Published in F. Gardin, G. Mauri, and M. G. Filippini (eds.), *Computational Intelligence, II. Proceedings of the International Symposium*, p. 217–228, Milan, Italy, September 1989. North-Holland.

What is Hybrid in Hybrid Representation and Reasoning Systems?

Bernhard Nebel
IBM Germany Scientific Center, IWBS
Stuttgart, F.R. Germany

Abstract

Hybrid knowledge representation and reasoning systems have received a lot of interest in recent years. In this paper, we will give a brief survey of the principles and ideas behind these systems and will then focus on one particular kind of such systems, namely, TBox/ABox systems. A formal analysis of these systems clarifies some of the subtleties and problems involved. Finally, the computational problems are investigated, which leads to two conclusions. First, worst-case tractability cannot be achieved in TBox/ABox systems. Second, even if some system-component uses only incomplete reasoning methods, it is nevertheless possible to achieve completeness in the overall system in certain special cases.

1 Introduction

Hybrid knowledge representation and reasoning systems (KRR systems) are systems that use more than one representation formalism and/or more than one reasoning component. Such a design addresses two important issues in current research on knowledge representation and reasoning, namely, *expressive power* and *efficiency of reasoning*, in the following way:

- The *expressiveness* is improved by either allowing for alternative representation of the same information by different formalisms for the sake of notational convenience or by combining different restricted formalisms to make up a more powerful formalism.

- The *efficiency of reasoning* is enhanced by employing different specialized reasoners in order to cut down the search space.

Of course, KRR systems are not the only systems which could be hybrid. Almost all AI systems use more than one representation formalism and more than one reasoning subsystem, for instance, the expert systems described in [5, 33] and almost all natural language systems (see e.g. [19]). However, in this case the issues are different. In expert systems and natural language systems, the representation formalisms and reasoning components are usually only loosely connected and the integration is a matter of software engineering and driven by the domain and task the system is built for. In KRR systems, on the other hand, the integration has to based on a domain-independent, formally firm base—a point, we will try to highlight in this paper.

In the following section, some examples of hybrid KRR systems are briefly sketched, their main characteristics are described and problems and benefits are discussed. Section 3 gives then a detailed formal description of one particular class of hybrid KRR systems—TBox/ABox systems—highlighting some of the critical properties that make them useful. Finally, in Section 4 the computational problems of TBox/ABox systems are analyzed leading to two important conclusions. First, worst-case tractability cannot be achieved in TBox/ABox systems. Second, even if the TBox reasoner is incomplete, it is possible to characterize an important special-case that makes the overall reasoner complete.

2 Hybrid Representation and Reasoning Systems

There are a number of examples of hybrid KRR systems described in the literature. First, there is a group of logic-based hybrid systems, for instance, the CAKE system [52], the family of TBox/ABox systems (e.g. KL-TWO [60], KRYPTON [9], KANDOR [48], BACK [61], MESON [14], LOOM [29], and CLASSIC [6]), theorem provers using sorted deduction [10, 17] or theory resolution [58], and constraint logic programming [22].

Second, we have a group of hybrid systems which integrate logic with other formalisms [16, 51]. In the following, we will concentrate on the former class.

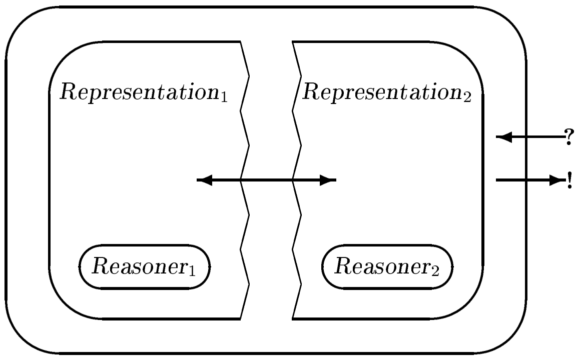

Figure 1: Architecture of hybrid KRR systems

In all cases, the principal architecture of the systems can be depicted as in Figure 1. However, there might be the case that there is only one representation, that is, different reasoners are operating on just one representation formalism, or that there is only one reasoner implying that one formalism is translated into the other before reasoning takes place (as e.g. in the system described in [47]). Actually, it is impossible to tell from the behavior of the system how many reasoners or formalisms are actually employed. This means that hybridness is not a *knowledge-level* property [26, 44] but a symbol-level property.

The most interesting issue in such systems is how the border line between the different reasoners and formalisms is drawn, and how the gap between them is bridged. In order to give an impression about the issues involved, some of the systems mentioned above will be briefly sketched.

2.1 CAKE

CAKE is a the representation kernel of a programmer's apprentice system [52]. One component is the *plan calculus* component, which reasons about programs represented by using a flow-chart style formalism with pre- and postconditions. The second component is RUP [32]—a TMS that supports reasoning in variable-free predicate logic with equality.

The link between these two formalisms is established by a sort of translation between the two formalisms. Plans and programs used in the plan calculus component are mapped to assertions that are stored in the active database provided by the RUP system. RUP in turn reports about detected inconsistencies, equalities etc. to the plan-calculus component.

Thus, in the case of CAKE we have two reasoners acting on the same information which is represented in different ways. The benefit of this system architecture is that domain-dependent reasoning is efficiently carried out in the plan calculus component, without overloading this component by reinventing the logical wheel.

2.2 The Family of TBox/ABox Systems

In TBox/ABox systems, two epistemological different kinds of knowledge are distinguished, namely, *terminological* and *assertional* knowledge [7], which is dealt with in different "boxes"—the former kind of knowledge in the TBox, the latter in the ABox. Terminological knowledge is concerned with the definitional meaning of *concepts*, e.g. a parent is defined as a person who has a child, while assertional knowledge is about the state of the world, e.g. that peter is a parent with the child mary.

Actually, we may further divide this class of systems into two subclasses, namely those that use full first-order logic in the ABox, such as KRYPTON, and those that use a restricted formalism (variable-free predicate logic or less). In the former case, the distinction between terminological and assertional representation may only be considered as a notational convenience because everything expressible in KRYPTON's TBox can also be asserted in the ABox.[1]

In the latter case—the ABox allows only for a expressively limited formalism—the combination of a TBox and an ABox leads to a powerful system such that the expressiveness of the overall formalism is more than the sum of its parts, as we will see in Section 3.

Parallel to the distinction along the lines of expressive power, one could distinguish the reasoning architecture. In the case when the ABox supports full first-order logic, the technical device to bridge the gap between ABox and TBox is partial theory resolution [9, 58]. In the other case, the combination is much easier to achieve because the ABox has only little influence on the TBox.

2.3 Sorted Logics

In sorted logics, the universe of discourse is divided up into subsets—called *sorts*—and variables are restricted

[1]However, when analyzing possible *belief revision* operations, it turns out that the distinction between TBox and ABox goes deeper. The techniques developed for revising assertional knowledge [35] are inappropriate for terminological knowledge [37, Ch. 7], indicating that the terminological/assertional distinction provides more than notational convenience.

to range over these subsets. Usually, the sorts are ordered in a *sort taxonomy.* Here again, different approaches are possible which have wide-ranging effects on the resulting expressiveness of the overall system and on the communication protocol between the general reasoner and the sort reasoner.

One approach, the so-called substitutional framework [17], uses sort symbols only to restrict variables and arguments of functions and predicates. In this case, the communication protocol is quite simple. The sort reasoner is only called during unification as one inference step.

This approach can be contrasted with a more liberal use of sort literals where sort literal may appear as one-place predicates in the first-order theory [10]. In this case, the communication between the general reasoner and the sort reasoner becomes more complicated because the sort reasoner may have to return results in the form of first-order formulas to the general reasoner.

2.4 Communication and Control

Similar to the tradeoff between expressiveness and tractability [27], in hybrid systems there seems to be a tradeoff between the degree of conceptual overlap between subformalisms and the simplicity of control and communication between the reasoners. In general, we can identify at least two different kinds of communication protocols between subcomponents in a hybrid system:

- A master/slave system, where the slave does one or more inference steps when called by the master. A necessary prerequisite for this protocol is that the subformalisms are conceptually quite independent, as in the case of CAKE, TBox/Abox systems with a expressively limited ABox, sorted logics in the substitutional framework, and constraint logic programming.

- A master/slave system, where the slave delivers its results in terms of the representation used by the master system. This strategy has to be used if there is a conceptual overlap in the representation formalisms, as e.g. in KRYPTON, in sorted logic in the non-substitutional framework, and in partial theory resolution. Here the interaction becomes much more complex and there might even be the potential danger that instead of reducing the search space the representation constructs generated by the slave reasoner enlarge the search space.

Of course, other architectures are conceivable. For instance, reasoners might interact cooperatively or they might be controlled by a metareasoner [21].

2.5 Requirements and Possible Pitfalls

First of all, the design of a hybrid KRR system should be done in a way such that the subformalisms employed are balanced in their expressiveness [42]. For instance, it does not make much sense to have the means to represent cardinalities of sets in a TBox without the expressive power of representing cardinalities in the ABox.

Second, the combination of subformalisms and subsystems should be more than the sum of their parts. This means that the subformalisms should be integrated, preferably by using a unified (formal) semantics or by employing some sort of projection or translation between the different semantics. Furthermore, this integration on the formal level should be mirrored by a tight integration of the reasoners.

In this context, two possible pitfalls should be mentioned. One is the danger of "computational explosion"—caused by an uncontrolled interaction between the reasoners. The other one is unpredictable and unprincipled incompleteness of a hybrid system caused by the interaction of incomplete reasoners—a phenomenon which may be called "computational implosion."

However, when integrating incomplete reasoners in a hybrid system there might also be the chance that the overall system can be characterized as "more complete" then its parts in the sense that it becomes possible to prove conditional completeness results.

3 TBox/ABox Systems

In order to make the points discussed above more concrete, one particular kind of hybrid KRR systems is described and analyzed in detail, namely, TBox/ABox systems as sketched in Section 2.2. However, we will only focus on the formal aspects, i.e., on the semantics of the employed representation formalism, the inferences supported by this semantics and their computational properties.

3.1 Formal Terminologies

The TBox serves as the component which supports the representation of terminologies. In the following, a simple terminological language, called $\mathcal{STL}$, will be introduced.[2]

[2]This is essentially the terminological language employed in the hybrid representation system MESON [14].

Terminologies are composed out of *terminological axioms* (*TA*) which relate a concept (the left hand side) to a concept description (the right hand side):

$$\begin{array}{lllr} TA & \to & A \stackrel{\cdot}{\leq} C & \textit{specialization} \\ & \mid & A \doteq C & \textit{equivalence.} \end{array}$$

with the additional restriction that no concept may occur more than once as a left hand side in a terminology. Giving an example,

$$\begin{array}{rcl} \mathsf{Workstation} & \stackrel{\cdot}{\leq} & \mathsf{Computer} \\ \mathsf{Cluster} & \doteq & \mathsf{Network} \sqcap \exists^{\leq 1}\mathsf{server}. \end{array}$$

is intended to mean that Workstation is a specialization of Computer and Cluster is equivalent to a Network with atleast 1 server.

The right hand sides of terminological axioms—the *concept descriptions* (denoted by C)—are composed out of *concepts* (denoted by A), restrictions on attributes—called *roles* (denoted by R)—and the following *description-forming operators*:

$$\begin{array}{lllr} C & \to & A & \textit{atomic concept} \\ & \mid & C \sqcap C' & \textit{concept conjunction} \\ & \mid & \forall R\colon C & \textit{value restriction} \\ & \mid & \exists^{\geq n} R & \textit{number restriction (min)} \\ & \mid & \exists^{\leq n} R & \textit{number restriction (max).} \end{array}$$

The sets of concepts, roles, and concept descriptions are denoted by Σ_A, Σ_R, and Σ_C, respectively.

In order to specify the meaning of terminologies formally, we define an *interpretation* $\mathcal{I}$ as a pair $\langle D, \cdot^{\mathcal{I}}\rangle$ with D an arbitrary set, the *domain*, and $\cdot^{\mathcal{I}}$, the *interpretation function*, a function from concepts to subsets of D and from roles to subsets of $D \times D$:

$$\cdot^{\mathcal{I}}\colon \left\{ \begin{array}{lll} \Sigma_A & \to & 2^D \\ \Sigma_R & \to & 2^{D\times D} \end{array} \right.$$

Simplifying notation, we will sometimes use an applicative style for referring to the set of elements which are related to a given element x by a role, also often called *role-filler set*:

$$R^{\mathcal{I}}(x) = \{y \in D \mid \langle x, y\rangle \in R^{\mathcal{I}}\}$$

Based on the interpretation of concepts and roles, the denotation of concept descriptions is defined inductively:

$$\begin{array}{rcl} (C \sqcap C')^{\mathcal{I}} & = & C^{\mathcal{I}} \cap C'^{\mathcal{I}} \\ (\forall R\colon C)^{\mathcal{I}} & = & \{x \in D \mid R^{\mathcal{I}}(x) \subseteq C^{\mathcal{I}}\} \\ (\exists^{\geq n} R)^{\mathcal{I}} & = & \{x \in D \mid \|R^{\mathcal{I}}(x)\| \geq n\} \\ (\exists^{\leq n} R)^{\mathcal{I}} & = & \{x \in D \mid \|R^{\mathcal{I}}(x)\| \leq n\}. \end{array}$$

This means that a concept is interpreted as standing for a set of objects—its *extension*—and a concept description receives its meaning by the application of straightforward set operations. For instance, the extension of Network $\sqcap \exists^{\leq 1}$server is simply the set of objects that are Networks that have at least one role-filler of the server role.

The concepts $\top$ and $\bot$ will be used as abbreviations for $\exists^{\geq 0} R$ and $\exists^{\leq 0} R \sqcap \exists^{\geq 1} R$, respectively, where R is any role. Thus, $\top$ is interpreted as the set of everything and $\bot$ is interpreted as the empty set.

An interpretation $\mathcal{I}$ *satisfies a terminological axiom* σ, written $\models_{\mathcal{I}} \sigma$, iff the sets denoted by the right hand and left hand side relate to each other as suggested by the symbols:

$$\begin{array}{lll} \models_{\mathcal{I}} A \doteq C & \text{iff} & A^{\mathcal{I}} = C^{\mathcal{I}} \\ \models_{\mathcal{I}} A \stackrel{\cdot}{\leq} C & \text{iff} & A^{\mathcal{I}} \subseteq C^{\mathcal{I}}. \end{array}$$

Furthermore, an interpretation $\mathcal{I}$ is a *model of a terminology T*, written $\models_{\mathcal{I}} T$, iff all terminological axioms in T are satisfied by $\mathcal{I}$.

In order to make these definitions more vivid, let us examine a small example:

$$\begin{array}{rcl} \mathsf{Workstation} & \stackrel{\cdot}{\leq} & \forall\mathsf{drive}\colon \mathsf{Disk} \\ \mathsf{Diskless\text{-}WS} & \doteq & \mathsf{Workstation} \sqcap \exists^{\leq 0}\mathsf{drive} \\ \mathsf{Diskbased\text{-}WS} & \doteq & \mathsf{Workstation} \sqcap \exists^{\geq 1}\mathsf{drive} \\ \mathsf{Network} & \stackrel{\cdot}{\leq} & \exists^{\geq 1}\mathsf{member} \sqcap \\ & & \forall\mathsf{member}\colon \mathsf{Workstation} \sqcap \\ & & \forall\mathsf{server}\colon \mathsf{Diskbased\text{-}WS} \\ \mathsf{Small\text{-}Network} & \doteq & \mathsf{Network} \sqcap \exists^{\leq 5}\mathsf{member} \\ \mathsf{Cluster} & \doteq & \mathsf{Network} \sqcap \exists^{\leq 4}\mathsf{member} \sqcap \\ & & \exists^{\geq 1}\mathsf{server} \\ \mathsf{Standalone\text{-}System} & \doteq & \mathsf{Network} \sqcap \exists^{\leq 1}\mathsf{member}. \end{array}$$

Having defined the formal meaning of terminologies, we can now say which *terminological formulas* are *entailed* by a terminology. Here, we will permit arbitrary formulas $C \doteq C'$ and $C \stackrel{\cdot}{\leq} C'$. Such a formula τ is *entailed* by a terminology T, written $T \models \tau$, iff τ is satisfied by all models of T. Applying this definition to the example above, it is easy to see that the following formulas are entailed:

$$\begin{array}{rcl} \mathsf{Cluster} & \stackrel{\cdot}{\leq} & \mathsf{Small\text{-}Network} \\ \mathsf{Diskless\text{-}WS} \sqcap \mathsf{Diskbased\text{-}WS} & \doteq & \bot. \end{array}$$

Based on the entailment relation between terminologies and formulas, it is possible to define a relation on the set of concept descriptions, namely, the *subsumption relation* $\preceq_T$ defined as

$$C \preceq_T C' \text{ iff } T \models C \;\dot{\leq}\; C'.$$

This relation is obviously a preorder (that is, transitive and reflexive) on the set Σ_C of concept descriptions composed from symbols appearing in T, and a partial order on the quotient of Σ_C with respect to the equivalence relation $\approx_T$ defined by

$$C \approx_T C' \text{ iff } T \models C \doteq C'.$$

3.2 Computational Services of a TBox

One service TBoxes usually provide, called *classification* [28], is the computation of a *concept taxonomy*, such as the one in Figure 2, which represents the *immediate subsumption relation* between concepts for the terminology in the previous subsection.

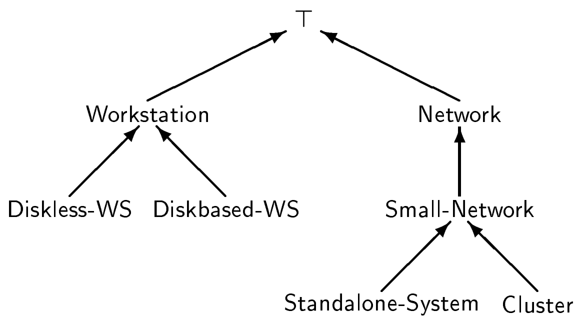

Figure 2: A concept taxonomy.

Evidently, subsumption and classification are intertwined. In order to compute the concept taxonomy, subsumption between concepts must be determined. Once the concept taxonomy has been computed, subsumption between concepts can be read off from the taxonomy,[3] that is, classification can be regarded as a kind of assert-time inference technique.

Obviously, classification is a versatile service for the knowledge acquisition task. Classification points out all implicit relationships between concepts which might have been missed when introducing a concept. As shown in [1, 15], classification can be used to drive the knowledge acquisition process by employing a number of reasonable heuristics such as that different concepts should not denote the same set and that no concept should be *incoherent*, that is, equivalent to $\perp$. Note that such incoherent concepts are quite useless because they denote the empty set, but they do not "infect" the knowledge base in the sense a contradictory proposition in logic does. Terminologies always have at least one model, namely, the trivial one interpreting every concept and role as the empty set.

Knowledge acquisition is not the only application where classification can be put to use. In general, any problem requiring *classification-based reasoning* [23] can exploit this service. This kind of reasoning proceeds along the following line. Given some concept description, identify the concepts which most accurately characterize the given description and use information associated with the identified concepts to do something meaningful, that is, the concepts are used as a kind of *conceptual coat rack* [62].

Making this idea less abstract, let us assume that we want to identify a plan in order to solve a problem. Now, we may define a hierarchy of problem concepts associating with each such problem concept a plan for solving the problem. Thus, given a particular problem description, classification can determine the most specialized set of problem concepts for which plans are known in order to solve the given problem [43]. Such an organization of problem-solving knowledge is not only very elegant and natural, but also makes maintenance of such a knowledge base easier and supports explanation facilities. Other examples of where this kind of representation and reasoning can be profitably exploited are computer configuration [45], natural language generation [40], presentation planning [3], and information retrieval [59].

3.3 Formal World Descriptions

However, in most of the applications cited above, one does not start with a description of, say, a particular problem, but one has a collection of *objects* (denoted by c, d) and *relationships* between them. Given such a *world description*, which is dealt with in the ABox, one wants to know the set of concepts most accurately describing those objects. In order to capture this formally, let us again extend our formalism. This time, however, we do not add new description-forming expressions or terminological axioms, but something, which will be called *world axioms* (*WA*) in order to describe objects by naming the concepts they shall be an *instance* of and to describe relationships between two objects by specifying *role relationships*:

$$\begin{array}{lcll} WA & \to & C(c) & \textit{instance description} \\ & | & r(c,d) & \textit{relation description.} \end{array}$$

Using the interpretation of concepts and roles given above, and interpreting objects c as elements of the

[3]Actually, in some representation systems not only the concept taxonomy but also the transitive closure is stored.

domain, i.e.,

$$c^{\mathcal{I}} \in D$$

using the *unique names hypothesis*

$$\text{if } c^{\mathcal{I}} = d^{\mathcal{I}} \text{ then } c = d,$$

a *world axiom is satisfied by an interpretation*, written $\models_{\mathcal{I}} \omega$, under the following conditions:

$$\begin{array}{rll} \models_{\mathcal{I}} C(c) & \text{iff} & c^{\mathcal{I}} \in C^{\mathcal{I}} \\ \models_{\mathcal{I}} R(c,d) & \text{iff} & d^{\mathcal{I}} \in R^{\mathcal{I}}(c^{\mathcal{I}}). \end{array}$$

Similar to the definition of a model of a terminology, we can say what we mean by a *model of a world description* W or by a *model of a world description* W *combined with a terminology* T, namely, an interpretation which satisfies all world axioms in W or all terminological and world axioms in $T \cup W$, respectively. Furthermore, we will say that c is an *instance* of C iff $T \cup W \models C(c)$.

3.4 Putting the Boxes together

Combining TBox and ABox,[4] we can describe a world by specifying the objects and relationships of interest by a world description W employing a terminology T. For instance, we may use the computer-network terminology in order to describe a particular configuration:

$$\begin{array}{l} \mathsf{Workstation(w1)} \\ \mathsf{Network(n1)} \\ (\exists^{\leq 3}\mathsf{member})(\mathsf{n1}) \\ \mathsf{server(n1, w1)} \end{array}$$

From this constellation it follows that w1 is a Diskbased-WS because every role-filler of the server role for Networks is a Diskbased-WS and, additionally, n1 is a Cluster and a Small-Network.

In general, TBox/ABox systems provide a computational service called *realization* [31] which computes for each object c the set of most specialized concepts $MSC(c)$ the object is an instance of. Formally, $MSC(c)$ is a minimal set of concepts[5] such that

$$\begin{array}{rll} \text{if } A \in MSC(c) & \text{then} & T \cup W \models A(c) \qquad (1) \\ \text{if } T \cup W \models B(c) & \text{then} & \exists A\colon A \in MSC(c) \land \\ & & \mathrm{T} \cup W \models A \stackrel{\cdot}{\leq} B. \qquad (2) \end{array}$$

[4]Recently, also rules found their way into such systems, a facility formally analyzed in [53].

[5]Actually, of equivalence classes of concepts in order to guarantee uniqueness.

In our case, condition (2) can be simplified because world descriptions form an (almost) *conservative extension* of terminologies, that is, entailment of terminological formulas depends (in all interesting cases) only on the terminology and not on the world description.

Theorem 1 *If a world description* W *and a terminology* T *are jointly satisfiable, then*

$$T \cup W \models \tau \;\; \textit{iff} \;\; T \models \tau$$

for all terminological formulas τ.[6]

This means that we can formulate condition (2) equivalently as

$$\begin{array}{rll} \text{if } T \cup W \models B(c) & \text{then} & \exists A\colon A \in MSC(c) \land \\ & & A \preceq_T B, \end{array}$$

provided the assumption of the theorem holds. This means in particular that after $MSC(c)$ has been computed, instance relationships for c can be determined by looking up subsumption in the concept taxonomy. Note that this property, which supports the first of the communication protocols mentioned in Section 2.4, heavily depends on the fact that the world axioms have very limited expressiveness. If arbitrary first-order formulas are permitted, as in KRYPTON, the computation of instance relationships becomes much more complicated.

In a presentation planning application [3], the information associated with the concepts in $MSC(c)$ might be used to decide how to represent a given object. In a database or information retrieval application, $MSC(c)$ can be used to *index* the data objects by the concepts in $MSC(c)$ [4, 6, 50]. *Query processing* can then be implemented as *classification* of a *query concept*, *retrieval* of all objects indexed by the immediate superconcepts of the query concept in the concept taxonomy, and *filtering* by testing each retrieved object against the query concept.

4 Algorithmic Considerations

Although we have talked about computational services, we haven't given algorithms which do the actual computations. However, instead of specifying inference algorithms (see, e.g. [27, 37, 41, 46, 56]), we will investigate the computational properties of the problems.

As we have seen in the previous section, subsumption determination is the central operation in a TBox

[6]Proofs for this and the following theorems and propositions can be found in [37].

system. This point is reinforced by the fact that all other interesting properties and relations, such as *equivalence of two concepts* ($C \approx_T C'$), *incoherency of a concept* ($C \approx_T \bot$), and *disjointness of two concepts* ($(C \sqcap C') \approx_T \bot$) can be reduced to subsumption in linear time.

Proposition 1 *Given a terminology T and two concept descriptions C, C':*

1. $C \approx_T \bot$ *iff* $C \preceq_T \bot$
2. $C \approx_T C'$ *iff* $C \preceq_T C'$ *and* $C' \preceq_T C$

Similarly, subsumption can be reduced to equivalence.

Proposition 2 *Given a terminology T and two concept descriptions C, C':*

$$C \preceq_T C' \;\; \textit{iff} \;\; C \approx_T (C \sqcap C').$$

Furthermore, if we had a concept-negation operator, such as the one introduced below, it would be possible to reduce subsumption to incoherency in linear time.

4.1 The Subsumption Problem

In other words, when looking for efficient inference algorithms for TBox systems, we have to find an efficient subsumption algorithm. In order to simplify matters, we will show how subsumption in arbitrary terminologies can be reduced to subsumption in the *empty terminology*, denoted by $\emptyset$.

First of all, a function *Exp* from concept descriptions and terminologies to concept descriptions is defined. This function repeatedly replaces all concepts A appearing in a given concept description C by the right hand side of the introduction of A until no further replacements are possible—adding a special fresh concept for every concept A introduced by a specialization axiom. Thus, $Exp(C,T)$ contains only *undefined concepts*, that is, concepts that do not appear as the left hand side of a terminological axiom in T. Obviously, *Exp* terminates if T does not contain *terminological cycles*, i.e., a direct or indirect occurrence of the concept introduced on the left hand side in the concept description on the right hand side.[7] Moreover, *Exp* does not change the meaning of a concept since the replaced subexpressions and the replacing expressions in C are identically interpreted in T.

[7]Although such terminological cycles are sometimes useful, they are usually not supported by TBox systems because their intuitive meaning is not completely clear and they complicate the design of subsumption algorithms (but cf. [36] and [37].)

Proposition 3 *For every model $\mathcal{I}$ of a terminology T:*

$$C^{\mathcal{I}} \;\; = \;\; (Exp(C,T))^{\mathcal{I}}.$$

From this observation it is almost immediate that subsumption in terminologies can be reduced to subsumption in the empty terminology $\emptyset$, i.e., to subsumption of concept descriptions, which will be called *term-subsumption*.

Theorem 2 *Given a terminology T and two concept descriptions C, C':*

$$C \preceq_T C' \;\; \textit{iff} \;\; Exp(C,T) \preceq_\emptyset Exp(C',T).$$

This means when developing subsumption algorithms, we have to consider only the description-forming part of the formalism. Considering the description-forming language specified in Section 3.1, it is easy to show that subsumption can be determined in polynomial time by adapting the subsumption algorithm and the proof of [27] to $\mathcal{STL}$.

4.2 Computational Complexity of Term-Subsumption

An interesting question coming up in this context is how far we can go in enhancing the expressiveness of a description-forming language without sacrificing efficiency, i.e., polynomial-time computational subsumption determination. A (partial) answer is given in Table 1. In order to interpret this table, some more term-forming operators, which have been used in TBox systems, have to be introduced, however. First of all, let us extend the concept-description language:

C	$\rightarrow$	$\ldots$	
	$\mid$	$C \sqcup C'$	*concept disjunction*
	$\mid$	$\neg C$	*concept negation*
	$\mid$	$\exists^{\geq n} R\colon C$	*generalized number restr.*
	$\mid$	$\exists R\colon C$	*generalized exist. restr.*
	$\mid$	$\exists R$	*existential restriction*
	$\mid$	$R \downarrow R'$	*role agreement.*

The meaning of *concept disjunction* and *negation* should be obvious. *Generalized number restrictions* permit us to describe concepts such as Parents who have at least 3 Male-persons as their children:

$$\exists^{\geq 3}\textsf{child}\colon \textsf{Male-person}.$$

Role agreement can be used to define, for instance, the concept of an Autobiography:

$$\textsf{Biography} \sqcap (\textsf{subject} \downarrow \textsf{author})$$

Name	Concept-Forming Operators	Role-Forming Operators	Complexity of Subsumption, Remarks, and References
$\mathcal{FL}^-$	$(C \sqcap C')$, $(\forall R\colon C)$, $(\exists R)$		polynomial [8, 27]
$\mathcal{STL}$	$(C \sqcap C')$, $(\forall R\colon C)$, $(\exists^{\geq n} R)$, $(\exists^{\leq n} R)$		polynomial, with ψ-terms [2] added probably still polynomial (CLASSIC, see [6])
$\mathcal{FL}$	$(C \sqcap C')$, $(\forall R\colon C)$, $(\exists R)$	$(R\vert_C)$	co-NP-hard [8, 27]
$\mathcal{ALE}$	$(C \sqcap C')$, $(\forall R\colon C)$, $(\exists R\colon C)$		NP-complete [11]
BACK	$(C \sqcap C')$, $(\forall R\colon C)$, $(\exists^{\geq n} R)$, $(\exists^{\leq n} R)$	$(R \sqcap R')$	co-NP-hard, without $(R \sqcap R')$ but with $(\exists^{\geq n} R\colon C)$ (KANDOR, see [48]) still co-NP-hard [34]
$\mathcal{ALC}$	$(C \sqcap C')$, $(C \sqcup C')$, $\neg C$, $(\forall R\colon C)$, $(\exists R\colon C)$		PSPACE-complete [56] with $(\exists^{\geq n} R)$ and $(\exists^{\leq n} R)$ still in PSPACE, with feature logic [57] added still decidable [20]
	$(C \sqcap C')$, $(\forall R\colon C)$, $(\exists^{\geq n} R)$, $(\exists^{\leq n} R)$	$(R \sqcap R')$, $(R\vert_C)$	polynomial for four-valued semantics [46]
$\mathcal{R}$		$(R \sqcap R')$, $\neg R$, $(R \circ R')$	undecidable [54]
KL-ONE*	$(C \sqcap C')$, $(\forall R\colon C)$, $(R \downarrow R')$	$(R \circ R')$	undecidable [55] slightly weaker result: [49]

Table 1: Complexity of subsumption in term-forming languages

The formal meaning of these concept-description operators can be specified as follows:

$$\begin{aligned}
(C \sqcup C')^{\mathcal{I}} &= C^{\mathcal{I}} \cup C'^{\mathcal{I}} \\
(\neg C)^{\mathcal{I}} &= D \setminus C^{\mathcal{I}} \\
(\exists^{\geq n} R\colon C)^{\mathcal{I}} &= \{x \in D \mid \|R^{\mathcal{I}}(x) \cap C^{\mathcal{I}}\| \geq n\} \\
(\exists R\colon C)^{\mathcal{I}} &= (\exists^{\geq 1} R\colon C)^{\mathcal{I}} \\
&= \{x \in D \mid R^{\mathcal{I}}(x) \cap C^{\mathcal{I}} \neq \emptyset\} \\
(\exists R)^{\mathcal{I}} &= (\exists R\colon \top)^{\mathcal{I}} \\
&= \{x \in D \mid R^{\mathcal{I}}(x) \neq \emptyset\} \\
(R \downarrow R')^{\mathcal{I}} &= \{x \in D \mid R^{\mathcal{I}}(x) = R'^{\mathcal{I}}(x)\}
\end{aligned}$$

Similar to concept-forming operators, role-forming operators are conceivable, and have indeed been used in different term-forming languages.[8] The symbol S will be used to denote *atomic roles*, while R will be used to denote *role-descriptions*:

R	$\rightarrow$	S	*atomic role*
	$\mid$	$R \sqcap R'$	*role conjunction*
	$\mid$	$\neg R$	*role negation*
	$\mid$	$R\vert_C$	*range restriction*
	$\mid$	$R \circ R'$	*role chain*

The formal meaning of these operators can be specified as follows:

$$\begin{aligned}
(R \sqcap R')^{\mathcal{I}} &= R^{\mathcal{I}} \cap R'^{\mathcal{I}} \\
(\neg R)^{\mathcal{I}} &= (D \times D) \setminus R^{\mathcal{I}} \\
(R|_C)^{\mathcal{I}} &= R^{\mathcal{I}} \cap (D \times C^{\mathcal{I}}) \\
(R \circ R')^{\mathcal{I}} &= \{\langle x, y\rangle \mid \exists z\colon z \in R^{\mathcal{I}}(x) \wedge y \in R'^{\mathcal{I}}(z)\}
\end{aligned}$$

As can be seen in Table 1, when trying to extend the term-forming part of $\mathcal{STL}$, one inevitably ends up with languages for which subsumption determination is intractable or, more precisely, NP-hard. Furthermore, introducing role-chains and agreements on roles or role-negation and conjunction, the result is even worse, subsumption determination becomes undecidable.

Since TBox/ABox systems are supposed to deliver answers in a reasonable amount of time, the situation described is quite disturbing. The following strategies have been applied to circumvent this problem:

1. Restricting the term-forming languages (e.g. KRYPTON, CLASSIC)
2. Using a weaker semantics which supports only tractable inferences [46].
3. Restricting the inferences the system can draw (e.g. KL-TWO, BACK, LOOM).

[8]Additionally, instead of using general roles, it would also be possible to use *features*—single-valued roles—as used in unification-based grammar formalisms, which are better behaved from a computational point of view [39].

All of these approaches have pros and cons. The first two strategies, which are favored in [27], may rule out a number of possible applications (see [13]), however, the restrictions are principled. The third approach (see e.g. LOOM [30]), on the other hand, restricts the inferences usually in an unprincipled way, i.e., it is often not clear which inferences the system will draw and which inferences are ignored.

4.3 Computational Complexity of Subsumption in Terminologies

As it turns out, the situation is even worse than described in the previous subsection. From a theoretical point of view, subsumption is inherently intractable. In Section 4.1, it was shown how to reduce *subsumption in a terminology* to *term-subsumption*. However, this reduction may lead to expressions that are not polynomially bounded in the size of the terminology, as the following example demonstrates:

$$\begin{aligned}
\mathsf{C}_1 &\doteq \forall \mathsf{r}\colon \mathsf{C}_0 \sqcap \forall \mathsf{r}'\colon \mathsf{C}_0 \\
\mathsf{C}_2 &\doteq \forall \mathsf{r}\colon \mathsf{C}_1 \sqcap \forall \mathsf{r}'\colon \mathsf{C}_1 \\
&\vdots \\
\mathsf{C}_n &\doteq \forall \mathsf{r}\colon \mathsf{C}_{n-1} \sqcap \forall \mathsf{r}'\colon \mathsf{C}_{n-1}.
\end{aligned}$$

Here, the size of $\mathit{Exp}(\mathsf{C}_n, T)$ is obviously proportional to 2^n. Of course, better algorithms are conceivable. But the subsumption problem in terminologies cannot be reduced generally to the subsumption problem in the empty terminology in linear time, since it is equivalent to the co-NP-complete problem of deciding *equivalence of nondeterministic automatons that accept finite languages* [18, p. 265].

Theorem 3 *Given a terminological formalism containing only the operators $\doteq$, $\sqcap$, and $\forall$, the problem of deciding whether $C \preceq_T C'$ in cycle-free terminologies is co-NP-complete.*[9]

This result means that "the goal of forging a powerful system out of tractable parts" [27, p. 89] cannot be achieved in the area of TBox systems. Furthermore, it means that almost all TBox systems described in the literature that have been conjectured or proven to be tractable with respect to subsumption over the description forming language can be blown

[9]A proof appears in [38].

up with a carefully thought out example. However, nobody seems to have noticed this fact, and, indeed, terminologies occurring in applications appear to be well-behaved. In this respect and with regard to the structure, our problem is similar to the type inference problem in ML, which seems to be solvable in linear time in all practical applications encountered so far, but is PSPACE-hard in general [24].

The conclusion one can draw from this strange situation is that although the theory of computational complexity can shade some light on the structure of a problem, one should not be scared by intractability in the first place. It may well be the case that it is possible to find algorithms that are well-behaved in all *normal cases*. Furthermore, it seems promising to reevaluate the results shown in Table 1 in this light.

In our case, a reasonable restriction on the form of terminologies, which can be considered as a "normal case," is that the "depth of a terminology" over roles and definitions is logarithmicly bounded by the size of the terminology [38]. This restriction, which seems to be met by all terminologies I have seen so far, ensures that the expanded concept expressions are polynomially bounded in size by the size of the terminology.

Probably, it is not possible to find always such simple restrictions. However, it may turn out that for all cases occurring in practice, sophisticated algorithms can avoid an exponential explosion (see e.g. [12, 25]).

4.4 The Instance Determination Problem

Viewed in isolation, the results reported in Table 1 and Theorem 3 are discouraging. However, as already pointed out above, Theorem 3 may not affect the efficiency of subsumption in practice, and, moreover, intractability of term-subsumption may not be significant if we look at the context in which TBoxes are used.

The main application of TBox/ABox systems is *classification-based reasoning*, that is, the determination of the concepts a particular object is an instance of in order to apply some rules or procedures to it. For this purpose, however, the computation of a concept taxonomy is not strictly necessary. It suffices to check (for all concepts) whether a given object is an instance or not. Unfortunately, though, in the general case, testing the instance relationship is of similar complexity as determining subsumption. But there are some special cases, in which the computation of instance relationships is easy.

For example, in the BACK system, which uses an incomplete subsumption algorithm because subsumption determination is intractable (see Table 1), the realization algorithm described in [41] and [37, Ch. 4] turns out to be complete in the important special case when the world description contains enough information in order to decide whether a role relationship between two objects holds or not—when the world description is *role-closed.*

This behavior has to do with the fact that role-closed world descriptions have a *canonical model* $\mathcal{M}$, that is, an instance relationship is entailed by $T \cup W$ if, and only if, it is satisfied by $\mathcal{M}$.[10] Furthermore, this canonical model can be easily (in polynomial time) computed when the world-description is role-closed.

Technically, this means that although the subsumption relation is not completely determined, all instance relations will be found—i.e., in this case, the overall system is more complete than the parts.

5 Summary and Conclusions

Hybrid KRR systems address two important research issues, namely, expressiveness of representation languages and efficiency of reasoning. We have reviewed different such systems and have identified some critical points concerning the architecture of such systems.

Concentrating on TBox/ABox systems, we showed the interplay of different formalisms on the semantic level and presented an analysis of the computational problems involved. Although the theoretical results suggest that efficient reasoning in such systems is impossible, we pointed out some directions how to design efficient systems. First, in designing inference algorithms, not the worst case, but the *normal case* should be considered and supported. Second, sometimes completeness of subsumption determination may be sacrificed and the emphasis should be shifted to regain completeness for the overall system in important special cases.

Acknowledgments

This paper greatly profited from my work in the KIT-BACK group and from discussions in the LILOG group.

References

[1] G. Abrett and M. H. Burstein. The KREME knowledge editing environment. *International Journal of Man-Machine Studies*, 27(2):103–126, 1987.

[2] H. Aït-Kaci. *A Lattice-Theoretic Approach to Computations Based on a Calculus of Partially*

[10] This property depends, obviously, on the expressiveness of the terminological language.

Ordered Type Structures. PhD thesis, University of Pennsylvania, Philadelphia, Pa., 1984.

[3] Y. Arens, L. Miller, S. C. Shapiro, and N. K. Sondheimer. Automatic construction of user-interface displays. In *Proceedings of the 7th National Conference of the American Association for Artificial Intelligence*, pages 808–813, Saint Paul, Minn., Aug. 1988.

[4] H. W. Beck, S. K. Gala, and S. B. Navathe. Classification as a query processing technique in the CANDIDE semantic data model. In *Proceedings of the International Data Engineering Conference, IEEE*, pages 572–581, Los Angeles, Cal., Feb. 1989.

[5] C. Beeri and E. Rivin. Integrated architecture for counseling expert systems. In *Proceedings of the International Symposium on Computational Intelligence '89*, Milan, Italy, Sep. 1989.

[6] 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*, pages 59–67, Portland, Oreg., June 1989.

[7] R. J. Brachman and H. J. Levesque. Competence in knowledge representation. In *Proceedings of the 2nd National Conference of the American Association for Artificial Intelligence*, pages 189–192, Pittsburgh, Pa., Aug. 1982.

[8] 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*, pages 34–37, Austin, Tex., Aug. 1984.

[9] 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*, pages 532–539, Los Angeles, Cal., Aug. 1985.

[10] A. G. Cohen. On the appearance of sortal literals: a non-substitutional framework for hybrid reasoning. In R. J. Brachman, H. J. Levesque, and R. Reiter, editors, *Proceedings of the 1st International Conference on Principles of Knowledge Representation and Reasoning*, pages 55–66, Toronto, Ont., May 1989.

[11] F. M. Donini, B. Hollunder, M. Lenzerini, A. M. Spaccamela, D. Nardi, and W. Nutt. *The Frontier of Tractability for Concept Description Languages.* DFKI-Report, DFKI, Kaiserslautern, West Germany, 1989. In preparation.

[12] J. Dörre and A. Eisele. Determining consistency of feature terms with distributed disjunctions. In *GWAI-89. 13th German Workshop on Artificial Intelligence*, pages 270–279, Springer-Verlag, Berlin, West Germany, 1989.

[13] 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., May 1989.

[14] J. Edelmann and B. Owsnicki. Data models in knowledge representation systems: a case study. In C. Rollinger and W. Horn, editors, *GWAI-86 und 2. Österreichische Artificial-Intelligence-Tagung*, pages 69–74, Springer-Verlag, Berlin, West Germany, 1986.

[15] T. W. Finin and D. Silverman. Interactive classification as a knowledge acquisition tool. In L. Kerschberg, editor, *Expert Database Systems—Proceedings From the 1st International Workshop*, pages 79–90, Benjamin/Cummings, Menlo Park, Cal., 1986.

[16] P. Floréen, P. Myllymäki, P. Orponen, and H. Tirri. Neural representation of concepts for robust inference. In *Proceedings of the International Symposium on Computational Intelligence '89*, Milan, Italy, Sep. 1989.

[17] A. M. Frisch. A general framework for sorted deduction: fundamental results on hybrid reasoning. In R. J. Brachman, H. J. Levesque, and R. Reiter, editors, *Proceedings of the 1st International Conference on Principles of Knowledge Representation and Reasoning*, pages 126–136, Toronto, Ont., May 1989.

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

[19] W. Hoeppner, T. Christaller, H. Marburger, K. Morik, B. Nebel, M. O'Leary, and W. Wahlster. Beyond domain-independence: experience with the development

of a German natural language access system to highly diverse background systems. In *Proceedings of the 8th International Joint Conference on Artificial Intelligence*, pages 115–121, Karlsruhe, West Germany, Aug. 1983.

[20] B. Hollunder. *Subsumption Algorithms for Some Attributive Description Languages.* Master's thesis, Department of Computer Science, Universität Kaiserslautern, Kaiserslautern, West Germany, Aug. 1989.

[21] E. J. Horvitz. Rational metareasoning and compilation for optimizing decisions under bounded resources. In *Proceedings of the International Symposium on Computational Intelligence '89*, Milan, Italy, Sep. 1989.

[22] J. Jaffar and J. Lassez. Constraint logic programming. In *Proceedings of the 14th ACM Symposium on Principles of Programming Languages*, pages 111–119, ACM, Munich, West Germany, Jan. 1987.

[23] 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*, pages 978–987, Philadelphia, Pa., Aug. 1986.

[24] P. C. Kanellakis and J. C. Mitchell. Polymorphic unification and ML typing. In *Proceedings of the 16th ACM Symposium on Principles of Programming Languages*, pages 5–15, Jan. 1989.

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

[26] H. J. Levesque. Foundations of a functional approach to knowledge representation. *Artificial Intelligence*, 23(2):155–212, 1984.

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

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

[29] R. MacGregor. A deductive pattern matcher. In *Proceedings of the 7th National Conference of the American Association for Artificial Intelligence*, pages 403–408, Saint Paul, Minn., Aug. 1988.

[30] 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.

[31] W. Mark. Realization. In J. G. Schmolze and R. J. Brachman, editors, *Proceedings of the 1981 KL-ONE Workshop*, pages 78–89, Cambridge, Mass., 1982. The proceedings have been published as BBN Report No. 4842 and Fairchild Technical Report No. 618.

[32] D. A. McAllester. *Reasoning Utility Package User's Manual.* AI Memo 667, AI Laboratory, Massachusetts Institute of Technology, Cambridge, Mass., Apr. 1982.

[33] M. Moia. Expert systems and hypertext: a promising integration for training. In *Proceedings of the International Symposium on Computational Intelligence '89*, Milan, Italy, Sep. 1989.

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

[35] B. Nebel. A knowledge level analysis of belief revision. In R. J. Brachman, H. J. Levesque, and R. Reiter, editors, *Proceedings of the 1st International Conference on Principles of Knowledge Representation and Reasoning*, pages 301–311, Toronto, Ont., May 1989.

[36] B. Nebel. On terminological cycles. In *Preprints of the Workshop on Formal Aspects of Semantic Networks*, Two Harbors, Cal., Feb. 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.

[37] B. Nebel. *Reasoning and Revision in Hybrid Representation Systems.* PhD thesis, Universität des Saarlandes, Saarbrücken, West Germany, June 1989.

[38] B. Nebel. *Terminological Reasoning is Inherently Intractable.* IWBS Report 82, IWBS, IBM Deutschland, Stuttgart, West Germany, Oct. 1989.

[39] B. Nebel and G. Smolka. Representation and reasoning with attributive descriptions. In K. Bläsius, U. Hedtstück, and C. Rollinger, editors, *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.

[40] B. Nebel and N. K. Sondheimer. NIGEL gets to know logic: an experiment in natural language generation taking a logical, knowledge-based view. In C. Rollinger and W. Horn, editors, *GWAI-86 und 2. Österreichische Artificial-Intelligence-Tagung*, pages 75–86, Springer-Verlag, Berlin, West Germany, 1986.

[41] B. Nebel and K. von Luck. Hybrid reasoning in BACK. In Z. W. Ras and L. Saitta, editors, *Methodologies for Intelligent Systems*, pages 260–269, North-Holland, Amsterdam, Holland, 1988.

[42] B. Nebel and K. von Luck. Issues of integration and balancing in hybrid knowledge representation systems. In K. Morik, editor, *GWAI-87. 11th German Workshop on Artificial Intelligence*, pages 114–123, Springer-Verlag, Berlin, West Germany, 1987.

[43] R. Neches, W. R. Swartout, and J. D. Moore. Explainable (and maintainable) expert systems. In *Proceedings of the 9th International Joint Conference on Artificial Intelligence*, pages 382–389, Los Angeles, Cal., Aug. 1985.

[44] A. Newell. The knowledge level. *Artificial Intelligence*, 18(1):87–127, 1982.

[45] B. Owsnicki-Klewe. Configuration as a consistency maintenance task. In W. Hoeppner, editor, *GWAI-88. 12th German Workshop on Artificial Intelligence*, pages 77–87, Springer-Verlag, Berlin, West Germany, 1988.

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

[47] P. F. Patel-Schneider. A hybrid, decidable, logic-based knowledge representation system. *Computational Intelligence*, 3(2):64–77, May 1987.

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

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

[50] P. F. Patel-Schneider, R. J. Brachman, and H. J. Levesque. ARGON: knowledge representation meets information retrieval. In *Proceedings of the 1st Conference on Artificial Intelligence Applications*, pages 280–286, Denver, Col., 1984.

[51] O. Raiman. Two heuristics integrating probabilities and logic: a preliminary report on parsimonious search. In *Proceedings of the International Symposium on Computational Intelligence '89*, Milan, Italy, Sep. 1989.

[52] C. Rich. The layered architecture of a system for reasoning about programs. In *Proceedings of the 9th International Joint Conference on Artificial Intelligence*, pages 540–546, Los Angeles, Cal., Aug. 1985.

[53] K. Schild. *Towards a Theory of Frames and Rules.* Master's thesis, Department of Computer Science, Technische Universität Berlin, Berlin, West Germany, 1989.

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

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

[56] M. Schmidt-Schauß and G. Smolka. Attributive concept descriptions with unions and complements. *Artificial Intelligence*, 1989. To appear. Also Available as SEKI Report SR-88-21, Department of Computer Science, Universität Kaiserslautern, Kaiserslautern, West Germany, 1988.

[57] G. Smolka. *A Feature Logic with Subsorts.* LILOG Report 33, IBM Deutschland, Stuttgart, May 1988.

[58] M. E. Stickel. Automated deduction by theory resolution. In *Proceedings of the 9th International Joint Conference on Artificial Intelligence*, pages 1181–1186, Los Angeles, Cal., Aug. 1985.

[59] F. N. Tou, M. D. Williams, R. E. Fikes, A. Henderson, and T. Malone. RABBIT: an intelligent database assistant. In *Proceedings of the 2nd National Conference of the American Association for Artificial Intelligence*, pages 314–318, Pittsburgh, Pa., Aug. 1982.

[60] M. B. Vilain. The restricted language architecture of a hybrid representation system. In *Proceedings of the 9th International Joint Conference on Artificial Intelligence*, pages 547–551, Los Angeles, Cal., Aug. 1985.

[61] 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, Jan. 1987.

[62] W. A. Woods. What's important about knowledge representation. *IEEE Computer*, 16(10):22–29, Oct. 1983.