Un nouvel algorithme de génération des itemsets fermés fréquents

Huaiguo Fu

CRIL-CNRS FRE2499, Université d’Artois - IUT de Lens
Rue de l’université SP 16, 62307 Lens cedex. France.
E-mail: fu@cril.univ-artois.fr
Motivation

- Develop data mining tools based on formal concept analysis
  - Concept lattices is an effective tool for data analysis
- Develop scalable concept lattice algorithm to reduce expensive cost when mining very large data
  - Problem: very large data still bothers us
  - Situation: applications of concept lattice suffer from the complexity of the concept lattice algorithms for large dataset
  - One solution: divide-and-conquer
Outline

1. Introduction
2. Concept lattice
3. Mining frequent closed itemsets
4. PFC algorithm
5. Conclusion
1 Introduction

2 Concept lattice

3 Mining frequent closed itemsets

4 PFC algorithm

5 Conclusion
The core of formal concept analysis

Derived from mathematical order theory and lattice theory

Study of the generation and relation of formal concepts:

☞ how objects can be hierarchically grouped together according to their common attributes
☞ generate knowledge rules from concept lattice
An effective tool

Concept lattice is an effective tool for

- Data analysis
- Knowledge discovery and data mining
- Information retrieval
- Software engineering
- ...

Introduction
Association rules mining (ARM)
  ☞ Frequent Closed Itemset (FCI) mining
    ➫ CLOSE, A-close, Titanic ...
    ➫ CLOSET, CHARM, CLOSET+ ...

Supervised classification
  ☞ GRAND, LEGAL, GALOIS
  ☞ RULEARNER, CIBLE, CLNN&CLNB ...

Clustering analysis based on concept lattices

Data visualization ...
Many concept lattice algorithms

- Chein (1969)
- Norris (1978)
- Bordat (1986)
- Dowling (1993)
- Kuznetsov (1993)
- Carpineto and Romano (1993, 1996)
- Nourine (1999)
- Lindig (2000)
- Valtchev (2002) ... 

However, large data impedes or limits the application of concept lattice algorithms

We need to develop scalable lattice-based algorithm for data mining
1 Introduction
2 Concept lattice
3 Mining frequent closed itemsets
4 PFC algorithm
5 Conclusion
Definition

**Formal context** is defined by a triple \((O, A, R)\), where \(O\) and \(A\) are two sets, and \(R\) is a relation between \(O\) and \(A\). The elements of \(O\) are called objects or transactions, while the elements of \(A\) are called items or attributes.

Example

Formal context \(D\): \((O, A, R)\)

\(O = \{1, 2, 3, 4, 5, 6, 7, 8\}\)
\(A = \{a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8\}\)

<table>
<thead>
<tr>
<th></th>
<th>a₁</th>
<th>a₂</th>
<th>a₃</th>
<th>a₄</th>
<th>a₅</th>
<th>a₆</th>
<th>a₇</th>
<th>a₈</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>3</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>4</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>5</td>
<td>×</td>
<td>×</td>
<td></td>
<td>×</td>
<td></td>
<td></td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td>×</td>
</tr>
<tr>
<td>7</td>
<td>×</td>
<td></td>
<td>×</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td>×</td>
</tr>
</tbody>
</table>
Definition

Two **closure operators** are defined as $O_1 \rightarrow O_1''$ for set $O$ and $A_1 \rightarrow A_1''$ for set $A$.

$O_1' := \{ a \in A \mid oRa \text{ for all } o \in O_1 \}$

$A_1' := \{ o \in O \mid oRa \text{ for all } a \in A_1 \}$

Definition

A **formal concept** of $(O, A, R)$ is a pair $(O_1, A_1)$ with $O_1 \subseteq O$, $A_1 \subseteq A$, $O_1 = A_1'$ and $A_1 = O_1'$. $O_1$ is called extent, $A_1$ is called intent.
Definition

We say that there is a hierarchical order between two formal concepts \((O_1, A_1)\) and \((O_2, A_2)\), if \(O_1 \subseteq O_2\) (or \(A_2 \subseteq A_1\)). All formal concepts with the hierarchical order of concepts form a complete lattice called **concept lattice**.
Definition

We say that there is a hierarchical order between two formal concepts \((O_1, A_1)\) and \((O_2, A_2)\), if \(O_1 \subseteq O_2\) (or \(A_2 \subseteq A_1\)). All formal concepts with the hierarchical order of concepts form a complete lattice called **concept lattice**.
Mining frequent closed itemsets

1. Introduction
2. Concept lattice
3. Mining frequent closed itemsets
4. PFC algorithm
5. Conclusion
Finding frequent patterns, associations, correlations, or causal structures among itemsets or objects in databases

Two sub-problems

- mining frequent itemsets
- generating association rules from frequent itemsets

Basic Concepts

- Formal context \((O, A, R)\), an itemset \(A_1\)
- support: \(support(A_1) = |A'_1|\)
- frequent itemset: if \(support(A_1) \geq \text{minsupport}\), given a minimum support threshold \(\text{minsupport}\)
Mining frequent closed itemsets

Algorithms of mining frequent itemsets

- Classic algorithm: Apriori and OCD [Agrawal et al., 96]
- Improvement of Apriori
  - AprioriTid,
  - AprioriHybrid
  - DHP (Direct Hashing and Pruning)
  - Partition
  - Sampling
  - DIC (Dynamic Itemset Counting)...
- Other strategies
  - FP-Growth ...
- Problem: too many frequent itemsets if the minsupport is low
  - Solutions:
    - Maximal frequent itemset mining
    - Frequent closed itemset mining
Definition

For formal context \((O, A, R)\), an itemset \(A_1 \subseteq A\) is a **closed itemset** iff \(A''_1 = A_1\).

Definition

We say that there is a hierarchical order between closed itemsets \(A_1\) and \(A_2\), if \(A_1 \subseteq A_2\). All closed itemsets with the hierarchical order form of a complete lattice called **closed itemset lattice**.
FCI mining algorithms

- Apriori-like algorithms
  - Close
  - A-Close
  - ...

- Hybrid algorithms
  - CHARM
  - CLOSET
  - CLOSET+
  - FP-Close
  - DCI-Closed
  - LCM
  - ...

Mining frequent closed itemsets
Introduction

Concept lattice

Mining frequent closed itemsets

PFC algorithm

Conclusion
Analysis of search space

- Search space of a formal context with 4 attributes

- Search space decomposition

```
A_m

A_{m-1}  A_{m-2}  A_{m-3}

am am-1 am-2 am-3

am-1 am am-1 am-2 am-3 am-2 am-1 am-3 am-3 am-2 am-1

am-3 am-2 am-1 am
```

```
A_{m-3}  A_{m-2}  A_{m-1}  A_{m-3}

am-1 am am-1 am-2 am-3 am-2 am-1 am-3 am-3 am-2 am-1

am-3 am-2 am-1 am
```
Analysis of search space

- Search space of a formal context with 4 attributes

- Search space decomposition
Analysis of search space

- Search space of a formal context with 4 attributes

- Search space decomposition
Definition

Given an attribute \( a_i \in A \) of the context \( (O, A, R) \), a set \( E \), \( a_i \not\in E \). We define \( a_i \otimes E = \{\{a_i\} \cup X \text{ for all } X \subseteq E\} \).

\[
A_k = \{a_k\} \cup \left( \bigcup_{\forall X_i \in \bigcup A_j} \{\{a_k\} \cup X_i\} \right)
\]

\[
= a_k \otimes \{a_{k+1}, a_{k+2}, \ldots, a_m\} \quad k + 1 \leq j \leq m
\]

Problems:

☞ how to balance the size of partitions
☞ whether each partition contains closed itemsets
A strategy of decomposition

- **Partitions** (for the preceding example: formal context $D$)

  - $partition_1$
    - $A_8$
    - $A_7$
    - $A_6$
    - $A_5$
  - $partition_2$
    - $A_4$
    - $A_3$
  - $partition_3$
    - $A_2$
  - $partition_4$
    - $A_1$

- **Results:** not good!
  - $Nb_c(partition_1) = 0$
  - $Nb_c(partition_2) = 0$
  - $Nb_c(partition_3) = 0$
  - $Nb_c(partition_1) = 17$

- **A solution:** order the attributes
  - $Nb_c(partition_1) = 6$
  - $Nb_c(partition_2) = 6$
  - $Nb_c(partition_3) = 4$
  - $Nb_c(partition_1) = 1$
A formal context is called **ordered data context** if we order the items of formal context by number of objects of each item from the smallest to the biggest one, and the items with the same objects are merged as one item.

**Example**

<table>
<thead>
<tr>
<th></th>
<th>a1</th>
<th>a2</th>
<th>a3</th>
<th>a4</th>
<th>a5</th>
<th>a6</th>
<th>a7</th>
<th>a8</th>
<th>a9</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>2</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>3</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>4</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>5</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>6</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>7</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>8</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
</tbody>
</table>

Formal context

<table>
<thead>
<tr>
<th></th>
<th>a5</th>
<th>a8</th>
<th>a6</th>
<th>a7</th>
<th>a4</th>
<th>a3</th>
<th>a2</th>
<th>a1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>2</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>3</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>4</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>5</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>6</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>7</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>8</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
</tbody>
</table>

Ordered data context
Definition

An item $a_i$ of a formal context $(O, A, R)$, all subsets of 
\{a_i, a_{i+1}, \ldots, a_{m-1}, a_m\} that include $a_i$, form a search sub-space (for closed itemset) that is called folding search sub-space (F3S) of $a_i$, denoted $F3S_i$.

Remark

The search space of closed itemsets:
$F3S_m \cup F3S_{m-1} \cup F3S_{m-2} \cup F3S_{m-3} \cup \cdots \cup F3S_i \cdots \cup F3S_1 \cup \emptyset$
PFC algorithm

An example

\[
(O, A, R) \implies (O, A^\prec, R), \text{ Ordered data context: } a_5 \ a_8 \ a_6 \ a_7 \ a_4 \ a_3 \ a_2 \ a_1
\]

**step 1:**

\[
DP = 0.5 \implies p_1 = 8, \ p_2 = 4, \ p_3 = 2, \ p_4 = 1 \implies p_1 \to a_1, \ p_2 \to a_7, \ p_3 \to a_8, \ p_4 \to a_5
\]

**step 2:**

**Partitions:**

- choose: \( a_4 \ a_3 \ a_2 \ a_1 \)
- remain: \( a_5 \ a_8 \ a_6 \ a_7 \)

- choose: \( a_6 \ a_7 \)
- remain: \( a_5 \ a_8 \)

- choose: \( a_8 \)
- remain: \( a_5 \)

- choose: \( a_5 \)
- remain: \( \emptyset \)

**step 3:**

Search the closed itemsets in each partition separately:

- \([a_1, a_7]: a_1, a_2a_1, a_3a_1, a_3a_2a_1, a_4a_1, a_4a_3a_1\)
- \([a_7, a_8]: a_7a_1, a_7a_2a_1, a_7a_4a_1, a_6a_4a_2a_1, a_6a_4a_3a_1, a_6a_4a_3a_2a_1\)
- \([a_8, a_5]: a_8a_1, a_8a_7a_2a_1, a_8a_7a_3a_1, a_8a_7a_3a_2a_1\)
- \([a_5]: a_5a_4a_3a_1\)
**PFC algorithm**

An example

<table>
<thead>
<tr>
<th>step 1:</th>
<th>((O, A, R) \rightarrow (O, A^&lt;, R),) Ordered data context: (a_5 \ a_8 \ a_6 \ a_7 \ a_4 \ a_3 \ a_2 \ a_1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(DP = 0.5 \rightarrow p_1 = 8, p_2 = 4, p_3 = 2, p_4 = 1 \rightarrow p_1 \rightarrow a_1, p_2 \rightarrow a_7, p_3 \rightarrow a_8, p_4 \rightarrow a_5)</td>
<td></td>
</tr>
</tbody>
</table>

**Partitions:**

- **choose:** \(a_4 \ a_3 \ a_2 \ a_1\)
- **remain:** \(a_5 \ a_8 \ a_6 \ a_7\)

- **choose:** \(a_6 \ a_7\)
- **remain:** \(a_5 \ a_8\)

- **choose:** \(a_8\)
- **remain:** \(a_5\)

- **choose:** \(a_5\)
- **remain:** \(\emptyset\)

**Search the closed itemsets in each partition separately:**

- \([a_1, a_7]: a_1, a_2 a_1, a_3 a_1, a_3 a_2 a_1, a_4 a_1, a_4 a_3 a_1\)
- \([a_7, a_8]: a_7 a_1, a_7 a_2 a_1, a_6 a_4 a_1, a_6 a_4 a_2 a_1, a_6 a_4 a_3 a_1, a_6 a_4 a_3 a_2 a_1\)
- \([a_8, a_5]: a_8 a_1, a_8 a_2 a_1, a_8 a_7 a_3 a_1, a_8 a_7 a_3 a_2 a_1\)
- \([a_5]: a_5 a_4 a_3 a_1\)

**Diagram:**

1. **Partition 1:** \([a_1, a_7]: a_1 \rightarrow a_4\)
2. **Partition 2:** \([a_7, a_8]: a_7 \rightarrow a_6\)
3. **Partition 3:** \([a_8, a_5]: a_8\)
4. **Partition 4:** \([a_5]: a_5\)
PFC algorithm

An example

**step 1:** \((O, A, R) \Rightarrow (O, A^\prec, R)\), Ordered data context: \(a_5 \ a_8 \ a_6 \ a_7 \ a_4 \ a_3 \ a_2 \ a_1\)

\[DP = 0.5 \Rightarrow p_1 = 8, \ p_2 = 4, \ p_3 = 2, \ p_4 = 1 \Rightarrow p_1 \rightarrow a_1, \ p_2 \rightarrow a_7, \ p_3 \rightarrow a_8, \ p_4 \rightarrow a_5\]

**step 2:**

**Partitions:**
- **choose:** \(a_4 \ a_3 \ a_2 \ a_1\)
- **remain:** \(a_5 \ a_8 \ a_6 \ a_7\)
- **choose:** \(a_6 \ a_7\)
- **remain:** \(a_5 \ a_8\)
- **choose:** \(a_8\)
- **remain:** \(a_5\)
- **choose:** \(a_5\)
- **remain:** \(\emptyset\)

Search the closed itemsets in each partition separately:
- \([a_1, a_7]: a_1, a_2a_1, a_3a_1, a_3a_2a_1, a_4a_1, a_4a_3a_1\)
- \([a_7, a_8]: a_7a_1, a_7a_2a_1, a_6a_4a_1, a_6a_4a_2a_1, a_6a_4a_3a_1, a_6a_4a_3a_2a_1\)
- \([a_8, a_5]: a_8a_7a_1, a_8a_7a_2a_1, a_8a_7a_3a_1, a_8a_7a_3a_2a_1\)
- \([a_5]: a_5a_4a_3a_1\)
An example

step 1: 

\[(O, A, R) \implies (O, A^<, R),\] Ordered data context: \(a_5 \ a_8 \ a_6 \ a_7 \ a_4 \ a_3 \ a_2 \ a_1\)

\(\text{DP} = 0.5 \implies p_1 = 8, \ p_2 = 4, \ p_3 = 2, \ p_4 = 1 \implies p_1 \to a_1, \ p_2 \to a_7, \ p_3 \to a_8, \ p_4 \to a_5\)

\[
\begin{align*}
\text{Partitions:} \\
\text{choose:} & \quad a_4 \ a_3 \ a_2 \ a_1 \\
\text{remain:} & \quad a_5 \ a_8 \ a_6 \ a_7 \\
\text{choose:} & \quad a_6 \ a_7 \\
\text{remain:} & \quad a_5 \ a_8 \\
\text{choose:} & \quad a_8 \\
\text{remain:} & \quad a_5 \\
\text{choose:} & \quad a_5 \\
\text{remain:} & \quad \emptyset
\end{align*}
\]

Search the closed itemsets in each partition separately:

\[
\begin{align*}
[a_1, a_7] & : a_1, a_2 a_1, a_3 a_1, a_3 a_2 a_1, a_4 a_1, a_4 a_3 a_1 \\
[a_7, a_8] & : a_7 a_1, a_7 a_2 a_1, a_6 a_4 a_1, a_6 a_4 a_2 a_1, a_6 a_4 a_3 a_1, a_6 a_4 a_3 a_2 a_1 \\
[a_8, a_5] & : a_8 a_7 a_1, a_8 a_7 a_2 a_1, a_8 a_7 a_3 a_1, a_8 a_7 a_3 a_2 a_1 \\
[a_5] & : a_5 a_4 a_3 a_1
\end{align*}
\]
step 1: \((O, A, R) \implies (O, A^\prec, R)\), Ordered data context: \(a_5\ a_8\ a_6\ a_7\ a_4\ a_3\ a_2\ a_1\)

\[DP = 0.5 \implies p_1 = 8, \ p_2 = 4, \ p_3 = 2, \ p_4 = 1 \implies p_1 \rightarrow a_1, \ p_2 \rightarrow a_7, \ p_3 \rightarrow a_8,\]
\[p_4 \rightarrow a_5\]

\begin{align*}
1 & \ 2 & 3 & 4 & 5 & 6 & 7 & 8 \\
\ 1 & a_5 & a_8 & a_6 & a_7 & a_4 & a_3 & a_2 & a_1 \\
\ p_4=1 & & & & & & & & \\
\ p_3=2 & & & & & & & & \\
\ p_2=4 & & & & & & & & \\
\ p_1=8 & & & & & & & &
\end{align*}

Partitions:
choose: \([a_4\ a_3\ a_2\ a_1]\)
remain: \(a_5\ a_8\ a_6\ a_7\)
choose: \([a_6\ a_7]\)
remain: \(a_5\ a_8\)
choose: \([a_8]\)
remain: \(a_5\)
choose: \([a_5]\)
remain: (empty)

Search the closed itemsets in each partition separately:

\([a_1, a_7]\): \(a_1, \ a_2a_1, \ a_3a_1, \ a_3a_2a_1, \ a_4a_1, \ a_4a_3a_1\)

\([a_7, a_8]\): \(a_7a_1, \ a_7a_2a_1, \ a_6a_4a_1, \ a_6a_4a_2a_1, \ a_6a_4a_3a_1, \ a_6a_4a_3a_2a_1\)

\([a_8, a_5]\): \(a_8a_7a_1, \ a_8a_7a_2a_1, \ a_8a_7a_3a_1, \ a_8a_7a_3a_2a_1\)

\([a_5]\): \(a_5a_4a_3a_1\)
PFC algorithm has two steps:

1) Determining the partitions
2) Generating independently all frequent closed itemsets of each partition

Failure \[\Rightarrow\] Success

\textit{divide-and-conquer}
PFC algorithm

Determining the partitions

- \((O, A, R) \Rightarrow (O, A^{\prec}, R)\) where
  \[ A^{\prec} = \{ a_1^{\prec}, a_2^{\prec}, \ldots, a_i^{\prec}, \ldots, a_m^{\prec} \} \]

- Some items of \(A^{\prec}\) are chosen to form an order set \(P\)
  \[ P = \{ a_{P_T}^{\prec}, a_{P_T-1}^{\prec}, \ldots, a_{P_k}^{\prec}, \ldots, a_{P_1}^{\prec} \}, |P| = T, a_{P_k}^{\prec} \in A^{\prec} \]
  \[ a_{P_T}^{\prec} < a_{P_T-1}^{\prec} < \ldots < a_{P_k}^{\prec} < \ldots < a_{P_2}^{\prec} < a_{P_1}^{\prec} = a_m^{\prec} \]

- \(DP\) is used to choose \(a_{P_k}^{\prec}\) (\(0 < DP < 1\))
  \[ DP = \frac{|\{a_1^{\prec}, \ldots, a_{P_k}^{\prec}\}|}{|\{a_1^{\prec}, \ldots, a_{P_k-1}^{\prec}\}|} \]

- The partitions: \([a_{P_k}^{\prec}, a_{P_k+1}^{\prec}]\) and \([a_{P_T}^{\prec}]\)
  
  - Interval \([a_{P_k}, a_{P_k+1}]\) is the search space from item \(a_{P_k}\) to \(a_{P_k+1}\)
  
  \[
  \left[ a_{P_k}^{\prec}, a_{P_k+1}^{\prec} \right] = \bigcup_{P_k \leq i < P_{k+1} \leq P_T} (F3S_i) \\
  \left[ a_{P_T}^{\prec} \right] = F3S_{P_T}
  \]
Definition:

Given an itemset $A_1 \subset A$, $A_1 = \{b_1, b_2, \ldots, b_i, \ldots, b_k\}$, $b_i \in A$. $A_1$ is an infrequent itemset. The candidate of next closed itemset of $A_1$, noted $C_{A_1}^{\cap}$, is $A_1 \cup a_i = (A_1 \cap (a_1, a_2, \ldots, a_{i-1}) \cup \{a_i\})''$, where $a_i < b_k$ and $a_i \notin A_1$, $a_i$ is the biggest one of $A$ with $A_1 < A_1 \cup a_i$ following the order: $a_1 < \ldots < a_i < \ldots < a_m$.

- Partitions are independent and non-overlapped
- For each partition, from an itemset $A_1$
  - If $|A'_1| \geq \text{minsupport}$, we search the next closure of $A_1$
  - If $|A'_1| < \text{minsupport}$, we search $C_{A_1}^{\cap}$. The closed itemsets between $A_1$ and $C_{A_1}^{\cap}$ are ignored
An example

Search FCIs with PFC

\[ a_{m-3} a_{m-1} a_{m-2} a_{m-1} a_m \]
Extended comparison of [BenYahia and Mephu, 04]

Comparison

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Format</th>
<th>Technique</th>
<th>Architecture</th>
<th>Parallelism</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-CLOSE</td>
<td>horizontal</td>
<td>Generate-and-test</td>
<td>Sequential</td>
<td>no</td>
</tr>
<tr>
<td>CLOSET+</td>
<td>horizontal</td>
<td>Hybrid, FP-tree</td>
<td>Sequential</td>
<td>no</td>
</tr>
<tr>
<td>CHARM</td>
<td>vertical</td>
<td>Hybrid, IT-tree, diffset</td>
<td>Sequential</td>
<td>no</td>
</tr>
<tr>
<td>PFC</td>
<td>relational</td>
<td>Hybrid, partition</td>
<td>Distributed</td>
<td>yes</td>
</tr>
</tbody>
</table>
Many FCI algorithms were compared in [FIMI03]
In order to get a reference of the performance of PFC
C++ for CLOSET+ vs Java for PFC
Sequential architecture

<table>
<thead>
<tr>
<th>Data</th>
<th>Objects</th>
<th>Items</th>
<th>Minsupport</th>
<th>FCI</th>
<th>PFC (msec)</th>
<th>CLOSET+</th>
<th>Ref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>audiology</td>
<td>26</td>
<td>110</td>
<td>1</td>
<td>30401</td>
<td>1302</td>
<td>8563</td>
<td>+</td>
</tr>
<tr>
<td>soybean-small</td>
<td>47</td>
<td>79</td>
<td>1</td>
<td>3541</td>
<td>516</td>
<td>431</td>
<td>-</td>
</tr>
<tr>
<td>lung-cancer</td>
<td>32</td>
<td>228</td>
<td>1</td>
<td>186092</td>
<td>21381</td>
<td>279689</td>
<td>+</td>
</tr>
<tr>
<td>promoters</td>
<td>106</td>
<td>228</td>
<td>3</td>
<td>298772</td>
<td>120426</td>
<td>111421</td>
<td>-</td>
</tr>
<tr>
<td>soybean-large</td>
<td>307</td>
<td>133</td>
<td>1</td>
<td>806030</td>
<td>357408</td>
<td>364524</td>
<td>+</td>
</tr>
<tr>
<td>dermatology</td>
<td>366</td>
<td>130</td>
<td>50</td>
<td>192203</td>
<td>20204</td>
<td>18387</td>
<td>-</td>
</tr>
<tr>
<td>breast-cancer-wis</td>
<td>699</td>
<td>110</td>
<td>1</td>
<td>9860</td>
<td>3529</td>
<td>1131</td>
<td>-</td>
</tr>
<tr>
<td>kr-vs-kp</td>
<td>3196</td>
<td>75</td>
<td>1100</td>
<td>2770846</td>
<td>1823092</td>
<td>483896</td>
<td>-</td>
</tr>
<tr>
<td>agaricus-lepiota</td>
<td>8124</td>
<td>124</td>
<td>100</td>
<td>38347</td>
<td>34815</td>
<td>1462</td>
<td>-</td>
</tr>
<tr>
<td>connect-4</td>
<td>67557</td>
<td>126</td>
<td>1000</td>
<td>2447136</td>
<td>1165806</td>
<td>65084</td>
<td>-</td>
</tr>
</tbody>
</table>

Discussion
- 3 contexts with good results
  - Density
  - or |A| >> |O|
Introduction

Concept lattice

Mining frequent closed itemsets

PFC algorithm

Conclusion
Conclusion

- PFC: a scalable algorithm
  - PFC algorithm can be used to generate FCIs
  - Preliminary performance results show that PFC is suitable for large data

- One limitation of PFC: formal context can be loaded in memory

- Perspectives
  - Analyze context characteristic
  - Optimize the parameter of algorithms
  - Parallel and distributed environment
  - Application