Published in *Artificial Intelligence* **34**: 371–383, 1988

Computational Complexity of Terminological Reasoning in BACK*

Bernhard Nebel
Technische Universität Berlin, CIS/KIT
Sekr. FR 5-8, Franklinstraße 28/29
D-1000 Berlin 10, West-Germany
e-mail: nebel@db0tui11.bitnet

Abstract

Terminological reasoning is a mode of reasoning all hybrid knowledge representation systems based on KL-ONE rely on. After a short introduction of what terminological reasoning amounts to, it is proven that a complete inference algorithm for the BACK system would be computationally intractable. Interestingly, this result also applies to the KANDOR system, which had been conjectured to realize complete terminological inferences with a tractable algorithm. More generally, together with an earlier paper of Brachman and Levesque it shows that terminological reasoning is intractable for any system using a non-trivial description language. Finally, consequences of this distressing result are briefly discussed.

1 Introduction

The BACK system[1] [13] belongs to the class of hybrid knowledge representation systems based on KL-ONE (cf. the article by Brachman and Schmolze [4]). As in any other system of this family, a *frame-based description language* (henceforth FDL), which can be viewed as a linear representation of *structural inheritance networks* as introduced by Brachman [2], is employed to represent *terminological* knowledge—knowledge about the terminology used to describe the world. A FDL allows the introduction of *concepts*[2] and *roles*[3] by specifying relationships to other

*This work was partially supported by the EEC and is part of the ESPRIT project 311, which involves the following participants: Nixdorf, Olivetti, Bull, Technische Universität Berlin, Universita di Bologna, Universität Hildesheim and Universita di Torino.

[1] The **B**erlin **A**dvanced **C**omputational **K**nowledge Representation System.

[2] I use the term *concept* here and in the sequel following the BACK terminology for what is called *generic concept* in KL-ONE and *frame* in [3].

[3] *Roles* correspond to *slots* in the frame terminology.

concepts, as in the following example:

> a man is (among other things)
> a human and a male-being
> a parent is (exactly)
> a human with at least one offspring
> a father is (exactly)
> a parent and a man
> a grandparent is (exactly)
> a human with at least one offspring which is a parent

Although there is a broad diversity of FDLs in different hybrid systems (e.g., KL-TWO [21], KRYPTON [5], KANDOR [16], MESON [7]), they are nevertheless very similar to each other. Despite superficial differences in the concrete syntax it is easy to identify the principal concept-forming operators. One important characteristic of these languages is that they take the notion of *definition* seriously[4]. This means that not only relationships between concepts that are *explicitly* given, such as the one between human and man in the example above, are considered to be important, but also the relationships which are *implicitly* present. For instance, grandparent is a specialization of parent, although this is not explicitly mentioned. If the set of objects described by these expressions is analyzed, it becomes obvious that all objects which could be called grandparents are *necessarily* parents as well, and therefore the former concept should be considered as a specialization of the latter.

Based on the observation that there is more represented than explicitly written down, it is obvious that we need some kind of reasoner which uncovers the hidden relationships. Of course, we will not get out more than we put in, i.e., the reasoning process will only give us answers to sensible queries, which in the context of a terminology can only be of *analytic* nature. For instance, a query whether there *exist* fathers is really off the track, because it refers to the world, not to the terminology! Sensible queries for a terminological reasoner can be classified as follows:

Subsumption Does $concept_1$ subsume $concept_2$, i.e., is the former a more general concept than the latter?

Classification Given a set of introduced concepts, what are the immediate subsumers and subsumees of a new concept?

Disjointness Are two concepts disjoint, i.e., is it impossible to describe any object by both concepts simultaneously?

Incoherency Is a concept incoherent, i.e., is it impossible to describe any object with this concept?

[4] For this reason *exceptions* and *procedural attachment* are not part of any FDL.

Property possession What properties does a certain concept possess, e.g., what are the restrictions on role-fillers? In traditional semantic network formalisms this is usually referred to as *inheritance*.

Some of these queries can be reduced to other query types, so that it is possible to specify a *minimal interface* for an *ideal system*, which is inevitable if a formal specification for the system is to be given and if the complexity of the necessary inference algorithms is to be analyzed. In our case, all the above query types can be reduced to *subsumption*, provided that we have access to the set of introduced concepts and roles.

Classification can be reduced to subsumption by determining for a given concept the subsumer and subsumee sets from the set of introduced concepts followed by filtering out all those concepts for which an intermediate concept can be found. Altogether, this process requires $O(n^2)$ subsume-operations, where n is the number of introduced concepts. Disjointness can be reduced to incoherency by querying whether the conjunction of the two concepts under investigation is incoherent. Incoherency in turn can be reduced to subsumption by querying whether a known incoherent concept subsumes the given concept. If it does, we know that the given concept must be an incoherent one as well[5]. Finally, property possession can be answered by a technique similar to the one used in the classification case.

Of course, in a real system, all the above query types would be included in the system interface for reasons of user convenience and efficiency. For instance, classification is an inference heavily used if a terminological reasoner is employed in a natural language generation system (cf. the work of Sondheimer and Nebel [20]). And because almost all terminological reasoners maintain an explicit hierarchy of introduced concepts—which is just the 'compiled' classification inferences—it is a natural consequence to provide classification as a service of the terminological reasoner. However, there seems to be some confusion whether classification is merely an implementation technique (a point of view taken by Brachman et al [6] and Patel-Schneider [16]) or an inference. The original formulation of Lipkis [12] seemed to go for the former, but for the reasons spelled out above, I would opt for both.

2 Complexity of Subsumption

As shown above, subsumption is the crucial point in terminological reasoning. If we are able to specify a good algorithm for this inference, we can perform all other

[5]Although this sounds strange, it is granted by both the intuitive and the formal semantics we will specify below. Furthermore, it reflects the fact that subsumption and incoherency detection are inherently intertwined.

inferences easily—in time polynomially proportional to the size of the problem description[6]. With clever implementation techniques we can even do better.

The first informal treatment of subsumption by Lipkis [12] led to a running system but left open the question of what is really done, i.e., what we know if the system detects that one concept subsumes another or that it does not—a short-coming of almost all knowledge representation systems in those days as D. McDermott noted [15]. The intuitive idea behind subsumption, however, was very clear, namely that

$$\begin{gathered} \mathit{concept}_1 \text{ subsumes } \mathit{concept}_2 \\ \Updownarrow \\ \text{all objects which are a } \mathit{concept}_2 \text{ are also a } \mathit{concept}_1 \end{gathered}$$

When this idea was first formalized by specifying a formal semantics for (a subset of) KL-ONE by Schmolze and Israel [19], it was discovered that the subsumption procedure implemented in KL-ONE was *sound*, i.e., every detected subsumption relationship was correct with respect to the semantics, but *incomplete*—some relationships were not detected by the procedure. This fact could have been taken as a starting point to develop a complete algorithm, but there are computational problems. In [3] Brachman and Levesque showed that even for a very small subset of the FDL used in KL-ONE the subsumption problem is *intractable*. More precisely, it was shown that subsumption in that particular FDL is at least as hard as the problem of determining the unsatisfiability of boolean formulas in conjunctive normal form, which is a *co-NP-complete* problem, a complementary problem to a *NP-complete* problem. The NP-complete problems, as well as the co-NP-complete ones, are strongly believed not to be solvable in time polynomially proportional to the size of the problem description (cf. [8]).

One way out of this distressing situation could be to investigate FDLs with different concept-forming operators that would allow for a complete and tractable subsumption algorithm. And this was indeed a program which was proposed by Brachman and Levesque in [3] in order to find the boundary between tractable and intractable FDLs. We will pursue this line of investigation by analyzing the FDL used in BACK. However, before we go into the details of analyzing computational complexity of subsumption, we show how the subsumption problem can be formalized, following the lines of Brachman and Levesque [3].

3 A Formal Treatment of Subsumption

In order to formalize subsumption, we first need to formalize the language under investigation. A FDL which suffices to capture all the operators used to formulate the example in the introduction can be described by BNF notation as follows:

[6] Levesque demanded in [11] that any knowledge representation system should have this property.

$$
\begin{array}{rcl}
\langle concept \rangle & ::= & \langle atom \rangle \mid \\
 & & (\textbf{and}\ \langle concept \rangle^{+}) \mid \\
 & & (\textbf{all}\ \langle role \rangle\ \langle concept \rangle) \mid \\
 & & (\textbf{some}\ \langle role \rangle) \\
\langle role \rangle & ::= & \langle atom \rangle \mid \\
 & & (\textbf{restr}\ \langle role \rangle\ \langle concept \rangle)
\end{array}
$$

This syntax does not capture the fact that concepts can be defined, but only that descriptions can be constructed by concept-forming operators. This, however, will suffice for investigating subsumption. We simply assume that names will be substituted by the expressions that define them. The introduction of partially defined concepts can be modelled by assuming additional anonymous atomic concepts. Man and grandparent, for example, could be described in the following way:

$$
\begin{array}{ll}
\mathsf{man} & \equiv (\textbf{and}\ \mathsf{human}\ \mathsf{male\text{-}being}\ \mathsf{Cprim}_1) \\
\mathsf{grandparent} & \equiv (\textbf{and}\ \mathsf{human} \\
 & \qquad\qquad (\textbf{some}\ (\textbf{restr}\ \mathsf{offspring} \\
 & \qquad\qquad\qquad\qquad (\textbf{and}\ \mathsf{human}\ (\textbf{some}\ \mathsf{offspring})))))
\end{array}
$$

The next step in formalizing the subsumption problem should be the specification of a formal semantics for this language. In following the informal intuitive definition of subsumption given in the last section we assign to each concept an *extension*, the objects described by that particular concept. Obviously, the extensions of different concepts are not independent, e.g., the extension of man has to be a subset of the extension of human regardless of the set of objects we are describing. These necessary condition on extensions of concepts can be formally described as follows:

Let $\mathcal{D}$ be any set of objects and $\mathcal{E}$ be any function from concepts to $\mathcal{D}$ and from roles to $\mathcal{D} \times \mathcal{D}$. $\mathcal{E}$ is called an *extension function* over $\mathcal{D}$ if and only if

$$
\begin{array}{rcl}
\mathcal{E}[(\textbf{and}\ C_1\ \ldots\ C_n)] & = & \{x \in \mathcal{D} | x \in \mathcal{E}[C_1] \wedge \ldots \wedge x \in \mathcal{E}[C_n]\} \\
\mathcal{E}[(\textbf{all}\ R\ C)] & = & \{x \in \mathcal{D} | \forall y : \langle x, y \rangle \in \mathcal{E}[R] \Rightarrow y \in \mathcal{E}[C]\} \\
\mathcal{E}[(\textbf{some}\ R)] & = & \{x \in \mathcal{D} | \exists y : \langle x, y \rangle \in \mathcal{E}[R]\} \\
\mathcal{E}[(\textbf{restr}\ R\ C)] & = & \{\langle x, y \rangle \in \mathcal{D} \times \mathcal{D} | \langle x, y \rangle \in \mathcal{E}[R] \wedge y \in \mathcal{E}[C]\}
\end{array}
$$

Now we are in position to say what subsumption means referring only to the formal notion of extension: We say that a concept C_1 subsumes a concept C_2 if and only if for any set $\mathcal{D}$ and any extension function $\mathcal{E}$ over $\mathcal{D}$ the following holds:

$$\forall d : d \in \mathcal{E}[C_2] \Rightarrow d \in \mathcal{E}[C_1]$$

The language described above was called $\mathcal{FL}$ by Brachman and Levesque [3] and proved to be intractable with respect to (complete) subsumption. A slightly more restrictive language, called $\mathcal{FL}^-$, without the **restr** operator, was shown to be acceptable from the perspective of computational complexity. Subsumption in this language can be computed with an $O(n^2)$ algorithm, n being the sum of the lengths of the two descriptions. Fortunately, it is possible to extend the expressiveness of $\mathcal{FL}^-$ without loosing tractability. For example, the generalization of the **some** operator to (**atleast** $\langle number \rangle$ $\langle role \rangle$), stating that there must be at least $\langle number \rangle$ different instances as role-fillers, does not present a problem. Going one step further, a complementary **atmost** operator might be added. And even this does not seem to endanger the tractability characteristic of the language. Alternatively to **atleast** and **atmost**, an **androle** operator may be added, which allows the creation of new roles by conjoining them, without endangering tractability[7].

At this point, the question might arise whether the simultaneous addition of **atleast**, **atmost** and **androle** would present any problem. It does indeed lead to problems. As we will see below, such a language also falls off the computational cliff, even for a restricted version of the **androle** operator. This proves to be rather important, because this FDL forms a subset of the FDL used in BACK.

4 Some Problems of Subsumption in BACK

As remarked above, a subset of the FDL used in BACK can be described by extending $\mathcal{FL}^-$ with the concept-forming operators **atleast**, **atmost** and the role-forming operator **androle**. The latter may even come in a restricted version: Only two arguments are permitted, and the second argument appears only in other **androle** expressions with the same first argument. This amounts to the introduction of *primitive subroles* or *primitive role differentiation*, as it is called in KL-ONE. The syntax of this language, which we will call $\mathcal{FL}^N$, can be given as follows:

---

[7]That all these additions preserve the tractability is left as an excercise to the interested reader.

$$
\begin{array}{rcl}
\langle concept \rangle & ::= & \langle atom \rangle \mid \\
 & & (\textbf{and}\ \langle concept \rangle^{+}) \mid \\
 & & (\textbf{all}\ \langle role \rangle\ \langle concept \rangle) \mid \\
 & & (\textbf{atleast}\ \langle number \rangle\ \langle role \rangle) \mid \\
 & & (\textbf{atmost}\ \langle number \rangle\ \langle role \rangle) \mid \\
\langle role \rangle & ::= & \langle atom \rangle \mid \\
 & & (\textbf{androle}\ \langle role \rangle\ \langle \textit{restricted-usage-role} \rangle)
\end{array}
$$

The additional semantics is the following (we only specify the additions to $\mathcal{FL}^-$):

$$
\begin{array}{rcl}
\mathcal{E}[(\textbf{atleast}\ N\ R)] & = & \{x \in \mathcal{D} |\ \|\{y \in \mathcal{D} | \langle x, y\rangle \in \mathcal{E}[R]\}\| \geq N\} \\
\mathcal{E}[(\textbf{atmost}\ N\ R)] & = & \{x \in \mathcal{D} |\ \|\{y \in \mathcal{D} | \langle x, y\rangle \in \mathcal{E}[R]\}\| \leq N\} \\
\mathcal{E}[(\textbf{androle}\ R\ P)] & = & \{\langle x, y\rangle \in \mathcal{D} \times \mathcal{D} | \langle x, y\rangle \in \mathcal{E}[R] \wedge \langle x, y\rangle \in \mathcal{E}[P]\}
\end{array}
$$

One obvious property of this FDL is that it is now possible to describe incoherent concepts, which was impossible with $\mathcal{FL}^-$. For instance,

(**and** (**atmost** 1 R) (**atleast** 2 R))

is an incoherent concept—the extension of this concept is necessarily empty. A second look reveals that the *actual* **atleast** restrictions depend on the disjointness of the concepts used in **all** expressions for subroles. This is illustrated by the following description:

(**and** (**atleast** 2 R)
    (**atleast** 2 (**androle** R Rprim_1))
    (**atleast** 1 (**androle** R Rprim_2))
    (**all** (**androle** R Rprim_1) (**atleast** 4 P))
    (**all** (**androle** R Rprim_2) (**atmost** 3 P)))

Although it was specified that R has at least two role-fillers, a stronger condition can be inferred from the description, namely that at least three distinct role-fillers are needed, because the fillers for the two subroles have to be necessarily distinct. That means that a complete subsumption algorithm has to take the disjointness of restrictions on subroles into account, otherwise it would miss that (**atleast** 3 R) subsumes the description above.

We therefore have to account for pairs of disjoint role-filler concepts of subroles (if we are going for a complete algorithm). This still seems to be managable in polynomial time, because there are 'only' $n \times (n-1)/2$ different pairs (with n being the number of subroles).

Taking a third look at the problem, however, we detect that there are even more complex cases, exemplified by the three descriptions below:

(**and** (**all** (**androle** R Rprim_1) (**atleast** 4 P))
    (**atleast** 1 (**androle** R Rprim_1)))
(**and** (**all** (**androle** R Rprim_2) (**atmost** 3 P))
    (**atleast** 1 (**androle** R Rprim_2)))
(**atmost** 1 R)

These descriptions are not pairwise disjoint; the conjunction of the three descriptions, however, leads to an incoherent concept. Assuming that these descriptions serve as arguments to **all** restrictions of subroles, the computation of the actual **atleast** restrictions for the superrole becomes even more complicated. We can regard this as a optimization problem: In the general case, the subsets of subroles leading to incoherent **all** restrictions have to be determined and then the **atleast** restriction for the superrole has to be computed by a minimization process. All this sounds very complicated and, in particular, the determination of the subsets of subroles leading to incoherent concepts for role-fillers sounds akward and is probably intractable. However, even if we assume that the subsets can be identified in reasonable time, there is still the minimization problem, which is intractable in a strong sense, as will be shown below.

5 Proof of Strong Co-NP-hardness

In order to show that subsumption in $\mathcal{FL}^N$ is co-NP-hard, the complement of a known NP-complete problem is transformed to a special-case subsumption problem, namely

SUBSUMES((**atleast** 3 R),X)

with X a description containing a set of **atleast** and **all** operators on subroles of R. The transformation is performed in way such that a solution to the special-case subsumption problem applies also to the co-NP-complete problem.

A natural candidate for the proof is the problem of SET SPLITTING, also known as HYPERGRAPH-2-COLORABILITY (cf. [8, p. 221]), which was proved to be NP-complete by Lovasz [10]. The formal description of that problem is:

> Given a collection C of subsets of a finite set S, is there a partition of S into two subsets S_1 and S_2 such that no subset in C is entirely contained in either S_1 or S_2?

A transformation from an instance of this problem to the description X with the desired property can be specified as follows. Given an instance of SET SPLITTING with $S = \{s_1, s_2, \ldots, s_n\}$ and $C = \{C_1, C_2, \ldots, C_m\}$ with each C_i having the form $C_i = \{s_{f(i,1)}, s_{f(i,2)}, \ldots, s_{f(i,\|C_i\|)}\}$ and letting

$$g(i,j) = \begin{cases} k & \text{if } s_j \in C_i \text{ and } f(i,k) = j \\ 0 & \text{otherwise} \end{cases}$$

then X has the form:

$$\begin{array}{l}
(\textbf{and}\ (\textbf{atleast}\ 1\ (\textbf{androle}\ \mathrm{R}\ \mathrm{Rprim}_1)) \\
\qquad (\textbf{all}\ (\textbf{androle}\ \mathrm{R}\ \mathrm{Rprim}_1)\ \pi(s_1)) \\
\qquad (\textbf{atleast}\ 1\ (\textbf{androle}\ \mathrm{R}\ \mathrm{Rprim}_2)) \\
\qquad (\textbf{all}\ (\textbf{androle}\ \mathrm{R}\ \mathrm{Rprim}_2)\ \pi(s_2)) \\
\qquad \qquad \vdots \\
\qquad (\textbf{atleast}\ 1\ (\textbf{androle}\ \mathrm{R}\ \mathrm{Rprim}_n)) \\
\qquad (\textbf{all}\ (\textbf{androle}\ \mathrm{R}\ \mathrm{Rprim}_n)\ \pi(s_n)))
\end{array}$$

The transformation function π is now specified in such a way that for each set C_i, the conjunction of $\pi(s_{f(i,k)})$, $1 \leq k \leq \|C_i\|$, forms an incoherent concept. This means the corresponding subroles cannot be filled with the same instance. On the other hand, each subset of the subroles with the property that the corresponding subset of S does not contain a set C_i can have the same role-filler. For this purpose, we assume m different roles R_i corresponding to the sets C_i:

$$\begin{aligned}\pi(s_j) = (\textbf{and}\ &(\textbf{atmost}\ \|C_1\|-1\ \mathrm{R}_1)\\ &(\textbf{atleast}\ 1\ (\textbf{androle}\ \mathrm{R}_1\ \mathrm{Rprim}_{1,g(1,j)}))\\ &(\textbf{all}\ (\textbf{androle}\ \mathrm{R}_1\ \mathrm{Rprim}_{1,g(1,j)})\ CP_{1,g(1,j)})\\ &\vdots\\ &(\textbf{atmost}\ \|C_m\|-1\ \mathrm{R}_m)\\ &(\textbf{atleast}\ 1\ (\textbf{androle}\ \mathrm{R}_m\ \mathrm{Rprim}_{m,g(m,j)}))\\ &(\textbf{all}\ (\textbf{androle}\ \mathrm{R}_m\ \mathrm{Rprim}_{m,g(m,j)})\ CP_{m,g(m,j)})\end{aligned}$$

Now the $CP_{i,j}$ are specified such that the conjunctions of $CP_{i,j}$ and $CP_{i,k}$ for all pairs of different j and k, $j \neq 0$, $k \neq 0$, are incoherent:

$$\begin{aligned}&\mathrm{CP}_{i,0} \equiv (\textbf{atleast}\ 0\ \mathrm{RCP}_i)\\ &\mathrm{CP}_{i,k} \equiv (\textbf{and}\ (\textbf{atleast}\ \mathrm{k}\ \mathrm{RCP}_i)\ (\textbf{atmost}\ \mathrm{k}\ \mathrm{RCP}_i))\ 1 \leq k \leq \|C_i\|\end{aligned}$$

This means that a conjunction of $\pi(s_j)$ is incoherent if and only if for some role R_i we have more than $\|C_i\| - 1$ different **atleast** restrictions on subroles of R_i.

The entire construction, which obviously can be performed in time polynomially proportional to the length of the original problem description, leads to the following result: If role R of concept X can be filled with two (or less) role-fillers, then there is a set splitting. On the other hand, if more than two role-fillers are necessary, then there cannot be a set splitting. This means that the special subsumption problem given above can be used to solve the complement of the SET SPLITTING problem, and thus subsumption in $\mathcal{FL}^N$ is co-NP-hard[8].

When a problem involving numbers (in our case the **atleast** and **atmost** restrictions) is proved to be (co-)NP-hard, there might still be the possibility that the problem is tractable in a weak sense—solvable by an algorithm with *pseudo-polynomial* complexity (cf. [8, pp. 91–92]). A problem has pseudo-polynomial complexity if it can be solved in time polynomially proportional to the numbers appearing in the problem description. The well-known KNAPSACK problem, for instance, has this property. In our case, however, even this possibility of weak tractability can be ruled out, because in the transformation, all numbers are bounded by the length of the problem description of the original problem (the cardinalities of the C_is). This leads to the following theorem:

Theorem 1 *Subsumption in $\mathcal{FL}^N$ is co-NP-hard in the strong sense.*

In analyzing the transformation, we may note that not the full expressive power of $\mathcal{FL}^N$ was used. For atomic roles, only **atleast** and **atmost** were needed. For subroles, only the **atleast** and **all** operators were used, and only for describing that the superroles are filled with at least a certain number of role-fillers of a particular concept. Therefore, the result does not only apply to $\mathcal{FL}^N$, but to all languages which can express those relationships, which leads to the next theorem:

[8]It is not obvious whether the problem is in co-NP or not.

Theorem 2 *Subsumption is co-NP-hard in the strong sense for any FDL with the expressive power of $\mathcal{FL}^-$ extended by* **atleast**, **atmost** *and the possibility to express that there are at least a certain number of role-fillers of a certain concept.*

In particular, the FDL used in KANDOR can be characterized in this sense, because it contains a special three-argument **atleast** operator with the meaning that there are at least a specified number of role-fillers for the given role of a particular concept. Thus, because of the arguments above, the conjecture of tractability for KANDOR by Patel-Schneider in [16, p. 16] does not hold, even not in the weak sense of [17, p. 345][9].

6 Consequences of this Result

The proof of strong NP-hardness for $\mathcal{FL}^N$ and similar FDLs, together with the result of Brachman and Levesque in [3] for $\mathcal{FL}$, shows that any FDL with reasonable expressive power implies the intractability of complete subsumption. However, although this sounds rather disturbing, FDLs are undoubtedly a very useful class of knowledge representation formalisms. Additionally, we know that almost all representation formalisms used in Artficial Intelligence are intractable or even undecidable. Therefore in practical systems tractable but incomplete algorithms are often used, as for example, in the terminological component of KL-TWO [9], in the reasoning maintenance system RUP [14], and in Allen's temporal reasoner [1].[10]

If, however, completeness is a goal one cannot dispense with, expressive power has to be severely restricted. In our case, one solution would be to sacrifice all operators that state relationships between roles, i.e., primitive subrole introduction and role-value-maps (another popular concept-forming operator). Alternatively, instead of general number restrictions, a limited set of restricted operators could be used, e.g., **some**, **none** and **unique**[11].

Another way out of this dilemma, pursued by Patel-Schneider in [17] and [18], could be to use a different semantics based on a four-valued logic, for which a complete and tractable subsumption algorithm even for very expressive FDLs can be specified. Another view of this solution is that it provides a sound algorithm for standard semantics and gives a precise account—a model theoretic one—for where incompleteness with respect to standard semantics arises. This meets all the demands for a representation formalism McDermott spelled out in [15]. However, this solution has, because of the weak semantics, the disadvantage that a lot of inferences cannot be drawn even though they might be 'obvious'. These

[9] And in fact, the KANDOR system fails to correctly determine subsumption confronted with concepts similar to the one used in the proof.

[10] In BACK a tractable, but incomplete, algorithm is used for terminlogical reasoning as well.

[11] Actually, this would prevent situations similar to the one used in the proof above. However, I am not 100% confident that it would really preserve tractability.

missed inferences are of the 'non-structural' kind, involving reasoning similar to *tertium non datur* and *modus ponens*.

We are thus confronted with a tradeoff between weak semantics with a complete subsumption algorithm, which misses a lot of inferences we intuitively would take for granted, and, on the other hand, strong semantics and an incomplete algorithm, which might miss inferences we never expected but which are implied by the semantics. From a pragmatic point of view it sometimes seems more worthwhile to choose the latter alternative, for example in natural language generation [20], because even though we might miss an inference granted by the semantics—which seems not be very likely in the normal case—it would not result in a disaster. The same seems to be true for other applications as well. The inferences which are computed can then only be characterized by an axiomatic or procedural account.

In conclusion, it is, of course, an unsatisfying (and surprising) state of affairs that the deductive power of a mechanized (i.e., tractable) reasoner cannot be described cleanly, by means of model theoretic semantics, without either tolerating incompleteness or ignoring some intuitively 'obvious' inferences. Nevertheless, model theoretic semantics is an invaluable analytic tool in testing our intuitions, as was shown in this paper.

Acknowledgement

Many of the ideas presented here are the result of the work in the KIT-BACK project. The other members of this project, Kai von Luck, Christof Peltason and Albrecht Schmiedel, deserve much of the credit for these ideas. Additionally, I wish to thank Peter Patel-Schneider, who sent me his thesis draft and the KANDOR system, answered all my questions patiently, and gave me some valuable hints. Kai, Christof, Albrecht, Peter, Stefan Wrobel and an anonymous referee were also helpful in criticizing early drafts of this paper.

References

[1] Allen, J. F., Maintaining Knowledge About Temporal Intervals, *Communications of the ACM* **26**(11) (1983) 832–843.

[2] Brachman, R. J., On the Epistomological Status of Semantic Networks, in: Findler, N. V. (Ed.), *Associative Networks: Representation and Use of Knowledge by Computers* (Academic Press, New York, 1979) 3–50.

[3] Brachman, R. J. and Levesque, H. J., The tractability of subsumption in frame-based description languages, in: *Proc. AAAI-84*, Austin, Tex. (1984) 34–37.

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

[5] Brachman, R. J., Pigman Gilbert, V. and Levesque, H. J., An Essential Hybrid Reasoning System: Knowledge and Symbol Level Accounts in KRYPTON, in: *Proc. 9th IJCAI*, Los Angeles, Cal. (1985) 532–539.

[6] Brachman, R. J., Fikes, R. E. and Levesque, H. J., KRYPTON: A Functional Approach to Knowledge Representation, in: Brachman, R. J. and Levesque, H. J. (Eds.), *Readings in Knowledge Representation* (Morgan Kaufmann, Los Altos, Cal., 1985) 411–430; a revised version of an article published in *IEEE Computer* **16**(10) (1983) 67–73.

[7] Edelmann, J. and Owsnicki, B., Data Models in Knowledge Representation Systems: A Case Study, in: Rollinger, C.-R. and Horn W. (Eds.), *GWAI-86 und 2. Österreichische Artificial-Intelligence-Tagung* (Springer, Berlin, F.R.G., 1986) 69–74.

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

[9] Kaczmarek, T., Bates, R., Robins, G., Recent Developments in NIKL, in: *Proc. AAAI-86*, Philadelphia, Pa. (1986) 978–987.

[10] Lovasz, L., Coverings and colorings of hypergraphs, in: *Proc. 4th Southeastern Conf. on Combinatorics, Graph Theory, and Computing* (Utilitas Mathematics Publishing, Winnipeg, Canada, 1973) 3–12.

[11] Levesque, H. J., Making Believers out of Computers, Computers and Thougth Lecture at IJCAI-85, Los Angeles, Cal., published in: *Artificial Intelligence* **30**(1) (1986) 81-108.

[12] Lipkis, T., A KL-ONE Classifier, in: Schmolze, J. G. and Brachman, R. J. (Eds.), *Proc. 1981 KL-ONE Workshop*, BBN Technical Report 4842, Bolt, Berank and Newman, Inc., Cambridge, Mass., 1982, 128–145.

[13] Luck, K. von, Nebel, B., Peltason, C. and Schmiedel, A., *The Anatomy of the BACK System*, KIT Report 41, Fachbereich Informatik, Technische Universität Berlin, Berlin, West-Germany, 1987.

[14] McAllester, D. A., *Reasoning Utility Package User's Manual*, AI Memo 667, AI Lab., MIT, Cambridge, Mass., 1982.

[15] McDermott, D., Tarskian semantics, or no notation without denotation!, in: *Cognitive Science* **2**(3) (1978) 277–282.

[16] Patel-Schneider, P. F., Small can be Beautiful in Knowledge Representation, in: *Proc. IEEE Workshop on Principles of Knowledge-Based Systems*, Denver, Colo. (1984) 11–16.

[17] Patel-Schneider, P. F., A Four-Valued Semantics for Frame-Based Description Languages, in: *Proc. AAAI-86*, Philadelphia, Pa. (1986) 344–348.

[18] Patel-Schneider, P. F., *Decidable, Logic-Based Knowledge Representation*, Ph.D. Thesis Draft, 1986.

[19] Schmolze, J. G. and Israel, D. J., KL-ONE: Semantics and Classification, in: Sidner, C. L., *Research in Knowledge Representation and Natural Language Understanding*, BBN Technical Report 5421, Bolt, Beranek and Newman, Inc., Cambridge, Mass., 1983, 27–39.

[20] Sondheimer, N. K. and Nebel, B., A Logical-Form and Knowledge-Base Design for Natural Language Generation, in: *Proc. AAAI-86*, Philadelphia, Pa. (1986) 612–618.

[21] Vilain, M. B., The Restricted Language Architecture of a Hybrid Representation System, in: *Proc. 9th IJCAI*, Los Angeles, Cal. (1985) 547–551.