in working notes of AAAI '94 Workshop on CBR, D. Aha ed., Seattle Washington, August, 1994.

A Comparison of Model-Based and Incremental Case-Based Approaches to Electronic Fault Diagnosis

Pádraig Cunningham
Department of Computer Science
Trinity College Dublin
College Green
Dublin 2
Ireland
Padraig.Cunningham@cs.tcd.ie

Barry Smyth
Hitachi Dublin Laboratory
16 Westland Row
Dublin 2
Ireland
barry@hdl.ie

Abstract

CBR seems well suited to fault diagnosis because diagnostic episodes naturally form cases and much of expert competence seems to be based on reuse of old solutions. However, in many diagnosis problems it is difficult to compile a complete case description in advance, consequently the conventional *one-shot* case retrieval methodology will not work. In this paper we introduce a set of fault diagnosis problems that have this characteristic and we describe a model-based goal-driven system that produces focused questions that request extra information required for diagnosis. The central contribution in this paper is a description of a CBR system that also has this characteristic of producing focused questions in diagnosis. We describe the information theoretic mechanism that allows the CBR system to do this and we present an evaluation of the CBR system and a comparison of the two systems.

Introduction

The major attraction of CBR is its cognitive plausibility. It is clear that much of human expert competence is based on the reuse of past solutions in solving new problems. However, one restriction on the dominant CBR methodology is that it tends to be *one-shot*, requiring that the target case description be available in advance and that the problem can be solved with a small number of retrievals. This methodology can be problematic for some diagnosis problems and in this paper we describe one such situation. The problem in the situation that we describe is that all information is not available in advance and there is a cost associated with getting information. Consequently, it is important that the amount of information requested is minimised. What is needed is a CBR methodology that is incremental[1], one that can indicate what extra information is needed during the diagnostic process. It is interesting to note that backward-chaining systems have precisely this advantage in fault-diagnosis; the goal-directed reasoning asks the user only for information that contributes to the hypothesis being examined.

In this paper we consider the problem of using CBR in electronic fault diagnosis and discuss the re-engineering of an existing model-based, goal-directed diagnosis system as a case-based system. Fault diagnosis seems a good candidate for CBR because it is clear that much of human expertise in fault diagnosis is experience based. Further, fault diagnosis seems naturally case-based with each diagnostic episode constituting a case. The problem that we have encountered is centred on the cost of gathering a useful case description. In our problem domain there can be up to a hundred symptoms that potentially impact on the diagnosis and there is a cost associated with gathering most of these. For this reason the naïve solution of collecting these readings to make a case is not practicable. This contrasts sharply with the solution implemented in the existing model-based system (called NODAL) (Cunningham & Brady 1987), (Cunningham 1988). This system uses goal-directed reasoning in the diagnosis and this has the advantage of only querying the user for symptoms that will contribute to the diagnosis. This *parsimony dividend* of backward-chaining systems has been recognised since the early days of MYCIN and, in this paper, we will examine modifications to the CBR idea that can operate with the same, or even less, information.

First we shall introduce the diagnostic task, describe the existing model-based system and explain how the goal-directed reasoning dictates the number of symptoms requested from the user. This is covered in the next section. Next we examine other case-based diagnosis and explanation systems and consider how they tackle or avoid our problem. In the final section we describe our approach to incremental CBR for this particular diagnosis problem and compare it with the existing model-based system (MBS).

Goal-driven diagnosis

It has long been recognised that goal-directed reasoning, or backward rule-chaining, has specific advantages in

[1] We believe that there are two important ways that a CBR system can be incremental. CBR can be incremental in that it builds its solutions using components taken from several cases (Smyth & Cunningham 1992), (Redmond 1990). Alternatively it can be incremental in the sense we me here; that is that the target specification is composed during the CBR process.

diagnosis. In circumstances where there is a cost associated with determining the various symptoms associated with the fault or illness the goal-driven reasoning can focus the user interaction and only request input that contributes to a diagnosis. This advantage is not restricted to simple rule-based systems; model-based systems can adopt a goal-driven reasoning strategy and enjoy the same advantage. NODAL is such a system for electronic fault diagnosis.

Fault diagnosis in switching mode power supplies

NODAL is a model-based system for fault diagnosis of switching mode power supplies (SMPS). The system is implemented in KEE a hybrid expert systems development environment. The original motivation for its development was to produce a generic diagnostic system for a class of electrical devices. NODAL has a generic reasoning mechanism and can be set up to work for a particular power supply by encoding the model of that power-supply in the system. The block diagram of one of the power-supplies used in testing the system is shown in Figure 1.

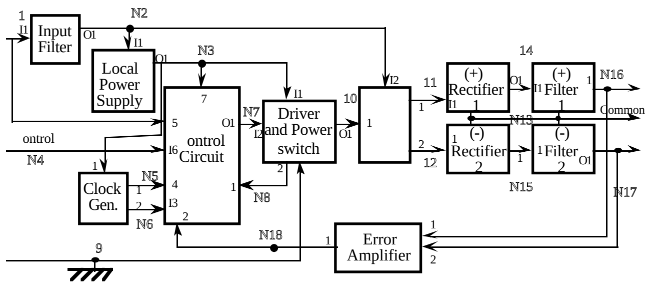


Figure 1. The block diagram of the 24V/12V power-supply.

The models in NODAL have a hierarchical structure. The top-level represents the blocks in the block diagram as frames, the main information on the frames being interconnection information and also some information about the characteristics of the blocks. These blocks are interconnected by nodes and these nodes themselves are represented as frames. These node frames carry information used during the diagnosis. For more complex power-supplies these blocks were further divided into sub-modules.

The detailed level of representation corresponds to detailed information available in schematics of the SMPS circuit. An example of the detail of the Local Power Supply module of the 24V/12V unit is shown in Figure 2 (a). The components are represented as frames that carry interconnection information and details of the characteristics of the components. Again, the interconnecting nodes are also represented as frames. The frame for the Q1 transistor is shown in Figure 2 (b).

NODAL was designed for use in a repair shop so the assumption that the circuit under examination has worked at some stage reduces the number of fault categories to be considered. Component failure accounts for over 95% of faults on SMPS that have failed in operation so NODAL is designed to detect these. Fault diagnosis of these SMPS involves locating the faulty module and finding the faulty component in that module.

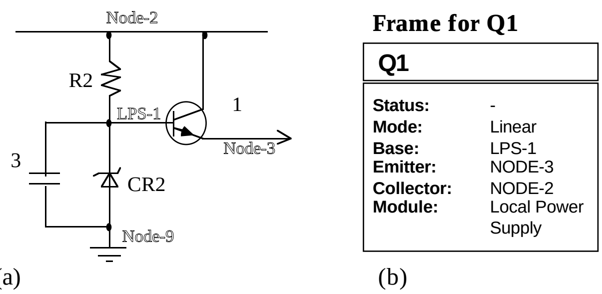


Figure 2. The detail of the Local Power Supply module in the 24V/12 circuit and the frame for the Q1 transistor in that circuit.

Since NODAL is designed to operate in a repair shop the first input in the diagnosis is the results from the test equipment on which it was confirmed that the unit was faulty. This input is shown in Figure 3. These function tests are performed on the unit as a 'black box', and measure outputs associated with test inputs. These tests will number between twenty and forty depending on the complexity of the circuit. However, because the internals of the unit are not being examined, the amount of diagnostic information that they carry is limited. The test results are processed by the Function Test Rules (a shallow reasoning component in NODAL) and a set of candidate faulty modules is produced.

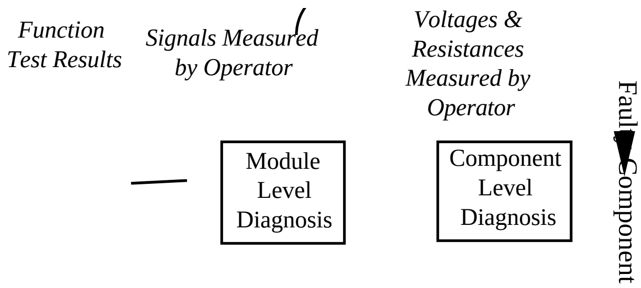


Figure 3. Data input in NODAL and the related reasoning processes.

In order to further isolate the fault it is necessary to perform some internal measurements on the unit. These measurements are taken at the nodes mentioned already. Measurements may involve estimating the goodness of a signal, or measuring voltages and resistances. This information is stored in the frames during the diagnosis. A typical circuit will have about 20 nodes at the module level and approaching 100 nodes altogether. Consequently there is a large number of measurements that can be taken during the diagnosis. The advantage of the goal-driven diagnosis is that it requests only measurements that contribute to its current hypothesis. In a typical session only about 20% of measurements are requested. This process will now be described in more detail.

Goal driven reasoning in NODAL

The output of the shallow reasoning process in NODAL is a candidate set of faulty modules. The model-based reasoning takes over at this stage with the objective of detecting the faulty module and the faulty component within that module. The behavioural component of the model is expressed as rules and the goal driven reasoning involves backward chaining through these rules applied to the model of the unit under test. A typical rule from the module level reasoning is as follows:-

```
IF
     (MODULE? HAS A BAD SIGNAL ON OUT-1 OR
      MODULE? HAS A BAD SIGNAL ON OUT-2)
 AND MODULE? HAS A GOOD SIGNAL ON IN-1
 AND MODULE? HAS A GOOD SIGNAL ON IN-2
THEN THE STATUS OF MODULE? IS FAULTY
```

The rule is expressed in the TellAndAsk knowledge-base query language of KEE as follows:-

```
(T-RULE-2-2
(IF ((OWN.VALUE TYPE ?MODULE TWO-TWO) AND
     (OWN.VALUE SIGNAL(GET.VALUE ?MODULE  OUT-1)?OS1) AND
     (OWN.VALUE SIGNAL(GET.VALUE ?MODULE  OUT-2)?OS2) AND
     ((EQUAL ?OS1 BAD) OR (EQUAL ?OS2 BAD)) AND
     (OWN.VALUE SIGNAL(GET.VALUE  ?MODULE IN-1) ?IS1) AND
     (EQUAL ?IS1 GOOD) AND
     (OWN.VALUE SIGNAL(GET.VALUE ?MODULE IN-2) ?IS2) AND
     (EQUAL ?IS2 GOOD))
THEN (A STATUS OF ?MODULE IS FAULTY)))
```

A typical rule from the component level reasoning is:-

```
(NPN.B.E.DOD.RULE
 (IF
  ((FIND (IN.CLASS ?NPN STD-NPN) DONT.ASK) AND
   (LISP (NOT (DIODE-BETWEEN-NODES-P 'B-E-DIODE
                        (UNIT.NAME (GET.VALUE ?NPN 'BASE))
                         (UNIT.NAME (GET.VALUE ?NPN 'EMITTER))))))
 THEN (OWN.VALUE STATUS ?NPN FAULTY)))
```

This rule checks to see if there is the correct diode voltage drop between the base and emitter of a transistor. Using the model of the circuit, the goal-directed reasoning uses these heuristics to determine first the faulty module, then the faulty component.

An example of a dialogue with NODAL will illustrate how this works in practice. The fault being analysed arises from removing the zener diode (CR2) in the Local Power Supply Module of the 24V/12V SMPS. This simulates that diode blowing open-circuit. At the point when the dialogue commences the system has already completed the shallow reasoning based on the function test rules and has established a candidate set of faulty modules. It then proceeds to try and prove one of these to be faulty (underlined text is input by the user):-

Setup for Test Vector 1

What is the SIGNAL of NODE-2? Good
What is the SIGNAL of NODE-3? Bad

It looks like the fault is in the LOCAL-POWER-SUPPLY
Switching to considering the circuit at a component level...

What is the VOLTAGE of NODE-2? 23.4
What is the VOLTAGE of LPS-1? 18.79
What is the VOLTAGE of NODE-3? 18.12
What is the VOLTAGE of NODE-9? 0

It looks like the fault is in CR2

Even by NODAL's standards this dialogue is particularly short as the first module to be examined proves to be the faulty one (for more example dialogues see (Cunningham 1988)). Nevertheless it serves to illustrate how the goal-directed reasoning focuses the requesting of measurements from the operator. This brings our requirements on a CBR system for the same task into focus. Since there is a cost associated with determining the inputs to the diagnosis it is important that the CBR system should not need to have all the inputs in advance. It should be able to direct the operator on what measurements are important just as the goal-driven system does.

Existing diagnostic CBR systems

Despite this difficulty that we identify in the use of CBR in diagnosis there has been considerable interesting and successful research in the area. Some systems that are worth highlighting are as follows:-

- The help desk application of Simoudis and Miller (Simoudis & Miller 1991)
- CASEY: a system for managing the diagnosis of cardiac disease (Koton 1988)
- PROTOS: a system for assisting in the diagnosis of audiology disorders. (Porter et al 1990)
- The GE help desk application of Kriegsmann & Barletta (Kriegsmann & Barletta 1993)

In the remainder of this section we will examine these systems focusing on the aspects that are relevant to our problem.

Simoudis & Miller's Help Desk System.

This help desk application is designed to assist product support engineers in diagnosing customer problems with the VMS operating system. The system uses a two-phase case retrieval process of surface feature-based retrieval followed by model-based validation. The surface features are inexpensive to obtain and include hardware and software system descriptions and data obtained from the core dump associated with the system failure. The first phase in the retrieval process returns all cases from the case-base that are similar according to these surface criteria. The model-based validation uses validation information from these cases to direct further inquiry into the target case. This validation process will determine whether the new problem is in fact similar to any of those retrieved from the case-base.

It appears that this strategy would not work in our situation because there is not an obvious set of surface features that would significantly reduce the case-base.

CASEY

In CASEY, Koton integrates case-based and model-based reasoning techniques to produce an expert system for managing the diagnosis of cardiac disease. Each case represents a single patient diagnosis and is composed of both descriptive features and solution features. The descriptive features correspond to the patient's observed symptoms and test results, whereas the solution features describe the diagnosis and suggested therapy.

Given some new patient description, CASEY will attempt to identify the causality underlying the observed disorder and propose therapy (the diagnosis solution) using a three-step process. First, CASEY will search for a case that is similar to the current patient diagnosis context. Second, it evaluates the significance of any differences between the target and base case. This evaluation is carried out using a set of *evidence principles* and will reject a match if these principles suggest that certain features of the base case cannot be applied to the target. Third, if none of these differences rule out the applicability of the case, then it adapts the solution of the base case to fit the target context. CASEY adapts a retrieved case using a set of *causal repair strategies* to modify the nodes or links in the casual explanation of the base case.

PROTOS

PROTOS is a classification and learning system for operation in clinical audiology (Porter et al. '90). PROTOS implements a type of prototype based classification whereby surfaces features of the case are used to identify a set of candidate categories. The system uses prototypical examples of these categories in the reminding process. The highest ranking of these remindings is selected and the system initiates a dialogue with a human expert to determine whether it is a correct diagnosis. If the diagnosis is incorrect the system attempts to adjust its links between case features and categories in order to produce a correct diagnosis. The focus of the work on PROTOS has been on concept acquisition in a weak theory domain rather than on the diagnosis itself. So, while PROTOS, does conduct a dialogue with its operator, the motivation is different to what we have in mind. It is concerned with knowledge acquisition while we want the dialogue to further the diagnosis itself.

The GE Help Desk System

Kriegsmann & Barletta view CBR as having a number of advantages over more conventional text-based or rule-based help-desk systems (Kriegsmann & Barletta 1993). Their prototype system aimed at investigating the role of CBR in providing help-desk functionality across a range of computer hardware, software, and networking problems. The problem being addressed in this system is similar to ours in that there is a cost associated with determining the case features.

The system offers the operator a template on which the target specification is to be entered. The operator is free to leave blanks in this template as the retrieval mechanism can operate with incomplete information. The system uses an inductively built decision tree to identify a group of candidate cases with contextually similar features to the target problem. For each candidate a score is computed using nearest-neighbour methods. The score reflects the similarity of the candidate to the target. If the initial target specification is too general or sparse to result in the retrieval of a single best case or even a small subset of candidates the system allows the operator to specify additional information to further focus the retrieval process. The determination of which additional features to specify is left to the operator.

Incremental CBR in $NODAL_{CBR}$

Our objective in this work is to explain how the desirable, goal-directed behaviour of the NODAL model-based diagnosis system might be transferred to an equivalent CBR system, which we call $NODAL_{CBR}$. What we have developed is a CBR system that can begin operation without a complete target case description and can generate queries that will help it to home in on a solution. We have implemented an information theoretic mechanism that identifies the maximally discriminant diagnostic features of the retrieved cases. Thus, rather that expecting the operator to chose the next test to apply, the system will propose one automatically. In addition the test chosen should maximise the amount of new information gained, and hence ensure the most rapid route to the desired diagnosis.

Case Representation & Retrieval

Our first objective has been to replace the first two stages of diagnosis in the old system (see Figure 3) with one CBR stage. Consequently the cases features are the information used in the diagnosis at this point, that is the function test results and the module level signal information. The case also contains a case name and the identity of the faulty module associated with these symptoms (the solution). A typical case structure is shown in Figure 4. It is in the nature of the diagnosis task that the case features are sparse. There is often a small number of results to the function tests because once a unit fails one of the early tests it is not possible to proceed with subsequent tests. In addition, a large portion of the signal features are not examined as the diagnosis quickly concentrates in a specific part of the circuit.

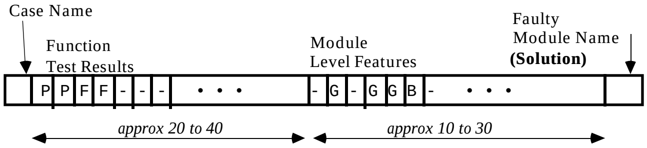


Figure 4. A typical case structure in $NODAL_{CBR}$.

We can now consider what happens during a typical run of the CBR system. In this first scenario the unit fails one function test, 3-POS-OUTPUT-VOLTAGE, after which it is not possible to continue with further function tests. Cases matching these function test results are returned from the case-base. In this example these are cases with the signal features shown in Table 1. At this stage the unit under test has not been probed for this signal information so we want the system to ask some discriminating questions - this was the particular strength of the old NODAL system.

Table 1. This chart shows the module level portion of several cases returned during the diagnosis.

Nodes G: good B: bad

Case Name	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**
Input-Filter-1	B																
Local-PS-1	G	B															
Driver&PS-1	G	G				G	G		B								
Control-Cct-1	G	G		G	G	B	G		G								G
Clock-Gen-1	G	G		B	B	G	G		G								
Clock-Gen-2	G	G		B	G	G	G		G								
Clock-Gen-3	G	G		G	B	G	G		G								
Driver&PS-2	G	G				G	B		G								
Driver&PS-3	G	G				G	B		B								
Xfmr-1	G	G		G	G	G	G		G	B	B		B	B			
Xfmr-2	G	G		G	G	G	G		G	B	G		B	G			
Xfmr-3	G	G		G	G	G	G		G		B		G	B			
Output-Rect-1	G	G		G	G	G	G		G	G			B				
Output-Rect-2	G	G		G	G	G	G		G		G		G	B			

The solution we have adopted is to use information theoretic criteria similar to those used in ID3 (Quinlan 1986) to determine the most *discriminating* feature to be measured at each stage in the narrowing down of this subset of cases.

Selecting Discriminating Features

This selection of discriminating features amounts to building a decision tree that will have leaf nodes corresponding to the different diagnoses **D** and the set of cases **C** will be located, or classified, on these nodes. It is important that the tree is in some sense minimal so the choice of which feature to test at any level of the tree is critical. In ID3 this is done by selecting features based on their information content or discriminatory power (Quinlan 1986). The process used in $\mathrm{NODAL}_{\mathrm{CBR}}$ is similar to that in ID3 except that the semantics of the branching in the decision tree is slightly different because of the large number of unknowns in the case features. A brief explanation of how the discrimination works is as follows:-

$\mathbf{D}=\{D_1,...,D_d\}$ the set of possible classes or diagnoses (7 in Table 1)

$\mathbf{C}=\{C_1,...,C_c\}$ the set of cases to classify (14 in Table 1)

$\mathbf{F}=\{F_1,...,F_f\}$ the set of descriptive features that will form the nodes of the decision tree. (17 in Table 1)

We can view the decision tree as an information source producing one of *d* messages from the set **D**. Let $|D_i|$ represent the number of cases with diagnosis D_i. Then the expected information needed to generate the appropriate message, for some case, using the tree is:-

$$I\left(|D_1|,\ldots,|D_d|\right) = -\sum_{i=1}^{d}\left(\frac{|D_i|}{|D_1|+\ldots+|D_d|}\bullet \log_2\left[\frac{|D_i|}{|D_1|+\ldots+|D_d|}\right]\right) \quad (1)$$

Consider the root decision node of the tree (see Figure 5). Assume this node tests the feature $F \in \mathbf{F}$ and this feature has possible values $\mathbf{V}=\{V_1,...,V_n\}$. Then **V** partitions **C** into n groups of cases, $G_1,...,G_n$; where G_i contains those cases that have value V_i for feature F.

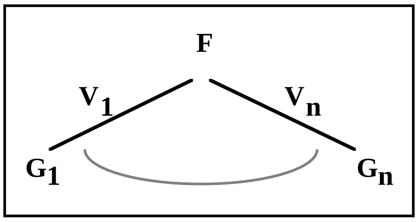


Figure 5. The root classification of the cases in **C**.

Let G_i contain $|D^i_j|$ cases with diagnosis D_j, that is $|D^i_j|$ instances of class D_j. Then the expected information required for the sub-tree of G_i is $I(|D^i_1|,...,|D^i_d|)$. We can obtain the expected information for the tree with F as root by computing the weighted average over all value branches of F as follows:-

$$E(F) = \sum_{i=1}^{n}\left(\frac{|D^i_1|+\ldots+|D^i_d|}{|D_1|+\ldots+|D_d|}\right)\bullet I\left(|D^i_1|,\ldots,|D^i_d|\right) \quad (2)$$

The weight of the *i*th branch is the proportion of cases in C that belong to G_i. The information gained from using F, or the *discriminatory power* of F, is:-

$$DP(F) = I\left(|D_1|,\ldots,|D_d|\right) - E(F) \quad (3)$$

So, at each stage in the reduction of the set of cases, the most discriminating feature is selected using this criterion. The user is requested to determine the value of this feature for the target case. The retrieved cases that cannot match on this feature are removed from the retrieved set. This process is repeated until the set reduces to one diagnosis or the target case proves to be dissimilar to all the retrieved cases. It is important to emphasise that a discrimination tree for the set of cases is not being produced, instead local discriminations are determined at run-time. This technique has proved remarkably successful; indeed it results in less questions being asked of the user that was the case with the original NODAL system.

Evaluation

In this section we will present a comparison of the MBR and CBR systems and evaluate the strengths of the CBR system. Returning to the example introduced earlier, we can continue with the dialogue generated by $\text{NODAL}_{\text{CBR}}$:-

Selecting function test failure cases : Retrieved 14
> (INPUT-FILTER-1 LOCAL-PS-1 DRIVER&PS-1 CONTROL-CCT-1 CLOCK-GEN-1 CLOCK-GEN-2 CLOCK-GEN-3 DRIVER&PS-2 DRIVER&PS-3 XFMR-1 XFMR-2 XFMR-3 OUTPUT-RECT-1 OUTPUT-RECT-1)

What is the value for N2 ? G
What is the value for N3 ? G
What is the value for N10 ? G
What is the value for N7 ? G
What is the value for N8 ? G
What is the value for N5 ? B
O.K.....

Selecting candidate modules : Retrieved 2
(CLOCK-GEN-1 CLOCK-GEN-2)

Validation:
The fault is in CLOCK-GENERATOR if
N6 is B or N6 is G

The 14 cases shown in Table 1 are returned and N2 is found to be the first most discriminating criteria. After 6 questions the faulty module is discovered. This compares with 7 questions in the model based reasoning of old NODAL:-

Setup for Test Vector 1

What is the SIGNAL of NODE-2? Good
What is the SIGNAL of NODE-3? Good
What is the SIGNAL of NODE-10? Good
What is the SIGNAL of NODE-8? Good
What is the SIGNAL of NODE-7? Good
What is the SIGNAL of NODE-5? Bad
What is the SIGNAL of NODE-6? Bad

It looks like the fault is in the CLOCK-GENERATOR
Switching to considering the circuit at a component level...

The CBR system performs better than the MBR system because it only requires enough information to uniquely classify the case in the case-base. In comparison, the MBR system requires enough information to verify a hypothesis in its knowledge base. The CBR system has a further validation phase where it informs the user of remaining information that will confirm that the cases match. The importance of this validation depends on the coverage of the case-base. It is not required when coverage is good.

When we compared the CBR system with the old system on a sample set of faults on the DC/DC circuit we found that is required only 83% of the user input that the MBR system did. This information is plotted on a case by case basis in Figure 6. Two other smaller evaluations are shown on Figure 7 (a) and (b). In these situations the number of questions is reduced to 35% and 33% respectively. From a situation where our initial aspiration was to produce a CBR system that would have the informational parsimony of a goal-driven system we find that the CBR system is *better* than the old NODAL system.

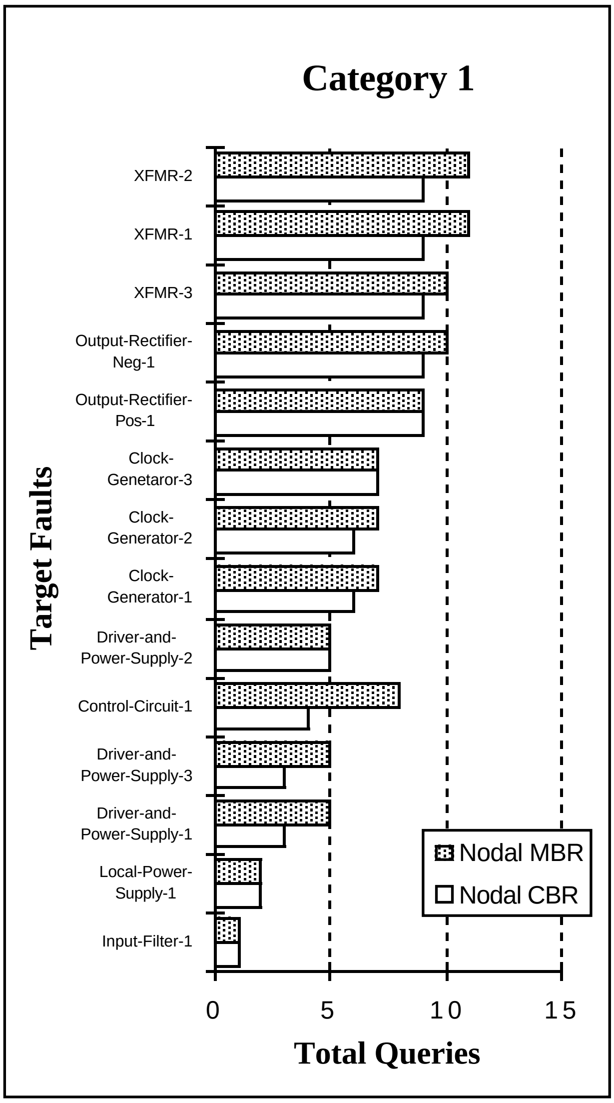
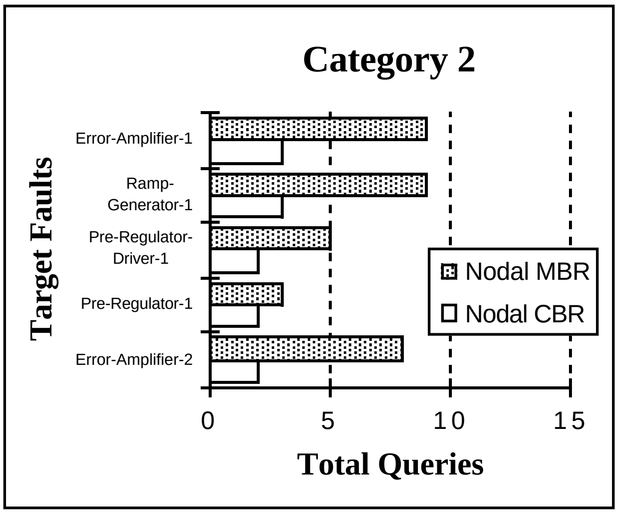
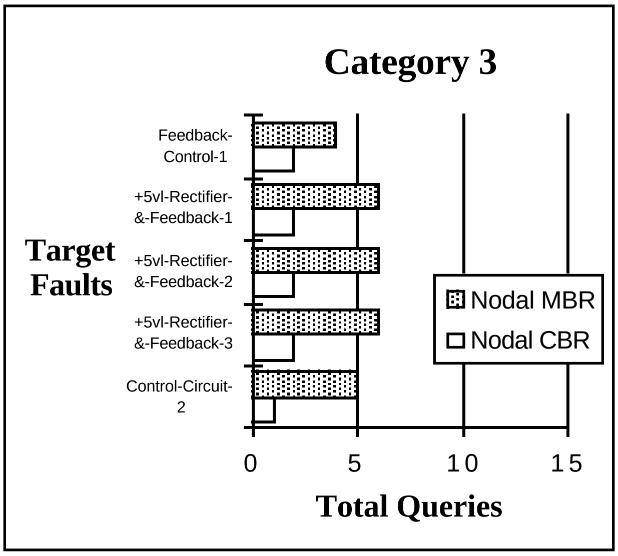


Figure 6. A comparison of the numbers of questions generated by the MBR and CBR systems in fault diagnosis.



(a)



(b)

Figure 7. Further comparisons of the numbers of questions generated by the MBR and CBR systems.

Conclusion

We have described a class of diagnosis systems where there is a potentially large amount of information that may be used in the diagnosis process. There is a cost associated with obtaining this information and a successful diagnosis can be made using a portion of this information. It is part of the conventional wisdom in AI that backward chaining systems are good at these types of problem because the goal directed reasoning produces focused questions, requesting only information that is relevant to the hypothesis being pursued. In the early part of this paper we have described NODAL, a model-based system of this type.

The main contribution of this paper has been the description of $NODAL_{CBR}$ a case-based reimplementation of a portion of the NODAL system. The important component in this re-implementation is the information theoretic criteria used to determine the next question to be asked of the operator. An evaluation of the two systems has shown that the CBR implementation can operate with *less* information than the model-based system. This is because, for our purposes, the criteria of minimum information is more parsimonious than the knowledge based heuristics in the MBR system.

We believe that many diagnosis problems have these characteristics and this technique can improve the usefulness of many case-based diagnostic systems.

References

Cunningham P.; Brady M. 1987. Qualitative reasoning in electronic fault diagnosis, in Proceedings of Tenth International Joint Conference on Artificial Intelligence, 443-445. ed. J. McDermott, Morgan Kaufmann, Milan Italy.

Cunningham P. 1988. Knowledge Representation in Electronic Fault Diagnosis. Ph. D. Thesis, Department of Computer Science, Dublin University, Trinity College, Ireland.

Koton P. 1988. Reasoning about Evidence in Causal Explanations, in Proceedings of Workshop on case-based reasoning (DARPA), 260-269, Clearwater, Florida, Morgan Kaufmann.

Kriegsmann M.; Barletta R. 1993. Building A Case-Based Help Desk Application. *IEEE Expert,* 8(6): 18-26.

Porter B.W.; Bareiss R.; Holte R.C. 1990. Concept Learning and Heuristic Classification in a Weak-Theory Domain, *Artificial Intelligence*, 45: 229-263.

Quinlan J.R. 1986. Induction of Decision Trees, *Machine Learning*, 1(1): 81-106.

Redmond M. A. 1990. Distributed cases for case-based reasoning: Facilitating use of multiple cases. in Proceedings of AAAI-90, AAAI Press/MIT Press, 304-309.

Simoudis E.; Miller J.S. 1991. The Application of CBR to Help Desk Applications. In Proceedings of the Case-Based Reasoning Workshop, 25-36. Washington D.C., U.S.A., Morgan Kaufmann.

Smyth B.; Cunningham P. 1992. Déjà Vu: A Hierarchical Case-Based Reasoning System for Software Design. In Proceedings of European Conference on Artificial Intelligence, 587-589, ed. Bernd Neumann, John Wiley, , Vienna Austria.