Published in *Artificial Intelligence* **43**: 235–249, 1990.

Terminological Reasoning is Inherently Intractable*

Bernhard Nebel
German Research Center for Artificial Intelligence
Stuhlsatzenhausweg 3, D-6600 Saarbrücken 11, West Germany

Abstract

Computational tractability has been a major concern in the area of terminological knowledge representation and reasoning. However, all analyses of the computational complexity of terminological reasoning are based on the hidden assumption that subsumption in terminologies reduces to subsumption of concept descriptions without a significant increase in computational complexity. In this paper it will be shown that this assumption, which seems to work in the "normal case," is nevertheless wrong. Subsumption in terminologies turns out to be co-NP-complete for a minimal terminological representation language that is a subset of every useful terminological language.

1 Introduction

One important task in modeling an application domain in artificial intelligence systems is to fix the vocabulary intended to describe the domain—the *terminology*—and to define interrelationships between the atomic parts of the terminology. *Terminological representation languages*, which are derived from KL-ONE [5], support this task. In such languages, it is possible to build up a terminology out of *atomic concepts* and attributes, usually called *roles*. The intended meaning of atomic concepts can be specified by providing *concept descriptions* made up of other concepts and *role restrictions*, as in the following informal example:

Woman	=	Person and Female,
Parent	=	Person with some child,
Grandmother	=	Woman with some child who is a Parent.

*The research described in this paper was carried out while the author was a guest researcher at the IBM Scientific Center, Institute for Knowledge-Based Systems, Stuttgart, West Germany.

The most important reasoning task in this context is the determination of *subsumption* between two concepts in the terminology, that is, whether all instances of one concept are necessarily instances of the other concept taking into account the definitions. For example, Grandmother is subsumed by Parent in the above terminology because everything that is a Grandmother is—by definition—also a Parent.

In their seminal paper [3] Brachman and Levesque analyzed an important subproblem of subsumption determination in terminologies, namely, the problem of deciding whether one concept description subsumes another one, assuming that all atomic concepts are *primitive*, i.e., undefined in the terminology. They showed that a minor and innocent looking change in the *concept-description language*[1] leads to a dramatic increase of the computational costs—an increase from polynomial complexity to co-NP-hardness.

This paper initiated a number of other investigations of the computational complexity in related concept-description languages [19, 24, 27, 28, 29], which provide us with a good picture of the computational problems associated with subsumption determination of concept descriptions. Additionally, a number of efforts probed how far one can go in designing terminological reasoning systems without "falling off the computational cliff." Some approaches aimed towards identifying restricted concept-description languages that permit tractable subsumption determination of concept descriptions, exercised, for example, in the terminological subcomponents of KANDOR [23], KRYPTON [4], MESON [8], and recently CLASSIC [1, 2]. Another interesting approach, pursued by Patel-Schneider [22], employs a weaker semantics to achieve tractability of subsumption determination of concept descriptions for quite expressive concept-description languages.

All of these efforts tried to achieve the goal of "forging a powerful system out of tractable parts" [15, p. 89]—and all of them have failed (at least, from a theoretical point of view). There is a "forgotten computational cliff" in terminological representation languages that leads to intractability for all representation languages that have at least the expressive power of the terminological representation language used in KRYPTON [4]. The reason is that subsumption in terminologies is computationally much harder than subsumption of pure concept descriptions.

In section 2, basic definitions of syntax and semantics of a small terminological representation languages are given and connected with the results of Brachman and Levesque [3]. A proof that subsumption in this language is co-NP-complete is presented in Section 3. In Section 4, reasons why the theoretical worst-case behavior occurs only seldomly in practice are analyzed. Finally, conclusions are discussed in Section 5.

---

[1] Called *frame-description language* in [3] and *terminological logic* in [22].

2 Terminological Representation and Reasoning

The terminological representation language that will be analyzed in this paper, called $\mathcal{TL}$, is a subset of the terminological language of KRYPTON presented in [4].[2] Starting with a finite set $\mathbf{A}$ of *atomic concepts* (denoted by A) and a finite set $\mathbf{R}$ of *atomic roles* (denoted by R), the set $\mathbf{D}$ of *concept descriptions* (denoted by C, D) is defined by the following rule:[3]

$$C, D \;\rightarrow\; A \mid C \sqcap D \mid \forall R\colon C.$$

The meaning of these expressions is given by a model-theoretic interpretation $\mathcal{I} = \langle \mathcal{D}, [\![\cdot]\!]^{\mathcal{I}} \rangle$, where $\mathcal{D}$ is an arbitrary set, the *domain*, and $[\![\cdot]\!]^{\mathcal{I}}$ is a function, the *interpretation function*, that maps atomic concepts to subsets of $\mathcal{D}$ and atomic roles to total functions from $\mathcal{D}$ to $2^{\mathcal{D}}$. The *denotation of concept descriptions* is defined inductively by

$$\begin{aligned} [\![C \sqcap D]\!]^{\mathcal{I}} &= [\![C]\!]^{\mathcal{I}} \cap [\![D]\!]^{\mathcal{I}} \\ [\![\forall R\colon C]\!]^{\mathcal{I}} &= \{d \in \mathcal{D} \mid [\![R]\!]^{\mathcal{I}}(d) \subseteq [\![C]\!]^{\mathcal{I}}\}. \end{aligned}$$

A *terminology* T is then a function $T\colon \mathbf{A} \rightarrow \mathbf{D}$, where $T(A)$ is the concept description defining the meaning of A or, if A is *primitive in the terminology*, $T(A) = A$. An interpretation $\mathcal{I}$ is a *model of* T iff

$$[\![A]\!]^{\mathcal{I}} = [\![T(A)]\!]^{\mathcal{I}} \;\; \textit{for all } A \in \mathbf{A}.$$

Now subsumption can be formalized as follows: *C is subsumed by D in the terminology T*, written $C \preceq_T D$ iff

$$[\![C]\!]^{\mathcal{I}} \subseteq [\![D]\!]^{\mathcal{I}} \;\; \textit{for all models } \mathcal{I} \textit{ of } T.$$

In [3] subsumption is defined over concept descriptions only or, alternatively, in the *empty terminology*, denoted by Id, for which $Id(A) = A$, for all $A \in \mathbf{A}$. In order to reduce our notion of subsumption to this one, the canonical extension of T to concept descriptions, denoted by $\widehat{T}$, is defined as follows:

$$\begin{aligned} \widehat{T}\colon \mathbf{D} &\rightarrow \mathbf{D} \\ \widehat{T}(C \sqcap D) &= \widehat{T}(C) \sqcap \widehat{T}(D) \\ \widehat{T}(\forall R\colon C) &= \forall R\colon \widehat{T}(C) \\ \widehat{T}(A) &= T(A). \end{aligned}$$

[2]Here, will give only the formal definitions and refer the reader to [4] for a more intuitive introduction.

[3]To ease notation, it will be assumed that ":" has higher precedence than "$\sqcap$".

In the following, only *acyclic terminologies* will be considered,[4] namely, those T such that there exists a natural number n with

$$\widehat{T}^n \quad = \quad \widehat{T}^{n+1}.$$

If such a number exists, then the least such n is called the *depth of* T, written $depth(T) = n$, and

$$\widetilde{T} \quad = \quad \widehat{T}^n$$

is called the *completely expanded terminology.* Note that $\widetilde{T}$ is a denotation-preserving function.

Proposition 1 *For every $C \in \mathbf{D}$, every acyclic terminology T, and every model $\mathcal{I}$ of T:*

$$[\![C]\!]^{\mathcal{I}} \quad = \quad [\![\widetilde{T}(C)]\!]^{\mathcal{I}}.$$

The semantic relation between T and $\widetilde{T}$ is even stronger. Defining an *initial partial interpretation* (denoted by $\check{\mathcal{I}} = \langle \mathcal{D}, [\![\cdot]\!]^{\check{\mathcal{I}}} \rangle$) as the restriction of an interpretation function to atomic roles and primitive atomic concepts, the next lemma provides us with an easy way to generate models of a terminology.

Lemma 1 *Let T be an acyclic terminology and $\check{\mathcal{I}}$ be an initial partial interpretation. Then there exists a model $\mathcal{I}$ of T such that*

$$[\![A]\!]^{\mathcal{I}} \;=\; [\![\widetilde{T}(A)]\!]^{\check{\mathcal{I}}} \;\; \textit{for all } A \in \mathbf{A}.$$

Proof: First of all, note that $[\![\widetilde{T}(A)]\!]^{\check{\mathcal{I}}}$ is well-defined since $\widetilde{T}(A)$ contains only atomic concepts primitive in T.

Now we will show by induction over the depth of T that any initial partial interpretation $\check{\mathcal{I}}$ can be extended to a model $\mathcal{I}$ of T. If $depth(T) = 1$, then setting $[\![A]\!]^{\mathcal{I}} = [\![T(A)]\!]^{\check{\mathcal{I}}}$, $A \in \mathbf{A}$, obviously leads to a model $\mathcal{I}$ of T since for all atomic concepts $[\![A]\!]^{\mathcal{I}} = [\![T(A)]\!]^{\mathcal{I}}$ is satisfied without leading to a conflict because no defined atomic concept appears in any $T(A)$.

For the induction step, assume $depth(T) = k+1$. Let T_k the restriction of T to *atomic concepts of level* k: $\mathbf{A}_k = \{A \in \mathbf{A} \mid \widehat{T}^k(A) = \widehat{T}^{k+1}(A)\}$. By the induction hypothesis, there exists a model $\mathcal{I}'$ of T_k that extends $\check{\mathcal{I}}$. Define an interpretation $\mathcal{I}$ by $[\![A]\!]^{\mathcal{I}} = [\![T(A)]\!]^{\mathcal{I}'}$. Since all atomic concepts in $T(A)$ are of level k, the extension of $\mathcal{I}'$ to $\mathcal{I}$ leads to the satisfaction of all equations $[\![A]\!]^{\mathcal{I}} = [\![T(A)]\!]^{\mathcal{I}}$ without introducing a conflict. Hence, $\mathcal{I}$ is a model of T.

[4] This restriction is usually taken for granted in terminological representation languages and systems [4, p. 534]. However, it can be given up without running into too much trouble [20].

Finally, by Proposition 1 and the fact that $\widetilde{T}(A)$ contains only atomic concepts primitive in T, which are identically interpreted by $\breve{\mathcal{I}}$ and $\mathcal{I}$, the desired equation follows. ■

Now the reduction from subsumption in terminologies to subsumption of concept descriptions is straightforward.

Theorem 1 *Let T be an acyclic terminology and C, D two concept descriptions. Then*

$$C \preceq_T D \quad \textit{iff} \quad \widetilde{T}(C) \preceq_{Id} \widetilde{T}(D).$$

Proof: For the "if" direction note that any model of T is also a model of Id. Thus subsumption in Id implies subsumption in T and with Proposition 1 the "if" direction follows.

For the converse direction, assume $\mathcal{I}$ is a model of Id. Since $\widetilde{T}(C)$ and $\widetilde{T}(D)$ contain only atomic concepts primitive in T, we know that for the initial partial interpretation $\breve{\mathcal{I}}$: $[\![\widetilde{T}(C)]\!]^{\breve{\mathcal{I}}} \subseteq [\![\widetilde{T}(D)]\!]^{\breve{\mathcal{I}}}$. By Lemma 1, this initial partial interpretation can be extended to a model $\mathcal{I}'$ of T and with Proposition 1 we have $[\![C]\!]^{\mathcal{I}'} \subseteq [\![D]\!]^{\mathcal{I}'}$. ■

Employing this theorem, the results of [3] can be applied to our language $\mathcal{TL}$. In particular, the algorithm SUBS? [3, p. 36], which computes subsumption in the empty terminology for a slightly more expressive concept-description language, can be used to define an algorithm TSUBS? that determines subsumption in arbitrary terminologies:

$$\text{TSUBS?}(C, D, T) \quad = \quad \text{SUBS?}(\widetilde{T}(C), \widetilde{T}(D)).$$

Unfortunately, TSUBS? does not inherit the polynomial complexity of SUBS?, as the following example illustrates:

$$\begin{array}{rcl}
T(\mathsf{C}_0) & = & \mathsf{C}_0 \\
T(\mathsf{C}_1) & = & \forall \mathsf{R}\colon \mathsf{C}_0 \sqcap \forall \mathsf{R}'\colon \mathsf{C}_0 \\
T(\mathsf{C}_2) & = & \forall \mathsf{R}\colon \mathsf{C}_1 \sqcap \forall \mathsf{R}'\colon \mathsf{C}_1 \\
 & \vdots & \\
T(\mathsf{C}_n) & = & \forall \mathsf{R}\colon \mathsf{C}_{n-1} \sqcap \forall \mathsf{R}'\colon \mathsf{C}_{n-1}.
\end{array}$$

The execution of TSUBS?$(\mathsf{C}_n, \mathsf{C}_n, T)$ leads to expressions $\widetilde{T}(C_n)$ of size $O(2^n)$ and, hence, to a running time exponential in the size of the terminology.

This situation seems to resemble the problem in first-order term-unification, where the string representation of a unified term can be exponential in the size of the original terms. Thus, one might hope that techniques similar to the ones used to design linear term-unification algorithms [25, 18] could be helpful. Unfortunately, this hope turns out to be unjustified, however.

3 The Complexity of Terminological Reasoning

Subsumption in $\mathcal{TL}$ is co-NP-complete, as will be shown by reducing the co-NP-complete problem of deciding whether two nondeterministic finite state automatons that accept finite languages are equivalent to *equivalence of concepts in a terminology*, written as $C \approx_T D$ and defined by

$$C \approx_T D \quad \textit{iff} \quad [\![C]\!]^{\mathcal{I}} = [\![D]\!]^{\mathcal{I}} \quad \textit{for all models } \mathcal{I} \textit{ of } T.$$

In order to prove this, first a conceptually simple normal form of concept descriptions is defined, using an *unfolding function* on concept descriptions:

$$\begin{array}{rcl} U\colon \mathbf{D} & \rightarrow & 2^{\mathbf{D}} \\ U(C \sqcap D) & = & U(C) \cup U(D) \\ U(\forall R\colon C) & = & \{(\forall R\colon D) \mid D \in U(C)\} \\ U(A) & = & \{A\}. \end{array}$$

The *completely unfolded form* U_T of a terminology T is defined by

$$U_T = U \circ \widetilde{T}.$$

Proposition 2 *Let T be an acyclic terminology and $C \in \mathbf{D}$. Then for all models $\mathcal{I}$ of T:*

$$[\![C]\!]^{\mathcal{I}} \;\; = \;\; [\![\textstyle\bigsqcap U_T(C)]\!]^{\mathcal{I}}.$$

A concept description of the form $\forall R_1\colon \forall R_2\colon \ldots \forall R_n\colon A$ will be called *linear description*, and written as $\forall W\colon A$ with $W = R_1 R_2 \ldots R_n$. For $n = 0$, that is, $W = \epsilon$, the convention $\forall \epsilon\colon A = A$ will be adopted.

Proposition 3 *Let T be an acyclic terminology and C be a concept description. Then every concept description $D \in U_T(C)$ is a linear description $D = \forall W\colon A$ such that A is primitive in T.*

Employing the completely unfolded form of a terminology,[5] equivalence can be decided by using a simple syntactic criterion.

Lemma 2 *Let T be an acyclic terminology, and let C and D be concept descriptions. Then*

$$C \approx_T D \quad \textit{iff} \quad U_T(C) = U_T(D).$$

---

[5]Incidentally, $U_T(A)$ is identical to the *canonicalized form for a concept* used in the KRYPTON system [26, p. 8].

Proof: The "if" direction follows directly from Proposition 2.

For the other direction assume that $(\forall W\colon B) \in U_T(C)$ but $(\forall W\colon B) \notin U_T(D)$. Now, we construct an initial partial interpretation $\check{\mathcal{I}}$ that can be extended to a model of T employing Lemma 1. This model will have the property that for an element $d_0 \in \mathcal{D}$, we have $d_0 \notin [\![C]\!]^{\mathcal{I}}$ but $d_0 \in [\![D]\!]^{\mathcal{I}}$.

Let $W = R_1 R_2 \ldots R_n$. Then set $\mathcal{D} = \{d_0, d_1, \ldots d_n\}$. Set $[\![A]\!]^{\check{\mathcal{I}}} = \mathcal{D}$ for all atomic concepts $A \neq B$ primitive in T and $[\![B]\!]^{\check{\mathcal{I}}} = \mathcal{D} - \{d_n\}$. Set $[\![R]\!]^{\check{\mathcal{I}}}(e) = \emptyset$ except for $[\![R_i]\!]^{\check{\mathcal{I}}}(d_{i-1}) = \{d_i\}$, for $1 \leq i \leq n$.

Extending $\check{\mathcal{I}}$ to a model $\mathcal{I}$ of T, we have $d_0 \notin [\![\forall W\colon B]\!]^{\mathcal{I}}$ because $d_n \notin [\![B]\!]^{\mathcal{I}}$. Furthermore, since $(\forall W\colon B) \in U_T(C)$:

$$d_0 \notin \bigcap\{[\![E]\!]^{\mathcal{I}} \mid E \in U_T(C)\} = [\![\textstyle\bigsqcap U_T(C)]\!]^{\mathcal{I}} = [\![C]\!]^{\mathcal{I}}.$$

On the other hand, $d_0 \in [\![\forall V\colon A]\!]^{\mathcal{I}}$ for every $V \neq W$ and arbitrary primitive atomic concepts A. This is easily seen when one considers the fact that $d \in [\![\forall R\colon E]\!]^{\mathcal{I}}$ if $[\![R]\!]^{\mathcal{I}}(d) = \emptyset$. Thus, we have $d_0 \in [\![D]\!]^{\mathcal{I}}$ and by this $C \not\approx_T D$. ∎

Now it will be shown that equivalence of automatons can be reduced to concept equivalence. A *nondeterministic finite state automaton* (NDFA)[6] is a tuple $\mathcal{A} = (\Sigma, \mathcal{Q}, \delta, q_0, \mathcal{F})$ where Σ is a set of *input symbols* (denoted by s), $\mathcal{Q}$ is a set of *states* (denoted by q), δ is a total function from $\Sigma \times \mathcal{Q}$ to $2^{\mathcal{Q}}$, $q_0 \in \mathcal{Q}$ is the *initial state*, and $\mathcal{F} \subseteq \mathcal{Q}$ is the set of *final* or *accepting states*.

A state $q' \in \mathcal{Q}$ is *reachable* from another state q *by a word* $w = s_1 s_2 \ldots s_n$ iff there exists a sequence of states $q_1, q_2, \ldots q_{n+1}$ with $q = q_1$, $q' = q_{n+1}$, and $q_{i+1} \in \delta(q_i, s_i)$, $1 \leq i \leq n$. The set of words w such that some final state $q' \in \mathcal{F}$ is reachable from q_0 by w is called the *language accepted by* $\mathcal{A}$ and denoted by $\mathcal{L}(\mathcal{A})$.

Two automatons $\mathcal{A}^1$ and $\mathcal{A}^2$ are *equivalent* iff $\mathcal{L}(\mathcal{A}^1) = \mathcal{L}(\mathcal{A}^2)$. Deciding this problem is PSPACE-complete in the general case and co-NP-complete if the accepted languages are finite [9, p. 265].

A state $q \in \mathcal{Q}$ is said to be *redundant* iff q is not reachable from q_0 by any word, or if q cannot reach a final state by any word. A NDFA is *nonredundant* iff it does not contain redundant states. Furthermore, a NDFA is *acyclic* iff no state can reach itself by a nonempty word.

Proposition 4 *For any NDFA $\mathcal{A}$ an equivalent nonredundant NDFA $\mathcal{A}'$ can be identified in polynomial time.*

Proof: It is possible to mark all states reachable from the initial state in polynomial time. Similarly, all states that can reach a final state can be marked in polynomial time. Taking the intersection of the two sets of marked states

[6] Without loss of generality, we consider only ϵ-free NDFAs.

results in the set of all nonredundant states. Restricting the automaton to this set results obviously in an equivalent automaton that is nonredundant. ■

Since a language of a NDFA cannot be infinite if the NDFA does not contain a cycle over nonredundant states, and since every cycle using nonredundant states leads to an infinite language, the next proposition is immediate.

Proposition 5 *Let $\mathcal{A}$ be an nonredundant NDFA. Then $\mathcal{L}(\mathcal{A})$ is finite if and only if $\mathcal{A}$ is acyclic.*

Thus, it suffices to consider acyclic, nonredundant NDFAs (ANDFAs) in the following, for which a translation to terminologies is specified. Given two ANDFAs $\{\mathcal{A}^i = (\Sigma, \mathcal{Q}^i, \delta^i, q_0^i, \mathcal{F}^i)\}_{i=1,2}$ with $\mathcal{Q}^1 \cap \mathcal{Q}^2 = \emptyset$, a terminology $T_{\mathcal{A}}$ is constructed:

$$\begin{aligned}
\mathbf{R} &= \Sigma \\
\mathbf{A} &= \mathcal{Q}^1 \cup \mathcal{Q}^2 \cup \{\mathsf{F}\} \\
T_{\mathcal{A}}(\mathsf{F}) &= \mathsf{F} \\
T_{\mathcal{A}}(q) &= \bigsqcap\Big(\{(\forall s\colon q') \mid q' \in \delta^i(q,s), i = 1,2\} \cup \{\mathsf{F} \mid q \in \mathcal{F}^1 \cup \mathcal{F}^2\}\Big).
\end{aligned}$$

Note that $T_{\mathcal{A}}$ is indeed a legal terminology. Since the ANDFAs are nonredundant, every state is either final or has an "outgoing" arc. Thus, every atomic concept q has a nonempty definition $T_{\mathcal{A}}(q)$. Furthermore, $T_{\mathcal{A}}$ is an acyclic terminology because ANDFAs are acyclic, that is, $U_{T_{\mathcal{A}}}$ is well-defined.

Lemma 3 *Let $T_{\mathcal{A}}$ be a terminology generated from two ANDFAs $\{\mathcal{A}^1, \mathcal{A}^2\}$. Then*

$$w \in \mathcal{L}(\mathcal{A}^i) \quad \textit{iff} \quad (\forall w\colon \mathsf{F}) \in U_{T_{\mathcal{A}}}(q_0^i)$$

Proof: First, by Proposition 3 we know that all $C \in U_{T_{\mathcal{A}}}(q)$ are linear descriptions. Since F is the only primitive atomic concept and Σ is the set of atomic roles, all C's must be linear description of the form $(\forall w\colon \mathsf{F})$, where $w \in \Sigma^*$.

Assume that $w = s_1 s_2 \ldots s_n$ is accepted by $\mathcal{A}^i$. Then there is a sequence of states $q_0, q_1, \ldots q_n$ with $q_0 = q_0^i$, $q_n \in \mathcal{F}^i$, and $q_{j+1} \in \delta^i(q_j, s_{j+1})$, for $0 \leq j \leq n-1$. Because of the way $T_{\mathcal{A}}$ was constructed and by induction over the length of w, it follows that $(\forall (s_{j+1}s_{j+2}\ldots s_n)\colon \mathsf{F}) \in U_{T_{\mathcal{A}}}(q_j)$, with $q_0 = q_0^i$, for $0 \leq j \leq n-1$. For this reason $(\forall (s_1 s_2 \ldots s_n)\colon \mathsf{F}) \in U_{T_{\mathcal{A}}}(q_0^i)$.

Conversely, assume that $(\forall w\colon \mathsf{F}) \in U_{T_{\mathcal{A}}}(q_0^i)$. Then, because of the way $T_{\mathcal{A}}$ was set up and by induction over the length of w, a state $q \in \mathcal{F}^i$ is reachable from q_0^i by w in $\mathcal{A}^i$, that is, $w \in \mathcal{L}(\mathcal{A}^i)$. ■

Theorem 2 *Equivalence in $\mathcal{TL}$ is co-NP-complete.*

Proof: By Lemma 2 and Lemma 3 it is immediate that

$$\mathcal{L}(\mathcal{A}^1) = \mathcal{L}(\mathcal{A}^2) \quad \textit{iff} \quad q_0^1 \approx_{T_\mathcal{A}} q_0^2. \tag{1}$$

Furthermore, since equivalence of nondeterministic automatons that accept finite languages is co-NP-complete, Proposition 4 and Proposition 5, and the fact that the terminology $T_\mathcal{A}$ can be constructed in time polynomial in the size of the ANDFAs, concept equivalence is co-NP-hard.

That concept equivalence is in co-NP follows from Lemma 2. After guessing a linear description, it suffices to expand and unfold the concept descriptions only along the chain of roles used in the guessed linear description in order to test whether the linear description is an element of the completely unfolded concepts or not. Since expansion and unfolding of a concept description along a given chain of roles can be performed in polynomial time, membership in co-NP follows. ■

Corollary 1 *Subsumption in $\mathcal{TL}$ is co-NP-complete.*

Proof: Subsumption can be reduced to concept equivalence in linear time and *vice versa* because of

$$\begin{aligned} C \preceq_T D \quad &\textit{iff} \quad C \approx_T C \sqcap D, \\ C \approx_T D \quad &\textit{iff} \quad (C \preceq_T D \quad \textit{and} \quad D \preceq_T C). \end{aligned}$$

■

Finally, there might be the question of how this result extends to the cyclic case. The answer is that this depends on the style of semantics one assigns to concepts which are defined by cycles. If one takes the loose semantics defined in this paper,[7] then the result cannot be straightforwardly extended. The reason is that for this style of semantics the two concepts X and Y in the following example are not equivalent:

$$\begin{aligned} T(\mathsf{X}) &= \forall \mathsf{R}{:}\, \mathsf{X} \sqcap \mathsf{F}, \\ T(\mathsf{Y}) &= \forall \mathsf{R}{:}\, \mathsf{Y} \sqcap \mathsf{F}, \end{aligned}$$

which raises the question for the computational complexity in this case. However, if an appropriate fixed point semantics (least or greatest) is used, the PSPACE-completeness result for NDFA equivalence will probably carry over to subsumption in terminologies.

[7] In [20] it is argued that this is the kind of semantics which captures the intuitive notion of such cycles.

4 Efficiency of Subsumption in Practice

As mentioned in Section 1, it was originally believed that subsumption in terminologies is no harder than subsumption of concept descriptions. The reason for this is probably that worst cases show up in practice with a very low frequency. In this respect and with regard to the structure of the problem, our result is very similar to a result about the complexity of type inference in *core* ML, which had been believed to be linear. Only recently it has been shown [11] that the problem is PSPACE-hard.

Looking for an explanation of why subsumption in terminologies is well-behaved, one notes that in case of TSUBS? the depth of a terminology leads to a combinatorial explosion. Assuming that $|C|$ denotes the size of a concept description and $\|M\|$ denotes the cardinality of the set M, the following parameters are important:

$$\begin{aligned} n &= \mathit{depth}(T), \\ m &= \max\Big(\Big\{|T(A)|\,\Big|\, A \in \mathbf{A}\Big\}\Big), \\ s &= m \times \|\mathbf{A}\|. \end{aligned}$$

Now, it is easy to see that the size of expressions $\widetilde{T}(C)$ generated by TSUBS? is at most $O(m^n)$. Thus, in all cases such that $n \le \log_m s$, which is a reasonable assumption, the expanded concept descriptions have a size of $O(s)$. Since SUBS? is quadratic in the size of the concept descriptions, TSUBS? is quadratic in the size of the knowledge base.

Although this approximation characterizes already a large class of all interesting problems, it is possible to specify an algorithm—called *classification algorithm* [16]—which handles an even larger class in polynomial time. This algorithm, which is used in most of the existing terminological representation systems, computes the subsumption relation between all atomic concepts—the so-called *concept taxonomy*—in advance, speeding up subsumption determination between atomic concepts at run-time. Furthermore, the computation of the concept taxonomy is done in a way such that a new atomic concept is inserted into the concept taxonomy only after all concepts used in role restrictions have been inserted into the concept taxonomy. This strategy avoids unnecessary recomputations of the subsumption relation for identical role restrictions, which led to an expontial explosion of TSUBS? when applied to the terminology presented in Section 2. Using the classification algorithm, it is possible to precompute the subsumption relation between all atomic concepts in polynomial time in this case.

The criticial point in the classification algorithm is the insertion of concepts used in role restrictions into the concept hierarchy. This can imply that a *conjunction* of two concepts must be inserted first. For instance, when we have

$$T(\mathsf{C}) \;=\; \forall \mathsf{R}{:}\, \mathsf{D} \sqcap \forall \mathsf{R}{:}\, \mathsf{E},$$

then C can be inserted into the concept taxonomy only after the conjunction $(\mathsf{D} \sqcap \mathsf{E})$ has been inserted. This in turn can imply that other conjunctions of role restrictions used in the definitions of D and E must be inserted into the concept taxonomy first.

Although, the worst-case computational costs of the classification algorithm are, of course, the same as for TSUBS?, it is difficult to find a terminology which blows up the classification algorithm. In order to give an impression what such a worst case might look like, here is an example:[8]

$$
\begin{array}{rcl}
T(\mathsf{C}_0) & = & \forall \mathsf{R}{:}\, \mathsf{C}_1 \sqcap \forall \mathsf{R}'{:}\, (\mathsf{C}_1 \sqcap \mathsf{C}_2) \\
T(\mathsf{C}_1) & = & \forall \mathsf{R}{:}\, \mathsf{C}_2 \sqcap \forall \mathsf{R}'{:}\, (\mathsf{C}_2 \sqcap \mathsf{C}_4) \\
 & \vdots & \\
T(\mathsf{C}_i) & = & \left\{ \begin{array}{ll} \forall \mathsf{R}{:}\, \mathsf{C}_{\mathsf{i}+1} \sqcap \forall \mathsf{R}'{:}\, (\mathsf{C}_{i+1} \sqcap \mathsf{C}_{2i}) & ,\ \textit{if } 2i \leq n \\ \forall \mathsf{R}{:}\, \mathsf{C}_{\mathsf{i}+1} \sqcap \forall \mathsf{R}'{:}\, \mathsf{C}_{i+1} & ,\ \textit{otherwise} \end{array} \right. \\
 & \vdots & \\
T(\mathsf{C}_n) & = & \mathsf{C}_n.
\end{array}
$$

Obviously, this terminology does not look very natural. More generally, it is the case that all terminologies I have seen so far are well-behaved in the sense that only a few conjunctions of concepts have to be inserted into the concept taxonomy in order to compute the subsumption relation for all atomic concepts. Moreover, a similar approximation as the one given above applies to classification. Let

$$
\begin{array}{rcl}
a & = & \|\mathbf{A}\|, \\
r & = & \|\mathbf{R}\|, \\
l & = & \max\Big(\Big\{|D| \,\Big|\, D \in U_T(A), A \in \mathbf{A}\Big\}\Big),
\end{array}
$$

then in the worst case $O(a \times r^l)$ conjunctions have to be inserted into the concept taxonomy. That means that under the plausible assumption that $l \leq \log_r a$, "only" $O(a^2)$ conjunctions of concepts have to be inserted.

5 Conclusion

Brachman and Levesque [15] argued that a knowledge representation system should be *dependable*, that is, sound and correct with respect to its formalization and, most importantly, that it should give an answer in a reasonable amount

[8]Note that this example can even blow up terminological systems which are deliberately designed to be incomplete in their reasoning, e.g. NIKL [10], BACK [30], LOOM [17], and SB-ONE [13]. The only requirement is that the equivalence $\forall R{:}\, C \sqcap \forall R{:}\, D \approx_T \forall R{:}\, (C \sqcap D)$ is handled completely.

of time. The latter is particularly important when the representation system provides services to a larger system, which depends on timely answers of the representation subsystem. In this case, it seems to be extremely desirable to restrict oneself to tractable forms of reasoning:

> As responsible computer scientists, we should not be providing a general inferential service if all we can say about it is that by and large it will probably work satisfactorily. [15, p. 81]

Unfortunately, it seems to be the case that intractability lurks behind every corner, as we have seen in this paper.[9] There are a number of possible strategies to cope with this problem. Two of them, favored in [15], are restricting the expressiveness of the representation language and weakening the semantics. Both strategies do not seem to be reasonable in our case, and they are not necessary, either. As long as a terminology obeys the restrictions discussed in the provious section—and I have not seen a terminology yet that violates these restriction—subsumption determination in $\mathcal{TL}$ is provably tractable.

Furthermore, although this strategy works probably only for a very limited number of intractable problems (e.g., subsumption in $\mathcal{TL}$ and type inference in *core* ML) without considerable problems, often there is no freedom regarding expressiveness or semantics. Changing the rules of the game can lead to uselessness of a representation formalism.[10] For instance, unification grammars rely heavily on disjunctive feature terms[11] in order to represent ambiguity of natural language expressions. Although reasoning with such disjunctive feature terms is worst-case intractable, nobody wants to give them up—they are simply necessary for the particular kind of problem one wants to solve. The challenge then is to identify the structure of normal cases and to design algorithms that deal with these normal cases [12, 6]—which are informally characterized by the fact that humans can deal with them effortlessly.

Summarizing, after discovering that a representation formalism leads to intractable reasoning problems, it is worthwhile to analyze how the representation formalism is *used*. If it turns out, that the cases occurring in practice can be handled by a polynomial algorithm, although reasoning in general is intractable (see also [14]), then it seems better to support these normal cases than to restrict the language or to weaken the semantics.

Acknowledgements

[9] Anybody believing that AI deals only with intractable problems has now evidence that terminological knowledge representation even in its weakest form still belongs to AI.

[10] This is a point also noted by Doyle and Patil [7] about the restricted language approach in terminological knowledge representation.

[11] Feature terms are very similar to concept descriptions. However, instead of multi-valued attributes (roles), single-valued attributes (features) are used (see e.g. [21]).

I am particularly grateful to Gert Smolka for a number of enlighting discussions and for his insightful comments on a draft version of this paper. I would also like to thank Alex Borgida, who gave me the hint that subsumption in terminologies resembles the type inference problem in ML. Finally, I would like to thank the two anonymous referees for their detailed comments on the paper.

References

[1] A. Borgida, R. J. Brachman, D. L. McGuinness, and L. A. Resnick, CLASSIC: a structural data model for objects, in: *Proceedings of the 1989 ACM SIGMOD International Conference on Mangement of Data*, Portland, Oreg. (1989) 59–67.

[2] R. J. Brachman, A. Borgida, D. L. McGuinness, and L. A. Resnick, The CLASSIC knowledge representation system, or, KL-ONE: the next generation, in: *Preprints of the Workshop on Formal Aspects of Semantic Networks*, Two Harbors, Cal. (1989).

[3] R. J. Brachman and H. J. Levesque, The tractability of subsumption in frame-based description languages, in: *Proceedings of the 4th National Conference of the American Association for Artificial Intelligence*, Austin, Tex. (1984) 34–37.

[4] R. J. Brachman, V. Pigman Gilbert, and H. J. Levesque, An essential hybrid reasoning system: knowledge and symbol level accounts in KRYPTON, in: *Proceedings of the 9th International Joint Conference on Artificial Intelligence*, Los Angeles, Cal. (1985) 532–539.

[5] R. J. Brachman and J. G. Schmolze, An overview of the KL-ONE knowledge representation system, *Cognitive Science* 9(2) (1985) 171–216.

[6] J. Dörre and A. Eisele, Determining consistency of feature terms with distributed disjunctions, in: D. Metzing, ed., *GWAI-89. 13th German Workshop on Artificial Intelligence*, (Springer-Verlag, Berlin, West Germany, 1989) 270–279.

[7] J. Doyle and R. S. Patil, Language restrictions, taxonomic classifications, and the utility of representation services, Technical Memo MIT/LCS/TM-387, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, Mass. (1989).

[8] J. Edelmann and B. Owsnicki, Data models in knowledge representation systems: a case study, in: C. Rollinger and W. Horn, eds., *GWAI-86 und*

2. Österreichische Artificial-Intelligence-Tagung, (Springer-Verlag, Berlin, West Germany, 1986) 69–74.

[9] M. R. Garey and D. S. Johnson, *Computers and Intractability—A Guide to the Theory of NP-Completeness*, (Freeman, San Francisco, Cal., 1979).

[10] T. S. Kaczmarek, R. Bates, and G. Robins, Recent developments in NIKL, in: *Proceedings of the 5th National Conference of the American Association for Artificial Intelligence*, Philadelphia, Pa. (1986) 978–987.

[11] P. C. Kanellakis and J. C. Mitchell, Polymorphic unification and ML typing, in: *Proceedings of the 16th ACM Symposium on Principles of Programming Languages* (1989) 5–15.

[12] R. T. Kasper, A unification method for disjunctive feature descriptions, in: *Proceedings of the 25th Annual Meeting of the ACL*, Stanford, Cal. (1987) 235–242.

[13] A. Kobsa, The SB-ONE knowledge representation workbench, in: *Preprints of the Workshop on Formal Aspects of Semantic Networks*, Two Harbors, Cal. (1989).

[14] H. J. Levesque, Logic and the complexity of reasoning, *Journal of Philosophical Logic* 17 (1988) 355–389.

[15] H. J. Levesque and R. J. Brachman, Expressiveness and tractability in knowledge representation and reasoning, *Computational Intelligence* 3 (1987) 78–93.

[16] T. Lipkis, A KL-ONE classifier, in: J. G. Schmolze and R. J. Brachman, eds., *Proceedings of the 1981 KL-ONE Workshop*, Cambridge, Mass. (1982) 128–145. The proceedings have been published as BBN Report No. 4842 and Fairchild Technical Report No. 618.

[17] R. MacGregor and R. Bates, The Loom knowledge representation language, Technical Report ISI/RS-87-188, University of Southern California, Information Science Institute, Marina del Rey, Cal. (1987).

[18] A. Martelli and U. Montanari, An efficient unification algorithm, *ACM Trans. Programming Languages and Systems* 4(2) (1982) 258–282.

[19] B. Nebel, Computational complexity of terminological reasoning in BACK, *Artificial Intelligence* 34(3) (1988) 371–383.

[20] B. Nebel, On terminological cycles, in: *Preprints of the Workshop on Formal Aspects of Semantic Networks*, Two Harbors, Cal. (1989). The proceedings

will be published by Morgan Kaufmann. A preliminary version is available as KIT Report Department of Science, Technische Universität Berlin, November 1987.

[21] B. Nebel and G. Smolka, Representation and reasoning with attributive descriptions, in: K. Bläsius, U. Hedtstück, and C. Rollinger, eds., *Sorts and Types in Artificial Intelligence*, (Springer-Verlag, Berlin, West Germany, 1989). To appear. Also available as IWBS Report 81, IBM Germany Scientific Center, IWBS, Stuttgart, West Germany, September 1989.

[22] P. F. Patel-Schneider, A four-valued semantics for terminological logics, *Artificial Intelligence* 38(3) (1989) 319–351.

[23] P. F. Patel-Schneider, Small can be beautiful in knowledge representation, in: *Proceedings of the IEEE Workshop on Principles of Knowledge-Based Systems*, Denver, Colo. (1984) 11–16.

[24] P. F. Patel-Schneider, Undecidability of subsumption in NIKL, *Artificial Intelligence* 39(2) (1989) 263–272.

[25] M. S. Paterson and M. N. Wegman, Linear unification, *Journal of Computer and System Sciences* 16 (1978) 158–167.

[26] V. Pigman, KRYPTON: description of an implementation, vol. 1, AI Technical Report 40, Schlumberger Palo Alto Research, Palo Alto, Cal. (1984).

[27] K. Schild, Undecidability of $\mathcal{U}$, KIT Report 67, Department of Computer Science, Technische Universität Berlin, Berlin, West Germany (1988).

[28] M. Schmidt-Schauß, Subsumption in KL-ONE is undecidable, in: R. J. Brachman, H. J. Levesque, and R. Reiter, eds., *Proceedings of the 1st International Conference on Principles of Knowledge Representation and Reasoning*, Toronto, Ont. (1989) 421–431.

[29] M. Schmidt-Schauß and G. Smolka, Attributive concept descriptions with unions and complements, *Artificial Intelligence* (1989). To appear. Also available as IWBS Report 68, IBM Germany Scientific Center, IWBS, Stuttgart, West Germany, June 1989.

[30] K. von Luck, B. Nebel, C. Peltason, and A. Schmiedel, The anatomy of the BACK system, KIT Report 41, Department of Computer Science, Technische Universität Berlin, Berlin, West Germany (1987).