Association Rules: Background

- Given:
  - a database of transactions
  - each transaction is a set of items
- Find all association rules that satisfy user-specified minimum support and minimum confidence constraints.
- *Example:* 30% of transactions that contain beer also contain diapers; 5% of transactions contain these items
  - 30% : *confidence* of the rule
  - 5% : *support* of the rule
- We are interested in *finding* all rules rather than *verifying* if a rule holds.
- Problem introduced in SIGMOD '93 paper, "Mining association rules between sets of items in large databases" by R. Agrawal, T. Imielinski, and A. Swami.

Application Examples

- Market Basket Analysis
  - " $*$ $\Longrightarrow$ Maintenance Agreement"

    What the store should do to boost Maintenance Agreement sales?
  - "Home Electronics $\Longrightarrow$ $*$ "

    What other products should the store stock up on if the store has a sale on Home Electronics?
- Attached mailing in direct marketing
- Detecting "ping-pong"ing of patients
  - transaction: patient
  - item: doctor/clinic visited by a patient
  - support of a rule: number of common patients
- HIC Australia "success story"
  - associations between medical payment codes
  - saved $500,000 per year per state

Problem Statement

- $\mathcal{I} = \{i_1, i_2, \ldots, i_m\}$: a set of literals, called items.
- Transaction T : a set of items such that $T \subseteq \mathcal{I}$.
- Database $\mathcal{D}$: a set of transactions.
- A transaction T *contains* X, a set of some items in $\mathcal{I}$, if $X \subseteq T$.
- An *association rule* is an implication of the form $X \Longrightarrow Y$, where $X, Y \subset \mathcal{I}$
- The rule $X \Longrightarrow Y$ holds in the transaction set $\mathcal{D}$ with *confidence* c if $c\%$ of transactions in $\mathcal{D}$ that contain X also contain Y.
- The rule $X \Longrightarrow Y$ has *support* s in the transaction set $\mathcal{D}$ if $s\%$ of transactions in $\mathcal{D}$ contain $X \cup Y$.

Find all rules that have support and confidence greater than user-specified minimum support and minimum confidence.

Problem Decomposition

1. Find all sets of items that have minimum support (*frequent itemsets*).
2. Use the frequent itemsets to generate the desired rules.

Problem Decomposition – Example

Transaction ID	Items Bought
1	Shoes, Shirt, Jacket
2	Shoes, Jacket
3	Shoes, Jeans
4	Shirt, Sweatshirt

For Minimum Support = 50% = 2 transactions,
and Minimum Confidence = 50%:

Frequent Itemset	Support
{Shoes}	75%
{Shirt}	50%
{Jacket}	50%
{Shoes, Jacket}	50%

For the rule Shoes $\Longrightarrow$ Jacket :

- Support = Support({Shoes, Jacket}) = 50%
- Confidence = $\frac{\text{Support}(\{\text{Shoes, Jacket}\})}{\text{Support}(\{\text{Shoes}\})} = \frac{50}{75} = 66.6\%$.

Jacket $\Longrightarrow$ Shoes has 50% support and 100% confidence.

Discovering Rules (1)

Naive Algorithm

for each frequent itemset l **do**
  for each subset c of l **do**
    if (support(l)/support($l-c$) $\geq$ *minconf*) **then**
      output the rule $(l-c) \Longrightarrow c$,
        with confidence = support(l)/support($l-c$)
        and support = support(l);

Example: $l = ABC$

c	A	B	C
$\frac{\text{supp}(l)}{\text{supp}(l-c)}$	$\frac{\lvert ABC\rvert}{\lvert BC\rvert}$	$\frac{\lvert ABC\rvert}{\lvert AC\rvert}$	$\frac{\lvert ABC\rvert}{\lvert AB\rvert}$
Rule	$BC \Longrightarrow A$	$AC \Longrightarrow B$	$AB \Longrightarrow C$

c	AB	BC	AC
$\frac{\text{supp}(l)}{\text{supp}(l-c)}$	$\frac{\lvert ABC\rvert}{\lvert C\rvert}$	$\frac{\lvert ABC\rvert}{\lvert A\rvert}$	$\frac{\lvert ABC\rvert}{\lvert B\rvert}$
Rule	$C \Longrightarrow AB$	$A \Longrightarrow BC$	$B \Longrightarrow AC$

Discovering Rules (2)

Lemma. If a consequent c generates a valid rule, so do all subsets of c. (e.g. $X \Longrightarrow YZ$, then $\exists\ XY \Longrightarrow Z$ and $XZ \Longrightarrow Y$.)

Consider the rule $(l-c) \Longrightarrow c$.

Now, if $c_1 \subset c$,

$$\begin{aligned} l-c_1 &\supset l-c \\ \text{support}(l-c_1) &\leq \text{support}(l-c) \\ \text{support}(l)/\text{support}(l-c_1) &\geq \text{support}(l)/\text{support}(l-c) \\ \text{conf}((l-c_1) \Longrightarrow c_1) &\geq \text{conf}((l-c) \Longrightarrow c) \end{aligned}$$

Example Consider a frequent itemset $ABCDE$.

If $ACDE \Longrightarrow B$ and $ABCE \Longrightarrow D$ are the only one-consequent rules with minimum confidence, then
$ACE \Longrightarrow BD$ is the only other rule that needs be tested.

The Apriori Algorithm

- L_k : Set of frequent itemsets of size k (those with minimum support).
- C_k : Set of candidate itemsets of size k (potentially frequent itemsets)

$L_1 = \{$frequent items$\}$;
for ($k = 1$; $L_k \neq \emptyset$; $k{+}{+}$) **do**
  begin
  C_{k+1} = New candidates generated from L_k;
  foreach transaction t in the 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 minimum support.
  end
Answer = $\bigcup_k L_k$;

Apriori – Example

Database $\mathcal{D}$

TID	Items
10	1 3 4
20	2 3 5
30	1 2 3 5
40	2 5

Minimum Support = 50% = 2 trans.

Scan $\mathcal{D}$ $\longrightarrow$

C_1

Itemset	Sup.
{1}	2
{2}	3
{3}	3
{4}	1
{5}	3

L_1

Itemset	Sup.
{1}	2
{2}	3
{3}	3
{5}	3

C_2

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

Scan $\mathcal{D}$ $\longrightarrow$

C_2

Itemset	Sup.
{1 2}	1
{1 3}	2
{1 5}	1
{2 3}	2
{2 5}	3
{3 5}	2

L_2

Itemset	Sup.
{1 3}	2
{2 3}	2
{2 5}	3
{3 5}	2

C_3

Itemset
{2 3 5}

Scan $\mathcal{D}$ $\longrightarrow$

C_3

Itemset	Sup.
{2 3 5}	2

L_3

Itemset	Sup.
{2 3 5}	2

Apriori Candidate Generation

Monotonicity Property: All subsets of a frequent itemset are frequent.

Given L_k, generate C_{k+1} in two steps:

$$L_3 = \boxed{\begin{array}{c} \{1\ 2\ 3\} \\ \{1\ 2\ 4\} \\ \{1\ 3\ 4\} \\ \{1\ 3\ 5\} \\ \{2\ 3\ 4\} \end{array}}$$

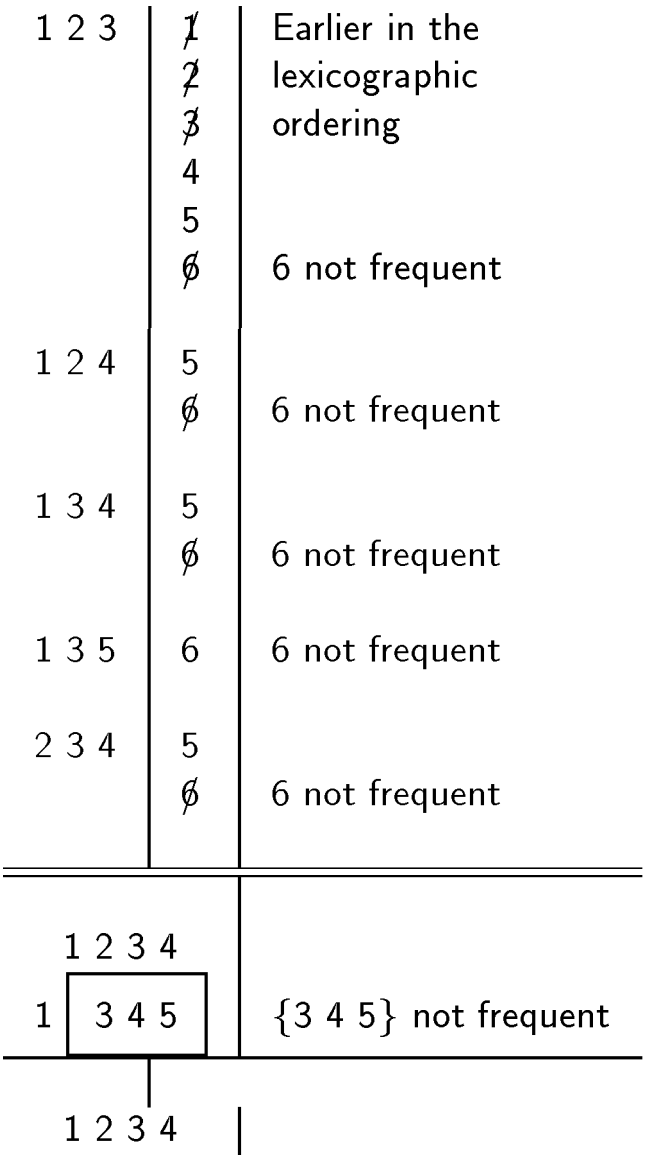
1. *Join* Step : Join L_k with L_k, with the join condition that the first $k-1$ items should be the same **and** $l^1[k] < l^2[k]$.

   Now, $C_4 = \{\{1\ 2\ 3\ 4\}, \{1\ \underline{3\ 4\ 5}\}\ \}$.

2. *Prune* Step : Delete all candidates which have a non-frequent subset.

   Now, $C_4 = \{\{1\ 2\ 3\ 4\}\}$.

Candidate Generation: Example



Earlier in the lexicographic ordering

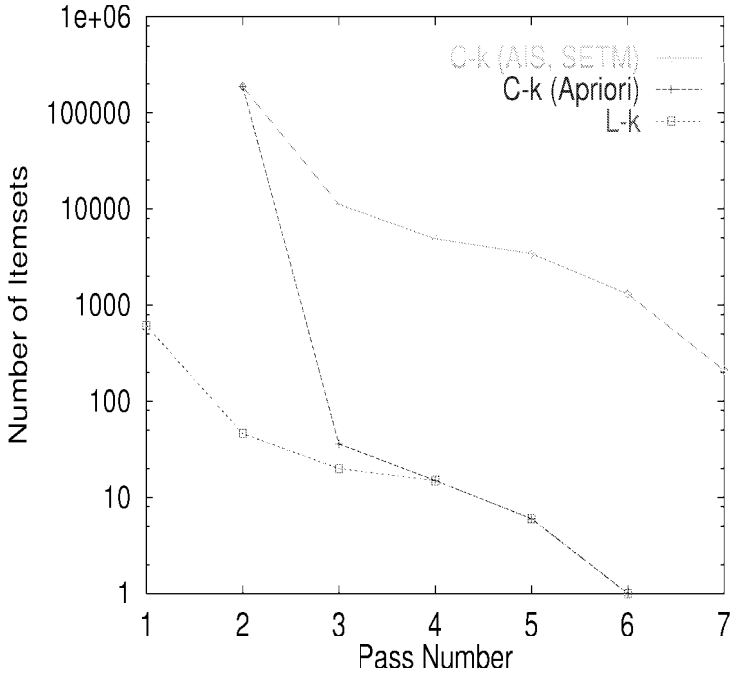
On-The-Fly vs. Apriori Candidate Generation

$$L_3 = \boxed{\begin{array}{c} \{1\ 2\ 3\} \\ \{1\ 2\ 4\} \\ \{1\ 3\ 4\} \\ \{1\ 3\ 5\} \\ \{2\ 3\ 4\} \end{array}}$$

Transaction T : $\{1\ 2\ 3\ 4\ 5\}$

$\{1\ 2\ 3\}$ contained in T.
$\Longrightarrow$ Generate $\{1\ 2\ 3\ 4\}$ and $\{1\ 2\ 3\ 5\}$.

Similarly, generate $\{1\ 2\ 4\ 5\}$, $\{1\ 3\ 4\ 5\}$ and $\{2\ 3\ 4\ 5\}$.



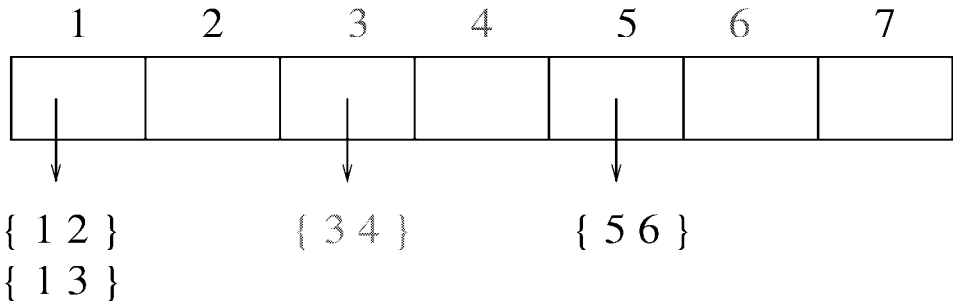
Finding Candidates Contained in a Transaction

Given

- a transaction T and
- a set of candidates C_k,

find all members of C_k which are contained in T.

C_2 : { {1, 2}, {1, 3}, {3, 4}, {5, 6} }
T : {3, 4, 6}



Only check itemsets in buckets corresponding to 3, 4, and 6, i.e., {3,4}.

- avg. number of items in trans. $\ll$ total number of items
- generalized into a *hash-tree*

Synthetic Data: Parameters

- Number of transactions (100,000 to 10 million)
- Average size of the Transactions (5 to 50)
- Number of items (1000 to 10,000)
- Average size of the maximal potentially frequent Itemsets (2 to 6)
- Number of maximal potentially Frequent itemsets