# Web Data Mining

# Example

Consider the following transactions:

(Bread, Eggs, Butter)

(Eggs, Bread, Milk)

(Cheese, Chips)

(Chips, Milk, Egg)

Find all rules that satisfy minimum support=0.5 and minimum confidence=0.5

# Road map

- Basic concepts of Association Rules
- Apriori algorithm
- Different data formats for mining
- Sequential pattern mining
- Summary

# Sequential pattern mining

- Association rule mining does not consider the order of transactions.
- In many applications such orderings are significant. E.g.,
  - in market basket analysis, it is interesting to know whether people buy some items in sequence,
    - e.g., buying bed first and then bed sheets some time later.
  - In Web usage mining, it is useful to find navigational patterns of users in a Web site from sequences of page visits of users

# Basic concepts

- Let $I = \{i_1, i_2, \ldots, i_m\}$ be a set of items.
- **Sequence:** An ordered list of itemsets.
- **Itemset/element**: A non-empty set of items $X \subseteq I$. We denote a sequence $s$ by $\langle a_1 a_2 \ldots a_r \rangle$, where $a_i$ is an itemset, which is also called an **element** of $s$.
- An element (or an itemset) of a sequence is denoted by $\{x_1, x_2, \ldots, x_k\}$, where $x_j \in I$ is an item.
- We assume without loss of generality that items in an element of a sequence are in **lexicographic order**.

# Basic concepts (contd)

- **Size**: The **size** of a sequence is the number of elements (or itemsets) in the sequence.
- **Length**: The **length** of a sequence is the number of items in the sequence.
  - A sequence of length $k$ is called ***k*-sequence**.
- A sequence $s_1 = \langle a_1 a_2 \ldots a_r \rangle$ is a **subsequence** of another sequence $s_2 = \langle b_1 b_2 \ldots b_v \rangle$, or $s_2$ is a **supersequence** of $s_1$, if there exist integers $1 \le j_1 < j_2 < \ldots < j_{r-1} < j_r \le v$ such that $a_1 \subseteq b_{j1}$, $a_2 \subseteq b_{j2}$, …, $a_r \subseteq b_{jr}$. We also say that $s_2$ **contains** $s_1$.

# An example

- Let $I = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$.
- Sequence $\langle\{3\}\{4, 5\}\{8\}\rangle$ is **contained** in (or is a **subsequence** of) $\langle\{6\}\ \{3, 7\}\{9\}\{4, 5, 8\}\{3, 8\}\rangle$
  - because $\{3\} \subseteq \{3, 7\}$, $\{4, 5\} \subseteq \{4, 5, 8\}$, and $\{8\} \subseteq \{3, 8\}$.
  - However, $\langle\{3\}\{8\}\rangle$ is not contained in $\langle\{3, 8\}\rangle$ or vice versa.
  - The size of the sequence $\langle\{3\}\{4, 5\}\{8\}\rangle$ is 3, and the length of the sequence is 4.

# Objective

- Given a set *S* of input data sequences (or sequence database), the problem of mining sequential patterns is to find all the sequences that have a user-specified minimum support.
- Each such sequence is called a **frequent sequence**, or a **sequential pattern**.
- The **support** for a sequence is the fraction of total data sequences in *S* that contains this sequence.

# Example

Table 1. A set of transactions sorted by customer ID and transaction time

| Customer ID | Transaction Time | Transaction (items bought) |
|---|---|---|
| 1 | July 20, 2005 | 30 |
| 1 | July 25, 2005 | 90 |
| 2 | July 9, 2005 | 10, 20 |
| 2 | July 14, 2005 | 30 |
| 2 | July 20, 2005 | 40, 60, 70 |
| 3 | July 25, 2005 | 30, 50, 70 |
| 4 | July 25, 2005 | 30 |
| 4 | July 29, 2005 | 40, 70 |
| 4 | August 2, 2005 | 90 |
| 5 | July 12, 2005 | 90 |

# Example (cond)

**Table 2.** Data sequences produced from the transaction database in Table 1.

| Customer ID | Data Sequence |
|---|---|
| 1 | ⟨{30} {90}⟩ |
| 2 | ⟨{10, 20} {30} {40, 60, 70}⟩ |
| 3 | ⟨{30, 50, 70}⟩ |
| 4 | ⟨{30} {40, 70} {90}⟩ |
| 5 | ⟨{90}⟩ |

**Table 3.** The final output sequential patterns

| | Sequential Patterns with Support ≥ 25% |
|---|---|
| 1-sequences | ⟨{30}⟩, ⟨{40}⟩, ⟨{70}⟩, ⟨{90}⟩ |
| 2-sequences | ⟨{30} {40}⟩, ⟨{30} {70}⟩, ⟨{30} {90}⟩, ⟨{40, 70}⟩ |
| 3-sequences | ⟨{30} {40, 70}⟩ |

# GSP mining algorithm

- Very similar to the Apriori algorithm

```
Algorithm GSP(S)
C1 ← init-pass(S);                              // the first pass over S
F1 ← {⟨{f}⟩ | f ∈ C1, f.count/n ≥ minsup};      // n is the number of sequences in S
for (k = 2; Fk-1 ≠ ∅; k++) do                  // subsequent passes over S
   Ck ← candidate-gen-SPM(Fk-1);
   for each data sequence s ∈ S do              // scan the data once
      for each candidate c ∈ Ck do
         if c is contained in s then
            c.count++;                          // increment the support count
      end
   end
   Fk ← {c ∈ Ck | c.count/n ≥ minsup}
end
return ∪k Fk;
```

**Fig. 12.** The GSP Algorithm for generating sequential patterns

# Candidate generation

**Function** candidate-gen-SPM($F_{k-1}$)

1. **Join step.** Candidate sequences are generated by joining $F_{k-1}$ with $F_{k-1}$. A sequence $s_1$ joins with $s_2$ if the subsequence obtained by dropping the first item of $s_1$ is the same as the subsequence obtained by dropping the last item of $s_2$. The candidate sequence generated by joining $s_1$ with $s_2$ is the sequence $s_1$ extended with the last item in $s_2$. There are two cases:
   - the added item forms a separate element if it was a separate element in $s_2$, and is appended at the end of $s_1$ in the merged sequence, and
   - the added item is part of the last element of $s_1$ in the merged sequence otherwise.

   When joining $F_1$ with $F_1$, we need to add the item in $s_2$ both as part of an itemset and as a separate element. That is, joining $\langle\{x\}\rangle$ with $\langle\{y\}\rangle$ gives us both $\langle\{x, y\}\rangle$ and $\langle\{x\}\{y\}\rangle$. Note that $x$ and $y$ in $\{x, y\}$ are ordered.
2. **Prune step**. A candidate sequence is pruned if any one of its $(k-1)$-subsequence is infrequent (without minimum support).

**Fig. 13.** The candidate-gen-SPM() function

# An example

**Table 4.** Candidate generation: an example

| Frequent 3-sequences | Candidate 4-sequences after joining | Candidate 4-sequences after pruning |
|---|---|---|
| ⟨{1, 2} {4}⟩ | ⟨{1, 2} {4, 5}⟩ | ⟨{1, 2} {4, 5}⟩ |
| ⟨{1, 2} {5}⟩ | ⟨{1, 2} {4} {6}⟩ | |
| ⟨{1} {4, 5}⟩ | | |
| ⟨{1, 4} {6}⟩ | | |
| ⟨{2} {4, 5}⟩ | | |
| ⟨{2} {4} {6}⟩ | | |

# Road map

- Basic concepts of Association Rules
- Apriori algorithm
- Different data formats for mining
- Mining with multiple minimum supports
- Mining class association rules
- Sequential pattern mining
- Summary

# Summary

- Association rule mining has been extensively studied in the data mining community.
- So is sequential pattern mining
- There are many efficient algorithms and model variations.
- Other related work includes
  - Multi-level or generalized rule mining
  - Constrained rule mining
  - Incremental rule mining
  - Maximal frequent itemset mining
  - Closed itemset mining
  - Rule interestingness and visualization
  - Parallel algorithms
  - …

# Example

| Customer ID | Transaction Data | Item bought |
|---|---|---|
| 1 | 1st January | Bread, Jam |
| 1 | 3st January | Milk, Butter |
| 2 | 2nd January | Bread, Butter |
| 2 | 5th January | Milk |
| 3 | 1st January | Milk |
| 3 | 4th January | Butter, Jam |
| 4 | 3rd January | Bread, Butter, Jam |

Find sequence with at least 50% support