IRENE L. TRAVIS
Information Services Division
World Bank
Washington, DC

Knowledge Representation
in Artificial Intelligence

ABSTRACT
The problems of knowledge representation and use in expert systems and the problems of organizing and searching information in libraries and other bibliographic systems have much in common. There are two basic paradigms for representing knowledge in the knowledge bases of expert systems: rule-based and object-based. Of the two, the rule-based approach has had more publicity, but the object-oriented approach, which will seem more familiar to librarians, is coming to be seen as a necessary complement to rules or even as the more basic system component. One of the principal unsolved problems in knowledge representation is how to provide expert systems and natural language processing systems with more world knowledge, particularly "common sense" knowledge, in order to make them more robust. A major project to build a knowledge base of such basic information is underway at the Microelectronics and Computer Corporation (MCC), a corporation financed by a consortium of American industry to carry out research in advanced computing and computing technology. Since the project represents an attempt to organize a very large and general body of knowledge for use, it can be hypothesized that it will face many of the same problems faced by librarians as they have done the same thing. The project's published goals and achievements at the midpoint of its ten-year life are reviewed from that perspective. Four barriers to such efforts are discussed: (1) the variability of human performance in tasks related to knowledge representation and search; (2) the paradox
of structure; (3) the double-edged nature of the 80/20 rule; and (4) the inertia of an installed base.

INTRODUCTION

One purpose of this paper is to provide a more general overview of knowledge representation (KR) in artificial intelligence (AI) than is provided by the discussion of particular projects in other papers in these proceedings. In addition, while most papers have focused on what AI can contribute to libraries, this paper will turn the topic on its head. It will review an advanced AI research project from the perspective of what librarianship has learned through experience with building and maintaining large complex knowledge bases.

KNOWLEDGE REPRESENTATION

AI began as a set of largely unrelated activities in areas such as game playing, theorem proving, robotics, natural language understanding, expert reasoning, machine vision, and other fields. Some were attempts to model human cognition; others were highly pragmatic. With the high level of activity in AI in the last ten to fifteen years, a body of generally accepted practice has evolved in some areas, particularly in expert systems (Buchanan & Smith, 1989).

Basic Paradigms

When many people think of AI, they immediately think of rules. This association stems from the fact that some of the most publicized AI programs and early expert systems, such as MYCIN (Shortliffe, 1976), used rules as their knowledge bases (KB). A KB is the body of subject knowledge that supports the performance of a “knowledge-based system,” such as an expert or natural language processing system.

MYCIN was also the model for at least the initial versions of many commercial expert system shells. Shells are programs that contain the facilities for constructing an expert system, but which do not contain any subject knowledge when sold. They are a fairly exact equivalent of a database management system (DBMS) except that they support expert systems instead of databases. Because of the widespread use of shells and the relative ease of dealing with knowledge in the form of English-like rules, rule-based systems have enjoyed considerable popularity.
Rules easily translate information about how to do things. That is, they are "procedurally oriented," while information about objects in their domain of interest is not pulled together, but scattered and embedded throughout the rules. Not surprisingly, the complementary view of the world represented in many KR schemes is object-centered or "object-oriented." Objects may be things, but they may also be actions or events. Objects may be created for classes or for individuals. They can contain procedural information, but such information is attached to the objects. These two methods for organizing knowledge, procedurally based and object-based, are complementary, rather like alphabetical and classified arrangements in indexing languages. Although there are some exceptions, the same information can generally be expressed under either paradigm, but certain kinds of information are easier to create, control, modify, search, and use in one format than in the other. Many shells that initially had only rule-based capabilities now, therefore, also include facilities for creating and using objects. For an excellent current overview of AI written for the nontechnical reader, the author highly recommends the new textbook, Computers and Thought (Sharples et al., 1989).

The Role of Relations

Regardless of which paradigm is adopted, relational analysis is an essential component. When one looks at a rule stated in natural language, for instance,

Some restaurants take reservations.

the importance of relational analysis may not be obvious, but rules must, in fact, be translated into a formal language, whether into lists for LISP programs, the structure of some other programming language, or the predicate calculus. As soon as such rules are placed in such formal structures, relational analysis is necessary.

Relations may link an attribute to an individual (Crowded, Chez Pierre); an individual to a class (Restaurant, Chez Pierre); or describe the relationship between two or more individuals or classes (EatsAt, John, Chez Pierre). Relations are always stated in a particular order, since they are not usually commutative.

Suffice it to say that it is this emphasis on relational analysis that provides the first strong indication that KR for AI and the traditional pursuits of librarians constructing classification schemes and thesauri are very closely linked. Relational analysis is essential to all index language construction, involving, as it does, the identification and specification of links among concepts. Although it may be carried out at varying levels of detail and complexity, from minimal synonym
identification to the role analysis in the PRECIS indexing language, relational analysis is at the heart of all exercises in vocabulary control.

Indeed, indexing languages are representations of knowledge. The range of kinds of knowledge represented is limited to concepts, and, moreover, the relations represented between concepts are relatively undifferentiated, usually being confined to synonymy and hierarchical relationships, with all others being lumped together as “related terms.” The representation only of concepts contrasts to the representation of procedures or other dynamic or sensory information in AI.

However, within this restricted scope, librarians have had considerable achievements. They have developed and maintained some very large and complex knowledge bases, and they have in some cases maintained them for over 100 years. Even many thesauri that are considered “modern,” in that they were developed for coordinate indexing systems instead of for card catalogs or for shelving books, have been in use for twenty-five or thirty years. No expert systems have that kind of history. Indeed, few have been maintained at all.

Specific Methods

Some specific methods for KR will now be considered: first, rules and their formalization into first-order predicate calculus; then, two object-oriented methods—semantic networks and frames.

**Procedurally Oriented Knowledge Structures**

The discussion of different KR methods will focus on an example of procedurally oriented information that was initially stated as two rules that might be part of a restaurant selection system (Figure 1). These rules are then translated to two procedurally oriented approaches. It is very natural to express procedural knowledge in such rules. If someone were explaining how restaurant reservations work, he or she would indeed explain it this way. Of course, the rules cannot actually be used as they appear in Figure 1, part (a).

Figure 1, part (b) shows a translation of the rules into first-order predicate calculus, which is familiar to many people from their general education, to show how they can be converted into a formal structure. The use of predicate calculus allows formal theorem-proving techniques to be used in ascertaining the truth of a proposition. Notice that the natural language relations appear as predicates with variables, e.g., Restaurant (x); Patron (y); EatsAt (x,y). Formalisms like predicate calculus delineate the relations inherent in the natural language text and the logical operations among them (AND, OR, NOT, entailment). Moreover, they allow the introduction of quantification, utilizing the universal quantifier “For every” (∀) or the existence quantifier “There exists” (∃).
(a) Knowledge Represented in Rules

Rule 1: If a restaurant accepts reservations and a patron makes reservations at the restaurant, then the patron eats at the restaurant.

Rule 2: If a restaurant does not require reservations and the maximum waiting time of the patron ≤ the average waiting time of the restaurant, then the patron eats at the restaurant.

(b) Knowledge Represented in Symbolic Logic

P1: \( \exists x \ ( \text{Restaurant}\ (x) \& \text{AcceptsReservations}\ (x)) \& (\forall y \ (\text{Patron}\ (y) \& \text{MakesReservation}\ (y,x)) \rightarrow \forall x \forall y \ (\text{EatsAt}(y,x)) \)

P2: \( \exists x \ ( \text{Restaurant}\ (x) \& \neg \text{RequiresReservations}\ (x)) \& (\forall y \ (\text{Patron}\ (y) \& (\text{MaxWaitTime}(y) \leq \text{AverWaitTime}(x)))) \rightarrow \forall x \forall y \ (\text{EatsAt}(y,x)) \)

(c) Knowledge Represented in a Semantic Network

(d) Knowledge Represented in Frames

![Diagram]

Figure 1. Alternative forms of part of a knowledge base to decide whether John will eat at Chez Pierre
Quantifiers are important in eliminating the ambiguity in many natural language statements. For instance, consider the proposition "Every patron eats at a restaurant." When the proposition is taken out of context, it is not clear whether (1) for every patron there exists a restaurant such that the patron eats there, or (2) there exists a (single) restaurant such that every patron eats at it. Quantifiers can express such differences precisely. Higher order logics can also be used to reason about cause and effect or possibilities.

Object-Oriented Knowledge Structures

Turning to the object-oriented approach, semantic networks, as pioneered by Quillian (1969), are structures that link concepts in a graph and may label the links as to type. Those familiar with the PRECIS indexing language will immediately feel at home with the diagram in Figure 1, part (c), where the horizontal links assign roles to concepts and the vertical ones provide hierarchical information. Objects can also have attributes, such as MakeReservation. What must be added is the procedural knowledge that was so easily stated in the rule-based approach. It is represented here as constraints on the kind of individuals that can serve as specific instances of patron and restaurant and is attached to the EatsAt object.
The same information may be translated quite exactly into frames (Figure 1, part [d]). A frame is created corresponding to each object from the semantic network, including the action EatsAt. Each object stores the attributes and constraints that were shown associated with the object. However, each frame also contains the hierarchical links between it and the broader classes of which it is a member, here shown as “Isa,” literally, “is a” slots. Slots in the children of a frame (for instance, the “John” frame is a child of the “Patron” frame) can then be “inherited” from the parent frame, as can default values for these slots. For greater precision, each slot in a child frame could contain a specific “Inherits from” instruction, in case the frame has multiple parents and different slots (attributes) inherit slot fillers (values) from different broader entities. Articles by Susanne M. Humphrey in these proceedings and elsewhere (Humphrey & Kapoor, 1988) illustrate this practice.

As in indexing languages, a list of the narrower terms or specific instances can also be added to a frame, shown in Figure 1, part (d) in the Instances slots. However, these values could also be computed by the system.

The inheritance of slots and slot fillers is very important. Notice that the system has been set up with the information that patrons will make reservations and that the maximum time they are likely to wait is twenty minutes. If the system has no information specifically about John and has no source to ask, then it can continue its processes by having the object for John use the information specified for Patrons as a class as a default. Frame 0004 shows the values inherited from Frame 0003 in brackets. If the system obtains specific information about John, that information overrides the inherited values. This approach is directly analogous to adding narrower terms automatically to a search statement in bibliographic systems. It also provides for considerable space saving, since the default values do not have to be repeated for each object that shares them.

Inheritance cannot be carried out in the rule-based system, even though hierarchical relationships can certainly be expressed, as will be illustrated presently. This limitation clearly makes for considerable inefficiencies in storing information about objects. Notice, on the other hand, that the quantification that was expressed in the predicate calculus example has been lost in the object-oriented approach and, thus, a certain potential for precision in expressing propositions. Nonetheless, the greater part of the information could be expressed in both systems, so both approaches are possible. They can be used exclusively or in conjunction with each other.

One should not conclude this methods section without also briefly mentioning more complex structures than rules or objects. While rules
and objects are powerful ways to present knowledge, they do not help very much in conveying the complicated sequences of events associated with many common activities, such as eating in a restaurant or going to work. Many AI systems, therefore, also incorporate scripts or scenarios, which, though they will have rules and objects as components, provide information about the usual order and type of actions to be expected when some commonly occurring event takes place. The entire role of making reservations in eating at a restaurant would be dealt with in a restaurant script along with many other things, such as the role of waiters and menus and money. For physical systems, qualitative models are also a possible way to represent more complex structures. They will be discussed in more detail presently.

Reasoning vs. Knowledge

As stated earlier, the KB is not the whole of an expert system. In fact, modern practice carefully distinguishes it from other components. In expert systems, the most important of these are usually the interface, which will not be discussed here, and the so-called "inference engine" or reasoning mechanism, which will be briefly reviewed to distinguish it from the KB itself.

Inference processes will be illustrated using a somewhat simpler example than restaurant reservations, but one involving more rules. As shown in Figure 2, suppose one is going to use rules stored in a KB to investigate the perpetually intriguing question of whether Socrates is mortal. How can it be done? While there are various special reasoning mechanisms that are invented to handle particular problems, there are two general and widely used methods for reasoning with a KB: (1) backward chaining, also called "hypothesis-driven" reasoning; and (2) forward chaining, also called "data-driven" reasoning.

An instance of backward chaining is shown in Figure 2, part (a). Knowing through rule R6 that Beings can only be mortal or immortal, the system would proceed to adopt each of these hypotheses in turn and see if either could be proven. In this instance, the system begins with the hypothesis that Socrates is immortal (Trial 1, Figure 2, part [a]). Then, it looks for rules having something about immortals in their consequent (the clause following the "then"). In other words, it looks for rules that specify something about the conditions for being an immortal. Finding R3, it then adopts its antecedents (specified in the "if" clause) and searches to see if they are the consequent of other rules. Since the hypothesis that Socrates is immortal cannot, in fact, be proven, the system then adopts the hypothesis that Socrates is mortal as a second trial and begins looking for rules having something about mortals in their consequent. As shown by the steps in Trial 2, this process eventually leads to R4 and the fact that Socrates is a human, which satisfies a
condition for being an animal (R1) and, therefore, a mortal (R2), and thus proves the hypothesis.

Alternatively, in forward chaining one starts with a known fact—in this case, that Socrates is human—and then tries to conclude that Socrates is mortal. If an antecedent matches the set of known facts, a rule can be activated or "fired." Knowing that Socrates is human would lead to R1—that humans are animals—and from there to R2—that animals are mortals, also answering our question.

In the example given, it appears that forward chaining is much more efficient than backward chaining, which is not necessarily the case. There could be many other rules in the KB about humans or animals that might have been abortively investigated before the proper sequence of rules was found. Either forward or backward chaining may be preferable, depending on the kind of problem being solved.

It is also worth noting that in this very simple example, almost all the relations are hierarchical ones. It therefore also demonstrates very clearly the potential advantages of an object-oriented approach. If the information in these particular rules had been recorded as a set of hierarchically linked objects (Figure 3), the object for Socrates would have inherited the slot "Life-expectancy" and the slot-filler "mortal" from the object for animals, and the question could have been answered by a simple lookup, instead of the search sequence described. Inconsistencies and gaps in the KB would also have been easier to detect and correct.

The example is oversimplified in other ways. All the facts in the KB are presumed to be true and none of them is inconsistent or contradictory. What the system is doing is attempting to establish a chain of facts that would allow it to deduce that Socrates is mortal. But much of human reasoning is not so certain or so strictly logical. One might notice that Socrates was aging and suspect that he was not immortal (abduction) or think that, since all other humans one had observed were mortal, Socrates was probably mortal too (induction). Abduction and induction are perfectly acceptable, practical methods of dealing with the world, even if they are not guaranteed to produce true results.

One can more clearly see this kind of knowledge and its use in the restaurant example above. There, the selection of the matching rule—that a patron will eat at a restaurant if his or her average wait time is less than the maximum wait time of the restaurant—is clearly only a heuristic, a rule of thumb, not a physical law. One of the strengths of knowledge-based systems is that they also attempt to resolve problems in the face of uncertain or insufficient information. One method for augmenting a KB to support such reasoning with uncertainty is to have weights associated with the facts or rules in the KB, usually referred
to as "certainty factors." Such weights must then have rules governing their combination.

Many different rules can be used for combining certainties in this way, such as taking the minimum of two certainty factors or multiplying them like probabilities, if they are on a scale (0 to 1). These rules, unlike the weights, would be part of the inference engine, not of the KB. The boundaries between the KB and the inference engine are, thus, carefully maintained.

Problem Areas

There are, of course, a great many unsolved problems in KR. Three particular problems will be mentioned here: KB quality, general or common sense knowledge, and machine learning.

KB Quality

A number of topics may be included under this heading. One is, of course, the accuracy and validity of the knowledge in a KB and how one determines either of these factors: in short, how one debugs the knowledge in a KB. There are several different problems in this category, the first being determining if the system’s answers correspond to the expert’s in all cases, and the second being the question of whether the expert is right. Even the first, given the nonalgorithmic nature of knowledge-based systems, their ability to modify their own information, and the difficulty in controlling the problems they will be set, presents major barriers. Needless to say, if the area is one, such as bibliographic
searching, in which there is not a single right answer against which to measure the results, the judgment of quality is difficult.

The more subjective the judgments involved, the more challenging it is to apply expert system technology: thus, the seemingly redundant admonition that in order to have an expert system, you must have an expert—or put another way, in order to have an expert system, someone must know how to solve the problem.

**General Knowledge and Common Sense**

One of the principles now recognized by AI knowledge engineers is that it is often useful to distinguish different levels of knowledge: problem specific, domain specific, or general. In an expert system, a considerable degree of performance can often be achieved by including only problem-specific knowledge, if the system is built to solve a very narrow problem and its problem input can be carefully controlled.

However, in many applications it is difficult to limit a knowledge-based system, even an expert system, to problems that require only problem-specific or even only domain knowledge. Without broader knowledge, the range of capabilities of a knowledge-based system is very limited, and the danger that it will be used beyond this range is always present. One pressing problem in AI is how to endow systems with a capability equivalent to the human one (admittedly fallible) of knowing what one does not know. Such a capability, as well as any general extension of system functions, requires the addition of domain or general knowledge.

Furthermore, there are many kinds of AI systems where it is difficult to maintain stringent limits on the problems that will confront the system, for example, in one that supports planning for military operations. To take another example, it is impossible to develop a system to understand fully even quite restricted sorts of text input from uncontrolled sources without having to equip it with much more than problem-specific knowledge. One can, of course, build a system that only attempts to extract and utilize some previously defined information of interest, such as a system that "reads" newspaper stories about terrorist incidents and extracts the basic facts (Lebowitz, 1980). But such a program will ignore or fail to interpret correctly anything that falls outside the scope of its particular filters or its world view.

While such limits are also present in humans, the difference is qualitative as well as quantitative. Such systems have not only a much more limited world view, but also no general principles to bring to bear on new problems. Moreover, they generally cannot learn from their experience or mistakes, since machine learning is still a very young science. Many potentially useful applications for libraries immediately confront the daunting breadth of knowledge required to duplicate the
expert behavior of librarians, who draw on a wide base of general and subject-specialized knowledge of the world, in addition to their own professional expertise and techniques for information description and search.

General knowledge is a particularly severe problem in AI, overwhelming both in the quantity of it that even the most ignorant human possesses, and its variety and complexity. Its quantity suggests cooperative efforts to build general KBs that could be used by multiple systems, such as the Cyc Project to be examined presently. Other cooperative efforts related to AI have also been proposed, for instance, the project advocated by Walker (1989) and others to build cooperatively a large corpus of text for natural language processing and promote the sharing and reuse of lexical resources. Of course, any such effort depends on a significant degree of agreement on the contents, structure, and future use of such a tool, which is not easily obtained at this stage of knowledge-based system development. One might compare the effort to the MARC format, which was a necessary prerequisite for a cooperative effort to convert library catalogs to machine-readable form, but which is an expensive data format because of its comprehensiveness and generality.

The attempt to deal with the complexity of general knowledge has opened up many fascinating problems, some of which have been treated for centuries in philosophy—particularly in logic, ontology, and epistemology, and others of which are relatively new as formal, defined, immediate problems. In the first category are such things as the representation of time, causality, possibility, probability, and belief, and the identification and naming of classes of matter, things, qualities, and actions in the world. In the latter class is the AI problem known as "common sense reasoning," including such subfields as qualitative modeling.

The problem addressed by qualitative modeling is, at base, one of the appropriate levels of detail of knowledge for reasoning for a particular application. Humans, for instance, know a lot about when a particular surface will be slippery relative to what they are wearing on their feet. Although one may sometimes be ambushed in an unfamiliar situation, such as nylon socks on carpeted stairs, humans are fairly good at predicting trouble. One knows about such things as hard, shiny surfaces, liquids on surfaces, ice and the relative traction afforded by bare feet, socks, new shoes, leather soles, rubber soles, cleats, or crampons, but how should this knowledge be represented in a KB? Should the KB have hundreds of specific rules about ice, wet floors, waxed tiles, etc., or, at the other extreme, should it have a set of equations for computing the exact degree of friction between the soles of running shoes and the ice on a sidewalk?
Neither approach seems reasonable. One does not allow for any
generality of observation and the other requires information that is
probably not available and certainly does not represent the way one
ordinarily reasons about the problem. Even if the necessary information
were available, there may be no analytically tractable or computationally
feasible solution.

Qualitative models seek to find a middle ground by reasoning with
qualitative information rather than specific quantities. Substances, for
instance, are hotter or colder, more or less slippery, smoother or rougher
than other substances. Water on a surface increases its slipperiness, but
the exact quantities are not measured or estimated.

Although this example involves common sense reasoning, the same
kinds of issues apply in representing expert knowledge. Problem- or
domain-specific rules are all right as far as they go, but robustness
requires that the system also have some general models to fall back on. Even in scientific areas such as medicine or applied geology, experts
use a mental model with appropriate levels of detail and appropriate
simplifications. A recent stimulating article by W. J. Clancey (1989)
has even suggested that all KBs for expert systems could be thought of as qualitative models, and that such a view would allow their power
to be compared and assessed against a common scale.

The problems of general or common sense knowledge are
particularly crystalized in the continuing debate about the relative roles
of grammar, semantics, pragmatics, and world knowledge in the
understanding of language. The problem, stated succinctly, is that
humans add a lot of background information in interpreting text. Consider a small vignette like the following:

When Joe’s alarm clock went off, he looked out the window.
Everything was covered with ice. He went back to bed.

In interpreting even this very simple story, one adds to the facts stated
that Joe was probably going somewhere because he had his alarm clock
set and because the first thing he did was to look outside. One also
knows that when he saw the ice, he knew he would have a traction
problem; thus, it was dangerous to go out. Also, whatever Joe was going
to do was not worth risking his life for. (For a recent review of the
state of the art of natural language processing, see Allen, 1989.) As this
example clearly demonstrates, the ability to parse sentences and assign
dictionary meanings to words is not by any means sufficient to allow
the interpretation and full understanding of even short pieces of text.
It must fit into some situation, event representation, world model, or
other knowledge structure that gives it context and allows its full
meaning to be extracted. The scripts described above are one approach
to handling stereotyped situations, but many very serious problems
remain.
Machine Learning

Finally, it is impossible to discuss problems in the future of knowledge-based systems without mentioning machine learning. Most people working in the field think that it will be impractical to have AI on a large scale unless machines can learn like humans—from experience, from teachers, or from reading. If one wants a real challenge, think of a machine that can learn from watching television!

There are a number of different kinds of learning, some of which involve generalization or abstraction. To take only one case, consider what is involved in having a machine learn by example. Returning to Joe and his running shoes, one would like the machine to be able to generalize about the outcome of walking on slippery surfaces after it had been presented with a number of instances of accidents occurring to people walking on ice, newly waxed floors, etc. However, this exercise requires that the machine extract the essential property of all the surfaces (let us call it slickness-when-walked-on-with-normal-footwear) and, moreover, understand the cause-and-effect relationship between slickness and the accidents.

This observation, of course, brings us back to our qualitative model. Is it possible for a machine to construct this model for itself, and, if so, on what basis? After all, humans have the actual physical experience of having our feet go out from under us from which to learn. The machine does not have the same kind of sensory input. It must also be able to recognize the common characteristic or set of characteristics in the examples in order to be able to generalize from them. Such recognition is a very difficult task for a machine, and one that is, essentially, classification. Indeed, librarians familiar with automatic classification and numerical taxonomy will be interested to know that these same techniques which were introduced in that field twenty-five years ago are now being tried for machine learning.

A MAJOR RESEARCH PROJECT FROM A LIBRARIAN’S PERSPECTIVE: THE CYC PROJECT

The above section has supplied some of the basic information needed to understand an AI application and to put an advanced research project in KR, such as the one to be discussed in the rest of this paper, in some context. It is now time, therefore, to move to the specific example promised.

The reasons for selecting this example, the Cyc Project at the Microelectronics and Computer Corporation, will quickly be obvious. The Cyc Project is the most ambitious attempt now underway anywhere in the world to build a very large and very general KB. In fact, the
researchers at MCC propose to build a core KB of about 10 million entries which would then be cooperatively expanded to an unknown size. This KB would contain the "consensus reality" that one needs in order to understand everything in a newspaper (including the ads, advice columns, etc.) and everything in a desktop encyclopedia. In other words, it tackles head-on the problem of world and common-sense knowledge which has been previously described as such a barrier to AI. Since many librarians specialize in organizing large collections of very broad coverage, these facts should immediately capture their interest.

One must admit, nonetheless, that it is also a "convenience sample" of one for the author, since the researchers published a substantial book (Lenat & Guha, 1989) on their experience and progress during the first five years of the project. All comments in this talk are based on this source.

In the technical terms discussed above, the Cyc KB utilizes both frames and predicate calculus for KR. The predicate calculus is used to express constraints, such as "Twins are not likely to have the same first name." It is the more powerful of the two representation forms and includes variants on the universal and existence quantifiers discussed above, except that, in this case, the domain over which the quantifiers operate is always specified. In other words, there are no expressions of the type "For every," but only of the type "ForAll<members of a specified set>.''

As was shown in Figure 2, the knowledge that is represented in each slot of a frame can also be considered a predicate. Despite this redundancy, Cyc retains the frame language because it provides a very efficient way to deal with one- and two-place predicates, which constitute the bulk of the information to be stored.

In order to develop the KB, it has been necessary to work out an extensive ontology, that is, to make decisions about what kinds of beings are to be represented in the universe of the system and what relations among them will be recorded. Since this ontology includes about two dozen different classes of things, such as SomethingOccurring, TangibleObject, IntangibleStuff, which have a complicated set of interrelationships, the reader must consult Lenat and Guha for a description. Probably the bulk of the intellectual effort to date has gone into this analysis, which reminds one of similar analyses carried out in the pursuit of universal faceted classification systems in libraries (see, for instance, Dahlberg, 1988). Closely allied with the ontology are the specialized inference mechanisms in the Cyc constraint language, CycL, of which there are more than a dozen.

The system does not use numerical certainty factors for reasoning with uncertainty because, after an initial experiment, the researchers concluded that they tended to lead to too many false inferences. This
problem arises because of the subjectivity involved in assigning highly differentiated weights with no objective standards. (Does anyone recall similar criticisms of manually weighted indexing?) Instead, it uses five truth values: T = default true; 100 = monotonically true; ~ = default false; 0 = monotonically false; and ~ = unknown. "Monotonically true" is assigned to statements whose conclusions must be true if their antecedents are true, for example, "If John is my brother, then we have the same mother and father." If the conclusion, that he is my brother, is not true, then the antecedent must be monotonically false. Things whose truth value is "default" true or false are believed to be true or false only in the absence of contradictory information.

So far as its eventual use is concerned, the project leaders hope that other researchers will build expert systems using the CycL language and the system's development capabilities, and thereby gain access to the Cyc KB and the benefits of the robustness of reasoning that they believe Cyc will eventually supply. Such projects would also extend the Cyc KB with new specialized but compatible information. The project leaders also hope that after enough information has been hand-coded into the system, it will have enough knowledge to be capable of substantial independent learning, say, through "reading" books or newspapers.

Before turning to more substantive comment on the project, it is perhaps useful at this point to try also to compare the scope of this project to some with which librarians are more familiar. In attempting to develop such a comparison in strictly quantitative terms, the author found herself to be very frustrated, since the various figures for the potential size of the system are quite inconsistently expressed in the few places in the book where they are described. However, she eventually stopped berating the authors and reminded herself that Cyc is a high-risk R&D project, not a contract to build a widget. In fairness to the authors, they do provide a specification of what functionality they want the system to have in 1994, which is far more important than exact size of the KB, no matter in what fashion one may choose to measure it.

However, some order-of-magnitude comparisons are possible. Lenat and Guha state in several places in their book that the KB must clearly contain at least millions of frames or their equivalent and tens of millions of pieces of data. The project also expects to devote two person-centuries to building the KB between now and 1994.

In comparing these numbers with, for instance, the Dewey Decimal Classification (DDC) (Dewey, 1989), the author made a rough estimate of the number of basic entries in that scheme (including the tables, but not the index or any synthesized numbers). This figure was in the neighborhood of only 30,000 to 40,000 entries, probably about 1/100th the number for frames contemplated for the KB.
While two person-centuries sounds like a lot of effort, a great deal more than that has been expended on the DDC over the years. One can grant that much of the expended effort has been in maintaining and redoing the scheme; developing it from scratch would be different. Still, the comparison does cause one to wonder about how much can be done, even with the amount of person power proposed. Not only will the rate of adding new entries probably deteriorate from whatever it is at present as the size of the KB grows, but anything being built over ten years' time will have serious maintenance problems before it is completed. These topics will be explored a bit further presently.

Hypotheses about Some Possibly Universal Problems of Large, General Knowledge Bases

One does not have to be a genius to develop a long list of problems such a project will have to solve: genius is required to solve them. Therefore, the exercise to be engaged in here of predicting some of these problems from the experience of librarians and commenting on whether they have arisen in the Cyc Project, and, if so, whether they have been successfully dealt with, is meant constructively and even humbly. The Cyc Project may or may not achieve what it is setting out to do, but its successes and failures will teach us a great deal. There is no way to experiment with large information systems at present except to build them. Moreover, the Cyc Project has an appealingly subversive character, not the least of which is that one suspects the project leaders are having fun. Spending one's days introspecting about why one does not believe certain articles in the National Enquirer might be quite addictive.

On a more serious note, however, there are at least four major barriers that have prevented a breakthrough in improving the effectiveness of large indexing languages and classification schemes beyond their present levels of utility. Since these barriers have arisen for general indexing and classification systems, they could certainly be expected to arise for this much more ambitious project.

The list below may seem strange at first glance because it is very general. There are hundreds of technical problems associated with Cyc, any one of which could generate pages of discussion and debate and any one of which could cause the project to founder. However, such debates are topics for the AI literature, not for this discussion. What the author is reacting to here is the statement that Lenat and Guha feel that the progress they have achieved in the past five years justifies raising their estimate of the feasibility of Cyc from 10-20 percent to 50-60 percent (p. 21). The four following points, listed somewhat facetiously, address why this author thinks the researchers may be overly
optimistic based on the experience of librarians in constructing and maintaining large knowledge bases:

1. The variability of human performance in tasks related to KR and search or "My way, your way, and the Cyc way";
2. The paradox of structuring knowledge or Is more less?;
3. The double-edged nature of the 80/20 rule or The Law of Diminishing Returns; and
4. The inertia of the installed base or The Monster That Ate the Library of Congress.

Variability or My Way, Your Way, and the Cyc Way

Few facts have been more astonishing to information scientists or should give AI researchers more sleepless nights than the repeatedly demonstrated figures for indexer and searcher consistency, or rather inconsistency, in information systems. Much like the participants in several simultaneous games of gossip, a group of well-trained indexers or searchers can begin with the same text or request for information and emerge with less than 20 percent agreement in the outcome of their tasks, once the baseline information or the document being indexed has been conceptualized by the indexer or searcher and the concepts translated to fit within a formal structure.

The problem is not improving consistency. The main difficulty is that we do not know whether these low consistency rates are good or bad (Cooper, 1969). Inconsistency arising from error or complexity in rules, such as is being addressed by the Indexing Aid Project at the National Library of Medicine described by Humphrey in these proceedings, is, indeed, a worthy target for improvement, but clearly identifiable error accounts for a relatively small fraction of the variation. How can we improve consistency without reducing variety, in particular, variety related to linguistic expression, which is so much at the heart of human intelligent behavior? Or is it also desirable to reduce variety and if so, on what points? These questions are the truly hard ones for which we do not have any very good answers.

What do these observations mean for Cyc? First, of course, they raise grave questions about the degree of consistency that can be obtained in the Cyc knowledge-base development effort, even with a high degree of automated support. Lenat and Guha (1989) recognize the potential for inconsistency (p. 21), but one does not have the impression that they understand how large a problem it is. More troubling, however, is that until the KB is used, they probably will not know (1) how inconsistent the database is, or (2) what kind of problems the inconsistency will pose for them. The latter is the more interesting question, but since Cyc is a new sort of venture, it is difficult to speculate about it. Perhaps Cyc will be all too human: that is, it will produce
useful results but have a high failure rate that cannot be self-diagnosed, particularly in the area of associations. The researchers are anxious for outside groups to make use of the Cyc KB, and it seems essential that this use be as early and vigorous as possible. Failure to exercise the KB as it is being built could produce some very unpleasant and expensive surprises.

The Paradox of Structure or Is More Less?

The principal paradox of structure is that it is simultaneously the essential ingredient and the primary barrier to the use of knowledge. At a personal level, everyone recognizes that the organization of his or her personal knowledge must be a key factor in the ability to exist in the world, to utilize sensory input, to interpret experiences and learn, or to use one's memory. Yet that very organization is a filter that can be a barrier to perceiving things that one should perceive or learning things that one needs to learn. A library shelving scheme, for instance, facilitates certain kinds of learning. Nonetheless, for all practical purposes, it prohibits others, presenting the user whose needs do not match its structure with something no better than a random ordering, at least in the worst case.

The broader and more unpredictable the use of a knowledge organization scheme, such as a general library classification, or even a subject database, such as ERIC, the more difficult it is for a high degree of organization to be universally helpful. This is a lesson that librarians think, at least, has been demonstrated even by comparative testing of retrieval systems. Some of the performance problems come just from the additional burden placed on the user by system complexity related to structure, which might be reduced or made less obvious through automated support. But much lies simply in the nature of knowledge, which is highly variable by culture and over time, and of information use, which filters it in many different ways.

How do the Cyc researchers expect to maintain the system's very highly structured and complex knowledge base? The project does have an answer for this problem, namely, that the system will become smart enough so that it can update itself with some coaching. If this cannot work, the Cyc researchers apparently would be among the first to recognize that continued maintenance by the same methods being used to build the KB would be untenable.

To take a topical example, librarians over the world are tearing their hair out considering how to update their systems to accommodate changes in European geography (an exercise they have gone through on several previous occasions during this century), but their problem is minimal beside that of updating the "common sense" information about world affairs, political systems, etc. that should eventually be
contained in Cyc. If Cyc cannot read the newspapers, it is in real trouble—
putting aside for the moment the problem of which newspapers it should
read. In fact, one hopes the project members are saving their dailies,
because the system is likely to be significantly out of date before they
can get it built. Are they going to have to set a cutoff date and issue
Cyc Circa July 1990 as a first release in 1994?

This author is perhaps even more bothered by the language and
cultural problems. Although one of the project leaders is apparently
an Asian immigrant, there does not appear to be any real appreciation
that not everyone may want an embodiment (or is it an “enrulement”)
of a “1991 California/Texas Yuppie-Techie” as their consensus KB about
the nature of the world. Yet no attempt is seen to bring in any broader
perspectives. Aside from the temporal difficulties already discussed, one
need consider only the probable analogs to the cultural bias that has
caused such problems for librarians outside the United States (and
sometimes for those in it) in using the DDC. One thing is certain,
however: if Cyc is built, it will be an amazing artifact. Cultural historians
will have a field day with it, at least if temporal snapshots of it are
archived. Imagine having such a record of 18th century France or 16th
century England!

The Double-Edge of the 80/20 Rule

The 80/20 rule, which has been repeatedly demonstrated to apply
to the automation of things related to language, holds that algorithms
or procedures can be found to handle 80 percent of the input with
20 percent of the effort. On the positive side, if one can identify and
isolate (with a low error rate) the 80 percent of the cases for which
the rules and procedures work well, a large percentage of the processing
is susceptible to automation. However, the qualification to that statement
is not trivial. It may be as difficult to throw an exception out of a
system as it is to handle it correctly in the first place.

The negative side of the 80/20 rule, however, is that 80 percent
of the effort covers only 20 percent of the cases, and this 20 percent
causes the system to become vastly larger and more complex than the
80 percent rules would have led one to believe. As a friend once remarked
from bitter experience, “When you have found the 80 percent algorithms,
you have defined the problem.” Consider, for instance, the Anglo-
American Cataloguing Rules (Gorman & Winkler, 1988). The base rules
occupy a few pages; the rest of this rather lengthy book is taken up
with exceptions. Just to make things worse, as the data or KB grow
in size, not only the absolute number but also probably the number
of types of aberrant cases grows almost without limit. Thus, large KBs
are inherently exponentially more complex than small ones, and such
a system can never handle all cases. Some error must be tolerated. Related
observations have been made by several other speakers at this conference who have addressed the law of diminishing returns in constructing a knowledge base.

The Cyc researchers mention some of these problems. In fact, their identification of the need for an ontology stems directly from the recognition that very large KBs are different. It answers the need they have identified to establish primitives in order to prohibit an infinite expansion of the database, much as librarians have attempted to do from time to time for the same reason. In addition, they have attempted to identify and address a range of problems before beginning any large-scale development in order to reduce backtracking.

Finally, they are attempting to encode information on a level of generality that would not bog them down in too much incidental detail. For instance, they would not record how to deal individually with the situation where a bag lady has scratched one's car with her cart vs. the situation where the owner of a Mercedes-Benz has rear-ended one's rattle trap. Cyc intends, instead, to record general principles about the right to try to recover when someone does damage to one's property and the notion that in order for someone to be able to give anyone something under any circumstances, they must have it—"it" in this case being money (Lenat & Guha, 1989, p. 22). This choice of level is directly related to the qualitative modeling problem previously discussed.

Nonetheless, this author is left with the nagging feeling that they have seriously underestimated the 80/20 problem. The book contains a great many descriptions of solutions that appear to be "80 percent algorithms." One has no way of knowing from the book how many other kinds of procedures have been incorporated into the system or what the researchers intend to do about handling exceptions, but there is cause to wonder. This area will be an interesting one to watch in future publications.

The Inertia of the Installed Base or the Monster that Ate the Library of Congress

Lastly, one of the problems with large systems is that they are large and one of the problems with systems that are used is that they are used. Both these sad facts of life tend to make it difficult to keep a large system up to date or make improvements to it, whether or not it is being updated automatically. This problem lies partially in the future for Cyc, but as soon as it contains a significant amount of data, design changes will become expensive. This fact is explicitly recognized by the Cyc researchers, as has been previously mentioned.

The conservative drag arising from the widespread use of systems is also well known to librarians, as it is to developers of commercial
software and hardware. If Cyc is widely used as a component of other systems or as a host for them, the users will expect that updates will not seriously disrupt their own systems, which will probably inhibit major changes. Also, the project cannot wait too long before it acquires users. In order to test Cyc thoroughly, real systems need to be built with it, but it will be difficult to get developers to use it when it is only a laboratory product.

Clearly, we have another conundrum here, which suggests to this author that Cyc may become a test bed rather than a living system. As a test bed, it could have a vital role in expert system and natural language research even if the knowledge in it were frozen at a certain date or if areas of its knowledge were never completed.

The Cyc Project and Libraries

If the preceding remarks have sometimes sounded negative, this discussion of Cyc can close on a more positive but quite appropriate note by considering what a Cyc-like KB could do for libraries. Cyc in its projected 1994 form would have general knowledge of the world equivalent to that, say, of a high school student. It would be able to do some fairly sophisticated reasoning with that knowledge and would have at least a limited ability to learn from generally available external sources, such as textbooks or newspapers. If such a system existed, it might, for instance, be able to provide the basis for a natural language front-end for the Sears List of Subject Headings (Rovira & Reyes, 1986) that really could search and reason in a humanlike fashion. It would probably have a deep understanding of the vocabulary and concepts represented in a list of that size and generality, but not something with a high percentage of specialist terminology. However, with a basic ability to learn, it might be extended for special purpose uses.

It is sobering to think that the degree of effort represented by Cyc might be required to get a very intelligent information retrieval system even to this point of development, but it may be true. The gap between word-matching and deep understanding of language is a very large one and one that will probably only be bridged as part of large, cooperative development efforts in which libraries might serve as participants as well as beneficiaries.

CONCLUSION

While most papers in these proceedings focus on what AI can do for libraries, this one has attempted to show some of the many parallel problems between KR on a large scale and the problems of designing,
developing, and maintaining large indexing languages and classification schemes. These connections are becoming better recognized. The teaching of AI in library schools is one such indication, and the recent founding of the International Society for Knowledge Organization by Ingetraut Dahlberg and others is also a step toward providing a forum for fruitful interchange of ideas between AI researchers and librarians. Indeed, the topics mentioned here are only some possible common interests. Just as many people trained in library science have become closely involved with data modeling and database design, many librarians could contribute to AI in general, whether through experience in building systems for libraries or through working on other applications. Both fields will benefit if the connections can be strengthened.

REFERENCES


