Integrating Rules in Term Subsumption Knowledge Representation Servers

Brian R Gaines

Knowledge Science Institute
University of Calgary
Calgary, Alberta, Canada T2N 1N4

gaines@cpsc.ucalgary.ca

Abstract

This paper addresses the integration of services for rule-based reasoning in knowledge representation servers based on term subsumption languages. As an alternative to previous constructions of rules as concept→concept links, a mechanism is proposed based on *intensional roles* implementing the axiom of comprehension in set theory. This has the benefit of providing both rules as previously defined, and set aggregation, using a simple mechanism that is of identical computational complexity to that for rules alone. The extensions proposed have been implemented as part of KRS, a knowledge representation server written as a class library in C++. The paper gives an example of their application to the ripple-down rule technique for large-scale knowledge base operation, acquisition and maintenance.

Introduction

In recent years there have been major advances in the theory and practice of knowledge representation systems originating from semantic nets. In particular the series of term subsumption languages commencing with KL-ONE (Brachman & Schmolze, 1985), developing through KRYPTON (Brachman, Gilbert & Levesque, 1985) and currently culminating in systems such as CLASSIC (Borgida, Brachman, McGuinness & Resnick, 1989) and LOOM (MacGregor, 1988) has reached a maturity of technology which offers the promise of knowledge representation 'utilities' or 'services' in Levesque's (1984) terminology. The logical foundations of the subsumption relation, techniques for its correct and complete calculation, and the interaction between representation power and the tractability of subsumption algorithms have been widely studied (Brachman & Levesque, 1984; Schmidt-Schauss, 1989; Nebel, 1990) and are becoming reasonably well-defined.

It is now feasible to develop knowledge representation servers on a par with floating-point arithmetic units and numeric libraries, as software (and perhaps ultimately hardware) modules with well-defined functionality and fast, reliable performance. As with arithmetic units, such servers by no means solve all the problems of a particular application domain, but they do greatly reduce the burden of system development, allowing effort to be focused on the specifics of particular systems rather than on what should be general utilities.

In practical terms, the current generation of term subsumption languages and associated theoretical studies may be seen as providing a well-defined and understood semantics for frame-based knowledge representation systems. However, the focus on terminological definitions has not been paralleled by similar in-depth analysis of rules in knowledge-based systems, and the provision of rules in term subsumption languages is simplistic compared with that of most expert system shells. As rules are regarded as part of the assertional, A-box, component and do not form part of terminological definitions in the T-box, this does not affect the theoretical analysis of subsumption. However, it restricts the services offered by a knowledge representation server, and requires for many tasks that additional inference engines be written apart from the server.

This paper addresses the problem of providing a powerful rule representation system well-integrated with a term subsumption language, with emphasis on knowledge acquisition issues such as the natural representation of rules with exceptions. As a side-effect the rule system defined also provides for the automatic formation of sets or aggregations of individuals. The next sections briefly outline the knowledge representation server and its visual language, the representation of rules within it through individuals with intensional, rather than extensional, role definitions, and some applications to knowledge acquisition and knowledge-base maintenance.

KRS: A Knowledge Representation Server

KRS is a knowledge representation server written as a class library in C++ with semantics that are a slight extension of those of CLASSIC (Borgida, Brachman, McGuinness & Resnick, 1989). KRS supports a textual input/output language similar to that of CLASSIC, but since its primary application is to knowledge acquisition (Gaines, 1990) it also supports an equivalent visual input/output language through an interactive grapher (Gaines, 1991). This visual language will be used for the exposition in this paper with some examples of its compilation into textual form.

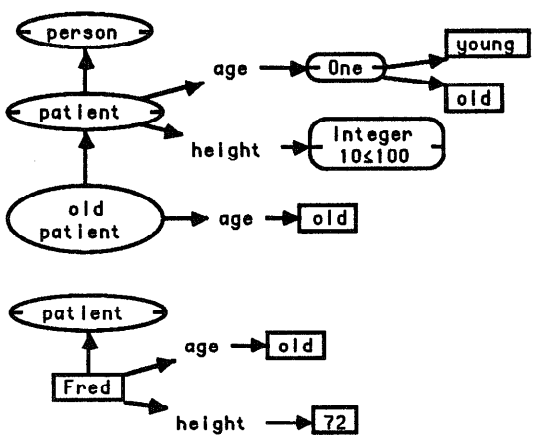
The visual language provides the means to represent knowledge structures as graphs of labelled nodes and arcs. The visual primitives of the language are:

The visual language provides the means to represent knowledge structures as graphs of labelled nodes and arcs. The visual primitives of the language are:

- Nodes, identified and typed as specified below.
- Two arc types linking nodes—a line with, and without, an arrow, respectively.
- Text strings labelling the nodes—with an associated equivalence relation based on lexical identity.
- Five distinctive text surrounds defining the node types—ovals (concepts), marked ovals (primitives), rectangles (individuals), no surround (roles or annotation), and marked rounded corner rectangles (constraints, e.g. cardinality and set inclusion).

The semantics of the arc types are overloaded and determined by the nodes joined. A line between two primitive nodes defines them as disjoint, for example, 

, and between two roles defines them as inverse, for example, parent — child . An arrow from one concept to another defines the first as subsumed by the second; from a concept to a role, that the role is part of the definition of the concept, etc. The graph at the top of Figure 1 shows the way in which concepts are defined and constraints and values asserted for individuals in the visual language. The resultant text statements are shown at the bottom.

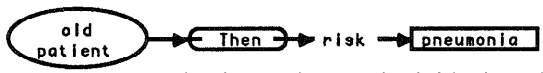



```
Primitive(person)
Primitive(patient, person,
  (All age, (One young, old))
  (All height, (Integer 10≤100))
)
Concept(old patient, patient,
  (All age, (Include old))
)
Individual(Fred, patient,
  (Fills age, old)
  (Fills height, 72)
)
```

Fig.1 Definitions and assertions in KRS visual and text languages

Integration of Rules

CLASSIC (Borgida, Brachman, McGuinness & Resnick, 1989) and LOOM (MacGregor, 1988) provide a basic form of rule in which the two concept definitions are linked in such a way that recognition of an individual as satisfying the constraints of the first concept leads to the assertion that it satisfies the constraints of the second. For example, if the following rule is added to the graph of Figure 1:



and the inference engine is run then the individual Fred will be recognized as an "old patient" and the constraint will be asserted will be made that the role "risk" for Fred includes "pneumonia."

The incorporation of rules as asserted concept→concept links integrates well with the way in which definitions in the T-box classify assertions in the A-box: if an individual is classified as falling under the first concept it is asserted to fall under the second; the first concept definition may be seen as specifying *sufficient* conditions for classification, and the second concept as additional *necessary* conditions.

McGregor (1988) has shown how such rules may be used to recognize recursive structures such as lists, and Figure 2 shows his construction in the visual language. The individual "c2" is recognized to be a "cons list" because it is asserted to be a "cons" and its cdr, "null", is asserted to be a "list". Hence, c2, through the rule, is asserted to be itself a "list". Then the individual "c1" is similarly recognized as a "cons list" because its cdr, "c2", has been asserted to be a "list".

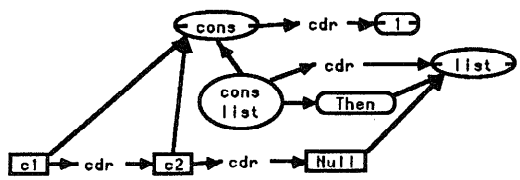


Fig.2 Recognition of a recursive structure using a rule

It may appear that a major limitation of this form of rule is that, since a single individual is classified at a time, it corresponds to an OPS5-style rule with only one free variable (McGregor, 1988). However, since the roles of an individual may be filled with other individuals, a rule classifying one individual may involve classifying, and making assertions about, several individuals. For example the rule with two free variables: ((data ?d) (clock ?c) (connected ?d ?c) → (connection-error ?d ?c)) in CADIE (Franke, 1990), has an equivalent in KRS as shown in Figure 3.

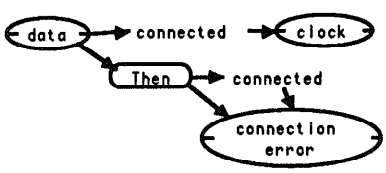


Fig.3 Rule with two free variables

If an individual that is asserted to be a "clock" is asserted to be "connected" to one that is asserted to be "data" then both individuals are asserted by the rule to have a "connection error". This construction in general clearly depends on the roles linking the free variables and would not be applicable to a rules whose premise references several completely independent individuals. However, this may be seen as a desirable constraint on rules integrated with frame-based knowledge representation schema—that a frame should exist, or be created, which brings into relation individuals accessed by the premise of a rule.

Mediating Rules through Aggregation

There is a more subtle criticism that can be made of the way in which rules are introduced into term subsumption languages. A rule is introduced as a new construct which links two concepts. It acts as an inferred assertion involving the classification of all individuals in the domain by the concepts that form the premise of a rule. However, the sets of individuals resulting from this classification are lost except in so far as it results in individuals being re-classified by the conclusion of the rule. There is a need in many applications of term subsumption languages for an aggregation operation that forms sets in a natural way (Allgayer & Reddig-Siekmann, 1990), and it is attractive to consider an alternative way of incorporating rules as the side effect of such aggregation.

The definition of role fillers is normally *extensional* in that specific individuals are named as fillers. Suppose one introduces a complementary way of filling a role *intensionally* by defining the concept which a role filler must satisfy and specifying that *all* individuals in the domain that satisfy it are fillers. This operationalizes Frege's *axiom of abstraction* or *comprehension*, that every concept defines a set. Introducing intensional role filling instead of rules as a new construct has the advantage of providing an explicit aggregation operator. Rules can be realized as a natural side-effect of the individuals actually being asserted to be in the intensional role. That is, the premise of the rule is now the intensional role definition, and the conclusion is the intensional role constraint.

As a first step to an aggregation operation, consider how Frege's axiom can be represented using inverse roles. Figure 4 shows a concept "intension x" whose role "member of" includes "extension x" whose role "member" is inverse to "member of" and constrained by "intension x". An individual asserted to be an "intension x" will be inferred to fill the "member" role of "extension x", and, conversely, an individual asserted to fill the "member" role of "extension x" will be inferred to be an "intension x".

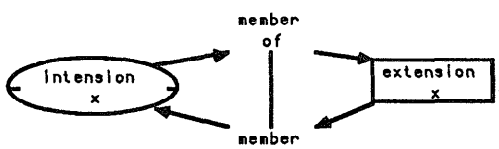


Fig.4 Axiom of comprehension with inverse roles

This construction is available in existing knowledge representation systems. For example, in LOOM, it is:

```
(defconcept intension-x :is (filled-by (:inverse member)
    extension-x))
```

The assertions of conceptual constraints on individuals:

```
(tell (intension-x ind1))   (tell (intension-x ind2))
```

then result in the query about members of extension-x:

```
(retrieve ?I (member extension-x ?I))
```

returning ind1 and ind2 as required.

The construction of Figure 4 is readily extended to have the side effect required for rules by having the arrow from "member" go, not to "intension x", but instead to another concept, "intension y" say, representing the conclusion of the rule. However, inverse roles alone cannot implement intensional role filling since it is still necessary to make the explicit assertion that an individual is an "intension x" rather than *recognize* this as being so.

Figure 5 shows how intensional role filling is introduced in KRS as a link from a concept, "intension x", to an individual, "extension x". This results in the inference engine aggregating the set of all individuals in the domain that are recognized as "intension x" and filling the "member" role of "extension x" with them. If the "member" role is constrained by a concept, "intension y", this also results in all the individuals recognized as "intension x" being additionally asserted to be "intension y", thus implementing a rule.

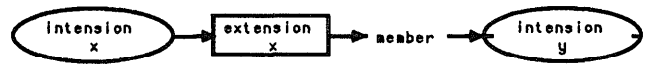


Fig.5 Intensional role, aggregation individual and rule

Intensional roles are proposed as a preferred alternative to concept→concept rules in term subsumption languages. They provide set aggregation and rules in one construct. Implementation in the inference engine involves no more computation than for rules alone. The additional storage space for the aggregations has not proved a significant overhead in a wide range of applications of KRS. Note that a "rule" in KRS is represented as an individual with an intensional role, and that the individual can have additional roles, and can itself be used as a role filler in other individuals. The availability of these features and explicit aggregations has proved invaluable in applications involving planning, configuration and scheduling.

Rules with Exceptions

A limitation of the rule schema described above is that they make no provision for the representation of rules with exceptions. Logically, if one does not require default reasoning, rules with exceptions can always be expanded into an equivalent, although generally larger, set of rules without exceptions. However, in knowledge representation for knowledge acquisition systems in particular, it is important to be able to encode expert knowledge in exact conformity with the expert's representation, which is often as a rule with exceptions. Hence, KRS offers a further extension in which one rule can be an exception to others.

As shown in Figure 6, an arrow from one individual to another means that the first individual acts as an exception in aggregation to the second. That is, that individuals are aggregated in the "member" role of the "aggregation rule" if they are classified as "premise rule" but not as "premise exception". Thus the exception mechanism provides for both exceptions to rules and differential aggregations. Since one individual may be an exception to many others, and may have many exceptions to itself, complex structures are readily represented. To reduce the number of arrows the exception arrow is taken to be transitive, so that an individual is an exception to all those individuals along its outgoing paths of exception arrows.

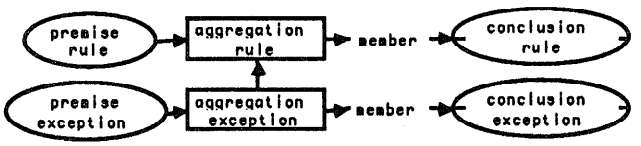


Fig.6 Representation of a rule with an exception

Figure 7 shows the solution to Cendrowska's (1987) contact lens problem represented in KRS as a set of two rules with three exceptions. This compares favorably with the minimal solution without exceptions, of nine rather more complex rules.

The implementation of intensional roles, aggregations, rules and exceptions in KRS is more subtle than has been indicated because no closed-world assumptions are made. Hence subsumption and recognition return one of three values, **true**, **false** or **open**, that is able to become either **true** or **false** as more assertions are made. If an individual is **open** in recognition for an intensional role this acts as **false** as far as placing it in the role is concerned but as **true** as far as exception propagation is concerned. Thus, the basic inference engine acts as a conservative, monotonic, reasoner in worlds where some roles or fillers are open.

It also supports a simple extension of the exception scheme to default reasoning in that the inference engine generates a list of **open** concept-individual pairs. Non-monotonic inference can commence with the default assumption that none of the open pairs will become **true** and hence not propagate **open** exceptions. If this leads to a consistent world it is the unique default extension. If it does not, a truth maintenance search can be invoked for maximal subsets of the open pairs that can be assumed **false**, resulting in zero to a number of possible extensions.

Ripple-Down Rules

Compton and Jansen (1990) have developed techniques for the acquisition and maintenance of large rule-based systems. They have applied their "ripple-down rule" techniques to the Garvan ES-1 knowledge base for thyroid diagnosis, which has grown to a size and complexity where it has become difficult to maintain.. The techniques rely on the efficient management of rules with a large number of exceptions, and the acquisition and maintenance procedures require access to aggregations of previously classified cases. Hence they provide a useful test of KRS capabilities in a domain where ongoing maintenance through continual knowledge acquisition is important, and where large rule sets and case bases are available.

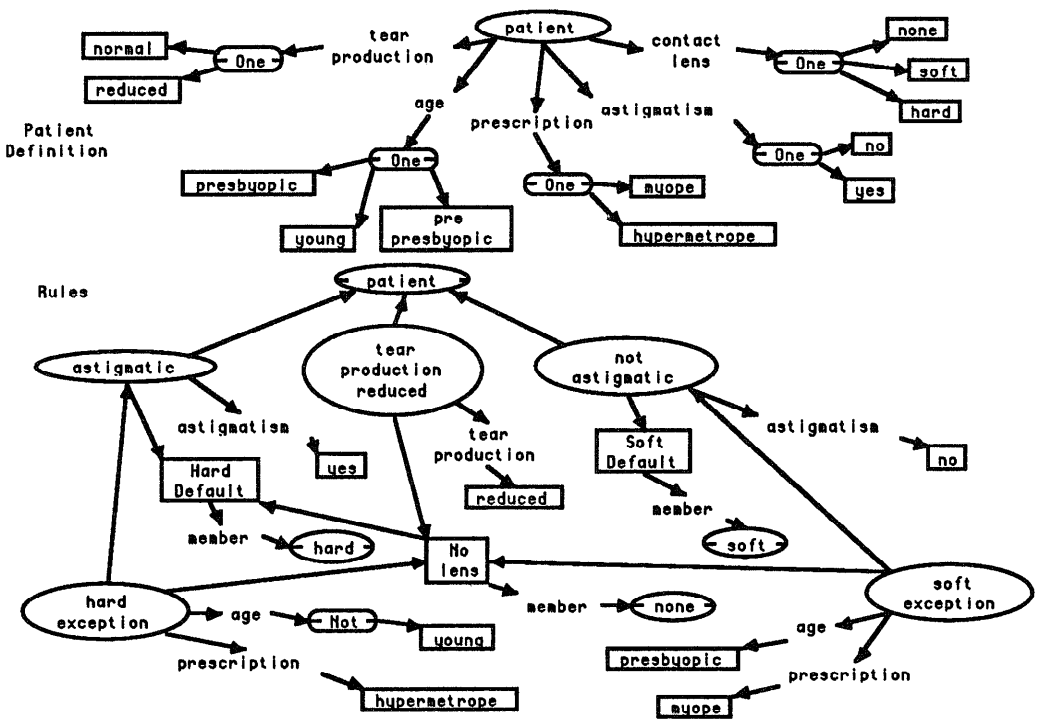


Fig.7 Solution to contact lens problem through rules with exceptions

The acquisition and maintenance procedures assume that an expert is available to make decisions about changes in the knowledge base. The objective of the ripple-down rule representation is to simplify the expert's task by allowing a new rule to be entered for a misdiagnosed case through a simple procedure that has only to take account of one existing rule and the cases that have already fallen under it. That is, the activities take place in a minimal context with a guarantee that no change will be made to the system's behavior outside that context.

Consider an empty knowledge base in which information about an individual "case 0" has been entered with a known diagnosis "diagnosis b". The expert creates a concept, "concept j", that will recognize "case 0" and attaches it to a rule, "rule j", that leads to "diagnosis b". Figure 8 shows the resultant state of the knowledge base.

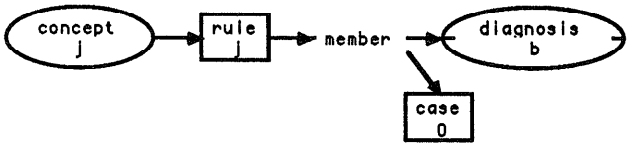


Fig.8 Initial case and rule entered in ripple-down rule base

Now consider the entry of further cases. If a case is identical to an existing one but has a different diagnosis then there is a conflict, otherwise there are four possibilities:

1. "Case 1" is identical to "case 0" and has same diagnosis. There is no need for a new rule or for the entry of the case.
2. "Case 2" falls under "concept j" and has "diagnosis b" but is not identical to "case 0". There is no need for a new rule but the case needs to be entered.
3. "Case 3" does not fall under "concept j" and has "diagnosis c". The expert proposes a new covering concept, "concept k", and rule, "rule k", following the same procedure as for "case 0". The existing rule, "rule j", is made an exception to this new rule.
4. "Case 4" falls under "concept j" but has a different diagnosis, "diagnosis a". A new rule is created by adding constraints to "concept j" that apply to "case 4" but not to "case 0" to create "concept i" subsumed by "concept j". The resultant "rule i" with "diagnosis a" is entered as an exception to "rule j".

Figure 9 shows the state of the knowledge base after this procedure has been followed.

Compton and Jansen's insight was that the procedure described could be repeated indefinitely as more cases are entered. It leads to a linear chain of rules such that a case may be seen as being entered at the beginning of the chain and rippling up (or "down"—KRS arrows are in the opposite direction to the original implementation diagrams) until it is recognized by the premise of a rule. The diagnosis is then determined by the conclusion of the rule.

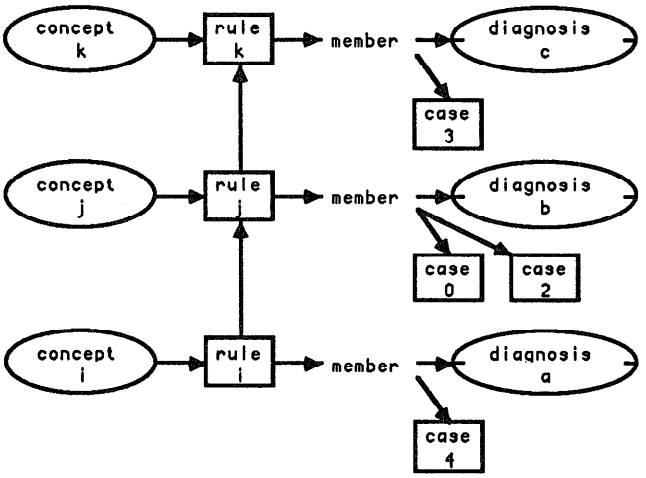


Fig.9 Ripple-down rule base after initial case entries

In knowledge acquisition and knowledge-base maintenance, if the case does not fall under any rule, a new one is created at the end of the chain as for "case 3" above. If it comes under a rule with an incorrect diagnosis, a new one is created with a premise that is subsumed by that of the existing rule and does not cover any of the cases that have already been diagnosed using the existing rule. This new rule is inserted in the chain as an exception to the one with the incorrect diagnosis.

A new rule can always be created using these procedures since a concept may be created that has precisely the constraints necessary to recognize uniquely the new case. However, the expert will often be able to generalize such a specific concept and, in so doing, need only consider discriminating the previous cases attached to the one rule.

The KRS rule mechanism supports both the chain of exceptions and the case aggregation required for ripple-down rules. The ripple-down version of the Garvan rules, 547 rules, together with 669 cases evaluated on 17 attributes, were loaded and run in KRS on a Macintosh II. The diagnoses took 0.4 seconds a case on average. Thus, even a microcomputer is able to support a significantly large knowledge-base operation, acquisition and maintenance using the ripple-down rules technique.

On obvious question regarding knowledge bases created through the ripple-down rules technique is the size-efficiency of the resultant knowledge base. Since rules, once entered, are not changed, if the expert over-generalizes or over-specifies, more rules will be generated than needed. Clearly size-efficiency is an empirical issue that can only be investigated over a number of knowledge bases created by different experts. When the data set of 669 cases are run through Induct (Gaines, 1989), an empirical induction algorithm that generates rules from cases with similar efficiency to C4.5 (Quinlan, 1987), a rule set giving correct diagnoses with 269 rules is generated, that is a reduction by about 50%. The resultant rule set, while smaller, may not be as understandable or acceptable as knowledge structure, and this is subject to further empirical investigation.

Conclusions

Developments in the theory and practice of term subsumption languages make possible generic knowledge representation servers offering efficient implementation of principled artificial intelligence techniques.

This paper has addressed the issue of integrating services for rule-based reasoning with knowledge representation servers based on term subsumption languages. As an alternative to previous constructions of rules as concept →concept links, a construction based on *intensional roles* is proposed. This has the benefit of providing rules as previously defined, and set aggregation, with a simple mechanism that is of identical computational complexity to that for rules alone. This mechanism extends simply to the representation of rules with exceptions.

It is interesting to examine what representational issues in the A-box are addressed by the new constructs. The exception mechanism itself implements concept negation in rule expression. In addition, since multiple concepts can point to the same aggregation individual, disjunctive rules can be expressed. These mechanisms also provide for the formation of set aggregations on a differential basis, essentially introducing concept disjunction and negation in both rules and set aggregation in the A-box.

The computational cost of the intensional role aggregation operation is identical to that of the rule mechanism it replaces. Intensional roles are processed exactly as if they were the premises of rules. The subsumption lattice that is computed as part of the KRS implementation of a term subsumption language is used to ensure very rapid recognition of those individuals which fall under the concept defining an intensional role. The exception links are processed quite separately after such recognition in a time that is negligible compared with the recognition itself.

The extensions proposed have been implemented as part of KRS, a knowledge representation server written as a class library in C++. The paper gives an example of their application to the *ripple-down rules* technique for large-scale knowledge base operation, acquisition and maintenance, and some evaluation of the space/time performance on the Garvan ES-1 knowledge-base.

Acknowledgements

This work was funded in part by the Natural Sciences and Engineering Research Council of Canada. I am grateful to Ron Brachman and Rob McGregor for access to their research on term subsumption languages, to Paul Compton and Bob Jansen for access to their research on ripple-down rules, to Paul Compton for access to the Garvan ES-1 knowledge base, and to Mildred Shaw for collaborative research on knowledge acquisition systems.

References

Allgayer, J. & Reddig-Siekmann, C. 1990. What KL-ONE lookalikes need to cope with natural language. Bläsius, K.H., Hedstück, & Rollinger, C.-R., Eds. *Sorts and Types in Artficial Intelligence*. pp.240-285. Berlin: Springer.

Borgida, A., Brachman, R.J., McGuiness, D.L. & Resnick, L.A. 1989. CLASSIC: a structural data model for objects. Clifford, J., Lindsay, B. & Maier, D., Eds. *Proceedings of 1989 ACM SIGMOD International Conference on the Management of Data*. pp.58-67. New York: ACM Press.

Brachman, R. J., Gilbert, V.P. & Levesque, H. J. 1985. An essential hybrid reasoning system: knowledge and symbol level accounts of KRYPTON. *Proceedings of IJCAI85*. pp.547-551. Morgan Kaufmann.

Brachman, R. J., & Levesque, H. J. 1984. The tractability of subsumption in frame-based description languages. *Proceedings of AAAI-84*. pp.34-37. Morgan Kaufmann.

Brachman, R.J. & Schmolze, J. 1985. An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9(2) 171-216.

Cendrowska, J. 1987. An algorithm for inducing modular rules. *International Journal of Man-Machine Studies* 27 (4), 349-370.

Compton, P. & Jansen, R. 1990. A philosophical basis for knowledge acquisition. *Knowledge Acquisition* 2(3), 241-258.

Franke, D.W. 1990. Imbedding rule inferencing in applications. *IEEE Expert*, 5(6) 8-14 (December).

Gaines, B.R. 1989. An Ounce of Knowledge is Worth a Ton of Data: Quantitative Studies of the Trade-Off between Expertise and Data based on Statistically Well-Founded Empirical Induction. *Proceedings of 6th International Workshop on Machine Learning*, pp.156-159. Morgan Kaufmann

Gaines, B.R. 1990. An architecture for integrated knowledge acquisition systems. Boose, J.H. & Gaines, B.R. Eds. *Proceedings of the Fifth AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop*. pp. 8-1-8-22. Banff (November).

Gaines, B.R. (1991). An interactive visual language for term subsumption visual languages. *IJCAI'91: Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*. Morgan Kaufmann.

Levesque, H. J. 1984. A logic of implicit and explicit belief. *Proc. AAAI-84*. pp.198-202. Morgan Kaufmann.

MacGregor, R.M. 1988. A deductive pattern matcher. *Proceedings of AAAI88*. pp.403-408. Morgan Kaufmann.

Nebel, B. 1990. *Reasoning and Revision in Hybrid Representation Systems*. Berlin: Springer.

Quinlan, J.R. 1987. "Simplifying decision trees," *International J. of Man-Machine Studies*, 27(3), 221-234.

Schmidt-Schauss, M. 1989. Subsumption in KL-ONE is undecidable *Proceedings of KR'89: First International Conference on Principles of Knowledge Representation and Reasoning*. pp.421-431. Morgan Kaufmann.