Knowledge Acquisition for Knowledge-Based Systems Workshop, Banff, Canada, Nov., 1986

Knowledge Acquisition for Fault Isolation Expert Systems

Kenneth De Jong
Computer Science Department
George Mason University
Fairfax, VA 22030
and
The Navy Center for Applied Research in AI
Code 5510
Naval Research Laboratory
Washington, D.C. 20375

ABSTRACT

For the past few years we have had the opportunity to explore the use of AI and expert system technology in a setting in which diagnostics aids for a large number of complex man-made systems are required. We have built (and discarded) several prototype fault isolation shells in the process of understanding how this technology can be usefully applied. We feel that we have now evolved an architecture which is well suited for this task and incorporates a powerful and highly visual interactive knowledge acquisition system.

This paper will provide an overview of the architecture of our latest prototype, describe our experiences with having Navy labs use the knowledge acquisition system, and summarize our plans to further automate the knowledge acquisition process.

INTRODUCTION

The expert system technology from the AI community is being applied to a wide variety of problems including fault isolation in complex man-made systems. The Navy has been interested in and supportive of the development of fault isolation expert systems which can improve the quality of their maintenance and troubleshooting activities. As an example, Navy technicians on aircraft carriers may be responsible for troubleshooting several hundred different (sub)systems for which he/she has had varying amounts of training (frequently little or none). To compensate, the Navy has and continues to invest heavily in automatic test equipment (ATE) to aid or replace these technicians. The quality of the "test programs" which drive these ATE stations varies dramatically in spite of a uniformly

high cost to acquire them.

Although it is tempting to leap to the conclusion that one could significantly improve this sort of troubleshooting activity with reasonably straightforward applications of current expert system technology, there are several aspects to the problem which raise significant technical issues. First, with several hundred different systems to maintain, it seems infeasible to think in terms of independently developed expert systems for each one. Rather one thinks in terms of a more general fault isolation shell providing a common knowledge acquisition/representation scheme for use with all subsystems. However, even with this level of generality, there are still several hundred knowledge bases to be built, debugged, and maintained in a context in which there can be considerable overlap and/or similarity in the content of many of the knowledge bases. These observations strongly suggest the development of a sophisticated knowledge acquisition system which can be used to facilitate the construction of a new knowledge base for a specific system in a variety of ways including re-using and/or adapting existing knowledge modules.

Compounding the problem of applying current expert system technology is the fact that, for many of the subsystems being maintained, there is little human expertise in the traditional sense of finding someone who is good at fixing a particular subsystem and capturing his/her knowledge in a set of associative rules. Rather, technicians depend heavily on the structural and functional descriptions contained in the technical manuals of the many subsystems they attempt to maintain. This suggests that simple rule-based architectures are not likely to be sufficient for the task at hand.

For the past few years we have had the opportunity to explore the use of AI and expert system technology in this setting. We have built (and discarded) several prototype fault isolation shells in the process of understanding how this technology can be usefully applied. We feel that we have now evolved an architecture which directly addresses the issues discussed above. In particular, we are using component networks with "local" rule bases as the means of building a "causal" model of the subsystems a technician is required to maintain. This has allowed us, among other things, to develop a highly visual and interactive knowledge acquisition system with access to existing (frequently generic) causal descriptions which can be "pulled in" and integrated into the knowledge base currently under construction. This form of knowledge representation also provides us with a framework for automating the knowledge acquisition process even more in two distinct ways. First, there is a significant amount of "off-line" reasoning that can be done with a causal model to infer (and incorporate) higher level associative rules of the form typically formed by expert technicians when very familiar with particular subsystems. Also, feedback from the fault isolation process can be used to refine, correct, and identify problems in the knowledge base.

The following sections will provide a brief overview of the architecture of our latest prototype (unglamorously named FIS, for Fault Isolation System), describe our experiences with having Navy labs use the knowledge acquisition system, and summarize our plans to further automate the knowledge acquisition process.

THE STRUCTURE OF A FIS KNOWLEDGE BASE

A great deal has been written about the design and implementation of diagnostic expert systems in general with much of the initial experience and "common wisdom" coming from applications in the medical domain. Although we benefited greatly from this body of accumulated knowledge, we felt that there were two important properties of our particular task domain that needed to be addressed and incorporated into the underlying design of FIS. First, since we are focusing our activities on fault isolation in man-made systems, deeper knowledge in the form of plans, schematics, principles of operation, etc. is available in addition to any attempts at forming a set of high level diagnostic rules. It was clear from the start that good technicians rely heavily on both kinds of knowledge when fault isolating. The second important characteristic of our domain of application was that most of the Navy fault isolation expert systems would have to be built for systems for which there was little human diagnostic expertise or experience, since in the ATE world automatic test equipment is delivered simultaneously with new systems.

These observations led to the design of a multi-level knowledge representation (and associated evidential reasoning mechanism) which is described in more detail elsewhere (De Jong 1984, Pipitone 1984, and Pipitone 1986). For our purposes, a brief description of the structure of the knowledge base will suffice in order to understand the knowledge acquisition issues in this context.

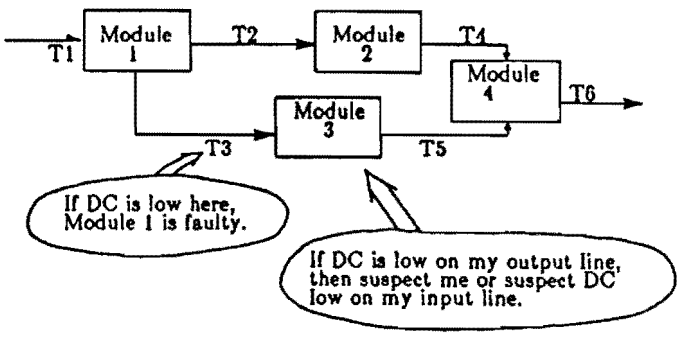

Figure 1: The Structure of a Simple Causal Model

We have chosen to represent deeper knowledge about man-made systems in terms of a qualitative causal network model in which nodes represent replaceable modules, arcs express relationships between modules, and arc labels indicating points at which evidence-gathering tests might be made. Causal knowledge is represented as collections of causal rules attached to modules and test points. Figure 1 gives a partial visualization of a causal model for a simple system containing four replaceable modules, six test points, arrows indicating some dependency relationships (information flow), and several causal rules. Notice that there is no *a priori* commitment to modeling a system at a particular level of detail. The evidential reasoning mechanism works at this level of abstraction regardless of what a replaceable module really is (a sub-system, a card, a gear) and what kind of system is being repaired (mechanical, electrical, optical). In addition, hierarchical relationships are easily represented by treating an object as a replaceable module at one level and as a "system" at a lower level with its own replaceable components.

Qualitative causal rules are attached to this dependency network in two ways. First, each module has a local causal rule base describing how that module behaves in isolation (independent of their placement in a particular system). Figure 1 illustrates one of several such rules attached to module 3. In the electronics domain, module 3 might represent an amplifier and have a collection of causal rules of this sort which are generic and inherited by every instance of a replaceable amplifier module. Figure 1 also illustrates that causal rules can also be attached to test points and generally represent configuration-specific knowledge of the sort an experienced technician might evolve over time as he/she accumulates experience with a particular class of systems. One of the important features of FIS is that it will uses causal rules of this second more global type if to improve the rate at which fault isolation occurs (measured in terms of the number of evidence-gathering tests required), but is quite capable of fault isolation in their absence using only rules associated with the local behavior of modules.

There are other kinds of information which, if available, can be included in a knowledge base such as *a priori* failure rates of modules, indications of the relative costs of making tests, and module replacement costs. FIS will use this information if present to improve the rate of fault isolation (measured in terms of cost-weighted tests). However, FIS does not require such information for its diagnostics activities.

THE KNOWLEDGE ACQUISITION PROCESS IN FIS

The preceding sections have provided some insight into the motivation for and structure of a FIS knowledge base. In this section we focus on the activities involved in constructing a knowledge base for a particular system (which in the ATE world is designated as the UUT, the unit under test).

It should be clear by now that, at a minimum, the knowledge engineer must construct a qualitative causal model for the UUT which represents the structure and behavior of the UUT down to the level of "replaceable module" appropriate for the particular task (e.g., board-level maintenance in a communications system). This, in turn, implies that domain experts are not the subject of intensive rule-extraction interviews, but rather are called upon to assist in the construction of causal models. We have found that this role shift for domain experts increases their interest in the knowledge acquisition process and, because the focus is on building a model rather than on the extraction of a frequently ill-defined and unarticulated set of rules, significantly reduces the time required to construct a usable knowledge base.

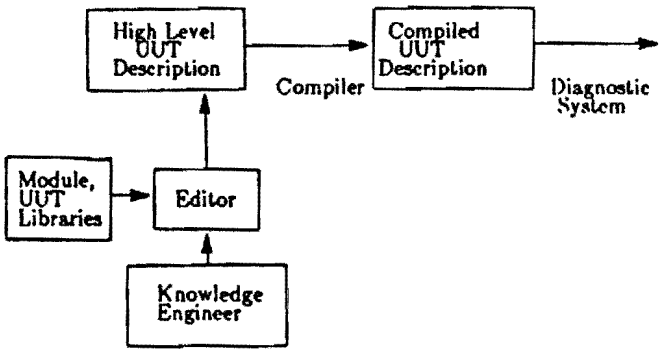

Figure 2: The Knowledge Acquisition System

Productivity can of course be further enhanced with a knowledge acquisition "front end" to assist in the knowledge base construction process. Figure 2 illustrates the basic knowledge acquisition system components which we provide for FIS. Since these causal models have a strong visual aspect to them, a display-oriented interactive editor is a natural choice for working with a knowledge base. In addition, because large systems are frequently constructed from similar components, provision is made to build up libraries of generic modules which can be "pulled in" and instantiated during the model construction process.

Although it is fairly easy to envision a knowledge acquisition front end which can facilitate network building, assisting in the acquisition of causal rules requires some careful thought, hard choices, and in our case continued experimentation. One very natural point of departure is to exploit the "object oriented" paradigm by building up libraries of generic module hierarchies so that each module instance automatically inherits a collection local causal rules. This can be quite effective

when modeling one or more systems at a level in which there are lots of instances of similar modules. However, one is still faced with providing assistance in building such libraries and one-of-a-kind modules. In this case, we have chosen initially to provide a terse rule language with built-in rule expansion capabilities to minimize the effort involved. However, we are not happy with the fact that this still places a large burden on the knowledge engineer and/or domain expert to provide FIS with reasonably consistent and complete sets of causal rules for modules. We have plans to address these problems in the near future and discuss the strategy briefly in the following section.

At this point in time, acquiring the more global causal rules associated with test points has not been a problem because there have been so few of them provided by domain experts! Even more interesting is the fact that the test point rules encountered are of the type that could have been derived by FIS via compile-time reasoning about the network. This raises some interesting issues and plans discussed in the following section.

COMPILE-TIME ACTIVITIES IN FIS

As Figure 2 illustrates, the interactive knowledge acquisition interface in FIS manipulates a high level version of the knowledge base intended to facilitate incremental acquisition. However, as with most other high level languages, there are internal representations which are far more efficient for use during execution (i.e., fault isolation in this case). As a consequence, we have found it useful to build a compilation phase into FIS to effect this transformation. It is also an opportunity to catch a number of rather straight forward knowledge base errors frequently made during knowledge acquisition such as missing or inconsistent causal rules associated with modules. For example, during incremental development of the simple model in Figure 2, one could have indicated that checking the frequency at T6 is a useful evidence-gathering test. Furthermore, the causal rule base associated with module 4 might quite legitimately leave open the possibility that frequency out of spec at T6 may be due to the fact that it is already out of spec at T5. This in turn requires some causal knowledge in module 3's rule base relating to frequency tests. If missing or inconsistent, they are flagged at this point.

Notice that this kind of consistency checking is accomplished by introducing into the compiler some of the evidential reasoning mechanisms used during fault isolation. This raises the interesting issue as to what other benefits might accrue from compile-time reasoning. The analogy we like to use is that of a technician studying the static description of a system and learning something of how it behaves. In a similar sense at compile time, FIS is "seeing" the whole network for the first time and is capable of deriving useful information about the behavior of the network from its static description. There are several interesting directions we are exploring.

The first is the ability to derive automatically from a given causal network, additional high level rules of the sort attached to test points. Such rules typically have the form: "If X is ever known to be true at this test point, then either module Y or Z are faulty". This is the sort of reasoning which, in the absence of such a rule, will have to be re-derived each time during fault isolation. Hence, studying the static structure at compile time can lead to derived rules which can dramatically increase the rate at which fault isolation occurs.

An even more intriguing and exciting use of compile-time reasoning comes from observing how the causal rule bases for unfamiliar modules are derived by domain experts. As one might expect, they generate the necessary rules by studying the sub-structure of a replaceable module which is itself a system describable by a causal network. Generally the lower level modules are simpler, more uniform, and better understood. This in turn suggests the possibility of automatically deriving higher-level module rules from lower level module descriptions. We have just begun exploring this opportunity to enhance the knowledge acquisition process and hope to report on our experiences with it in the near future.

DEBUGGING AND REFINING FIS KNOWLEDGE BASES

Even with rigorous compile-time checks there are still possibilities for errors in a causal network which will only show up as incorrect fault isolation behavior. Hence FIS provides the usual sorts of tracing and debugging mechanisms during fault isolation. Here we can and do take advantage of the visual nature of causal networks, displaying them in color and imposing on them graphical indications of how the evidential reasoning mechanism and fault isolation process is proceeding. Combining this with a rudimentary explanation facility in the form of ambiguity sets (which modules are suspect) and causal links (the derived chains of causality which made modules suspects) has produced a reasonably efficient interface for debugging and refining a particular knowledge base.

At the same time it is clear that more rigor testing of the fault isolation behavior will be required for real Navy applications. We hope to further automate this activity in the coming year.

CURRENT STATUS AND FUTURE PLANS

FIS as described here was originally written in Franz Lisp on a VAX/780 running Unix. During the past year it has been converted to Common Lisp and ported various Lisp machines and M68000-based workstations. It was demonstrated at AAAI-86 running on a Symbolics 3640.

The Naval Air Engineering Center and Harris Corporation have been working closely with us to evaluate the usefulness of FIS on actual Navy systems. Figure 3 shows one of the typical subsystems of a Doppler radar unit which is serving as an evaluation test bed. In this particular application, technicians and/or ATE stations are expected to fault isolate to the board level of the various sub-systems.

REFERENCES

De Jong, K., "Applying AI to the Diagnosis of Complex System Failures" (1984), Proc. Oakland AI Conference, pp. 121-122.

Pipitone, F., "An Expert System for Electronics Troubleshooting Based on Function and Connectivity" (1984), Proc. IEEE 1st Conf. on AI Applications, pp. 133-138.

Pipitone, F., "The FIS Electronics Troubleshooting System" (1986), IEEE Computer, July 1986, pp. 68-76.