4

BNL 40314

Europhysics Conf. on Control Systems for Experimental Physics, Villars-Sur-Ollon, Switzerland, 9/28-10/2/87.

Conf-8709151--9

THE IMPLEMENTATION OF ARTIFICIAL INTELLIGENCE IN CONTROL SYSTEMS*

R. Koul and D.P. Weygand
AGS Department, Brookhaven National Laboratory
Associated Universities, Inc., Upton, New York 11973 USA

BNL--40314

DE88 003497

The current emphasis in the Artificial Intelligence (AI)[1-3] field is on knowledge based expert systems, and the corresponding fields of knowledge engineering. Systems which lack domain-specific knowledge are unable to perform anything but trivial tasks, while sophisticated expert systems have the promise of at least equaling the performance of a human expert.

In AI, knowledge representation is the combination of data structures, and the procedures which operate on these data structures, so as to lead to knowledgeable behavior; we do not in general refer to the data structures as knowledge, but rather merely as facts. We consider the combination of facts and procedures as the knowledge of the system.

The ability to represent knowledge is often considered essential to build a system with reasoning capabilities.[3] In computer science a good solution often depends on a good representation, thus the first step in software development is the selection of a proper representation. For AI applications, this choice is particularly important, and, in fact, some researchers do not segregate AI from the representation problem.

In an attempt to classify knowledge, we say that knowledge about a domain consists of facts about the domain, rules for predicting changes,

There are significant problems; the representation of knowledge is separate from the heuristic part. First order logic does now allow direct representation of theorem about theorems. Thus, heuristic knowledge is not easily represented in a formal logic system. In addition, the standard way of proving theorems (refutation) becomes badly bogged down when the amount of knowledge grows in a so-called combination explosion.

Production Systems

Production systems describe various 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 its rules. Uniformity may allow the system to rewrite rules.

Production systems are best suited for domains that knowledge about a domain consists of facts about the domain, rules for predicting changes, consequences of actions, prediction of unobserved facts, etc., and finally meta-knowledge (both meta-facts and meta- rules), that is knowledge about how to use knowledge.

An intelligent agent is an entity that knows facts (and meta-facts) about its domain, perceives new or changing facts, reasons about these facts, acts accordingly, and finally, learns new rules based on its actions. This suggests that a knowledge representation scheme must support perception, reasoning, planning, control and learning.

Logical Representation or (Formal Reasoning)[1,2,4]

The classical approach to representing knowledge about the world, e.g. "All birds have wings" would translate into the mathematical formulas;

$$\forall x. \text{bird}(x) \rightarrow \text{has wings}(x).$$

The advantage of formal logic as a representation scheme is that there is a well defined set of rules (Rules of Interface) 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 primarily because theorem proving i.e., deriving new facts from old using the rules of interference, 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).

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

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.

Procedural Knowledge

Humans generally go about their common tasks through preformed 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, generally this is a predetermined route, not derived anew each morning.[5]

However, we may often be called upon to modify our plans, generally in a piecemeal fashion (detour in the road, car won't start, etc.). Procedurism 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. Procedural representations have advantages, primarily related to efficiency. Domain-specific heuristic knowledge is easily represented, and side effects of actions, a problem in all systems, is most easily handled procedurally, since the procedure which takes the action may update the database and account for 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 procedural systems to analyze and change their behavior.

Frames[2,6]

Frames provide a convenient structure 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 there are various datum. Data may be inherited (say through an A-KIND-OF-SLOT). Request or additions for data may activate procedures (demons) (say through IF-NEEDED facet). Old data, may activate demons (say through the IF-ADDED facet).

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

Direct Analogical Representation

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

A good example of direct representation is Gelerntel's Geometry Proving Machine[7], an early automated theorem prover. This program proved simple theorems in Euclidean geometry, and relied heavily on a diagram to guide the proof, much as a

As modern accelerators are becoming more complex day by day, the operators role is becoming more significant. The reliability and performance of the operator becomes a crucial factor in fast commissioning and start-up, and continued stable operation at optimum efficiency. Model-based control is one of the techniques whose aim is to improve the reliability and enhance the capabilities of the operator, and therefore of the whole control system. The main aim of such systems will be to shift control from operators to computers.

GOLD is a generic orbit and lattice debugger program developed by Martin Lee (SLAC) and Scott Clearwater (LANL)[9]. This consists of

a) A Lattice modelling program (comfort)
b) a beam simulator (plus)
c) graphical workstation environment (VAX)
d) Expert System (ABLE).

There are other modelling techniques which are used at the Los Alamos National Laboratory[10]. One of these is object-oriented programming (OOP). This encourages developing a computer model which more closely resembles the real-world problem. This system uses a symbolic model, implemented in the Knowledge Engineering Environment (KEE) expert system shell. Such a model has casual relationships built in, so actions leading up to an event can be easily described.

At Brookhaven National Laboratory[11], we are developing an expert system to aid in accelerator control. Considerable attention was directed to the representation of knowledge. One must represent the broad range of interrelated information an expert knows about the domain. One needs a symbolic language capable of dynamic storage allocation, just as automated theorem prover. 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 this case, the diagram was used as a powerful pruning heuristic. Any hypothesis was considered false if it was not true in the diagram. The diagram was also used to establish opionions and 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 representation lies in the difference between observation and deduction. Observaton of facts, in many cases, can be achieved quickly and easily.

Architecture

Conventional accelerator control systems are collections of hardware and software intended to[8]:

a) Receive operator requests for appropriate changes in current and magnet settings.
b) Convey information about the state of the beam.
c) Performs similar functions for other subsystems like RF, vacuum, etc.

The conventional software has acquired in time "user friendliness". The operator is the key component of such systems when the machine is being commissioned for different modes of operation. If unexpected behavior is observed, the operator is expected to take appropriate action to bring things back to normal. At this stage the appropriate action of the operator depends upon the operator's knowledge of that particular accelerator.

knows about the domain. One needs a symbolic language capable of dynamic storage allocation, just as an expert's knowledge is both symbolic and dynamic. Thus, expert systems are differentiated from high performance, special purpose programs by their use of dynamic symbolic data representations and symbolic reasoning.

The most important issue which we faced in constructing an expert system for beam control was the representation of knowledge. Beam control knowledge may be divided into three domains. The first is the control segment, i.e., how does one actually change or examine the state of a device. The second segment is the knowledge of how devices effect the beam, i.e., what the specific device's relationship is to the beam. Finally, there are the rules governing how one varies the state of various beam devices to achieve a desired effect or state.

The very lowest level of knowledge is the set of control processes. Within the control processes are embedded all the details of how to control a particular device, i.e., how to change the state of the device or read back the current state of the device. We have developed the control system around groups of controllable entities which we call logical devices. Ultimately, only logical devices are controllable. It is the domain of higher levels of software to create abstractions constructed out of logical devices. We group all knowledge concerning the control of logical devices into the so-called control segment.

There are two types of data within the control segment. One is the software which formats the messages to be sent to devices, and the other is static data about logical devices. Device messages are entirely implemented in C, with a formal fixed syntax.

We next considered knowledge of accelerator devices at a higher level, that is with regard to the functionality. The word device now takes a wider meaning. It may not simply be a logical device with a single set point, but rather may be a set of devices which together form a single function. However, before we consider knowledge of the functional devices, we describe our use of frames, as frames will be used as the representations of higher levels of knowledge.

A frame provides a structural representation of an object or class of objects. In fact, frames may be used to represent more than just objects and classes, but rather any abstract concept. For example, the rules that form the basis of a rule-based system may themselves be represented as frames. One frame may represent a specific beam profile monitor (harp), while another frame may represent data common to all harps. A device's functionality may be represented in its frame, either directly or inherited through its A-KIND-OF slot. Most knowledge is procedural which is attached to the frames via demons.

Frames representing devices which are sensory in nature must have slots which identify the parameter(s) that a device is sensing. For example, there are processes running at all times which obediently report that a device has fallen out of tolerance, without any comprehension of the consequences of this particular device being out of tolerance. In fact, the only task of these processes is to inform a cognizant human, i.e., an accelerator operator, that a device is malfunctioning so that the human may assess the significance of the failure to take appropriate action. The expert system however, may recognize the impact of a device's malfunction.

References

1. T. Winograd, Understanding Natural Languages, New York: Academic Press, 1972.

2. T. Winograd, Frame Representation and the Declaration/Procedural Controversy, Bobron and Collins, 1975, pp. 185-210.

3. W.A. Woods, Important issues in knowledge representation, Proceedings of the IEEE, Vol. 74, No. 10, 1986.

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

5. M.P. Georgeff and A.L. Lansky, Procedural knowledge, Proceedings of the IEEE, Vol. 74, No. 10, 1986.

6. M. Minsky, Psychology of Computer Vision, New York, McGraw-Hill, P. Winston, Ed., 1975, pp. 211-277.

7. E.A. Feigenbaum and T. Fledman, Computers and Thought, New York: McGraw-Hill, 1963, pp. 134-152.

8. S. Sidhu, "Model-based accelerator controls, what, why and how", presented at the Model Based Accelerator Controls Workshop at Brookhaven National Laboratory, Upton, NY, August 16-18, 1987. (in press)

9. M. Lee and S. Clearwater, "Generic orbit and lattice debugger", presented at the Model Based Accelerator Controls Workshop at Brookhaven National Laboratory, Upton, NY, August 16-18, 1987. (in press)

the failure to take appropriate action. The expert system however, may recognize the impact of a device's malfunction.

While the expert system is currently only under development, it is now able to accept some of the more mundane tasks of accelerator control. It is beginning to take over archiving functions and is capable of some simple tuning. In addition it is beginning to incorporate accelerator modelling programs into its repertoire with the intent of applying its expertise to synchrotron control.

Accelerator Controls Workshop at Brookhaven National Laboratory, Upton, NY, August 16-18, 1987. (in press)

10. D.E. Schultz, and R.R. Silbar, "Merging AI and numerical modelling for acçelerator control", presented at the Model Based Accelerator Controls Workshop at Brookhaven National Laboratory, Upton, NY, August 16-18, 1987. (in press)

11. D.P. Weygand, "Artificial intelligence and Accelerator control", presented at the Particle Accelerator Conference, Washington, D.C., March 16-19, 1987.

DISCLAIMER