A Generic Algorithm for Generating Closed Sets of a Binary Relation

Alain Gély

LIMOS, Université Blaise Pascal,
Campus des Ceźeaux, 63170 Aubière Cedex,
France
gely@isima.fr

Abstract. In this paper we propose a "divide and conquer" based generating algorithm for closed sets of a binary relation. We show that some existing algorithms are particular instances of our algorithm. This allows us to compare those algorithms and exhibit that the practical efficiency relies on the number of invalid closed sets generated. This number strongly depends on a choice function and the structure of the lattice. We exhibit a class of lattices for which no invalid closed sets are generated and thus reduce time complexity for such lattices. We made several tests which illustrate the impact of the choice function in practical efficiency.

Keywords: Generation algorithm, Closure operator, Galois or concept lattice.

1 Introduction

Closure systems or lattices are mathematical structures which are used for many applications in computer science and in particular, for discovering information in databases [1, 2].

In this paper, we give a generic divide and conquer algorithm which generates closed sets of a given binary relation. The main idea of this approach is to generate closed sets which contain an element a, then closed sets which do not contain a. This approach is then recursively applied. Moreover, we show that existing algorithms, in the spirit of generating algorithms, are particular instances of this algorithm, by only modifying the choice function. Time complexity of our algorithm is the same as that of Ganter's algorithm [5]. However, our algorithm has some extra features which can be of interest: It is simple, general, well suited for particular classes of lattices and well suited for distributed computation. It can be used as a basis for comparing different algorithms or to conceive new ones, using polynomial space.

The aim of this paper is to generate all the closed sets associated to a binary relation. This is a classical problem which could be equivalently formulated as "Generate all the maximal bicliques of a bipartite graph" or "Generate all the maximal 1-rectangles of a Boolean matrix". Number of closed sets of a binary relation may be exponential in the size of the relation.

B. Ganter and R. Godin (Eds.): ICFCA 2005, LNAI 3403, pp. 223–234, 2005.

Complexity of a generating algorithm is based on the size of the input as well as the size of the output (i.e. the number of closed sets). In this paper, when time complexity is given without mention of the output size, we talk about the time complexity *per closed set*.

We only want to generate all the closed sets, and not build the lattice structure. As we do not need to store all the closed sets, we do not allow an exponential use of memory. Thus, the algorithm presented here uses, at worst, a polynomial space in the size of the binary relation.

To describe our algorithm, we first recall some notations we will use in this paper. For definitions and proofs not given here, we refer to [7].

In this paper, we consider a binary relation $R = (J, M, I)$ where elements of J are usually called the objects, those of M the attributes of R. For $j \in J$, $m \in M$ one has jIm iff there exists an edge between j and m, i.e. if the object j possesses the attribute m. We denote usually the attributes by numbers, and the objects by small letters.

Recall Galois connection is given by the following derivation operators:

- For a set $A \subseteq J$, $A' = \{m \in M \mid jIm \text{ for all } j \in A\}$.
- For a set $B \subseteq M$, $B' = \{j \in J \mid jIm \text{ for all } m \in B\}$.

We consider the classical closure operator $'' : 2^J \to 2^J$, defined by applying twice the derivation operators. Let X be a set, then X'' is the closure of X.

2 A Generic Algorithm

The main idea of our algorithm is to generate closed sets which contain element a and closed sets which do not contain a. And then apply this principle recursively. However, input data of our algorithm is a binary relation R. Main difficulty is thus to determine two sub-relations R_1 and R_2 of R such that closed sets containing a can be generated from R_1, while closed sets which do not contain a can be generated from R_2.

Let $\mathcal{F}(R)$ be the set of all closed subsets of J with respect to the binary relation $R = (J, M, <)$. Given $a \in J$, the closed sets of R may be decomposed according to whether they contain a or not.

- $\mathcal{F}_a(R) = \{F \in \mathcal{F}(R) \mid a \in F\}$, i.e. closed sets which contain a.
- $\mathcal{F}_{\overline{a}}(R) = \{F \in \mathcal{F}(R) \mid a \notin F\}$, i.e. closed sets which do not contain a.

Clearly $\mathcal{F}_a(R)$ is a closure system on J, but the set $\mathcal{F}_{\overline{a}}(R)$ may not be a closure system since it may not contain a top element (see Figure 1).

Since $\mathcal{F}_a(R)$ is a closure system on J, we can find a sub-relation $R_1 \subseteq R$ such that $\mathcal{F}(R_1) = \mathcal{F}_a(R)$. More formally $R_1 = (J, a', <)$ since a closed set X of R that contains a must satisfy $X' \subseteq a'$. Note that, in a sub-relation, $<$ denotes the restriction of $<$ in R for present elements in the sub-relation.

On the other hand, since $\mathcal{F}_{\overline{a}}(R)$ is not always a closure system, we need to find $R_2 \subseteq R$ such that $\mathcal{F}_{\overline{a}}(R) \subseteq \mathcal{F}(R_2)$. We define R_2 by $R_2 = (J \setminus \uparrow a, M, <)$