To Appear in Journal of Knowledge Based Systems

An Incremental Retrieval Mechanism for Case-Based Electronic Fault Diagnosis

Pádraig Cunningham
Department of Computer Science
Trinity College Dublin
College Green
Dublin 2
Ireland

Barry Smyth
Department of Computer Science
University College Dublin
Belfield
Dublin 4
Ireland

Andrea Bonzano
Department of Computer Science
Trinity College Dublin
College Green
Dublin 2
Ireland

Abstract

One problem with using CBR for diagnosis is that a full case description may not be available at the beginning of the diagnosis. The standard CBR methodology requires a detailed case description in order to perform case retrieval and this is often not practical in diagnosis. We describe two fault diagnosis tasks where many features may make up a case description but only a few features are required in an individual diagnosis. We evaluate an incremental CBR mechanism that can initiate case retrieval with a skeletal case description and will elicit extra discriminating information during the diagnostic process.

Keywords: Case-based reasoning, case retrieval, electronic fault diagnosis.

1 Introduction

The fact that human problem solving competence is often based on reasoning from examples supports the use of case-based reasoning (CBR) for developing knowledge-based systems. In particular, good performance in both technical and medical diagnosis is often dependent on remembering similar cases encountered in the past. However, an analysis of the use of CBR in diagnosis illustrates that the structure of conventional CBR is very rigid when compared with the flexibility of reuse that humans exhibit in problem solving. A particular problem in using CBR for many diagnosis tasks is that a complete case description is needed in advance of case-retrieval. This is often not practical as the case can be characterised by a large set of symptoms or test results, not all of which are required in order to make a diagnosis. Moreover, many of these features will be expensive to determine so it is desirable that the number required to deliver a good diagnosis should be minimised.

In this paper we describe an incremental case retrieval mechanism that can initiate case retrieval with a brief case description. This brief description is used to retrieve a matching subset of the case-base. This retrieved set is analysed to determine discriminating tests that the operator is asked to perform. This procedure offers an incremental case retrieval mechanism that retrieves good matches while requiring a minimal case description [6].

The mechanics of this procedure are described in detail in section 3. First we illustrate the motivation for this approach by describing two diagnosis tasks that require incremental case retrieval. These tasks involve troubleshooting a switching mode power supply and a microprocessor board. In section 4 we describe some experiments on troubleshooting these circuits that illustrate the effectiveness of incremental CBR.

2 CBR in Diagnosis: the need for I-CBR

It has long been recognised that knowledge-based solutions to diagnostic tasks are readily structured as goal directed reasoning systems. This approach has the important advantage that the goal-directed reasoning mechanism can drive the generation of queries for the user. In this way the user is only asked to provide information or perform tests that are of use to prove or disprove a particular hypothesis [1][2]. The user will only need to provide a subset of all the information that might possibly be relevant to a particular fault or diagnosis. This is important because acquiring some information may be expensive; whether it be taking measurements on a circuit, or performing tests on a patient.

It will be evident from the examples presented later in this section that diagnostic tasks require information that is readily available; for instance the results of some function test results that will be performed automatically to determine if the circuit is faulty. In addition to this the diagnosis will also require information that is expensive to obtain; for example, diagnostic tests or detailed measurements in the circuit.

All this causes problems for the conventional model of CBR because it requires a full description of the target problem to initiate case retrieval. Our model of incremental CBR (I-CBR) addresses this issue by separating information into *free* and *expensive* features. Case retrieval can be initiated using just the free features with the user then being asked for selected expensive features to narrow down the set of retrieved cases. Before elaborating on how this I-

CBR mechanism works we will describe some diagnosis and classification problems that fit this structure.

2.1 Microprocessor Example

If we look at the problem of troubleshooting a microprocessor board we will see that the features required for diagnosis do fit this *free/expensive* structure. There are many features or tests that may be relevant to a particular diagnosis but only a subset of these are required to establish a diagnosis. A block diagram for a particular microprocessor board is shown in Figure 1.

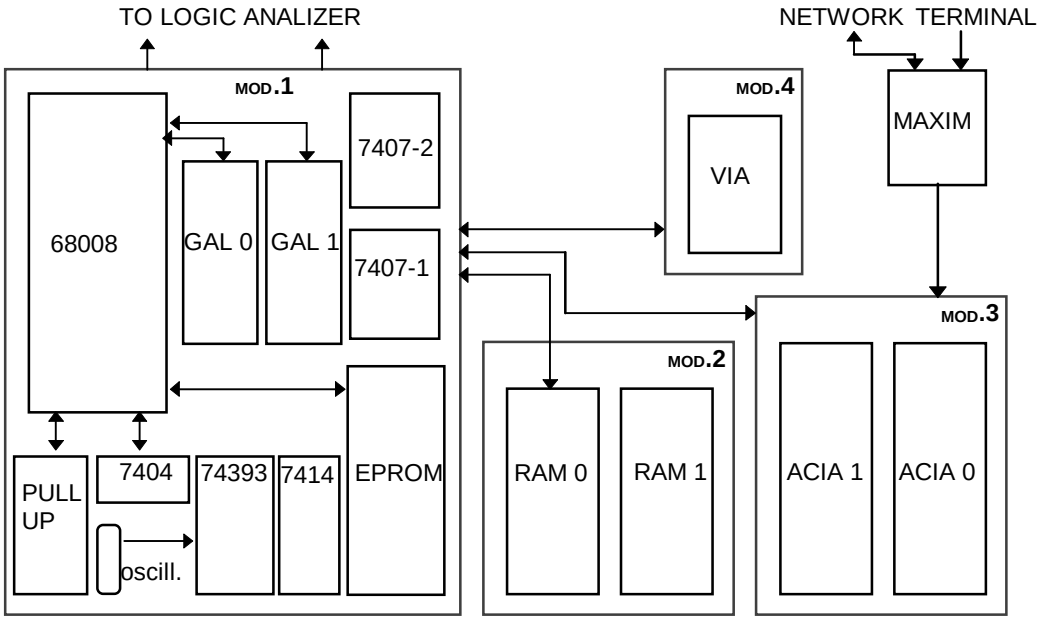

Figure 1. Block diagram for a Motorola 68008 based microprocessor board.

We are not concerned with the detailed operation of this system, instead we can concentrate on troubleshooting issues. The fifteen chips of the circuit can be divided into four main modules:-

Mod 1: This is the kernel microprocessor module, containing the processor, the EPROM, some clock circuitry and some glue logic. This is the most complex module in the system.

Mod 2: The memory; two 2Kbyte RAMs.

Mod 3: The Serial I/O module; two ACIAs.

Mod 4: The Parallel I/O Module.

The operating system (called the monitor program) is burnt into the EPROM and allows loading of programs into RAM, examining and changing individual memory locations and some other useful functions. The monitor program is automatically executed when the board is powered up but it is preceded by a simple diagnostic program that tries to write and read data from RAM and from the I/O modules. Many of the possible faults will cause the system to fail this boot process. However, the system may manage to boot successfully with some faults, such as faults in the I/O modules.

Diagnostic information is obtained either with a logic analyser or with an oscilloscope. With the logic analyser it is possible to examine step by step all the instructions of the diagnostic program. With some faults the diagnostic program fails to start (catastrophic error); with other faults only a few instructions reveal problems (fault located in some peripherals). It is possible to use the oscilloscope to read signals at various points in the circuit, particularly at the output pins of different chips in the circuit. Evidently the results from the diagnostic program can be considered *free* features. Information gathered using the oscilloscope is *expensive* and the amount required should be minimised.

2.2 Fault diagnosis in switching mode power supplies

A model-based fault diagnosis system already exists for the other diagnosis problem we wish to consider. This system, called NODAL, is a system for fault diagnosis of switching mode power supplies (SMPS) [2]. It 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 that NODAL can troubleshoot is shown in Figure 2.

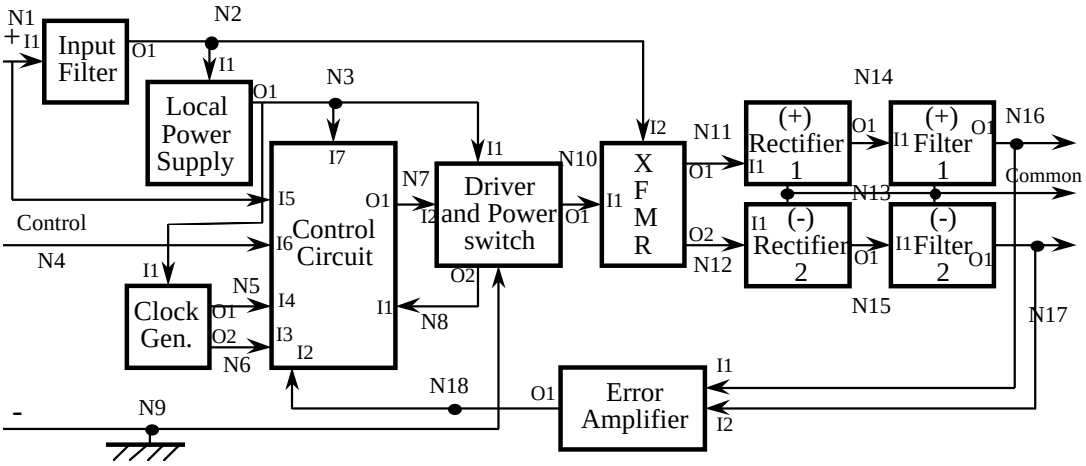

Figure 2. The block diagram of the 24V/12V power-supply.

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 3 (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 3 (b).

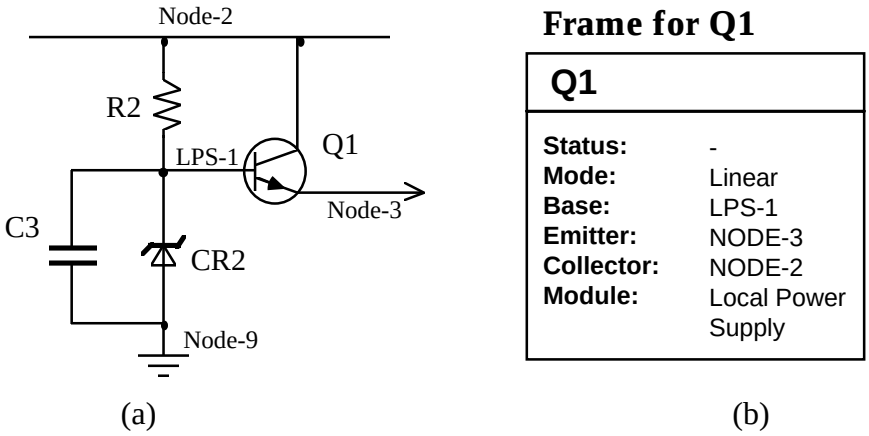

Figure 3. The detail of the Local Power Supply module in the 24V/12 circuit and the frame for the Q1 transistor in that circuit.

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.

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 4. 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.

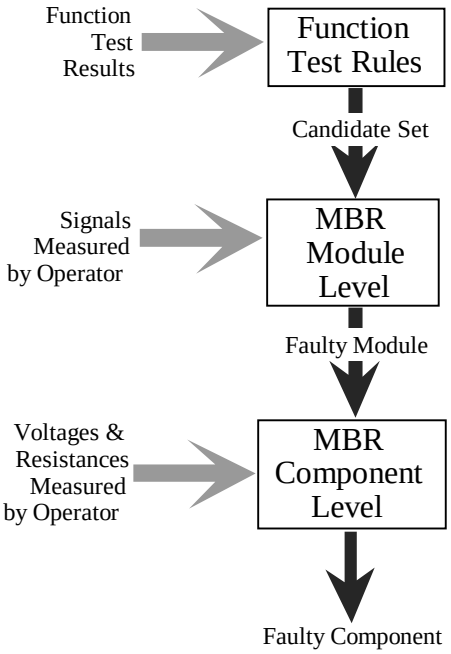

Figure 4. Data input in NODAL and the related reasoning processes.

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.

In a CBR implementation of NODAL the Function Test results are *free* features, the other measurements performed by the operator are *expensive* features. The goal directed reasoning in the model-based implementation has the advantage that it reduces the amount of expensive features required.

It is worth mentioning that our motivation in developing a CBR system to solve the SMPS problem was to see if it would be easier to develop than the MBR system. It is to be expected that the knowledge engineering requirement in developing a CBR system should be less than that for an MBR system. The challenge for I-CBR was that it should be as economical in requiring information as the MBR system. We did find that the CBR system was much quicker to develop than the original NODAL. However, this is hardly conclusive as the experience might have been very different if the systems were developed in the reverse order.

3 Incremental CBR

These two examples illustrate diagnosis problems where some features are freely available and others are expensive to obtain. This is also a characteristic of help-desk problems as reported by Kriegsmann and Barletta [3]. They describe a CBR system for providing help-desk functionality across a range of computer hardware, software, and networking problems. The problem being addressed in that system is similar to ours in that there is a cost associated with determining the case features. Their 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.

This two stage retrieval process is similar to I-CBR with the key difference being that, with I-CBR, the system specifies the additional features to be provided. I-CBR works as follows:-

Step 0: Generate a **candidate set** of cases based on initial features available. The operator provides the system with the values of the free features for the target case. The **candidate set** is made up of cases that match on these features.

Step 1: Select the most discriminating expensive feature in the candidate set.

Step 2: Query the operator for the value of that feature in the target case.

Step 3: Narrow down the candidate set based on this information.
(i.e. Eliminate cases that *cannot* match this feature, cases for which the value of this feature is unknown are allowed to remain in the candidate set.)

Step 4: Repeat from 1 until a unique diagnosis remains.

The main objective is to provide added information about the target case that will allow the system to reduce the candidate set. The key issue here is the process in Step 1 where the system determines the test to perform next. One possibility is that this test selection process itself should be case-based. Alternatively the operator may be given the discretion to select the information to provide next [3]. In I-CBR the candidate set is analysed and the most discriminating feature is selected based on information theoretic criteria. This process is described in full in section 4.2 but first we discuss the generation of the initial candidate set.

3.1 Building the Initial Candidate Set

The first step in I-CBR is to build an initial candidate set of cases that match the free features in the target case. This could be done using a flat search of the case-base but since I-CBR is implemented on top of a frame system called KRELL a simple activation based mechanism is employed. This is more economical than flat search with retrieval time increasing less than linearly with the size of the case base.

This simple activation based mechanism depends on the fact that KRELL supports inverse links. This has the effect of making accessible from the target case cases that have matching features. An example of this is shown in the diagram in Figure 5.

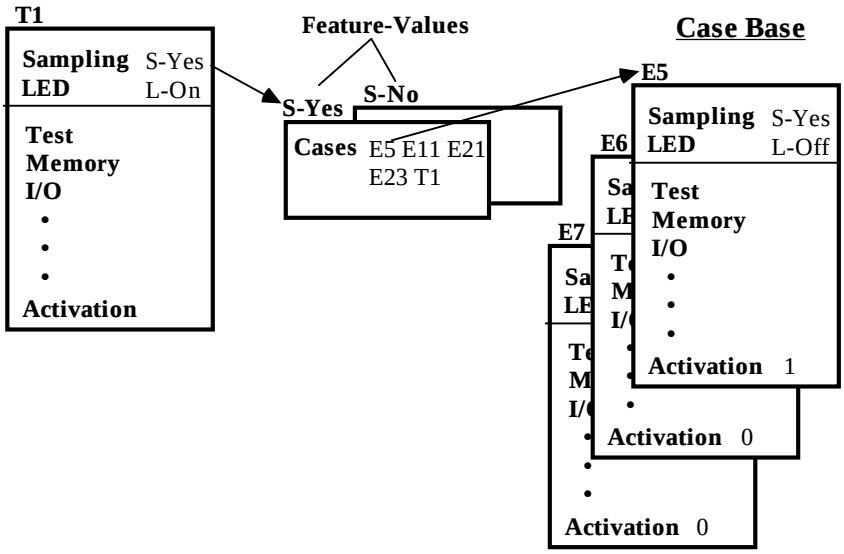

Figure 5. An example of the use of inverse links in selecting a candidate set.

In this example the target case is the case **T1** on the left of the diagram. The objective of the candidate selection process is to retrieve all cases that match on the two free feature values **S-Yes** and **L-On**. The inverse facility in KRELL is used to declare that the **Sampling** attribute has inverse **Cases**. This means that every time a case is given a particular value for **Sampling** the frame for that value has that case added to its **Cases** slot. This means that matching cases in the case-base can be accessed via the **Feature-Value** frames.

The spreading activation process takes each of the free feature values in the target case in turn and increments the activation of cases in the case-base that share that value. The names of

activated cases are remembered on an activation list. When the activation spreading is complete the cases on the activation list with maximum activation form the candidate set.

3.2 Selecting Discriminating Features

The diagram in Figure 6 shows one of the case structures from μI-CBR, the I-CBR system for troubleshooting the microprocessor board described in section 2.1. In this case there are two free features and the values for these in the target case would be available at the beginning of the diagnosis.

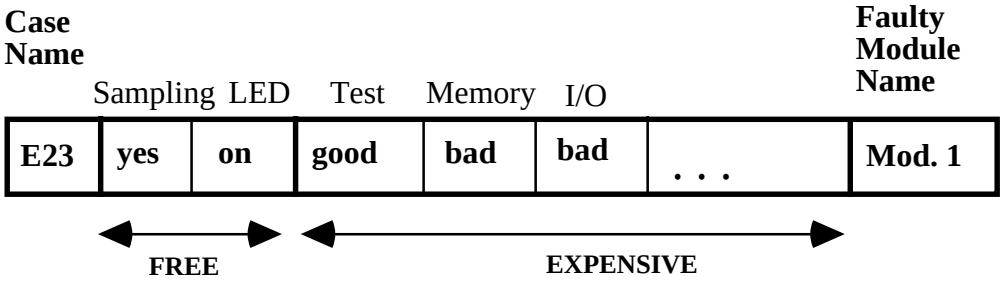

Figure 6. Case from microprocessor board diagnosis system.

The first step as described in section 3 is to select a candidate set of cases that match this basic description. In our system we only select cases that match these free features exactly. Presumably looser selection criteria might be appropriate in other situations. An example candidate set is shown in Table 1. This example shows nine cases in the candidate set. These cases have seven expensive features and describe six different faults.

Table 1. An example candidate set from the low level case-base of faults within Mod.1, the microprocessor module.

	FEATURES							
CASE Description	Output Enable	Add. Strobe	Gal1 Pin 20	FC0	P-U Pin 4	P-U Pin 13	cnter Pin 8	**Fault**
No Vcc on 68008	no	3V-4V	3V	3V	high	high	low	68008
No Vcc on GAL 0	sign.2	4V-5V	4V	4V	sign.1	sign.1	low	GAL 0
(D3) 68008 sa 0	sign.2	4V-5V	3V	0V-1V	high	high	low	68008
(D0) EPROM sa 0	noise	4V-5V	3V	4V	high	high	low	EPROM
(10) 74393 sa 1	no	4V-5V	3V	0V-1V	high	high	high	CLOCK
oscillator	no	low	3V	3V	low	high	4V	CLOCK
(A17)68008 sa 0	sign.2	4V-5V	3V	4V	high	high	low	68008
BERR=1 (GAL 1)	no	4V-5V	3V	0V-1V	low	high	low	GAL 1
(5) pull-up sa 0	noise	norm.	norm.	norm.	high	norm.	norm.	Pull-up

The objective now is to narrow down this set to a set describing a unique fault by asking the 'minimum' number of questions of the operator.

This is very similar to what happens in the induction of decision trees using ID3 [4] or in cost-sensitive classification [8][9]. The important difference is that I-CBR is a lazy learning technique and performs the analysis at run-time instead of in advance. In I-CBR a tree is not actually built, instead a single path is traced from the root to one of the leaf nodes.

The mechanism of selecting discriminating features is best explained in terms of 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

[4]. The process used in I-CBR is similar to that in ID3 except that the semantics of the branching in the decision tree is slightly different because of the possibility 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	(6 in Table 1)
$\mathbf{C}=\{C_1,...,C_c\}$	the set of cases to classify	(9 in Table 1)
$\mathbf{F}=\{F_1,...,F_f\}$	the set of expensive features, one of which is selected at each decision point	(7 in Table 1)

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

$$I\left(\frac{|D_1|}{|D_1|+...+|D_d|},...,\frac{|D_d|}{|D_1|+...+|D_d|}\right)=-\sum_{j=1}^{d}\left(\frac{|D_j|}{|D_1|+...+|D_d|}\bullet\log_2\left\lceil\frac{|D_j|}{|D_1|+...+|D_d|}\right\rceil\right) \qquad (1)$$

Consider the complete set of matching cases (see Figure 7). Assume we test 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.

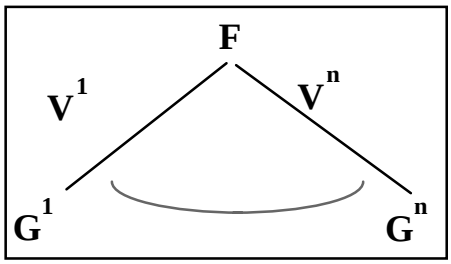

Figure 7. The root classification of the cases in **C**.

Let G^i contain $|D^i_j|$ cases with diagnosis D_j, that is $|D^i_j|$ instances of class D_j. The probability of a case belonging to G^i is (i.e. probability of a case having the ith. value for attribute F) :-

$$\frac{|D^i_1|+,...,+|D^i_d|}{|D_1|+...+|D_d|} \qquad (2)$$

So after testing F the remaining information associated with the subsets, $G^1,...,G^n$ is:-

$$\text{Remainder}(F)=\sum_{i=1}^{n}\left(\frac{|D^i_1|+,...,+|D^i_d|}{|D_1|+...+|D_d|}\right)\bullet I\left(\frac{|D^i_1|}{|D^i_1|+...+|D^i_d|},...,\frac{|D^i_d|}{|D^i_1|+...+|D^i_d|}\right) \qquad (3)$$

The weight of the ith. subset 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:-

$$DP(F)=I\left(\frac{|D_1|}{|D_1|+...+|D_d|},...,\frac{|D_d|}{|D_1|+...+|D_d|}\right)-\text{Remainder}(F) \qquad (4)$$

Thus the feature that leaves the smallest Remainder is the most discriminating.

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 cases in the candidate set 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 for retrieving good matches while requiring a minimum number of expensive feature values.

4 I-CBR in action

In evaluating this incremental CBR mechanism there were two important criteria; the accuracy of the retrieval mechanism and the amount of test data requested from the operator. These two issues are considered in detail in section 4.2 and 4.3. Before that we will describe the details of our CBR solution to troubleshooting the microprocessor board.

4.1 μI-CBR

The hierarchical modular structure of the microprocessor circuit is reflected in the design of the CBR system (see also [5][7]). There is a high level CBR sub-system that locates the fault to one of the modules described in section 2.1. The high-level case-base is composed of 53 cases, with 5 features each. Two of these are free features because they are easily acquired with the logic analyser and a LED that tells when the board is in the halt mode. The other three features require some effort to determine.

There is a lower level case-base for each of the modules in order to locate a fault to an individual component within each module. The most complex module is Mod. 1 (the Kernel). Each case in this low-level case-base is represented by ten features. Only one of these features comes from the logic analyser, the rest are taken with the oscilloscope.
A typical dialog with the system is as follows:

```
What is the value for I/O ?                    bad
What is the value for TEST ?                   bad

solution: the fault is in MODULE-1
cases retrieved: E7 E8 E11 E15 E16 E26 E27 E28 E36

Switching to working with the low-level case-base

What is the value for ADDRESS-STROBE ?         4V-5V
What is the value for OUTPUT-ENABLE ?          noise

solution: the fault is in EPROM
cases retrieved: I-E36
```

In this situation there are two free features; the status of the LED mentioned earlier, and a feature indicating whether or not the logic analyser is able to sample. These two features are used to narrow down the high-level case-base to the initial candidate set. Then the system analyses the cases in the candidate set to determine the most discriminating attribute, the value of which will be determined from the user. This process is repeated until either, a unique solution exists or no solution remains. In this situation a unique solution is found after two questions. The system then switches to working with the low-level case-base for Module-1 and the faulty component is located after two more questions.

4.2 Evaluation of μI-CBR

The evaluation of this system focused on the performance of the high-level CBR sub-system. The system was tested in two modes. The first test mode involved using one of the cases already in the case-base as target. This is useful for evaluating the speed and economy of retrieval. However it tells us little about accuracy because if a perfect match exists in the case-base the I-CBR mechanism will always find it.

The second test mode used one of the cases as target but removed it from the case-base. This strategy is quite harsh particularly in the low-level case-base where the number of possible cases is small. If a case is unusual, removing it from the case-base makes it impossible to predict it. It is well known that fault frequencies follow the Pareto principle with 20% of the faults accounting for 80% of occurrences. It seems unreasonable to actually exclude a representative of a frequently occurring fault from the case-base during evaluation.

The graphs in Figure 8 show the results of the evaluation of the economy of the I-CBR retrieval. This data relates to retrieval of cases from the high-level case-base. The first graph shows data on retrieval economy with the target case in the case-base and the second graph show the data on tests without the target in the case-base. The other important dimension being considered in this evaluation is the impact of the free features on the case-retrieval process. There is little cost associated with providing these features to the diagnosis system; the question is whether they are useful in the diagnosis.

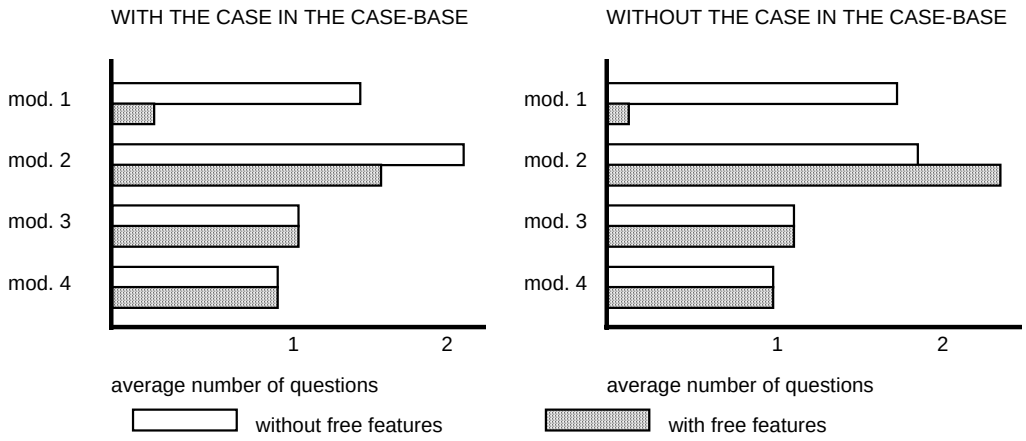

Figure 8. Some statistics on the economy of the retrieval mechanism.

The white bars in the graphs show retrieval cost without free features. In that part of the evaluation the free features were considered as expensive features (i.e. not available in advance). In that situation there is no initial reduction of the case-base based on the free feature data. The conclusion from this data would be that the free features are quite discriminating in detecting faults in Mod. 1 but not useful in detecting faults in Mod. 3 and Mod. 4. So, aggregated over all faults, it is useful to have the information contained in the free features.

In conclusion , the maximum number of questions that can be asked with the free features already available is three and without free features available is five. The average number actually asked is 0.74 and 1.56 respectively. (The first figure is less than one because for some faults the combination of free feature values directly locates a case.) The second issue in evaluating the system is accuracy and producing a meaningful evaluation of this is quite problematic. As already said, the system will always retrieve a perfect match if there is one in the case-base. The tests on retrieval with the high-level case base produced 2 errors from 53

when the target is not in the case-base. The errors rise to about 30% with the smaller low-level case-bases if the target is not in the case-base. So we can be optimistic that if the case-base is of reasonable size the retrieval accuracy will be good.

4.3 Evaluation of NODAL$_{CBR}$

Since a model-based system already exists for troubleshooting the switching-mode power supplies it is instructive to compare the economy of NODAL$_{CBR}$ in asking questions with it. The cases in NODAL$_{CBR}$ have the structure shown in Figure 9. The free features correspond to the function test results mentioned in section 2.2.

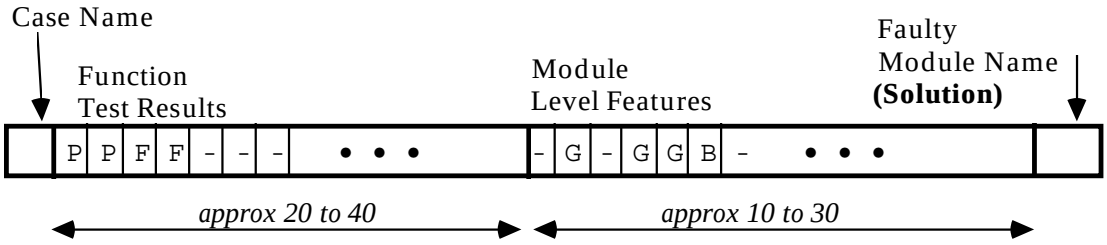

Figure 9. A typical case structure in NODAL$_{CBR}$.

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 2. 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 2. 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-Power-Supply-1	G	B															
Driver&Power-Switch-1	G	G				G	G		B								
Control-Circuit-1	G	G		G	G	B	G		G								G
Clock-Generator-1	G	G		B	B	G	G		G								
Clock-Generator-2	G	G		B	G	G	G		G								
Clock-Generator-3	G	G		G	B	G	G		G								
Driver&Power-Switch-2	G	G				G	B		G								
Driver&Power-Switch-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-Rectifier-Pos-1	G	G		G	G	G	G		G	G			B				
Output-Rectifier-Pos-2	G	G		G	G	G	G		G		G		G	B			

The dialogue with the system proceeds as follows:-

```
Selecting function test failure cases : Retrieved 14
  > (INPUT-FILTER-1 LOCAL-POWER-SUPPLY-1 DRIVER-AND-POWER-SWITCH-1 CONTROL-
CIRCUIT-1 CLOCK-GENERATOR-1 CLOCK-GENERATOR-2 CLOCK-GENERATOR-3 DRIVER-AND-
POWER-SWITCH-2 DRIVER-AND-POWER-SWITCH-3 XFMR-1 XFMR-2 XFMR-3 OUTPUT-
RECTIFIER-POS-1 OUTPUT-RECTIFIER-NEG-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-GENERATOR-1 CLOCK-GENERATOR-2)

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

The 14 cases shown in Table 2 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 a specific power-supply 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 the chart marked Category 1 in Figure 10. Two other smaller evaluations are shown in the other charts in Figure 10. 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.

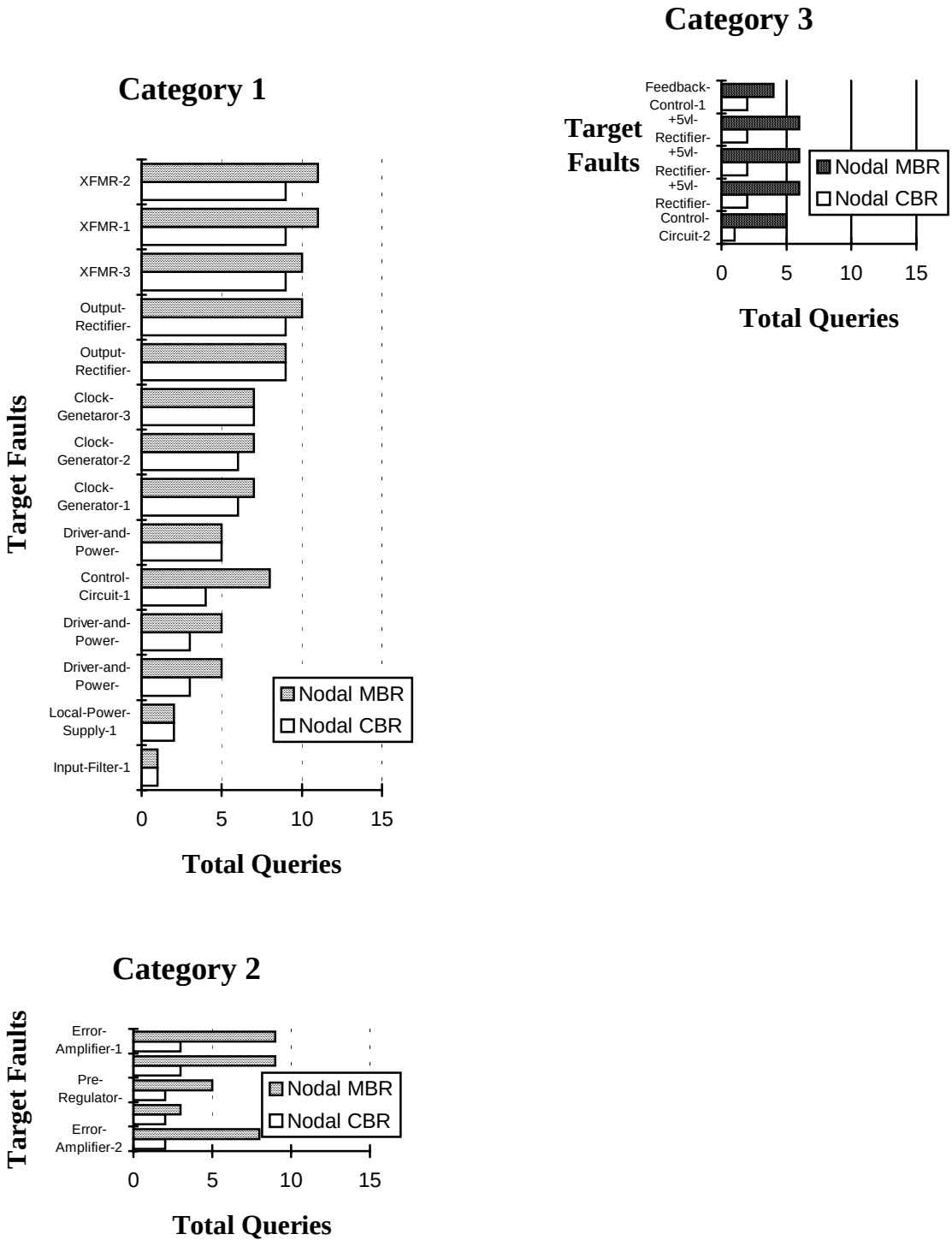

Figure 10. Some comparisons of the numbers of questions generated by the MBR and CBR systems.

5 Experiences

In diagnosis conclusions are drawn by abduction from a set of observed symptoms. This set of symptoms is normally only a subset of the observations that might be relevant. This task

structure is incompatible with the standard CBR strategy where a complete problem description is required in advance. In the incremental model of CBR presented here case retrieval can be initiated with a partial case description and the system indicates further information that might be useful in reducing the set of fault candidates. This means that the operators exploration of the faulty system is guided by the incremental CBR mechanism.

The obvious alternative to CBR in developing knowledge based systems for fault diagnosis is model based reasoning (MBR). As mentioned earlier, goal-directed MBR systems for fault diagnosis can readily generate questions for the user. They have the added advantage that the line of questioning seems logical as the system is evaluating a hypothesis or goal internally. In I-CBR questions are selected on information theoretical criteria with the motivation of reducing the candidate set. The experience with μI-CBR and with $NODAL_{CBR}$ is that the line of questioning may appear rather odd. There is no locality of reference with questions hopping between different parts of the board. The extent to which this is a problem needs to be evaluated by asking an experienced test technician to evaluate the system from a user perspective.

The process of reducing the candidate set must happen at run time and can be computationally expensive if the conflict set is large. The solution that we have adopted is to control the inclusion of new cases in the case-base. New cases are not included if identical ones already exist. This case learning mechanism has a spin-off advantage in that it highlights any conflicts in the case-base. This is important because conflicts or inconsistencies in the case-base cause the discrimination mechanism to fail.

An important postscript to these exercises that is true for CBR in general is that CBR does not eliminate the need for knowledge engineering. This is well understood by CBR practitioners and was evident during the development of these systems. Before developing the CBR system the circuit had to be sufficiently well understood to partition it into meaningful modules. The diagnostic process had to be understood in order to select useful tests and to partition test results into categories. CBR may reduce knowledge engineering requirements but it does not eliminate them.

6 Conclusions and Future Work

In this paper we have identified a class of diagnosis problem for which a complete problem description is not available at the beginning of the diagnosis. This presents a problem for the conventional CBR strategy where a complete case description is required in advance. We have introduced I-CBR, a case retrieval method that can initiate case retrieval using a brief case description and prompt the operator for extra discriminating information about the target case.

We have evaluated this retrieval mechanism in two systems for electronic fault diagnosis and found that it is economical in the amount of information required for case retrieval and accurate if the case-base has sufficient coverage.

One extension that might be useful in other diagnosis problems would be to include a measure of the relative cost of tests in the test selection process. The techniques for cost sensitive learning introduced by Tan and Schlimmer [8] could be used to do this. However, this was not considered useful in either of the two problems considered here because the expensive tests all have more or less the same cost associated with them.

References

[1] Cunningham P and Brady M, Qualitative reasoning in electronic fault diagnosis, in *Proceedings of IJCAI-'87*, McDermott J, ed., (Morgan Kaufmann 1987) 443-445.

[2] Cunningham P, *Knowledge Representation in Electronic Fault Diagnosis,* Ph.D. Thesis, Department of Computer Science, Trinity College Dublin (1988).

[3] Kriegsmann M, and Barletta R, Building A Case-Based Help Desk Application, *IEEE Expert,* 8 (1993)18-26.

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

[5] ond M A, Distributed cases for case-based reasoning: Facilitating use of multiple cases, in *Proceedings of AAAI-90,* (AAAI Press/MIT Press, 1990), 304-309.

[6] Smyth B and Cunningham P, A Comparison of Incremental Case-Based Reasoning and Inductive Learning in *Advances in Case-Based Reasoning*, Lecture Notes in Artificial Intelligence, Haton J-P, Keane M, and Manago M, eds., (Springer Verlag 1995) 151-164.

[7] Smyth B and Cunningham P, Déjà Vu: A Hierarchical Case-Based Reasoning System for Software Design, in *Proceedings of ECAI*, B Neumann, ed., (John Wiley 1992) 587-589.

[8] Tan M and Schlimmer J, Cost-sensitive concept learning of sensor use in approach and recognition. *Proceedings of ML-89,* (1989) 392-395.

[9] Turney P D, Cost-Sensitive Classification: Empirical Evaluation of a Hybrid Genetic Decision Tree Induction Algorithm, *Journal of Artificial Intelligence Research*, 2, (1995), 369-409.