Foundation of
Data Science and Analytics :

Association Mining (Frequent Pattern Analysis) - Basic Concepts

Dr. Arun K. Timalsina

Materials Adaptation :

Jiawei Han, Micheline Kamber, and Jian Pei, **Data Mining: Concepts and Techniques**, 3rd Edition, Morgan Kaufmann, 2011.

Mining Frequent Patterns, Association and Correlations

- Basic Concepts
- Frequent Itemset Mining Methods
- Which Patterns Are Interesting?—Pattern Evaluation Methods
- Summary

What Is Frequent Pattern Analysis?

- **Frequent pattern**: a pattern (a set of items, subsequences, substructures, etc.) that occurs frequently in a data set
- First proposed by Agrawal, Imielinski, and Swami [AIS**93**] in the context of frequent itemsets and association rule mining
- Motivation: Finding inherent regularities in data
  - What products were often purchased together?— Beer and diapers?!
  - 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?
- Applications
  - Basket data analysis, cross-marketing, catalog design, sale campaign analysis, Web log (click stream) analysis, and DNA sequence analysis.

Why Is Freq. Pattern Mining Important?

- Freq. pattern: An intrinsic and important property of datasets
- Foundation for many essential data mining tasks
  - Association, correlation, and causality analysis
  - Sequential, structural (e.g., sub-graph) patterns
  - Pattern analysis in spatiotemporal, multimedia, time-series, and stream data
  - Classification: discriminative, frequent pattern analysis
  - Cluster analysis: frequent pattern-based clustering
  - Data warehousing: iceberg cube and cube-gradient
  - Semantic data compression: fascicles
  - Broad applications

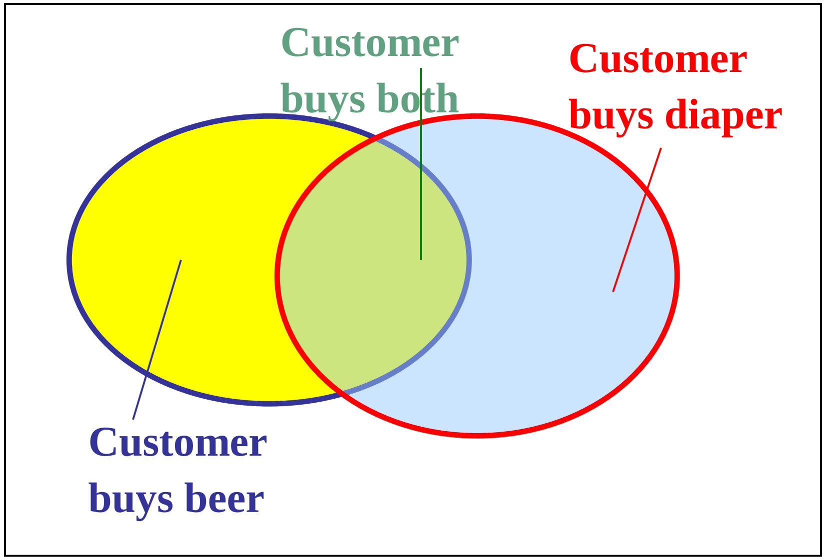
Market Basket Example

- ? Where should detergents be placed in the Store to maximize their sales?
- ? Are window cleaning products purchased when detergents and orange juice are bought together?
- ? Is soda typically purchased with bananas? Does the brand of soda make a difference?
- ? How are the demographics of the neighborhood affecting what customers are buying?

Image source: deepclimate.org

Basic Concepts: Frequent Patterns

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



- itemset: A set of one or more items
- k-itemset X = $\{x_1, \ldots, x_k\}$
- *(absolute) support*, or, *support count* of X: Frequency or occurrence of an itemset X
- *(relative) support*, *s*, is the fraction of transactions that contains X (i.e., the probability that a transaction contains X)
- An itemset X is *frequent* if X's support is no less than a *minsup* threshold

$$Rule:\ X \Rightarrow Y$$

$$Support = \frac{frq(X,Y)}{N}$$

$$Confidence = \frac{frq(X,Y)}{frq(X)}$$

$$Lift = \frac{Support}{Supp(X) \times Supp(Y)}$$

$$Rule:\ X \Rightarrow Y$$

$$Support = \frac{frq(X,Y)}{N}$$

$$Confidence = \frac{frq(X,Y)}{frq(X)}$$

$$Lift = \frac{Support}{Supp(X) \times Supp(Y)}$$

Transactions

$$Rule:\ X \Rightarrow Y$$

$$Support = \frac{frq(X,Y)}{N}$$

$$Confidence = \frac{frq(X,Y)}{frq(X)}$$

$$Lift = \frac{Support}{Supp(X) \times Supp(Y)}$$

Transactions →

Calculated values of different measures →

Rule	Support	Confidence	Lift
$A \Rightarrow D$	2/5	2/3	10/9
$C \Rightarrow A$			
$A \Rightarrow C$			
$B \& C \Rightarrow D$			

$$Rule:\ X \Rightarrow Y$$

$$Support = \frac{frq(X,Y)}{N}$$

$$Confidence = \frac{frq(X,Y)}{frq(X)}$$

$$Lift = \frac{Support}{Supp(X) \times Supp(Y)}$$

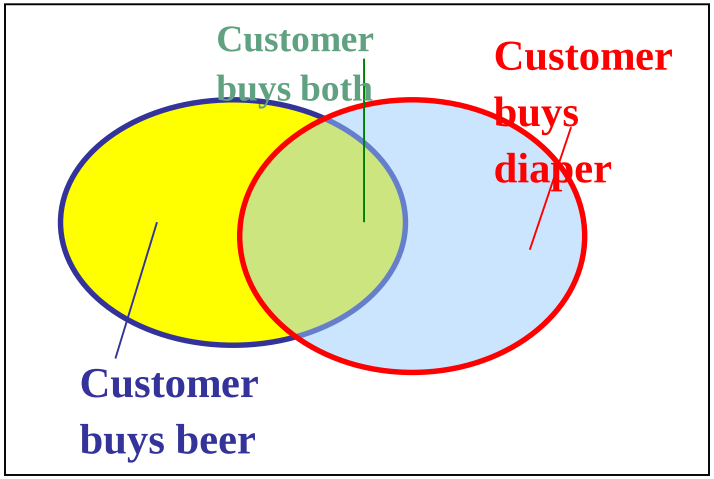
Transactions →

Calculated values of different measures →

Rule	Support	Confidence	Lift
$A \Rightarrow D$	2/5	2/3	10/9
$C \Rightarrow A$	2/5	2/4	5/6
$A \Rightarrow C$	2/5	2/3	5/6
$B \,\&\, C \Rightarrow D$	1/5	1/3	5/9

Basic Concepts: Association Rules

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



- Find all the rules *X* → *Y* with minimum support and confidence
  - support, *s*, probability that a transaction contains X ∪ Y
  - confidence, *c*, conditional probability that a transaction having X also contains *Y*

Let minsup = 50%, minconf = 50%

Freq. Pat.: Beer:3, Nuts:3, Diaper:4, Eggs:3, {Beer, Diaper}:3

- Association rules: *X* → *Y* (***s***, ***c***)
  - *Beer* → *Diaper* (60%, 100%)
  - *Diaper* → *Beer* (60%, 75%)

Closed Patterns and Max-Patterns

- A long pattern contains a combinatorial number of sub-patterns, e.g., {a_1, …, a_{100}} contains $({}_{100}{}^{1}) + ({}_{100}{}^{2}) + \ldots + ({}_{1}{}^{1}{}_{0}{}^{0}{}_{0}{}^{0}) = 2^{100} - 1 = 1.27*10^{30}$ sub-patterns!
- Solution: *Mine closed patterns and max-patterns instead*
- An itemset X is closed if X is *frequent* and there exists *no super-pattern* Y ɔ X, *with the same support* as X (proposed by Pasquier, et al. @ ICDT'99)
- An itemset X is a max-pattern if X is frequent and there exists no frequent super-pattern Y ɔ X (proposed by Bayardo @ SIGMOD'98)
- Closed pattern is a lossless compression of freq. patterns
  - Reducing the # of patterns and rules

Closed Patterns and Max-Patterns

- Exercise. DB = $\{<a_1, \ldots, a_{100}>, < a_1, \ldots, a_{50}>\}$
  - Min_sup = 1.
- What is the set of closed itemset?
  - $<a_1, \ldots, a_{100}>$: 1
  - $< a_1, \ldots, a_{50}>$: 2
- What is the set of max-pattern?
  - $<a_1, \ldots, a_{100}>$: 1
- What is the set of all patterns?
  - !!

Computational Complexity of Frequent Itemset Mining

- How many itemsets are potentially to be generated in the worst case?
  - The number of frequent itemsets to be generated is sensitive to the *minsup* threshold
  - When *minsup* is low, there exist potentially an exponential number of frequent itemsets
  - The worst case: If all rules exist then 2^N from powerset formula otherwise sum of combinatorial formula C(N, R); approximation : N^R where N: # distinct items, and R: max length of transactions, M,N both large

Computational Complexity of Frequent Itemset Mining

- How many itemsets are potentially to be generated in the worst case?
  - The number of frequent itemsets to be generated is sensitive to the *minsup* threshold
  - When *minsup* is low, there exist potentially an exponential number of frequent itemsets
  - The worst case: M^N where M: # distinct items, and N: max length of transactions
- The worst case complexity vs. the expected probability
  - Ex. Suppose BhatBhateni has 10^4 kinds of products
    - The chance to pick up one product 10^{-4}
    - The chance to pick up a particular set of 10 products: $\sim 10^{-40}$
    - What is the chance this particular set of 10 products to be frequent 10^3 times in 10^9 transactions?

Mining Frequent Patterns, Association and Correlations: Basic Concepts and Methods

- Basic Concepts
- Frequent Itemset Mining Methods
- Which Patterns Are Interesting?—Pattern Evaluation Methods
- Summary

Scalable Frequent Itemset Mining Methods

- Apriori: A Candidate Generation-and-Test Approach
- Improving the Efficiency of Apriori
- FPGrowth: A Frequent Pattern-Growth Approach

The Downward Closure Property and Scalable Mining Methods

- The downward closure property of frequent patterns
  - Any subset of a frequent itemset must be frequent
  - If **{beer, diaper, nuts}** is frequent,

    so is **{beer, diaper}**

    : i.e., every transaction having

    {beer, diaper, nuts} also contains {beer, diaper}
- Scalable mining methods: Three major approaches
  - **Apriori** (Agrawal & Srikant@VLDB'94)
  - **Frequent pattern growth**(FPgrowth—Han,Pei&Yin @SIGMOD'00)
  - **Vertical data format approach**(Charm—Zaki & Hsiao @SDM'02)

Apriori: A Candidate Generation & Test Approach

- Apriori pruning principle: If there is any itemset which is infrequent, its superset should not be generated/tested! (Agrawal & Srikant @VLDB'94, Mannila, et al. @ KDD' 94)
- Method:
  - Initially, scan DB once to get frequent 1-itemset
  - Generate length (k+1) candidate itemsets from length k frequent itemsets
  - Test the candidates against DB
  - Terminate when no frequent or candidate set can be generated

The Apriori Algorithm—An Example

Database TDB

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

Threshold : $Sup_{min} = 2$

1st scan →

C_1

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

The Apriori Algorithm—An Example

Database TDB

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

Threshold : $\text{Sup}_{\text{min}} = 2$

1st scan →

C_1

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

The Apriori Algorithm—An Example

Database TDB

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

Threshold : $Sup_{min} = 2$

1st scan →

C_1

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

→ L_1

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

The Apriori Algorithm—An Example

Database TDB

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

Threshold : $Sup_{min} = 2$

1st scan →

C_1

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

→ L_1

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

The Apriori Algorithm—An Example

$Sup_{min} = 2$

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

L_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}

The Apriori Algorithm—An Example

$Sup_{min} = 2$

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

L_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}

2nd scan

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

The Apriori Algorithm—An Example

$Sup_{min} = 2$

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

L_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}

2nd scan

C_2

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

The Apriori Algorithm—An Example

$Sup_{min} = 2$

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

L_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}

2nd scan

C_2

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

L_2

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

The Apriori Algorithm—An Example

$Sup_{min} = 2$

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

L_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}

2nd scan

C_2

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

L_2

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

C_3

Itemset
{B, C, E}

The Apriori Algorithm—An Example

$Sup_{min} = 2$

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

L_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}

2nd scan

C_2

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

L_2

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

C_3

Itemset
{B, C, E}

3rd scan

L_3

Itemset	sup
{B, C, E}	2

The Apriori Algorithm (Pseudo-Code)

C_k: Candidate itemset of size k

L_k : frequent itemset of size k

```
$L_1$ = {frequent items};
for ($k$ = 1; $L_k$ !=∅; $k$++) 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 $\cup_k L_k$;
```

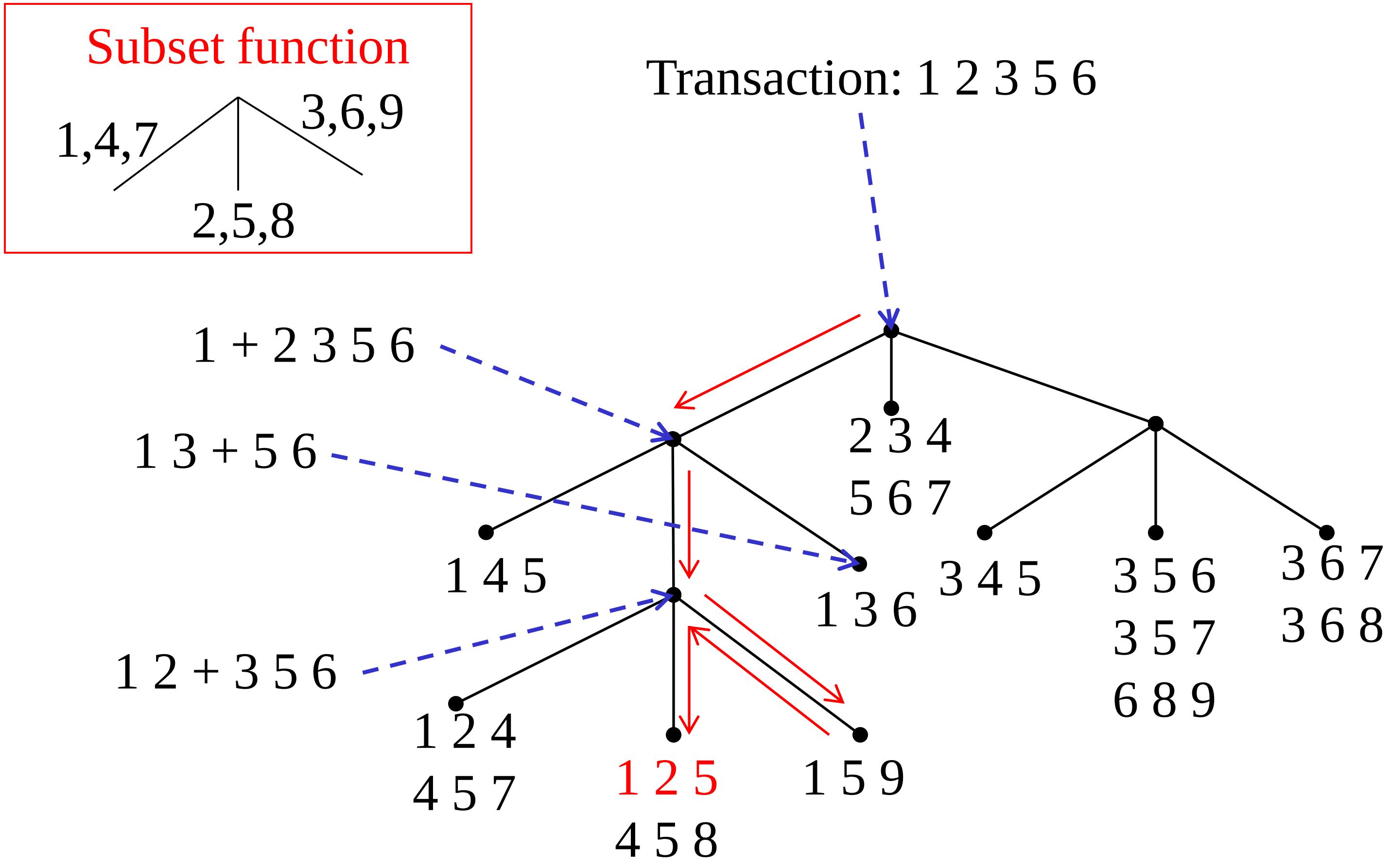
Implementation of Apriori

- How to generate candidates?
  - Step 1: self-joining L_k
  - Step 2: pruning
- Example of Candidate-generation
  - L_3={*abc, abd, acd, ace, bcd*}
  - Self-joining: L_3*L_3
    - *abcd* from *abc* and *abd*
    - *acde* from *acd* and *ace*
  - Pruning:
    - *acde* is removed because *ade* is not in L_3
  - C_4 = {*abcd*}

How to Count Supports of Candidates?

- Why counting supports of candidates a problem?
  - The total number of candidates can be very huge
  - One transaction may contain many candidates
- Method:
  - Candidate itemsets are stored in a *hash-tree*
  - *Leaf* node of hash-tree contains a list of itemsets and counts
  - *Interior* node contains a hash table
  - *Subset function*: finds all the candidates contained in a transaction

Counting Supports of Candidates Using Hash Tree



Subset function

1,4,7

2,5,8

3,6,9

Transaction: 1 2 3 5 6

1 + 2 3 5 6

1 3 + 5 6

1 2 + 3 5 6

2 3 4
5 6 7

1 4 5

1 3 6

3 4 5

3 5 6
3 5 7
6 8 9

3 6 7
3 6 8

1 2 4
4 5 7

1 2 5
4 5 8

1 5 9

Candidate Generation: An SQL Implementation

- SQL Implementation of candidate generation
  - Suppose the items in L_{k-1} are listed in an order
  - Step 1: self-joining L_{k-1}

    ```
    insert into $C_k$
    select $p.item_1, p.item_2, \ldots, p.item_{k-1}, q.item_{k-1}$
    from $L_{k-1}$ p, $L_{k-1}$ q
    where $p.item_1=q.item_1, \ldots, p.item_{k-2}=q.item_{k-2}, p.item_{k-1} < q.item_{k-1}$
    ```

  - Step 2: pruning

    ```
    forall itemsets c in $C_k$ do
        forall (k-1)-subsets s of c do
            if (s is not in $L_{k-1}$) then delete c from $C_k$
    ```

- Use object-relational extensions like UDFs, BLOBs, and Table functions for efficient implementation [See: S. Sarawagi, S. Thomas, and R. Agrawal. Integrating association rule mining with relational database systems: Alternatives and implications. SIGMOD'98]

The Apriori Algorithm—An Example

$Sup_{min} = 2$

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

L_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}

2nd scan

C_2

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

L_2

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

C_3

Itemset
{B, C, E}

3rd scan

L_3

Itemset	sup
{B, C, E}	2

2019 –Exam Question

Here is the sample transaction data records of famous grocery store, TTP (Taja Tarkari Pasal on a particular morning of a day. Considering minimum support 40% and minimum confidence of 50%, answer the followings:

A	Aaloo
B	Bandaa
C	Chamsur
D	Dhaniya
G	Ghiraunla
K	Kaauli
M	Mula
P	Paalungo
R	RaayoSaag
T	Tamaatar

T1	K, A, T, D
T2	A,B,D
T3	C, P, M
T4	A,G,K,T
T5	R,M,A,B
T6	A,B,C,D,P
T7	K,M,A,D,T
T8	R,C,P,D
T9	B,R.D.K,T
T10	T,K,M,A

(a) Using Apriori algorithm, identify the list of frequent items. (You need to show all the steps of calculations.)

(b) Find out all strong association rules of TTP transaction (i.e. X ^ Y ➔ Z).

L_1

C_2

L_2

L_3

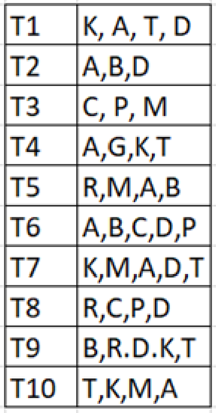
C_1

A	7	K	5
B	4	M	4
C	3	P	3
D	6	R	3
G	1	T	5

2019 –Exam Question

Here is the sample transaction data records of famous grocery store, TTP (Taja Tarkari Pasal) on a particular morning of a day. Considering minimum support 40% and minimum confidence of 50%, answer the followings:

A	Aaloo
B	Bandaa
C	Chamsur
D	Dhaniya
G	Ghiraunla
K	Kaauli
M	Mula
P	Paalungo
R	RaayoSaag
T	Tamaatar



T1	K, A, T, D
T2	A,B,D
T3	C, P, M
T4	A,G,K,T
T5	R,M,A,B
T6	A,B,C,D,P
T7	K,M,A,D,T
T8	R,C,P,D
T9	B,R.D.K,T
T10	T,K,M,A

(a) Using Apriori algorithm, identify the list of frequent items. (You need to show all the steps of calculations.)

(b) Find out all strong association rules of TTP transaction (i.e. X ^ Y ➔ Z).

L_1 C_2 L_2 L_3

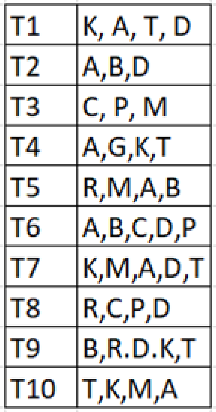
A	7	K	5
B	4	M	4
C	3	P	3
D	6	R	3
G	1	T	5

A	7
B	4
D	6
K	5
M	4
T	5

2019 –Exam Question

Here is the sample transaction data records of famous grocery store, TTP (Taja Tarkari Pasal) on a particular morning of a day. Considering minimum support 40% and minimum confidence of 50%, answer the followings:

A	Aaloo
B	Bandaa
C	Chamsur
D	Dhaniya
G	Ghiraunla
K	Kaauli
M	Mula
P	Paalungo
R	RaayoSaag
T	Tamaatar



T1	K, A, T, D
T2	A,B,D
T3	C, P, M
T4	A,G,K,T
T5	R,M,A,B
T6	A,B,C,D,P
T7	K,M,A,D,T
T8	R,C,P,D
T9	B,R.D.K,T
T10	T,K,M,A

(a) Using Apriori algorithm, identify the list of frequent items. (You need to show all the steps of calculations.)

(b) Find out all strong association rules of TTP transaction (i.e. X ^ Y ➔ Z).

A	7	K	5
B	4	M	4
C	3	P	3
D	6	R	3
G	1	T	5

L_1

A	7
B	4
D	6
K	5
M	4
T	5

C_2

L_2

AD	4
AK	4
AT	4
KT	5

L_3

2019 –Exam Question

Here is the sample transaction data records of famous grocery store, TTP (Taja Tarkari Pasal) on a particular morning of a day. Considering minimum support 40% and minimum confidence of 50%, answer the followings:

A	Aaloo
B	Bandaa
C	Chamsur
D	Dhaniya
G	Ghiraunla
K	Kaauli
M	Mula
P	Paalungo
R	RaayoSaag
T	Tamaatar

T1	K, A, T, D
T2	A,B,D
T3	C, P, M
T4	A,G,K,T
T5	R,M,A,B
T6	A,B,C,D,P
T7	K,M,A,D,T
T8	R,C,P,D
T9	B,R.D.K,T
T10	T,K,M,A

(a) Using Apriori algorithm, identify the list of frequent items. (You need to show all the steps of calculations.)

(b) Find out all strong association rules of TTP transaction (i.e. X ^ Y ➔ Z).

A	7	K	5
B	4	M	4
C	3	P	3
D	6	R	3
G	1	T	5

L_1

A	7
B	4
D	6
K	5
M	4
T	5

C_2

L_2

AD	4
AK	4
AT	4
KT	5

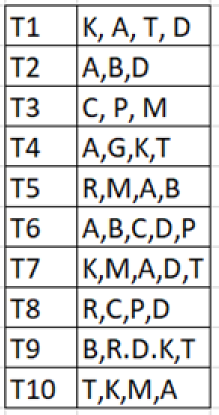
L_3

AKT	4

2019 –Exam Question

Here is the sample transaction data records of famous grocery store, TTP (Taja Tarkari Pasal) on a particular morning of a day. Considering minimum support 40% and minimum confidence of 50%, answer the followings:

A	Aaloo
B	Bandaa
C	Chamsur
D	Dhaniya
G	Ghiraunla
K	Kaauli
M	Mula
P	Paalungo
R	RaayoSaag
T	Tamaatar



T1	K, A, T, D
T2	A,B,D
T3	C, P, M
T4	A,G,K,T
T5	R,M,A,B
T6	A,B,C,D,P
T7	K,M,A,D,T
T8	R,C,P,D
T9	B,R.D.K,T
T10	T,K,M,A

(a) Using Apriori algorithm, identify the list of frequent items. (You need to show all the steps of calculations.)

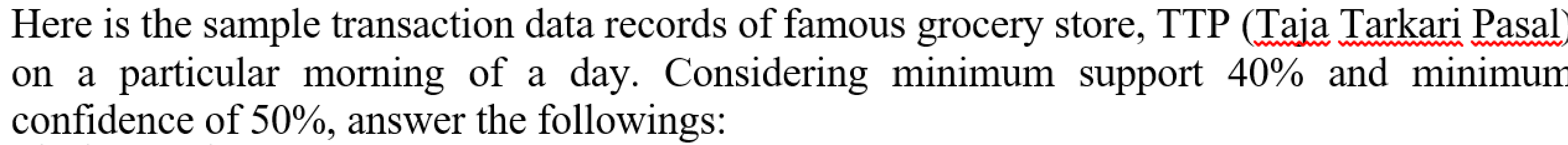
(b) Find out all strong association rules of TTP transaction (i.e. X ^ Y ➔ Z).

L_3

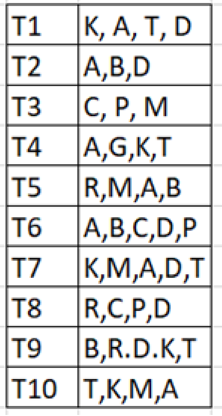
AKT	4

A^K ➔ T
A^T ➔ K
K^T ➔ A

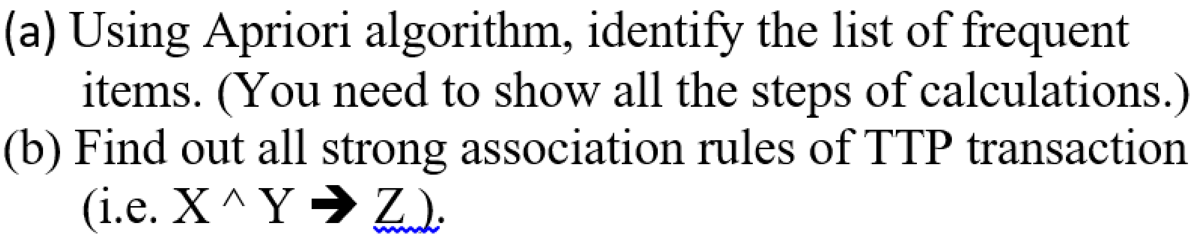
2019 –Exam Question


Here is the sample transaction data records of famous grocery store, TTP (Taja Tarkari Pasal) on a particular morning of a day. Considering minimum support 40% and minimum confidence of 50%, answer the followings:

A	Aaloo
B	Bandaa
C	Chamsur
D	Dhaniya
G	Ghiraunla
K	Kaauli
M	Mula
P	Paalungo
R	RaayoSaag
T	Tamaatar



T1	K, A, T, D
T2	A,B,D
T3	C, P, M
T4	A,G,K,T
T5	R,M,A,B
T6	A,B,C,D,P
T7	K,M,A,D,T
T8	R,C,P,D
T9	B,R.D.K,T
T10	T,K,M,A


(a) Using Apriori algorithm, identify the list of frequent items. (You need to show all the steps of calculations.)

(b) Find out all strong association rules of TTP transaction (i.e. X ^ Y ➔ Z).

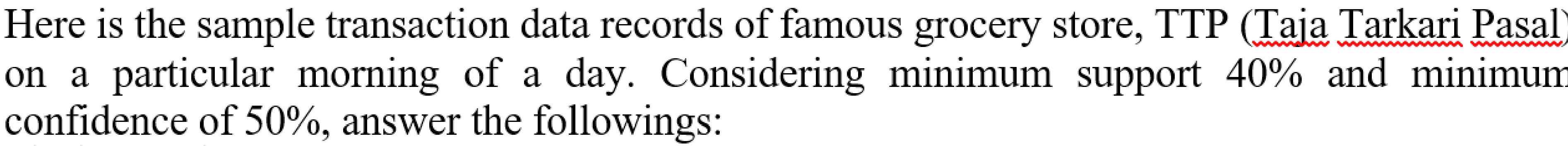
L_3

AKT	4

A^K ➔ T : s (A^K^T) ; s (A^K^T) / s(A^K)

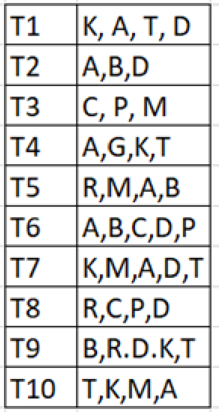
S (A^K^T) = 4
S (A^K) = 4
c

2019 –Exam Question

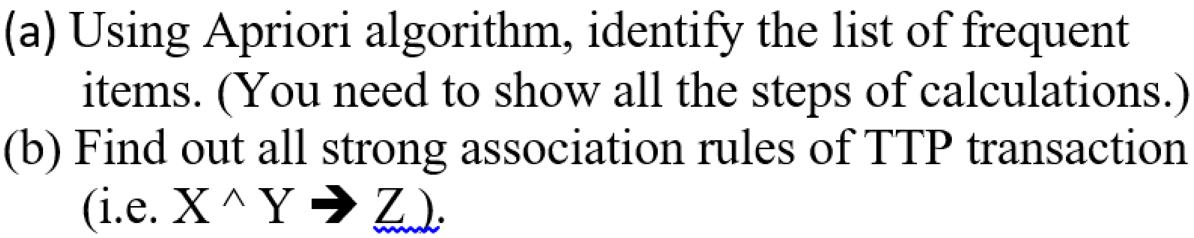


Here is the sample transaction data records of famous grocery store, TTP (Taja Tarkari Pasal) on a particular morning of a day. Considering minimum support 40% and minimum confidence of 50%, answer the followings:

A	Aaloo
B	Bandaa
C	Chamsur
D	Dhaniya
G	Ghiraunla
K	Kaauli
M	Mula
P	Paalungo
R	RaayoSaag
T	Tamaatar



T1	K, A, T, D
T2	A,B,D
T3	C, P, M
T4	A,G,K,T
T5	R,M,A,B
T6	A,B,C,D,P
T7	K,M,A,D,T
T8	R,C,P,D
T9	B,R.D.K,T
T10	T,K,M,A



(a) Using Apriori algorithm, identify the list of frequent items. (You need to show all the steps of calculations.)

(b) Find out all strong association rules of TTP transaction (i.e. X ^ Y ➔ Z).

L_3

AKT	4

A^K ➔ T (s, c) = (0.4, 1)

A^T ➔ K (s, c) = (0.4, 1)

K^T ➔ A (s, c) = (0.4, 0.8)

2019 –Exam Question

Here is the sample transaction data records of famous grocery store, TTP (Taja Tarkari Pasal) on a particular morning of a day. Considering minimum support 40% and minimum confidence of 50%, answer the followings:

A	Aaloo
B	Bandaa
C	Chamsur
D	Dhaniya
G	Ghiraunla
K	Kaauli
M	Mula
P	Paalungo
R	RaayoSaag
T	Tamaatar

T1	K, A, T, D
T2	A,B,D
T3	C, P, M
T4	A,G,K,T
T5	R,M,A,B
T6	A,B,C,D,P
T7	K,M,A,D,T
T8	R,C,P,D
T9	B,R.D.K,T
T10	T,K,M,A

(a) Using Apriori algorithm, identify the list of frequent items. (You need to show all the steps of calculations.)

(b) Find out all strong association rules of TTP transaction (i.e. X ^ Y ➔ Z).

L_3

AKT	4

A^K ➔ T (s, c) = (0.4, 1)

A^T ➔ K (s, c) = (0.4, 1)

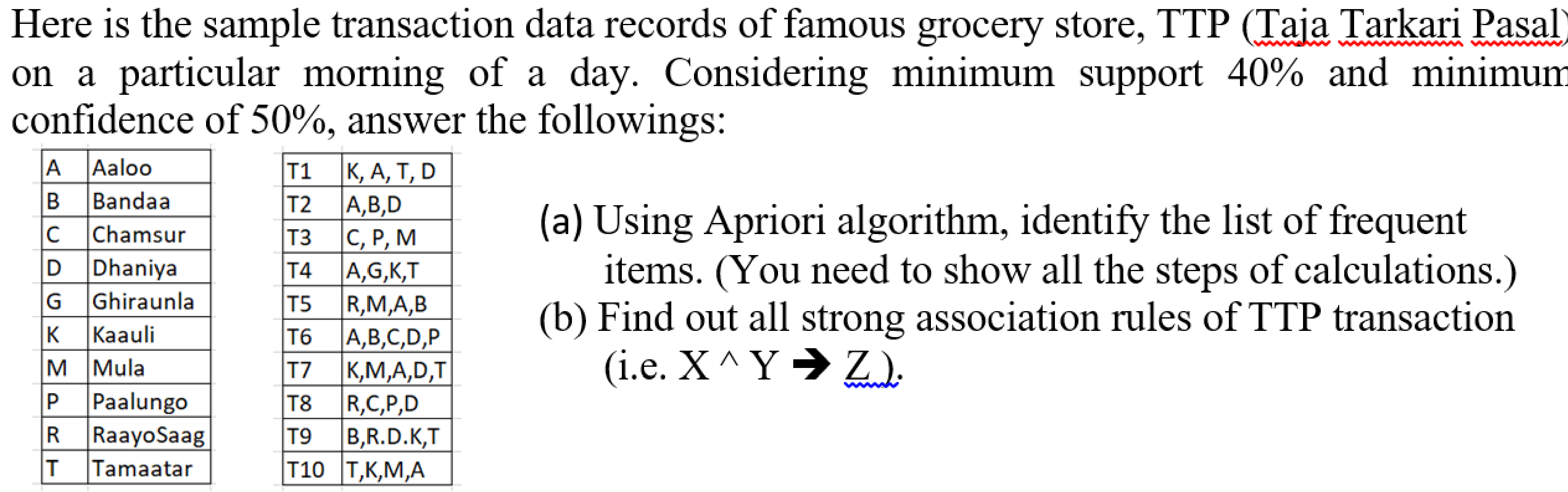
K^T ➔ A (s, c) = (0.4, 0.8)

A^K ➔ T

S = p(A^K^T) = 4/10 = 0.4

C = p(A^K^T) / p(A^K) = 0.4/0.4 = 1

2019 –Exam Question



Here is the sample transaction data records of famous grocery store, TTP (Taja Tarkari Pasal) on a particular morning of a day. Considering minimum support 40% and minimum confidence of 50%, answer the followings:

A	Aaloo
B	Bandaa
C	Chamsur
D	Dhaniya
G	Ghiraunla
K	Kaauli
M	Mula
P	Paalungo
R	RaayoSaag
T	Tamaatar

T1	K, A, T, D
T2	A,B,D
T3	C, P, M
T4	A,G,K,T
T5	R,M,A,B
T6	A,B,C,D,P
T7	K,M,A,D,T
T8	R,C,P,D
T9	B,R.D.K,T
T10	T,K,M,A

(a) Using Apriori algorithm, identify the list of frequent items. (You need to show all the steps of calculations.)

(b) Find out all strong association rules of TTP transaction (i.e. X ^ Y ➔ Z).

c) If the support threshold is decreased to 30% and confidence threshold is decreased to 40%, will there be any association rules to be added?

Scalable Frequent Itemset Mining Methods

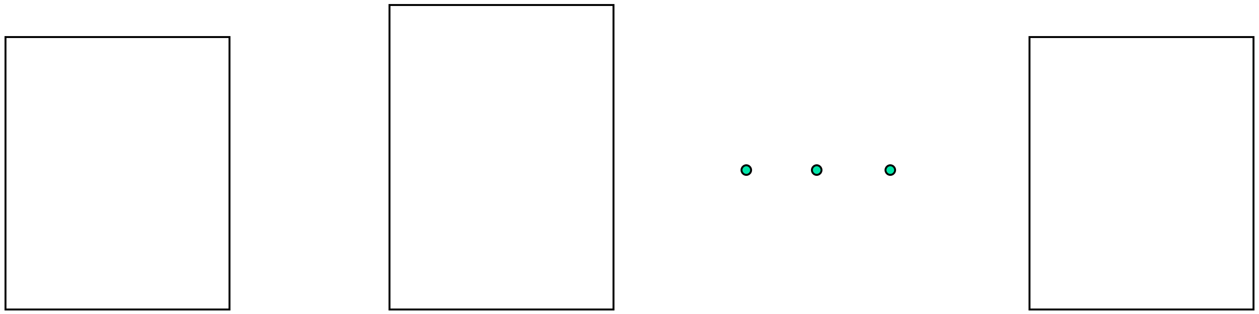
- Apriori: A Candidate Generation-and-Test Approach
- Improving the Efficiency of Apriori
- FPGrowth: A Frequent Pattern-Growth Approach

Further Improvement of the Apriori Method

- Major computational 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

Partition: Scan Database Only Twice

- Any itemset that is potentially frequent in DB must be frequent in at least one of the partitions of DB
  - Scan 1: partition database and find local frequent patterns
  - Scan 2: consolidate global frequent patterns
- A. Savasere, E. Omiecinski and S. Navathe, *VLDB'95*



$DB_1 \quad + \quad DB_2 \quad + \quad \cdots \quad + \quad DB_k \quad = \quad DB$

$sup_1(i) < \sigma DB_1 \qquad sup_2(i) < \sigma DB_2 \qquad sup_k(i) < \sigma DB_k \qquad sup(i) < \sigma DB$

DHP: Reduce the Number of Candidates

- A *k*-itemset whose corresponding hashing bucket count is below the threshold cannot be frequent
  - Candidates: a, b, c, d, e
  - Hash entries
    - {ab, ad, ae}
    - {bd, be, de} ....
  - Frequent 1-itemset: a, b, d, e
  - ab is not a candidate 2-itemset if the sum of count of {ab, ad, ae} is below support threshold
- J. Park, M. Chen, and P. Yu. An effective hash-based algorithm for mining association rules. *SIGMOD'95*

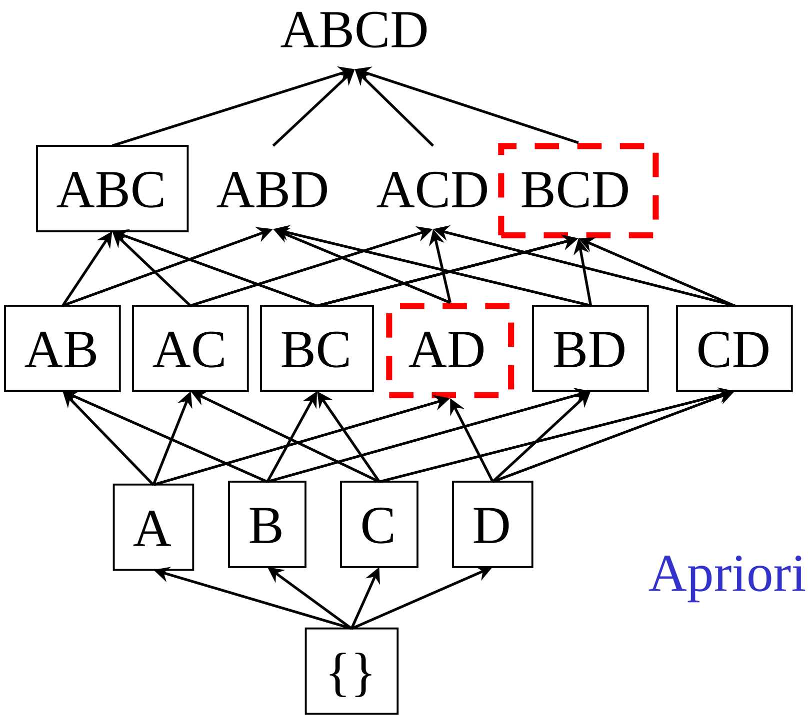
count	itemsets
35	{ab, ad, ae}
88	{bd, be, de}
. . .	. . .
102	{yz, qs, wt}

Hash Table

Sampling for Frequent Patterns

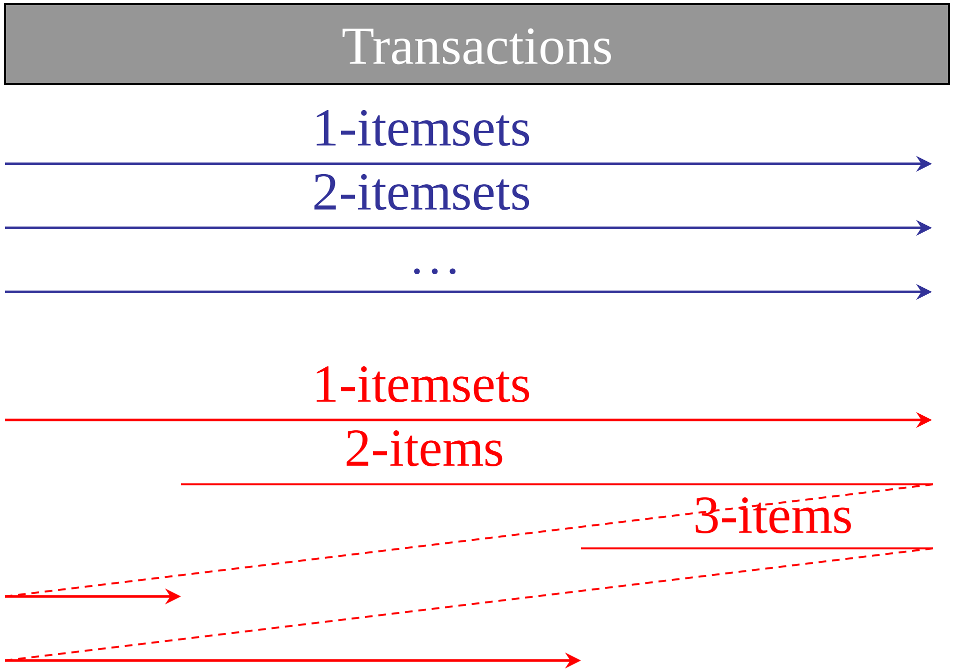
- Select a sample of original database, mine frequent patterns within sample using Apriori
- Scan database once to verify frequent itemsets found in sample, only *borders* of closure of frequent patterns are checked : Ex.: check *abcd* instead of *ab, ac, …, etc.*
- Scan database again to find missed frequent patterns
- H. Toivonen. Sampling large databases for association rules. In *VLDB'96*

DIC: Reduce Number of Scans



Itemset lattice

- Once both A & D are determined frequent, counting of AD begins
- Once all length-2 subsets of BCD are determined frequent, the counting of BCD begins



S. Brin R. Motwani, J. Ullman, and S. Tsur. Dynamic itemset counting and implication rules for market basket data. *SIGMOD'97*

Pattern-Growth Approach: Mining Frequent Patterns Without Candidate Generation

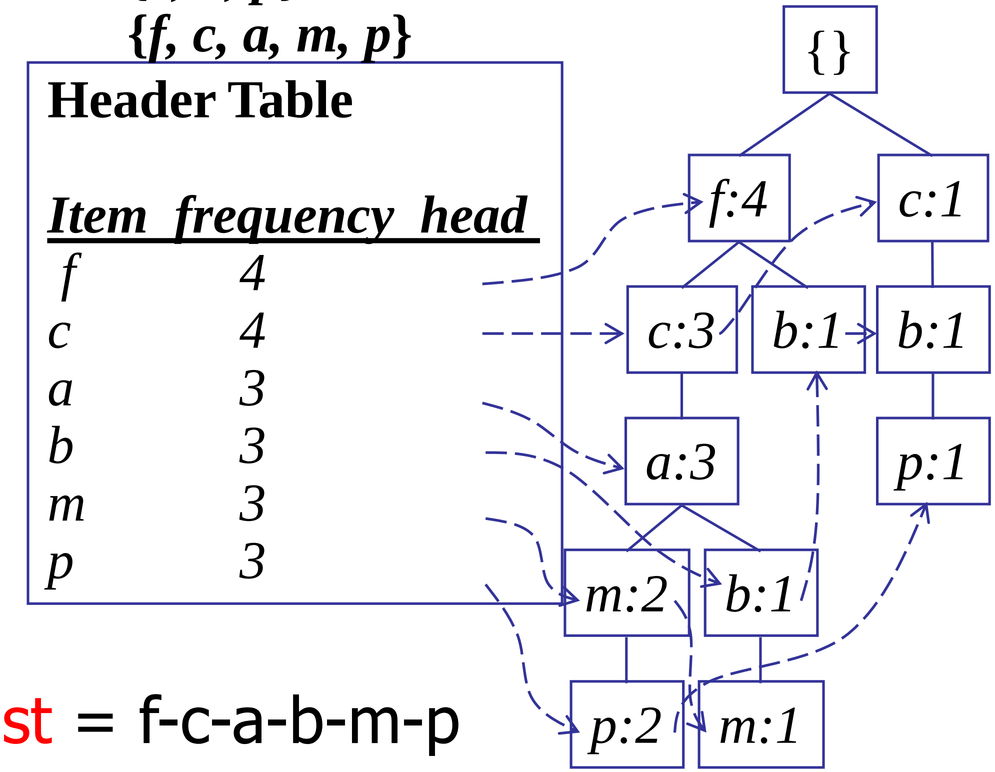
- Bottlenecks of the Apriori approach
  - Breadth-first (i.e., level-wise) search
  - Candidate generation and test
    - Often generates a huge number of candidates
- The FPGrowth Approach (J. Han, J. Pei, and Y. Yin, SIGMOD' 00)
  - Depth-first search
  - Avoid explicit candidate generation
- Major philosophy: Grow long patterns from short ones using local frequent items only
  - "abc" is a frequent pattern
  - Get all transactions having "abc", i.e., project DB on abc: DB|abc
  - "d" is a local frequent item in DB|abc → abcd is a frequent pattern

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**

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



Header Table

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

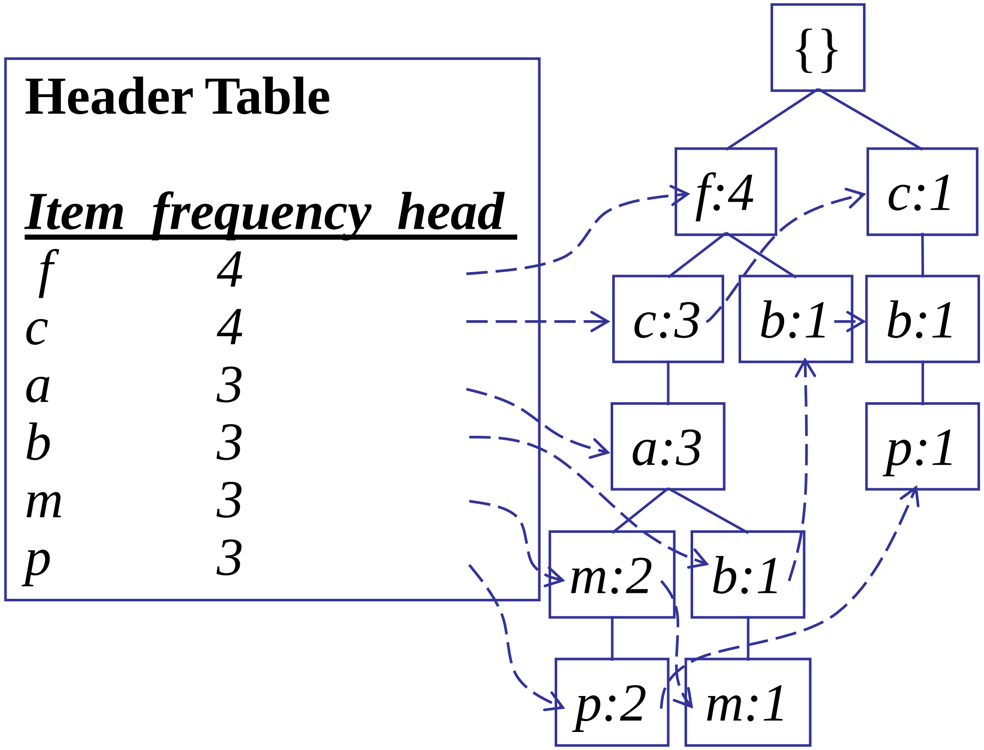
F-list = f-c-a-b-m-p

Partition Patterns and Databases

- Frequent patterns can be partitioned into subsets according to f-list
  - F-list = f-c-a-b-m-p
  - Patterns containing p
  - Patterns having m but no p
  - …
  - Patterns having c but no a nor b, m, p
  - Pattern f
- Completeness and non-redundency

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



Header Table

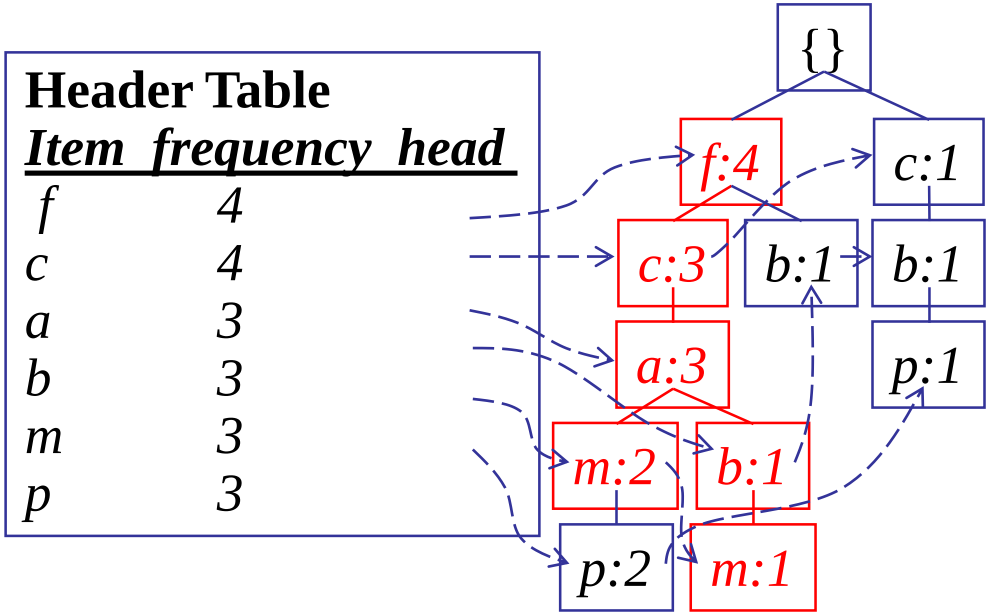
Item	*frequency*	*head*
f	4	
c	4	
a	3	
b	3	
m	3	
p	3	

***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***

From Conditional Pattern-bases to Conditional FP-trees

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



Header Table

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

m-conditional **pattern base:**

fca:2, fcab:1

➔

m-conditional **FP-tree**

➔

All frequent patterns relate to *m*

m,

fm, cm, am,

fcm, fam, cam,

fcam,

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)

The Frequent Pattern Growth Mining Method

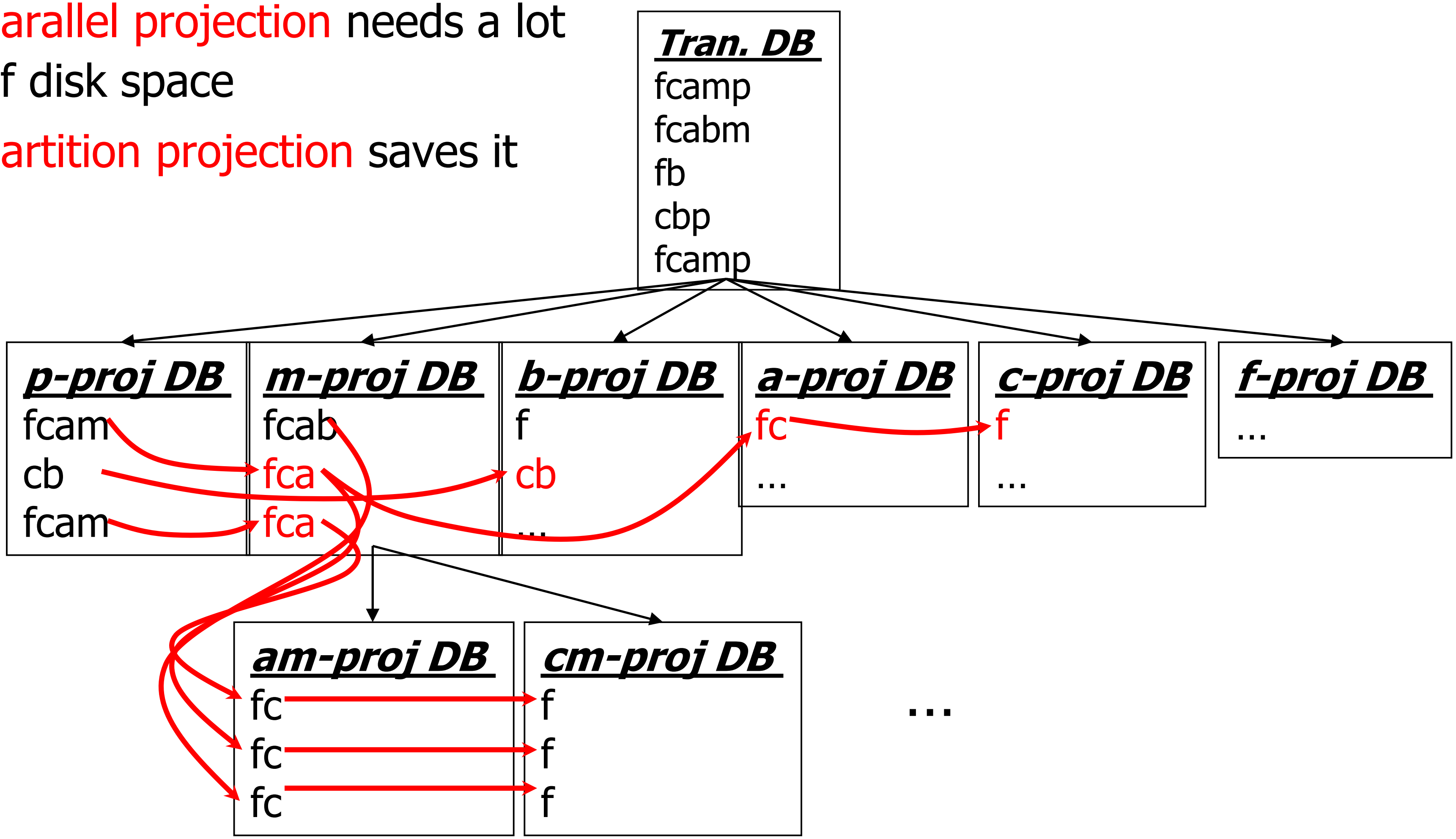
- Idea: Frequent pattern growth
  - Recursively grow frequent patterns by pattern and database partition
- Method
  - For each frequent item, construct its conditional pattern-base, and then its conditional FP-tree
  - Repeat the process on each newly created conditional FP-tree
  - Until the resulting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

Scaling FP-growth by Database Projection

- What about if FP-tree cannot fit in memory?
  - DB projection
- First partition a database into a set of projected DBs
- Then construct and mine FP-tree for each projected DB
- Parallel projection vs. partition projection techniques
  - Parallel projection
    - Project the DB in parallel for each frequent item
    - Parallel projection is space costly
    - All the partitions can be processed in parallel
  - Partition projection
    - Partition the DB based on the ordered frequent items
    - Passing unprocessed parts to subsequent partitions

Partition-Based Projection

- Parallel projection needs a lot of disk space
- Partition projection saves it



Tran. DB
fcamp
fcabm
fb
cbp
fcamp

p-proj DB
fcam
cb
fcam

m-proj DB
fcab
fca
fca

b-proj DB
f
cb
…

a-proj DB
fc
…

c-proj DB
f
…

f-proj DB
…

am-proj DB
fc
fc
fc

cm-proj DB
f
f
f

…

Advantages of the Pattern Growth Approach

- Divide-and-conquer:
  - Decompose both the mining task and DB according to the frequent patterns obtained so far
  - Lead 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
- A good open-source implementation and refinement of FPGrowth
  - FPGrowth+ (Grahne and J. Zhu, FIMI'03)

Further Improvements of Mining Methods

- AFOPT (Liu, et al. @ KDD'03)
  - A "push-right" method for mining condensed frequent pattern (CFP) tree
- Carpenter (Pan, et al. @ KDD'03)
  - Mine data sets with small rows but numerous columns
  - Construct a row-enumeration tree for efficient mining
- FPgrowth+ (Grahne and Zhu, FIMI'03)
  - Efficiently Using Prefix-Trees in Mining Frequent Itemsets, Proc. ICDM'03 Int. Workshop on Frequent Itemset Mining Implementations (FIMI'03), Melbourne, FL, Nov. 2003
- TD-Close (Liu, et al, SDM'06)

Extension of Pattern Growth Mining Methodology

- Mining closed frequent itemsets and max-patterns
  - CLOSET (DMKD’00), FPclose, and FPMax (Grahne & Zhu, Fimi’03)
- Mining sequential patterns
  - PrefixSpan (ICDE’01), CloSpan (SDM’03), BIDE (ICDE’04)
- Mining graph patterns
  - gSpan (ICDM’02), CloseGraph (KDD’03)
- Constraint-based mining of frequent patterns
  - Convertible constraints (ICDE’01), gPrune (PAKDD’03)
- Computing iceberg data cubes with complex measures
  - H-tree, H-cubing, and Star-cubing (SIGMOD’01, VLDB’03)
- Pattern-growth-based Clustering
  - MaPle (Pei, et al., ICDM’03)
- Pattern-Growth-Based Classification
  - Mining frequent and discriminative patterns (Cheng, et al, ICDE’07)