A Novel Algorithm for Building Concept Lattice

Jirapond Muangprathub

Faculty of Science and Industrial Technology
Prince of Songkla University
Suratthani Campus, Suratthani, Thailand 84000

Copyright © 2014 Jirapond Muangprathub. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Abstract

Concept lattice is a method for data analysis in order to describe a natural attributes of information representation in hierarchical structure model. The key success for knowledge representation with concept lattice is the algorithm to generate the set of all formal concepts and then their construction. Thus, this paper improves and focuses on building concept lattice with providing theoretical FCA from presenting algorithm. The complexity of proposed algorithm for building the set of formal concepts is quadratic function. In addition, the complexity of algorithm for building concept lattice is $O(|F| + m^2 \ast p)$. The result shows that our proposed is better than mostly works by using time complexity analysis.

Keywords: Knowledge representation, Concept lattice, Formal concept lattice, Lattice construction

1 Introduction

Concept lattice has been successfully applied in many domains such as data mining [17], ontology [2], information science [26], machine learning [22], case-based reasoning (CBR) [8], software engineering [24], etc. It can be employed in these domains to obtain explicit and implicit knowledge from their representation. Explicit knowledge can describe attributes and objects of information represented in the hierarchical structure model while implicit knowledge can elicit knowledge dependency. Some domains essentially require incremental
structure to facilitate dynamic databases. In effect, algorithm to grow concept lattice and its incremental construction are both equally key successes for knowledge representation in each domain.

Traditional, concept lattice algorithms can be grouped into: build a concept lattice from the obtained formal concept and incrementally constructed concept lattice. For example, algorithms of Bordat [14, 16, 21], Ganter [3], Close by One [20], Nourine [15], and Lindig [7] are developed to enhance performance of their algorithm that is not addressed on incremental concept lattice construction. In contrast, algorithms of Godin [18], Dowling [6], and AddIntent [9, 10] avoid repeated generation of all formal concept with incremental structure. However, some applications require both of them such as CBR system. If CBR system represent knowledge with concept lattice, the first step of such CBR process all of previous experiences as constructing concept lattice while next step using incremental concept lattice construction for retaining the new situation. In previous work [13], we addressed algorithm for incremental construction of knowledge. Thus, this paper improves and focuses on building concept lattice with providing theoretical FCA from presenting algorithm.

This article is organized as follows. In Section 2 provides basic notions of FCA. Section 3, we briefly review related works of constructing concept lattice and incremental concept lattice construction. Section 4 presents the remarks to achieve completely concept lattice structure and proposed algorithms. In addition, the discussion and analysis the proposed algorithm are presented in this section. Finally, Section 5 concludes the article.

2 Preliminary Notes

In this section we provide the basic notions and definitions of concept lattice in FCA taken from [4].

Definition 2.1 A formal context $K := (G, M, I)$ consists of two sets $G$ and $M$ and a relation $I$ between $G$ and $M$. The elements of $G$ are called the objects and the elements of $M$ are called the attributes of the context. In order to express that an object $g$ is in a relation $I$ with an attribute $m$, we write $gI m$ or $(g, m) \in I$ and read it as “the object $g$ has the attribute $m$”.

Definition 2.2 For a set $A \subseteq G$ of objects we define

$$A' := \{m \in M | gI m \text{ for all } g \in A\}$$

(the set of attributes common to the objects in $A$). Correspondingly, for a set $B$ of attributes we define

$$B' := \{g \in G | gI m \text{ for all } m \in B\}$$

(the set of objects which have all attributes in $B$).
Definition 2.3 A formal concept of the formal context \((G, M, I)\) is a pair \((A, B)\) with \(A \subseteq G\), \(B \subseteq M\), \(A' = B\) and \(B' = A\). We call \(A\) the extent and \(B\) the intent of the formal concept \((A, B)\). \(\mathcal{B}(G, M, I)\) denotes the set of all formal concepts of the formal context \((G, M, I)\).

Definition 2.4 If \((A_1, B_1)\) and \((A_2, B_2)\) are formal concepts of formal context, \((A_1, B_1)\) is called a subconcept of \((A_2, B_2)\), provided that \(A_1 \subseteq A_2\) (which is equivalent to \(B_2 \subseteq B_1\)). In this case, \((A_2, B_2)\) is a superconcept of \((A_1, B_1)\), and we write \((A_1, B_1) \leq (A_2, B_2)\). The relation \(\leq\) is called the hierarchical order (or simply order) of the formal concepts. The set of all formal concepts of \((G, M, I)\) ordered in this way denoted by \(\mathcal{B}(G, M, I)\) and is called the concept lattice of the formal context \((G, M, I)\).

3 Related Works

Concept lattice is a fundamental structure in FCA. Construction of concept lattice consists of generating of all formal concepts and building concept lattice from the generated formal concept. There are many algorithms surveyed in [5, 11, 16, 21]. In this section, we can summarize the review of algorithms for concept lattice construction based on idea of each algorithm and its time complexity.

Ganter is initially developed NextClosure algorithms [3]. The idea of this algorithm is that the finding set of formal concepts using closures is based on ordering subsets of objects that only some subsets needs to be examined. The time complexity for generating all formal concept is \(O(|G||C||M|)\) where \(G\) is objects, \(M\) is attributes, and \(C\) is formal concepts. One advantage of NextClosure algorithm is that every formal concept is created only once. The drawback is that it does not build concept lattice construction. Moreover, this algorithm is improved memory size by using dichotomic search [11]. Although, memory size better than original version but the time complexity is \(O(|G|^2|C||M|)\).

Bordat’s algorithm as mentioned in [14, 16, 21] used attribute cache and computed intents by subsequently computing intersections of object intents. The time complexity is \(O(|G||C||M|^2)\). Lindig proposed algorithm that computes formal concepts together with concept lattice structure [7]. The idea of this algorithm is that it begins from top formal concept as the first level. The next level will contain children where each child node is computed the lower neighbours that are subset of its parent. Its time complexity is \(O(|G||C||M|^2)\).

Close by one’s algorithms was proposed by Kuznetsov [20]. This algorithm uses a similar notion of canonicity, a similar method for selecting subsets, and computes intersections of non-object intent and object intents. Its time complexity is \(O(|G|^2|C||M|)\). Next, Andrews [19] improve this algorithm. In-Close algorithm was proposed based on incremental closure and matrix searching.
The author represented formal context in matrix form to provide searching of all formal concept. The input is associated to currently lattice by searching formal context. Thus, this algorithm will face to the complexity of searching.

The next algorithms based on intersection among the sets of objects of each attributes. [12, 15, 16, 23]. These algorithms find all formal concepts from initial attribute extents. Each formal concept is examined by intersection between its lower neighbours. The limitation of these algorithms is that the same formal concepts can be generated several times. The time complexity in [15], [16], and [23] are $O(|C||M|(|G| + |M|))$, $O(|M||G| + |M|))$, and $O(|C||M|(|G| + |M|))$, respectively.

Another algorithm for generating of all formal concepts and building concept lattice was proposed by Choi [25] and Troy et. al. [1]. Choi [25] proposed the fast algorithm for generating along formal concepts with building concept lattice based on frequent closed itemset. The iterative of examining for each formal concept is occurred and requires space for generating closed itemsets. Moreover, this algorithm faces to delete formal concept that is not closed that enhance more times. The main advantage of this algorithm is that both formal concept and concept lattice are built together. Troy et. al. [1] presented multitage algorithm for constructing concept lattice (MCA). There are two strategies such that intersections by joining one attribute concept at a time and repeatedly forming pairwise intersections starting from the attribute concepts.

4 Algorithms for Building Concept Lattice

In this section, we provide theoretical FCA and propose the algorithm for knowledge representation. To build completely construction of FCA based on concept lattice, in this paper present formal concept generation and concept lattice construction developed incremental concept lattice from our previous work [13].

4.1 Algorithm for constructing formal concepts

The set of formal concepts is achieved from the formal context that identifies groups of objects with some common attributes. Form definitions 2.3, each formal concept of formal context $(G, M, I)$ has the form $(A', A'')$ for some subset $A \subseteq G$ and the form $(B', B'')$ for some subset $B \subseteq M$. Conversely all such pairs are formal concepts. The implies every extension is the intersection of attribute extensions and every intension is the intersection of object intensions.

**Remark 4.1** If a pair $(X, Y)$ be formal concept $\mathfrak{B}(G, M, I)$, then $F_X$, $F_Y$ are a family of extent and intent in $(G, M, I)$ where $F_X = \{ \bigcup X | X = \bigcap m', \forall m \in M \}$ and $F_Y = \{ \bigcup Y | Y = X' \}$. 
From Remark 4.1, we can apply it to generate formal concepts \( \mathfrak{B}(G, M, I) \) with our proposed algorithm following Algorithm 1. This algorithm consists of two parts (i.e., extent and intent), where input data is formal context. First part, \( \text{FindExtentX()} \) is function to generate a family of extent following as remark 4.1. This function consist of finding \( m' \) and a family of extent. We estimate the time complexity with \( O(|m| + \frac{m(m-3)}{2}) \). The front operation derived from finding \( m' \) with \( |m| \). The back operation is estimated with intersection between two different attributes. This operation is selection two things as \( \left( \frac{n!}{(n-2)!2!} \right) \) where \( n \) is amount of all things and eliminate itself as \( n \).

The rest part is to find intent of formal concepts as line 5. This operation is estimated with \( O(|m| + \frac{m(m-3)}{2}) \) because of intersection operation of above derived extent. Thus, the time complexity for constructing of the set of formal concepts is \( O(|m|^2 - 3m + 2) \), quadratic function.

**Algorithm 1** Algorithm for constructing of the set of formal concepts

<table>
<thead>
<tr>
<th>Input</th>
<th>The formal context ((G, M, I)).</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output</td>
<td>Set of the formal concepts ( \mathfrak{B}(G, M, I) )</td>
</tr>
<tr>
<td>Method :</td>
<td></td>
</tr>
<tr>
<td>1.</td>
<td>( F_X = \text{FindExtentX}() );</td>
</tr>
<tr>
<td>2.</td>
<td>( F_Y = \emptyset );</td>
</tr>
<tr>
<td>3.</td>
<td>For ( i=0 ) to (</td>
</tr>
<tr>
<td>4.</td>
<td>{ //find intersection from extent ( X[i] )</td>
</tr>
<tr>
<td>5.</td>
<td>( F_Y = F_Y \cup F_X[i]' );</td>
</tr>
<tr>
<td>6.</td>
<td>} //Return formal concept ((X,Y))</td>
</tr>
</tbody>
</table>

**function** \( \text{FindExtentX()} \) // input is formal context \((G, M, I)\).  
1. For \( i=0 \) to \( |M| \)  
2. {  
3. \( F_{X,initial} = m' \);  
4. \( F_X = F_{X,initial} \);  
5. }  
6. For \( j=0 \) to \( |F_{X,initial}| - 1 \)  
7. For \( k=j \) to \( |F_{X,initial}| - 1 \)  
8. {  
9. \( \text{IntersFx} = F_{X,initial}[i] \cap F_{X,initial}[j] \);  
10. If \( \text{IntersFx} \notin F_{X,initial} \)  
11. \( F_X = \bigcup \text{IntersFx} \);  
12. } // Return a family of extent \( F_X \)

### 4.2 Construction of concept lattice

To build the relation (or link) among formal concepts into concept lattice, the algorithm for constructing this concept lattice is required according to definition 2.4. To achieve of concept lattice, the remark 4.2 is proposed to support for building the relation among formal concepts.
Remark 4.2 Let $F$ be a family of extent in $\mathfrak{B}(G, M, I)$, $(X_1, Y_1), (X_2, Y_2) \in \mathfrak{B}$ and $X_1, X_2 \in F$, then $(X_1, Y_1) \leq (X_2, Y_2) \iff |X_1| \leq |X_2|$ and $X'_1 \cap X'_2 \neq \emptyset$.

From Remark 4.2, we can apply to construct concept lattice. The levels of concept lattice are considered from size of extent. Starting from the bottom element of concept lattice, the algorithm builds the first level by considering size of extent of formal concept with equal (1). The next level contains parents of all formal concepts in current level. For each formal concept in current level (child), parent is linked when the intersection between attribute extent of child and parent is not empty set. However, if child is not linked with parents in next level, it will be added into parent in this next level and then these parents will be children in future. The complexity of this algorithm $O(|F| + |m^2*p|)$, where $|F|$ derive from $\text{FindSetExtLength}()$ that find length of each case intent in $\mathfrak{B}(G, M, I)$.

**Algorithm 2** Algorithm for construction of concept lattice

<table>
<thead>
<tr>
<th>Input</th>
<th>Set of the formal concepts $\mathfrak{B}(G, M, I)$.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output</td>
<td>Concept lattice $\mathfrak{B}(G, M, I)$</td>
</tr>
<tr>
<td>Method</td>
<td></td>
</tr>
<tr>
<td>1.</td>
<td>$\text{Bottom} = (G, \emptyset)$;</td>
</tr>
<tr>
<td>2.</td>
<td>$\text{currentLevel} = \emptyset$;</td>
</tr>
<tr>
<td>3.</td>
<td>BufLevel = $\text{FindSetExtLength}()$; // SetLength is sorted by ascending</td>
</tr>
<tr>
<td>4.</td>
<td>if (BufLevel.SetLength = 1)</td>
</tr>
<tr>
<td>5.</td>
<td>$\mathfrak{B} \rightarrow$ Add link into Bottom;</td>
</tr>
<tr>
<td>6.</td>
<td>$\text{currentLevel} = \text{BufLevel.GroupSameLength}[0]$;</td>
</tr>
<tr>
<td>7.</td>
<td>// The first group length is 1</td>
</tr>
<tr>
<td>8.</td>
<td>For i = 1 to n // n = size of SetLength = $</td>
</tr>
<tr>
<td>9.</td>
<td>For j = 0 to m // m = size of currentLevel</td>
</tr>
<tr>
<td>10.</td>
<td>For k = 0 to p // p = size of group length is i = $</td>
</tr>
<tr>
<td>11.</td>
<td>if $X'_k \cap X'_j \neq \emptyset$</td>
</tr>
<tr>
<td>12.</td>
<td>$\mathfrak{B} \rightarrow$ Add link into Bottom;</td>
</tr>
<tr>
<td>13.</td>
<td>//Return concept lattice $\mathfrak{B}$</td>
</tr>
</tbody>
</table>

**function** $\text{FindSetExtLength}()$ // input is formal context $(G, M, I)$.

| 1.      | $\text{SetLength} = \emptyset$;                  |
| 2.      | $\text{GroupSameLength} = \emptyset$;            |
| 3.      | For i = 1 to $|F|$ //Find length of extent in formal concept (SetLength) |
| 4.      | { //and group same length of each length (GroupSameLength) |
| 5.      | If $|X_i| \notin \text{SetLength}$               |
| 6.      | { // add new length of extent $X_i$ and $X_i$ into same length such that |
| 7.      | $|X_i| = \text{SetLength}$;                      |
| 8.      | $X_i = \text{GroupSameLength}$;                  |
| 9.      | }                                                |
| 10.     | } //Return SetLength and GroupSameLength         |
The proposed algorithms for generating of the set of formal concepts have the time complexity as $O(|m|^2 - 3m + 2)$. Namely, this proposed algorithm is quadratic function while the mostly algorithms is polynomial function. Our algorithm support data type in redundancy of either 1 or 0 in formal context input.

For the propose algorithm for building concept lattice construction, the complexity time is $O(|F| + |m^2 \times p|)$, where $|F|$ is length of each case intent in $\mathfrak{B}(G, M, I)$ and $p$ is size of members in each formal concept. Mostly, the previous works spend more time because this step involves set of attribute, object, formal concept, and so on. To consider our complexity time, we found that our algorithm is better than some previous algorithm with comparing variable in function.

5 Conclusion

Concept lattice is widely used for many applications to provides relationship of generalization and specialization among formal concepts. From a practical point of view, the initially task of achieving knowledge from concept lattice structure is to obtain the set of formal concepts and to build construction with them. Thus, we propose remarks to achieve completely concept lattice structure. To illustrate the proposed methods, we present algorithms and discussion with previous works by using time complexity analysis. The result shows that our proposed is better than mostly works as quadratic function in generating the set of formal concepts. Afterwards, the obtained formal concepts are combined into concept lattice with the complexity time is $O(|F| + |m^2 \times p|)$, where $|F|$ is length of each case intent in $\mathfrak{B}(G, M, I)$ and $p$ is size of members in each formal concept.

Acknowledgements. The authors are deeply grateful to Faculty of Science and Industrial Technology, Prince of Songkla University, Thailand. This research was financially supported by Prince of Songkla University, Surat Thani Campus and Hat Yai Campus, 2013.

References


Received: December 4, 2013