Workshop on Model Based Accelerator Controls, BNL, 8/17-18/87.

BNL 40592

BNL--40592

DE88 004863

THE REPRESENTATION OF KNOWLEDGE WITHIN MODEL-BASED CONTROL SYSTEMS*

D.P. Weygand and R. Koul
Brookhaven National Laboratory
Upton, New York 11973

Introduction

The ability to represent knowledge is often considered essential to build systems with reasoning capabilities. In computer science, a good solution often depends on a good representation.[1] The first step in development of most computer applications is selection of a representation for the input, output, and intermediate results that the program will operate upon. For applications in artificial intelligence, this initial choice of representation is especially important. This is because the possible representational paradigms are diverse and the forcing criteria for the choice are usually not clear in the beginning. Yet, the consequences of an inadequate choice can be devastating in the later state of a project if it is discovered that critical information cannot be encoded within the chosen representational paradigm. Problems arise when designing representational systems to support any kind of KNOWLEDGE-BASED SYSTEM, that is a computer system that uses knowledge to perform some task. The general case of knowledge-based systems can be thought of as reasoning agents applying knowledge to achieve goals.

Artificial Intelligence (AI) research involves building computer systems to perform tasks of perception and reasoning, as well as storage and retrieval of data. The problem of automatically perceiving large patterns in data is a perceptual task that begins to be important for many expert systems applications.

Most of AI research assumes that what needs to be represented is known a priori; an AI researcher's job is just figuring out how to encode the information in the system's data structure and procedures.

Knowledge

Often the questions are asked: "What kind of knowledge is needed to behave knowledgeably?"[2] "What things do we know about?" To approach these questions, following is a list of types of knowledge that might need to be represented in AI systems.

*Work performed under the auspices of U.S. Department of Energy.

A. Objects. Typically, we think of knowledge in terms of facts about objects in the world around us; e.g., birds have wings.

B. Events. We also know about actions and events in the world; e.g., Mary married John in 1985.

C. Performance. A behavior like riding a bicycle involves knowledge beyond that of "object" and "event"--knowledge about how to do things, the performance of skill.

D. Meta Knowledge. Knowledge about what human beings know is called meta-knowledge. For example, we often know the extent and origin of our knowledge of a particular subject, about the reliability of certain information or about the relative importance of specific facts about the world. Meta-knowledge also includes what we know about our own performance as cognitivie processors; our strengths, weaknesses, confusability, and levels of expertise in different domains.

Using Knowledge

The most important consideration in examining and composing knowledge representation schemes is the eventual use of the knowledge.[3] The goals of AI systems can be described in terms of cognitive tasks like recognizing objects, answering questions, and manipulating robotic devices. But, actual use of the knowledge in these programs involves three stages:

1. acquiring more knowledge,
2. retrieving facts from the knowledge base relevant to the problems at hand, and
3. reasoning about these facts in search of a solution.

We usually think of learning as an accumulation of knowledge, but it involves more than the addition of new facts in our brains. Knowledge acquisition involves relating something new to what we already know.

We retrieve knowledge by determining what knowledge is relevant to a given problem which becomes crucial when the system "knows" many different things. Humans are incredibly proficient at this task and many representation schemes that have been directly concerned with this issue have been based on ideas about human memory.

When the system is required to do something that it has not been explicitly told how to do, it must reason. It must figure out what it needs to know from what it already knows. For instance, suppose an information retrieval program "knows" only that robins are birds and that all birds have wings. Keep in mind only that it contains data structure and procedures that would allow it to answer the questions. If we then ask it, "do robins have wings?", the program must reason to answer the query. In problems of any complexity, the ability to do this becomes increasingly important. The system must be able to deduce and verify a multitude of new facts beyond those it has been told explicitly.

Following are the different kinds of reasonings or representations one might imagine:

1. Formal Reasoning or Logical Representation.

2. Procedural Representations.

3. Production Systems.

4. Direct (Analogical) Representations.

5. Frames.

6. Meta-knowledge.

Logical Representation (Formal Reasoning)[4],[5]

Formal reasoning involves the systematic manipulation of data structures and deduction of new ones following the pre-specified rules of inference. Mathematical logic is the archetypical format representation.

The classical approach to representing knowledge about the world, e.g., "all birds have wings", is formal logic.

$$\forall x \text{ birds } (x) \rightarrow \text{has\_wings } (x)$$

The advantage of formal logic as a representation scheme is that there is a well-defined set of rules (Rules of Inference) by which facts known to be true can be used to derive new facts also known to be true.

There exists two forms of logic, propositional logic and predicate logic.

First order logic allows quantification over individuals, but not over predicates. First order predicate logic has been popular in AI research but theorem proving, i.e., deriving new facts from old using the Rules of Inference, can be mechanized. In addition, first order predicate logic is sound (impossible to prove a false theorem) and complete (any true theorem can be proved).

Logic has strengths and weaknesses. Firstly, it is a natural way to express certain types of knowledge. Consider "all birds have wings" and "robins are birds" combine to infer "robins have wings". Logical syntax is clear and precise, lending itself to automated manipulation. Logic lends itself to expansion, as new facts are learned or derived, since assertions are independent of each other.

But, there are significant problems. Firstly, the representation of knowledge is separate from the heuristic part. First order logic does not allow direct representation of theorems about theorems. Thus, heuristic knowledge is not easily represented in a formal logic system. The standard way of proving theorems (refutation) becomes badly bogged down when the amount of knowledge grows via the so-called combinatorial explosion.

STRIPS[6]

STRIPS is a problem-solving program which plans a series of goals for a robot to achieve in order to accomplish some task. The world model consists only of rooms, doors, and boxes.

STRIPS works by searching a space of world models to find a model in which the desired goal is achieved. Its state space representation consists of a world model and a list of goals to achieve.

Goals are represented as formulas in predicate calculus, for example:

NEXT_TO (ROBOT,BOX1)

interpreted to mean that the robot should be in a room adjacent to a room containing BOX1).

World models are expressed as clauses:

IN_ROOM (ROBOT,ROOM1)
CONNECTS (DOOR1, ROOM1, ROOM2)

Given a new goal, the programs use a resolution-based theorem prover to see if the goal is satisfied in the current world model. In general, the proof fails, and the program switches to a means-end analysis to determine differences between desired goal state and current world model, and chooses operators which will minimize this difference.

Production Systems

Production system describes several different systems based on the idea of condition-action pairs (productions).

In general, a production system consists of three parts; the rule-base, the current context or state, and a controller. Production systems are good tools for domain-specific expert systems. They provide a good "database" for representing an expert's heuristic information, and they are relatively efficient and accurate. Rules may be added, deleted, and changed independently. Uniformity of the rule representation (together with rule independence) allows the system to analyze its performance with respect to the validity of rules. Uniformity may allow the system to rewrite rules.

Production systems are best suited for domains where knowledge is diffuse, i.e., consisting of many independent facts or actions, and domains where knowledge is separate from where it will be used.

EMYCIN[7]

EMYCIN (domain-independent MYCIN) provides an example skeleton of an expert system. It is particularly well suited to deductive problems, e.g., fault diagnosis.

Knowledge Representation

Domain-specific knowledge is represented by production rules. The rule language is as follows:

RULE: = IF <antecedent> THEN <action> ELSE <action>

<antecedent>: = AND's and OR's of some conditions
or predicates of <context>

<context>: = [<attribute> <object> <value>]

<action>: = <consequent> OR <procedure>

<consequent>: = [<associative triplet> <certainty factor>]

A context tree provides some of the inheritance features of frames.

Certainty factors associated with the context provides a means of handling uncertainty. Predicates may evaluate to TRUE (with a cut on certainty) or may provide fuzzy-set functions that indicate degree of truth. (AND returns minimum certainty values, OR returns the maximum.)

The action part of a rule consists of updating certainty factors, or executing attached procedures.

The EMYCIN Inference Engine

Basic control strategy is backward chaining, its initial goal to evaluate the value of a top-level context. To achieve this, it retrieves a precomputed list of rules whose consequents are known to bear on a particular goal; it systematically applies the rules until the certainty is established or the rule list is exhausted.

DENDRAL[8]

DENDRAL, an expert system which has its beginnings in 1965, identifies candidate molecular structures from mass spectral and nuclear magnetic resonance data. Its performance is characterized as efficient and accurate on a very limited domain, and consequently is moving into a commercial environment.

DENDRAL's knowledge came from hand-crafted expert knowledge. Although it does not reason of basic chemical principles, an effort was spent on META-DENDRAL, a pre-program which attempted to derive some of DENDRAL's rules from basic principles.

Knowledge is represented in two forms. Candidate molecular structures were generated procedurally from special code, the spectrum evaluator was a production system with coded rules. Thus, knowledge acquisition was achieved via re-programming or rule editing.

Procedural Knowledge[3]

Procedural reasoning uses simulations to answer questions, and solves problems. For example, when we use a program to answer "what is the sum of 3 and 4?", it uses or "runs" a procedural model of arithmetic.

Humans generally go about their common tasks through pre-formed plans or procedures. Actually searching the space of possible actions (the basis of most AI programs) is rare. For example, consider the task of driving to work; in each of us the plan for solution has been previously determined, not derived each morning again. However, we may often be asked to modify our plans, generally in a piecemeal fashion (detour in the road, car won't start, etc.).

Procedualism argues that knowledge about a domain is intrinsically tied to the knowledge about how the knowledge is to be used. In addition, much of what we know is purely procedural. For example, if during the solution of some problem you need to know the sine of 37°, your knowledge consists of knowing how to use your calculator to compute the sine of 37°, but the calculator, via a procedure, actually knows how to compute the sine of 37°.

Procedural representations have advantages, primarily related to efficiency. Domain-specific heuristic knowledge is easily represented. Also, side-effects of actions, a problem in all systems, is most easily handled in a procedural representation, since the procedure which takes the action may update the database caused by side effects. This is of particular importance in a large system which spans various domains.

On the negative side, knowledge in procedures is difficult to access and change. It is extremely difficult for a system to analyze and change its behavior.

Frames[9]

Frames provide a structure (framework) for representing knowledge. Frames support various features which makes the organization of knowledge more simple.

Frames consist of information about objects. For a given frame there are various slots, for each slot there are various facets, for each facet various datum.

Data may be inherited (say through the A-KIND-OF slot).

Requests for data can activate procedures (demons) (say through the IF-NEEDED facet).

Addition of new data, or changing old data, may activate demons (say through the IF-ADDED facet).

Thus, underlying the declarative nature of frames is a procedural structure, through demons. When demons are activated, the attached procedures provide a means of choosing appropriate methods (to the current context). General problem-solving methods may be aided by domain-specific heuristics included procedurally at the slot level.

Direct Analogical Representations

Reasoning by analogy seems to be a natural mode of thought for humans, but so far difficult to accomplish in AI programs.

Direct representations is a class of representation which represents knowledge in a particularly natural way. An example is a street map--distance between points in the real world correspond to distances between analagous points of the representation.

Compare this to propositional forms of representation, where proximity in the database has no connection to actual location of points.

A good example of direct representation is Gelernter's Geometry Proving Machine, an early automated theorem prover.[10] This program proved simple theorems in Euclidean Geometry, and relied heavily on a diagram to guide the proof, much as a high school student first draws an appropriate diagram to solve a geometry problem.

In Gelernter's case, the diagram was used as a powerful heuristic--any hypothesis is false if it is not true in the diagram.

The diagram was also used to establish obvious facts; for example, ordering of points. The diagram was also used to construct new items, like line segments, triangles, etc., if necessary.

A strong advantage of analogical representations lies in the difference between observation and deduction. Observation of facts, in many cases, can be achieved quickly and easily. This is typified in Gelernter's program--before a proof would be attempted, quick observation was used as a pruning heuristic.

In model-based control systems (particularly where models are at best approximations), this may be where the model best fits in--as a primarily pruning heuristic.

Also, an accelerator or beam line also serves as a analogical representation itself, of itself, for regions where the models break down.

Summary

Control of any complicated device is both complex and compound. A truly expert system designed to emulate human control, with the current AI technology, will almost certainly require various subsystems, each optimized with its own tailored representation. In addition, there will need to be a master system which is capable of comprehending the results of each subsystem, with the charge of correlating conclusions.

REFERENCES

1. Wood, W.A., "Important issues in knowledge representations", Proceedings of the IEEE, Vol. 74, No. 10, 1986.

2. McCarthy, T., Hayes, P.J., "Some Philosophical Problems from the Standpoint of Artificial Intelligence", Edinburgh University Press, pp. 363-502. D. Michie and B. Metzler editors.

3. Barr, A. and Feigenbaum, E.A.," The Handbook of Artificial Intelligence", William Kaufman, Inc., Vol. II, pp. 143-216.

4. Winograd, T., "Understanding Natural Languages, New York Academic Press, 1972.

5. Hayes, P.J., In defense of logic, IJCAI, Vol. 5, pp. 559-565.

6. Fikes, R.E., Hart, P. and Nilson, N.J., Learning and executing generalized Robot Plans, Artifical Intelligence Vol. 3, pp. 251-288, 1972.

7. Shortliffe, E.H., Computer-based Medical Consultation: MYCIN, American Elsevier, New York, 1976.

8. Lindsay, R.K., Buchanan, B.G., Feigenbaum, E.A., Lederberg, T., Application of Artificial Intelligence for Organic Chemistry, The DENDRAL Project, McGraw-Hill, New York, 1980.

9. Minsky, M., A framework for representing knowledge, Psychology of Computer Vision, P.H. Winston, editor, McGraw-Hill, New York, 1975.

10. Feigenbaum, A.E. and Fledman, T., Computer and Thought, New York, McGraw-Hill, 1963, pp. 134-152.

DISCLAIMER