CS522 Advanced Database Systems
Mining Frequent Patterns

Chengyu Sun
California State University, Los Angeles

Sales Transactions

TID	Transactions
1	Beef, Chicken, Milk
2	Beef, Cheese
3	Cheese, Boots
4	Beef, Chicken, Cheese
5	Beef, Chicken, Clothes, Cheese, Milk
6	Chicken, Clothes, Milk
7	Chicken, Clothes, Milk
8	Beef, Milk

Support Count

- The support count, or frequency, of a itemset is the number of the transactions that contain the itemset
  - Item, Itemset, and Transaction
- Examples:
  - `support_count({beef})=5`
  - `support_count({beef,chicken,milk})=??`

Frequent Itemset

- An itemset is frequent if its support count is greater than or equals to a minimum support count threshold
  - `support_count(X)≥min_sup`

The Need for Closed Frequent Itemsets

- Two transactions
  - $<a_1, a_2, \ldots, a_{100}>$ and $<a_1, a_2, \ldots, a_{50}>$
- `min_sup=1`
- # of frequent itemsets??

Closed Frequent Itemset

- An itemset **X** is closed if there exists no *proper superset* of **X** that has the same support count
- A closed frequent itemset is an itemset that is both *closed* and *frequent*

Closed Frequent Itemset Example

- Two transactions
  - <$a_1, a_2, \ldots, a_{100}$> and <$a_1, a_2, \ldots, a_{50}$>
- `min_sup=1`
- Closed frequent itemset(s)??

Maximal Frequent Itemset

- An itemset **X** is a maximal frequent itemset if **X** is frequent and there exists no *proper superset* of **X** that is also frequent
- Example: if `{a,b,c}` is a maximal frequent itemset, which one of these *cannot* be a MFI
  - `{a,b,c,d}, {a,c}, {b,d}`

Maximal Frequent Itemset Example

- Two transactions
  - <$a_1, a_2, \ldots, a_{100}$> and <$a_1, a_2, \ldots, a_{50}$>
- `min_sup=1`
- Maximal frequent itemset(s)??
- Maximal frequent itemset vs. closed frequent itemset??

From Frequent Itemsets to Association Rules

- `{chicken,cheese}` is a frequent set
- `{chicken}`⇒`{cheese}`??
- Or is it `{cheese}`⇒`{chicken}`??

Association Rules

- **A⇒B**
  - **A** and **B** are itemsets
  - **A**∩**B**=∅

Support

- The support of **A⇒B** is the percentage of the transactions that contain **A**∪**B**

$$\text{support}(A \Rightarrow B) = P(A \cup B) = \frac{\text{support\_count}(A \cup B)}{|D|}$$

P(A∪B) is the probability that a transaction contains A∪B
D is the set of the transactions

Confidence

- The confidence of **A**⇒**B** is the percentage of the transactions containing **A** that also contains **B**

$$\text{confidence}(A \Rightarrow B) = P(B \mid A) = \frac{\text{support\_count}(A \cup B)}{\text{support\_count}(A)}$$

Support and Confidence Example

- `{chicken}⇒{cheese}`??
- `{cheese}⇒{chicken}`??

Strong Association Rule

- An association rule is strong if it satisfies both a minimum support threshold (`min_sup`) and a minimum confidence threshold (`min_conf`)
- Why do we need both *support* and *confidence*??

Association Rule Mining

- Find strong association rules
  - Find all frequent itemsets
  - Generate strong association rules from the frequent itemsets

The Apriori Property

- All nonempty subsets of a frequent itemset must also be frequent
- Or, if an itemset is not frequent, its supersets cannot be frequent either

Finding Frequent Itemsets – The Apriori Algorithm

- Given `min_sup`
- Find the frequent 1-itemsets L_1
- Find the the frequent k-itemsets L_k by joining the itemsets in L_{k-1}
- Stop when L_k is empty

Apriori Algorithm Example

beef	1
chicken	2
milk	3
cheese	4
boots	5
clothes	6

TID	Transactions
1	1, 2, 3
2	1, 4
3	4, 5
4	1, 2, 4
5	1, 2, 6, 4, 3
6	2, 6, 3
7	2, 6, 3
8	1, 3

- Support 25%

L_1

- Scan the data once to get the count of each item
- Remove the items that do not meet `min_sup`

C_1	support_count	L_1
{1}	5	{1}
{2}	5	{2}
{3}	5	{3}
{4}	4	{4}
{5}	1	
{6}	3	{6}

L_2

- $C_2=L_1 \times L_1$
- Scan the dataset again for the support_count of C_2, then remove non-frequent itemsets from C_2, i.e. $C_2 \rightarrow L_2$

C_2	support_count	L_2
{1,2}	3	{1,2}
{1,3}	3	{1,3}
{1,4}	3	{1,4}
{1,6}	1	
{2,3}	4	{2,3}
{2,4}	2	{2,4}
{2,6}	3	{2,6}
{3,4}	1	
{3,6}	3	{3,6}
{4,6}	1	

L_3

- ??

From L_{k-1} to C_k

- Let l_i be an itemset in L_{k-1}, and $l_i[j]$ be the jth item in l_i
- Items in an itemset are sorted, i.e. $l_i[1]<l_i[2]<\ldots<l_i[k-1]$
- l_1 and l_2 are joinable if
  - Their first k-2 items are the same, and
  - $l_1[k-1]<l_2[k-1]$

From C_k to L_k

- Reduce the size of C_k using the Apriori property
  - any (k-1)-subset of an candidate must be frequent, i.e. in L_{k-1}
- Scan the dataset to get the support counts

Generate Association Rules from Frequent Itemsets

- For each frequent itemset `l`, generate all nonempty subset of `l`
- For every nonempty subset of `s` of `l`, output rule `s⇒(l-s)` if `conf(s ⇒(l-s))≥min_conf`

Confidence-based Pruning …

- `conf({a,b}⇒{c,d})<min_conf`
  - `conf({a}⇒{c,d})??`
  - `conf({a,b,e}⇒{c,d})??`
  - `conf({a}⇒{b,c,d})??`

… Confidence-based Pruning

- If `conf(s⇒(l-s))<min_conf`, then `conf(s'⇒(l-s'))<min_conf` where `s'⊆s`.
- Example: `conf({a,b}⇒{c,d})<min_conf`
  - ??

Limitations of the Apriori Algorithm

- Multiple scans of the datasets
  - How many??
- Need to generate a large number of candidate sets

FP-Growth Algorithm

- Frequent-pattern Growth
- Mine frequent itemsets *without candidate generation*

FP-Growth Example

TID	Transactions
1	I1, I2, I5
2	I2, I4
3	I2, I3, I6
4	I1, I2, I4
5	I1, I3
6	I2, I3
7	I1, I3
8	I1, I2, I3, I5
9	I1, I2, I3

min_sup=2

L

- Scan the dataset and find the frequent 1-itemsets
- Sort the 1-itemsets by support count in descending order

L

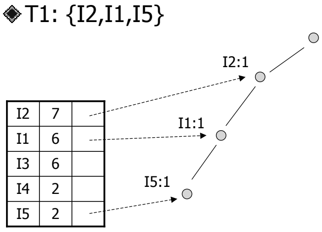
I2: 7
I1: 6
I3: 6
I4: 2
I5: 2

FP-tree

- Each transaction is processed in `L` order (why??) and becomes a branch in the FP tree
- Each node is linked from `L`

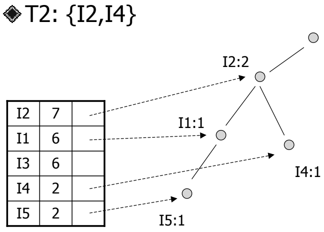
FP-tree Construction …

- T1: {I2,I1,I5}



… FP-tree Construction …

- T2: {I2,I4}



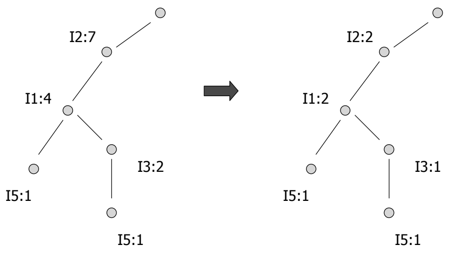
… FP-tree Construction

- ??

Mining the FP-tree

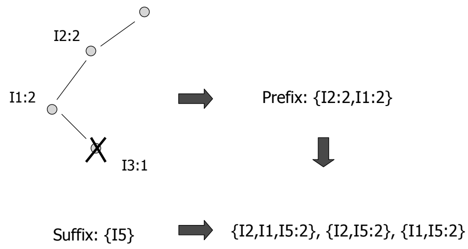
- For each item `i` in `L` (in ascending order), find the branch(s) in the FP tree that ends in `i`
- If there's only one branch, generate the frequent itemsets that end in `i`; otherwise run the tree mining algorithm recursively on the subtree

Mining the FP-tree – I5 ...



Conditional FP-tree for I5

... Mining the FP-tree – I5 ...

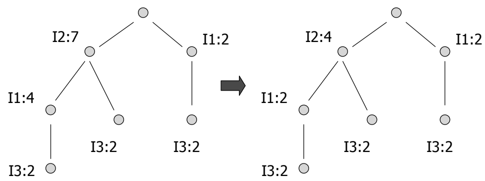


... Mining the FP-tree – I5

- All frequent patterns with suffix I5

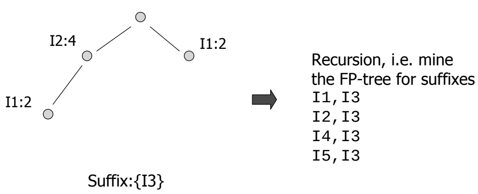
  {I2,I1,I5:2}, {I2,I5:2}, {I1,I5:2} and {I5:2}

Mining The FP-tree – I3 ...

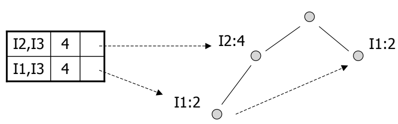


Conditional FP-tree for I3

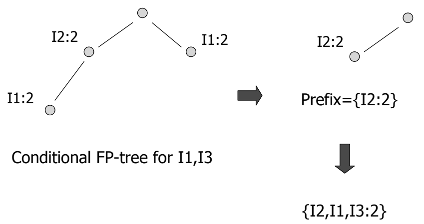
... Mining The FP-tree – I3 ...



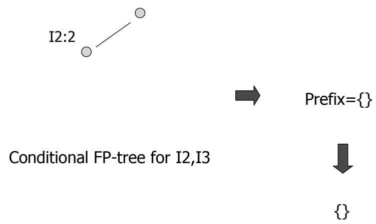
... Mining The FP-tree – I3 ...



... Mining The FP-tree – I3 ...



... Mining The FP-tree – I3 ...



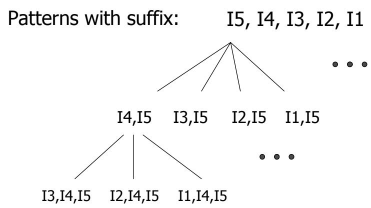
... Mining The FP-tree – I3

- All frequent patterns with suffix I3

  {I2,I1,I3:2}, and {I2,I3:4}, {I1,I3:4}, and {I3:6}

About FP-tree Mining

- A divide-and-conquer approach



Optimization Techniques

- Data partitioning
- Vertical data format
- Pruning conditions for mining closed frequent itemsets
  - Superset and subset checking
    - Pattern tree

Data Partitioning

- Divide dataset into n non-overlapping partitions such that *each partition fits into main memory*
- Find local frequent itemsets in each partition with `min_sup` (1 scan)
- All local frequent itemsets form a candidate set
  - *Does it include all global frequent itemsets??*
- Find global frequent itemsets from candidates (1 scan)

Vertical Data Format

Item	TID_set
I1	T1,T4,T5,T7,T8,T9
I2	T1,T2,T3,T4,T6,T8,T9
I3	T3,T5,T6,T7,T8,T9
I4	T2,T4
I5	T1,T8

- And how does it help??

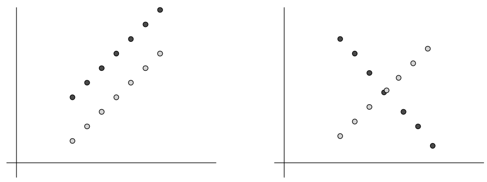
Strong Association Rules Could Be Misleading …

- Example:
  - 10,000 transactions
  - 6,000 transactions included games
  - 7,500 transactions included videos
  - 4,000 transactions included both
- {game} ⇒{video}
  - Support?? Confidence??

… Strong Association Rules Could Be Misleading

- Does buying game really imply buying video as well??

Correlation



Correlation Measures for Association Rules

- Lift
- χ^2
- All_confidence
- Cosine

Contingency Table

	game	!game	total
video	??	??	**??**
!video	??	??	**??**
total	**??**	**??**	**??**

Lift

$$lift(A,B) = \frac{P(A \cup B)}{P(A)P(B)}$$

- **A** and **B** are
  - Independent if `lift(A,B)=1`
  - Correlated if `lift(A,B)>1`
  - Negatively correlated if `lift(A,B)<1`
- `lift({game},{video})=??`

χ^2

- Two attributes A and B
  - A has r possible values
  - B has c possible values
- Event (A=a_i, B=b_j)
  - Observed frequency: o_{ij}
  - Expected frequency: e_{ij}=count(A=a_i)*count(B=b_j)/N

$$\chi^2 = \sum_{i=1}^{n}\sum_{j=1}^{m}\frac{(o_{ij}-e_{ij})^2}{e_{ij}}$$

χ^2 Example – Observed Frequency

	male	female	**total**
fiction	250	200	**450**
non-fiction	50	1000	**1050**
total	**300**	**1200**	**1500**

χ^2 Example – Expected Frequency

	male	female	**total**
fiction	??	??	**450**
non-fiction	??	??	**1050**
total	**300**	**1200**	**1500**

Contingency Table and χ^2

	male	female	**total**
fiction	250(90)	200(360)	**450**
non-fiction	50(210)	1000(840)	**1050**
total	**300**	**1200**	**1500**

$\chi^2=(250-90)^2/90+(50-210)^2/210+(200-360)^2/360+(1000-840)^2/840 =507.93$

χ^2 Test

- Tests the hypothesis that **A** and **B** are *independent*
- Degree of freedom **k**=(r-1)*(c-1)
- Significance probability level < 0.05
- χ^2 Test Table
  - E.g. http://www.itl.nist.gov/div898/handbook/eda/section3/eda3674.htm

All_confidence

◈ $\mathbf{X}=\{i_1,i_2,\ldots,i_k\}$

$$all\_conf(X)=\frac{\sup(X)}{\text{max\_item\_sup}(X)}=\frac{\sup(X)}{\max\{\sup(i_j)\mid\forall i_j\in X\}}$$

Example: `all_conf(game,video)=??`

About all_confidence

◈ `all_confidence(A,B)=??`

- If A and B are completely *positively* correlated
- If A and B are completely *negatively* correlated
- If A and B are independent

Cosine Measure

$$\text{cosine}(A,B)=\frac{P(A\cup B)}{\sqrt{P(A)\times P(B)}}=\frac{\sup(A\cup B)}{\sqrt{\sup(A)\times\sup(B)}}$$

Cosine vs. Lift

$$lift(A,B)=\frac{P(A\cup B)}{P(A)P(B)}=\frac{\frac{\sup(A\cup B)}{N}}{\frac{\sup(A)}{N}\frac{\sup(B)}{N}}=\frac{N\sup(A\cup B)}{\sup(A)\sup(B)}$$

$$\text{cosine}(A,B)=\frac{P(A\cup B)}{\sqrt{P(A)\times P(B)}}=\frac{\frac{\sup(A\cup B)}{N}}{\sqrt{\frac{\sup(A)\sup(B)}{N^2}}}=\frac{\sup(A\cup B)}{\sqrt{\sup(A)\sup(B)}}$$

Correlation Measures Recap

	lift	χ^2	all_conf	cosine
c.p.c				
c.n.c				
i				

c.p.c – completely positively correlated
c.n.c – completed negatively correlated
i – independent of each other

Choosing Correlation Measures …

datasets	mc	m′c	mc′	m′c′	all_conf	cosine	lift	χ^2
A_1	1,000	100	100	100,000	0.91	0.91	83.64	83,452.6
A_2	1,000	100	100	10,000	0.91	0.91	9.26	9,055.7
A_3	1,000	100	100	1,000	0.91	0.91	1.82	1,472.7
A_4	1,000	100	100	0	0.91	0.91	0.99	9.9
B	1,000	1,000	1,000	1,000	0.50	0.50	1.00	0.0

mc: # of transactions that contain both milk and coffee
m′c′: # of transactions that contain neither milk nor coffee

… Choosing Correlation Measures

- `all_confidence` and `cosine` are null-invariant, while `lift` and χ^2 are not
- `all_confidence` has the Apriori property
- `all_confidence` and `cosine` should be augmented with other measures when the result is not conclusive

Mining Sequential Patterns

- <{computer},{printer},{printer cartridge}>
- <{bread,milk},{bread,milk},{bread,milk }…>
- <{home.jsp},{search.jsp},{product.jsp} ,{product.jsp},{search.jsp}…>

Terminology and Notations

- Item, itemset
- Event = itemset
- A sequence is an ordered list of events
  - $<e_1e_2e_3\ldots e_l>$
  - E.g. <(a)(abc)(bc)(d)(ac)(f)>
- The length of a sequence is the number of items in the sequence, i.e. *not the number of events*

Sequences vs. Itemsets

- {a,b,c}
  - # of 3-itemset(s)??
  - # of 3-sequence(s)??

Subsequence

- $A=<a_1a_2a_3\ldots a_n>$
- $B=<b_1b_2b_3\ldots b_m>$
- A is a *subsequence* of B if there exists $1 \le j_1 < j_2 < \ldots < j_n \le m$ such that $a_1 \subseteq b_{j1}, a_2 \subseteq b_{j2}, \ldots, a_n \subseteq b_{jn}$

Subsequence Example

- `s=<(abc)(de)(f)>`
- What are the subsequences of `s`??

Sequential Pattern

- If A is a subsequence of B, we say B *contains* A
- The support count of A is the number of sequences that contain A
- A is *frequent* if `support_count(A)≥min_sup`
- A frequent sequence is called a sequential pattern

Apriori Property Again

- Every nonempty subsequence of a frequent sequence is frequent

GSP Algorithm

- *Generalized Sequential Patterns*
- An extension of the Apriori algorithm for mining sequential patterns

GSP Example

SID	Sequence
1	<(a)(ab)(a)>
2	<(a)(c)(bc)>
3	<(ab)(c)(b)>
4	<(a)(c)(c)>

min_sup=2

L_1

C_1	support_count	L_1
a	4	<(a)>
b	3	<(b)>
c	3	<(c)>

L_2

C_2	support_count	L_2
<(a)(a)>	1	
<(a)(b)>	3	<(a)(b)>
<(a)(c)>	3	<(a)(c)>
<(b)(a)>	1	
<(b)(b)>	1	
<(b)(c)>	1	
<(c)(a)>	0	
<(c)(b)>	2	<(c)(b)>
<(c)(c)>	2	<(c)(c)>
<(ab)>	2	<(ab)>
<(ac)>	0	
<(bc)>	1	

From L_{k-1} to C_k

- Two sequences s_1 and s_2 are joinable if the subsequence obtained by dropping the first item in s_1 is the same as the subsequence obtained by dropping the last item in s_2
- The joined sequence is s_1 concatenated with the last item `i` of s_2
  - If the last two items in s_2 are in the same event, `i` is merged into the last event of s_1;
  - Otherwise `i` becomes a separate event

L3

C_3	support_count	L_3
<(a)(c)(b)>	2	<(a)(c)(b)>
<(a)(c)(c)>	2	<(a)(c)(c)>

Candidate Pruning

- A k-sequence can be pruned if one of its (k-1)-subsequence is not frequent

L_3	Candidate generation → C_4	Pruning → C_4
<(1)(2)(3)>	<(1)(2)(3)(4)>	<(1)(2 5)(3)>
<(1)(2 5)>	<(1)(2 5)(3)>	
<(1)(5)(3)>	<(1)(5)(3 4)>	
<(2)(3)(4)>	<(2)(3)(4)(5)>	
<(2 5)(3)>	<(2 5)(3 4)>	
<(3)(4)(5)>		
<(5)(3 4)>		

Summary

- Frequent itemsets, association rules, sequential patterns
  - Measures: support, confidence, correlation
  - Algorithms: Apriori, FP-Growth, rule generation, GPS
  - Optimizations: partitioning, vertical data format, various pruning techniques