From: AAAI Technical Report WS-99-14. Compilation copyright © 1999, AAAI (www.aaai.org). All rights reserved.

First-Order Context and Formal Concept Analysis

Laurent Chaudron and Nicolas Maille
ONERA CERT 2 avenue Edouard Belin
BP 4025 - 31055 Toulouse Cedex 04 - FRANCE
Phone: +33 5 62 25 26 55 Fax: +33 5 62 25 25 64
{chaudron, maille}@cert.fr

Key words: Formal Concept Analysis, Context Based Knowledge Exploration, First-Order Lattices, Concepts Discovery, Context Based Rules, Logic Programming

Abstract

Formal Concept Analysis -also called "Galois Lattices"- is an algebraic model based on propositional calculus that is used for symbolic knowledge exploration from a formal context. The aim of this paper is to design the theoretical models required for the extension of Formal Concept Analysis to first-order logic so as to improve both the expression power as a knowledge mining tool upon first order contexts, and the relevance of its results.
Our contribution consists in: i) a synthesis of the basic notions of FCA, ii) the design of the Cube Model dedicated to the conjunctions of literals, iii) the design of a complete first-order logic formal concept analysis of first-order contexts. The approach is described from the theoretical point of view, implementations in logic programming and applications are also briefly presented.

Introduction

As far as knowledge information is concerned, the induction of concepts from a context described by data and information is a pivotal topic; generally, if numerical valuations (belief measures, preferences...) can be defined on the considered data, numerical or mixed methods can be directly used: rough sets (Pawlak 1991; Skowron & Polkowski 1998), Cartesian space model (Ichino & Yaguchi 1998; Ichino & Ono 1998)... In some cases, requirements or accessibility constraints imply to rest only on symbolic attributes. Thus, more fundamental models and techniques are required; Formal Concept Analysis (Ganter & Wille 1996) is a suitable candidate for such a purpose. Unfortunately, FCA theory relies only on propositional calculus and an extension is required so as to characterize the context with more expressive attributes. Indeed, our applications (cooperative system design, activity modeling, symbolic fusion...) require means to represent both symbolic information as predicates and numerical parameters. But in our approach, we do not use numerical features to capture symbolic notions in order to keep the semantic on each piece of the model. This methodological constraint leads to cautiously define the elements of the model dedicated to the representation of the basic elements of the context and their exploitations. This is the purpose of our work.
The relations between FCA and first-order logic have been studied, especially in (Zickwolf 1991); our work is more focused on the formal prerequisites and the programming conditions of the definition of a consistent model of first-order Logic FCA (say: **1LFCA**). This leads to search for first-order corresponding operators to the fundamental ones in FCA: set union, set intersection, and the Galois connections (one must notice that nothing else than these three operators is needed to generate the FCA theory). This is the purpose of the Cube lattice model thanks to which relevant definitions of 1LFCA can be formulated and implemented.
As it has been the case for FCA, the study of 1LFCA has created some results which may be of a theoretical interest for themselves, even if one may also expect 1LFCA to offer more powerful exploration tools and in particular more suitable "numerical⟶symbolic" translations for the contexts of the considered applications. Thus, the motivations of this work are both pragmatic and theoretical.

Foreword: In the next section, the foundations of classical FCA are recalled in a revised new form which is adapted to the extension to first-order logic. In the sequel, a classical term is underlined, while a term defined by the authors is quoted within a box. In the context of the paper the proofs are omitted (they can be found in (Chaudron & Maille 1999)). Propositions, lemmas and theorems are labeled within the same sequence.

Formal Concept Analysis

The basic notions of FCA

Formal Concept Analysis (say: FCA) is a set-theoretical model for concepts that reflects the philosophical understanding of a concept as a context-based unit of thought consisting in two parts: the

extend, which contains all the entities (the objects, the examples...) belonging to the concept, and the *intend*, which is the collection of all the attributes (the characteristics, the properties...) shared by the entities (Arnaud & Nicole 1662). For example, the pair: *({B747, A3XX, DC9} , {wings, engines})* may naturally induce a simple and common concept...
Based upon Galois connections, FCA was first described in (Barbut & Monjardet 1970) and in the 80's R. Wille designed a dedicated theory and program at the University of Darmstadt. An introduction can be found in (Davey & Priestley 1990) and FCA theory and applications are now described in the reference book (Ganter & Wille 1996). FCA is frequently used as a preprocessing tool for classification (Carpineto & Romano 1996), but in our approach, we stay closer to the original purpose of FCA. In FCA, the basic notion that models the knowledge about a specific domain is the formal *context* –described as a binary relation between two finite sets– from which concepts and conceptual double hierarchies can be formally derived so as to form the mathematical structure of a lattice[1] with respect to a subconcept-superconcept relation. FCA is used for self-emergent classification of objects, detection of hidden implications between objects, construction of concept sequences, object recognition, aggregation of data and information, knowledge representation and analysis.

Definitions and properties of FCA

Definition 1 A formal *context* C is defined as a triple (O, P, ζ) where O (objects or entities) and P (properties or attributes) are finite sets and ζ is a mapping from O onto $\mathcal{P}(P)$. For o in O, $\zeta(o)$ indicates the subset of attributes possessed by o (ζ is frequently synthetised in a table).

Example 1: Let $C_1 = (O_1, P_1)$, $O_1 = Obj_{i \in [1,3]}$, $P_1 = Pro_{j \in [1,5]}$,
$\zeta(Obj1)$={*Pro1,Pro2,Pro5*}
$\zeta(Obj2)$={*Pro2,Pro3,Pro4,Pro5*}
$\zeta(Obj3)$={*Pro1,Pro3,Pro5*}
In the sequel we assume that (O, P, ζ) denotes the working context.

Definition 2 The *dual* operators (denoted as $'$) between O and P are defined as follows: $(\forall A \subset O)(\forall B \subset P)$,
$A' =_{def} \bigcap_{o \in A} \zeta(o)$,
$B' =_{def} \{o \in O | B \subset \zeta(o)\}$.

[1] given two internal operators $\sqcap$ (infimum) and $\sqcup$ (supremum) on a set E, $(E, \sqcap, \sqcup)$ is a lattice, iff$_{def}$: $\sqcap$ and $\sqcup$ are idempotent, associative, commutative and they verify the absorption law $x \sqcap (x \sqcup y) = x$ and $x \sqcup (x \sqcap y) = x$. A lattice is always an ordered set: the relation $\leq$ defined on E as: $(x \leq y) \leftrightarrow_{def} (x \sqcap y) = x$ is an order relation for which $\sqcap$ and $\sqcup$ represent the greatest lower bound and the least upper bound (Birkhoff 1940).

A' is the set of attributes common to all the objects in A, and B' is the set of the objects possessing their attributes in B.

Example 2: in C_1, if $A = \{Obj1\}$, Then $A' = \{Pro1, Pro2, Pro5\}$, and $(A'' = A)$. If $B = \{Pro1, Pro2\}$, then $B' = \{Obj1\}$, and $(B'' = A')$. One can also notice the case of the empty set: as a subset of O: $\emptyset' = P$, and conversely, as a subset of P: $\emptyset' = O$.

Remarks:
- Depending on the properties of sets O and P, the dual operators are called *polarities* in (Birkhoff 1940), whereas when defined by their basic properties (see Proposition 1) they are traditionally known as *Galois connections*.
- Notation $'$ is the same for subsets of O and P as both play a symmetric role in the theory. This symmetry disappears when P deals with predicates instead of propositions.
- Usually the context is suppose to verify $O \cap P = \emptyset$ but all cases can be considered.
- The attributes are atomic positive formulas. In FCA the negation operator is not considered explicitly.
- In this section, the definitions, the propositions and the proofs are based on the usual set operations. More precisely the definitions and arguments rely only on the set of properties induced by the fact that $(\mathcal{P}(O), \subset, \cup, \cap)$ and $(\mathcal{P}(P), \subset, \cup, \cap)$ are lattices. This allow the extension of FCA to first-order logic to be "smoothly" achieved.

Proposition 1 the dual operators verify the following properties: for all A_i subset of O (the symmetric properties stand for any subset of P):
$(i) A_1 \subset A_2 \Rightarrow A_2' \subset A_1'$,
$(ii) A \subset A''$,
$(iii) A' = A'''$,
$(iv) (A_1 \cap A_2)' \supset A_1' \cup A_2'$,
$(v) (A_1 \cup A_2)' = A_1' \cap A_2'$.

Definition 3 Given $A \subset O$ and $B \subset P$, the pair (A, B) is a *concept* iff$_{def}$ $A' = B$ and $B' = A$.
A is the *extend* of the concept and B is the *intend*.
The set of all concepts defined on the context (O, P, ζ) is denoted as $\underline{L}$.

Example 3: in C_1, ({Obj2,Obj1},{Pro5,Pro2}) and ({Obj3,Obj1},{Pro5,Pro1}) are concepts.

Two questions arise: what is the structure of L and how to determine it? This is the role of the next theoretical results.

Definitions 4 The *supremum* $\vee$ and *infimum* $\wedge$ operators are defined on L as follows: for all concepts (A_1, B_1) and (A_2, B_2) in L:
$(A_1, B_1) \vee (A_2, B_2) =_{def} ((A_1 \cup A_2)'', B_1 \cap B_2)$
$(A_1, B_1) \wedge (A_2, B_2) =_{def} (A_1 \cap A_2, (B_1 \cup B_2)'')$

Theorem 2 $(L, \ll, \vee, \wedge)$ is a complete lattice.

The formal definitions match with the intuitive notion of concepts as a pair of a collection of examples and their characteristics: the more objects there are the less characteristics they share. The relation between comparable concepts works like communicating vessels:

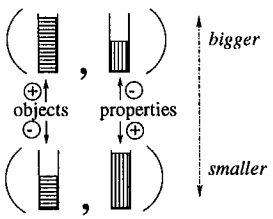

Figure 1: The order relation between concepts

Example 4: the concept lattice L_1 generated by context C_1 is constituted of 8 concepts:

8: ({},{Pro1,Pro2,Pro5,Pro3,Pro4})
7: ({Obj3},{Pro1,Pro3,Pro5}))
6: ({Obj2},{Pro2,Pro3,Pro4,Pro5})
5: ({Obj1},{Pro1,Pro2,Pro5})
4: ({Obj3,Obj2},{Pro5,Pro3})
3: ({Obj2,Obj1},{Pro5,Pro2})
2: ({Obj3,Obj1},{Pro5,Pro1})
1: ({Obj1,Obj2,Obj3},{Pro5})

L_1 can be represented by its Hasse-diagram (figure 2). On

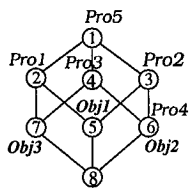

Figure 2: The Concept Lattice L_1

such a diagram, an element labeled by an object Obj_i represents the concept with the smallest set containing Obj_i, and an element labeled by a property Pro_j represents the concept with the smallest set containing it. Hence, a given concept ⓝ inherits all the properties which are linked above it in the diagram and ⓝ is constituted of all the objects which are linked below it.

Application: System Analysis

This real application is related to the design of distributed information systems. A first analysis was made with classical FCA in (Chaudron & Barès 1997); in the present section, we recall the description of the problem and we analyze the extensions of the context to first order logic. This example may appear very simple at a first glance: a few systems characterized by a few features. But it appears clearly that a hand-made analysis would be quite impossible to be made. FCA has helped for such a process (the specific details of the application are not detailled here).

Given a set of information systems characterized by functionalities (e.g. medical database access, languages translations, transmission capability), they are supposed to operate together in order to achieve peace maintenance or rescue missions. The problem is how to design the subgroups of systems, sharing same capabilities so as to better manage the whole set of systems? FCA appears to be a appropriate tool for such a system analysis.

The following context C_2 (with fictitious identifiers) describes the features of five real categories of systems: O_2 = *{VEH, TRANS, XFLR6, SYSF99, SYS33}*. VEH is an on-board information system of a vehicle, TRANS is a transmission system, XFLR6 is a mobile surgery cell, SYS33 is a local information system and SYSF99 is a global rescue management system. The context C_2 is defined as follows[2]:

	VEH	TRA	XFL	SY9	SY3
omouv(meca)	×			×	
opo(ue)		×			×
opo(odb)		×		×	×
omouv(ue)		×			×
cr(tir)		×			×
dem(tir)		×	×		×
opo(gu)			×	×	×
opo(SYS33)				×	×
opo(rens)				×	×
opo(odb)		×		×	×
cr(sit)			×	×	×
cr(sit2)		×			×
prev(tirnucami)					×

The properties of the systems define a set of 12 literals. As they contain no variable they can be considered as propositional calculus literals. For instance: *omouv(meca)* (resp: *omouv(ue)* means that the system can manage (access, store, transmit) instructions related to the moving of specified vehicle (resp: a small group of vehicles, such as a surgery cell); an attribute of the kind *opo(x)*, with $x \in \{ue, gu, odb, rens\}$ denotes the capability of managing the structural organization of the different teams; *cr(sit)* means that the system can access to the state of the current situation: staff, wounded people, material... The Concept Lattice L_2 derived from C_2 is constituted of 11 concepts defining interoperable groups (figure 3) in which knowledge explorations are processed thanks to programming capabilities.

Implementation and Exploitation

A Prolog program was designed so as to implement the Galois connections and the function which associates to each context C its Concept Lattice L: $C \longrightarrow L$. The determination algorithm relies on Theorem 3: the concept lattice L is determined by the finite sequence T_n of sets of concepts:

[2] C_2 summarizes a four-pages specification file.

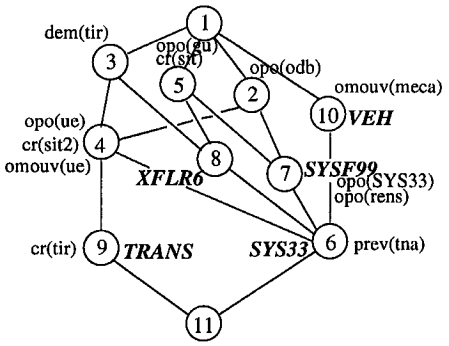

Figure 3: The System Analysis Concept Lattice L_2

$T_0 = \{(\emptyset, P)\}$, $T_1 = \{(o'', o')|\ o \in O\}$
$T_{n+1} = \{c_i \vee c_j|\ c_{i,j} \in T_n\}$
$T_{n_0} = \{(O, \emptyset)\}$, $L = \bigcup_{n \in [0,n_0]} T_n$

The context and the concept lattice can be considered as a global Prolog knowledge base $C \cup L$ on which knowledge exploration experiments are performed[3].

The knowledge base $C \cup L$ is used so as to look for contextual dependencies between either objects or attributes. The induction of the context-based rules is given by the generic frame of (Guigues & Duquenne 1986) widely developed in *ConImp* by Burmeister (Burmeister 1987). The techniques are based on the fundamental Lemma:

Lemma 3 $(\forall A \in P), \models (A \rightarrow (A'' - A))$.

Based on this Lemma, a context rule generator was implemented so as to compute the set of rules deducible form the context. The logical links between the features are captured by first order rules which are translated into production rules (these algorithms are not detailed here). The core of these symbolic inductions from the context is based on the Galois connections which allow various knowledge exploration to be processed, such as: "from the present context, what are the properties that are necessarily deduced when systems have to operate by sharing capabilities on rescue operations planning ($opo(x)$) and medical staff movements ($omouv(y)$)?". The solution is given by the Galois closure of $\{opo(x), omouv(y)\}$ in C_2 (fortunately, the answer is in a deductive form) the following interdependency is proved:
$\{opo(odb), omouv(ue)\} \rightarrow \{dem(tir), cr(sit2), opo(ue)\}$.
The improvement of this FCA based knowledge mining tool is currently under study.

[3] it must be noticed that the computing time is not a critical problem in our projects as the situations we are analyzing are characterized by a high complexity and a very low evolution time. The lattice must be completely developed and is completely developed.

In this example, it is easy to notice that, from the propositional calculus point of view *opo(odb)* and *opo(ue)* are different literals whereas it appears that they should share a property like *opo(x)* (with x variable) if the context could be considered within a first-order frame. This is the purpose of the next section.

The cube model

The cube model is dedicated to formalize conjunctions of properties, in this study, it will allow first order contexts to be defined.
Let us denote $\mathcal{C}_0$ the set of all the finite subsets of positive literal of the propositional calculus. For each FCA context (O, P, ζ), we have $\zeta(O) \subset \mathcal{C}_0$ and the logical properties deduced by FCA relies implicitly on the fact that $(\mathcal{C}_0, \subset, \cap, \cup)$ is a lattice in which the partial order $\subset$ is consistent with the logical implication $\leftarrow$. The aim of the cube model is to build, in a first-order logic frame, the same kind of algebraic structure. The cube model is based on a classical first order language (*Const, Var, Funct, Pred*) whose set of terms *Term* is the functional closure of *Const* $\cup$ *Var* by *Funct*. Such an approach has been used for knowledge representation in intelligence systems and cooperative systems (Chaudron *et al.* 1997). The elementary properties are represented by literals and the elements of their power set $\boxed{\mathcal{C}}$ are called $\boxed{\text{logical cubes}}$[4], they are interpreted as the conjunction of the literals. Cubes play a dual role besides the classical clauses, and by default their variables are existentially quantified.

In expert or deductive systems, knowledge squares with general rules that are captured by clauses: $c = \{trans(x), \neg resc(x)\}$ represents the information "rescue systems have transmission capabilities"; the associated logical formula is $\forall x\ (trans(x) \vee \neg resc(x))$. When we plan to describe as a context the state of an observed situation, cubes are more adequate and $c = \{trans(x), \neg resc(x)\}$ means: "there is an object with transmission capabilities that is not a rescue system". The logical interpretation is $\exists x\ (trans(x) \wedge \neg resc(x))$.
As far as knowledge representation of a context is concerned, we would like the order relation induced on $\mathcal{C}$ to capture the intuitive notion of "information enrichment". But such an "enrichment" can be obtained via different means: quantity of information, precision of terms, logical dependency. Example: $\{cr(sit), opo(gu)\}$ is more informative than $\{opo(gu)\}$ for the number of literals is higher; and $\{cr(sit)\}$ is more informative than $\{cr(x)\}$ (with x variable) for a sake of precision. Unfortunately, the combination of both intuitive criteria may lead to a contradiction: $\{cr(x), opo(gu)\}$ cannot be consistently compared to $\{cr(sit)\}$.

[4] "cube" is the name that was used for the first time by A. Thayse (Thayse & col. 1989). The denotator "product" was previously used –but not defined– in 1975 by Vere (Vere 1975).

The previous cases highlight the need for sound definitions to the intuitive concepts of union and intersection of two finite information sets in accordance to the following requirements: the infimum has to capture the common features (while giving more information than the empty set frequently generated by the unification rule); the supremum has to cope with the contradictory criteria: quantity/precision of the information (while giving a more synthetic result than the set union). Such purposes are achieved thanks to an algebraic approach.

Definition 5 $(\forall c_i \in \mathcal{C})$ $\boxed{c_1 \text{ subsumes } c_2}$, and we write $\boxed{c_1 \leq_c c_2}$, when $c_1\sigma \subseteq c_2$ for some substitution σ.
c_1 is said to be $\boxed{\text{equivalent}}$ to c_2, $\boxed{c_1 \equiv_c c_2}$ when $c_1 \leq_c c_2$ and $c_2 \leq_c c_1$.

As $\leq_c$ is a preorder, $\equiv_c$ is an equivalence relation.

Remark: symbol $\subseteq$ is equivalent to $\subset$ and represents the set inclusion. $\subsetneq$ means "included but not equal" and must not be confused with $\not\subseteq$ (not included).

Definition 6 A cube c is $\boxed{\text{reducible}}$ if there exists a substitution θ such that $c\theta \subsetneq c$.

Proposition 4 An irreducible reduction of a cube c always exists and is unique up to variable renaming. It is denoted as $\boxed{reduc(c)}$ and the set of reduced cubes is denoted as $\boxed{\mathcal{C}^r}$. We also have $reduc(c) \equiv_c c$.

Remark: if a cube c_0 does not contain any variable (i.e. $c_0 \in \mathcal{C}_0$) then $reduc(c_0) = c_0$.

Example 5: It is clear that *reduc({a(x),a(1)}) = {a(1)}*, but *reduc({a(1,x),a(y,2)}) = {a(1,x),a(y,2)}*.

The order relation $\leq_c$ formalizes information enrichment; two cubes are equivalent iff they capture the same piece of knowledge. As any information can be represented by a reducible cube, we will define the infimum and supremum operators on $\mathcal{C}^r$.
We adopt the approach of (Huet 1976) and (Lassez, Maher, & Marriott 1987) which allows a lattice on the terms algebra to be defined properly thanks to the *anti-unification* operator.

Definition 7 Let Φ be any bijection from $Term \times Term$ onto Var.

$antiunif_\Phi(f(s_1,...,s_m), f(t_1,...,t_m)) =_{def}$
$\quad f(antiunif_\Phi(s_1,t_1),...,antiunif_\Phi(t_m,s_m))$
$\quad$ for every function or constant symbol f,
$antiunif_\Phi(s,t) =_{def} \Phi(s,t)$ otherwise.

This definition is extended to anti-unification on literals and finally on cubes as a mapping: $antiunif_\Phi : \mathcal{C} \times \mathcal{C} \rightarrow \mathcal{C}$ which is associative and commutative.

Example 6: $p(x, g(y,b))$ is the anti-unified literal of $p(a, g(a,b))$ and $p(1, g(b,b))$.
In fact, anti-unification allows the infimum to generalize the terms so as to properly enrich the set intersection on the cubes. The result of the anti-unification of two cubes c_1 and c_2 is defined as the union of the anti-unification of every couple (l_1, l_2) based on the same predicate name and such that l_1 belongs to c_1 and l_2 belongs to c_2.

Definition 8 Let c_1 and c_2 belong to $\mathcal{C}^r$. The infimum and supremum operators $\boxed{\cup_c}$ and $\boxed{\cap_c}$ are defined on $\mathcal{C}^r$ as follows:

$c_1 \cup_c c_2 =_{def} reduc(c_1 \cup c_2)$;
$c_1 \cap_c c_2 =_{def} reduc[antiunif_\Phi(c_1, c_2)]$.

Theorem 5 $(\mathcal{C}^r, \leq_c, \cup_c, \cap_c)$ is a lattice.

Remark: by a simple duality argument, a lattice structure on cubes can be derived from the lattice structure defined on clauses (modulo $\equiv_c$) by Plotkin (see (Plotkin 1970) page 163). This lattice on $\mathcal{C}_{/\equiv_c}$ includes the present lattice on $\mathcal{C}^r$ (up to variable renaming) but the definitions of infimum and supremum are presented here in a more mathematical way allowing an easier implementation to be made, as the extension of Huet's recursive anti-unification algorithm to the cubes lends itself well to a CLP implementation. Thus the cube approach proposes a more easily computable definition of the lattice than the initial definition of Plotkin. Moreover this definition is restricted to a class of cubes that fits properly to capture the knowledge of our applications; it has also been implemented in Prolog.

Corollary 6 $(\mathcal{C}_0, \subset, \cup, \cap)$ is a sublattice of $(\mathcal{C}^r, \leq_c, \cup_c, \cap_c)$.

First-order FCA

We are now ready to define a new formal concept analysis model in which the attributes of the objects are captured by a set of literals from a first order logical language instead of literals from a propositional language. Thus an object of the context is characterized by a logical cube.

Definitions 8 A $\boxed{\text{first order context}}$ is a pair (O, ξ) where O is finite set of objects, and ξ is a mapping from O onto $\mathcal{C}^r$.
The definition is an adaptation of the propositional case of Definition 1: $\mathcal{P}(P)$ corresponds to $\xi(O)$. Each object o in O has one and only one image $p = \xi(o)$ in $\mathcal{C}^r$ which represents the set of properties of o.

Definitions 9 The dual operators ' and ° between O and $\xi(O)$ are defined by:

$A' =_{def} \cap_{c\ o_i \in A} \xi(o_i)$

$B^\circ =_{def} \{o_i \in O | B \leq_c \xi(o_i)\}$.

The dual operator from properties to objects, denoted as ° and the changing of ζ in ξ are the only differences between classical and first order definition of a context. Thanks to Corollary 7, is is easy to see that if $B \in \mathcal{C}_0$ then $B^\circ \equiv B'$ in the sense of classical FCA. Furthermore, the lattice $(\mathcal{P}(P), \subset, \cup, \cap)$ can be replaced by $(\mathcal{C}^r, \leq_c, \cup_c, \cap_c)$ (and the proofs of each statement are reduced to mere copies).

Proposition 8 The dual operators verify (i) to (v) conditions of Proposition 1.

Definitions 10 A first order concept is a pair (A, B) $A \subset O$, $B \in \mathcal{C}^r$ such that:

$A' = B$ (up to variable renaming)

$B^\circ = A$.

The set of all first-order concepts defined by the context (O, ξ) is denoted as $\mathcal{L}^1$.

Proposition 8 For first-order concepts (A_1, B_1) and (A_2, B_2) the relation defined by: $(A_1, B_1) \sqsubseteq (A_2, B_2) \Leftrightarrow A_1 \subseteq A_2 (\Leftrightarrow B_2 \leq_c B_1)$ is an order relation on $\mathcal{L}^1$.

Definitions 11 The supremum: $\sqcup$ and infimum: $\sqcap$ are respectively defined on $\mathcal{L}^1$ as follows:

$(A_1, B_1) \sqcup (A_2, B_2) =_{def} ((A_1 \cup A_2)'^\circ, B_1 \cap_c B_2)$

$(A_1, B_1) \sqcap (A_2, B_2) =_{def} (A_1 \cap A_2, (B_1 \cup_c B_2)^{\circ\prime})$

Theorem 9 $(\mathcal{L}^1, \sqsubseteq, \sqcup, \sqcap)$ is a lattice.

Back to the application of system analysis, with the same context C_2, one can determine two new 1LFCA concepts ② ' and ⑤ '. They represent the generalized attributes $opo(Var1)$, $cr(Var2)$ and $omouv(Var3)$ (with Var_i variables) which were searched for.

Corollary 10 1LFCA is an extension of FCA.

Moreover it is easy to notice that if a first order context (O, ξ) contains no variable, it can be considered as a classical context and its classical concept lattice, say L^0 can be computed. Due to the presence of functional terms and thanks to the anti-unification, the computation of its first-order concept lattice, $\mathcal{L}^1$ may induce the presence of variables, hence $L^0 \subseteq \mathcal{L}^1$, as it is verified in the system analysis.

1LFCA Implementation

The FCA algorithms were adapted to the 1LFCA definitions with respect to the Cube lattice program, this gave the first first-order logic concept lattice generator

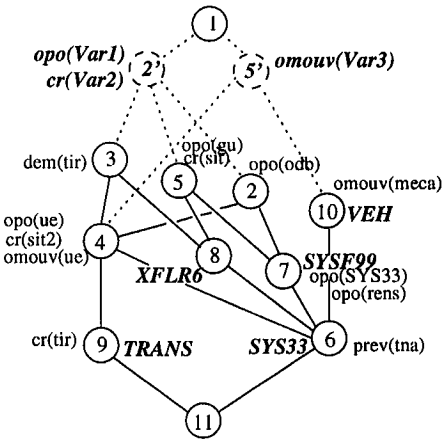

Figure 4: The first-Order Concept Lattice $\mathcal{L}^1$

(it allowed the system analysis of figure 4 to be computed). This extension is a recent result and it has to be more intensively experimented. In particular, the powerful cube tool will allow inner logical constraints to be taken into account thanks to variables: in the cube $\{foo(Var1, Var2), bar(Var2, trick)\}$ *foo* and *bar* are linked through variable $Var2$. Furthermore, a comparison between the 1LFCA capabilities and those of proximate models such as logical scaling (Prediger 1997) has been launched.

Indeed, the favorite domain of FCA and 1LFCA is the symbolic knowledge, but thanks to the CLP experiments, many improvements are expected for the analysis of large numerical databases.

Conclusion

The integration of the *Cube* model – a lattice structure on conjunctions of first-order literals – in FCA constitutes the core of the design of a complete first-order FCA based on the definition of first-order formal contexts.

The next step will be the extension of the Cube model to a Constraint-Cube model – a lattice of constrained cubes – so as to offer a powerful numerical and symbolic data analysis and knowledge discovery on *constrained first-order contexts.*

The current study focuses on first order context based rules generation which related to the classical approaches of the ILP[5] community.

From the theoretical point of view, the relations between the changes of the context and the correlated evolutions of the concept lattice is a main objective. The new applications are the pilot activity modeling (in which context dependent incidents are searched for), and the correlations between the context and the interactions between a person and a machine.

[5] Inductive Logic Programming

References

Arnaud, A., and Nicole, P. 1662. *La logique ou l'art de penser.* Paris: Gallimard, 1992. in French.

Barbut, M., and Monjardet, B. 1970. *Ordre et classification : Algèbre et combinatoire.* Collection Hachette Université. Paris: Hachette. in French.

Birkhoff, G. 1940. *Lattice Theory.* ACM.

Burmeister, P. 1987. *Programm zur Formalen Begriffsanalyse einwertiger Kontexte.* TH Darmstadt. English Version.

Carpineto, C., and Romano, G. 1996. A lattice conceptual clustering system and its application to browsing retrievial. *Machine Learning* 24:95–122.

Chaudron, L., and Barès, M. 1997. Interoperability of systems: from distributed information to cooperation. In *15th IJCAI'97 Workshop: "IA in Distributed Information Networking".*

Chaudron, L., and Maille, N. 1999. Le modéle des cubes : représentation algébrique des conjonctions de propriétés. Technical Report 1/7601.43 DCSD-T, Onera-Cert. in French. See:`ftp.cert.fr:/pub/chaudron/rapportcubes99.ps.gz`.

Chaudron, L.; Cossart, C.; Maille, N.; and Tessier, C. 1997. A purely symbolic model for dynamic scene interpretation. *International Journal on Artificial Intelligence Tools* 6(4):635–664.

Davey, B., and Priestley, H. 1990. *Introduction to Lattices and Order.* Cambridge University Press.

Ganter, B., and Wille, R. 1996. *Formale Begriffsanalyse: Mathematische Grundlagen.* Berlin–Heidelberg: Springer–Verlag. In German.

Guigues, J.-L., and Duquenne, V. 1986. Familles minimales d'implications informatives résultant d'un tableau de données binaires. *Math. Sci. Hum.* 95:5–18. In French.

Huet, G. 1976. *Résolution d'équations dans des langages d'ordre 1,2,...,ω.* Ph.D. Dissertation, Université de Paris VII. Thèse d'État (in French).

Ichino, M., and Ono, Y. 1998. A new feature selection method to extract functional structures from multidimensional symbolic data. *IEICE transactions on information and systems* E81-D(6):556–564.

Ichino, M., and Yaguchi, H. 1998. *Data Science, Classification and related Methods.* Springer-Verlag. chapter Symbolic Pattern Classifiers Based on the Cartesian System Model, 358–369.

Lassez, J.-L.; Maher, M.; and Marriott, K. 1987. *Foundations of Deductive Databases and Logic Programming.* J. Minker. chapter Unification Revisited.

Pawlak, Z. 1991. *Rough Sets.* Dordrecht: Kluwer Academic.

Plotkin, G. D. 1970. *Machine Intelligence*, volume 5. Edinburgh: Edinburgh University Press. chapter 8. A note on inductive generalization, 153–163. B. Meltzer and D. Michie ed.

Prediger, S. 1997. Logical scaling in formal concept analysis. In Lukose, D.; Delugach, H.; Keeler, M.; Searle, L.; and Sowa, J. F., eds., *Conceptual structures: Fulfilling Peirce's dream*, number 1257 in Lecture Notes in Artificial Intelligence. Berlin–Heidelberg–New York: Springer–Verlag.

Skowron, A., and Polkowski, L. 1998. *Rough Sets in Knowledge Discovery.* Heidelberg: Physica-Verlag.

Thayse, A., and col. 1989. *Approche logique de l'Intelligence artificielle.* Paris: Dunod. in French.

Vere, S. 1975. Induction of concepts in the predicate calculus. In *IJCAI'75 Proceedings of the 4th Intl. Conf. on Artificial Intelligence*, 281–287.

Zickwolf, M. 1991. *Rule exploration: First Order Logic in Formal Concept Analysis.* Ph.D. Dissertation, THD Darmstadt University. Short English version, nr1580, June 1993.