance Ratio): measure the imbalance of two itemsets A and B in rule implications

$$IR(A,B) = \frac{|s(A) - s(B)|}{s(A) + s(B) - s(A \cup B)}$$

- Kulczynski and Imbalance Ratio (IR) together present a clear picture for all the three datasets D_4 through D_6
  - D_4 is neutral & balanced; D_5 is neutral but imbalanced; D_6 is neutral but very imbalanced
  - For such "balanced" skewness:
    - treat it as neutral and indicate its skewness using the imbalance ratio (IR)

Dataset	mc	$\neg mc$	$m\neg c$	$\neg m\neg c$	$Jaccard$	$Cosine$	$Kulc$	IR
D_1	10,000	1,000	1,000	100,000	0.83	0.91	0.91	0
D_2	10,000	1,000	1,000	100	0.83	0.91	0.91	0
D_3	100	1,000	1,000	100,000	0.05	0.09	0.09	0
D_4	1,000	1,000	1,000	100,000	0.33	0.5	0.5	0
D_5	1,000	100	10,000	100,000	0.09	0.29	0.5	0.89
D_6	1,000	10	100,000	100,000	0.01	0.10	0.5	0.99

Measures Selection

- For effective pattern evaluation, some key points can be referred to select the appropriate measure
- Null value cases are predominant in many large datasets
  - Neither milk nor coffee is in most of the baskets
  - neither Mike nor Jim is an author in most of the papers; etc.
- Null-invariance is an important property
- Lift, χ^2 and cosine are good measures if null transactions are not predominant
  - *Kulczynski + Imbalance Ratio* should be used to judge the interestingness of a pattern for large datasets with significant amount of null value cases

Summary

- **Basic Concepts**
  - Frequent Patterns, Association Rules
- **Frequent Itemset Mining Methods**
  - The Downward Closure Property and The Apriori Algorithm
  - Challenges and Improvements of Apriori
  - ECLAT & FP-tree
- **Pattern Interestingness Evaluation Measures**:
  - Interestingness Measures: Lift and χ^2
  - Null-Invariant Measures
  - Comparison of Interestingness Measures

References

- Jiawei Han and Micheline Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann Publishers, 2001 (ISBN:1-55860-489-8).