Deduction as Parsing: Tractable Classification in the KL-ONE Framework

Marc Vilain

The Mitre Corporation
Burlington Road, Bedford, MA 01730
Internet: MBV@MITRE.ORG

Abstract

This paper presents some complexity results for deductive recognition in the framework of languages such as KL-ONE. In particular, it focuses on classification operations that are usually performed in these languages through subsumption computations. The paper presents a simple language that encompasses and extends earlier KL-ONE-based recognition frameworks. By relying on parsing algorithms, the paper shows that a significant class of recognition problems in this language can be performed in polynomial time. This is in marked contrast to the exponentiality and undecidability results that have recently been obtained for subsumption in even some of the most restricted variants of KL-ONE.

THE SINGLE MOST RECURRING THEME in knowledge representation, and possibly the most successfully applied, is that of types and type taxonomies. Much of the literature, both practical and theoretical, takes the need for explicit type representations as a given. In particular, considerable attention has been paid over the years to languages derived from KL-ONE, as the central concerns of these languages are types and type definitions. The hallmark of KL-ONE and its descendants is that these languages provide a set of type-forming operators that allow one to define types (usually called concepts) and interrelate them through a notion of type subsumption. Subsumption is usually treated as necessary type entailment, and its computational characteristics are one of the main concerns of this body of work.

Unfortunately, most studies of subsumption have produced pessimistic results. Even for simple type languages subsumption is intractable [Brachman & Levesque 1984, Levesque & Brachman 1987], and for barely less simple languages it is undecidable [Schmidt-Schauß 1989, Patel-Schneider 1989]. These results notwithstanding, subsum-ption remains of interest to the KR community in that it enables a form of automated recognition. Type classifiers such as that of Lipkis use subsumption to index types automatically into a taxonomy [Schmolze & Lipkis 1983]. In my own work on the KL-TWO representation system [Vilain 1985], subsumption-based classification was used to recognize those types instantiated by an individual, and thereby to index rules applicable to that individual.

The relevance of KL-TWO to the present paper is that despite the intractability and undecidability of subsumption in the general case, the classification of individuals as performed in KL-TWO is computable in polynomial time. This computational discrepancy arises from the specific (and restricted) ways in which KL-TWO relies on subsumption. The present paper is an examination of how this enables tractable classification. It presents a recognition language that at first doesn't look at all like a language from the KL-ONE family, but that embodies and extends those aspects of KL-TWO that lead to tractable classification. The bulk of the paper is concerned with demonstrating for this language the tractability of a classification-like recognition process that is based on notions of chart parsing.

KL-ONE languages and classification

RECENT ACTIVITY IN THE KL-ONE COMMUNITY has led to a proliferation of KL-ONE-inspired representation languages.[1] They share the common goal of formalizing frames as type structures called concepts; slots are treated as binary relations called roles. Another shared characteristic is that they all provide a number of concept- and role-forming operators. A typical collection of these is shown in Figure 1 below, along with a typical account of their semantics. These operators are put to use in forming terminological definitions, for example:

DOCTOR = (and PERSON
(exists (vrdiff HAS-DEGREE MED-DEGREE)))
; *a person with a degree that is a medical degree.*

DOCTPARENT = (and PERSON (all CHILD DOCTOR))
; *a person all of whose children are doctors.*

In most accounts of KL-ONE, subsumption is treated as necessary denotational inclusion. To be precise, a concept c_1 is said to subsume a concept c_2 just in case $[\![c_2]\!] \subseteq [\![c_1]\!]$, and a role r_1 is said to subsume another role r_2 just in case $[\![r_2]\!] \subseteq [\![r_1]\!]$. By this definition, the DOCTPARENT concept is subsumed by such concepts as "parent of graduates," e.g.,

(and PERSON (all CHILD (exists HAS-DEGREE)))

A variety of algorithms exist for computing subsumption on the basis of concept and role definitions but, as mentioned above, these are tractable only for restricted cases. Brachman and Levesque [1984], for one, exhibit a

1. For a comprehensive catalogue, see [Patel-Schneider *et al.* 1990] or [Woods & Schmolze 1990].

Expressions (α)	Denotations of expressions ($[\![\alpha]\!]$)
c, a concept, or r, a role	$[\![c]\!] = \mu(c) \subseteq \delta$ and $[\![r]\!] = \mu(r) \subseteq \delta \times \delta$
(and $c_1 \ldots c_n$)	$[\![(\text{and } c_1 \ldots c_n)]\!] = [\![c_1]\!] \cap \ldots \cap [\![c_n]\!]$
(exists r)	$[\![(\text{exists } r)]\!] = \{x \in \delta \mid \exists\, y \in \delta \text{ such that } \langle x, y\rangle \in [\![r]\!]\}$
(all r c)	$[\![(\text{all } r\ c)]\!] = \{x \in \delta \mid \forall\, y \in \delta, \text{if } \langle x, y\rangle \in [\![r]\!] \text{ then } y \in [\![c]\!]\}$
(= r_1 r_2)	$[\![(= r_1\ r_2)]\!] = \{x \in \delta \mid \forall\, y \in \delta, \langle x, y\rangle \in [\![r_1]\!] \text{ just in case } \langle x, y\rangle \in [\![r_2]\!]\}$
(vrdiff r c)	$[\![(\text{vrdiff } r\ c)]\!] = \{\langle x, y\rangle \in [\![r]\!] \mid y \in [\![c]\!]\}$
(chain $r_1 \ldots r_n$)	$[\![(\text{chain } r_1 \ldots r_n)]\!] = \{\langle x, y\rangle \in \delta \times \delta \mid \exists\, z_1 \ldots z_{n-1} \text{ such that } \langle x, z_1\rangle \in [\![r_1]\!] \ldots \langle z_{n-1}, y\rangle \in [\![r_n]\!]\}$

The semantics of concept- and role-forming expressions is given with respect to to a domain δ (a set of entities) and a modeling function μ that assigns interpretations in δ to atomic expressions. That is, $\mu(c) \subseteq \delta$ and $\mu(r) \subseteq \delta \times \delta$, for each atomic concept c and role r. The c_i and r_i are either atomic or complex, except in the first row, where they are strictly atomic.

Figure 1: Common type-forming operators.

polynomial algorithm for subsumption for the language defined by atomic concepts and roles, and expressions formed with the *and*, *all*, and *exists* operators. However, they then extend the language to include *vrdiff* expressions, and show this leads the computation of subsumption to be co-NP-complete. Schmidt-Schauß [1989] goes further, showing that the operators *and*, *all*, =, and *chain* are sufficient to make computing subsumption undecidable. For another such proof, see [Patel-Schneider 1989].

In contrast, the process of individual (or instance) classification can be performed tractably despite being ultimately based on subsumption. The underlying reason for this is that instance classification, at least as done in KL-TWO, does not require the full complement of type-forming operators that are typically considered in analyzing subsumption. In KL-TWO, instance classification proceeds from ground propositions predicated of individuals, e.g.,

(PERSON *fred*) ∧ (HAS-DEGREE *fred md*) ∧ (MED-DEGREE *md*)

Individuals are classified by generalizing their predicating propositions to terminological definitions which are then matched to the terminological database. In this case, *fred*'s generalization would be

(and PERSON (exists (vrdiff HAS-DEGREE MED-DEGREE)))

which (trivially) subsumes the definition of DOCTOR.

This kind of classification was used in KL-TWO as the basis of a deductive rule matcher.[2] In this case, rules that might contain a pattern such as (DOCTOR *?x*) could still be indexed and applied to the individual *fred*, despite their not literally matching any ground predication of *fred*.

The reason this instance classification scheme does not lead to intractability turns out to be due to the impossibility of recognizing *directly* any concepts that impose universal restrictions on roles, i.e., concepts defined with the *all* or = concept-forming operators. Take the case of DOCTPARENT, defined as (and PERSON (all CHILD DOCTOR)), and say we have

(PERSON *fred*) ∧ (CHILD *fred mary*) ∧ (DOCTOR *mary*)

We are not entitled from this alone to conclude monotonically that (DOCTPARENT *fred*)! This would only be inferrable if we had, for example, independent knowledge that the set of individuals satisfying $\lambda\, x$ (CHILD *fred* x) had cardinality $n \leq 1$. Knowing this would then allow us to reduce the definition of DOCTPARENT as a special case to the concept

(and PERSON (exists (vrdiff CHILD DOCTOR))),

which is in fact satisfied by *fred*.

In the general case, one can show that without recourse to external non-ground information such as role filler cardinality, universal restrictions can not be recognized through instance classification.[3] Putting things most simply, one can not use *deduction* to recognize universals from ground facts as doing so is by definition *induction*.

In effect, the type language recognized by KL-TWO consists of those concepts and roles that are definable with the *and*, *exists*, and *vrdiff* operators, or that can be reduced to such concepts and roles through cardinality considerations. One can readily verify that subsumption in this language can be computed in polynomial time, and by extension, so can the process of instance classification performed by KL-TWO (more on this below). This language is clearly rudimentary, but it does suggest a strategy for achieving tractability of recognition in a more expressive language.

2. This indexing strategy was first described in the KL-ONE framework by Mark [1982]. See also [Schubert, *et al.* 1979]. The term *deductive pattern matcher* is due to Mac Gregor [1988].

3. As the preceding discussion suggests, KL-TWO relies on this kind of cardinality information to reduce universal restrictions to existential ones. Similar ly, the recent CLASSIC system [Brachman *et al.* 1990] provides a *close* operator to specify that the system has been given a complete enumeration of the fillers of a role.

Extending the language of recognition

THE APPROACH ADOPTED HERE departs from the KL-ONE tradition in separating those aspects of a concept's definition that impose existential restrictions and those that impose universal restrictions. More precisely, I mean by the existential aspects of a concept's definition those characteristics that are provided by operators whose semantic account introduces either no variables, or only existentially quantified variables. Examples of such operators are the *and*, *exists*, *vrdiff*, and *chain* operators of Figure 1. The universal aspects of a concept are those provided by operators that require the introduction of universally quantified variables, e.g., *all* and =.

This distinction between universal and existential restrictions is rendered operational in a simple representation language called RHO (ρ), after the Greek letter for "r" (as in ρecognition or ρepresentation). RHO provides two sub-languages for this purpose: an axiomatic language to state the existential restrictions on a type, and a frame-like language to state the universal ones. For the PARENT type, for example, we might state the following.

(PARENT u_1) ← (PERSON u_1) + (CHILD u_1 e_1)
; i.e., a person for which there exists a child
(frame PARENT (PERSON (CHILD))
; i.e., all of a parent's children are persons

Following the argument in the preceding pages, only the existential axioms of RHO are used in recognizing instances of a type τ. The universal statements about τ are irrelevant for recognition, and only come into play once instances of τ have been recognized. One can thus think of the axiomatic sub-language of RHO as providing sufficiency rules for a type, i.e., rules specifying conditions whose satisfaction is sufficient to guarantee membership in the type. Conversely, the universal statements can be seen as necessity rules, contingent conditions that must necessarily hold of instances of the type. This particular conceptualization of necessary and sufficient conditions for type membership is dictated in part by this work's concern with recognition, and is thus limited in scope. Nevertheless, the terminology of necessity and sufficiency affords a convenient operational characterization of the sub-languages of RHO, and as such will be used throughout this paper.

Sufficiency axioms in RHO

Ignoring necessity conditions until later in this paper, sufficiency conditions are specified in RHO with function-free Horn axioms. In addition to the sufficiency condition for the PARENT type above (PARENT is a canonical KL-ONE concept), some examples of this syntax are given below. The first corresponds to a *vrdiff* role definition in KL-ONE, and the second mentions three-place relations, which have been mostly ignored in KL-ONE-inspired languages, with some exceptions (e.g. [Schmolze 1989]).

(DAUGHTER u_1 u_2) ← (CHILD u_1 u_2) + (FEMALE u_2)

(PASS-TO u_1 u_2 u_3) ← (THROW-TO u_1 u_2 u_3) + (CATCH u_2 u_3)

To be precise, the syntax of sufficiency conditions in RHO follows the axiom schema below, where the π^i are predicates, with α_i the arity of predicate π^i.

$$(\pi^0\ \xi^0_1 \ldots \xi^0_{\alpha_0}) \leftarrow (\pi^1\ \xi^1_1 \ldots \xi^1_{\alpha_1}) + \cdots + (\pi^n\ \xi^n_1 \ldots \xi^n_{\alpha_n})$$

The ξ terms in this schema must be either constants or variables—function terms are excluded. As is usual with Horn clauses, variables appearing on both the left-hand side and right-hand side of a sufficiency condition are implicitly universally quantified; variables appearing only on the right-hand side are implicitly existentially quantified. This is reflected in the names of the variables in the preceding examples: the u_i are universal, and the e_i are existential. As a result of this way of interpreting variables, sufficiency rules in RHO are precluded from recognizing universally quantified restrictions such as those imposed on concepts in KL-ONE by *all*.

To see this, assume without loss of generality that there are no constants on the left-hand side of a sufficiency axiom satisfying the schema above, i.e., no constants in

$$\xi^0_1 \ldots \xi^0_{\alpha_0}.$$

The sufficiency axiom can then be factored into a predicate definition with existential quantification, and a universally quantified sufficiency assertion.

$$\pi' \equiv \lambda\, \vec{\xi^0}\ \exists\, \vec{\xi^r}\ (\pi^1\, \vec{\xi^1}) \wedge \cdots \wedge (\pi^n\, \vec{\xi^n})$$

$$\forall\, \vec{\xi^0}\ (\pi'\, \vec{\xi^0}) \supset (\pi_0\, \vec{\xi^0})$$

The $\vec{\xi^i}$ designate the arguments of the π^i, and $\vec{\xi^r}$ designates those variables appearing on the right hand side of the axiom but not on the left hand side.

For example, the sufficiency axiom for PARENT above can be rendered into the following existential predicate definition and universal axiom.

PARENT' ≡ λ u_1 ∃ e_1 (PERSON u_1) ∧ (CHILD u_1 e_1)

∀ u_1 (PARENT' u_1) ⊃ (PARENT u_1)

For one- and two-place predicates, the predicate definition can be further expressed in terms of KL-ONE-like type-forming operators, in this case

PARENT' ≡ (and PERSON (exists CHILD))

Expressions (α)	Denotations of expressions ($[\![\alpha]\!]$)
c, a concept, or r, a role	$[\![c]\!] = \mu(c) \subseteq \delta$ and $[\![r]\!] = \mu(r) \subseteq \delta \times \delta$
(and $c_1 \ldots c_n$)	$[\![(\text{and } c_1 \ldots c_n)]\!] = [\![c_1]\!] \cap \ldots \cap [\![c_n]\!]$
(and $r_1 \ldots r_n$)	$[\![(\text{and } r_1 \ldots r_n)]\!] = [\![r_1]\!] \cap \ldots \cap [\![r_n]\!]$
(exists $r_1 \ldots r_n$)	$[\![(\text{exists } r_1 \ldots r_n)]\!] = \{x \in \delta \mid \exists\, y \in \delta$ such that $\langle x, y\rangle \in [\![r_1]\!] \cap \ldots \cap [\![r_n]\!]\}$
(vrdiff r c)	$[\![(\text{vrdiff } r\ c)]\!] = \{\langle x, y\rangle \in [\![r]\!] \mid y \in [\![c]\!]\}$
(chain $r_1 \ldots r_n$)	$[\![(\text{chain } r_1 \ldots r_n)]\!] = \{\langle x, y\rangle \in \delta \times \delta \mid \exists\, z_1 \ldots z_{n-1}$ such that $\langle x, z_1\rangle \in [\![r_1]\!] \ldots \langle z_{n-1}, y\rangle \in [\![r_n]\!]\}$

The domain δ the modeling function μ, concept terms c_i and role terms r_i are as in Figure 1.

Figure 2: Type-forming operators for sufficiency axioms.

The reason sufficiency axioms must be factored into two formulæ, one definitional and the other axiomatic, is that RHO allows several sufficiency axioms to support the recognition of the same left-hand side. For example, a multimedia messaging domain (*cf.* CONSUL [Mark 1982]) might afford multiple ways of recognizing a priority message:

(PRIORITY-MSG u_1) $\leftarrow$ (MSG u_1) + (FROM u_1 e_1) + (BOSS e_1)

(PRIORITY-MSG u_1) $\leftarrow$ (MSG u_1) + (REPLY-BY u_1 *today*)

and so forth. It is clear that though these axioms may each provide a sufficient recognition criterion for PRIORITY-MSG, they are not definitional in the traditional KL-ONE sense.

Semantics of sufficiency axioms

These notions are formalized through a straightforward model-theoretic semantics for the language. As usual, constants denote elements of a domain δ, and a predicate π of arity α denotes a subset of δ if $\alpha = 1$, and a subset of

$$\underbrace{\delta \times \ldots \times \delta}_{\alpha},$$

if $\alpha > 1$.

Next, let μ be a model function for δ, that is, a function that assigns to a constant κ in the language some element of δ, and that assigns to a predicate π in the language a subset of the appropriate cross-product of δ (given the arity of π). Let β designate a variable binding function that maps variables among the ξ terms to constants in the language, and maps constants to themselves. Then μ is said to interpret a set of sufficiency axioms if for each such axiom

$$(\pi^0\ \xi^0_1 \ldots \xi^0_{\alpha_0}) \leftarrow (\pi^1\ \xi^1_1 \ldots \xi^1_{\alpha_1}) + \cdots + (\pi^n\ \xi^n_1 \ldots \xi^n_{\alpha_n})$$

and for any β,

$$\langle \mu(\beta(\xi^0_1)) \ldots \mu(\beta(\xi^0_{\alpha_0}))\rangle$$

is in $\mu(\pi^0)$ whenever

$$\langle \mu(\beta(\xi^i_1)) \ldots \mu(\beta(\xi^i_{\alpha_n}))\rangle$$

is in $\mu(\pi^i)$ for each of the π^i. That is, in any model that interprets the axioms, whenever the right-hand side of an axiom is instantiated, so is its left-hand side.

RHO and the KL-ONE languages

As promised, the syntax of sufficiency conditions in RHO bears little resemblance to the traditional syntax of KL-ONE languages. However, as noted above, these conditions can be factored in part into KL-ONE-like predicate definitions, for which it is possible to define a predicate-forming specification language. In general, this language has cumbersome syntax, mostly because of the need to indicate the co-references specified by variables in the axioms. Restricting one's attention to axioms with predicates of arity 2 or less (i.e., concepts and roles), it is possible to define a KL-ONE-like language that covers an interesting set of special cases among these axioms. This language is given in Figure 2.

In addition, one can organize the predicates defined by these axioms into multiple type taxonomies, one for each predicate arity. To do so, we just place any predicate π^i on the right-hand side above predicate π^0 whenever their arities match (i.e., $\alpha_i = \alpha_0$), and whenever

$$\vec{\xi^i} = \vec{\xi^0},$$

i.e., the arguments of π^i are identical to those of π^0. The taxonomies so defined do not encode any subsumption relations, but they do have the property subsumption hierarchies have that any instantiation of a predicate π in the taxonomy is also an instantiation of all those predicates above π in the taxonomy.

A parsing algorithm for recognition

ONE OF THE ATTRACTIVE CONSEQUENCES of casting recognition criteria as deductive rules is that this approach leads to a direct implementation of recognition as a parsing problem. It may seem unlikely at first that a deductive process should be amenable to being treated as parsing. However, deductive recognition is a very particular kind of

deduction, and its analogies to parsing are multiple. Just as grammars allow one to compose syntactic constituents into larger syntactic units, deductive rules allow one to combine propositions to deduce oher propositions. Just as a bottom-up recognizer starts from atomic "facts" (the positions of words in a buffer) and proceeds to find all paths to the start symbol, a forward-chaining deductive recognizer starts from base propositions and deduces all their consequences.

The recognition strategy adopted here is based on notions of bottom-up chart parsing. As syntacticians will notice, not all characteristics of this kind of parser actually come to bear on deductive recognition. By casting the problem in this way, however, the computational properties of the recognition problem in RHO are made clear.

The principal data structure in a chart parser is the chart [Kay 1980], a two-dimensional table indexed by buffer position that records which constituents have been parsed out of the buffer. Letting Ch designate the chart, a given constituent $\chi \in Ch[i, j]$ just in case χ has been parsed out of the buffer between positions i and j. For deductive recognition, the chart is an α-dimensional table, where α is the maximal arity of predicates in the recognition rules. Chart cells are indexed by individual constants of the language. If a proposition $(\pi\ \kappa_1 \ldots \kappa_n)$ has been given as a premise, or has been deduced from other propositions, then π is entered in cell $Ch[\kappa_1 \ldots \kappa_n]$ of the chart . For expository purposes, one can assume that Ch is implemented as an α-dimensional array A. The chart entry indexed by individuals $\kappa_1 \ldots \kappa_n$ is thus $A(N(\kappa_1),\ldots, N(\kappa_n), 0, \ldots, 0)$, where N is a function that returns an integer code (≥ 1) for each constant κ_i and the 0's fill in any missing indices of A if the κ_i are fewer than α in number. This storage strategy is clearly simple-minded and wasteful of space; many improvements are possible (see [Ullman 1988, 1989]).

The deductive recognition process operates bottom up, and as with many syntactic recognizers for context-free languages, intermediate states of recognition are encoded as "dotted rules" [Earley 1970]. Initially a rule's dot is to the left of the first term on the right-hand side. If this term matches an entry in the chart, a copy of the rule is created that advances the dot past the matching term, indicating that this new rule must match on its second term. This process of creating dotted rules is continued until the right-hand side is exhausted, whereupon the left-hand side is added to the chart. As with unification grammars [Shieber 1986], the variable bindings that ensue from matching a term to a chart entry are associated with the resulting dotted rule, and are consulted in subsequent matches. For further details on this process, and for specific algorithms, see [Earley 1970; Kay 1980; or Shieber 1986].

Computational properties

It is straightforward to verify that this parsing algorithm correctly computes all the deductions sanctioned by a set of sufficiency rules. In effect, the chart implicitly defines a modeling function μ: $\langle \mu(\kappa_1) \ldots \mu(\kappa_n) \rangle \in \mu(\pi)$ just in case π is entered into the chart in cell $Ch[\kappa_1 \ldots \kappa_n]$. That this μ provides an interpretation of the sufficiency rules follows directly from the way the parsing process adds entries into the chart.

The computational complexity of this algorithm, unfortunately, is unappealing. In the general case, unification grammars have the formal power of a Turing machine. This result does not fully apply here, principally because function symbols are barred from the recognition rules. Nevertheless, the problem of computing the deduction sanctioned by sufficiency rules can be readily shown to be NP-complete, by reduction from subgraph isomorphism or from conjunctive Boolean query [Garey & Johnson 1979].

The reduction from subgraph isomorphism is of particular interest, as it helps reveal a tractable subclass of deductive recognition problems. The reduction proceeds by mapping the matching graph (the putative subgraph) into a recognition rule. For each node $n_1 \ldots n_n$ in the matching graph, a variable $v_1 \ldots v_n$ is created, and for each vertex from n_i to n_j, the term (CONNECTS v_i v_j) is added to the rule. The rule's left-hand side is the term (SG $v_1 \ldots v_n$). The target supergraph is then mapped to ground propositions (CONNECTS m_i m_j), where m_i and m_j are target graph nodes. The matching graph is isomorphic to a subgraph of the target just in case (SG $m_1 \ldots m_n$) is recognized for some $m_1 \ldots m_n$.

Achieving tractability

This reduction demonstrates that the potential computational burden of deductive recognition arises from the need to perform arbitrary graph matching to instantiate a rule. This suggests that if one could restrict deductive rules so as to preclude graph matching, recognition might be tractable. The following criterion provides such a restriction.

The criterion is based on a partial order $\angle$ defined by the existential variables introduced on the rule's right-hand side. Starting from the universal variables u_i on the left-hand side, which are clustered as the bottom element of $\angle$, an undirected graph of the partial order is obtained by consulting terms on the right-hand side in order of decreasing arity. If a term mentions variables $v_1 \ldots v_i$ that are already in the order and variables $v_j \ldots v_n$ that are not, a cluster c containing $v_j \ldots v_n$ is first added to the graph. Then, for any two variables (old or new) in $v_1 \ldots v_n$, an

undirected link is added between their variable clusters, provided there is not already a path between the two that only mentions the variable clusters of $v_1 \ldots v_n$. The rule is said to define a variable tree just in case the graph of $\angle$ is a tree rooted at the u_i cluster. Note that the examples given earlier in this paper meet this criterion, as does any rule equivalent to an expression formed with the operators in Figure 2. For example, the first sufficiency axiom given above for PARENT,

(PARENT u_1) $\leftarrow$ (PERSON u_1) + (CHILD u_1 e_1),

only has two variables, which are clustered and ordered as $\{u_1\} \angle \{e_1\}$. This trivially defines a variable tree.

Given this notion, it can then be shown that recognition over any set of rules that define variable trees can be performed in polynomial time. The proof proceeds by estimating the amount of matching necessary to instantiate the left-hand side of a rule r. Say $|\kappa|$ is the total number of constants appearing in ground propositions, and α is the maximal predicate arity in the rules. Then to obtain all possible consistent instantiations of the variables in a variable cluster of r can only require $O(|\kappa|^{\alpha})$ matches. Because r defines a variable tree, at most another $O(|\kappa|^{\alpha})$ matches are necessary to relate the cluster to all other clusters for r. The number of such clusters is bounded by a parameter $|\xi|$, the maximal number of variables in a rule. The total number of left-hand side instantiations is again bounded by $|\kappa|^{\alpha}$. The overall cost of fully instantiating the left-hand sides is thus $O(|\xi|\ |\kappa|^{3\alpha})$, a polynomial in $|\kappa|$ with $|\xi|$ and α as parameters.

Further issues

The preceding result demonstrates that despite the undecidability of subsumption in most KL-ONE-like languages, the related problem of instance classification, or recognition, is in fact tractable to a significant degree. That is the main result of this paper. Beyond this, a number of issues remain that I would like to address before closing.

Universal conditions

The discussion to this point has largely ignored the specification of universal restrictions, such as the selectional restrictions on predicates that one might want to impose in order to state, e.g., that all of a person's children are persons. As I suggested above, RHO provides universal (or necessity) conditions such as these with a simple frame language, in this case:

(frame PARENT (PERSON (CHILD)).

The details of RHO's necessity language are not of great interest here. By a fairly pedestrian set of meaning postulates, statements such as the preceding are interpreted as universal restrictions on the fillers of relations, in this case:

$\forall\ x, y$ (PARENT x) $\wedge$ (CHILD x, y) $\supset$ (PERSON y).

In a somewhat perverse twist, it is worth pointing out that necessity conditions can actually be recast in terms of RHO's sufficiency rules, as in (PERSON y) $\leftarrow$ (CHILD x y).

RHO in perspective

I should note that though the framework adopted here is loosely based on chart parsing, that is not strictly necessary. The notion of implementing deduction with parsing strategies is, I would hope, both provocative and amusing, but in truth it is related to the way rule matching is implemented in certain forward-chaining rule languages, for example McAllester's RUP [McAllester 1982]. An interesting topic of further study would be to compare the performance, theoretical and actual, of this "parsing deducer" to that of traditional forward-chaining architectures such as OPS5 [Forgy 1982] or TREAT [Miranker 1987].

In closing I should acknowledge that there are significant representational issues lurking behind this work. By separating existential and universal and existential restrictions, and by making their semantics implicational (as opposed to biconditional), RHO rejects the standard KL-ONE notion of necessary and sufficient type definitions. Along these lines, a counter-example that has been levelled in criticism of this work is the concept of an all-girl school. It is easy to define this concept in KL-ONE, but no non-trivial sufficiency axiom can be given for it in RHO. Such criticisms are certainly valid, but are somewhat beside the point. The whole point of this work is to support *deductive* recognition from ground facts; recognizing an all-girl school by observing its students' gender is a non-monotonic inference that is *inductive* in nature.

As a point of speculation, my suspicion is that progress towards this particular non-monotonic reasoning problem will come from representational approaches that pay attention to issues of learning. Research is proceeding in this direction, including some of my own [Vilain, Koton & Chase 1990]. This area holds much promise for the future.

Acknowledgements

This research was begun while I was affiliated with Bolt, Beranek, and Newman Inc. (BBN), where Remko Scha offered much patience and attention. Bill Woods and Steve Minton provided critical insights, and Ellen Hays provided incisive criticism. I also owe much gratitude to the Arnold Arboretum of Harvard University for providing the setting in which these ideas were first developed.

References

Brachman, R. J. & Levesque, H. J. (1984). The tractability of subsumption in frame-based description languages. In *Proceedings of the Fourth National Conference on Artificial Intelligence* (AAAI84), Austin, TX, 34–37.

Earley, J. (1970). An efficient context-free parsing algorithm. *Communications of the ACM* 13: 84-102. Reprinted in [Grosz *et al.* 1986].

Forgy, C. L. (1982). A fast algorithm for the many pattern/many object pattern matching problem. *Artificial Intelligence* 19: 17-37.

Garey, M. R. & Johnson D. S. (1979). *Computers and intractability: a guide to the theory of NP-completness.* New York, NY: W. H. Freeman and Co.

Grosz, B. J., Sparck Jones, K., & Webber, B. L., eds. (1986). *Readings in Natural Language Processing.* Los Altos, CA: Morgan-Kaufmann Publishers.

Kay, M. (1980). *Algorithm schemata and data structures in syntactic processing.* Technical report CSL-80-12, Xerox Palo Alto Research Center. Reprinted in [Grosz *et al.* 1986].

Levesque, H. J. & Brachman, R. J. (1987). Expressiveness and tractability in knowledge representation and reasoning. *Computational Intelligence* 3: 78-93.

Mark, W. S. (1982). Realization. In Schmolze J. G., and Brachman, R. J., eds. *Proceedings of the Second KL-ONE Workshop.* Technical Report No. 4842, Bolt Beranek and Newman, Inc., Cambridge, MA, 78-89.

McAllester, D. A. (1982). *Reasoning utilities package user's manual.* AI memo 667 MIT AI Lab.

Mac Gregor, R. M. (1988). A deductive pattern matcher. In *Proceedings of the Seventh National Conference on Artificial Intelligence* (AAAI88), Saint Paul, MN, 403–408.

Miranker, D. P. (1987). TREAT: *A new and efficient match algorithm for ai production systems.* PhD dissertation, Dept. of Computer Science, Columbia University.

Patel-Schneider, P. F. (1984). Undecidability of subsumption in NIKL. *Artificial Intelligence* 39: 263–272.

Patel-Schneider, P. F., Owsnicki-Klewe, B., Kobsa, A., Guarino, N., Mac Gregor, R., Mark, W. S., McGuiness, D. L., Nebel, B., Schmiedel, A., & Yen, J. (1990). Term subsumption language in knowledge representation. *AI Magazine* 11(2): 16–23.

Schmolze, J. G. (1989). Terminological knowledge representation systems supporting n-ary terms. In *Proceedings of the First International Conference on Knowledge Representation* (KR89), Toronto, 432–443.

Schmolze, J. G. & Lipkis, T. (1983). Classification in the KL-ONE knowledge representation system. In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence* (IJCAI83), Karlsruhe, Germany.

Schmidt-Schauß, M. (1989). Subsumption in KL-ONE is undecidable. In *Proceedings of the First International Conference on Knowledge Representation* (KR89), Toronto, 421–431.

Schubert, L. K., Goebel, R. G., & Cercone, N. J. (1979). The structure and organization of a semantic net for comprehension and inference. In Findler, N. V., ed., *Associative networks: Representation and use of knowledge by computers.* New York, NY: Academic Press.

Shieber, S. M. (1986). *An introduction to unification-based approach to grammar.* CSLI lecture notes no. 4. Distributed by University of Chicago Press.

Ullman, J.D. (1988). *Principles of database and knowledge-base systems, vol.* 1. Rockville, MD: Computer Science Press.

Ullman, J.D. (1989). *Principles of database and knowledge-base systems, vol.* 2. Rockville, MD: Computer Science Press.

Vilain, M. B. (1985). The restricted language architecture of a hybrid representation system. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence* (IJCAI85), Los Angeles, CA, 547-551.

Vilain, M. B., Koton, P. & Chase, M. P. (1990). On analytical and similarity-based classification. In *Proceedings of the Eighth National Conference on Artificial Intelligence* (AAAI90), Boston, MA, 867–874.

Woods, W. A. & Schmolze, J. G. (1990). The KL-ONE family. To appear in *Computers and Mathematics with Applications*, special issue on semantic networks in artificial intelligence. Also available as Technical Report TR-20-90, Aiken Computation Lab, Harvard University.