A Computationally Tractable Terminological Logic*

Fabrizio Sebastiani & Umberto Straccia†
Istituto di Elaborazione dell'Informazione
Consiglio Nazionale delle Ricerche
Via S. Maria, 46 - 56126 Pisa (Italy)

Abstract

Terminological Logics are knowledge representation formalisms of enormous applicative interest, as they are specifically oriented to the vast class of application domains that are describable by means of taxonomic organizations of complex objects. A number of results relative to the computational complexity of terminological logics have recently appeared in the literature. Unfortunately, most of these results are "negative" in nature, as they show that, in the logics they refer to, deciding *subsumption* (i.e. the metalinguistic relation which corresponds to validity in standard logics) is computationally intractable. In this paper, after briefly introducing the fundamental concepts underlying terminological logics, we show that computing subsumption is $O(n \log n)$ in the $\mathcal{ALN}$ logic, an extension of Brachman and Levesque's $\mathcal{FL}^-$ logic. $\mathcal{ALN}$ is obtained by endowing $\mathcal{FL}^-$ with the two operators **atleast** and **atmost**, which allow the specification of number restrictions, and the operator **a-not**, which introduces a limited form of negation. The result we present is of theoretical significance, in that

*This paper appears in the *Proceedings of SCAI-91, 3rd Scandinavian Conference on Artificial Intelligence*, Roskilde, Denmark, 1991, pp. 307–315. Edited by Brian Mayoh. Amsterdam, The Netherlands: IOS Press.

†This research has been partially funded by the Progetto Finalizzato "Sistemi Informatici e Calcolo Parallelo".

$\mathcal{ALN}$ is one of the few terminological logics that have been shown tractable. $\mathcal{ALN}$ is also a pragmatically significant extension of $\mathcal{FL}^-$, as it results from the addition of operators of considerable applicative interest.

1 Introduction

Terminological Logics (TLs, also known as *Frame Representation Languages* or *Concept Description Languages*) are knowledge representation formalisms of enormous applicative interest, as they are specifically oriented to the vast class of application domains that are describable by means of taxonomic organizations of complex objects. Such logics have a non formalized counterpart in a large class of early knowledge representation languages based on semantic networks and inspired by the notion of *frame*, originally introduced by Minsky [9]. Unfortunately these languages, of which KL-ONE [2] may be considered the founding father, did not possess a formal semantics; hence, the fact that the notion of inference they enforced were the same as the user expected relied on an improbable like-mindedness among designer, implementor and user. TLs may then be seen as the result of a work of distillation and formalization that was carried out on KL-ONE-like languages, a work that resulted in linguistic constructs of dubious semantic content being purged while the others were brought back within the canons of mathematical logic.

TLs are especially popular among knowledge engineers because of the vastness of their possible applications (in fact, many are the application domains whose formalization may be accomplished, at least partially, by means of a taxonomic structure), and because of their easy and intuitive syntax. This syntax might be dubbed *object-oriented*, as it encourages the structuring of a knowledge base into "clusters of knowledge" that represent structured descriptions of classes of individuals belonging to the domain of discourse. Borrowing a terminology that is typical of the discipline of data modelling, we might say that the syntax of TLs supports the definition of knowledge by providing mechanisms for *abstraction*, *generalization* and *aggregation*.

Unlike better known logics (such as e.g. first order logic), whose main syntactic constituents are formulae, TLs have *terms* as their primary syntactic expressions. A term is usually an expression that denotes monadic or dyadic

relations on the domain of discourse. For example, in a TL that allows for the term-forming operators **and**, **atleast** and **atmost**[1], the expression (**and** `polygon` (**atleast** `3 sides`) (**atmost** `3 sides`)) is a term that denotes the set of all those polygons that have exactly three sides (i.e. the set of triangles). In general, the languages of TLs allow for a number of term-forming operators by means of which one may build complex terms starting from a basic repertory of simple terms (*viz.* predicate symbols).

As remarked before, TLs are not logics of formulae; hence, the meta-linguistic relation of interest is not validity, but *subsumption* (i.e. "conceptual containment") between terms. A theorem prover for a TL is thus a program that, given two terms, decides whether the former subsumes the latter, i.e. whether all individuals denoted by the latter are also denoted by the former. For example, such a theorem prover will be able to answer affirmatively to the question whether the term (**and** `polygon` (**atmost** `4 sides`)) (which denotes the set made out of all triangles and quadrangles) subsumes (**and** `polygon` (**atleast** `3 sides`) (**atmost** `3 sides`)). A knowledge base will thus be a hierarchy (or *taxonomy*) of terms in the form of a directed acyclic graph, with the hierarchical ordering being determined by the relation of subsumption.

Of course, the applicability of TLs to problems of realistic size largely depends on the expressive power of the logics themselves (i.e. on the possibility of expressing more and more complex terms by means of the linguistic devices made available by the language), and on the possibility of specifying theorem provers that, besides being correct and complete with respect to the logic in question, are able to decide subsumption *efficiently*. A thorough understanding of TLs requires then a rigorous and systematic study of the relationship between the expressive power and the computational complexity of TLs.

A definition that be at the same time rigorous and general of what the expressive power of TLs is[2] would not be very significant in itself, as it could hardly abstract from logics and operators that have never been discussed in the literature but are in principle both possible and significant. In other words, such a definition is made problematic by the fact that any definition of what the space of possible TLs is would probably be arbitrary in nature. However, it is possible to go a first step in this direction, a step that will

[1]The semantics of these operators will be formally specified in Section 2.

[2]For an attempt in this sense, see the work by Baader [1].

be sufficiently significant for our purposes and general enough to encompass much of what is being done nowadays on TLs. In order to do so, let us consider a "reference" set Σ of operators, a set that we will choose sufficiently large (for example, we might choose the set of all the term-forming operators that we have encountered throughout our wanderings in the literature of TLs), and let us define a TL as a subset of Σ. The relationship "has less expressive power than" may now be defined as the relationship of containment between subsets of Σ, and is then a partial order defined on the lattice of the subsets of Σ[3].

It is well known that there is a tradeoff between the expressive power of a logic and its computational tractability; hence, in general, the relation "has less expressive power than" coincides with the relation "is decidable at least as efficiently as". One of the fundamental problems of the research on TLs is that of individuating the TLs which are "optimal" from the standpoint of this tradeoff, that is, individuating those logics which are most expressive among the ones for which a polynomial decision algorithm can be given. A number of results relative to the computational complexity of the decision problem in TLs have recently appeared in the literature. The best known result in this sense is the one due to Brachman and Levesque [3, 8], who have shown that subsumption may be intractable even in TLs that are relatively poor from an expressive point of view. In particular, they have shown that the decision problem is co-NP-hard for $\mathcal{FL} = \{\textbf{and}, \textbf{all}, \textbf{some}, \textbf{restr}\}$, while it is $O(n^2)$ for $\mathcal{FL}^- = \mathcal{FL} \setminus \{\textbf{restr}\}$.

Unfortunately, nearly all results appeared after the one by Brachman and Levesque are "negative" in nature, as they show that, in the logics they refer to, computing subsumption is either undecidable [12, 13, 14] or computationally intractable [7, 10, 11, 15]. The only positive result in this respect is the one due to Donini *et al.* [5], who have recently given a tractability result for the logic {**and**, **a-not**, **r-all**, **c-some**}.

In this work we show that computing subsumption is $O(n \log n)$ in the logic $\mathcal{ALN}$, an extension of Brachman and Levesque's $\mathcal{FL}^-$ obtained by adding the two operators **atleast** and **atmost**, which allow the specification of numerical restrictions, and the operator **a-not**, which allows a limited

[3]For the sake of simplicity, we will assume that none among the operators in Σ may be expressed by means of a combination of other operators of Σ; this will allow us to say that different subsets always have different expressive power.

form of negation. This result also confirms a hypothesis by von Luck and Owsnicki-Klewe [18], who had conjectured that $\mathcal{FL}^{\mathcal{N}-} = \mathcal{ALN} \setminus \{\textbf{a-not}\}$ had polynomial complexity. The result we deal with is of theoretical significance, as $\mathcal{ALN}$ is to date one of the few TLs that have been shown tractable. $\mathcal{ALN}$ is also a pragmatically significant extension of $\mathcal{FL}^{-}$, as it results from the addition of operators that, as argued e.g. in [17], are of considerable applicative interest.

The rest of this paper is organized as follows. In Section 2 we describe the syntax and formal semantics of $\mathcal{ALN}$, and define the notion of subsumption between $\mathcal{ALN}$ terms. In Section 3 we sketch the proof of the tractability of $\mathcal{ALN}$. Section 4 concludes.

2 Syntax and semantics of the $\mathcal{ALN}$ logic

The $\mathcal{ALN}$ logic, like many other TLs, allows the specification of two fundamental types of terms: *concepts* and *roles*. Concepts are terms denoting sets of individuals, and are, so to speak, the first-class citizens of $\mathcal{ALN}$; in fact, it is only on concepts that the subsumption relation is defined. Roles are terms denoting binary relations between individuals; their function is to allow the specification of structural constituents of concepts. For example, the basic repertory of simple terms (called *atoms*) that are used in order to build more complex terms might contain the concept `polygon`, denoting the set of polygons, and the role `side`, denoting all those pairs of individuals $\langle x, y \rangle$ such that y is one of the sides of x; this would allow us to define more complex concepts, such as e.g. (**and** `polygon` (**atleast** `3 sides`) (**atmost** `3 sides`)) discussed in the introduction, and to subsequently define other concepts by using those defined before.

The syntax of $\mathcal{ALN}$ is the following:

$$
\begin{array}{lcl}
\langle \textit{concept} \rangle & ::= & \langle \textit{atom} \rangle \\
 & & \mid (\textbf{top}) \mid (\textbf{bottom}) \\
 & & \mid (\textbf{and}\ \langle \textit{concept} \rangle^{+}) \\
 & & \mid (\textbf{all}\ \langle \textit{role} \rangle\ \langle \textit{concept} \rangle) \\
 & & \mid (\textbf{a-not}\ \langle \textit{atom} \rangle) \\
 & & \mid (\textbf{atleast}\ \langle \textit{positive integer} \rangle\ \langle \textit{role} \rangle) \\
 & & \mid (\textbf{atmost}\ \langle \textit{positive integer} \rangle\ \langle \textit{role} \rangle) \\
\langle \textit{role} \rangle & ::= & \langle \textit{atom} \rangle
\end{array}
$$

Therefore, the language of $\mathcal{ALN}$ allows us to define concepts of a certain complexity, such as e.g.

```
(and  polyhedron
      (atmost 4 face)
      (atleast 4 face)
      (all face  (and  polygon
                       (atleast 3 sides)
                       (atmost 3 sides))))
```

which denotes the set of all those polyhedra which have exactly four triangular faces, i.e. the set of tetrahedra. It is possible to check that the meaning of this term is exactly the one described by computing it according to the formal semantics of $\mathcal{ALN}$, which is detailed in the following definition.

Definition 1
Let $\mathcal{D}$ be a nonempty set of individuals and $\mathcal{E}$ a function from concepts into subsets of $\mathcal{D}$ and from roles into subsets of $\mathcal{D} \times \mathcal{D}$. $\mathcal{E}$ is an **extension function** *over $\mathcal{D}$ if and only if:*

1. $\mathcal{E}[\![(\textbf{top})]\!] = \mathcal{D}$
2. $\mathcal{E}[\![(\textbf{bottom})]\!] = \emptyset$
3. $\mathcal{E}[\![(\textbf{and } F_1 \ldots F_n)\ ]\!] = \bigcap_{i=1}^{n} \mathcal{E}[\![F_i]\!]$
4. $\mathcal{E}[\![(\textbf{all } R\ F)]\!] = \{x \in \mathcal{D} | \forall y \langle x, y\rangle \in \mathcal{E}[\![R]\!] \Rightarrow y \in \mathcal{E}[\![F]\!]\}$
5. $\mathcal{E}[\![(\textbf{a-not } F]\!] = \mathcal{D} \setminus \mathcal{E}[\![F]\!]$
6. $\mathcal{E}[\![(\textbf{atleast } n\ R)]\!] = \{x \in \mathcal{D} |\ \| \{y \in \mathcal{D} |\ \langle x, y\rangle \in \mathcal{E}[\![R]\!]\} \| \geq n\}$
7. $\mathcal{E}[\![(\textbf{atmost } n\ R)]\!] = \{x \in \mathcal{D} |\ \| \{y \in \mathcal{D} |\ \langle x, y\rangle \in \mathcal{E}[\![R]\!]\} \| \leq n\}$

It is easy to check that the **and** operator enjoys the commutative, associative and left distributive (wrt **all**) properties. The notion of subsumption between concepts is then specified by means of the following definition.

Definition 2
Let F_1 and F_2 be two concepts of $\mathcal{ALN}$. F_2 is **subsumed** *by F_1 if and only if for every nonempty set of individuals $\mathcal{D}$ and every extension function $\mathcal{E}$ over $\mathcal{D}$ it is true that $\mathcal{E}[\![F_2]\!] \subseteq \mathcal{E}[\![F_1]\!]$.*

3 The computational tractability of $\mathcal{ALN}$

In this section we will show that there exists an algorithm which decides in time $O(n \log n)$ if a concept F_1 subsumes a concept F_2 in $\mathcal{ALN}$. The algorithm is essentially divided in two parts: the first part reduces a concept into a "normal form", while the second part checks whether there exists a relation of subsumption between two already normalized concepts. The algorithm is structurally similar to the subsumption algorithm given for $\mathcal{FL}^-$ in [8], but is a non-trivial extension of it, as non-trivial is the proof of its completeness wrt $\mathcal{ALN}$.

The proofs of correctness and completeness that will be described below are given by induction on the structural complexity of the concepts taken into consideration. A precise specification of what we mean by structural complexity is given by the following definition.

Definition 3
The **depth** *(or* **structural complexity***) of a concept is given by the maximal number of nestings of the* **all** *operator.* ∎

The choice of the **all** operator is due to its being the only concept-forming operator employing roles, and hence the only operator that shapes concepts in a tree-like fashion.

We now switch to discussing the normalization phase. In this phase we will use a new operator **card**, which is not present in $\mathcal{ALN}$; (**card** $\langle n_1, n_2 \rangle R$) denotes the set of those individuals that have at least n_1 and at most n_2 R. The **card** operator is then a "macro" made out of the **and**, **atleast** and **atmost** operators.

Definition 4
A concept F is in **normal form** *if and only if $F \equiv$(***bottom***), or $F \equiv$(***top***), or $F =$(***and*** $F_1 \ldots F_n$) with every F_i a factor; a concept F is a* **factor** *if and only if it is an atom, or $F =$(***a-not*** a) with a an atom, or $F =$(***all*** R F') with F' a concept in normal form different from* (**bottom**) *and* (**top**)*, or $F =$(***card*** $\langle n_1, n_2 \rangle R$) with $n_1 \leq n_2$ and such that if $n_2 = \infty$, then $n_1 > 0$. In addition, the factors $F_1, \ldots, F_n$ of a concept in normal form must be such that:*

- *if $i \neq j$ and F_i, F_j are atoms, then $F_i \neq F_j$;*

- *if* F_i=(**a-not** a) *and* F_j=(**a-not** a') *and* $i \neq j$, *then* $a \neq a'$;
- *if* F_i *is an atom, then there does not exist* j *such that* F_j=(**a-not** F_i);
- *if* F_i=(**all** R F) *and* F_j=(**all** R' F') *and* $i \neq j$, *then* $R' \neq R$;
- *if* F_i=(**card** I R) *and* F_j=(**card** I' R') *and* $i \neq j$, *then* $R' \neq R$;
- *if* F_i=(**all** R F), *then there does not exist* j *such that* a_j=(**card** $\langle 0,0 \rangle$ R). ∎

In the full paper we describe a normalization algorithm NORM, and we prove the following lemma that describes its properties.

Lemma 1 *Normalization*
Let F *be a concept of* $\mathcal{ALN}$. *The* NORM*348–353 algorithm reduces, in time* $O(|F| \log |F|)$, *the concept* F *in a normal form* F' *such that* $\mathcal{E}[\![F]\!] = \mathcal{E}[\![F']\!]$ *for any extension function* $\mathcal{E}$, *and such that* $|F'|$ *is* $O(|F|)$. ∎

Normalization structures a concept in a tree-like form, with **all** operators as internal nodes and all other constituents as leaves ordered from left to right according to the lexicographical order. Normalization preserves then both the extension and (modulo a multiplicative constant) the length of the concept being normalized.

We are now ready to describe SUBS, the subsumption algorithm. SUBS takes as input two concepts in normal form and decides whether a relation of subsumption exists between them.

Algorithm 1 *SUBS(F_1,F_2)*

Input: two concepts F_1, F_2 in normal form;

Output: *true* if F_2 is subsumed by F_1; *false* otherwise;

Body:

1. if $F_1 \equiv$ (**top**) or $F_2 \equiv$ (**bottom**), return *true*; otherwise:
2. if $F_1 \equiv$ (**bottom**) or $F_2 \equiv$ (**top**), return *false*; otherwise, let F_1 =(**and** $a_1 \ldots a_n$) and F_2 =(**and** $b_1 \ldots b_m$); return *true* if and only if, for all $1 \leq i \leq n$, the following conditions are satisfied:

3. if a_i is an atom or a_i =(**a-not** a), then one among the b_j's is a_i;
4. if a_i =(**all** R x), then one among the b_j's has the form (**all** R y) and SUBS(x,y)=*true*, or one among the b_j's has the form (**card** $\langle 0,0\rangle R$);
5. if a_i =(**card** $\langle n_1,n_2\rangle R$), then one among the b_j's has the form (**card** $\langle n_3,n_4\rangle R$), with $n_1 \leq n_3$ and $n_4 \leq n_2$. ∎

It should be noted that, by definition of normal form, steps 1 and 2 are taken at most once (at the first call of SUBS). In the full paper we prove the following lemma.

Lemma 2 *Complexity*
The SUBS(F_1, F_2) *algorithm runs in time* $O(|F_1|\log|F_2|)$. ∎

Given the $O(n\log n)$ result, with $n = max(|F_1|,|F_2|)$, for the normalization of concepts F_1 and F_2, we may conclude that the complexity of computing subsumption between two not yet normalized concepts is also $O(n\log n)$.

The following lemma is easily proven by induction on the maximal depth of the two concepts involved.

Lemma 3 *Correctness*
If SUBS(F_1, F_2)=*true, then* F_2 *is subsumed by* F_1. ∎

Let us now switch to the completeness proof for SUBS. In the full paper we describe the BUILD-EXTENSION+ algorithm which, given a set of individuals Φ, a concept F in normal form different from (**bottom**) and (**top**), and an extension function $\mathcal{E}$ defined over a domain $\mathcal{D}$ such that $\Phi \cap \mathcal{D} = \emptyset$, "expands" $\mathcal{E}$ by building an extension function $\mathcal{E}'$ defined over a domain $\mathcal{D}'$ such that $\Phi \subseteq \mathcal{D}'$ and $\Phi \subseteq \mathcal{E}'[\![F]\!]$, and such that for every nonempty set $\Psi \subseteq \mathcal{D}$ it turns out that $\Psi \subseteq \mathcal{E}[\![F]\!]$ if and only if $\Psi \subseteq \mathcal{E}'[\![F]\!]$. This algorithm will be useful in determining that, if SUBS returns *false*, there exists an extension function $\mathcal{E}$ and a set of individuals Φ such that $\Phi \subseteq \mathcal{E}[\![F_2]\!]$ and $\Phi \not\subseteq \mathcal{E}[\![F_1]\!]$, and from this that F_2 is not subsumed by F_1.

Let us see an example of the application of BUILD-EXTENSION+. Let $F \equiv$(**and** (**all** R A) (**card** $\langle 2,3\rangle R$) B); let $\Phi = \{d\}$ and let $\mathcal{E}$ be the extension function defined over $\mathcal{D} = \{a,b\}$ and such that $\mathcal{E}[\![B]\!] = \{a\}$, $\mathcal{E}[\![A]\!] = \{a,b\}$ and $\mathcal{E}[\![R]\!] = \{\langle a,a\rangle, \langle a,b\rangle\}$. It should be observed that $a \in \mathcal{E}[\![F]\!]$, while $b \notin$

$\mathcal{E}[\![F]\!]$. By running BUILD-EXTENSION+(F,Φ,$\mathcal{E}$,$\mathcal{D}$) we obtain the extension function $\mathcal{E}'$ defined over $\mathcal{D}' = \{a, b, d, d_1^*, d_2^*\}$ and such that $\mathcal{E}'[\![B]\!] = \{a, d\}$, $\mathcal{E}'[\![A]\!] = \{a, b, d_1^*, d_2^*\}$ and $\mathcal{E}'[\![R]\!] = \{\langle a, a\rangle, \langle a, b\rangle, \langle d, d_1^*\rangle, \langle d, d_2^*\rangle\}$. It is easy to check that $\Phi \subseteq \mathcal{E}'[\![F]\!]$. It should be noted that it is still true that $a \in \mathcal{E}'[\![F]\!]$ e $b \notin \mathcal{E}'[\![F]\!]$.

The BUILD-EXTENSION+ plays a critical role in the proof of the following lemma.

Lemma 4 *Existence*
Let F=(**and** $a_1 \ldots a_n$) *be a concept in normal form,* Φ *and* Ψ *two sets of individuals,* $\Psi \subseteq \mathcal{D}$ *a nonempty set,* $\mathcal{E}$ *an extension function over* $\mathcal{D}$ *such that* $\Phi \cap \mathcal{D} = \emptyset$ *and* $\mathcal{E}'$ *the extension function defined over the domain* $\mathcal{D}'$ *which is returned by the algorithm* BUILD-EXTENSION+(F,Φ,$\mathcal{E}$,$\mathcal{D}$). *The following conditions hold:*

- $\Phi \subseteq \mathcal{D}'$
- $\Phi \subseteq \mathcal{E}'[\![F]\!]$;
- $\Psi \subseteq \mathcal{E}[\![F]\!]$ *if and only if* $\Psi \subseteq \mathcal{E}'[\![F]\!]$. ∎

We now switch to the proof of completeness.

Lemma 5 *Completeness*
Let F_1 *and* F_2 *be two concepts in normal form. If* SUBS(F_1, F_2)=false, *then* F_2 *is not subsumed by* F_1.

The proof of this lemma is rather complex, and is then described in detail only in the full paper. The proof proceeds by induction on the maximal depth k of F_1 and F_2; by using the existence lemma it may be shown that, when SUBS(F_1, F_2) returns *false*, there exists an extension function $\mathcal{E}$ and a nonempty set of individuals Φ such that $\Phi \subseteq \mathcal{E}[\![F_2]\!]$ e $\Phi \not\subseteq \mathcal{E}[\![F_1]\!]$.

Given the normalization, complexity, correctness and completeness lemmas, the following theorem follows.

Theorem 1
Given two concepts F_1 *and* F_2 *in* $\mathcal{ALN}$, *it may be determined in* $O(n \log n)$ *if* F_2 *is subsumed by* F_1, *where* $n = max(\{|F_1|, |F_2|\})$. ∎

4 Conclusion

In this work we have shown that there exists an $O(n \log n)$ algorithm which decides whether F_1 subsumes F_2, where F_1 and F_2 are concepts of the $\mathcal{ALN}$ logic. This result is significant for two reasons. First, $\mathcal{ALN}$ is one of the few logics for knowledge representation that is endowed with a correct, complete and tractable decision algorithm. Second, recent research has shown that a number of TLs which are slightly more expressive do not enjoy this property: $\mathcal{ALN} \cup \{\mathbf{restr}\}$ is notoriously co-NP-hard [3, 8], and $\mathcal{ALN} \cup \{\mathbf{androle}\}$ is also co-NP-hard [10]. Logics that are strictly more expressive than $\mathcal{ALN}$ but are at the same time computationally tractable may be hard to find.

References

[1] Franz Baader. A formal definition for the expressive power of knowledge representation languages. In *Proceedings of ECAI-90, 9th European Conference on Artificial Intelligence*, pages 53–58, Stockholm, Sweden, 1990.

[2] Ronald J. Brachman. A structural paradigm for representing knowledge. Technical Report 3605, Bolt Beranek and Newman, Cambridge, MA, 1978.

[3] Ronald J. Brachman and Hector J. Levesque. The tractability of subsumption in frame-based description languages. In *Proceedings of AAAI-84, 4th Conference of the American Association for Artificial Intelligence*, pages 34–37, Austin, TX, 1984. [a] An extended version appears as [8].

[4] Ronald J. Brachman and Hector J. Levesque, editors. *Readings in knowledge representation*. Morgan Kaufmann, San Mateo, CA, 1985.

[5] Francesco M. Donini, Maurizio Lenzerini, and Daniele Nardi. An efficient method for hybrid deduction. In *Proceedings of ECAI-90, 9th European Conference on Artificial Intelligence*, pages 246–252, Stockholm, Sweden, 1990.

[6] John Haugeland, editor. *Mind design*. The MIT Press, Cambridge, MA, 1981.

[7] Bernhard Hollunder, Werner Nutt, and Manfred Schmidt-Schauß. Subsumption algorithms for concept description languages. In *Proceedings of ECAI-90, 9th European Conference on Artificial Intelligence*, pages 348–353, Stockholm, Sweden, 1990.

[8] Hector J. Levesque and Ronald J. Brachman. Expressiveness and tractability in knowledge representation and reasoning. *Computational Intelligence*, 3:78–93, 1987.

[9] Marvin Minsky. A framework for representing knowledge. In Patrick J. Winston, editor, *The psychology of computer vision*, pages 211–277. McGraw-Hill, New York, NY, 1975. [a] An extended version appears in [4], pp. 245–262, and in [6], pp. 95–128.

[10] Bernhard Nebel. Computational complexity of terminological reasoning in BACK. *Artificial Intelligence*, 34:371–383, 1988.

[11] Bernhard Nebel. Terminological reasoning is inherently intractable. *Artificial Intelligence*, 43:235–249, 1990.

[12] Peter F. Patel-Schneider. Undecidability of subsumption in NIKL. *Artificial Intelligence*, 39:263–272, 1989.

[13] Klaus Schild. Undecidability of $\mathcal{U}$. Technical Report KIT 67, Department of Computer Science, Technische Universität Berlin, Berlin, FRG, 1988.

[14] Manfred Schmidt-Schauß. Subsumption in KL-ONE is undecidable. In *Proceedings of KR-89, 1st International Conference on Principles of Knowledge Representation and Reasoning*, pages 421–431, Toronto, Ontario, 1989.

[15] Manfred Schmidt-Schauß and Gert Smolka. Attributive concept descriptions with complements. *Artificial Intelligence*, 48:1–26, 1991.

[16] Fabrizio Sebastiani and Umberto Straccia. A computationally tractable terminological logic. Technical Report IEI-B4-45-1990, Istituto di Elaborazione dell'Informazione - CNR, Pisa, Italy, 1990.

[17] Kai von Luck. Semantic networks with number restricted roles (or: another story about Clyde). In *Proceedings of the 14th German Workshop on Artificial Intelligence*, pages 58–68, Eringerfeld, FRG, 1986.

[18] Kai von Luck and Bernd Owsnicki-Klewe. New AI formalisms for knowledge representation: a case study. Technical Report KIT 57, Fachbereich Informatik, Technische Universität Berlin, Berlin, FRG, 1987.