Available online at www.sciencedirect.com

ScienceDirect

Procedia Manufacturing 22 (2018) 11–18

Procedia MANUFACTURING

www.elsevier.com/locate/procedia

11th International Conference Interdisciplinarity in Engineering, INTER-ENG 2017, 5-6 October 2017, Tirgu-Mures, Romania

Efficiency analsyis of concept lattice construction algorithms

László Kovács[a,*]

[a] University of Miskolc, Miskolc-Egyetemvaros, H-3515, Hungary

Abstract

The theory of Formal Concept Analysis provides an efficient method for conceptualization of formal contexts. The methods for conceptualization are applied mainly in knowledge engineering and data mining areas. The key element in formal conceptualization applications is the concept set construction phase. The main goal of the research work is to analyse the different incremental methods and to optimize their efficiency. The proposed method is based on the novel combination of the Ferre and InClose methods providing an efficient update operation. Based on the performed comparison tests, the proposed method outperforms the main incremental methods and for some parameter ranges, its efficiency is comparable with the efficiency of the best batch methods.

Peer-review under responsibility of the scientific committee of the 11th International Conference Interdisciplinarity in Engineering.

Keywords: formal concept analysis ; FCA; cost analysis

1. Introduction

The theory of Formal Concept Analysis (FCA) provides a tool to model and to manage the architecture of human concepts in an efficient formal way. The procedures of FCA can be used in many different application areas [12], like ontology building, machine learning methods or data mining techniques. The FCA provides methods for qualitative analysis of object-attribute relationship and to construct a lattice of the formal concepts implicated. The theory of FCA was introduced and built up in the 1980's by Rudolf Wille and Bernhard Ganter [1]. The roots of FCA originate in the theory of Galois connections and in the applied lattice and order theory developed later by Garrett Birkhoff [2].

A key aspect in practical application of FCA tools is the efficiency of concept set construction as in real life domains the context may be large resulting in several millions of generated concepts. In the literature, there are two main approaches, the construction methods can work in batch or in incremental mode. In batch mode, the full context is known already at the beginning of the set building process. This kind of methods generate the output structure for a fixed given input context [3,5]. This approach is used for static investigation when there is no change in the input context. The incremental construction method is used for problems where the initial context is extended incrementally. Incremental methods are used for applications where context is changing time by time. The modeling of cognitive

* Corresponding author. Tel.: +36-20-334-9765.
E-mail address: kovacs@iit.uni-miskolc.hu

Peer-review under responsibility of the scientific committee of the 11th International Conference Interdisciplinarity in Engineering.
10.1016/j.promfg.2018.03.003

learning processes is a good example for such dynamic problem domains.

In FCA [4], a formal context is defined as a triplet $< A, B, I >$, where A denotes a set of objects, B is a set of attributes with boolean value and I is a relation between A and B. Relation aIb is met if and only if attribute b is true for object a.

Two derivation operators are introduced as a mapping between the powersets of A and B:

$$X' = \{b|b \in B, \forall a \in X : aIb\}$$

$$Y' = \{a|a \in A, \forall b \in Y : aIb\},$$

where $X \subseteq A$, $Y \subseteq B$. The compositions of these derivations are closure operators:

$$X" = (X')'$$

$$Y" = (Y')'.$$

For a context $< A, B, I >$, a formal concept is defined as a pair (X, Y), where

$$X = Y'$$

$$Y = X'$$

is met. The components of a formal concept satisfy the $X = X", Y = Y"$ conditions, too. Component X is called the extent of the concept, while Y is the intent part.

On the set of formal concepts $C(X, Y)$, a partial ordering relation is defined in the following way:

$$(X_1, Y_1) \leq (X_2, Y_2) := X_1 \subseteq X_2.$$

The $< C(X, Y), \leq>$ structure is a lattice and it is called the concept lattice.

2. Incremental concept set building methods

The history of incremental concept set building methods can be divided into four main phases. The first phase is the pre-80s era where the investigation was focusing on mathematical aspects of the construction of Galois lattices. Among the related proposals on incremental methods, the work of Norris [7] can be emphasized having the largest impact on the next generation. The proposed algorithm generates maximal rectangles in the binary context matrix in an incremental way. The algorithm uses the canonicity test having a cost linear with the number of input objects. In this work, like in the other works of this era, no explicit cost calculation is given. The comparison tests performed in the evaluation of newer algorithms show that this method can provide a relatively good efficiency and comparable with the Ganter [1] methods running in batch mode. The Norris method is especially suitable for dense contexts. The estimated cost complexity is $O(C \cdot N^2 \cdot M)$ where C is the number of generated concepts, N is the number of objects and M is the number of attributes.

The next main phase is the era in the 90s. In this era, the terminology of FCA was introduced and the computer implementation of the algorithm became an important factor in evaluation. The first methods on lattice and set building in FCA were using batch mode, like the methods of Ganter and Bordat. The work of Godin [4] presented first in 1991 is a new kind of incremental method on FCA. The theoretical foundation of the incremental concept formation algorithms is based on the theory of Galois lattices. The algorithm uses a hash table to find the concepts already generated. The Godin algorithm should be used primarily for small and sparse contexts as it belongs to the $O(C^2)$ complexity class.