Automatic Extraction of Component Models from Fault Knowledge: The Diagnostic Remodeler (DR) Algorithm

Abu-Hakima, Suhayya

References

Abu-Hakima, S. [1988], *RATIONALE: A Tool for Developing Knowledge-Based Systems that Explain by Reasoning Explicitly.* Masters thesis, Carleton University, Ottawa, Canada. May 1988.

Abu-Hakima, S., and Oppacher, F. [1990],Improving explanations in knowledge-based systems: RATIONALE. Knowledge Acquisition journal (2). 301-343. December 1990.

Abu-Hakima, S. [1994], Diagnostic techniques in knowledge-based systems: a review of approaches, applications and issues. 100 p. To be published as an NRC ERA (1994).

Abu-Hakima, S. [1993], Automatic Knowledge Acquisition in Diagnosis. Proceedings of DX-93, Fourth International Workshop on Principles of Diagnosis. Aberystwyth, Wales. 236-250. (1993).

Abu-Hakima, S., Halasz, M., and Phan, S. [1993], An Approach to Hypermedia in Diagnostic Systems. Chapter 10 in Intelligent Multimedia Interfaces, Edited by Mark Maybury, AAAI Press (1993).

Althoff, K-D., Maurer, F., and Rehbold, R. [1990],"Multiple knowledge acquisition strategies in MOLTKE." *Current trends in knowledge acquisition*, Published by IOS, Amsterdam, Netherlands. pp. 21-40, (1990).

Boose, J.H. [1991], "Knowledge acquisition tools, methods and mediating representations", *Knowledge Acquisition for Knowledge-Based Systems*, Edited by H. Motoda, R. Mizoguchi, J. Boose and B. Gaines. Published by IOS Press, the Netherlands. pp. 25-62, (1991).

Chandrasekaran, B., Gomez, F., Mittal, S. and Smith, J.W. [1979], "An approach to medical diagnosis based on conceptual structures". Proceedings of the Joint Conference on Artificial Intelligence, 134-142.(1979).

Chandrasekaran, B. and Tanner, M.C. [1986], "Uncertainty handling in expert systems: uniform vs. task-specific formalisms". In Kanal L.N. and Lemmer J., editors, Uncertainty in Artificial Intelligence, North Holland Publishing.(1986).

Clancey, W.J. [1986], "From GUIDON to NEOMYCIN and HERACLES in twenty short lessons: ORN final report 1979-1985". The AI Magazine, pp. 40-60,(1986).

Davis, R. [1984], "Diagnostic reasoning based on structure and behaviour", Artificial Intelligence, Vol. 24, pp. 347-410 (1984).

de Kleer, J., Williams, B.C. [1987], "Diagnosing Multiple Faults", *Artificial Intelligence*, vol. 32, (1987).

Gaines, B. and Shaw, M. [1991], "Foundations of Knowledge Acquisition", *Knowledge Acquisition for Knowledge-Based Systems*, Edited by H. Motoda, R. Mizoguchi, J. Boose and B. Gaines. Published by IOS Press, the Netherlands. pp. 3-23, (1991).

Halasz, M., Davidson, P., Abu-Hakima, S., and Phan, S. [1992], JETA: A Knowledge-based Approach to Aircraft Gas Turbine Engine Maintenance. Journal of Applied Intelligence, 2, pp. 25-46, Kluwer Academic Publishers, NRC 31832, (1992).

Hamscher, W. and Struss, P. [1990], "Model-Based Diagnosis", *AAAI-90 Tutorial Notes*, Received at the eighth national conference of artificial intelligence, Boston, Massachusetts. July 29, 1990. pp. 1-179, (1990).

Kumar, A. [1994], "Reasoning about function: AAAI-93 Workshops Summary Report", AI MAgazine, Spring 1994, Vol. 15, No. 1, (1994).

Malin, J.T., and Liefker, D.B. [1991], "Representing functions/procedures and processes/structures for analysis of effects of failures on functions and operations", Proceedings of 1991 NASA Goddard Conference on Space Applications of Artificial Intelligence, Greenbelt, Maryland, pp. 141-151, (May 13-15 1991).

Modesitt, K.L.[1990], "Inductive knowledge acquisition experience with commercial tools for space shuttle main engine testing", *Proceedings of the fifth conference on Artificial Intelligence for space applications*, NASA Marshall Flight Center, pp. 487-496, (May 1990).

Pazzani, M.J. [1990], "Learning in order to avoid search in logic programming", *Computers & Mathematics with Applications*, 20(9-10), pp. 101-10, (1990).

Scott, A.C., Clancey, W.J., Davis, R. and Shortliffe, E.H. [1977], "Explanation capabilities of production-based consultation systems". Stanford University, Dept. of Computer Science, Report No. STAN-CS-77-593,(1977).

Sticklen, J., Chandrasekaran, B., and Bond, W.E. [1988], "Distributed causal reasoning for knowledge acquisition: a functional approach to device understanding", 3rd AAAI sponsored Knowledge Acquisition Workshop for Knowledge-Based Systems, Banff, Canada, pp. 34-1 to 34-18, (1988).

Struss, P. [1989], "New Techniques in Model-based Diagnosis", *Proceedings of Knowledge-based Computer Systems*, Bombay, India, (11-13 December 1989).

Struss, P., Dressler, O. [1989], "Physical Negation - Integrating Fault Models into the General Diagnostic Engine", *Proceedings of the 11th International Joint Conference on Artificial Intelligence (IJCAI-89)*, Detroit, MI, (20-25 August 1989).

```
[function(main_fuel_nozzles,flow_control(WF+)),
input(main_fuel_nozzles,flow(WF+)),
output(main_fuel_nozzles,flow(WF+)),
regulator(main_fuel_nozzles,regulation_control(N+)),
behaviour(main_fuel_nozzles,behaviour_is_proportional(WF+,N+,
[increase_in(N+),increases(WF+),decrease_in(N+),decreases(WF+)]))],

[[main_fuel_nozzles,
[[gap_for_mode,fuel_flow_control,
extracted,EGT,input,[N+,WF],output,[WF+]],
[gap_for_mode,fuel_flow_control,
extracted,RPM,input,[N+,WF],output,[WF+]],
[gap_for_mode,fuel_flow_control,
extracted,N1,input,[N+,WF],output,[WF+]],[]]]],


[function(main_fuel_pump,delivers(WF+)),
input(main_fuel_pump,fluid(WF+)),
output(main_fuel_pump,fluid(WF+)),
regulator(main_fuel_pump,N1+),
behaviour(for(main_fuel_pump),
behaviour_is_proportional(WF+,N1+,
[increase_in(N1+),increases(WF+),decrease_in(N1+),decreases(WF+)])
)],

[function(main_fuel_pump_fuel_filter,filters(WF+)),
input(main_fuel_pump_fuel_filter,fluid(WF+)),
output(main_fuel_pump_fuel_filter,fluid(WF+)),
regulator(main_fuel_pump_fuel_filter,none),
behaviour(for(main_fuel_pump_fuel_filter),
behaviour_is_proportional(WF+,WF+,
[increase_in(WF+),increases(WF+),decrease_in(WF+),decreases(WF+)]
))],

[function(overspeed_governor,controls([WF/P3-])),
input(overspeed_governor,control([[N+]])),
output(overspeed_governor,control([[WF/P3-]])),
regulator(overspeed_governor,regulation_control([[N+]])),
behaviour(overspeed_governor,
behaviour_is_piecewise_linear([[[(mode overspeed_regulation),
input_control_variable(N+)]]],
[[[[the_behaviour_is_proportional_for(overspeed_regulation),
the_input_control_variable(N+),
results_in_regulated_output_variable(WF/P3-),
at_limit(N+),decrease(WF/P3-),increases(N+),increases(WF/P3-)]]]]))],
```

Issues

There are several issues that I intended to answer in the implementation of the *DR* algorithm. The first is what is the exact form of the learned model when some or no background knowledge is used. If no background knowledge is used is the model much more than a causal rather than a component behaviour model with explicit representation of function? The answer here is that a minimum amount of device dependent knowledge is used to map the JETA notation to text as shown in the sample output of step 4. If no device independent background knowledge (such as that I started for an the engine component library) is used then the extraction of gaps between the JETA-encoded model and a general one is not possible. Using no background knowledge it is possible to extract a component-to-component model with explicit parameteric paths from FBR knowledge. Extracting the relationships on these behavioural paths (increasing/decreasing/no change) requires generalized background knowledge.

The *DR* algorithm has been implemented as a general algorithm useful in generating models for devices other than jet engines. It is not obvious that its background knowledge will make it specific for generating a particular model. However the FBR knowledge used as input will make it specific to generating a model for a particular device. To answer this question I have generated a 30-node knowledge base for the diagnosis of a coffee maker (a very different device than an engine). I have successfully generated a component behaviour model with explicit function for all the coffee maker components. This illustrates the generality of the algorithm for devices. However, another question is whether or not the algorithm can be generalized further so that it diagnosis abstract (e.g. software) versus physical (e.g. jet engine, coffee makers, etc.) systems.

Finally, the *DR* algorithm requires a highly structured FBR knowledge base. One key question is what criteria will allow it to extract a causal model from a rule versus a frame-based fault model?

Conclusion

This paper argues that automated knowledge acquisition of models for diagnosis has had limited success in both failure-driven diagnosis and model-based diagnosis. The *DR* algorithm for the automated generation of behavioural models with an explicit representation of function from fault-based knowledge is described. An example of fault-based knowledge from the Jet Engine Troubleshooting Assistant is used to demonstrate how a model of the main fuel system of a jet engine can be extracted with *DR* from the fault knowledge.

-map out the identified components of the subsystem
-relate the components through shared parameters
-match derived component model with device dependent knowledge to derive exact parameter behaviours
-match derived component model with to library component model to extract function and uncover gaps

An Example: Extracting Models from JETA's Fault Knowledge Using DR

An analysis of the JETA fault knowledge shows layers of knowledge which can be visualized as leaves of diagnostic trees. The topmost layer is an entry point to jet engine faults and subsequent layers organize the faults into various branches. The phases of operation branches lead to various symptomatic nodes labelled as snags. These snags in turn are refinable down to repair and replacement nodes which represent the terminal nodes of the diagnostic hierarchy[1]. If one examines the knowledge encoded in these terminal nodes more closely one discovers that they represent faults directly on physical engine components. These physical component fault nodes can be grouped into those affecting one of thirteen subsystems by their nomenclature. One can follow the five steps of the *DR* algorithm introduced above to discover the behavioural and functional component model for the main fuel system of the jet engine.

Step 1:

One can identify 9 replace nodes through the JETA node frame slot 'node-type'.

Step 2

If one takes a specific subsystem, the MFS (Main Fuel System), one can extract the names of 3 fuel system replacement nodes by pattern matching with the node nomenclature *N-MFS-XXX (this is an internal representation that was used by the knowledge engineer to distinguish between nodes):

1. main fuel control (MFC)
2. overspeed governor for MFC (OSG)
3. main fuel pump supplying MFC (MFP)

Step 3

For each of the 3replacement nodes parents connecting sibling nodes can be extracted, for example:

- the MFC and MFP nodes share the parent node loss of fuel flow
- the OSG node shares with the MFC the engine speed hang-up parent node
- the main fuel control (MFC) delivers fuel to the main engine fuel nozzles (FN)
- the pressurizing and drain valve (PDV) is connected to the main fuel nozzle FN and both share fuel flow

Step 4

A causal topological network can be the basis for hypothesized component-behaviour relations. Sibling nodes are clustered based on shared parent links. Example DR output relations that form part of the network include:

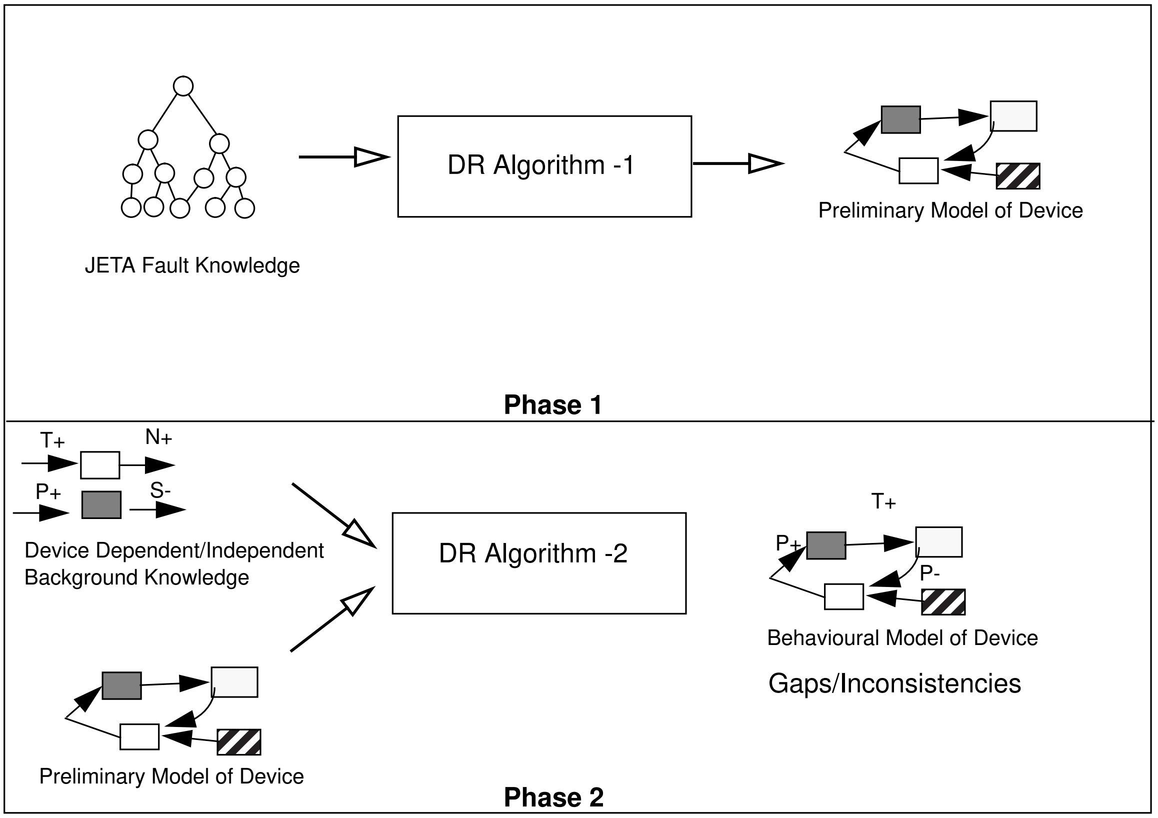
```
[[main_fuel_control,
is_a([control,for,[steady_speed_control,speed_cutback_control,
    acceleration_fuel_limit_control, deceleration_fuel_limit_control,
    variable_geometry_scheduling,proportional_speed_control,
    fru_fuel_selection]]),
and_is_connected_to(main_fuel_pump),
with_connectivity_parameter([weight_of_fuel_flow])],

[main_fuel_control_fuel_filter,
is_a([filter,for,[filtering_fuel]]),
and_is_connected_to(main_fuel_control),
with_connectivity_parameter([weight_of_fuel_flow])],

[main_fuel_nozzles,
is_a([nozzle,for,[fuel_flow_control]]),
and_is_connected_to(main_fuel_control),
with_connectivity_parameter([measured_rpm_engine_speed,
    single_spool_engine_speed,weight_of_fuel_flow])],

[main_fuel_nozzles,
is_a([nozzle,for,[fuel_flow_control]]),
and_is_connected_to(pressurizing_and_drain_valve),
with_connectivity_parameter(fuel_pump_inlet_pressure[])],

[overspeed_governor,
is_a([con-
trol,for,[overspeed_regulation],[fuel_control_overspeed_governor]]),
and_is_connected_to(main_fuel_control),
with_connectivity_parameter([measured_rpm_engine_speed,
    single_spool_engine_speed,weight_of_fuel_flow])],

[pressurizing_and_drain_valve,
is_a([control,for,[fuel_overflow]]),
and_is_connected_to(main_fuel_control),
with_connectivity_parameter([weight_of_fuel_flow])]]

.......
```

Step 5

Step 4 output is matched against device independent/dependent background knowledge and gaps identified. For example (note only the gaps for the main fuel nozzles are identified):

[1] The diagnostic hierarchy is referred to as a network since it includes relations that are not directly inherited that allow the JETA reasoner to jump around between nodes thus forming more of a network than a hierarchy.

Figure 1: The Phases of the Diagnostic Remodeler (DR) Algorithm



from a sink to a source and needs a signal (control signal such as pressure) to increase or decrease the flow of liquid. It also includes some knowledge about feedback control in moderating the flow of a liquid to a source based on the level of the liquid at the source. The device dependent knowledge includes the specifics of control modes for a type of control of the device. For example, the Main Fuel Control includes modes on steady engine speed control, speed cutback control, deceleration fuel limit control, variable geometry scheduling, etc. In a general sense these are all control modes but in a specific sense they have different control signals (in one case it is engine speed, in another it is fuel flow and in yet another mode it is the air bleed valve positions). The device dependent and independent background knowledge is matched against the extracted JETA knowledge to uncover gaps.

To achieve the knowledge-rich learning proposed as the output for *DR* one requires the use of a structured and explicit knowledge representation that can adequately represent diagnostic causality. This is achieved by extracting a model of the connections between the components in the subsystem to be modelled. These connections are further used to extract the variables (such as engine speed, fuel flow, etc.) that typify the behaviour between components. The second phase of the *DR* algorithm matches the functional background knowledge of a device exemplified by its device model of components and their behaviours with the original fault knowledge. The purpose of the second phase is to find inconsistencies and gaps in the fault knowledge. The gaps discovered in the fault knowledge could then be used to diagnose novel faults.

The objective of the *DR* algorithm is to discover and refine a component behavioural model with explicit function. In the most general sense the algorithm must identify the components of the device, generate links between those components and generate hypotheses for the behaviour and function of the components.

To achieve this the *DR* algorithm must perform five steps:

1. identify the terminal nodes in the diagnostic hierarchy

   -these represent component nodes that have no child or sibling refinements
2. identify the component nodes in the diagnostic hierarchy related to the subsystem to be modelled (if required)

   -perform a pattern match with the known name or its derivatives (possibly acronyms) that match the subsystem
3. identify the parents and siblings of the nodes

   -backtrack from end node to parent node and tag

   -tag shared parents of a node

   -tag siblings of a parent
4. extract relations (behaviours) between the sibling nodes

   -cluster nodes related by parental nodes

   -movement from the terminal nodes to the root node represents symptomatic information (parameters)
5. match device model against background knowledge and output gaps for verification to the user

integration of various machine learning algorithms with partially hand-coded knowledge bases used in FBR or models that represent behaviour and function generated for MBR.

Automatic Component Behaviour & Function Generation: the *DR* algorithm

Hypothesis

Humans use failure-driven reasoning for successful device diagnosis and repair. As humans reason about diagnosis and repair they build primitive mental models of how a device functions and fails. The hypothesis for the *Diagnostic Remodeler* algorithm is that knowledge of failure and repair embodied in most structured diagnostic knowledge-based systems can be used to derive rudimentary device component models. The *DR* algorithm extracts rudimentary device component models from fault knowledge that represent structure, behaviour and function.

Motivation

A great deal of effort is expended hand-coding complex knowledge bases for diagnostic FBR. The artifacts these diagnostic systems are developed for are often expensive machines which have been designed and continuously modified so that no existing accurate schematic or design of their behaviour or resultant function remains. The J85-CAN-15 is a jet engine which is the first application of JETA. The J85-CAN-15 engine was designed in the 1950's and has easily had at least one modification a year since its launch. As a result of modifications and stresses of daily use (flying in the arctic and flying in desert heat) the jet engine is a very different device than was originally designed and sometimes displays inexplicable behaviour. No existing design schematics can completely capture the engines's behaviour or completely predict its function. It is also a very difficult device to diagnose. For these reasons a tool such as JETA was developed. As is typical with FBR systems, JETA does not diagnose novel faults. Learning the device component model, its behaviour and functionality using the FBR knowledge provides the technician with a tool that can achieve model-based diagnosis. For these reasons it was concluded that the *DR* algorithm should be implemented.

Background

If we follow the de Kleer [de Kleer and Williams 87] approach which represents device function as a set of components with behaviour. The device can be diagnosed by assuming a faulty component and enumerating the behavioural states that the fault propagates in the remainder of the device. This is compared to the behaviour that a technician is observing in attempting to isolate a problem. Model-based diagnosis can detect novel faults since the behaviour of the device is the basis of its knowledge representation and reasoning. Fault-based reasoning uses the faults in the functioning of a device rather than its actual behaviour, hence FBR cannot detect novel faults. However, model-based reasoning can lead to a combinatorial explosion in producing a diagnosis for complex systems (for example, an aircraft engine) and it does not lend itself to causal explanation.

I have implemented the *DR* algorithm [Abu-Hakima 93] intended to address the automatic generation of a functional model of a device from its fault knowledge. That implies the automatic generation of MBR knowledge from FBR knowledge. By extracting a functional model both fault and model-based diagnosis can be pursued in a single system gaining from the advantages of the two approaches while minimizing the disadvantages. The *DR* algorithm is being applied in the area of complex electromechanical devices, specifically jet engines.

Objectives of the DR Algorithm

DR is an algorithm that takes as input the fault knowledge of a device. It is also necessary to take as input some background knowledge related to the device to attempt to learn its full component structure and connectivity. *DR* initially extracts from the fault knowledge base all references to device components and subsystems. Given these components the algorithm backtracks through a diagnostic hierarchy of nodes to generate hypotheses for component connectivity. To further establish component connectivity, *DR* examines symptomatic or parametric knowledge that activates the diagnostic nodes. Symptomatic knowledge is knowledge of device failure which can be used to generate hypotheses about correct device function. This knowledge is used to derive behavioural knowledge between components.

Approach to the *DR* Algorithm

The top level design of the *DR* algorithm is shown in Figure 1. Two phases clearly divide the operation of the algorithm. In the first phase, an existing knowledge base that diagnoses a complex electromechanical system is used as input to *DR*. The Jet Engine Troubleshooting Assistant (JETA) is a system implemented to diagnose faults with aircraft engines [Halasz et al. 92]. Two types of background knowledge, device dependent and device independent knowledge are used as the second inputs to the DR algorithm. This device independent background knowledge is in the form of a component library and is general in nature, for example it includes knowledge that a pump delivers some liquid

model and fault-based diagnosis to deal with GDE's shortcomings.

Other MBR authors have argued about the definition of device functionality versus behaviour. Sticklen in [Sticklen et al. 88] describes modelling a device's functionality by:

- decomposing the device into sub-devices,
- stating abstractly the functions, goals and purpose of the device and
- representing the manner of achieving the device functions, goals and purpose.

A good definition of functionality is one which argues that function is the set of goals the device is intended or designed to achieve [Malin and Liefker 91]. As stated in the abstract the use of function in this paper is based on the perspective that function complements behaviour where the derived function is more abstract than the behaviour derived [Kumar 94].

Automatic Acquisition of Models for Diagnosis

Machine learning is a key approach in knowledge acquisition for diagnosis. Machine learning includes empirical and analytic learning. Empirical learning focuses on learning for classification (including learning rules from real or simulated data for diagnosis). Analytic learning addresses learning for problem solving tasks which include planning, diagnosis, design, natural language understanding, control and execution. There has been an explosion of work in machine learning in recent years. It is viewed as one of the key approaches of reducing the knowledge acquisition bottleneck [Boose 91; Gaines and Shaw 91].

The MOLTKE (MOdels, Learning and Temporal Knowledge in Expert systems) testbed for diagnosis under development at the University of Kaiserslautern, Germany is described in [Althoff et al. 90]. The system is designed to acquire device knowledge for diagnosis. It has an MBR mechanism for acquiring device models based on their components. A *component* of the model includes a name, ports to other components (with optional test costs), possible internal states (with optional test costs), behaviour of the component (either in state tables or rules that represent the constraints the component sets up between its ports and states), subparts and their interconnections (if the component is non-atomic), typical malfunctions with name and effects (model typical behavior when the component fails) and a priori probability of failure). No direct reference to device function is made. MOLTKE uses case-based reasoning to acquire and refine knowledge that is generalized to a fault-based hierarchy. It also uses explanation-based learning to refine the rules in the fault-based hierarchy to get the minimum reasoning paths for a solution. MOLTKE has been applied to a Computerized Numerical Control (CNC) machining center. It is also under investigation for the problem of driving mining machines.

ACES (Attitude Control Expert System) diagnoses anomalies in the attitude control system of the DSCS-III satellite [Pazzani 90]. ACES is fault-based (rules represented as Prolog predicates). A fault is confirmed or denied by comparing the observed behavior to that predicted with a simulator. In the case where the simulation denies the fault, the heuristic that proposed the fault is expanded to include the tests that the simulator performed to rule out the fault. In this manner the simulator generates expected behaviour given a particular fault. ACES uses explanation-based learning (EBL) to identify the conditions under which the heuristic will propose a fault that is denied. The author concludes that failure-driven learning finds sufficient conditions for ruling out a fault and success-driven learning finds sufficient conditions for establishing a fault (but not necessarily ruling others out). Pazzani's work is novel and very relevant to the refinement of fault-based knowledge using model-based reasoning and explanation-based learning.

There has been tremendous activity in machine learning in recent years. In empirical learning classification algorithms such as ID3 and AQ have been used to induce diagnostic rules from real or simulated data. Classification learning extracts rules from positive and negative examples. In analytic learning explanation-based learning has been used in the form of speedup learning to generalize diagnostic rules and shorten reasoning chains. I believe that neither classification nor EBL addresses the problem of *knowledge-rich* learning where structured knowledge is learned. Such rich knowledge would result from learning to produce hypothesis hierarchies such as those described in fault-based reasoning. In addition, learning from structured knowledge to produce new knowledge, such as learning a device model from its fault hierarchy has not been addressed. Learning complex structures especially for diagnosis is by no means an easy problem but it is one that needs to be further addressed by a combination of researchers in both the machine learning and diagnosis fields. Some researchers which have combined learning (empirical or analytic) with FBR and MBR have met with more success as exemplified by the complex systems above. I believe that the key to resolving the knowledge acquisition bottleneck in diagnosis lies in the

JETA's troubleshooting knowledge is represented as a diagnostic network that is hierarchical in nature. Each node in the network corresponds to a decision point in the troubleshooting process that mimics the problem solving strategy of an expert engine technician. At the top level JETA is attempting to reason about device function in terms of actual engine operation phases (i.e. start-up, acceleration, decceleration, etc.). It refines problems encountered in engine function at the phases of operation until it can identify symptoms that represent component failures. Thus, at the top level JETA can be thought of as reasoning about overall function and systematically refining its reasoning to failed behaviour on device components. As a result the links in JETA's diagnostic network represent relations directing the flow of control between nodes. The overall network is much broader than it is deep since there are many components and associated symptoms. The number of nodes along a network path varies from four to twelve in a network of approximately 200 nodes. Possible next moves in the network are represented as children of a node. Any node can have multiple parents since a component malfunction may be due to many causes. The troubleshooting knowledge is hand-coded at each diagnostic node as a frame using a custom command language. In JETA as in RATIONALE, advice generating slots are included in the frame and their contents are output to the user as diagnoses or procedures to follow to find a fault. In JETA, advice is supported with a schematic or a graph. An indexed database of schematics and graphs is kept so that only pointers to the database are kept in the frame. The current implementation of JETA links text, graphs and schematics.

Function in Model-Based Diagnosis

Model-based reasoning (MBR) for diagnosis concentrates on reasoning about the expected and correct functioning of a device. A device is modelled based on its components and their expected behaviour [Hamscher and Struss 90]. Such models range from quantitative ones to qualitative ones and all attempt to approximate device behaviour as accurately as possible. Once a device model is stabilized then a device's observed behaviour can be predicted from the model. If a discrepancy in behaviour is detected then possible candidates based on assumed component faults are generated. These candidates are generated based on assumptions that describe correct model behaviour. Sequential diagnosis is used to choose observations, augment a prediction for the candidate faults and update the list of candidates until a dominant candidate is found.

In MBR there are many conflicting definitions for models. They range from causal models represented as semantic networks with links specifying the relations between component nodes to full blown numerical simulations for complex systems and processes that have taken decades to perfect. Generating models is a key problem in MBR. Some researchers generate causal models, others generate models with structure and behaviour while others generate functional models for devices. Knowledge in models has thus far been hand-coded by experts that understand device component behaviour and function.

Davis was one of the earlier proponents of MBR. In [Davis 84] he describes a theory to exploit reasoning on the basis of device structure and behaviour. He defines paths of causal interpretation. He also describes constraint suspension used to identify which components are responsible for which faults. He argues that we need to balance complexity versus model completeness in diagnosis thus we need to enumerate and layer categories of failure. Quite a bit of work has followed Davis' examples and theories.

De Kleer and Williams published a key paper on MBR for diagnosis describing GDE, the General Diagnostic Engine [de Kleer and Williams 87]. GDE infers behaviour from device structure and functionality. It is applied to digital circuits and makes use of an ATMS (Assumption-Based Truth Maintenance System). This work forms the cornerstone of ATMS-based model-based reasoning systems. It was followed by many papers that criticized the approach as not computationally practical in diagnosing faults with large complex systems. Some of the papers criticizing GDE propose the use of hierarchical fault-based reasoning to reduce the computational complexity of de Kleer and Williams' approach. Struss has developed GDE+ which handles: simple dynamic aspects, multiple tests, hierarchical knowledge and unreliable observations [Struss 89]. GDE+ is a partial migration back to take advantage of heuristic or empirical diagnoses using fault-based reasoning. Struss points out that neither GDE nor GDE+ address: changing device structures, complex temporal behaviour (feedback), uncertainty or the use of qualitative models in reasoning. In [Struss and Dressler 89] the authors advocate the representation of a fault view for each component. They point out that a fault and a healthy view (state) for a component cannot be true in the same time instant (consistent belief rule). They also give the 'no good inference rule' where the node and its opposite which represents a fault cannot be true at the same instant. The ATMS is then modified to reason with the fault as well as the no-fault behaviour of a device. Their work gives excellent insight into combining

Automatic Extraction of Device Models from Fault Knowledge: the Diagnostic Remodeler (DR) Algorithm

NRC 37141

Suhayya Abu-Hakima
Knowledge Systems Laboratory, Institute for Information Technology
Building M-50, Montreal Road, National Research Council of Canada
Ottawa, Canada, K1A 0R6
email: suhayya@ai.iit.nrc.ca,
tel: (613) 993-8551, fax: (613) 952-7151

Abstract

This paper argues that automated knowledge acquisition for diagnosis has had limited success in both failure-driven diagnosis and model-based diagnosis. The paper describes fault-based and model-based reasoning for diagnosis and surveys some of the approaches to knowledge acquisition in both areas. The Diagnostic Remodeler (DR) algorithm has been implemented for the automated generation of behavioural component models with function from fault-based knowledge. The use of function in this paper is based on the perspective that function complements behaviour where the derived function is more abstract than the behaviour derived by DR [Kumar 94]. DR uses as its first application example the fault-based knowledge base of the Jet Engine Troubleshooting Assistant (JETA). DR is used to extract the model of the Main Fuel System using the knowledge base and two types of background knowledge as input: device dependent and device independent knowledge. This paper is the first presentation of preliminary results of the implemented DR algorithm.

Function in Fault-Based Diagnosis

Fault-based reasoning (FBR) is used in many diagnostic systems. Knowledge in FBR is largely based on maintenance manuals and interviews with experts intended to capture heuristic knowledge about the maintenance and repair of a device or process. The maintenance and repair is directed at keeping a device functioning in a predictable manner. The knowledge in these systems is often represented as hand-coded rules or frames which are organized into troubleshooting hierarchies. At the top level of the hierarchy is the general knowledge representing a problem with device function. This general problem is refined systematically until the leaf nodes of the hierarchy which represent physical repairs to the device are reached. Once these repairs are achieved by a human technician some diagnostic systems re-test to confirm that the symptoms and diagnosed faults are cleared through backtracking in the hierarchy.

FBR systems have evolved considerably since the development of MYCIN [Scott et al. 77; Clancey 86]. MYCIN was developed to provide advice treatment for microbial infections. The MYCIN programs started with hand-coded rules which later evolved into meta-rules in NEOMYCIN to provide some structure to an otherwise flat knowledge base. The MYCIN approach remains a very widely used approach in FBR systems as described in the literature review of [Abu-Hakima 94].

Diagnosis is often referred to as a classification problem. Chandrasekaran and his colleagues developed MDX, a system that diagnosis a form of liver disease, cholestasis [Chanrasekaran et al. 79]. MDX has a diagnostic hierarchy which is referred to as a conceptual hierarchy since it guides the reasoner globally through diagnoses clustered as concepts that establish local contexts. Local uncertainties and hand-coded knowledge represented in frames are used to guide the diagnosis [Chandrasekaran and Tanner 86]. MDX has served as a model for many well-structured diagnostic systems including RATIONALE [Abu-Hakima 88] and JETA [Halasz et al. 92].

RATIONALE is a workstation diagnosis system that reasons explicitly so that it may support the user with sophisticated explanations of diagnoses that help justify diagnostic system behaviour and clarify reasoning. This approach was found to be ideal for explicitly representing causal knowledge of problems with device function so that it may be explained [Abu-Hakima and Oppacher 90]. RATIONALE diagnoses faults with Xerox workstations. It generates dynamic and static template-based explanations that include why, how and what-if responses. Explanation remains a major objective of FBR systems and most systems have why and how explanation but do not necessarily generate hypothetical (what-if) ones. RATIONALE's knowledge is in hand-coded frames.

The Jet Engine Troubleshooting Assistant (JETA) is a tool developed to assist a technician in diagnosing aircraft engines using a hypermedia interface which provides contextual help. For a diagnostic application to properly support hypermedia, one requires a structured manner by which to represent the knowledge, reason about it interactively, display it dynamically and explain it to the user (see [Abu-Hakima et al. 93] for a thorough description of JETA's hypermedia interface). JETA's knowledge representation and reasoning strategies are more flexible than those of other diagnostic systems including RATIONALE's.

References

Abu-Hakima, S. [1988], *RATIONALE: A Tool for Developing Knowledge-Based Systems that Explain by Reasoning Explicitly.* Masters thesis, Carleton University, Ottawa, Canada. May 1988.

Abu-Hakima, S., and Oppacher, F. [1990],Improving explanations in knowledge-based systems: RATIONALE. Knowledge Acquisition journal (2). 301-343. December 1990.

Abu-Hakima, S. [1994], Diagnostic techniques in knowledge-based systems: a review of approaches, applications and issues. 100 p. To be published as an NRC ERA (1994).

Abu-Hakima, S. [1993], Automatic Knowledge Acquisition in Diagnosis. Proceedings of DX-93, Fourth International Workshop on Principles of Diagnosis. Aberystwyth, Wales. 236-250. (1993).

Abu-Hakima, S., Halasz, M., and Phan, S. [1993], An Approach to Hypermedia in Diagnostic Systems. Chapter 10 in Intelligent Multimedia Interfaces, Edited by Mark Maybury, AAAI Press (1993).

Althoff, K-D., Maurer, F., and Rehbold, R. [1990],"Multiple knowledge acquisition strategies in MOLTKE." *Current trends in knowledge acquisition*, Published by IOS, Amsterdam, Netherlands. pp. 21-40, (1990).

Boose, J.H. [1991], "Knowledge acquisition tools, methods and mediating representations", *Knowledge Acquisition for Knowledge-Based Systems*, Edited by H. Motoda, R. Mizoguchi, J. Boose and B. Gaines. Published by IOS Press, the Netherlands. pp. 25-62, (1991).

Chandrasekaran, B., Gomez, F., Mittal, S. and Smith, J.W. [1979], "An approach to medical diagnosis based on conceptual structures". Proceedings of the Joint Conference on Artificial Intelligence, 134-142.(1979).

Chandrasekaran, B. and Tanner, M.C. [1986], "Uncertainty handling in expert systems: uniform vs. task-specific formalisms". In Kanal L.N. and Lemmer J., editors, Uncertainty in Artificial Intelligence, North Holland Publishing.(1986).

Clancey, W.J. [1986], "From GUIDON to NEOMYCIN and HERACLES in twenty short lessons: ORN final report 1979-1985". The AI Magazine, pp. 40-60,(1986).

Davis, R. [1984], "Diagnostic reasoning based on structure and behaviour", Artificial Intelligence, Vol. 24, pp. 347-410 (1984).

de Kleer, J., Williams, B.C. [1987], "Diagnosing Multiple Faults", *Artificial Intelligence*, vol. 32, (1987).

Gaines, B. and Shaw, M. [1991], "Foundations of Knowledge Acquisition", *Knowledge Acquisition for Knowledge-Based Systems*, Edited by H. Motoda, R. Mizoguchi, J. Boose and B. Gaines. Published by IOS Press, the Netherlands. pp. 3-23, (1991).

Halasz, M., Davidson, P., Abu-Hakima, S., and Phan, S. [1992], JETA: A Knowledge-based Approach to Aircraft Gas Turbine Engine Maintenance. Journal of Applied Intelligence, 2, pp. 25-46, Kluwer Academic Publishers, NRC 31832, (1992).

Hamscher, W. and Struss, P. [1990], "Model-Based Diagnosis", *AAAI-90 Tutorial Notes*, Received at the eighth national conference of artificial intelligence, Boston, Massachusetts. July 29, 1990. pp. 1-179, (1990).

Kumar, A. [1994], "Reasoning about function: AAAI-93 Workshops Summary Report", AI MAgazine, Spring 1994, Vol. 15, No. 1, (1994).

Malin, J.T., and Liefker, D.B. [1991], "Representing functions/procedures and processes/structures for analysis of effects of failures on functions and operations", Proceedings of 1991 NASA Goddard Conference on Space Applications of Artificial Intelligence, Greenbelt, Maryland, pp. 141-151, (May 13-15 1991).

Modesitt, K.L.[1990], "Inductive knowledge acquisition experience with commercial tools for space shuttle main engine testing", *Proceedings of the fifth conference on Artificial Intelligence for space applications*, NASA Marshall Flight Center, pp. 487-496, (May 1990).

Pazzani, M.J. [1990], "Learning in order to avoid search in logic programming", *Computers & Mathematics with Applications*, 20(9-10), pp. 101-10, (1990).

Scott, A.C., Clancey, W.J., Davis, R. and Shortliffe, E.H. [1977], "Explanation capabilities of production-based consultation systems". Stanford University, Dept. of Computer Science, Report No. STAN-CS-77-593,(1977).

Sticklen, J., Chandrasekaran, B., and Bond, W.E. [1988], "Distributed causal reasoning for knowledge acquisition: a functional approach to device understanding", 3rd AAAI sponsored Knowledge Acquisition Workshop for Knowledge-Based Systems, Banff, Canada, pp. 34-1 to 34-18, (1988).

Struss, P. [1989], "New Techniques in Model-based Diagnosis", *Proceedings of Knowledge-based Computer Systems*, Bombay, India, (11-13 December 1989).

Struss, P., Dressler, O. [1989], "Physical Negation - Integrating Fault Models into the General Diagnostic Engine", *Proceedings of the 11th International Joint Conference on Artificial Intelligence (IJCAI-89)*, Detroit, MI, (20-25 August 1989).

```
[function(main_fuel_nozzles,flow_control(WF+)),
input(main_fuel_nozzles,flow(WF+)),
output(main_fuel_nozzles,flow(WF+)),
regulator(main_fuel_nozzles,regulation_control(N+)),
behaviour(main_fuel_nozzles,behaviour_is_proportional(WF+,N+,
[increase_in(N+),increases(WF+),decrease_in(N+),decreases(WF+)]))],

[[main_fuel_nozzles,
[[gap_for_mode,fuel_flow_control,
extracted,EGT,input,[N+,WF],output,[WF+]],
[gap_for_mode,fuel_flow_control,
extracted,RPM,input,[N+,WF],output,[WF+]],
[gap_for_mode,fuel_flow_control,
extracted,N1,input,[N+,WF],output,[WF+]],[]]]],


[function(main_fuel_pump,delivers(WF+)),
input(main_fuel_pump,fluid(WF+)),
output(main_fuel_pump,fluid(WF+)),
regulator(main_fuel_pump,N1+),
behaviour(for(main_fuel_pump),
behaviour_is_proportional(WF+,N1+,
[increase_in(N1+),increases(WF+),decrease_in(N1+),decreases(WF+)])
)],

[function(main_fuel_pump_fuel_filter,filters(WF+)),
input(main_fuel_pump_fuel_filter,fluid(WF+)),
output(main_fuel_pump_fuel_filter,fluid(WF+)),
regulator(main_fuel_pump_fuel_filter,none),
behaviour(for(main_fuel_pump_fuel_filter),
behaviour_is_proportional(WF+,WF+,
[increase_in(WF+),increases(WF+),decrease_in(WF+),decreases(WF+)]
))],

[function(overspeed_governor,controls([WF/P3-])),
input(overspeed_governor,control([[N+]])),
output(overspeed_governor,control([[WF/P3-]])),
regulator(overspeed_governor,regulation_control([[N+]])),
behaviour(overspeed_governor,
behaviour_is_piecewise_linear([[[(mode overspeed_regulation),
input_control_variable(N+)]]],
[[[[the_behaviour_is_proportional_for(overspeed_regulation),
the_input_control_variable(N+),
results_in_regulated_output_variable(WF/P3-),
at_limit(N+),decrease(WF/P3-),increases(N+),increases(WF/P3-)]]]]))],
```

Issues

There are several issues that I intended to answer in the implementation of the *DR* algorithm. The first is what is the exact form of the learned model when some or no background knowledge is used. If no background knowledge is used is the model much more than a causal rather than a component behaviour model with explicit representation of function? The answer here is that a minimum amount of device dependent knowledge is used to map the JETA notation to text as shown in the sample output of step 4. If no device independent background knowledge (such as that I started for an the engine component library) is used then the extraction of gaps between the JETA-encoded model and a general one is not possible. Using no background knowledge it is possible to extract a component-to-component model with explicit parameteric paths from FBR knowledge. Extracting the relationships on these behavioural paths (increasing/decreasing/no change) requires generalized background knowledge.

The *DR* algorithm has been implemented as a general algorithm useful in generating models for devices other than jet engines. It is not obvious that its background knowledge will make it specific for generating a particular model. However the FBR knowledge used as input will make it specific to generating a model for a particular device. To answer this question I have generated a 30-node knowledge base for the diagnosis of a coffee maker (a very different device than an engine). I have successfully generated a component behaviour model with explicit function for all the coffee maker components. This illustrates the generality of the algorithm for devices. However, another question is whether or not the algorithm can be generalized further so that it diagnosis abstract (e.g. software) versus physical (e.g. jet engine, coffee makers, etc.) systems.

Finally, the *DR* algorithm requires a highly structured FBR knowledge base. One key question is what criteria will allow it to extract a causal model from a rule versus a frame-based fault model?

Conclusion

This paper argues that automated knowledge acquisition of models for diagnosis has had limited success in both failure-driven diagnosis and model-based diagnosis. The *DR* algorithm for the automated generation of behavioural models with an explicit representation of function from fault-based knowledge is described. An example of fault-based knowledge from the Jet Engine Troubleshooting Assistant is used to demonstrate how a model of the main fuel system of a jet engine can be extracted with *DR* from the fault knowledge.

-map out the identified components of the subsystem

-relate the components through shared parameters

-match derived component model with device dependent knowledge to derive exact parameter behaviours

-match derived component model with to library component model to extract function and uncover gaps

An Example: Extracting Models from JETA's Fault Knowledge Using DR

An analysis of the JETA fault knowledge shows layers of knowledge which can be visualized as leaves of diagnostic trees. The topmost layer is an entry point to jet engine faults and subsequent layers organize the faults into various branches. The phases of operation branches lead to various symptomatic nodes labelled as snags. These snags in turn are refinable down to repair and replacement nodes which represent the terminal nodes of the diagnostic hierarchy[1]. If one examines the knowledge encoded in these terminal nodes more closely one discovers that they represent faults directly on physical engine components. These physical component fault nodes can be grouped into those affecting one of thirteen subsystems by their nomenclature. One can follow the five steps of the *DR* algorithm introduced above to discover the behavioural and functional component model for the main fuel system of the jet engine.

Step 1:

One can identify 9 replace nodes through the JETA node frame slot 'node-type'.

Step 2

If one takes a specific subsystem, the MFS (Main Fuel System), one can extract the names of 3 fuel system replacement nodes by pattern matching with the node nomenclature *N-MFS-XXX (this is an internal representation that was used by the knowledge engineer to distinguish between nodes):

1. main fuel control (MFC)
2. overspeed governor for MFC (OSG)
3. main fuel pump supplying MFC (MFP)

Step 3

For each of the 3replacement nodes parents connecting sibling nodes can be extracted, for example:

- the MFC and MFP nodes share the parent node loss of fuel flow
- the OSG node shares with the MFC the engine speed hang-up parent node
- the main fuel control (MFC) delivers fuel to the main engine fuel nozzles (FN)
- the pressurizing and drain valve (PDV) is connected to the main fuel nozzle FN and both share fuel flow

Step 4

A causal topological network can be the basis for hypothesized component-behaviour relations. Sibling nodes are clustered based on shared parent links. Example DR output relations that form part of the network include:

```
[[main_fuel_control,
is_a([control,for,[steady_speed_control,speed_cutback_control,
        acceleration_fuel_limit_control, deceleration_fuel_limit_control,
        variable_geometry_scheduling,proportional_speed_control,
        fru_fuel_selection]]),
and_is_connected_to(main_fuel_pump),
with_connectivity_parameter([weight_of_fuel_flow])],

[main_fuel_control_fuel_filter,
is_a([filter,for,[filtering_fuel]]),
and_is_connected_to(main_fuel_control),
with_connectivity_parameter([weight_of_fuel_flow])],

[main_fuel_nozzles,
is_a([nozzle,for,[fuel_flow_control]]),
and_is_connected_to(main_fuel_control),
with_connectivity_parameter([measured_rpm_engine_speed,
        single_spool_engine_speed,weight_of_fuel_flow])],

[main_fuel_nozzles,
is_a([nozzle,for,[fuel_flow_control]]),
and_is_connected_to(pressurizing_and_drain_valve),
with_connectivity_parameter(fuel_pump_inlet_pressure[])],

[overspeed_governor,
is_a([control,for,[overspeed_regulation],[fuel_control_overspeed_governor]]),
and_is_connected_to(main_fuel_control),
with_connectivity_parameter([measured_rpm_engine_speed,
        single_spool_engine_speed,weight_of_fuel_flow])],

[pressurizing_and_drain_valve,
is_a([control,for,[fuel_overflow]]),
and_is_connected_to(main_fuel_control),
with_connectivity_parameter([weight_of_fuel_flow])]]
```

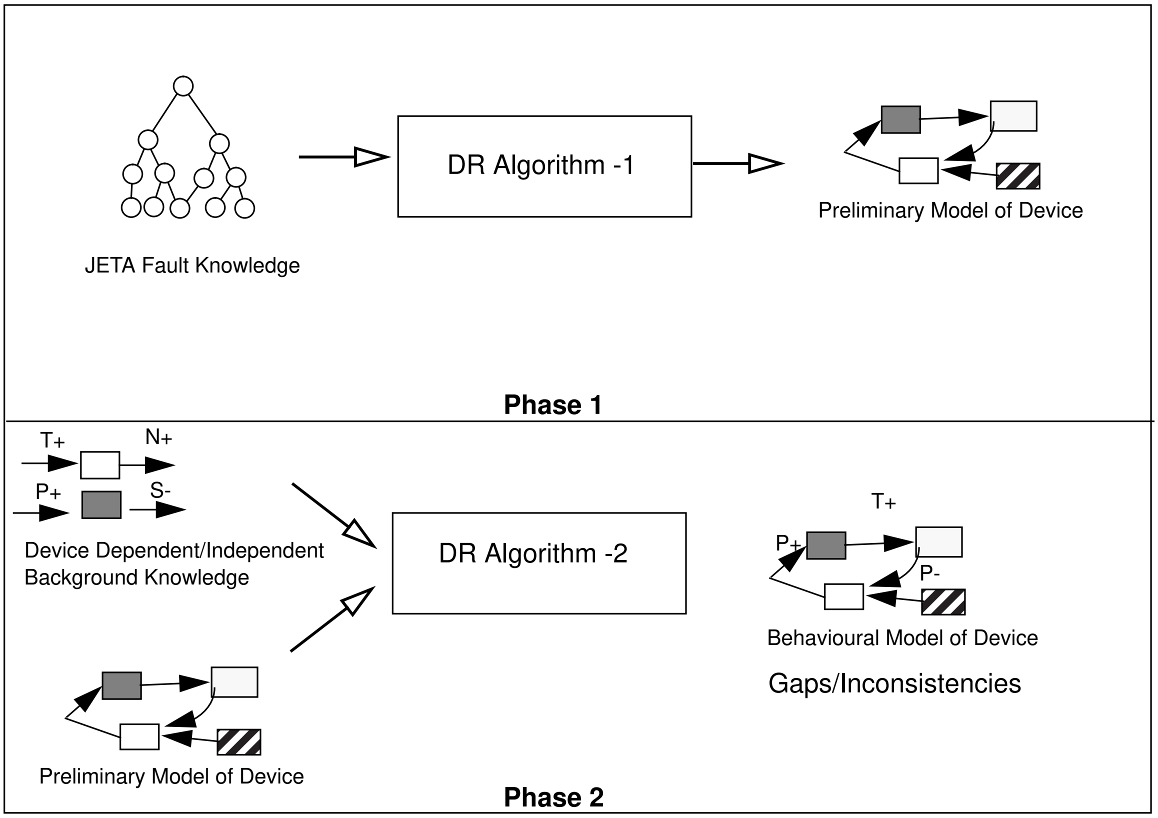
....

Step 5

Step 4 output is matched against device independent/dependent background knowledge and gaps identified. For example (note only the gaps for the main fuel nozzles are identified):

[1] The diagnostic hierarchy is referred to as a network since it includes relations that are not directly inherited that allow the JETA reasoner to jump around between nodes thus forming more of a network than a hierarchy.

Figure 1: The Phases of the Diagnostic Remodeler (DR) Algorithm



from a sink to a source and needs a signal (control signal such as pressure) to increase or decrease the flow of liquid. It also includes some knowledge about feedback control in moderating the flow of a liquid to a source based on the level of the liquid at the source. The device dependent knowledge includes the specifics of control modes for a type of control of the device. For example, the Main Fuel Control includes modes on steady engine speed control, speed cutback control, deceleration fuel limit control, variable geometry scheduling, etc. In a general sense these are all control modes but in a specific sense they have different control signals (in one case it is engine speed, in another it is fuel flow and in yet another mode it is the air bleed valve positions). The device dependent and independent background knowledge is matched against the extracted JETA knowledge to uncover gaps.

To achieve the knowledge-rich learning proposed as the output for *DR* one requires the use of a structured and explicit knowledge representation that can adequately represent diagnostic causality. This is achieved by extracting a model of the connections between the components in the subsystem to be modelled. These connections are further used to extract the variables (such as engine speed, fuel flow, etc.) that typify the behaviour between components. The second phase of the *DR* algorithm matches the functional background knowledge of a device exemplified by its device model of components and their behaviours with the original fault knowledge. The purpose of the second phase is to find inconsistencies and gaps in the fault knowledge. The gaps discovered in the fault knowledge could then be used to diagnose novel faults.

The objective of the *DR* algorithm is to discover and refine a component behavioural model with explicit function. In the most general sense the algorithm must identify the components of the device, generate links between those components and generate hypotheses for the behaviour and function of the components.

To achieve this the *DR* algorithm must perform five steps:

1. identify the terminal nodes in the diagnostic hierarchy
   -these represent component nodes that have no child or sibling refinements
2. identify the component nodes in the diagnostic hierarchy related to the subsystem to be modelled (if required)
   -perform a pattern match with the known name or its derivatives (possibly acronyms) that match the subsystem
3. identify the parents and siblings of the nodes
   -backtrack from end node to parent node and tag
   -tag shared parents of a node
   -tag siblings of a parent
4. extract relations (behaviours) between the sibling nodes
   -cluster nodes related by parental nodes
   -movement from the terminal nodes to the root node represents symptomatic information (parameters)
5. match device model against background knowledge and output gaps for verification to the user

integration of various machine learning algorithms with partially hand-coded knowledge bases used in FBR or models that represent behaviour and function generated for MBR.

Automatic Component Behaviour & Function Generation: the *DR* algorithm

Hypothesis

Humans use failure-driven reasoning for successful device diagnosis and repair. As humans reason about diagnosis and repair they build primitive mental models of how a device functions and fails. The hypothesis for the *Diagnostic Remodeler* algorithm is that knowledge of failure and repair embodied in most structured diagnostic knowledge-based systems can be used to derive rudimentary device component models. The *DR* algorithm extracts rudimentary device component models from fault knowledge that represent structure, behaviour and function.

Motivation

A great deal of effort is expended hand-coding complex knowledge bases for diagnostic FBR. The artifacts these diagnostic systems are developed for are often expensive machines which have been designed and continuously modified so that no existing accurate schematic or design of their behaviour or resultant function remains. The J85-CAN-15 is a jet engine which is the first application of JETA. The J85-CAN-15 engine was designed in the 1950's and has easily had at least one modification a year since its launch. As a result of modifications and stresses of daily use (flying in the arctic and flying in desert heat) the jet engine is a very different device than was originally designed and sometimes displays inexplicable behaviour. No existing design schematics can completely capture the engines's behaviour or completely predict its function. It is also a very difficult device to diagnose. For these reasons a tool such as JETA was developed. As is typical with FBR systems, JETA does not diagnose novel faults. Learning the device component model, its behaviour and functionality using the FBR knowledge provides the technician with a tool that can achieve model-based diagnosis. For these reasons it was concluded that the *DR* algorithm should be implemented.

Background

If we follow the de Kleer [de Kleer and Williams 87] approach which represents device function as a set of components with behaviour. The device can be diagnosed by assuming a faulty component and enumerating the behavioural states that the fault propagates in the remainder of the device. This is compared to the behaviour that a technician is observing in attempting to isolate a problem. Model-based diagnosis can detect novel faults since the behaviour of the device is the basis of its knowledge representation and reasoning. Fault-based reasoning uses the faults in the functioning of a device rather than its actual behaviour, hence FBR cannot detect novel faults. However, model-based reasoning can lead to a combinatorial explosion in producing a diagnosis for complex systems (for example, an aircraft engine) and it does not lend itself to causal explanation.

I have implemented the *DR* algorithm [Abu-Hakima 93] intended to address the automatic generation of a functional model of a device from its fault knowledge. That implies the automatic generation of MBR knowledge from FBR knowledge. By extracting a functional model both fault and model-based diagnosis can be pursued in a single system gaining from the advantages of the two approaches while minimizing the disadvantages. The *DR* algorithm is being applied in the area of complex electromechanical devices, specifically jet engines.

Objectives of the DR Algorithm

DR is an algorithm that takes as input the fault knowledge of a device. It is also necessary to take as input some background knowledge related to the device to attempt to learn its full component structure and connectivity. *DR* initially extracts from the fault knowledge base all references to device components and subsystems. Given these components the algorithm backtracks through a diagnostic hierarchy of nodes to generate hypotheses for component connectivity. To further establish component connectivity, *DR* examines symptomatic or parametric knowledge that activates the diagnostic nodes. Symptomatic knowledge is knowledge of device failure which can be used to generate hypotheses about correct device function. This knowledge is used to derive behavioural knowledge between components.

Approach to the *DR* Algorithm

The top level design of the *DR* algorithm is shown in Figure 1. Two phases clearly divide the operation of the algorithm. In the first phase, an existing knowledge base that diagnoses a complex electromechanical system is used as input to *DR*. The Jet Engine Troubleshooting Assistant (JETA) is a system implemented to diagnose faults with aircraft engines [Halasz et al. 92]. Two types of background knowledge, device dependent and device independent knowledge are used as the second inputs to the DR algorithm. This device independent background knowledge is in the form of a component library and is general in nature, for example it includes knowledge that a pump delivers some liquid

model and fault-based diagnosis to deal with GDE's shortcomings.

Other MBR authors have argued about the definition of device functionality versus behaviour. Sticklen in [Sticklen et al. 88] describes modelling a device's functionality by:

- decomposing the device into sub-devices,
- stating abstractly the functions, goals and purpose of the device and
- representing the manner of achieving the device functions, goals and purpose.

A good definition of functionality is one which argues that function is the set of goals the device is intended or designed to achieve [Malin and Liefker 91]. As stated in the abstract the use of function in this paper is based on the perspective that function complements behaviour where the derived function is more abstract than the behaviour derived [Kumar 94].

Automatic Acquisition of Models for Diagnosis

Machine learning is a key approach in knowledge acquisition for diagnosis. Machine learning includes empirical and analytic learning. Empirical learning focuses on learning for classification (including learning rules from real or simulated data for diagnosis). Analytic learning addresses learning for problem solving tasks which include planning, diagnosis, design, natural language understanding, control and execution. There has been an explosion of work in machine learning in recent years. It is viewed as one of the key approaches of reducing the knowledge acquisition bottleneck [Boose 91; Gaines and Shaw 91].

The MOLTKE (MOdels, Learning and Temporal Knowledge in Expert systems) testbed for diagnosis under development at the University of Kaiserslautern, Germany is described in [Althoff et al. 90]. The system is designed to acquire device knowledge for diagnosis. It has an MBR mechanism for acquiring device models based on their components. A *component* of the model includes a name, ports to other components (with optional test costs), possible internal states (with optional test costs), behaviour of the component (either in state tables or rules that represent the constraints the component sets up between its ports and states), subparts and their interconnections (if the component is non-atomic), typical malfunctions with name and effects (model typical behavior when the component fails) and a priori probability of failure). No direct reference to device function is made. MOLTKE uses case-based reasoning to acquire and refine knowledge that is generalized to a fault-based hierarchy. It also uses explanation-based learning to refine the rules in the fault-based hierarchy to get the minimum reasoning paths for a solution. MOLTKE has been applied to a Computerized Numerical Control (CNC) machining center. It is also under investigation for the problem of driving mining machines.

ACES (Attitude Control Expert System) diagnoses anomalies in the attitude control system of the DSCS-III satellite [Pazzani 90]. ACES is fault-based (rules represented as Prolog predicates). A fault is confirmed or denied by comparing the observed behavior to that predicted with a simulator. In the case where the simulation denies the fault, the heuristic that proposed the fault is expanded to include the tests that the simulator performed to rule out the fault. In this manner the simulator generates expected behaviour given a particular fault. ACES uses explanation-based learning (EBL) to identify the conditions under which the heuristic will propose a fault that is denied. The author concludes that failure-driven learning finds sufficient conditions for ruling out a fault and success-driven learning finds sufficient conditions for establishing a fault (but not necessarily ruling others out). Pazzani's work is novel and very relevant to the refinement of fault-based knowledge using model-based reasoning and explanation-based learning.

There has been tremendous activity in machine learning in recent years. In empirical learning classification algorithms such as ID3 and AQ have been used to induce diagnostic rules from real or simulated data. Classification learning extracts rules from positive and negative examples. In analytic learning explanation-based learning has been used in the form of speedup learning to generalize diagnostic rules and shorten reasoning chains. I believe that neither classification nor EBL addresses the problem of *knowledge-rich* learning where structured knowledge is learned. Such rich knowledge would result from learning to produce hypothesis hierarchies such as those described in fault-based reasoning. In addition, learning from structured knowledge to produce new knowledge, such as learning a device model from its fault hierarchy has not been addressed. Learning complex structures especially for diagnosis is by no means an easy problem but it is one that needs to be further addressed by a combination of researchers in both the machine learning and diagnosis fields. Some researchers which have combined learning (empirical or analytic) with FBR and MBR have met with more success as exemplified by the complex systems above. I believe that the key to resolving the knowledge acquisition bottleneck in diagnosis lies in the

JETA's troubleshooting knowledge is represented as a diagnostic network that is hierarchical in nature. Each node in the network corresponds to a decision point in the troubleshooting process that mimics the problem solving strategy of an expert engine technician. At the top level JETA is attempting to reason about device function in terms of actual engine operation phases (i.e. start-up, acceleration, decceleration, etc.). It refines problems encountered in engine function at the phases of operation until it can identify symptoms that represent component failures. Thus, at the top level JETA can be thought of as reasoning about overall function and systematically refining its reasoning to failed behaviour on device components. As a result the links in JETA's diagnostic network represent relations directing the flow of control between nodes. The overall network is much broader than it is deep since there are many components and associated symptoms. The number of nodes along a network path varies from four to twelve in a network of approximately 200 nodes. Possible next moves in the network are represented as children of a node. Any node can have multiple parents since a component malfunction may be due to many causes. The troubleshooting knowledge is hand-coded at each diagnostic node as a frame using a custom command language. In JETA as in RATIONALE, advice generating slots are included in the frame and their contents are output to the user as diagnoses or procedures to follow to find a fault. In JETA, advice is supported with a schematic or a graph. An indexed database of schematics and graphs is kept so that only pointers to the database are kept in the frame. The current implementation of JETA links text, graphs and schematics.

Function in Model-Based Diagnosis

Model-based reasoning (MBR) for diagnosis concentrates on reasoning about the expected and correct functioning of a device. A device is modelled based on its components and their expected behaviour [Hamscher and Struss 90]. Such models range from quantitative ones to qualitative ones and all attempt to approximate device behaviour as accurately as possible. Once a device model is stabilized then a device's observed behaviour can be predicted from the model. If a discrepancy in behaviour is detected then possible candidates based on assumed component faults are generated. These candidates are generated based on assumptions that describe correct model behaviour. Sequential diagnosis is used to choose observations, augment a prediction for the candidate faults and update the list of candidates until a dominant candidate is found.

In MBR there are many conflicting definitions for models. They range from causal models represented as semantic networks with links specifying the relations between component nodes to full blown numerical simulations for complex systems and processes that have taken decades to perfect. Generating models is a key problem in MBR. Some researchers generate causal models, others generate models with structure and behaviour while others generate functional models for devices. Knowledge in models has thus far been hand-coded by experts that understand device component behaviour and function.

Davis was one of the earlier proponents of MBR. In [Davis 84] he describes a theory to exploit reasoning on the basis of device structure and behaviour. He defines paths of causal interpretation. He also describes constraint suspension used to identify which components are responsible for which faults. He argues that we need to balance complexity versus model completeness in diagnosis thus we need to enumerate and layer categories of failure. Quite a bit of work has followed Davis' examples and theories.

De Kleer and Williams published a key paper on MBR for diagnosis describing GDE, the General Diagnostic Engine [de Kleer and Williams 87]. GDE infers behaviour from device structure and functionality. It is applied to digital circuits and makes use of an ATMS (Assumption-Based Truth Maintenance System). This work forms the cornerstone of ATMS-based model-based reasoning systems. It was followed by many papers that criticized the approach as not computationally practical in diagnosing faults with large complex systems. Some of the papers criticizing GDE propose the use of hierarchical fault-based reasoning to reduce the computational complexity of de Kleer and Williams' approach. Struss has developed GDE+ which handles: simple dynamic aspects, multiple tests, hierarchical knowledge and unreliable observations [Struss 89]. GDE+ is a partial migration back to take advantage of heuristic or empirical diagnoses using fault-based reasoning. Struss points out that neither GDE nor GDE+ address: changing device structures, complex temporal behaviour (feedback), uncertainty or the use of qualitative models in reasoning. In [Struss and Dressler 89] the authors advocate the representation of a fault view for each component. They point out that a fault and a healthy view (state) for a component cannot be true in the same time instant (consistent belief rule). They also give the 'no good inference rule' where the node and its opposite which represents a fault cannot be true at the same instant. The ATMS is then modified to reason with the fault as well as the no-fault behaviour of a device. Their work gives excellent insight into combining

Automatic Extraction of Device Models from Fault Knowledge: the Diagnostic Remodeler (DR) Algorithm

NRC 37141

Suhayya Abu-Hakima
Knowledge Systems Laboratory, Institute for Information Technology
Building M-50, Montreal Road, National Research Council of Canada
Ottawa, Canada, K1A 0R6
email: suhayya@ai.iit.nrc.ca,
tel: (613) 993-8551, fax: (613) 952-7151

Abstract

This paper argues that automated knowledge acquisition for diagnosis has had limited success in both failure-driven diagnosis and model-based diagnosis. The paper describes fault-based and model-based reasoning for diagnosis and surveys some of the approaches to knowledge acquisition in both areas. The Diagnostic Remodeler (DR) algorithm has been implemented for the automated generation of behavioural component models with function from fault-based knowledge. The use of function in this paper is based on the perspective that function complements behaviour where the derived function is more abstract than the behaviour derived by DR [Kumar 94]. DR uses as its first application example the fault-based knowledge base of the Jet Engine Troubleshooting Assistant (JETA). DR is used to extract the model of the Main Fuel System using the knowledge base and two types of background knowledge as input: device dependent and device independent knowledge. This paper is the first presentation of preliminary results of the implemented DR algorithm.

Function in Fault-Based Diagnosis

Fault-based reasoning (FBR) is used in many diagnostic systems. Knowledge in FBR is largely based on maintenance manuals and interviews with experts intended to capture heuristic knowledge about the maintenance and repair of a device or process. The maintenance and repair is directed at keeping a device functioning in a predictable manner. The knowledge in these systems is often represented as hand-coded rules or frames which are organized into troubleshooting hierarchies. At the top level of the hierarchy is the general knowledge representing a problem with device function. This general problem is refined systematically until the leaf nodes of the hierarchy which represent physical repairs to the device are reached. Once these repairs are achieved by a human technician some diagnostic systems re-test to confirm that the symptoms and diagnosed faults are cleared through backtracking in the hierarchy.

FBR systems have evolved considerably since the development of MYCIN [Scott et al. 77; Clancey 86]. MYCIN was developed to provide advice treatment for microbial infections. The MYCIN programs started with hand-coded rules which later evolved into meta-rules in NEOMYCIN to provide some structure to an otherwise flat knowledge base. The MYCIN approach remains a very widely used approach in FBR systems as described in the literature review of [Abu-Hakima 94].

Diagnosis is often referred to as a classification problem. Chandrasekaran and his colleagues developed MDX, a system that diagnosis a form of liver disease, cholestasis [Chanrasekaran et al. 79]. MDX has a diagnostic hierarchy which is referred to as a conceptual hierarchy since it guides the reasoner globally through diagnoses clustered as concepts that establish local contexts. Local uncertainties and hand-coded knowledge represented in frames are used to guide the diagnosis [Chandrasekaran and Tanner 86]. MDX has served as a model for many well-structured diagnostic systems including RATIONALE [Abu-Hakima 88] and JETA [Halasz et al. 92].

RATIONALE is a workstation diagnosis system that reasons explicitly so that it may support the user with sophisticated explanations of diagnoses that help justify diagnostic system behaviour and clarify reasoning. This approach was found to be ideal for explicitly representing causal knowledge of problems with device function so that it may be explained [Abu-Hakima and Oppacher 90]. RATIONALE diagnoses faults with Xerox workstations. It generates dynamic and static template-based explanations that include why, how and what-if responses. Explanation remains a major objective of FBR systems and most systems have why and how explanation but do not necessarily generate hypothetical (what-if) ones. RATIONALE's knowledge is in hand-coded frames.

The Jet Engine Troubleshooting Assistant (JETA) is a tool developed to assist a technician in diagnosing aircraft engines using a hypermedia interface which provides contextual help. For a diagnostic application to properly support hypermedia, one requires a structured manner by which to represent the knowledge, reason about it interactively, display it dynamically and explain it to the user (see [Abu-Hakima et al. 93] for a thorough description of JETA's hypermedia interface). JETA's knowledge representation and reasoning strategies are more flexible than those of other diagnostic systems including RATIONALE's.

References

Abu-Hakima, S. [1988], *RATIONALE: A Tool for Developing Knowledge-Based Systems that Explain by Reasoning Explicitly.* Masters thesis, Carleton University, Ottawa, Canada. May 1988.

Abu-Hakima, S., and Oppacher, F. [1990],Improving explanations in knowledge-based systems: RATIONALE. Knowledge Acquisition journal (2). 301-343. December 1990.

Abu-Hakima, S. [1994], Diagnostic techniques in knowledge-based systems: a review of approaches, applications and issues. 100 p. To be published as an NRC ERA (1994).

Abu-Hakima, S. [1993], Automatic Knowledge Acquisition in Diagnosis. Proceedings of DX-93, Fourth International Workshop on Principles of Diagnosis. Aberystwyth, Wales. 236-250. (1993).

Abu-Hakima, S., Halasz, M., and Phan, S. [1993], An Approach to Hypermedia in Diagnostic Systems. Chapter 10 in Intelligent Multimedia Interfaces, Edited by Mark Maybury, AAAI Press (1993).

Althoff, K-D., Maurer, F., and Rehbold, R. [1990],"Multiple knowledge acquisition strategies in MOLTKE." *Current trends in knowledge acquisition*, Published by IOS, Amsterdam, Netherlands. pp. 21-40, (1990).

Boose, J.H. [1991], "Knowledge acquisition tools, methods and mediating representations", *Knowledge Acquisition for Knowledge-Based Systems*, Edited by H. Motoda, R. Mizoguchi, J. Boose and B. Gaines. Published by IOS Press, the Netherlands. pp. 25-62, (1991).

Chandrasekaran, B., Gomez, F., Mittal, S. and Smith, J.W. [1979], "An approach to medical diagnosis based on conceptual structures". Proceedings of the Joint Conference on Artificial Intelligence, 134-142.(1979).

Chandrasekaran, B. and Tanner, M.C. [1986], "Uncertainty handling in expert systems: uniform vs. task-specific formalisms". In Kanal L.N. and Lemmer J., editors, Uncertainty in Artificial Intelligence, North Holland Publishing.(1986).

Clancey, W.J. [1986], "From GUIDON to NEOMYCIN and HERACLES in twenty short lessons: ORN final report 1979-1985". The AI Magazine, pp. 40-60,(1986).

Davis, R. [1984], "Diagnostic reasoning based on structure and behaviour", Artificial Intelligence, Vol. 24, pp. 347-410 (1984).

de Kleer, J., Williams, B.C. [1987], "Diagnosing Multiple Faults", *Artificial Intelligence*, vol. 32, (1987).

Gaines, B. and Shaw, M. [1991], "Foundations of Knowledge Acquisition", *Knowledge Acquisition for Knowledge-Based Systems*, Edited by H. Motoda, R. Mizoguchi, J. Boose and B. Gaines. Published by IOS Press, the Netherlands. pp. 3-23, (1991).

Halasz, M., Davidson, P., Abu-Hakima, S., and Phan, S. [1992], JETA: A Knowledge-based Approach to Aircraft Gas Turbine Engine Maintenance. Journal of Applied Intelligence, 2, pp. 25-46, Kluwer Academic Publishers, NRC 31832, (1992).

Hamscher, W. and Struss, P. [1990], "Model-Based Diagnosis", *AAAI-90 Tutorial Notes*, Received at the eighth national conference of artificial intelligence, Boston, Massachusetts. July 29, 1990. pp. 1-179, (1990).

Kumar, A. [1994], "Reasoning about function: AAAI-93 Workshops Summary Report", AI MAgazine, Spring 1994, Vol. 15, No. 1, (1994).

Malin, J.T., and Liefker, D.B. [1991], "Representing functions/procedures and processes/structures for analysis of effects of failures on functions and operations", Proceedings of 1991 NASA Goddard Conference on Space Applications of Artificial Intelligence, Greenbelt, Maryland, pp. 141-151, (May 13-15 1991).

Modesitt, K.L.[1990], "Inductive knowledge acquisition experience with commercial tools for space shuttle main engine testing", *Proceedings of the fifth conference on Artificial Intelligence for space applications*, NASA Marshall Flight Center, pp. 487-496, (May 1990).

Pazzani, M.J. [1990], "Learning in order to avoid search in logic programming", *Computers & Mathematics with Applications*, 20(9-10), pp. 101-10, (1990).

Scott, A.C., Clancey, W.J., Davis, R. and Shortliffe, E.H. [1977], "Explanation capabilities of production-based consultation systems". Stanford University, Dept. of Computer Science, Report No. STAN-CS-77-593,(1977).

Sticklen, J., Chandrasekaran, B., and Bond, W.E. [1988], "Distributed causal reasoning for knowledge acquisition: a functional approach to device understanding", 3rd AAAI sponsored Knowledge Acquisition Workshop for Knowledge-Based Systems, Banff, Canada, pp. 34-1 to 34-18, (1988).

Struss, P. [1989], "New Techniques in Model-based Diagnosis", *Proceedings of Knowledge-based Computer Systems*, Bombay, India, (11-13 December 1989).

Struss, P., Dressler, O. [1989], "Physical Negation - Integrating Fault Models into the General Diagnostic Engine", *Proceedings of the 11th International Joint Conference on Artificial Intelligence (IJCAI-89)*, Detroit, MI, (20-25 August 1989).

```
[function(main_fuel_nozzles,flow_control(WF+)),
input(main_fuel_nozzles,flow(WF+)),
output(main_fuel_nozzles,flow(WF+)),
regulator(main_fuel_nozzles,regulation_control(N+)),
behaviour(main_fuel_nozzles,behaviour_is_proportional(WF+,N+,
[increase_in(N+),increases(WF+),decrease_in(N+),decreases(WF+)]))],

[[main_fuel_nozzles,
[[gap_for_mode,fuel_flow_control,
extracted,EGT,input,[N+,WF],output,[WF+]],
[gap_for_mode,fuel_flow_control,
extracted,RPM,input,[N+,WF],output,[WF+]],
[gap_for_mode,fuel_flow_control,
extracted,N1,input,[N+,WF],output,[WF+]],[]]]],


[function(main_fuel_pump,delivers(WF+)),
input(main_fuel_pump,fluid(WF+)),
output(main_fuel_pump,fluid(WF+)),
regulator(main_fuel_pump,N1+),
behaviour(for(main_fuel_pump),
behaviour_is_proportional(WF+,N1+,
[increase_in(N1+),increases(WF+),decrease_in(N1+),decreases(WF+)])
)],

[function(main_fuel_pump_fuel_filter,filters(WF+)),
input(main_fuel_pump_fuel_filter,fluid(WF+)),
output(main_fuel_pump_fuel_filter,fluid(WF+)),
regulator(main_fuel_pump_fuel_filter,none),
behaviour(for(main_fuel_pump_fuel_filter),
behaviour_is_proportional(WF+,WF+,
[increase_in(WF+),increases(WF+),decrease_in(WF+),decreases(WF+)]
))],

[function(overspeed_governor,controls([WF/P3-])),
input(overspeed_governor,control([[N+]])),
output(overspeed_governor,control([[WF/P3-]])),
regulator(overspeed_governor,regulation_control([[N+]])),
behaviour(overspeed_governor,
behaviour_is_piecewise_linear([[[(mode overspeed_regulation),
input_control_variable(N+)]]],
[[[[the_behaviour_is_proportional_for(overspeed_regulation),
the_input_control_variable(N+),
results_in_regulated_output_variable(WF/P3-),
at_limit(N+),decrease(WF/P3-),increases(N+),increases(WF/P3-)]]]]))],
```

Issues

There are several issues that I intended to answer in the implementation of the *DR* algorithm. The first is what is the exact form of the learned model when some or no background knowledge is used. If no background knowledge is used is the model much more than a causal rather than a component behaviour model with explicit representation of function? The answer here is that a minimum amount of device dependent knowledge is used to map the JETA notation to text as shown in the sample output of step 4. If no device independent background knowledge (such as that I started for an the engine component library) is used then the extraction of gaps between the JETA-encoded model and a general one is not possible. Using no background knowledge it is possible to extract a component-to-component model with explicit parameteric paths from FBR knowledge. Extracting the relationships on these behavioural paths (increasing/decreasing/no change) requires generalized background knowledge.

The *DR* algorithm has been implemented as a general algorithm useful in generating models for devices other than jet engines. It is not obvious that its background knowledge will make it specific for generating a particular model. However the FBR knowledge used as input will make it specific to generating a model for a particular device. To answer this question I have generated a 30-node knowledge base for the diagnosis of a coffee maker (a very different device than an engine). I have successfully generated a component behaviour model with explicit function for all the coffee maker components. This illustrates the generality of the algorithm for devices. However, another question is whether or not the algorithm can be generalized further so that it diagnosis abstract (e.g. software) versus physical (e.g. jet engine, coffee makers, etc.) systems.

Finally, the *DR* algorithm requires a highly structured FBR knowledge base. One key question is what criteria will allow it to extract a causal model from a rule versus a frame-based fault model?

Conclusion

This paper argues that automated knowledge acquisition of models for diagnosis has had limited success in both failure-driven diagnosis and model-based diagnosis. The *DR* algorithm for the automated generation of behavioural models with an explicit representation of function from fault-based knowledge is described. An example of fault-based knowledge from the Jet Engine Troubleshooting Assistant is used to demonstrate how a model of the main fuel system of a jet engine can be extracted with *DR* from the fault knowledge.

-map out the identified components of the subsystem

-relate the components through shared parameters

-match derived component model with device dependent knowledge to derive exact parameter behaviours

-match derived component model with to library component model to extract function and uncover gaps

An Example: Extracting Models from JETA's Fault Knowledge Using DR

An analysis of the JETA fault knowledge shows layers of knowledge which can be visualized as leaves of diagnostic trees. The topmost layer is an entry point to jet engine faults and subsequent layers organize the faults into various branches. The phases of operation branches lead to various symptomatic nodes labelled as snags. These snags in turn are refinable down to repair and replacement nodes which represent the terminal nodes of the diagnostic hierarchy[1]. If one examines the knowledge encoded in these terminal nodes more closely one discovers that they represent faults directly on physical engine components. These physical component fault nodes can be grouped into those affecting one of thirteen subsystems by their nomenclature. One can follow the five steps of the *DR* algorithm introduced above to discover the behavioural and functional component model for the main fuel system of the jet engine.

Step 1:

One can identify 9 replace nodes through the JETA node frame slot 'node-type'.

Step 2

If one takes a specific subsystem, the MFS (Main Fuel System), one can extract the names of 3 fuel system replacement nodes by pattern matching with the node nomenclature *N-MFS-XXX (this is an internal representation that was used by the knowledge engineer to distinguish between nodes):

1. main fuel control (MFC)
2. overspeed governor for MFC (OSG)
3. main fuel pump supplying MFC (MFP)

Step 3

For each of the 3replacement nodes parents connecting sibling nodes can be extracted, for example:

- the MFC and MFP nodes share the parent node loss of fuel flow
- the OSG node shares with the MFC the engine speed hang-up parent node
- the main fuel control (MFC) delivers fuel to the main engine fuel nozzles (FN)
- the pressurizing and drain valve (PDV) is connected to the main fuel nozzle FN and both share fuel flow

Step 4

A causal topological network can be the basis for hypothesized component-behaviour relations. Sibling nodes are clustered based on shared parent links. Example DR output relations that form part of the network include:

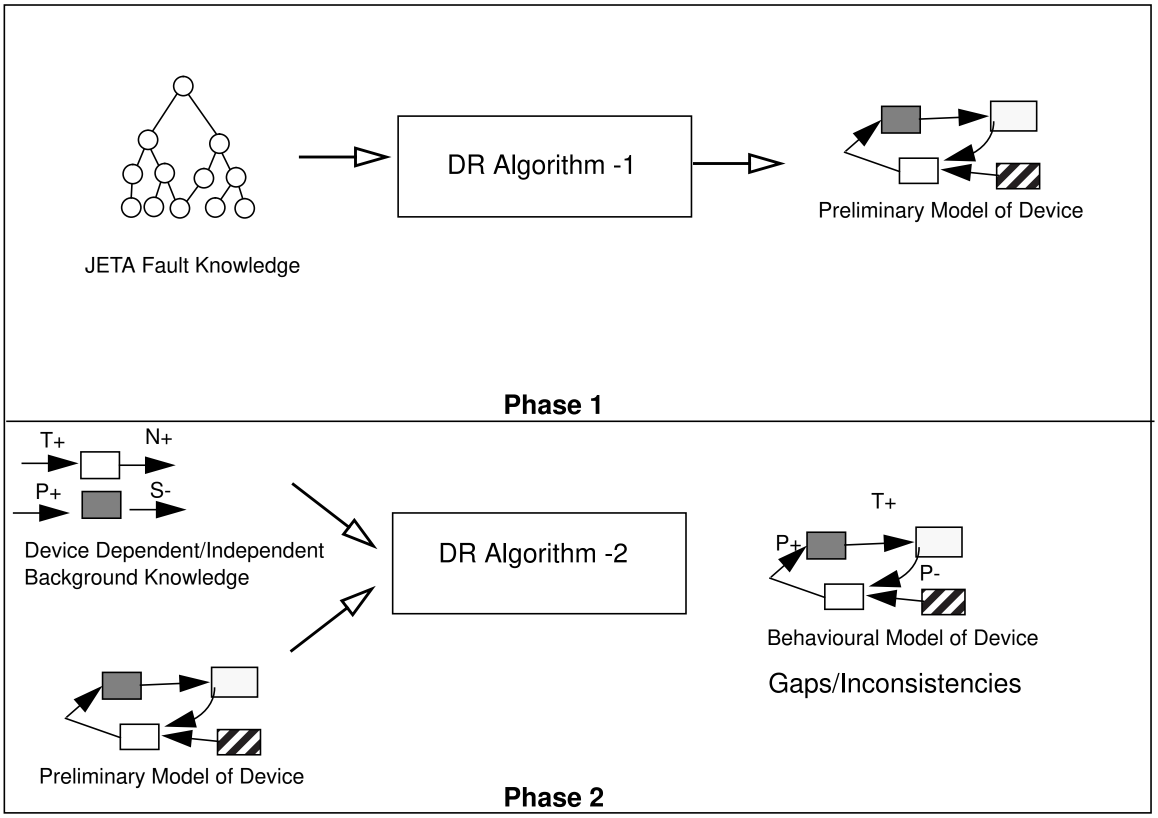
```
[[main_fuel_control,
is_a([control,for,[steady_speed_control,speed_cutback_control,
    acceleration_fuel_limit_control, deceleration_fuel_limit_control,
    variable_geometry_scheduling,proportional_speed_control,
    fru_fuel_selection]]),
and_is_connected_to(main_fuel_pump),
with_connectivity_parameter([weight_of_fuel_flow])],

[main_fuel_control_fuel_filter,
is_a([filter,for,[filtering_fuel]]),
and_is_connected_to(main_fuel_control),
with_connectivity_parameter([weight_of_fuel_flow])],

[main_fuel_nozzles,
is_a([nozzle,for,[fuel_flow_control]]),
and_is_connected_to(main_fuel_control),
with_connectivity_parameter([measured_rpm_engine_speed,
    single_spool_engine_speed,weight_of_fuel_flow])],

[main_fuel_nozzles,
is_a([nozzle,for,[fuel_flow_control]]),
and_is_connected_to(pressurizing_and_drain_valve),
with_connectivity_parameter(fuel_pump_inlet_pressure[])],

[overspeed_governor,
is_a([con-
trol,for,[overspeed_regulation],[fuel_control_overspeed_governor]]),
and_is_connected_to(main_fuel_control),
with_connectivity_parameter([measured_rpm_engine_speed,
    single_spool_engine_speed,weight_of_fuel_flow])],

[pressurizing_and_drain_valve,
is_a([control,for,[fuel_overflow]]),
and_is_connected_to(main_fuel_control),
with_connectivity_parameter([weight_of_fuel_flow])]]

.......
```

Step 5

Step 4 output is matched against device independent/ dependent background knowledge and gaps identified. For example (note only the gaps for the main fuel nozzles are identified):

[1] The diagnostic hierarchy is referred to as a network since it includes relations that are not directly inherited that allow the JETA reasoner to jump around between nodes thus forming more of a network than a hierarchy.

Figure 1: The Phases of the Diagnostic Remodeler (DR) Algorithm



from a sink to a source and needs a signal (control signal such as pressure) to increase or decrease the flow of liquid. It also includes some knowledge about feedback control in moderating the flow of a liquid to a source based on the level of the liquid at the source. The device dependent knowledge includes the specifics of control modes for a type of control of the device. For example, the Main Fuel Control includes modes on steady engine speed control, speed cutback control, deceleration fuel limit control, variable geometry scheduling, etc. In a general sense these are all control modes but in a specific sense they have different control signals (in one case it is engine speed, in another it is fuel flow and in yet another mode it is the air bleed valve positions). The device dependent and independent background knowledge is matched against the extracted JETA knowledge to uncover gaps.

To achieve the knowledge-rich learning proposed as the output for *DR* one requires the use of a structured and explicit knowledge representation that can adequately represent diagnostic causality. This is achieved by extracting a model of the connections between the components in the subsystem to be modelled. These connections are further used to extract the variables (such as engine speed, fuel flow, etc.) that typify the behaviour between components. The second phase of the *DR* algorithm matches the functional background knowledge of a device exemplified by its device model of components and their behaviours with the original fault knowledge. The purpose of the second phase is to find inconsistencies and gaps in the fault knowledge. The gaps discovered in the fault knowledge could then be used to diagnose novel faults.

The objective of the *DR* algorithm is to discover and refine a component behavioural model with explicit function. In the most general sense the algorithm must identify the components of the device, generate links between those components and generate hypotheses for the behaviour and function of the components.

To achieve this the *DR* algorithm must perform five steps:

1. identify the terminal nodes in the diagnostic hierarchy
   -these represent component nodes that have no child or sibling refinements
2. identify the component nodes in the diagnostic hierarchy related to the subsystem to be modelled (if required)
   -perform a pattern match with the known name or its derivatives (possibly acronyms) that match the subsystem
3. identify the parents and siblings of the nodes
   -backtrack from end node to parent node and tag
   -tag shared parents of a node
   -tag siblings of a parent
4. extract relations (behaviours) between the sibling nodes
   -cluster nodes related by parental nodes
   -movement from the terminal nodes to the root node represents symptomatic information (parameters)
5. match device model against background knowledge and output gaps for verification to the user

integration of various machine learning algorithms with partially hand-coded knowledge bases used in FBR or models that represent behaviour and function generated for MBR.

Automatic Component Behaviour & Function Generation: the *DR* algorithm

Hypothesis

Humans use failure-driven reasoning for successful device diagnosis and repair. As humans reason about diagnosis and repair they build primitive mental models of how a device functions and fails. The hypothesis for the *Diagnostic Remodeler* algorithm is that knowledge of failure and repair embodied in most structured diagnostic knowledge-based systems can be used to derive rudimentary device component models. The *DR* algorithm extracts rudimentary device component models from fault knowledge that represent structure, behaviour and function.

Motivation

A great deal of effort is expended hand-coding complex knowledge bases for diagnostic FBR. The artifacts these diagnostic systems are developed for are often expensive machines which have been designed and continuously modified so that no existing accurate schematic or design of their behaviour or resultant function remains. The J85-CAN-15 is a jet engine which is the first application of JETA. The J85-CAN-15 engine was designed in the 1950's and has easily had at least one modification a year since its launch. As a result of modifications and stresses of daily use (flying in the arctic and flying in desert heat) the jet engine is a very different device than was originally designed and sometimes displays inexplicable behaviour. No existing design schematics can completely capture the engines's behaviour or completely predict its function. It is also a very difficult device to diagnose. For these reasons a tool such as JETA was developed. As is typical with FBR systems, JETA does not diagnose novel faults. Learning the device component model, its behaviour and functionality using the FBR knowledge provides the technician with a tool that can achieve model-based diagnosis. For these reasons it was concluded that the *DR* algorithm should be implemented.

Background

If we follow the de Kleer [de Kleer and Williams 87] approach which represents device function as a set of components with behaviour. The device can be diagnosed by assuming a faulty component and enumerating the behavioural states that the fault propagates in the remainder of the device. This is compared to the behaviour that a technician is observing in attempting to isolate a problem. Model-based diagnosis can detect novel faults since the behaviour of the device is the basis of its knowledge representation and reasoning. Fault-based reasoning uses the faults in the functioning of a device rather than its actual behaviour, hence FBR cannot detect novel faults. However, model-based reasoning can lead to a combinatorial explosion in producing a diagnosis for complex systems (for example, an aircraft engine) and it does not lend itself to causal explanation.

I have implemented the *DR* algorithm [Abu-Hakima 93] intended to address the automatic generation of a functional model of a device from its fault knowledge. That implies the automatic generation of MBR knowledge from FBR knowledge. By extracting a functional model both fault and model-based diagnosis can be pursued in a single system gaining from the advantages of the two approaches while minimizing the disadvantages. The *DR* algorithm is being applied in the area of complex electromechanical devices, specifically jet engines.

Objectives of the DR Algorithm

DR is an algorithm that takes as input the fault knowledge of a device. It is also necessary to take as input some background knowledge related to the device to attempt to learn its full component structure and connectivity. *DR* initially extracts from the fault knowledge base all references to device components and subsystems. Given these components the algorithm backtracks through a diagnostic hierarchy of nodes to generate hypotheses for component connectivity. To further establish component connectivity, *DR* examines symptomatic or parametric knowledge that activates the diagnostic nodes. Symptomatic knowledge is knowledge of device failure which can be used to generate hypotheses about correct device function. This knowledge is used to derive behavioural knowledge between components.

Approach to the *DR* Algorithm

The top level design of the *DR* algorithm is shown in Figure 1. Two phases clearly divide the operation of the algorithm. In the first phase, an existing knowledge base that diagnoses a complex electromechanical system is used as input to *DR*. The Jet Engine Troubleshooting Assistant (JETA) is a system implemented to diagnose faults with aircraft engines [Halasz et al. 92]. Two types of background knowledge, device dependent and device independent knowledge are used as the second inputs to the DR algorithm. This device independent background knowledge is in the form of a component library and is general in nature, for example it includes knowledge that a pump delivers some liquid

model and fault-based diagnosis to deal with GDE's shortcomings.

Other MBR authors have argued about the definition of device functionality versus behaviour. Sticklen in [Sticklen et al. 88] describes modelling a device's functionality by:

- decomposing the device into sub-devices,
- stating abstractly the functions, goals and purpose of the device and
- representing the manner of achieving the device functions, goals and purpose.

A good definition of functionality is one which argues that function is the set of goals the device is intended or designed to achieve [Malin and Liefker 91]. As stated in the abstract the use of function in this paper is based on the perspective that function complements behaviour where the derived function is more abstract than the behaviour derived [Kumar 94].

Automatic Acquisition of Models for Diagnosis

Machine learning is a key approach in knowledge acquisition for diagnosis. Machine learning includes empirical and analytic learning. Empirical learning focuses on learning for classification (including learning rules from real or simulated data for diagnosis). Analytic learning addresses learning for problem solving tasks which include planning, diagnosis, design, natural language understanding, control and execution. There has been an explosion of work in machine learning in recent years. It is viewed as one of the key approaches of reducing the knowledge acquisition bottleneck [Boose 91; Gaines and Shaw 91].

The MOLTKE (MOdels, Learning and Temporal Knowledge in Expert systems) testbed for diagnosis under development at the University of Kaiserslautern, Germany is described in [Althoff et al. 90]. The system is designed to acquire device knowledge for diagnosis. It has an MBR mechanism for acquiring device models based on their components. A *component* of the model includes a name, ports to other components (with optional test costs), possible internal states (with optional test costs), behaviour of the component (either in state tables or rules that represent the constraints the component sets up between its ports and states), subparts and their interconnections (if the component is non-atomic), typical malfunctions with name and effects (model typical behavior when the component fails) and a priori probability of failure). No direct reference to device function is made. MOLTKE uses case-based reasoning to acquire and refine knowledge that is generalized to a fault-based hierarchy. It also uses explanation-based learning to refine the rules in the fault-based hierarchy to get the minimum reasoning paths for a solution. MOLTKE has been applied to a Computerized Numerical Control (CNC) machining center. It is also under investigation for the problem of driving mining machines.

ACES (Attitude Control Expert System) diagnoses anomalies in the attitude control system of the DSCS-III satellite [Pazzani 90]. ACES is fault-based (rules represented as Prolog predicates). A fault is confirmed or denied by comparing the observed behavior to that predicted with a simulator. In the case where the simulation denies the fault, the heuristic that proposed the fault is expanded to include the tests that the simulator performed to rule out the fault. In this manner the simulator generates expected behaviour given a particular fault. ACES uses explanation-based learning (EBL) to identify the conditions under which the heuristic will propose a fault that is denied. The author concludes that failure-driven learning finds sufficient conditions for ruling out a fault and success-driven learning finds sufficient conditions for establishing a fault (but not necessarily ruling others out). Pazzani's work is novel and very relevant to the refinement of fault-based knowledge using model-based reasoning and explanation-based learning.

There has been tremendous activity in machine learning in recent years. In empirical learning classification algorithms such as ID3 and AQ have been used to induce diagnostic rules from real or simulated data. Classification learning extracts rules from positive and negative examples. In analytic learning explanation-based learning has been used in the form of speedup learning to generalize diagnostic rules and shorten reasoning chains. I believe that neither classification nor EBL addresses the problem of *knowledge-rich* learning where structured knowledge is learned. Such rich knowledge would result from learning to produce hypothesis hierarchies such as those described in fault-based reasoning. In addition, learning from structured knowledge to produce new knowledge, such as learning a device model from its fault hierarchy has not been addressed. Learning complex structures especially for diagnosis is by no means an easy problem but it is one that needs to be further addressed by a combination of researchers in both the machine learning and diagnosis fields. Some researchers which have combined learning (empirical or analytic) with FBR and MBR have met with more success as exemplified by the complex systems above. I believe that the key to resolving the knowledge acquisition bottleneck in diagnosis lies in the

JETA's troubleshooting knowledge is represented as a diagnostic network that is hierarchical in nature. Each node in the network corresponds to a decision point in the troubleshooting process that mimics the problem solving strategy of an expert engine technician. At the top level JETA is attempting to reason about device function in terms of actual engine operation phases (i.e. start-up, acceleration, decceleration, etc.). It refines problems encountered in engine function at the phases of operation until it can identify symptoms that represent component failures. Thus, at the top level JETA can be thought of as reasoning about overall function and systematically refining its reasoning to failed behaviour on device components. As a result the links in JETA's diagnostic network represent relations directing the flow of control between nodes. The overall network is much broader than it is deep since there are many components and associated symptoms. The number of nodes along a network path varies from four to twelve in a network of approximately 200 nodes. Possible next moves in the network are represented as children of a node. Any node can have multiple parents since a component malfunction may be due to many causes. The troubleshooting knowledge is hand-coded at each diagnostic node as a frame using a custom command language. In JETA as in RATIONALE, advice generating slots are included in the frame and their contents are output to the user as diagnoses or procedures to follow to find a fault. In JETA, advice is supported with a schematic or a graph. An indexed database of schematics and graphs is kept so that only pointers to the database are kept in the frame. The current implementation of JETA links text, graphs and schematics.

Function in Model-Based Diagnosis

Model-based reasoning (MBR) for diagnosis concentrates on reasoning about the expected and correct functioning of a device. A device is modelled based on its components and their expected behaviour [Hamscher and Struss 90]. Such models range from quantitative ones to qualitative ones and all attempt to approximate device behaviour as accurately as possible. Once a device model is stabilized then a device's observed behaviour can be predicted from the model. If a discrepancy in behaviour is detected then possible candidates based on assumed component faults are generated. These candidates are generated based on assumptions that describe correct model behaviour. Sequential diagnosis is used to choose observations, augment a prediction for the candidate faults and update the list of candidates until a dominant candidate is found.

In MBR there are many conflicting definitions for models. They range from causal models represented as semantic networks with links specifying the relations between component nodes to full blown numerical simulations for complex systems and processes that have taken decades to perfect. Generating models is a key problem in MBR. Some researchers generate causal models, others generate models with structure and behaviour while others generate functional models for devices. Knowledge in models has thus far been hand-coded by experts that understand device component behaviour and function.

Davis was one of the earlier proponents of MBR. In [Davis 84] he describes a theory to exploit reasoning on the basis of device structure and behaviour. He defines paths of causal interpretation. He also describes constraint suspension used to identify which components are responsible for which faults. He argues that we need to balance complexity versus model completeness in diagnosis thus we need to enumerate and layer categories of failure. Quite a bit of work has followed Davis' examples and theories.

De Kleer and Williams published a key paper on MBR for diagnosis describing GDE, the General Diagnostic Engine [de Kleer and Williams 87]. GDE infers behaviour from device structure and functionality. It is applied to digital circuits and makes use of an ATMS (Assumption-Based Truth Maintenance System). This work forms the cornerstone of ATMS-based model-based reasoning systems. It was followed by many papers that criticized the approach as not computationally practical in diagnosing faults with large complex systems. Some of the papers criticizing GDE propose the use of hierarchical fault-based reasoning to reduce the computational complexity of de Kleer and Williams' approach. Struss has developed GDE+ which handles: simple dynamic aspects, multiple tests, hierarchical knowledge and unreliable observations [Struss 89]. GDE+ is a partial migration back to take advantage of heuristic or empirical diagnoses using fault-based reasoning. Struss points out that neither GDE nor GDE+ address: changing device structures, complex temporal behaviour (feedback), uncertainty or the use of qualitative models in reasoning. In [Struss and Dressler 89] the authors advocate the representation of a fault view for each component. They point out that a fault and a healthy view (state) for a component cannot be true in the same time instant (consistent belief rule). They also give the 'no good inference rule' where the node and its opposite which represents a fault cannot be true at the same instant. The ATMS is then modified to reason with the fault as well as the no-fault behaviour of a device. Their work gives excellent insight into combining

Automatic Extraction of Device Models from Fault Knowledge: the Diagnostic Remodeler (DR) Algorithm

NRC 37141

Suhayya Abu-Hakima
Knowledge Systems Laboratory, Institute for Information Technology
Building M-50, Montreal Road, National Research Council of Canada
Ottawa, Canada, K1A 0R6
email: suhayya@ai.iit.nrc.ca,
tel: (613) 993-8551, fax: (613) 952-7151

Abstract

This paper argues that automated knowledge acquisition for diagnosis has had limited success in both failure-driven diagnosis and model-based diagnosis. The paper describes fault-based and model-based reasoning for diagnosis and surveys some of the approaches to knowledge acquisition in both areas. The Diagnostic Remodeler (DR) algorithm has been implemented for the automated generation of behavioural component models with function from fault-based knowledge. The use of function in this paper is based on the perspective that function complements behaviour where the derived function is more abstract than the behaviour derived by DR [Kumar 94]. DR uses as its first application example the fault-based knowledge base of the Jet Engine Troubleshooting Assistant (JETA). DR is used to extract the model of the Main Fuel System using the knowledge base and two types of background knowledge as input: device dependent and device independent knowledge. This paper is the first presentation of preliminary results of the implemented DR algorithm.

Function in Fault-Based Diagnosis

Fault-based reasoning (FBR) is used in many diagnostic systems. Knowledge in FBR is largely based on maintenance manuals and interviews with experts intended to capture heuristic knowledge about the maintenance and repair of a device or process. The maintenance and repair is directed at keeping a device functioning in a predictable manner. The knowledge in these systems is often represented as hand-coded rules or frames which are organized into troubleshooting hierarchies. At the top level of the hierarchy is the general knowledge representing a problem with device function. This general problem is refined systematically until the leaf nodes of the hierarchy which represent physical repairs to the device are reached. Once these repairs are achieved by a human technician some diagnostic systems re-test to confirm that the symptoms and diagnosed faults are cleared through backtracking in the hierarchy.

FBR systems have evolved considerably since the development of MYCIN [Scott et al. 77; Clancey 86]. MYCIN was developed to provide advice treatment for microbial infections. The MYCIN programs started with hand-coded rules which later evolved into meta-rules in NEOMYCIN to provide some structure to an otherwise flat knowledge base. The MYCIN approach remains a very widely used approach in FBR systems as described in the literature review of [Abu-Hakima 94].

Diagnosis is often referred to as a classification problem. Chandrasekaran and his colleagues developed MDX, a system that diagnosis a form of liver disease, cholestasis [Chanrasekaran et al. 79]. MDX has a diagnostic hierarchy which is referred to as a conceptual hierarchy since it guides the reasoner globally through diagnoses clustered as concepts that establish local contexts. Local uncertainties and hand-coded knowledge represented in frames are used to guide the diagnosis [Chandrasekaran and Tanner 86]. MDX has served as a model for many well-structured diagnostic systems including RATIONALE [Abu-Hakima 88] and JETA [Halasz et al. 92].

RATIONALE is a workstation diagnosis system that reasons explicitly so that it may support the user with sophisticated explanations of diagnoses that help justify diagnostic system behaviour and clarify reasoning. This approach was found to be ideal for explicitly representing causal knowledge of problems with device function so that it may be explained [Abu-Hakima and Oppacher 90]. RATIONALE diagnoses faults with Xerox workstations. It generates dynamic and static template-based explanations that include why, how and what-if responses. Explanation remains a major objective of FBR systems and most systems have why and how explanation but do not necessarily generate hypothetical (what-if) ones. RATIONALE's knowledge is in hand-coded frames.

The Jet Engine Troubleshooting Assistant (JETA) is a tool developed to assist a technician in diagnosing aircraft engines using a hypermedia interface which provides contextual help. For a diagnostic application to properly support hypermedia, one requires a structured manner by which to represent the knowledge, reason about it interactively, display it dynamically and explain it to the user (see [Abu-Hakima et al. 93] for a thorough description of JETA's hypermedia interface). JETA's knowledge representation and reasoning strategies are more flexible than those of other diagnostic systems including RATIONALE's.