

# On sets of terms: a study of a generalisation relation and of its algorithmic properties

Colin DE LA HIGUERA,

Marie-Catherine DANIEL-VATONNE \*

*Département d'Informatique Fondamentale (DIF) LIRMM, 161 rue Ada,  
34 392 Montpellier Cedex 5, France. Internet: delahiguera@lirmm.fr*

**Abstract.** Signatures, as introduced to cope with semantics of programming languages, provide an interesting framework for knowledge representation. They allow structured and recursive properties that attribute representation does not permit, and at the same time seem to be algorithmically treatable. We recall how a generalisation relation on terms can be introduced by means of the classical special symbol  $\Omega$ , whose meaning is "indeterminate", and the partial order inductively defined on the terms. This formalism can nevertheless not take into account negation or disjunction. We provide in this paper a generalisation relation on sets of terms to withdraw these restrictions. This relation has different definitions (syntactical, set-theoretic, and based on a closed world assumption) that are proved to be equivalent for specific classes of signatures. They yield an equivalence relation whose canonical form is studied. We also study the combinatorial and algorithmic issues of the generalisation relation: the following problems are proved to be decidable (on finite data) but generally at least NP-complete: Does set  $A$  generalise set  $B$ ? Are sets  $A$  and  $B$  equivalent? Is set  $B$  the canonical form of set  $A$ ? Compute the negation of set  $A$ . . . . In the case where one of the sets contains only terms maximal for the generalisation relation, the first three problems are shown to be polynomial.

## 1. Introduction

The question of choosing a data representation formalism arises immediately in the field of machine learning, or even more generally when one is interested in knowledge representation. The choice of the appropriate formalism leads to different algorithmic possibilities and specific expressive power. Two main trends have been followed: attribute-value representations have been well studied for several years: they lead to polynomial algorithms solving clustering, or learning problems (see [26] and [32]). On the other hand researchers in Artificial Intelligence are concerned with working on powerful and expressive representations: these are often related with first order logic (for instance as in [7], [21], [27], [28] or [31]), and the descriptions can be seen as labelled graphs [29]. The algorithmic treatment of these graphs is alas far too expensive [13], so a typical idea is to work on subsets of graphs with good combinatorial (hence algorithmic) properties ([8], [23]). These graphs turn out quite naturally to be tree-like. Our feeling is that one can gain on both points (expressiveness and complexity) by using directly a representation language intermediate to

---

\* Current address: IREMA, 15 Av. R. Cassin, BP 7151, 97715 St-Denis messag. Cedex 9 La Réunion, France. Internet: mcdv@univ-reunion.fr

the attribute-value one and the first-order logic one. The mathematical background for this language is algebraic ([10], [16]), the tree-like structures are terms, for which a syntactic partial order is defined, corresponding to a natural generalisation. This formalism has been extensively studied in [11]. The purpose of this paper is to investigate this generalisation, to extend it to set-generalisation and to study its algorithmic consequences.

The idea of representing knowledge (objects or concepts) through terms is not new: in Artificial Intelligence Michalski [25] introduced terms in a language that also used (inner) disjunction and some form of negation. These are first-order terms, *i.e.* they use variables to describe unknown or linked information.

Early work in the seventies ([24, [31]) considered convincingly the operations of generalisation (also called antiunification or disunification) and specialisation (unification) on such first order terms. This trend of ideas was followed by more work done in the direction of Machine Learning, where disjunction and negation were studied in [9] and [22]. These ideas were studied in an unsorted world (corresponding to PROLOG terms for instance), although these results usually extend to many-sorted signatures. Nevertheless adding sorts is a reasonable step when wanting to describe the real world : any given term will have a meaning, whereas in traditional Knowledge Representation languages (as [8] or [29] for instance) one can construct descriptions by using symbols in any manner: this can lead to the construction of meaningless terms. Another step further (that we will not follow in this paper) is to allow the sort system to be eased : variable arities, partial order on types, and arguments whose order is flexible [1]. But certain operations become then more difficult to define (computing the set of specialisations, the negation...).

In this paper we will consider many sorted signatures : a symbol has only one profile (one type, one arity). Take for example a block world, where we wish to describe stacks of objects. A recursive definition of admissible stacks could be:

A *stack* is either empty or obtained by putting an *object on top* of a stack.

An *object* is a *cube* or a *ball*, and is either *red*, *blue* or *green*.

Thus a specific instance of a stack (a blue ball on top of a red cube) could be described by the term

*on-top (blue, ball, on-top (red, cube, empty))*

Easy concepts (corresponding to subsets of the set of all stacks) can be described by allowing a special constant  $\Omega$ , whose intuitive meaning is "indeterminate" or "unknown". Thus the set of all stacks of height at least 1, and with a red object on the top can be described by

*on-top (red,  $\Omega$ ,  $\Omega$ )*

The two  $\Omega$ s of this term are typed: the first one corresponds to "any shape", the second to "any stack including the empty one". This minimal language is sufficient to treat conjunctions of conditions (*blue* and *ball...*); a generalisation relation can be defined on terms, intuitively corresponding to set inclusion, and computable in an easy way, by means of a term  $t_1$  generalises a term  $t_2$ , if there are independent substitutions of some  $\Omega$ s in  $t_1$  yielding  $t_2$ . Furthermore this generalisation provides the set of all terms of type  $s$  with a median semi-lattice structure [4]. Yet to enrich the representation language one must consider other logical operators than the simple conjunction: disjunction can be described by sets of terms such as

$\{(on-top (red, ball, \Omega), on-top (green, cube, \Omega))\}$

that describes the set of stacks who have as top object either a red ball or a green cube. The alternative disjunction proposed for example in [1] or [21] is the *inner* disjunction, where one would be able to describe a concept such as

*on-top ([red or blue], ball,  $\Omega$ )*

This attractive route will not be followed in this paper because of the computational problems it leads to [25]. The second logical operator one can wish to incorporate into a representation language is negation. This is required in Machine Learning to induce concepts through negative information (counter-examples) as in [17] or [26]. Defining negation starts with accepting or refusing the Closed World Assumption (*CWA*). Computer science, through Databases [30] or PROLOG-like languages [24] tends to use this assumption quite systematically, even though it does not fit with incremental systems: if new facts are added (that could not be described before) then the result of the negation may change. The alternative is not to compute the result of the negation, *i.e.* keep the described concept in implicit (negative) form. We will in this paper suppose the signature fixed and base ourselves on the Closed World Assumption: only the terms constructible by this signature will be taken as meaningful. We will study in this paper the negation of a term and of a disjunction of terms. For instance

$$\text{not } (\text{on-top } (\text{red}, \text{ball}, \Omega)) = \{ \text{empty}, \text{on-top } (\text{blue}, \Omega, \Omega), \\ \text{on-top } (\text{green}, \Omega, \Omega), \text{on-top } (\Omega, \text{cube}, \Omega) \}$$

In the case of first order terms, and also using the closed world-assumption, it is proved (in [9] or [22]) that the explicit (positive) negation of a term may not be finite. The negation of a term  $f(x, x)$  is indeed infinite as soon as the set of all terms is so. A first way to make sure that negation introduces finite sets, is to consider constrained terms [9], where constraints concern the variables. We avoid the problem and many others by using *linear* terms only (each  $\Omega$  can be seen as a first order variable, that is only allowed to appear once). The advantage is computational facilities : one does not have to tackle occur-check problems related with unification, and all basic operations on terms are linear. Of course the language is not as powerful as the one where variables are used to define "roles" [27]. Such terms have also been investigated in [22] where they are called *unrestricted*: the authors prove that the problem of computing the explicit negation for such terms is decidable. The natural question that arises is the cost of such a computation. We will prove in the last section of the paper, among other complexity results that the cost is exponential.

The next section contains some mathematical preliminaries and notations. In section 3 we will give most technical definitions and properties concerning signatures, terms, and relationships between terms. Most of these will be classical for readers acquainted with Algebraic Semantics. Also a specific substitution will be defined whose purpose will be to specialise in a width-first fashion. We will through an example illustrate the description power of our model yet also show why a normal form on terms needs to be introduced.

Section 4 will provide us with different definitions of generalisation between sets of terms. It will be shown that these definitions can be in certain circumstances equivalent. The symmetrical closure of this generalisation relation will yield an equivalence between sets for which a canonical set will be proposed and studied. A logical counter-part to this is that the negation of a term (and of a set of terms) can be defined. This gives sense to the following questions: Does set  $A$  generalise set  $B$ ? Are sets  $A$  and  $B$  equivalent? Is set  $B$  the canonical form of set  $A$ ? Compute the negation of set  $A$ ... The algorithmic study of these questions is done in section 5. It can be argued that sets of terms define disjunctions, and that set generalisation is an extension to terms of implication in propositional calculus. This leads to the fact that the above problems are at least as difficult as their counterpart in propositional calculus. We have nevertheless chosen to give the complete proofs here, and use this remark only in an informal way, as this allows to keep the present paper self-contained.

## 2. Mathematical preliminaries

Notation  $\stackrel{\text{def}}{=}$  will be used when introducing some new object or property.

For each integer  $n$  we denote  $[n]$  the possibly empty set  $\{1, \dots, n\}$  of non null integers less or equal to  $n$ .

For a given set  $S$ ,  $S^*$  is the *free monoid generated by  $S$* , i.e. the set of all words written on alphabet  $S$ , including the *empty word* denoted  $\varepsilon$ .

*Trees* will only be used in this paper to build intuition on terms; we nevertheless recall that finite trees and terms are equivalent and that trees can be defined in the following way:

Let  $P$  denote the set of non null integers. A tree domain  $D$  is a subset of  $P^*$  such that:

- (i) if  $n_1 n_2 \dots n_k n \in D$ , then  $n_1 n_2 \dots n_k \in D$
- (ii) if  $n_1 n_2 \dots n_k n \in D$  and  $0 < m < n$  then  $n_1 n_2 \dots n_k m \in D$

A tree is then a mapping from a tree domain  $D$  into a given set of symbols. The tree is finite if and only if the tree domain is a finite set.

We will be dealing with partial orders: for a given set  $A$  and a relation  $\leq$  on  $A$ ,  $(A, \leq)$  is a *partially ordered set* if  $\leq$  is reflexive, antisymmetric and transitive. For a given subset  $A$  of  $E$ , an element  $m$  of  $A$  is *maximal* if no other element  $a$  in  $A$  is such that  $m < a$ .

An *antichain* in  $A$  is a subset of  $A$  of two by two incomparable elements.

## 3. Signatures and terms

We will in this first section give the main mathematical tools our model for data representation requires. These are issued from works on algebraic semantics of programs ([10], [14], [16]). The recursive definitions that many-sorted algebras deal with have since been used in other areas, such as for definition and inference of recursive types [3], or as basis of a programming language [1]. Here terms will be used to represent data, and sets of data. The construction of  $\Omega$ -complete  $F$ -algebras (as in [16] for instance) leads to the introduction of a special symbol denoted  $\Omega$  and a partial relation whose direct interpretation is "has less information than". In the context of data representation this relation on terms will indicate generalisation: thus most specific generalisations and most general specialisations will be well-defined. We will close this section through an example which will lead to an improvement of the generalisation relation *via* the introduction of an equivalence relation between terms. A usual way to define semantics is to associate to a description formalism a standard model-theoretic semantics as in [1] and [19]. We will not technically discuss the semantical aspects here: hence what "to generalise" or to "specialise" really mean depends on common sense only.

**Definition 1** A signature  $\Sigma$  is a foursome  $(S, F, \alpha, \sigma)$  where:

$S$  is a finite set whose elements are called sorts.

$F$  is a finite set whose elements are called function symbols.

$\alpha$  is a mapping from  $F$  into  $S^*$ .  $\alpha(f)$  will be called the arity of  $f$ .

$\sigma$  is a mapping from  $F$  into  $S$ .  $\sigma(f)$  will be called the type of  $f$ .

We shall denote by  $F^s$  the subset of  $F$  of symbols with type  $s$ .

We use the notation  $f: s_1 \times \dots \times s_n \rightarrow s$ , to denote the fact that  $f$  has arity  $s_1 \dots s_n$  and type  $s$ . " $s_1 \times \dots \times s_n \rightarrow s$ " is called a *profile*. When the arity of a symbol  $f$  is equal to the empty word  $\varepsilon$ , this will be written  $f: s$ . In contrast with some other definitions of a signature, our own definition requires each function symbol to have exactly one profile  $s_1 \times \dots \times s_n \rightarrow s$ .

**Definition 2**  $\mathbf{T}_{\Sigma}^s$  is the set of all terms of type  $s$  over a signature  $\Sigma$ . It can be defined inductively by the following inference rule<sup>1</sup>:

$$\frac{f : s_1 \times \dots \times s_n \rightarrow s, t_1 : s_1, \dots, t_n : s_n}{f(t_1, \dots, t_n) : s}$$

We denote  $t : s$  for  $t \in \mathbf{T}_{\Sigma}^s$ .

A term is therefore always some  $f(t_1, \dots, t_n)$ . When  $n = 0$  we can write  $f$  instead of  $f()$ . Hence a term has a sort equal to the one of the symbol heading the term. From the above definition it follows as a base case that if  $f$  is a constant ( $\alpha(f) = \varepsilon$ , or alternatively  $f : s$ ),  $f$  is also a term of type  $s$ . There will usually be a base sort 0. Also the depth of a term  $t$  is defined as follows:

**Definition 3**  $\text{depth}(f) = 0$

$\text{depth}(f(t_1, \dots, t_n)) = 1 + \max(\text{depth}(t_1), \dots, \text{depth}(t_n))$  if  $n > 0$ .

**Example 1** With  $S = \{0, 1, 2\}$ ,  $F = \{a, b, c, d, f\}$  and  $f : 1 \times 0 \times 2 \rightarrow 0$ ;  $c : 0$ ;  $a, b : 1$ ;  $d : 2$  we have, for instance:  $f(a, f(b, c, d), d) \in \mathbf{T}_{\Sigma}^0$

Typically this term can be represented by the tree depicted in Figure 1. ♦

The definition of trees has been given in section 2. We will not describe here in detail the tree representation of terms, but use it for examples. For more details on tree-term relationship reader can refer to [10] or [11].

We now introduce a special symbol  $\Omega$  that will enable us to define a partial relation among terms<sup>2</sup>.

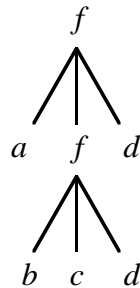
**Definition 4** For each sort  $s$  in  $S$  we add  $\Omega_s$  whose type is  $s$  and arity  $\varepsilon$ . The set of terms of type  $s$  on  $F \cup \{\Omega_s / s \in S\}$  will be denoted  $\mathbf{T}_{\Sigma, \Omega}^s$ .

For each sort  $s$ , we define a partial order  $\leq_s$  on  $\mathbf{T}_{\Sigma, \Omega}^s$  as follows:

**Definition 5** For  $t, t' \in \mathbf{T}_{\Sigma, \Omega}^s$

$$t \leq_s t' \stackrel{\text{def}}{\equiv} \begin{array}{l} \text{either } t = \Omega_s \\ \text{or } t = f(t_1, \dots, t_n), t' = f(t'_1, \dots, t'_n) \\ \text{and } \forall i \in [n] t_i \leq_{s_i} t'_i \text{ with } t_i, t'_i \in \mathbf{T}_{\Sigma, \Omega}^{s_i} \end{array}$$

One can notice that there is a minimum element in  $(\mathbf{T}_{\Sigma, \Omega}^s, \leq_s)$  which is  $\Omega_s$ .



**Figure 1**

<sup>1</sup> Other classical definitions can be found in [11].

<sup>2</sup> This is a classical way [9,13] to approximate eventually infinite terms. We will not be dealing with these in this paper.

In the sequel, we will omit the index of  $\leq$  and  $t \leq t'$  will mean:  $t, t'$  share a same type  $s$  and  $t \leq_s t'$ .

Other definitions go as follows:

**Definition 6** For every  $t$  in  $\mathbf{T}_{\Sigma, \Omega}^s$

$t'$  is an instance of  $t$  if  $t'$  contains no  $\Omega$  as subterm, and  $t \leq t'$

$\text{Inst}(t)$  is the set of all instances of  $t$ , and for a set  $A$  (included in  $\mathbf{T}_{\Sigma, \Omega}^s$ )  $\text{Inst}(A)$  is the set of terms that are instance of any term in  $A$ .

A term is elementary if it contains exactly one symbol different from  $\Omega$ , hence is either a constant in  $F$  or a term  $f(\Omega_{s_1}, \dots, \Omega_{s_n})$ .

The set of all elementary terms of type  $s$  will be denoted  $\text{elem}(s)$ .

**Example 2** With  $S=\{0, 1, 2\}$ ,  $F=\{a, b, c, d, f\}$ ,  $f: 1 \times 0 \times 2 \rightarrow 0$ ;  $c : 0$ ;  $a, b:1$ ;  $d: 2$

we have :  $f(a, c, d)$  is an instance of  $f(\Omega_1, \Omega_0, d)$

$\text{Inst}(f(\Omega_1, c, \Omega_2)) = \{f(a, c, d), f(b, c, d)\}$

$c$  is elementary

$\text{elem}(0) = \{f(\Omega_1, \Omega_0, \Omega_2), c\}$ . ♦

Questions such as "how many terms are there of this sort?" or "how many instances of this term are there?" can be answered by considering only the signature  $\Sigma$ . We will require all sorts to be useful, *i.e.*  $\sigma$  to be an onto mapping ( $\forall s \in S, F^s \neq \emptyset$ ). We can also differentiate the signatures for which all terms can be instantiated:

**Definition 7** A signature  $\Sigma=(S, F, \alpha, \sigma)$  is in reduced form if for each  $s$  in  $S$ , each term in  $\mathbf{T}_{\Sigma, \Omega}^s$  has at least one instance.

The proof of the following lemma is straightforward and shall be omitted<sup>3</sup>:

**Lemma 1** A signature  $\Sigma=(S, F, \alpha, \sigma)$  is in reduced form if and only if  $\forall s \in S, \text{Inst}(\Omega_s) \neq \emptyset$ .

$\mathbf{T}_{\Sigma, \Omega}^s$  is a well-ordered set. When  $S$  is in reduced form, instances are maximal elements of  $\mathbf{T}_{\Sigma, \Omega}^s$ . The elementary operation is the comparison in between two terms, but the typical lattice operations can also be defined. The *meet* ( $\wedge$ ) of two terms is easy to compute, the *join* ( $\vee$ ) only exists when 2 terms are *compatible*:

**Proposition 1** For  $t, t' \in \mathbf{T}_{\Sigma, \Omega}^s$ :

if  $t = \Omega_s$  or  $t' = \Omega_s$  then  $t \wedge t' = \Omega_s$

if  $t = f(t_1, \dots, t_n)$  or  $t' = f'(t'_1, \dots, t'_n)$  and  $f \neq f'$  then  $t \wedge t' = \Omega_s$

if  $t = f(t_1, \dots, t_n)$ ,  $t' = f(t'_1, \dots, t'_n)$  then  $t \wedge t' = f(t_1 \wedge t'_1, \dots, t_n \wedge t'_n)$

**Definition 8** For  $t, t' \in \mathbf{T}_{\Sigma, \Omega}^s$ :

$t$  and  $t'$  are compatible ( $t \nabla t'$ )  $\stackrel{\text{def}}{=} \equiv$

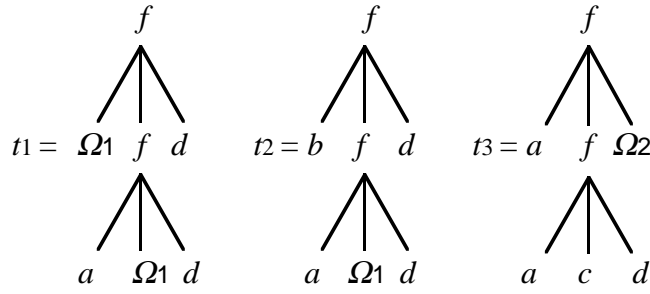
either  $t = \Omega_s$

or  $t' = \Omega_s$

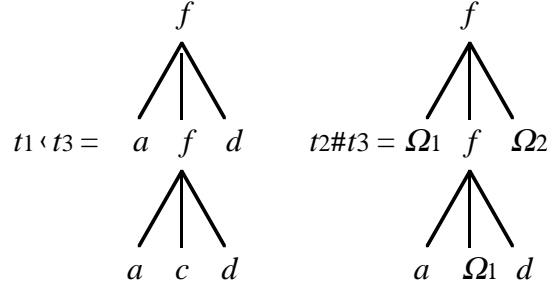
or  $t = f(t_1, \dots, t_n)$ ,  $t' = f(t'_1, \dots, t'_n)$  and  $\forall i \in [n], t_i \nabla t'_i$

If  $t$  and  $t'$  are not compatible we will write  $t \nexists t'$ .

<sup>3</sup> It also yields that checking if a given signature is in reduced form is a decidable question: it corresponds to checking if in a context-free grammar all variables generate a non-empty language. An algorithm can be found in [2].



**Figure 2**



**Figure 3**

**Proposition 2** For  $t, t' \in \mathbf{T}_{\Sigma, \Omega}^s$ :

if  $t \nabla t'$  then

either  $t = \Omega_s$  and  $t \vee t' = t'$

or  $t' = \Omega_s$  and  $t \vee t' = t$

or  $t = f(t_1, \dots, t_n)$ ,  $t' = f(t'_1, \dots, t'_n)$  and  $t \vee t' = f(t_1 \vee t'_1, \dots, t_n \vee t'_n)$

if  $t \nabla t'$  then  $t \vee t'$  is undefined.

The proofs of propositions 1 and 2 can be found in [16].

**Example 3** For the signature  $\Sigma_{\Omega}$  obtained by adding  $\Omega_0, \Omega_1, \Omega_2$  to  $\Sigma$  of example 1, let  $t_1, t_2$  and  $t_3$  be the terms depicted as trees in Figure 2.

We have  $t_1 \leq t_2$ ,  $t_2 \nabla t_3$ , but  $t_1 \nabla t_3$  and the resulting terms  $t_1 \vee t_3$  and  $t_2 \wedge t_3$  are represented in Figure 3.  $\blacklozenge$

In the sequel, the index of symbol  $\Omega$  will be omitted when context allows it.

The meet and join properties give  $\mathbf{T}_{\Sigma, \Omega}^s$  a specific combinatorial structure:

**Definition 9** [4] A median semi-lattice is a meet semi-lattice such that:

(i) every principal ideal  $\{x / x \leq a\}$  is a distributive lattice

(ii) any three elements have an upper bound whenever each pair of them does.

**Theorem 1**  $\mathbf{T}_{\Sigma, \Omega}^s$  is a median semi-lattice.

**Proof** (i) A lattice  $L$  is distributive  $\Leftrightarrow \forall a, b, c \in L, a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$  [6].

If two terms  $t_1$  and  $t_2$  belong to a same ideal, they are compatible: hence for three terms  $a, b, c$  in a same ideal,  $a \vee (b \wedge c)$  is defined, and so are  $a \vee b$  and  $a \vee c$ . And by induction on the depth of  $a$  it follows:

Base case

$\text{depth}(a) = 0$ ,

if  $a = \Omega$ ,  $a \vee (b \wedge c) = b \wedge c$  and  $a \vee b = b$ ,  $a \vee c = c \Rightarrow (a \vee b) \wedge (a \vee c) = b \vee c$ .

if  $a = f : s$ , as  $a \nabla (b \wedge c)$ ,  $b \wedge c = \Omega$  or  $b \wedge c = f$ . If  $b \wedge c = f$  then we have  $b = f$  or  $c = f$ , and  $(a \vee b) \wedge (a \vee c) = f$ . If not  $(a \vee b) \wedge (a \vee c) = \Omega$ .

*Induction step*

Suppose  $depth(a) = k + 1$ . Then there are two cases :

-  $b \wedge c = \Omega$ , and we are back to the base case

-  $b \wedge c = f(u_1, \dots, u_n)$ . So  $b = f(b_1, \dots, b_n)$ ,  $c = f(c_1, \dots, c_n)$ . By compatibility, we also have  $a = f(a_1, \dots, a_n)$ , and by the induction hypothesis  $\forall i \in [n], a_i \vee (b_i \wedge c_i) = (a_i \vee b_i) \wedge (a_i \vee c_i)$ . This yields by construction that  $a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$ .

(ii) Take any  $a, b, c$  such that  $a \vee b, a \vee c$  and  $b \vee c$  exist. Then we prove that  $a \vee (b \vee c)$  exists (associativity of  $\vee$  yields that  $a \vee b \vee c$  exists). By induction on  $depth(a)$ :

*Base case*

$depth(a) = 0$ ,

if  $a = \Omega$ ,  $a \vee (b \vee c) = b \vee c$ .

if  $a = f$ , as  $a \nabla b$  and  $a \nabla c$ ,  $a \vee b = f$  and  $a \vee c = f$ . Thus  $a \vee (b \vee c) = f$ .

*Induction step*

Suppose  $depth(a) = k + 1 : a = f(a_1, \dots, a_n)$ . Then since  $a \nabla b$  and  $a \nabla c$ ,  $b = f(b_1, \dots, b_n)$  or  $\Omega$ , and  $c = f(c_1, \dots, c_n)$ , or  $\Omega$ , so

either  $b = \Omega$  and  $c = \Omega$ , and  $a \vee (b \vee c) = a$

or  $b \vee c = f(u_1, \dots, u_n)$  where  $\forall i \in [n] u_i = b_i \vee c_i$ , and by induction hypothesis, there exists  $\forall i \in [n], w_i = a_i \vee (b_i \vee c_i)$ , hence  $a \vee (b \vee c) = f(w_1, \dots, w_n)$ .  $\blacklozenge$

In order to specialise terms it is necessary to substitute some  $\Omega$  by a term of same type. We shall define a substitution here, which will allow to compute step by step all specialisations of a given term:

**Definition 10** Let  $t \in T_{\Sigma, \Omega}^s$

if  $t = \Omega_s$   $Sub_{\Omega}(t) \stackrel{\text{def}}{=} elem(s)$

if  $t = f$   $Sub_{\Omega}(t) \stackrel{\text{def}}{=} \{f\}$

if  $t = f(t_1, \dots, t_n)$   $Sub_{\Omega}(t) \stackrel{\text{def}}{=} \{f(u_1, \dots, u_n) / \forall i \in [n], u_i \in Sub_{\Omega}(t_i)\}$ .

This substitution can be iterated:

**Definition 11** Let  $t \in T_{\Sigma, \Omega}^s$

$Sub_{\Omega}^0(t) \stackrel{\text{def}}{=} \{t\}$

$Sub_{\Omega}^{i+1}(t) \stackrel{\text{def}}{=} \{Sub_{\Omega}(t') / t' \in Sub_{\Omega}^i(t)\}$

Hence by this definition  $Sub_{\Omega}^1(t) = Sub_{\Omega}(t)$ . This leads to the fact that by iteration of substitution one can reach any term:

**Lemma 2**  $\forall t, t' \in T_{\Sigma, \Omega}^s, t \leq t' \Rightarrow \exists n \in \mathbb{N}, \exists u \in Sub_{\Omega}^n(t), t' \leq u$

**Proof** By induction on the depth of  $t'$

If  $depth(t') = 0$ , either  $t' = \Omega \Rightarrow t = \Omega$  so  $t' \in Sub_{\Omega}^0(t)$ , or  $t' = f \Rightarrow t' \in Sub_{\Omega}^1(t)$ .

Suppose now the lemma true for all terms  $t'$  of depth at most  $k$ , and let  $depth(t') = k+1$ , with

$t' = f(t'_1, \dots, t'_n)$ , then either  $t = \Omega$  so  $\exists u \in Sub_{\Omega}^{k+1}(t)$  with  $t' \leq u$  or  $t = f(t_1, \dots, t_n)$  and as  $t_i$

$\leq t'_i$  by induction hypothesis  $\forall i \in [n], \exists u_i \in Sub_{\Omega}^k(t_i)$  and  $t'_i \leq u_i$ .

Hence  $u = f(u_1, \dots, u_n) \in Sub_{\Omega}^k(t)$  with  $k = Max\{k_1, \dots, k_n\}$  is such that  $t' \leq u$ .  $\blacklozenge$

**Example 4** Again with  $f: 1 \times 0 \times 2 \rightarrow 0$ ;  $c: 0$ ;  $a, b: 1$ ;  $d: 2$ ; and  $\Omega$ s of all types

$$Sub_{\Omega}(\Omega) = \{f(\Omega, \Omega, \Omega), c\}$$

$$Sub_{\Omega}^2(\Omega) = \{c, f(a, f(\Omega, \Omega, \Omega), d), f(b, f(\Omega, \Omega, \Omega), d), f(a, c, d), f(b, c, d)\}$$

And term  $t = f(a, \Omega, \Omega)$  is such that  $\Omega \leq t$  and  $\exists u \in Sub_{\Omega}^2(t), t \leq u$ : one can for instance take  $u = f(a, c, d)$ . This example also shows that equality ( $t' = u$ ) instead of inequality in lemma 2 does not hold.  $\blacklozenge$

It can be noticed that this substitution corresponds to a special case of the "outside-in" set substitution as defined in [16]. Such a substitution is necessary in this case for algorithmic reasons: a breadth first research through the set of specialisations insures that (because of lemma 2) a specialised term can be reached in finite time.

There is an immediate property relating maximal terms, compatibility and substitution:

**Corollary 1**  $\forall t \in T_{\Sigma, \Omega}^s,$

$$t \nabla t' \text{ and } t' \text{ is maximal} \Rightarrow t \leq t'$$

$$t \leq t' \text{ and } t' \text{ maximal} \Rightarrow \exists n \in \mathbb{N}, t' \in Sub_{\Omega}^n(t)$$

It is reasonable to interpret this order as a generalisation<sup>4</sup> *i.e.*  $t_1 \leq t_2$  indicates that  $t_1$  generalises  $t_2$ , or  $t_2$  specialises  $t_1$ . Furthermore  $t_1$  is an *immediate generalisation* of  $t_2$  if and only if  $t_1 \leq t_2$  and  $t_1 \neq t_2$  and  $(t_1 \leq t_3 \leq t_2 \Rightarrow t_3 = t_1 \text{ or } t_3 = t_2)$ . Indeed a term can be seen as a property, so defining in intention a set of objects, and the set of objects described by  $t_2$  will be included in the set described by  $t_1$  when  $t_1 \leq t_2$ <sup>5</sup>.

For these same *common sense semantics*,  $t_1 \vee t_2$  will denote the *most general common specialisation*,  $t_1 \wedge t_2$  the *most specific common generalisation*. This aspect has already been noticed as offering interesting possibilities to using algebras in artificial intelligence [20]. The generalisation it implies is also very close to the one studied by several authors (for instance [21], [28] or [31]: instead of using the unique symbol  $\Omega$  to indicate non-determination, variables can play the part. Generalisation will in that case be described by anti-unification, and specialisation by unification. The expressiveness of the language is thus stronger than with the terms we define, but as it is pointed out for instance by Nebel [27], this has a cost: usually checking generalisation, finding the most specialised generalisation take polynomial time, but compatibility and the computation of the most general specialisation can be extremely expensive. An interesting alternative is proposed by Ait-Kaci [1] who uses terms (called *psi-terms*) defined on a somewhat different signature (a partial order on symbols, that defines a local generalisation relation between them) and who notes that variables play two roles: if a variable appears only once it will be considered as a wild card, and thus would correspond to an  $\Omega$ ; if a same variable is used in one term twice or more times then it is a tag: only the equality relation between variables is allowed and again specialisation and finding a common specialisation with another term correspond to unification. Of course if we forbid tags in the (*psi-*) terms, and take as signature a flat meet semi-lattice, the expressiveness of classical terms and *psi-terms* would be equivalent<sup>6</sup>.

<sup>4</sup> The way one reads the order varies in between the different authors. A common alternative in Knowledge Representation is to read  $R \leq R'$  as  $R'$  is more general than  $R$ . Also instead of "generalisation" some authors use "subsumption" ([22], [23]). The computational cost of subsumption has been investigated for different representation languages in [13] and [18].

<sup>5</sup> This point of view, based on the extent/intent couple of a concept has been widely addressed to by R.Wille [28]: we will only use it here to infer the basic semantics of terms and relations between terms.

<sup>6</sup> There are other differences with *psi-terms*: for instance there is no arity attached to the symbols: this allows more freedom in the construction of terms, but the closed world assumption cannot be assumed, hence there are problems with negation [8]. Also the fact the signature is many-sorted induces algorithmic differences.

Our feeling is that the basic operations should be computationally simple, and this justifies the study of the properties of the model we propose, even though this must not exclude an extension of the model to incorporate more expressive power, for example by admitting tags.

We now wish to extend this generalisation order a little further: certain situations cannot be represented with this order; these will be shown through the following example:

**Example 5** The signature we will use is described in Figure 4: it is used to represent colour-blind people.

A quick discussion about the choice of signature  $\Sigma$  above leads to the following points:

- Symbols *mother* and *father* are recursive: they allow us to consider unbounded terms and hence information concerning any generation.
- The signature has to be many-sorted, sex is an important issue, as colour-blindness is transmitted by women only.
- There exists no maximal term of base sort *person*. To allow maximal terms, the sort *male* (or *female*) would have to contain some final (constant) symbol. Such a symbol is in this case difficult to imagine. The signature  $\Sigma$  hence is not in reduced form.

Figure 5 shows a term whose interpretation could be: a human whose father is not colour-blind and who has blue eyes, and whose mother's father is colour-blind with blue eyes. But the interpretation of the (different) term in Figure 6 would equally be the same: having eyes but not knowing their characteristics, or having a mother does not add any new information. It can be shown [12] that some symbols are *mute*: they can be traced in the signature as singletons of their sort. Can these symbols be eliminated? The answer is yes for non recursive ones (like *eyes*) but no for recursive ones (like *mother* or *father*): take for instance symbol *father*; obviously eliminating it would slacken the expressive power and on the other hand no other symbol of same type can be added (what would such a symbol mean?).

$F$	$\sigma$	$\alpha$
<i>human</i>	<i>person</i>	<i>male female</i>
<i>father</i>	<i>male</i>	<i>eyesight male female</i>
<i>mother</i>	<i>female</i>	<i>male female</i>
<i>eyes</i>	<i>eyesight</i>	<i>c-blindness colour</i>
<i>cb</i>	<i>c-blindness</i>	$\epsilon$
<i>ncb</i>	<i>c-blindness</i>	$\epsilon$
<i>blue</i>	<i>colour</i>	$\epsilon$
<i>brown</i>	<i>colour</i>	$\epsilon$

Figure 4

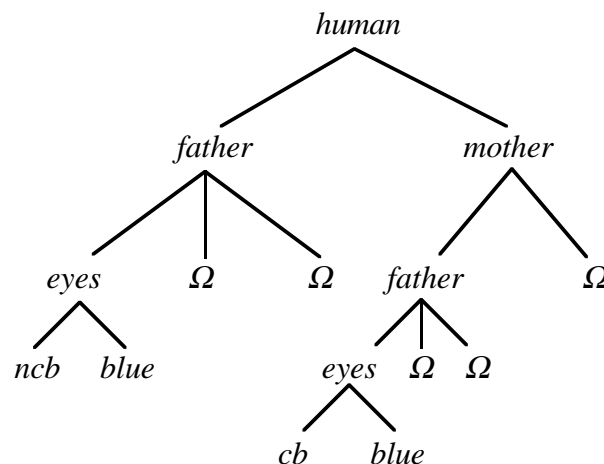
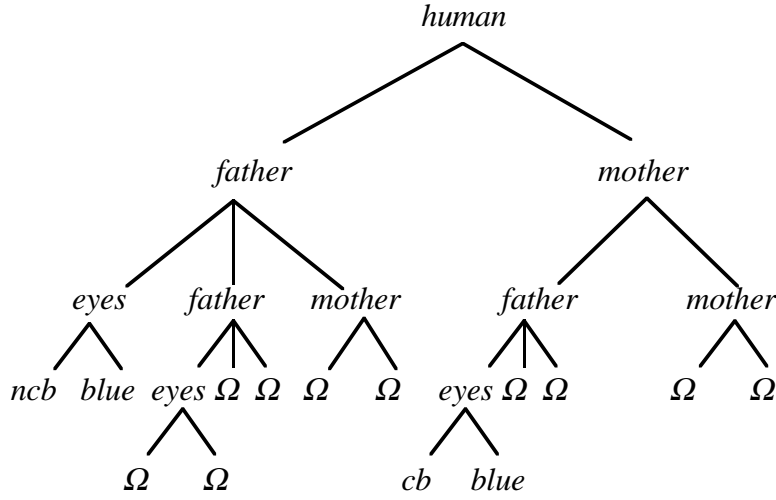


Figure 5



**Figure 6**

Therefore the presence of such symbols induces a semantical equivalence that is not described by  $\leq^7$ . ♦

A first idea yielding a more accurate generalisation relation in between two terms, taking into account the remarks from example 5 has been proposed in [12]: it consists in giving a syntactical counterpart to the reasonable equivalence between two terms, and introducing the normal form of a term by eliminating the unnecessary symbols:

**Definition 12** A symbol  $f \in F$  is mute  $\stackrel{\text{def}}{=} \forall f' \in F, \sigma(f') = \sigma(f) \Rightarrow f = f'$ .

Let  $t \in \mathbf{T}_{\Sigma, \Omega}^s$ , the normal form of  $t$  denoted  $N(t)$  is:

$\Omega_s$	if $t = \Omega_s$ or $t = f$ and $f$ is mute $t$ and $\sigma(f) = s$ or $t = f(t_1, \dots, t_n)$ and $f$ is mute and $\forall i \in [n], N(t_i) = \Omega$ and $\sigma(f) = s$ .
$f$	if $t = f$ and $f$ is not mute
$f(N(t_1), \dots, N(t_n))$	if not.

The normal form is unique, it can be computed in linear time. It preserves the order relation. For example the term depicted Figure 5 is the normal form of the one from Figure 6. One can also adapt partial order  $\leq$  (and operations  $\wedge$  and  $\vee$ ) to fit with the normal form [12]. This enables us to have a better generalisation relation between two descriptions, and in the set of equivalent descriptions to choose the easiest and smallest term: the normal form.

#### 4. A generalisation relation on sets of terms

In this section we introduce through an example the necessity of considering sets of terms. We show that the one-to-one generalisation directly induced from relation  $\leq$  is insufficient to grasp what is required from a correct generalisation relation. We give 3 possible definitions of such a generalisation and show that they coincide provided that certain conditions are satisfied. The unique relation thus obtained is shown to be decidable and

<sup>7</sup> A similar example is proposed in [22]. The author argues that the fact that one can not stop until reaching the first humans to obtain a complete description, gives an argument against using the CWA. In our case, complete description is not necessary, and it justifies working on partial descriptions.

compatible with an extension of the previously defined normal form. We conclude this section by presenting the negation of a set, and proving that it respects those properties a negation is bound to.

The following example does not contain recursion, but it will be sufficient to show why considering sets of terms is an issue.

**Example 6** We wish to represent men and women with 3 attributes: beauty (*phys.*), intelligence (*int.*), and size. For each sort 2 "values" are proposed. This is done in the signature described in Figure 7. What is the negation of term *man(beautiful, intelligent, Ω)*? Intuitively one single term can not describe it: a negation could be "being a woman", "being an ugly man" or "being a dumb man". "Being a dog" can be discarded by the Closed World Assumption: this description cannot be obtained by the signature. So to describe the negation it is necessary to represent all possibilities of incompatible terms. This can be done through taking the most general of these terms, hence a correct negation could be described through the set of 3 terms:

$$\{ woman(\Omega, \Omega, \Omega), man(ugly, \Omega, \Omega), man(\Omega, dumb, \Omega) \}.$$

But since that forces us to manipulate sets of terms, how does the generalisation relation transpose to such sets? Is one-to-one generalisation sufficient? Take for instance (for the same signature as above) the set of terms:

$$G = \{ man(ugly, \Omega, tall), man(\Omega, dumb, small) \}$$

This set generalises the completely instantiated terms:

$$man(ugly, dumb, tall) \text{ and } man(ugly, dumb, small)$$

by one-to-one use of relation  $\leq$ . But then, using again the Closed World Assumption (a man in our signature can only be tall or small!) these are all the possible instances of the term:

$$man(ugly, dumb, \Omega)$$

So this term itself must be generalised, but is not so by either term in  $G$ . Thus one-to-one generalisation is only a sufficient condition to reflect set generalisation. ♦

Not all sets of terms will need to be considered. It is reasonably easy to notice that if in a given set of terms, there are two comparable elements (take  $t \leq t'$ ), then the largest element  $t'$  can be eliminated. By renewing this operation, we obtain sets of two by two incomparable terms. Such sets are called *antichains*.

We will first define  $\mathcal{A}_0$ , the one-to-one generalisation<sup>8</sup> between sets of terms:

**Definition 13** Let  $A, A'$  be two subsets of  $\mathbf{T}_{\Sigma, \Omega}^s$   
 $A \mathcal{A}_0 A' \stackrel{\text{def}}{=} \forall a' \in A', \exists a \in A \text{ such that } a \leq a'$ .

$F$	$\sigma$	$\alpha$
<i>man</i>	<i>sex</i>	phys. int. size
<i>woman</i>	<i>sex</i>	phys. int. size
<i>beautiful</i>	<i>phys.</i>	$\epsilon$
<i>ugly</i>	<i>phys.</i>	$\epsilon$
<i>intelligent</i>	<i>int.</i>	$\epsilon$
<i>dumb</i>	<i>int.</i>	$\epsilon$
<i>tall</i>	<i>size</i>	$\epsilon$
<i>small</i>	<i>size</i>	$\epsilon$

**Figure 7**

<sup>8</sup>  $A \mathcal{A} B$  will be read  $A$  generalises  $B$ .

In the previous example we showed that one-to-one generalisation is not fine enough. Nevertheless we will require that any generalisation relation  $\#_1$  respects one-to-one generalisation, i.e.  $A \#_0 A' \Rightarrow A \#_1 A'$ .

We can now propose 3 possible extensions of  $\leq$  to sets.

The first definition is based on the idea that a set will be generalised by another if the set of instances of the first is included in the set of instances of the latter:

**Definition 14** Let  $A, A'$  be two subsets of  $\mathbf{T}_{\Sigma, \Omega}^s$   $A \#_1 A' \stackrel{\text{def}}{=} \text{Inst}(A) \supset \text{Inst}(A')$ .

The second definition we give is based on the Closed World Assumption: a set will generalise another if any term incompatible with the more specific set is also incompatible with the generalising set:

**Definition 15** Let  $A, A'$  be two subsets of  $\mathbf{T}_{\Sigma, \Omega}^s$   $A \#_2 A' \stackrel{\text{def}}{=} \text{all terms incompatible with all } a \text{ in } A \text{ are incompatible with all } a' \text{ in } A' \text{ i.e.: } \forall a \in A, t \nabla a \Rightarrow \forall a' \in A', t \nabla a'$ .

Our last definition of generalisation will be an extension of the definition of  $\#_0$ :

**Definition 16** Let  $A, A'$  be two subsets of  $\mathbf{T}_{\Sigma, \Omega}^s$   $A \#_3 A' \stackrel{\text{def}}{=} \exists n \in \mathbb{N}, A \#_0 \text{Sub}_{\Omega}^n(A')$

Relation  $\#_3$  as defined above is the least relation satisfying the following condition :

$$A \#_3 A' \Leftrightarrow \forall a' \in A', [\exists a \in A, a \leq a' \text{ or } A \#_3 \text{Sub}_{\Omega}(a')].$$

In order to increase readability when a set  $A$  is reduced to a single element  $a$ , we shall from now on forget the brackets.

**Proposition 3**  $\#_1, \#_2$  and  $\#_3$  respect one-to-one generalisation.

**Proof** Suppose we have  $A \#_0 A'$ . Then  $\forall a' \in A', \forall t \in \text{Inst}(a'), \exists a \in A$  such that  $a \leq a'$ , hence  $t \in \text{Inst}(a)$ , so  $A \#_1 A'$ . Suppose now  $A \#_2 A'$ . Then there must be some term  $t$  and some  $a'$  in  $A'$  such that  $t \nabla a'$  but  $\forall a \in A, t \nabla a$ . But as there exists some  $a$  in  $A$  with  $a \leq a'$ , then this  $a$  would be such that  $t \nabla a$ . This is clearly a contradiction, so  $A \#_2 A'$ .

Finally we have:  $A \#_0 A' \Rightarrow \forall a' \in A', A \#_0 \{a'\} \Rightarrow \forall a' \in A', A \#_3 \{a'\} \Rightarrow A \#_3 A'$ . ♦

Let us straight away notice that definition 14, which seems the easiest, based on the direct set-theoretic idea that generalisation corresponds to instance inclusion does not hold for all signatures:

Take for instance  $F = \{a, b, f, g, h\}$ ,  $a, b : 2, g, h : 1 \times \dots \times 2 \rightarrow 1, f : 1 \rightarrow 0$

No maximal term of type 0 or 1 can be constructed. In this case we have  $\forall t \in \mathbf{T}_{\Sigma, \Omega}^0$ ,  $\text{Inst}(t) = \emptyset$  so no instances can be constructed for term  $\Omega_0$ . Therefore relation  $\#_1$  becomes a tautology and is therefore meaningless.

But example 5 gives the case of a signature where recursivity does not end, and where the signature is meaningful: it must nevertheless be regarded as correct.

The two latter definitions are equivalent:

**Proposition 4**  $A \#_2 A' \Leftrightarrow A \#_3 A'$ .

**Proof** Suppose  $A \#_3 A'$  then there exists some  $a'$  in  $A'$  with  $A \#_3 a'$ . But then there exists some  $a''$  in  $\text{Sub}_{\Omega}^k(a')$  such that all  $a$  in  $A$  are incompatible with  $a''$ . Hence  $A \#_2 A'$ . Conversely

suppose  $A \not\leq_2 A'$ : then there exists  $t$  such that  $\forall a \in A, t \not\leq a$  and  $\exists a' \in A', t \leq a'$ . Hence  $t \vee a'$  exists. By lemma 2 there exists some integer  $n$  such that  $\exists u \in \text{Sub}_\Omega^n(a')$  and  $t \vee a' \leq u$ . And in this case  $\forall a \in A, u \leq a$ , hence  $A \leq_3 A'$ .  $\blacklozenge$

In the case where the signature is in *reduced* form, i.e. each term generalises at least one maximal element, we have:

**Proposition 5** *If  $\Sigma$  is a signature in reduced form ( $\forall s \in S, \mathbf{T}_{\Sigma, \Omega}^s \neq \emptyset$ ) then the three following properties are equivalent:*

- (i)  $A \leq_1 A'$
- (ii)  $A \leq_2 A'$
- (iii)  $A \leq_3 A'$ .

**Proof** We shall first prove  $A \leq_3 A' \Rightarrow A \leq_1 A'$ .  $\forall a' \in A', \forall n \in \mathbb{N}, \forall t \in \text{Sub}_\Omega^n(a'), \exists a \in A, t \leq a$ , i.e. by corollary 1, for all instances  $u$  of all  $a'$  in  $A'$  we have:  $\exists a \in A$  such that  $a \leq u$ . But this means  $a \leq u$ , so  $u$  is an instance of some  $a$  in  $A$ .

We will now prove  $A \leq_1 A' \Rightarrow A \leq_2 A'$

Suppose  $A \not\leq_2 A'$  then there exists some  $t$  in  $\mathbf{T}_{\Sigma, \Omega}^0$  such that  $\forall a \in A, t \not\leq a$ , but  $\exists a' \in A', t \leq a'$ . So  $t \vee a'$  exists, and as signature  $\Sigma$  is reduced there is some instance  $u$  of  $t \vee a'$ . But  $u$  is also necessarily incompatible with all  $a$  in  $A$ . We thus have  $\exists u \in \text{Inst}(A')$  but  $u \notin \text{Inst}(A)$ . Thus  $A \not\leq_1 A'$ .

And as from proposition 4 we have  $A \leq_2 A' \Rightarrow A \leq_3 A'$ , proposition 5 is now proved.  $\blacklozenge$

We will in the sequel work in the general case: the signature will hence not be supposed reduced. The results above allow us to abbreviate notation  $\leq_i$  (for  $i = 2$  or  $3$ , or even  $1$  if the signature is in reduced form) into  $\leq$ . We will also write  $A \leq_5 B$  when  $A \leq B$  and  $B \leq A$ .

**Proposition 6** *Let  $T$  be a set of terms, let  $M$  be the subset of minimal elements in  $T$ . Then  $M \leq_5 T$ .*

**Proof** As  $M \leq_1 T$ , obviously  $T \leq_4 M$ . Conversely, relation  $\leq$  being noetherian (see [16] and [20]), for every  $s$  in  $T$  there exists some minimal element  $m$  in  $M$  such that  $m \leq s$ , so  $M \leq_0 T$  and then  $M \leq_4 T$ .  $\blacklozenge$

Such a subset  $M$  has the property of being an *antichain*: a set of two by two incomparable elements (by  $\leq$ ). Proposition 6 only confirms the preliminary remark under which one can work on antichains only, which is what we will do in the sequel.

It can be noticed that the first definition is the most natural one, the second one only works due to a correct closed world assumption. The third definition is the constructive definition as will be clear after the next properties, and the algorithm they yield.

**Proposition 7**

- a)  $A \leq_4 A' \Leftrightarrow \forall t \in A', A \leq_4 t$ .
- b)  $A \leq_4 t \Rightarrow \exists a \in A, a \leq t$ .
- c)  $A \leq_4 t \Leftrightarrow \{a \in A / a \leq t\} \leq_4 t$ .
- d)  $A \leq_4 t \Leftrightarrow A \leq_4 \text{Sub}_\Omega(t)$ .
- e)  $\exists a \in A, a \leq t \Rightarrow A \leq_4 t$ .

All 5 properties are direct consequences of definition 16 and propositions 4, 5 & 6.

These properties mean that the question  $A \leq B?$ , for a given signature  $\Sigma$  and two finite<sup>9</sup> antichains  $A$  and  $B$  is decidable. It should be noticed that the question is not trivial as the sets of terms used in definition 14 and 15 can be of infinite size. We shall prove that the following algorithm computes the answer.

### Algorithm 1

```
Function generalises_n ( $A, B$ : antichain): Boolean;
Variables: test: Boolean,  $b$ : term;
begin
test := true;
For all  $b$  in  $B$  do if not generalises( $A, b$ ) then test := false;
return(test)
end;
```

```
Function generalises( $A$ : antichain;  $b$ : term): Boolean;
Variables  $A'$ : antichain;  $a$ : term;
begin
 $A' := \{a \in A \mid a \nabla b\}$ ;
if  $A' = \emptyset$  then return (false)
else if  $\exists a \in A', a \leq b$ 
then return (true)
else return (generalises_n( $A', \text{Sub}_\Omega(b)$ ))
end;
```

**Proof** Validity is due to proposition 7.

Termination can be proved by the following argument:

right-hand side terms increase in size due to the computation of  $\text{Sub}_\Omega(b)$ . This size can be bounded by the size of the largest term in  $A$ : lemma 2 proves that at some point we will have, by lemma 2, either  $a \leq b$  or  $a \nabla b$ .  $\blacklozenge$

We give a final result which shows that this formalism includes our previous normal forms:

**Proposition 8**  $t \leq t' \Leftrightarrow N(t) = N(t')$ .

**Sketch of proof** First  $N(t) \leq t \Rightarrow N(t) \leq t$ . But also it can be proved by induction on  $\text{depth}(u)$  that  $t \nabla u \Rightarrow N(t) \nabla u$ . So  $t \leq N(t)$ .

Thus we have  $t \leq N(t)$ , and hence the result.  $\blacklozenge$

An obvious question now arises: as we have a normal form for terms does there exist some canonical form for antichains? The purpose of the canonical form we propose is:

1- it is unique

2- Call  $N$  the canonical form of  $A$ , then for a set  $B$ :  $A \leq B \Leftrightarrow N \leq B$

It yields that the one-to-one generalisation is correct for the normal form. The algorithmic consequences will come from the fact that one-to-one generalisation is far easier to manipulate than the other set generalisations.

---

<sup>9</sup> It is reasonable, from a data representation point of view, to deal with finite antichains only. Nevertheless dealing with finitely described antichains yields many interesting open questions.

**Definition 17** An antichain  $A$  is  $\leq$ -minimal  $\stackrel{\text{def}}{=} \forall t \in \mathbf{T}_{\Sigma, \Omega}^s, A \not\leq t \Leftrightarrow A \not\leq_0 t$ .

The point of this property is that if  $A$  is  $\leq$ -minimal then  $A \leq_0 B$  can be checked in polynomial time by testing for each  $t$  in  $B$  if  $a \leq t$  for some  $a$  in  $A$ .

**Example 7** With  $S = \{0, 1\}$ ,  $F = \{a, b, f\}$  and  $f: 1 \times 1 \rightarrow 0$ ,  $a, b: 1$  the following sets are equivalent (for 45):

$$A = \{f(a, a, a), f(a, a, b), f(b, a, a), f(b, b, a)\}$$

$$B = \{f(a, a, \Omega), f(b, \Omega, a)\}$$

$$C = \{f(a, a, \Omega), f(b, \Omega, a), f(\Omega, a, a)\}$$

$C$  is the only  $\leq$ -minimal one. It is easy to check that the smallest equivalent set  $D$  such that  $D \leq_0 C$  is  $C$  itself.  $\blacklozenge$

**Theorem 2** For every antichain  $A$  there exists a unique antichain  $B$  such that:

1)  $A \leq_0 B$

2)  $B$  is  $\leq$ -minimal.

**Proof Existence**

Let  $W$  be the set of all  $t$  such that  $A \leq t$ . Take  $B$  as the set of all minimal elements of  $W$ . Then we obviously have

1)  $A \leq_0 B$  since  $B \subseteq W$  and  $A \leq W$ .

1')  $B \leq_0 A$  since  $\forall a \in A, a \in W$ .

2)  $B$  is  $\leq$ -minimal by construction of  $B$ .

*Unicity*

Take  $B$  and  $B'$  two such antichains. Then  $B \leq_0 B'$ . Since  $B$  is  $\leq$ -minimal,  $\forall b' \in B, \exists b \in B, b \leq b'$ . But since  $B'$  is also  $\leq$ -minimal,  $\exists b'' \in B, b'' \leq b'$ . But as  $B'$  is an antichain  $b \leq b'' \Rightarrow b = b'' = b'$ . Hence all elements in  $B'$  are in  $B$ . For symmetrical reasons all elements in  $B$  are in  $B'$ . Hence  $B = B'$ .  $\blacklozenge$

Theorem 2 states the existence for every antichain (hence set) of terms  $A$ , of a sole equivalent  $\leq$ -minimal antichain  $B$ . This antichain will be denoted  $\text{infmax}(A)$ .

We can now define a negation relation on terms and sets of terms:

**Definition 18** Let  $A$  be a set of terms of type  $s$ :

$\sim A$  is  $\text{infmax}(B)$  where  $B = \{t \in \mathbf{T}_{\Sigma, \Omega}^s / \forall a \in A, t \not\leq a\}$ .

Obviously the set of terms incompatible with all terms in  $A$  can be very large. But by proposition 4 there exists some antichain equivalent to this set. Taking the  $\text{infmax}$  is then natural. The fact that this gives us a negation will be clear after the next proposition:

**Proposition 9**

1)  $\sim \sim A = \text{infmax}(A)$

2)  $\{t \in \mathbf{T}_{\Sigma, \Omega}^s / A \leq t\} \cap (\{t \in \mathbf{T}_{\Sigma, \Omega}^s / \sim A \leq t\}) = \emptyset$

3)  $A \not\leq \sim A \leq \Omega$ .

**Proof** We first prove point 1)

All we have to prove is that  $A \leq_0 \sim \sim A$ . By construction we have that  $\sim \sim A$  is  $\leq$ -minimal, hence that  $\sim \sim A = \text{infmax}(\sim \sim A)$  and if the former is proved also  $\sim \sim A = \text{infmax}(A)$ .

All elements in  $A$  are incompatible with all those in  $\sim A$ . Therefore by definition 18, we have  $\sim \sim A \leq_0 A$ .

Conversely, by definition 15 of relation 4, as all terms incompatible with  $A$  are by construction incompatible with  $\sim\sim A$ ,  $A \leq \sim\sim A$  holds.

Suppose now point 2) is false: there exists some term  $t$  such that  $\exists a \in A, a \leq t$  and  $\exists a' \in \sim A, a' \leq t$ . But since  $a$  and  $a'$  are necessarily incompatible, they cannot have a common specialisation.

Point 3) also holds by definition 15 of relation 4: since no term can be incompatible with all terms in  $A$  and  $\sim A$ , we have  $\forall t \in \mathbf{T}_{\Sigma, \Omega}^s, A \leq \sim A \leq t$  hence  $\text{infmax}(A \cup \sim A) = \Omega$   $\blacklozenge$

Our feeling is that relation 4 can be seen as very general: as will be seen in the next section when signatures are used to represent Boolean terms, how 4 extends the implication of propositional logic: indeed a term can be seen as a property written in some conjunctive form, and the disjunction of properties will here be the set of terms. Thus finding the term counterpart of implication is an obvious question, as with this we have a rich knowledge representation system [5]: implication, negation and disjunction are now defined.

This motivates a closer algorithmic study of relation 4. The previous algorithm 1 is clearly exponential (complete substitution, so construction of  $\text{Sub}_{\Omega}(t)$  is itself exponential). We shall therefore in the next section study, in the general case and in easier subcases, the complexity of some problems related with the generalisation relation.

## 5. Algorithmic aspects

An obvious goal in data representation is to manipulate this data. Thus algorithms are to be written, their complexity to be studied. In the case of term-to-term generalisation (as defined in section 3) the problems are easily and quickly solved: checking whether a term generalises another, computing the normal form of a term, verifying compatibility of 2 terms, calculating their most general specialisation or their most specific generalisation are all problems that can be solved by some algorithm running in time linear in the size of the terms involved [12]. But what of the set-to-set relation 4? What is the complexity of the problem  $A \leq B$ ? How expensive is it to compute the negation of an antichain, the canonical form ( $\text{infmax}$ ) of an antichain? We will in this last section study the complexity of five natural problems, first for arbitrary signatures and antichains, and then for special cases of antichains.

In the same way as for terms the elementary problems were checking whether  $t_1 \leq t_2$  or computing the normal form of a term, the pertinent problems for term generalisation also concern the operation of generalisation itself, the computation of the normal form, and the negation problem. Practically we propose the following five base problems:

- $P_1$ : Instance: 2 antichains  $A$  and  $B$ , a signature  $\Sigma = (S, F, \alpha, \sigma)$ .  
Answer: "Yes" if  $A \leq B$ .
- $P_2$ : Instance: 2 antichains  $A$  and  $B$ , a signature  $\Sigma = (S, F, \alpha, \sigma)$ .  
Answer: "Yes" if  $A \leq \text{infmax}(B)$ .
- $P_3$ : Instance: an antichain  $A$ , a signature  $\Sigma = (S, F, \alpha, \sigma)$ .  
Answer: the antichain  $\sim A$ .
- $P_4$ : Instance: 2 antichains  $A$  and  $B$ , a signature  $\Sigma = (S, F, \alpha, \sigma)$ .  
Answer: "Yes" if  $B = \text{infmax}(A)$ .
- $P_5$ : Instance: an antichain  $A$ , a signature  $\Sigma = (S, F, \alpha, \sigma)$ .  
Answer: the antichain  $\text{infmax}(A)$ .

Clearly the problems are related: solving  $P_1$ , one gets the solution to  $P_2$  by definition of 4.5. If  $P_3$  can be solved, then by proposition 6, one obtains by double negation an answer to

$P_5$  (hence also  $P_4$ ). And if one can solve  $P_5$ , then an answer to problems  $P_1$  and  $P_2$  is given by theorem 2.

Before entering the discussion of how these problems are to be solved, we must explain how the instances are to be measured: for an antichain  $A$ , the size of this antichain, *i.e.* the number of terms is obviously relevant: call it  $n_A$ . But the size of the terms (the number of symbols occurring in it) has also got to be reckoned with; it will be sufficient here to take the maximum, as we will not be dealing with average complexity. So for an antichain  $A$  define:

$$m_A = \max\{size(t) / t \in A\} \text{ where}$$

$$size(\Omega) = 0$$

$$size(f) = 1$$

$$size(f(t_1, \dots, t_n)) = 1 + \sum_{i=1}^n size(t_i).$$

But the signature is not necessarily completely given with the terms in an antichain. Yet for the different problems  $P_1$  to  $P_5$  it will be required to answer to questions such that: "is term  $t$ , that can also be obtained by the signature, generalised?". Hence the signature is part of the instances of a problem. The size of a signature depends essentially on the coding of the arity function. We thus define the weight  ${}_i\Sigma_i$  of the signature  $\Sigma = (S, F, \alpha, \sigma)$  by

$${}_i\Sigma_i = {}_iF_i + \sum_{f \in F} {}_i\alpha(f)_i.$$

We first study the general case where antichains  $A$  and  $B$  are given on arbitrary signatures: this general case will lead to propositions 10, 11 and 12.

**Proposition 10**  $P_1$  ( $A \leq B$ ?) and  $P_2$  ( $A \leq_5 B$ ?) are co-NP-complete.

**Proof** We will prove that  $P_1$  is co-NP-complete, which will also yield the proof that  $P_2$  is equally so.

This will be done by transformation from the non-tautology problem [15] which is known to be NP-complete<sup>10</sup>.

The signature we will use is straightforward: for  $U = \{x_1, \dots, x_k\}$ , define  $F = \{f\}$  "  $\{a_i, b_i / \forall i \in [k]\}$  and  $f: 1 \times 2 \times \dots \times k \rightarrow 0; \forall i \in [k] a_i, b_i: i$ .

A Boolean expression  $E$  is a disjunction of Boolean terms, where a boolean term is a conjunction of literals, *i.e.*  $E = \bigvee_{i \in [n]} c_i$

We can now encode each Boolean term in  $E$  by a term:

For a Boolean term  $c$  in  $E$  we create a term  $f(y_1, \dots, y_k)$  where

$$\forall i \in [k], y_i = a \text{ if } x_i \text{ appears in } c$$

$$b \text{ if } \sim x_i \text{ appears in } c$$

$$\Omega \text{ if neither appear in } c.$$

We thus obtain a set of terms  $T$  associated with  $E$  and it is easy to see that  $E$  is a tautology if and only if  $T \leq \Omega$ . The construction of  $T$  is direct (so polynomial), so  $P_1$  is polynomial only if the tautology problem is.

Now given 2 antichains  $A$  and  $B$ , if we are also given a term  $t$  such that  $B \leq t$ , but  $t$  is incompatible with all terms in  $A$ , this proves that  $A \leq B$  does not hold. All this can be checked in polynomial time, hence the co-NP-completeness of our problem. ♦

**Proposition 11** For  $P_3$  (compute  $\sim A$ ), the size of  $\sim A$  can be exponential in the number of terms in  $A$ .

<sup>10</sup> Instance: A Boolean expression  $E$  over a set  $U$  of variables, in disjunctive normal form.  
Question: is  $E$  not a tautology? , *i.e.* is there a truth assignment for  $U$  that makes  $E$  false?

**Proof** We first sketch the proof that  $\sim A$  is finite. This is not the case for arbitrary first-order terms with variables [8, 17]. Note  $k$  the largest depth of terms in  $A$ . Then suppose there exists some term  $t$  with  $\text{depth}(t) > k + 1$ . There must exist a symbol different from  $\Omega$  in  $t$  at maximal distance from the root. But it is easy to see that one can substitute this symbol by  $\Omega$  still getting a term incompatible with all terms in  $A$ . Hence  $t$  does not belong to  $\sim A$ . And the number of terms of depth less or equal to  $k+1$  is finite.

Now to prove that the computation can take exponential time (and size), take the following signature :

$$F = \{f, a, b\} \quad f: 1^{2k} \rightarrow 0; \quad a, b: 1;$$

$$\text{Let } A = \{f(x_1, \dots, x_{2k}) / \exists i \in [k], x_{2i} = a, x_{2i-1} = a \text{ and } \forall j \neq i, x_{2j} = b, x_{2j-1} = b\}$$

Thus  $\text{wt}(A) = k$ . Now let  $B = \{f(x_1, \dots, x_{2k}) / \forall i \in [k], (x_{2i} = b, x_{2i-1} = \Omega) \text{ or } (x_{2i} = \Omega, x_{2i-1} = b)\}$ . One can check that  $B \not\leq A$ , and this results in:  $\text{wt}(\sim A) \geq 2^k$ .  $\blacklozenge$

**Proposition 12**  $P_4$  is co-NP-complete and  $P_5$  is NP-hard.

**Proof**  $P_4$  is at least as difficult as  $P_1$ : the same construction as in the proof of proposition 10 gives that  $E$  is a tautology if and only if  $\Omega = \text{infmax}(T)$  where  $T$  is the set of terms constructed on expression  $E$ . Obviously  $P_4$  is not NP-complete, as it would lead to  $P_1$  being NP-complete, hence "not a tautology" also. Co-NP-completeness of  $P_4$  results from the following argument:

$B \neq \text{infmax}(A)$  if and only if one of the following holds:

1)  $B \not\leq A$

2)  $B$  is not  $\leq$ -minimal.

The first point can be checked in polynomial time when a term  $t$  is given such that either  $\exists a \in A, a \leq t$  but  $\forall b \in B, b \not\leq t$  or conversely.

The second point requires also a given term  $t$ . Let  $G$  be the set of all terms such that  $B \not\leq t$  but there is no  $b$  in  $B$  such that  $b \leq t$ . Let  $g$  be a maximal element of  $G$  (such an element exists because of the third definition of 4). It is now possible to prove in polynomial time that  $B$  is not  $\leq$ -minimal by taking each immediate specialisation  $s$  of  $g$  and checking that there exists some  $b$  in  $B$  with  $b \leq s$ . Hence  $B$  is not  $\leq$ -minimal.

An immediate corollary is that  $P_5$  is also NP-hard. The question of whether  $P_5$  is NP-equivalent remains open.  $\blacklozenge$

These results could allow us to be pessimistic as to using the model on practical cases (for instance in Machine Learning). Nevertheless it should be noticed that we would obtain the same results with flat signatures, *i.e.* those that correspond to descriptions with Boolean variables, so the model although richer (see [12]) is not algorithmically worse on these problems. It can also be argued that the same five problems do not have to be considered for arbitrary signatures or antichains. This leads us to present some conditional results: a first result concerns the negation of a single term:

**Proposition 13**  $\sim t$  can be computed in  $O(m_t * \sum \text{wt}_i)$  time, where  $m_t$  is the number of symbols in  $t$  and  $\sum \text{wt}_i$  the weight of the signature.

To prove this proposition we will need the next additional lemma:

**Lemma 3**  $t' \in \sim t \Leftrightarrow$

either  $t'$  is elementary and incompatible with  $t$

or  $t = f(t_1, \dots, t_n), \quad t' = f(t'_1, \dots, t'_n)$  and  $\exists i \in [n], t'_i \in \sim t_i$  and  $\forall j \neq i, t'_j = \Omega$ .

**Proof** By induction on the depth of  $t$ :

If  $depth(t) = 0$ ,  $t = \Omega \Rightarrow \sim t = \emptyset$ .  
 $t' = f \Rightarrow [t' \in \sim t \Rightarrow t' \text{ is elementary.}]$

Suppose the lemma true for all terms of depth at most  $k$ , let  $depth(t) = k + 1$ ,  $t = f(t_1, \dots, t_n)$  and  $t' \in \sim t$ . Then either  $t'$  is elementary and incompatible with  $t$  or  $t' = f(t'_1, \dots, t'_n)$  and  $f = f'$  (if not we are in the case above). Since  $t \nabla t'$  all  $t'_i$  cannot be  $\Omega$ . But if at least two of them are different from  $\Omega$ , say  $t'_i$  and  $t'_j$ , we have by incompatibility:  $t'' = f(t''_1, \dots, t''_n) \nabla t$  with  $\forall l \in [n], l \neq j \Rightarrow t''_l = t'_l, t''_j = \Omega$ . But  $t'' \leq t'$  so  $t' \in \sim t$ . So  $\exists i \in [n], t'_i \nabla t_i$ , and  $\forall j \neq i, t'_j = \Omega$ . For the same reasons  $t'_i$  is minimal such that  $t'_i \nabla t_i$  so  $t'_i \in \sim t_i$ . And by the induction hypothesis ( $depth(t_i) \leq k$ ),  $t'_i$  respects lemma 3, and so does  $t'$ . ♦

**Corollary 3** If  $t' \in \sim t$  then  $t'$  can be represented by a tree that has non  $\Omega$  symbols on only one branch.

**Proof of proposition 13** There are as many terms in  $\sim t$  as ways of taking any alternative symbol of same sort as any symbol in  $t$ . So the number of terms in  $\sim t$  is bounded by  $\max\{i F^s i, s \in S\} * size(t)$ , and lemma 3 yields directly a polynomial algorithm that checks whether  $t' \in \sim t$ . This ends the proof of proposition 13. ♦

A second way of simplifying the problems is by allowing only specific sorts of terms in the antichains. It is for instance a classical assumption in Machine Learning that in attribute/value representation, all attributes must have a value for a given example or object [5]. This in the case of term representation would mean that for a term to correspond to the description of an example (as opposed to a concept), it should be completely specified, *i.e.* the term would not contain symbol  $\Omega$ . An antichain containing only such maximal terms will itself be called a *maximal term antichain*.

**Proposition 14** The following problems can be computed in polynomial time:

$P_1$ :  $A \leq B$ ?

when  $A$  is a maximal term antichain  
or when  $B$  is a maximal term antichain.

$P_2$ :  $A \leq B$ ?

when  $A$  is a maximal term antichain  
or when  $B$  is a maximal term antichain.

$P_4$ :  $B = \text{infmax}(A)$ ?

when  $A$  is a maximal term antichain  
or when  $B$  is a maximal term antichain.

$P_5$ : Compute  $\text{infmax}(A)$

when  $A$  is a maximal term antichain.

**Proof** Let  $B$  contain only maximal terms. Then  $A \leq B$  if and only if  $\forall b \in B, \exists a \in A, a \leq b$ . This can be checked in polynomial time.

Conversely let  $A$  contain only maximal terms. Then algorithm 1 gives us an answer to  $A \leq B$  in polynomial time: it requires testing  $A \leq b$  for each  $b$  in  $B$ , and indeed each time a  $\Omega$  in  $b$  is substituted by an elementary term the number of terms compatible with the result of the substitution diminishes. So the total number of times function "induces" is called for each  $b$  is bounded by the size of  $A$ .

Since  $P_1$  is polynomially tractable in both cases, it follows that  $P_2$  ( $A \leq B$ ?) is also so each time one of the 2 antichains is composed only with maximal terms.

We will now prove directly that  $P_5$  is polynomially tractable: it then follows as a corollary that  $P_4$  is also so.

All we have to prove is that  $t \in \text{infmax}(A) \Rightarrow \exists t_1, t_2 \in A, t = N(t_1 \wedge t_2)$  where  $N$  is the normal form function given in definition 12. This holds through the following argument: if  $t$  belongs to  $\text{infmax}(A)$ , then all maximal terms  $t'$  such that  $t \leq t'$  belong to  $A$ . Now to each occurrence of  $\Omega_i$  in  $t$  we can associate the set  $\text{Inst}(\Omega_i)$ , of all maximal terms of sort  $i$ . There are 4 cases:

$\text{Inst}(\Omega_i)$  is infinite: then  $t$  cannot belong to  $\text{infmax}(A)$ .

$\text{Inst}(\Omega_i) = \emptyset$ : impossible since  $t \in \text{infmax}(A) \Rightarrow \exists t' \in A, t \leq t'$ .

$\text{Inst}(\Omega_i) = \{x\}$ : then take  $a_i = b_i = x$

If not there are at least 2 different elements in  $\text{Inst}(\Omega_i)$ :

Choose them such that  $N(a_i \wedge b_i) = \Omega_i$ . (This is always possible)

We can now construct 2 maximal terms  $t_1$  and  $t_2$  by substituting in  $t$  each  $\Omega_i$ , for each sort  $i$  by a term  $a_i$  giving  $t_1$  (respectively each  $\Omega_i$ , for each sort  $i$  by term  $b_i$  giving  $t_2$ ). And since  $t \leq t_1, t \leq t_2$  we have  $t_1, t_2 \in A$ . And by the normal form property, we have  $t = N(t_1 \wedge t_2)$ .

A direct but not optimised algorithm giving us  $\text{infmax}(A)$  is to take all couples  $(t_1, t_2)$  in  $A$ , compute  $t = N(t_1 \wedge t_2)$ , check by algorithm 1 if  $t \in A$ , and if the answer is yes keep  $t$ . From the set of terms thus obtained, keep the minimal elements: these give us  $\text{infmax}(A)$ .

All these operations are polynomial and since there is only a polynomial number of pairs  $(t_1, t_2)$  in  $A$ , it insures us that  $P_5$  is polynomially tractable. Hence so is  $P_4$ .  $\blacklozenge$

The proof of proposition 11 uses maximal terms, so the size of  $\sim A$  can be exponential in the size  $n_A$  of  $A$ , even in this restricted case.

## 6. Conclusion

Relation 4 we have defined and studied in this paper is a natural extension of implication in propositional calculus. Its capacity to deal with structured knowledge representation makes terms and relations between terms (or sets of terms) worthy candidates to extend results, techniques and of course programs dealing with attribute-value representations, logical representations and also grammatical ones (see [5], [17]). This work has already started: for instance clustering term-described information through Galois lattices has been done in [12]. Other problems from the fields of Machine Learning or Data Analysis should also be transplanted to structured data: learning from examples, rule inference, discrimination problems ([5], [26], [32]). In all cases a generalisation relation is required [28]: the one proposed in this paper should be a good candidate. The complexity of the different problems studied in part 5 stresses nevertheless the fact that even obvious problems can be algorithmically difficult. A final word concerns the treatment of negation: the one we propose in this paper relies on the closed world assumption. Nevertheless it is probably here that the main advantage of using "linear" terms appears, as one can compute in finite (but possibly exponential) time the negation not only of a term but of a set of terms also, which is not the case in classic first-order terms.

## Acknowledgement

We are grateful to Olivier Gascuel for many discussions and useful comments on earlier drafts of this paper. We would also like to thank an anonymous referee for many improving remarks.

## References

- [1] H. Aït-Kaci, "An algebraic semantics approach to the effective resolution of type equations". *Theoretical Computer Science*, vol. **45**, pp 293-351, 1986.
- [2] J-M. Autebert, "Langages algébriques". *Masson ed.*, Paris, 1987.
- [3] R.M. Amadio, L. Cardelli, "Subtyping recursive types". In *proceedings POPL'91*, appeared in *ACM*, pp 104-118, 1991.
- [4] H.J. Bandelt, J.P. Barthelemy, "Medians in median graphs". *Discrete Applied Mathematics* **8**, pp 131-142, 1984.
- [5] A. Barr, E.A. Feigenbaum, "The Handbook of Artificial Intelligence". Vol.1, *W. Kaufmann ed.*, California 1981.
- [6] G. Birkhoff, "Lattice Theory". *Colloquium Publications*, vol. XXV, AMS, Providence, 3rd edition, 1967.
- [7] R.J. Brachman, J. Schmolze. "An overview of the KL-ONE Knowledge Representation system. *Cognitive Science* **9** (2), pp171-216, 1985.
- [8] M. Chein, M-L. Mugnier, "Conceptual graphs: Fundamental notions". *Revue d'Intelligence Artificielle*, vol. **6**, n°4, pp 365-406, 1992.
- [9] H. Comon, "Disunification: a survey". In J.-L. Lassez and G. Plotkin, editors, *Computational Logic: Essays in Honor of Alan Robinson*, MIT Press, pp 322-359, 1991.
- [10] B. Courcelle, "Equivalence and transformations of regular systems. Applications to recursive program schemes and grammars". *Theoretical Computer Science*, Vol. **42.**, pp 1-122, 1986.
- [11] M.C. Daniel-Vatonne, "Les termes : un modèle de représentation et de structuration de données symboliques". Ph.D. Thesis, Université de Montpellier 2, 1993.
- [12] M.C. Daniel-Vatonne, C. de la Higuera, "Les termes: un modèle algébrique de représentation et structuration de données symboliques". *Mathématiques, Informatique et Sciences Humaines*, **122**, 1993.
- [13] F. M. Donini, M. Lenzerini, D. Nardi, W. Nutt, "The complexity of concept languages". In *Proc. of the sec. int. Conf. on Knowledge Representation and Reasoning*, pp 151-162.
- [14] J.H. Gallier, "Logic for computer science: foundations of automatic theorem proving". *Harper & Row Publishers*, New York, 1986.
- [15] M.R. Garey, D.S. Johnson, "Computers and intractability: guide to the theory of NP-completeness". *Victor Klee Ed.*, *W.H.Freeman&co.*, San Francisco, 1979.
- [16] I. Guessarian, "Algebraic Semantics". *LNCS 99*, Springer-Verlag ed., Berlin, 1981.
- [17] D. Haussler, "Quantifying Inductive Bias: AI Learning Algorithms and Valiant's Learning Framework". *Artificial Intelligence* **36**, pp177-221, 1988.
- [18] H.B. Hollunder, W. Nutt, M. Schmidt-Schauss, "Subsumption algorithms for concept description languages". In *Proc. 9th ECAI*, 1990.
- [19] J-U.Kietz, K.Morik, "A Polynomial Approach to the Constructive Induction of Structural Knowledge". *Machine Learning* **14**, pp 193-217, 1994.
- [20] R. Lalement, "Logique, Réduction, Résolution". *Masson ed.*, Paris, 1990.
- [21] J-L. Lassez, M.J. Maher & K. Marriott, "Unification revisited". *LNCS 306*, Springer-Verlag ed., Berlin, pp 67-113, 1986.
- [22] J-L. Lassez, K. Marriott, "Explicit representation of terms defined by counter examples". *Journal of Automated Reasoning* **3**, pp 301-317, 1987.
- [23] M. Liquière, J. Sallantin, "INNE: induction in networks". *IEEE International Workshop on Tools for AI*, Fairfax USA, 1989.
- [24] J.W. Lloyd, "Foundations of Logic Programming". *Springer-Verlag*, second edition, 1987.
- [25] R.S. Michalski, "A theory and Methodology of Inductive Learning". *Artificial Intelligence* **20**, pp 111-161, 1983.

- [26] T. Mitchell, "Version spaces: a candidate elimination approach to rule learning". In *proceedings of I.J.C.A.I.*, vol. **5**, pp 305-310, 1977.
- [27] B. Nebel, "Reasoning and revision in Hybrid Representation systems". *Lecture Notes in Artificial Intelligence* **422**, Springer-Verlag, ISBN 3-540-52443-6, 1987.
- [28] G.D. Plotkin, "A further note on Inductive Generalization". *Machine Intelligence* **6**, pp 101-124, 1971.
- [29] J.F. Sowa, "Conceptual structures: Information Processing in Mind and Machine". *Addison-Wesley pub.*, Reading (USA), 1984.
- [30] J.D. Uhlman, "Principles of Database and Knowledge-base systems". *Principles of Computer Science Series*, Computer Science Press, 1988.
- [31] S.A. Vere, "Induction of concepts in the predicate calculus". In *proceedings of IJCAI-4*, pp 281-287, 1975.
- [32] R. Wille, "Restructuring Lattice Theory: an approach based on hierarchies of concepts". In *Ordered Sets*, I.Rival Ed., Reidel, Dordrecht-Boston, pp 445-470, 1982.