Knowledge-Based Systems 89 (2015) 411–419

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

A fast incremental algorithm for deleting objects from a concept lattice

Ligeng Zou, Zuping Zhang *, Jun Long, Hao Zhang

School of Information Science and Engineering, Central South University, Changsha, Hunan 410083, China

ARTICLE INFO

Article history:
Received 16 February 2015
Received in revised form 10 July 2015
Accepted 19 July 2015
Available online 23 July 2015

Keywords:
Formal concept analysis
Concept lattice
Incremental algorithm
Formal context reduction

ABSTRACT

The formal context may not be fixed in a real-life application of formal concept analysis, which means that we have to update the present lattice or compute a new lattice from scratch. In this paper, we propose an efficient incremental algorithm, referred to as FastDeletion, to delete objects from a concept lattice. The algorithm improves two fundamental procedures shared by other algorithms. These two procedures include determining which concepts need to be deleted and fixing the covering relation. We describe the algorithm thoroughly, prove correctness of our improvements, discuss time complexity issues, and present an experimental evaluation of its performance and comparison with another algorithm. Empirical analyses demonstrate that our algorithm is superior when applied to various types of formal contexts.

1. Introduction

Formal Concept Analysis (FCA) was introduced in the early 1980s by Rudolf Wille [1–3]. It has been considered as a powerful tool to analyze object-attribute relational data and discover knowledge. FCA has been widely used in various disciplines such as data mining [4], software engineering [5], linguistics [6], ontology engineering [7], bioinformatics [8,9] and information retrieval [10,11]. Readers can refer to [12] for an extensive overview of FCA-based methods in different application domains.

As the underlying structure of FCA, concept lattice is of solid mathematical foundations and it is capable of visualizing partially ordered concepts. For the last twenty years, many efficient algorithms for constructing concept lattices have been developed, including FCbO [13], In-Close [14,15] and AddIntent [16,17], etc. Some of those algorithms such as AddIntent, referred to as incremental construction algorithms, can handle the problem of adding a new object to a formal context. However, adding new objects is not the only case in context alteration. Removal or alteration of existing objects of a formal context can also bring changes to the corresponding concept lattice. Altering objects can be interpreted as first deleting the old objects and then adding the new objects. Therefore, both adding objects and deleting objects are crucial to processing dynamic datasets. However, the topic of deleting objects from a formal context has not been studied very thoroughly in the literature. In general, there are two different ways of dealing with the removal of objects. One of them is to compute a new lattice from scratch, and the other is to update the present lattice. Evidently, the latter should be a better option in most situations, which can be conducted by incremental algorithms such as RemoveObject [18] and DeleteObject [19].

In this paper, we introduce a new incremental algorithm for deleting objects from a concept lattice. The algorithm we propose improves two fundamental procedures shared by other incremental algorithms, i.e., identifying deleted concepts and fixing the lattice order relation. The proposed algorithm performs considerably better than other methods when applied to all kinds of datasets.

The paper is organized as follows. In Section 2, we recall some basic definitions and propositions of FCA. Section 3 gives a brief survey of incremental algorithms for removing objects. Section 4 describes our algorithm and proves the correctness of our proposed improvements. In Section 5, we discuss time complexity issues. Section 6 presents an experimental evaluation of the performance of the algorithm. Our work is concluded in Section 7.

2. Preliminaries

In this section, we introduce basic FCA notions and conventions. All definitions and propositions are assumed they can be found in [1,2] which the reader is kindly referred to for a more detailed description.

* Corresponding author.
E-mail addresses: ligeng-zou@csu.edu.cn (L. Zou), zpzhang@csu.edu.cn (Z. Zhang), jlong@csu.edu.cn (J. Long), hao@csu.edu.cn (H. Zhang).

http://dx.doi.org/10.1016/j.knosys.2015.07.022
0950-7051/

Table 1
Example of a formal context.

	a	b	c	d	e
1	×	×	×	×	×
2	×		×		×
3				×	
4		×			×
5	×	×	×	×	

Definition 1. A *formal context* is a triple of sets $K = (G, M, I)$, where $I \subseteq G \times M$ is a binary relation between G and M. The elements in G and M are called *objects* and *attributes*, respectively. $g I m$ or $(g, m) \in I$ indicates that the object g has the attribute m.

A formal context can be represented by a cross table (or matrix) where every row is an object and every column is an attribute. Crosses in the table represent the incidence relation I. An example of a formal context is illustrated in Table 1.

Definition 2. For a set $A \subseteq G$ of objects we define the set of attributes common to all objects in A as:

$$A^{\uparrow I} = \{m \in M | \forall g \in A, g I m\}.$$

Correspondingly, for a set $B \subseteq M$ of attributes we define the set of objects that have all attributes in B as:

$$B^{\downarrow I} = \{g \in G | \forall m \in B, g I m\}.$$

Definition 3. A *formal concept* of a formal context $K = (G, M, I)$ is defined as a pair (A, B) where $A \subseteq G, B \subseteq M, A^{\uparrow I} = B$ and $B^{\downarrow I} = A$. A and B are called the *extent* and the *intent* of the concept (A, B), respectively.

Definition 4. Let (X_1, Y_1) and (X_2, Y_2) be two formal concepts of a given formal context K. (X_1, Y_1) is called a *superconcept* of (X_2, Y_2) and (X_2, Y_2) is called a *subconcept* of (X_1, Y_1) if $X_2 \subseteq X_1$ (or equivalently, $Y_1 \subseteq Y_2$) which can be denoted by $(X_2, Y_2) \leqslant (X_1, Y_1)$. The set of all formal concepts of K together with the superconcept-subconcept relation makes a complete lattice that is called the *concept lattice* of the context.

Since the superconcept-subconcept relation is a natural partial order relation, we can simply adopt the definition of neighboring nodes of order theory here.

Definition 5. Let c_1 and c_2 be two concepts of a given formal context K. We say c_1 is a *lower neighbor* (or a *child*) of c_2 and c_2 is an *upper neighbor* (or a *parent*) of c_1, if $c_1 \leqslant c_2$ and there is no other concept c_3 with $c_3 \neq c_1, c_3 \neq c_2$ and $c_1 \leqslant c_3 \leqslant c_2$. This relationship (also called the *covering relation*) is denoted by $c_1 \prec c_2$.

Like any other partially ordered sets, concept lattices can be represented by line diagrams (or Hasse diagrams). In a line diagram, only neighboring nodes are connected by edges and c_2 should be above c_1 if $c_1 \prec c_2$. For instance, Fig. 1 is the Hasse diagram of the concept lattice derived from Table 1.

For a better understanding of our proposal, we give proofs of our proposed improvements in Section 4. These proofs have theoretical foundations rooted in the following two elementary propositions.

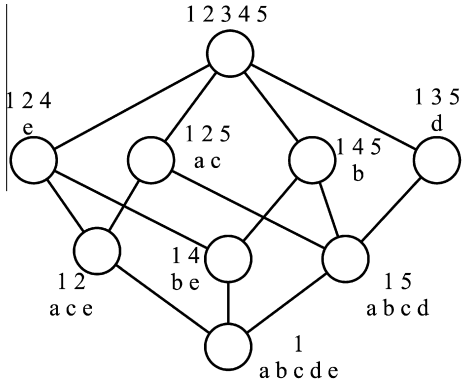

Fig. 1. Concept lattice of the formal context in Table 1.

Proposition 1. *If $K = (G, M, I)$ is a formal context, $A, A_1, A_2 \subseteq G$ are sets of objects and $B, B_1, B_2 \subseteq M$ are sets of attributes, then,*

(1) $A_1 \subseteq A_2 \Rightarrow A_2^{\uparrow I} \subseteq A_1^{\uparrow I}$,
(2) $B_1 \subseteq B_2 \Rightarrow B_2^{\downarrow I} \subseteq B_1^{\downarrow I}$,
(3) $A \subseteq A^{\uparrow I \downarrow I}$,
(4) $B \subseteq B^{\downarrow I \uparrow I}$,
(5) $A^{\uparrow I} = A^{\uparrow I \downarrow I \uparrow I}$,
(6) $B^{\downarrow I} = B^{\downarrow I \uparrow I \downarrow I}$,
(7) $A \subseteq B^{\downarrow I} \Longleftrightarrow B \subseteq A^{\uparrow I} \Longleftrightarrow A \times B \subseteq I$.

Corollary 1. *$A^{\uparrow I \downarrow I}$ is the smallest extent that includes A, and $B^{\downarrow I \uparrow I}$ is the smallest intent that includes B.*

Corollary 2. *$A \subseteq G$ is the extent of a formal concept if and only if $A = A^{\uparrow I \downarrow I}$. Similarly, $B \subseteq M$ is the intent of a formal concept if and only if $B = B^{\downarrow I \uparrow I}$.*

Proposition 2. *If T is an index set and, for every $t \in T, A_t \subseteq G$ is a set of objects, then*

$$\left(\bigcup_{t \in T} A_t\right)^{\uparrow I} = \bigcap_{t \in T} A_t^{\uparrow I}.$$

The same holds for sets of attributes too.

3. Related work

In this section, we take a look at some basic concepts of incremental algorithms for removing objects from a concept lattice. As we mentioned earlier, updating the current lattice makes more sense than constructing a new one. Incremental algorithms update the lattice by processing deleted objects one by one. A significant characteristic of those algorithms is that they do not consider any information regarding objects that have not been processed.

Removal of objects has not been sufficiently studied in the literature compared with lattice construction. In general, there are two kinds of incremental algorithms for deleting objects. One of them can be found in [18,20], and the other was first mentioned in [21] without details, then implemented in [19] with a refinement. The main idea shared by those algorithms is to modify concepts containing the deleted object and delete concepts if necessary. We can describe concepts during the update in terms of the following definition [18,19] based on results from previous researches.