Maximal, Closed and Generator Itemsets

Philippe Fournier-Viger

http://www.philippe-Fournier-viger.com

Source code and datasets available in the SPMF library
Introduction

- In previous videos, we talked about frequent itemset mining.
- It is a data analysis task that has many applications.
- However, a problem is that the number of frequent itemsets is often very very large.
- Besides, frequent itemsets often have some form of redundancy.
- Today, we will talk about this problem and a solution, which is to discover concise representations of frequent itemsets.
Frequent itemset mining

**Input:** A transaction database

<table>
<thead>
<tr>
<th>Transaction</th>
<th>Items appearing in the transaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>{pasta, lemon, bread, orange}</td>
</tr>
<tr>
<td>T2</td>
<td>{pasta, lemon}</td>
</tr>
<tr>
<td>T3</td>
<td>{pasta, orange, cake}</td>
</tr>
<tr>
<td>T4</td>
<td>{pasta, lemon, orange, cake}</td>
</tr>
</tbody>
</table>

\[ \text{minsup} = 2 \]
Frequent itemset mining

**Input:** A transaction database

<table>
<thead>
<tr>
<th>Transaction</th>
<th>Items appearing in the transaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>{pasta, lemon, bread, orange}</td>
</tr>
<tr>
<td>T2</td>
<td>{pasta, lemon}</td>
</tr>
<tr>
<td>T3</td>
<td>{pasta, orange, cake}</td>
</tr>
<tr>
<td>T4</td>
<td>{pasta, lemon, orange, cake}</td>
</tr>
</tbody>
</table>

\[ \text{minsup} = 2 \]

**Output:** The frequent itemsets

{lemon}, {pasta}, {orange}, {cake}, {lemon, pasta}, {lemon, orange}, {pasta, orange}, {pasta, cake}, {orange, cake}, {lemon, pasta, orange}, {pasta, orange, cake}
Frequent itemsets

{pasta}  support = 4
{lemon}  support = 3
{orange} support = 3
{cake}  support = 2

{pasta, lemon} support: 3
{pasta, orange} support: 3
{pasta, cake}  support: 2
{lemon, orange}  support: 2
{orange, cake}  support: 2

{pasta, lemon, orange} support: 2
{pasta, orange, cake} support: 2
Frequent itemsets

- \{pasta\}  \text{support} = 4
- \{lemon\}  \text{support} = 3
- \{orange\} \text{support} = 3
- \{cake\}  \text{support} = 2

- \{pasta, lemon\} \text{support}: 3
- \{pasta, orange\} \text{support}: 3
- \{pasta, cake\}  \text{support}: 2
- \{lemon, orange\} \text{support}: 2
- \{orange, cake\}  \text{support}: 2

- \{pasta, lemon, orange\} \text{support}: 2
- \{pasta, orange, cake\} \text{support}: 2

There are 10 frequent itemsets!
minsup = 2

The frequent itemsets

l = lemon
p = pasta
b = bread
0 = orange
c = cake

frequent itemsets

Infrequent itemsets
minsups = 2

The frequent itemsets

We can observe that if an itemset is frequent then all its subsets are frequent
minsup = 2

The frequent itemsets

We call the largest frequent itemsets the **maximal frequent itemsets**
minsуп = 2

The frequent itemsets

Maximal frequent itemsets are interesting because if we find them, we can recover all the other frequent itemsets.

l = lemon
p = pasta
b = bread
0 = orange
c = cake
In this example \((\text{minsup} = 2)\), there are only two maximal itemsets:

- \{pasta, lemon, orange\} \quad \text{support: 2}
- \{pasta, orange, cake\} \quad \text{support: 2}
**Maximal frequent itemsets**

**Definition:** A frequent itemset $X$ is **maximal** if there is no frequent itemset $Y$ such that $X \subseteq Y$.

In this example ($\text{minsup} = 2$), there are only two maximal itemsets:

- $\{\text{pasta, lemon, orange}\}$  \hspace{1cm} \text{support: 2}
- $\{\text{pasta, orange, cake}\}$  \hspace{1cm} \text{support: 2}

We can recover all other frequent itemsets from the maximal itemsets by simply enumerating their subsets:
Maximal frequent itemsets

**Definition:** A frequent itemset $X$ is **maximal** if there is no frequent itemset $Y$ such that $X \subset Y$.

In this example ($\text{minsup} = 2$), there are only two maximal itemsets:

- \{pasta, lemon, orange\}  support: 2
- \{pasta, orange, cake\}  support: 2

We can **recover** all other frequent itemsets from the maximal itemsets by simply enumerating their subsets:

- \{lemon\}, \{pasta\}, \{orange\}, \{cake\}, \{lemon, pasta\}, \{lemon, orange\}, \{pasta, orange\}, \{pasta, cake\}, \{orange, cake\},
Maximal frequent itemsets

**Definition:** A frequent itemset \( X \) is maximal if there is no frequent itemset \( Y \) such that \( X \subset Y \).

In this example \((\text{minsup} = 2)\), there are only two maximal itemsets:

- \{pasta, lemon, orange\} \quad \text{support: 2}
- \{pasta, orange, cake\} \quad \text{support: 2}

We can **recover** all other frequent itemsets from the maximal itemsets by simply enumerating their subsets:

\{lemon\}, \{pasta\}, \{orange\}, \{cake\}, \{lemon, pasta\}, \{lemon, orange\}, \{pasta, orange\}, \{pasta, cake\}, \{orange, cake\},

But we cannot recover their support!
Maximal itemsets are compact

If we find the maximal frequent itemsets, we can find much less itemsets.

Example: *Chess* dataset and minsup = 0.4

6,439,702 frequent itemsets

38,050 frequent maximal itemsets

168 times smaller
How to find the maximal itemsets?

- **Naïve approach**
  - by post-processing
  - inefficient!

- **Efficient algorithms:**
  - GenMax: a modified version of ECLAT
  - FPMAX: a modified version of FP-Growth
  - LCMMax: a modified version of LCM
  - … and many others!

- Those algorithms adopt various strategies to find the maximal itemsets without having to find all frequent itemsets.
What are the alternatives?

- Is there some other ways of summarizing the frequent itemsets?
- Another popular representation of frequent itemsets is the **closed itemsets** (Pasquier et al., 1999)
Could we do better?

- Is there some other ways of summarizing the frequent itemsets?
- Another popular representation of frequent itemsets is the closed itemsets (Pasquier et al., 1999)

**Definition:** A (frequent) itemset $X$ is closed if there exists no (frequent) itemset $Y$ such that $X \subset Y$ and $\text{sup}(X) = \text{sup}(Y)$. 
## Closed itemsets

{pasta}  
support = 4  
closed

{lemon}  
support = 3

{orange}  
support = 3

{cake}  
support = 2

{pasta, lemon}  
support: 3  
closed

{pasta, orange}  
support: 3  
closed

{pasta, cake}  
support: 2

{lemon, orange}  
support: 2

{orange, cake}  
support: 2

{pasta, lemon, orange}  
support: 2  
closed

{pasta, orange, cake}  
support: 2  
closed
minsup = 2

The frequent closed itemsets

minsup = 2

frequent itemsets

infrequent itemsets

support

frequent non closed itemsets

frequent closed itemsets
Closed itemsets are compact

Example: *Chess* dataset and $\text{minsup} = 0.4$

- 6,439,702 frequent itemsets
- 1,361,157 frequent closed itemsets
- 38,050 frequent maximal itemsets
Recovering all frequent itemsets

Definition: A (frequent) itemset $X$ is closed if there exists no (frequent) itemset $Y$ such that $X \subseteq Y$ and $\text{sup}(X) = \text{sup}(Y)$.

In this example ($\text{minsup} = 2$), there are only five closed itemsets:

- \{pasta\} \quad \text{support} = 4 \quad \text{closed}
- \{pasta, lemon\} \quad \text{support: 3} \quad \text{closed}
- \{pasta, orange\} \quad \text{support: 3} \quad \text{closed}
- \{pasta, lemon, orange\} \quad \text{support: 2} \quad \text{closed}
- \{pasta, orange, cake\} \quad \text{support: 2} \quad \text{closed}
Recovering all frequent itemsets

**Definition:** A (frequent) itemset $X$ is **closed** if there exists no (frequent) itemset $Y$ such that $X \subseteq Y$ and $\text{sup}(X) = \text{sup}(Y)$.

In this example ($\text{minsup} = 2$), there are only five closed itemsets:

- \{pasta\}  \quad \text{support} = 4  \quad \text{closed}
- \{pasta, lemon\}  \quad \text{support} = 3  \quad \text{closed}
- \{pasta, orange\}  \quad \text{support} = 3  \quad \text{closed}
- \{pasta, lemon, orange\}  \quad \text{support} = 2  \quad \text{closed}
- \{pasta, orange, cake\}  \quad \text{support} = 2  \quad \text{closed}

By enumerating all their subsets, we can find all the other frequent itemsets:

- \{pasta, cake\}, \{lemon, orange\}, \{orange, cake\}, \{lemon, \{orange\}, \{cake\}
Recovering all frequent itemsets

**Definition:** A (frequent) **itemset** $X$ is **closed** if there exists no (frequent) itemset $Y$ such that $X \subseteq Y$ and $\text{sup}(X) = \text{sup}(Y)$.

In this example ($\text{minsup} = 2$), there are only five closed itemsets:

- \{pasta\} \quad \text{support} = 4 \quad \text{closed}
- \{pasta, lemon\} \quad \text{support: 3} \quad \text{closed}
- \{pasta, orange\} \quad \text{support: 3} \quad \text{closed}
- \{pasta, lemon, orange\} \quad \text{support: 2} \quad \text{closed}
- \{pasta, orange, cake\} \quad \text{support: 2} \quad \text{closed}

By enumerating all their subsets, we can find all the other frequent itemsets:

- \{pasta, cake\}, \{lemon, orange\}, \{orange, cake\}, \{lemon, \{orange\} , \{cake\}

But how to find the support of these itemsets?
Steps to find the support of a frequent itemset $X$ from closed itemsets

1. Find the closure of $X$.
2. Return the support of the closure of $X$.

**Definition:** The closure of an itemset $X$ is the smallest closed itemset $Y$ such that $X \subseteq Y$. It is denoted as $c(X)$.
Steps to find the support of a frequent itemset \( X \) from closed itemsets

1. Find the closure of \( X \).
2. Return the support of the closure of \( X \).

**Definition:** The closure of an itemset \( X \) is the smallest closed itemset \( Y \) such that \( X \subseteq Y \). It is denoted as \( c(X) \).

Example:

- \( \{ \text{pasta} \} \)  
  - support = 4  
  - closed
- \( \{ \text{pasta, lemon} \} \)  
  - support: 3  
  - closed
- \( \{ \text{pasta, orange} \} \)  
  - support: 3  
  - closed
- \( \{ \text{pasta, lemon, orange} \} \)  
  - support: 2  
  - closed
- \( \{ \text{pasta, orange, cake} \} \)  
  - support: 2  
  - closed

The closure of \( \{ \text{cake} \} \) is \( c(\{ \text{cake} \}) \) ???
Steps to find the support of a frequent itemset $X$ from closed itemsets

1. Find the closure of $X$.
2. Return the support of the closure of $X$

**Definition:** The closure of an itemset $X$ is the smallest closed itemset $Y$ such that $X \subseteq Y$. It is denoted as $c(X)$

**Example:**

- $\{\text{pasta}\}$ support = 4 closed
- $\{\text{pasta, lemon}\}$ support: 3 closed
- $\{\text{pasta, orange}\}$ support: 3 closed
- $\{\text{pasta, lemon, orange}\}$ support: 2 closed
- $\{\text{pasta, orange, cake}\}$ support: 2 closed

The closure of $\{\text{cake}\}$ is $c(\{\text{cake}\}) = \{\text{pasta, orange, cake}\}$
Thus, the support of $\{\text{cake}\}$ is the same, that is 2.
Itemsets and their closures

\[ \text{minsup} = 2 \]

\[ \emptyset \]

support

frequent non closed itemsets

frequent closed itemsets
Itemsets that have the same closure

- They all have the same support
- They all appear in the same transactions.

**Example:**

\{\text{cake}\}, \{\text{pasta, cake}\}, \{\text{orange, cake}\}, \{\text{pasta, orange, cake}\}

They all have a support of 2
They all appear in \text{T3} and \text{T4}

<table>
<thead>
<tr>
<th>Trans.</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>{\text{pasta, lemon, bread, orange}}</td>
</tr>
<tr>
<td>T2</td>
<td>{\text{pasta, lemon}}</td>
</tr>
<tr>
<td>T3</td>
<td>{\text{pasta, orange, cake}}</td>
</tr>
<tr>
<td>T4</td>
<td>{\text{pasta, lemon, orange cake}}</td>
</tr>
</tbody>
</table>
Itemsets that have the same closure

- They all have the same support
- They all appear in the same transactions.

Example:

{cake}
{pasta, cake},
{orange, cake},
{pasta, orange, cake}

They all have a support of 2
They all appears in \( T_3 \) and \( T_4 \)

Observation:
The closure is the intersection of transactions \( T_3 \) and \( T_4 \)

<table>
<thead>
<tr>
<th>Trans.</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>{pasta, lemon, bread, orange}</td>
</tr>
<tr>
<td>T2</td>
<td>{pasta, lemon}</td>
</tr>
<tr>
<td>T3</td>
<td>{pasta, orange, cake}</td>
</tr>
<tr>
<td>T4</td>
<td>{pasta, lemon, orange cake}</td>
</tr>
</tbody>
</table>
Itemsets and their closures

\[ \text{minsup} = 2 \]

support
frequent non closed itemsets
frequent closed itemsets
minsup = 2

Note: Itemsets having the same closure are sometimes said to be from a same equivalence class.
How to find the closed itemsets?

- **Naïve approach:**
  - by post-processing
  - **inefficient!**

- **Efficient algorithms:**
  - Charm, DCI_Closed, FPClose, Closet, LCM, etc.
  - Find frequent itemsets by avoiding generating all frequent itemsets.
  - Several strategies and properties.
Charm (Zaki, 2001)

- Based on ECLAT
- Depth-first search

Four cases must be checked to avoid generating all frequent itemsets.
Four cases

Suppose that the algorithm is considering combining an itemset $X_1$ with an itemset $X_2$ such that $X_2 \succ X_1$.

Property 1

**IF** $t(X_1) = t(X_2)$, **THEN** $t(X_1) = t(X_2) = t(X_1 \cup X_2)$.

- Thus, we can replace all occurrences of $X_1$ by $X_1 \cup X_2$.
- We do not need to consider $X_2$.

Notation: $t(X)$ denotes the set of transactions containing an itemset $X$. 
Four cases

Suppose that the algorithm is considering combining an itemset \( X_1 \) with an itemset \( X_2 \) such that \( X_2 \succ X_1 \).

Property 2

**IF** \( t(X_1) \subset t(X_2) \), **THEN** \( t(X_1) = t(X_1 \cup X_2) \neq t(X_2) \).

- Thus, we can replace all occurrences of \( X_1 \) by \( X_1 \cup X_2 \).
- But we must keep \( X_2 \).
Four cases

Suppose that the algorithm is considering combining an itemset $X_1$ with an itemset $X_2$ such that $X_2 \succ X_1$.

Property 3

Si $t(X_1) \supset t(X_2)$, alors $t(X_2) = t(X_1 \cup X_2) \neq t(X_1)$.

- Thus, we can replace all occurrences of $X_2$ by $X_1 \cup X_2$.
- But we must keep $X_1$. 
Four cases

Suppose that the algorithm is considering combining an itemset $X_1$ with an itemset $X_2$ such that $X_2 > X_1$.

Property 4

IF $t(X_1) \neq t(X_2)$, then $t(X_2) \neq t(X_1 \cup X_2) \neq t(X_1)$.

• We cannot remove anything.
Charm

- These four properties allow to:
  - eliminate several frequent itemsets
  - thus, reduce the search space
- However, the algorithm can still generate several itemsets that are not closed.
- We must add a **closure checking test**.
Closure checking

- For each frequent itemset $X$ found, we compare it to the other frequent closed itemsets found until now.
- If $X$ is contained in an itemset that has been already found, we do not keep $X$.
- Otherwise, $X$ is closed. It is added to the set of closed itemsets.
- How to implement this test?
  - store closed itemsets in a hash table
  - hash function is the list of transaction ids $t(.)$. 
Pseudocode of Charm

**CHARM** ($\delta \subseteq I \times T$, $\text{minsup}$):
1. Nodes = \{ $I_j \times t(I_j) : I_j \in I \land |t(I_j)| \geq \text{minsup}$ \}
2. **CHARM-EXTEND** (Nodes, $C$)

**CHARM-EXTEND** (Nodes, $C$):
3. **for each** $X_i \times t(X_i)$ in Nodes
4. NewN = $\emptyset$ and $X = X_i$
5. **for each** $X_j \times t(X_j)$ in Nodes, with $j > i$
6. $X = X \cup X_j$ and $Y = t(X_i) \cap t(X_j)$
7. **CHARM-PROPERTY** (Nodes, NewN)
8. **if** NewN $\neq \emptyset$ **then** **CHARM-EXTEND** (NewN)
9. $C = C \cup X$ // vérifier le test de closure

**CHARM-PROPERTY** (Nodes, NewN):
10. **if** ($|Y| \geq \text{minsup}$) **then**
11. **if** $t(X_i) = t(X_j)$ **then** //Property 1
12. Remove $X_j$ from Nodes
13. Replace all $X_i$ with $X$
14. **else if** $t(X_i) \subset t(X_j)$ **then** //Property 2
15. Replace all $X_i$ with $X$
16. **else if** $t(X_i) \supset t(X_j)$ **then** //Property 3
17. Remove $X_j$ from Nodes
18. Add $X \times Y$ to NewN
19. **else if** $t(X_i) \neq t(X_j)$ **then** //Property 4
20. Add $X \times Y$ to NewN

**Nodes** the set of frequent itemsets of size 1

$I$ is the set of items

$T$ is the set of transactions

$C$ is the set of all closed itemsets found

$X \times t(X)$ denotes an $X$ and its list of transaction $t(X)$
Optimizations

- Implementation with diffsets (**dCharm**).
- Implementation with bit vectors.
- Use a **triangular matrix** to count the support of itemsets of size 2 by scanning the database once.

![Triangular Matrix Illustration](image)

Illustration: szathmary, 2006
Maximal vs Closed Itemsets

- **Closed frequent itemsets**
  - Can recover all frequent itemsets
  - Can recover the support

- **Maximal frequent itemsets**
  - Can recover all frequent itemsets
  - Cannot recover the support
  - A subset of closed itemsets, sometimes much smaller.

Is there other compact representations of frequent itemsets? Yes →
The generator itemsets (or key itemsets)

**Definition:** A (frequent) itemset $X$ is a generator if there exists no (frequent) itemset $Y$ such that $Y \subset X$ and $\text{sup}(X) = \text{sup}(Y)$. 


The generator itemsets (or key itemsets)

**Definition:** A (frequent) itemset $X$ is a **generator** if there exists no (frequent) itemset $Y$ such that $Y \subset X$ and $\sup(X) = \sup(Y)$.

In the example (\textit{minsup} = 2), there are five **frequent generator itemsets**:

- $\{\}$ support: 4
- $\{\text{lemon}\}$ support: 3
- $\{\text{orange}\}$ support: 3
- $\{\text{cake}\}$ support: 2
- $\{\text{lemon,orange}\}$ support: 2
\text{minsup} = 2 \quad \text{The frequent generator itemsets}

\begin{itemize}
\item \text{l} = \text{lemon}
\item \text{p} = \text{pasta}
\item \text{b} = \text{bread}
\item \text{o} = \text{orange}
\item \text{c} = \text{cake}
\end{itemize}

frequent generator itemsets
other frequent itemsets
minsup = 2

The frequent generator itemsets

Some observations:
- The **empty set** can be a generator (if no other itemset appears in all transactions)
- While each equivalence class has always one closed itemset, it could have more than one generator (not in this example)
- In some cases, a closed itemset can be a generator (not in this example)

l = lemon
p = pasta
b = bread
0 = orange
c = cake
Why generators are interesting?

- They are the smallest sets of items that describe a set of transactions. e.g. the smallest sets of products common to a group of customers.
- Some studies have shown that generator patterns can provide better accuracy for classification than maximal or closed itemsets.
Why generators are interesting?

- Also, if we find **generator** and **closed** itemsets, we can use them to generate compact representations of association rules of the form:

  \[
  \text{generator} \rightarrow \text{closed itemset - generator}
  \]

  \[
  \{\text{cake}\} \rightarrow \{\text{pasta}, \text{orange}\} \quad \text{support: 2}
  \]
  \[
  \text{confidence: 100%}
  \]

See the paper about IGB association rules for details.
How to find the generator itemsets?

- **Naïve approach:**
  - by post-processing
  - inefficient!

- **Efficient algorithms:**
  - Pascal, DefMe, etc.
  - Find generator itemsets by avoiding generating all frequent itemsets.
  - **Key search space pruning property:**
    An itemset can only be a generator if all its subsets are also generators.
References – maximal itemsets

**FPMax**

**LCM_Max**
References – closed itemsets

AprioriClose (Close)

- Nicolas Pasquier, Yves Bastide, Rafik Taouil, Lotfi Lakhal: Discovering Frequent Closed Itemsets for Association Rules. ICDT 1999: 398-416

Charm

References – generator itemsets

**DefMe**


**Pascal**


**IGB**

Other references

- Data Mining: The Textbook by Aggarwal (2015)
- Data Mining and Analysis Fundamental Concepts and Algorithms by Zaki & Meira (2014)