Building a pruned inheritance lattice for relational descriptions

Mélanie Courtine [1], Isabelle Bournaud [2]

[1] LIP6, Université Paris VI, 4, place Jussieu
F-75252 Paris Cedex 05, France
Melanie.Courtine@lip6.fr
[2] LRI, Bat. 490 Université Paris-Sud, Av. du Général de Gaulle,
F-91405 Orsay Cedex, France
Isabelle.Bournaud@lri.fr

Abstract. In the last ten years, several approaches for knowledge discovery in databases based upon the construction of a concept lattice have been developed. Most of them are dedicated to binary or propositional descriptions. This paper presents an approach to build a particular concept lattice, called the *Generalization Space*, for relational descriptions. It is based upon an iterative reformulation of the descriptions into a sequence of languages more and more expressive. The anytime property of the algorithm allows one to use it on large databases, and experiments show that its complexity grows linearly with the number of descriptions.

1 Introduction

Several approaches for Knowledge Discovery in Databases based on the construction of a concept lattice have been developed in the last ten years [11], [6], [19], [16]. Concept lattices -or Galois lattices [1], [21] - give a formal framework to organize concepts into a hierarchy. The main difficulty in using concept lattices lies in their construction. Guénoche analysis the four main algorithms of concept lattice construction for binary descriptions and shows that they are not suitable for large databases because of their exponential complexity [14]. More recent works show that it is easier to build a partial concept lattice (the complexity is quadratic with the number of descriptions) than the complete one [8], [13].
Our research concerns the automatic organization of *relational* descriptions, i.e. data represented using an expressive formalism such as first-order logic, description logics, conceptual graphs, …, into a particular concept lattice. To avoid the inherent problem of combinatorial explosion due to the relations, we propose to take *gradually* into account the relations. The approach presented here, called KIDS, extends the propositional approach COING [2] to a relational framework. Given a set of objects described using conceptual graphs [20] and domain knowledge represented in a generalization lattice, COING builds the propositional Generalization Space of the objects. Thanks to an iterative reformulation of the descriptions, KIDS progressively

enriches this space with more and more expressive descriptions at each step of the algorithm.

In the following section, we remind some definitions about concept lattices and define the Generalization Space. Our algorithm to construct a relational Generalization Space is described in the third part. We first briefly recall the main aspects of the propositional algorithm COING and then explain how it is extended to a relational framework in the KIDS system. In the next section, we present an application of our approach to organize a Chinese characters database. These experiments show the feasibility of the proposed approach. A comparison with related works is given in the part 5. We give in section 6 directions for futur work.

2 Concept lattice, partial concept lattice and generalization space

2.1 Concept lattice

A concept lattice (also called a Galois lattice) is a conceptual hierarchy built on a set of objects O described by a set of descriptions D [1], [21]. A node n_i of the lattice is called a "(formal) concept". It is a pair (o_i, d_i) where o_i -the *coverage* of the concept- is the subset of O covered by the node and d_i -the *description* of the concept- is a subset of D which are common to all the objects of o_i.

A partial ordering relation among the nodes, the *subsumption* relation, is defined as follows : let $n_1 = (o_1, d_1)$ and $n_2 = (o_2, d_2)$, $n_1 \leq n_2$ iif $o_1 \subseteq o_2$ and $d_2 \subseteq d_1$. In the *Hasse diagram* representing the lattice, the nodes are organized according to this relation: there is an edge between n_1 and n_2, if $n_1 \leq n_2$ and there is no other node n_3 in the lattice such that $n_1 \leq n_3 \leq n_2$. n_1 is a *parent* of n_2 and n_2 is a *child* of n_1. The *concept lattice* is the set of *all* the concepts supplied with this partial order.

A concept lattice is redundant. Given a concept $n = (o, d)$, its coverage o belongs to the coverage of each ancestor of n and its description d appears in the description of each descendant of n. Two partial concept lattices have been defined to limit redundancy:

- The *X'-inheritance concept lattice* is represented by all the pairs (o, d') where d' is the non redundant elements of d [11],
- The *X'-pruned Galois lattice* [11], also called the *Galois sub-hierarchy* [8], is generated from the X'-inheritance concept lattice by eliminating the pairs whose d' set is empty. The pruned lattice contains less nodes than the concept lattice associated, its structure is not necessarily a lattice, but it allows one to reconstruct the concept lattice without loss of information.

2.2 Generalization Space

Given a set of object descriptions and a generalization language, the Generalization Space is a *pruned inheritance concept lattice* where each node description is the *most specific generalization* of the descriptions of the objects covered by the node. In the case of a propositional generalization language, the most specific generalization of a set of descriptions is *unique*. In the other case, a node description contains all the most specific generalizations –w.r.t. the considered generalization language– of the set of descriptions.
Deriving a pruned inheritance concept lattice from a concept lattice is easy. However, existing methods to build concept lattices are not suitable for large databases because of their exponential complexity with the number of objects [14]. In the following part, we present our ascending approach to build Generalization Spaces: COING builds a propositional Generalization Space while KIDS enriches that Generalization Space with relational descriptions.

3 Building a Generalization Space

3.1 A Generalization Space for propositional data

Given a set of objects described using conceptual graphs [20] and domain knowledge represented in a generalization lattice [11], COING builds the propositional Generalization Space of the descriptions [2]. In order to deal with the problem of generalizing relational descriptions [15], COING reformulates each conceptual graph describing an object into a set of *independant* arcs. The main advantage of this reformulation is to limit the complexity of the GS construction (in the worst case quadratic with the number of objects, and linear in pratice [2]). This reformulation has been initially proposed in [12].
COING is an ascending method: it starts from the descriptions as sets of arcs to build the nodes. Each arc of the descriptions is generalized w.r.t. the generalization lattice. The generalized arcs covering the same set of objects are clustered into the same node, and then filtered in order to keep only the most specific ones[1]. The following figure 1 presents the propositional Generalization Space built by COING for the three houses h1, h2 and h3.

[1] The reader interested in a more precise presentation of COING should refer to [2], [3].

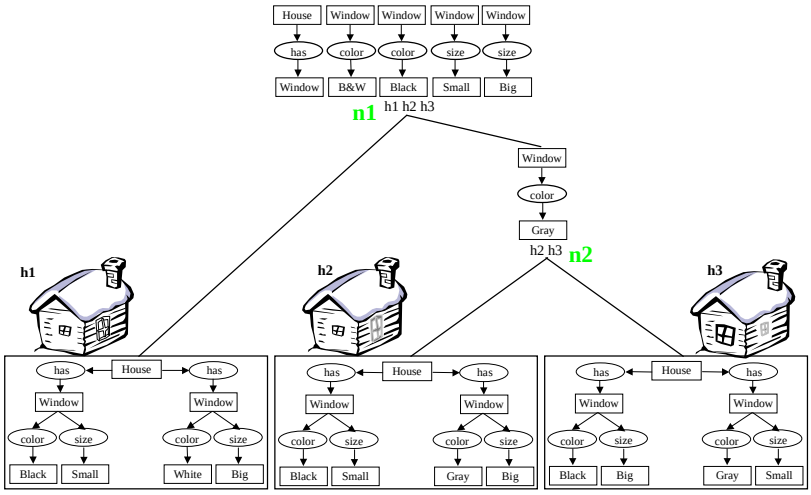

Fig. 1. A Generalization Space for propositional data.

This Generalization Space contains two class nodes (n1 and n2) and three object nodes corresponding to the houses (box nodes). The node n2, for example, clusters the houses h2 and h3. Its coverage is {h2, h3} and its description is the arc `[Window]->(color)->[Gray].` This node indicates that h2 and h3 have at least a `gray` window in common in their descriptions and that this property is not shared by any other object considered. Thanks to the inheritance structure of the GS, we may add the description of the root node (n1) to this description. More precisely, we add the arcs from n1 which are not generalizations of arcs from n2, for example the arc `[Window]->(Size)->[Big].` Thus, the node n2 indicates that the two houses h2 and h3 have `window(s)`, which have a size (`Small,Big`) and a color (`Gray, Black).`

If COING has a low complexity, it does not take into account the relational aspect of the descriptions: the graphs describing the objects are decomposed into a set of independent arcs and relations among arcs are lost. In the following section, we give the principle of our approach to extend COING to a relational framework.

3.2 A Generalization Space for relational data

KIDS gradually enriches the propositional Generalization Space built by COING while using a sequence of language which is made more and more expressive at each iteration. The property of the Generalization Space used in KIDS is the following :

> *If there exists a common sub-graph SG among the descriptions of a given set of objects o, then there is a node in the GS built by COING whose coverage contains o and whose complete description -completed with the inherited arcs- contains the arcs of SG.*

This property allows us to use the propositional GS in order to find sub-graphs generalizing a set of object descriptions. The idea is to search for more and more complex sub-graphs. The heuristic used by KIDS to enrich the propositional Gener-

alization Space is based on the fact that a sub-graph of i arcs is a sub-graph of (i-1) arcs + 1 arc. We defined the notion of *candidate* node : a node of the GS is a candidate node for KIDS at level i if it has been modified at level i - 1.

In practice, KIDS starts with the propositional GS built by COING using a language of arcs (COING is the 1st level of KIDS). At its second level, KIDS reformulates object descriptions based upon a language of two connected arcs. At its third level, KIDS reformulates object descriptions based upon a language of three connected arcs, etc. . Let us notice that at a given level, KIDS does not reformulate the descriptions of all the objects, but only the descriptions of objects appearing in the candidate nodes.

At each level, KIDS may refine the description of existing nodes (it consists in linking an arc to an existing sub-graph) or add new nodes to the Generalization Space found at the previous level. Thus, the GS is not completely reconstructed at each iteration of KIDS and the KIDS algorithm is "anytime". The node descriptions at a given iteration (level) are maximally specific w.r.t. the language corresponding to this level. If a node description generalized an object description then this object is necessarily in the coverage of that node.

Another main aspect of KIDS, is that it uses the propositional learner COING to perform the reformulated descriptions. In order to allow COING to do this, we have defined the notion of "abstract arc". The reader should refer to [4] for a more precise presentation of the KIDS algorithm. Complexity results of KIDS are given in section 4.2.

Figure 2 above presents a relational Generalization Space found by KIDS for the three houses h1, h2 and h3.At the 2nd level, KIDS found common sub-structures which were not find by COING. For example, the node n1 show the fact that the three houses have (at least) two windows and that each of them have a color (`W&B` or `Black`) and a size (`unknown, Small` or `Big`). Furthermore, KIDS clusters h1 and h2 into a node, and only these two houses, since they have a small black window in common and this window does not appear in the description of h3 (even if h3 has a small window and a black window but it is not the same window). This similarity was not found by COING and required to use KIDS (at the second level, first iteration) since it is a particular composition of *two* arcs. It is useless to perform KIDS at the next level since the use of structures of three connected arcs allows ones to reformulate the descriptions without loosing information [4].

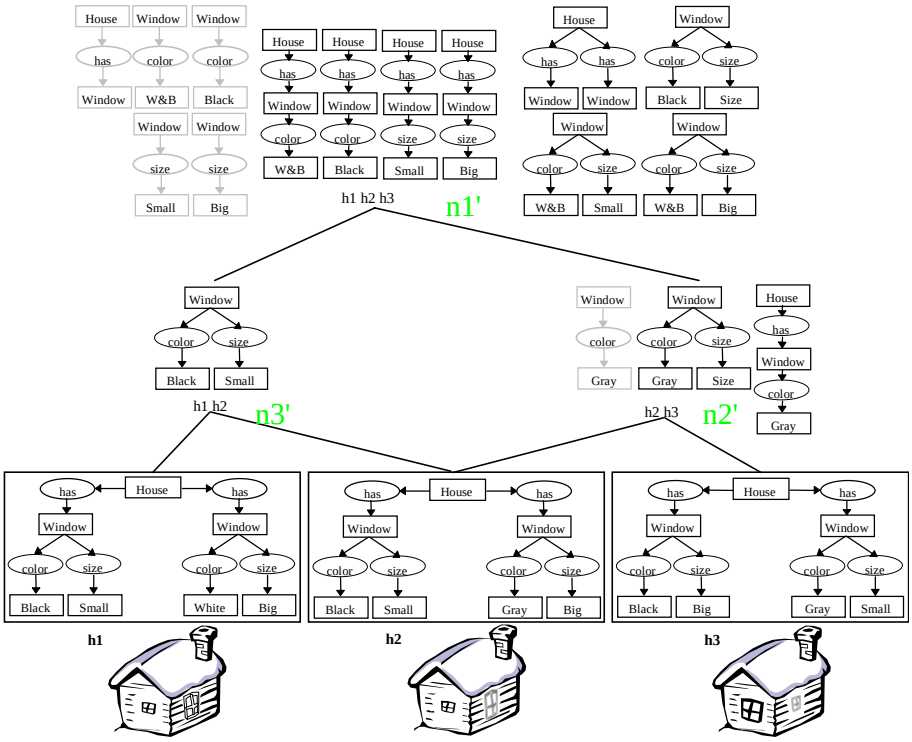

Fig. 2. A relational Generalization Space for three houses

4 Experimentations on a Chinese characters database

This section presents an application of the above method in the framework of the construction of classifications of Chinese characters. We briefly remind the context of this work[2]. These experiments aim to show the feasibility of KIDS in terms of complexity and to illustrate its interest for relational data organization.

4.1 Description of the relational data

The database considered is a collection of 6780 Chinese characters. Each character is represented by a conceptual graph. Characters are described by : their initial and final pronunciation, the ton of this pronunciation, the components (between 1 and 5) and their relative positions and the key component. For example, the conceptual graph of

[2] For more information about this application, the reader should refer to [2].

figure 3 represents the character 情, which is composed of the radicals C5381 and C2843, which is pronounced " qing ", which is in ton 2 and means "feeling".

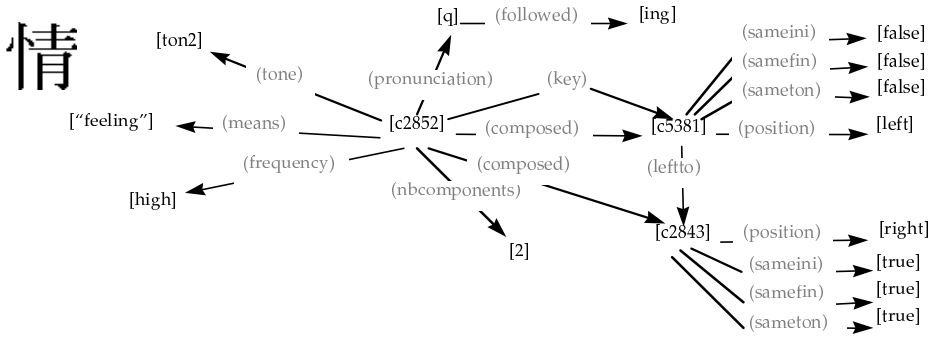

Fig. 3. Conceptual graph describing the character 情.

Part of the generalization lattices used for Chinese characters is presented on the following figure 4:

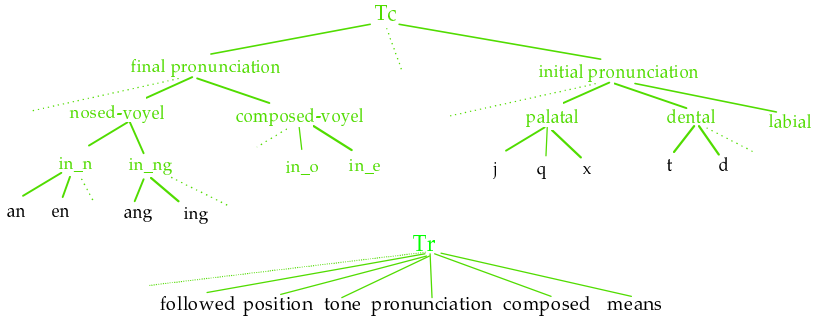

Fig. 4. Part of the generalization lattices for Chinese characters.

4.2 Complexity results

We evaluated KIDS on several databases of characters composed of 10 to 160 or 416 characters. Figure 5 shows the total time required for generating the GS for these databases using the COING (KIDS 1st level) and the KIDS algorithms.

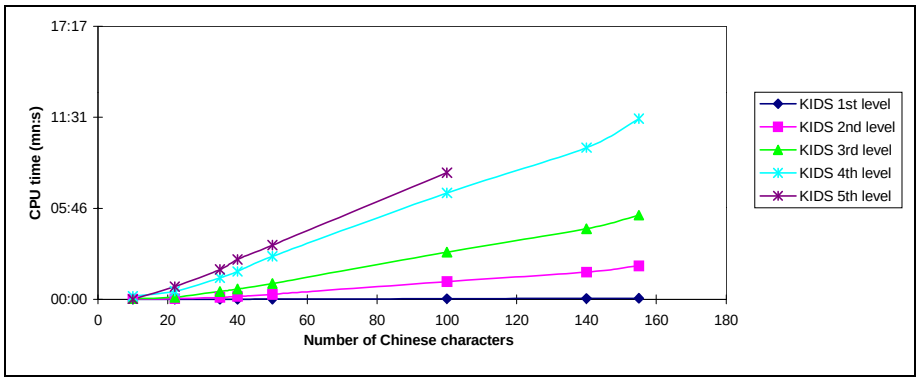

Fig. 5. Average execution time of KIDS on Chinese characters databases.

As shown on figure 5, in practice, the CPU time of the proposed algorithms is linear (it is quadratic in the worst case for COING [2]) with the number of objects. This results may be surprising because, as it manipulates sub-graphs, KIDS introduces a complexity factor. In effect, the theoretical complexity of KIDS in the worst case is exponential. However, the combinatorial explosion due to the generalization of sub-graphs is limited since the bigger the level of KIDS is (i.e. the more complex are the graphs to generalize) the less the number of sub-graphs to perform is.

The level introduces a multiplicative factor; the time necessary to move to the next level is very close to be constant (cf. figure 5).

During these experiments, we also evaluated the evolution of the number of nodes of the GS as a function of the algorithm used. For COING, this number is in the worst case in O(N) [2]. Figure 15 summarizes these results.

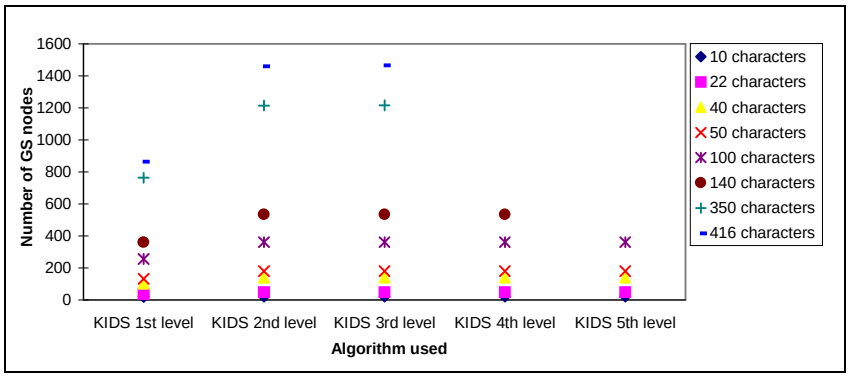

Fig. 6. Evolution of the number of nodes of the GS.

This graph shows that the number of nodes of the GS grows until a specific level – 2nd level for the small bases and 3rd level for the largest – then it becomes constant.

This may be explained by the fact that from a specific level, KIDS does not allow to create new nodes, but only to enrich the description of existing ones.

5 Related Works

A Generalization Space is a pruned inheritance concept lattice since nodes whose description is empty do not appear in it. If this may be considered as a drawback for some applications, Dicky shows that this structure contains the same information that the concept lattice but requires less memory [8]. Furthermore, the pruned inheritance concept lattice is a useful structure to discover a set of association rules as it allows one to directly extract *valid* and *informative* rules [19], [5].

Recent works [8], [13] show that it is easier to build a partial lattice (quadratic complexity with the number of objects) than the complete one (exponential complexity). Experiments illustrate that the complexity of the Generalization Space construction (for COING and KIDS) is, in practice, linear with the number of objects [4].

An important limitation of most existing methods to build concept lattices is that they are dedicated to binary or propositional descriptions [21], [6], [11]. The KIDS approach considers descriptions represented using a more expressive language -the conceptual graph formalism. Other works deal with the construction of concept lattices for conceptual graphs [17], [12].

The complexity of GRAAL is depending on the complexity of the generalization relation defined on the considered sub-graphs [17]. In practice, Liquière limits the graphs to locally injective ones since the complexity of the generalization relation is polynomial for such graphs. The main difference between KIDS and GRAAL is that in KIDS one does not have to limit a priori the structure of the graphs describing the objects to be able to perform them with a reasonable complexity. Another advantage of KIDS lies in its anytime property which allows one to stop the process at anytime and to have a result.

The approach proposed by Godin [12] and the one developed in COING are quiet similar. They are both based on a graph reformulation into a set of independent arcs. In order to "reconstruct" sub-graph from the set of independent arcs describing a node of the lattice, Godin uses the fact that the decomposition of a sub-graph as a set of independent arcs may be done without loosing information if the considered sub-graph has some properties (a same concept type appears only once in the sub-graph).

Incremental approaches to build a concept lattice [6], [11], or a partial one [8], [13] update the lattice whenever new objects or new features are added in O or in D. Our approach is not incremental: the addition of a new object requires a complete reconstruction of the GS. This is a consequence of using a generalization lattice over the types describing the objects and searching for maximally specific generalizations.

6 Conclusion and futur works

We presented an approach to build a relational Generalization Space. This approach is based upon an iterative reformulation of the descriptions into a sequence of languages more and more expressive. The complexity of this anytime algorithm is, in practice,

linear with the number of objects. The databases used to evaluate this work concern different domains : Chinese characters, car collision reports, paleontological data and the DNA sequence.

The first perspective of this work is to provide KIDS with a more efficient processing of numerical data. Currently, the numerical information contained in the descriptions is processed as symbols; the implicit order existing among numbers is not taken into account. A preprocessing on descriptions would make it possible to determine a hierarchy of generalization on numerical values.

Another possible improvement of the algorithm is to define methods to evaluate the interest of KIDS for a given database. Indeed, when the concepts in the objects of a conceptual graphs database appear only once, it is not necessary to apply KIDS to this database, because the decomposition does not cause any loss of information. On the contrary, if a concept appears several times in the objects descriptions (like in the houses), it is not possible to differentiate them. So, we can consider a pre-processing on the data to evaluate the maximal level to which KIDS needs to be applied.

References

1. Barbut, M., & Montjardet, B. (1970). Ordre et classification. Hachette.
2. Bournaud, I. (1996). Regroupement conceptuel pour l'organisation de connaissances. Thèse de Doctorat, Université de Paris VI, France.
3. Bournaud, I. & Ganascia, J.G. (1997). Accounting for Domain Knowledge in the Construction of a Generalization Space. ICCS, Lectures Notes in AI n°1257, Springer-Verlag, pp. 446-459.
4. Bournaud, I., Courtine, M. & Zucker, J.D. (2000). Abstractions for Knowledge Organization of Relational Descriptions. Symposium on Abstraction, Reformulation and Approximation, SARA'2000, Lectures Notes in AI n°1864, Springer-Verlag, pp. 87-106.
5. Bournaud, I. & Courtine, M. (2001). Un Espace de Généralisations pour l'Extraction de Règles d'Association. Journées Francophones d'Extraction et de Gestion des Connaissances, EGC 2001, H.Briand et F.Guillet (Eds), Editions Hermès, pp. 129-135.
6. Carpineto, C. & Romano, G. (1993). GALOIS : An order-theoretic approach to conceptual clustering. Tenth International Conference on Machine Learning, pp33-40.
7. Chein, M. & Mugnier, M.L. (1992). Conceptual Graphs : Fundamental Notions. Revue d'Intelligence Artificielle, Volume 6, Numéro 4 , pp.365-406.
8. Dicky, H., Dony, C., Huchard, M. & Libourel, T. (1994). Un algorithme d'insertion avec restructuration dans les hiérarchies de classes. Actes de Langages et Modèles à Objets, Grenoble.
9. Fisher D. (1987). Knowledge Acquisition Via Incremental Conceptual Clustering. In: Michalski, R.S., Carbonell, J., Mitchell, T.(eds.): Machine Learning: An Artificial Intelligence Approach. San Mateo, CA, Morgan Kaufmann. II, pp. 139-172.
10. Gennari, J. H., Langley, P., Fisher, D. (1989). Models of incremental concept formation. Artificial Intelligence 40-1(3), pp.11-61.
11. Godin, R., Mineau, G., Missaoui, R. & Mili, H. (1995a). Méthodes de classification conceptuelle basées sur les treillis de Galois et applications. Revue d'Intelligence Artificielle, Volume 9, Numéro 2, pp.105-137.
12. Godin, R., Mineau, G., & Missaoui, R. (1995b). Incremental structuring of knowledge bases. International Knowledge Retrieval, Use and Storage for Efficiency, Santa Cruz, pp. 179-198.
13. Godin, R., & Chau, T.T. (2000). Comparaison d'algortihmes de construction de hiérarchies de classes. L'Objet, Volume 5, Number 2, pp.321-338.

14.Guénoche, A. (1990). Construction du treillis de Galois d'une relation binaire. Mathématiques Informatique et Sciences Humaines, Volume 109, pp.41-53.
15.Haussler, D. (1989). Learning conjunctive concepts in Structural Domains. Machine Learning, Volume 4, pp.7-40.
16.Hereth, J., Stumme G., Wille, R., & Wille, U. (2000). Conceptual Knowledge Discovery and Data Analysis, Technical Report n°2092, Technische Universitat Darmsadt.
17.Liquière, M. & Sallantin, J. (1998). Structural machine learning with Galois lattice and Graphs. Fifteenth International Conference on Machine Learning.
18.Michalski, R.S., & Stepp, R. (1982). Learning from observations: conceptual clustering. In Machine Learning: An Artificial Approach, Volume 1, Tioga Publishing.
19.Simon, A. (2000). Outils de classificatoires par objets pour l'extraction de connaissances dans des bases de données. Thèse de Doctorat, Université de Henri Poincaré Nancy 1, France.
20.Sowa, J.F. (1984). Conceptual Structures : Information Processing in Mind and Machine, Readings, Massachusetts, Addison-Wesley.
21.Wille, R. (1982). Restructuring Lattice Theory. Symposium of Ordered Sets, I.Rival (Ed), pp.445-470.