MDS: An Integrated Architecture for Associational and Model-based Diagnosis

Xudong W. Yu
Dept. of CS
SIUE
Edwardsville, IL 62026
xyu@siue.edu

Gautam Biswas
Dept. of EECS
Vanderbilt University
Nashville, TN 37235
biswas@vuse.vanderbilt.edu

Jerry Weinberg
Dept. of CS
SIUE
Edwardsville, IL 62026
jweinbe@siue.edu

Abstract:

This paper discusses the design and implementation of an integrated diagnosis system, MDS (Multi-level Diagnosis System), which combines associational and model-based approaches to diagnosis. The design and implementation of the associational module is tailored to achieving efficiency in routine diagnostic problem solving, and to providing a desirable interface for the users. The model-based diagnosis module is developed to achieve completeness and consistency in the fault isolation task, and to avoid the brittleness that often occurs in associational systems. MDS addresses the important issue of combining the use of “deeper” knowledge in the form of a system model with “shallow” (or associational) knowledge using a diagnostic controller to improve completeness and consistency without sacrificing efficiency. The diagnostic controller also employs a methodology for automated knowledge refinement, by identifying incomplete and inconsistent rules and diagnostic tests in the associational module and by performing updates to correct problems. This paper focuses on the design and implementation of the diagnostic controller.

1. Introduction

A field mechanic looks at the trouble report generated by the flight crew for an airplane which pulled up to the gate about 20 minutes ago, then walks up to the airplane, performs a half dozen tests, and after deliberating for a while, ends up swapping a valve in the pneumatic system, which fixes the problem indicated in the report.

The mechanic then walks to another airplane with a similar trouble report and attempts to isolate the problem following the same procedures, but this time, cannot seem to isolate the problem. He then sits down, studies the schematic diagrams of the pneumatic system for about *thirty minutes, goes back to the airplane, performs a few more tests, and finally discovers that the problem this time is a broken wire at the back of the controller.*

An understanding of the mechanic's reasoning processes and the activities he performs provides much insight into how human experts solve complex problems. This understanding can be essential when we construct automated computer-based systems and computer assistants for complex problem solving tasks. In the first case, the trouble report immediately indicated to the mechanic that the situation was similar to cases he had successfully dealt with before. Based on his experience, he made a quick guess as to where the problem might be and proceeded to verify his conjecture by doing some relevant tests. The results confirmed his initial guess, and he quickly fixed the problem by swapping the faulty valve. In this case, the mechanic relied mainly on his past experience to quickly narrow down the problem to a small number of possible faults, and a few directed tests helped him confirm the true fault.

This scenario suggests that effective automated troubleshooting systems can be built by capturing experts' experience, and transferring them into "rules of thumb" that can be encoded into computer programs. From the early 70's, computer systems have been developed that focus on the use of this knowledge for effective diagnosis of complex systems. These systems have had their share of successes, but they have also exhibited a number of very important limitations. Most notably, they tend to be brittle, i.e., they fail in situations that are not specifically covered by the set of explicated rules. Factors that contribute to this brittleness include the incompleteness of experts' knowledge and the difficulties in translating experts' heuristics into a cohesive set of rules.

To see how the associational systems can be improved, it is again useful to look at what our expert mechanic did when faced with an unfamiliar situation. He first tried to cast the new problem (in the second airplane) in terms of problems he had seen before, but soon realized that this problem was different and solutions he had tried before wouldn't work. He then went back to study the system from a more fundamental perspective (i.e., study the system schematics and functionality to determine components that could be linked to the observed symptoms). This led to new possibilities. Additional tests related to these hypotheses helped to isolate the real problem.

This example illustrates that effective diagnosis can also be accomplished by relying on fundamental knowledge about a system. Since the mid 80s, diagnosis system design has focused on the use of this fundamental knowledge to achieve efficient and effective diagnosis (e.g., [8]). The fundamental knowledge is typically expressed as a model of the correctly functioning system that can be used to generate expected system behavior. For diagnosis, this behavior is compared with the observed behaviors of the system, and discrepancies are analyzed to derive possible faults. Typically, system models that derive behavior from structure are easier to construct for human-engineered systems, where design documents in the form of schematics and system specifications are readily available. This is in contrast to systems, such as physiological processes, where the mapping from structure to function is not well defined, and the same component can play different roles in different contexts. The ability of a model-based diagnosis system to correctly identify different faults is very dependent on the existence of a good *model* of the system ([8]). Therefore, constructing adequate models becomes a crucial task for successful diagnosis.

The use of more fundamental and first principle knowledge in model-based diagnosis systems enable them to deal with a more comprehensive set of situations and previously unseen faults. This mitigates the *brittleness* problem suffered by traditional associational systems. However, a disadvantage of using model-based diagnosis techniques, especially for large, complex systems with large number of components and possible interactions, is that the diagnosis process can become computationally expensive. This is not to say that the model-based diagnosis system is always going to be less efficient. As Davis has correctly pointed out ([9]), the computational speed of the system does not depend on the form in which knowledge is used, but on the level of detail that this knowledge represents. Knowledge based systems that use associational knowledge derived from human experts often tend to produce more efficient performance because the experts have already succeeded in "inventing the right vocabulary and the right set of abstractions" ([9]) for the tasks at hand. It is not easy to use generic model building techniques to develop models that include the right level of detail to cover all faults of interest while ensuring irrelevant details are omitted. Expert designers and analysts who are good at creating system models, are often not the expert diagnosticians. To address these problems, AI researchers have investigated strategies that focus the diagnosis to make it more efficient using techniques for model abstraction ([4]) and functional decomposition ([13]). Prior knowledge about component failures, expressed as probabilities have also been used to prune the search space ([11]).

Our research addresses the above problem by adopting an integrated approach to diagnosis. The primary goal is to develop a robust and efficient diagnosis system for complex, real-world systems. This approach combines the associational knowledge available from human experts with a model-based approach. This idea is to employ the more efficient and tailored associational knowledge for routine faults. When the associational module fails, the model-based diagnosis component is invoked to derive a correct solution. This way the system maintains a good tradeoff between efficiency and robustness. In addition, our system employs knowledge refinement techniques to transfer new knowledge to the associational module from the model-based component to improve problem solving efficiency. In a way, this transfers more generic, fundamental knowledge into a form more useful for diagnosis. This process leads to improvements in efficiency of diagnosis without sacrificing reliability.

These ideas are incorporated into Multi-level Diagnostic System (MDS), a system that assists mechanics[1] in the diagnosis of complex aircraft subsystems ([38]). These subsystems, such as the pneumatic system of an airplane, cover multiple domains (hydraulic, mechanic, thermodynamic, and electric), and individual components can assume multiple behavior modes. MDS contains two diagnostic modules: (i) an associational module that uses heuristics extracted from troubleshooting manuals and human experts, (ii) an *MBD* (*Model-Based Diagnostic*) module that relies on behavior and functional knowledge of the system to be diagnosed. The MBD module performs steady-state diagnosis based on the assumption that (i) the system is operating in a normal, steady state before the failure and (ii) the components degrade in a gradual, continuous manner. The activities of the two modules and the knowledge/data transfer between the two are handled by a third module, the *diagnostic controller*.

This paper focuses on the design and implementation of the diagnostic controller. This involves two primary tasks: (i) controlling the interaction among the two diagnostic modules, and

[1] Our original target users are the mechanics at Federal Express Corporation.

(ii) updating and refining the associational module using results derived from the MBD module when the associational module is found to be incomplete or in error. To facilitate this discussion, the two diagnosis modules are briefly presented in section 5, but additional details can be found in ([2] [3] [39]). The system has been applied to trouble-shooting the pneumatic system of the DC-10 aircraft.

2. Background

A number of techniques have been developed for automated computer-based diagnosis in different domains. Early programs employed *decision trees*, *fault directories*, and *probability theory* techniques for diagnosis tasks. These approaches were successful in simple applications that involved well-understood systems in narrow and carefully chosen domains. However, they suffered serious drawbacks when attempts were made to scale them up to complex systems where the possible number of interactions among components is high. The primary pattern was that the complexity resulted in incomplete and inconsistent diagnostic knowledge, which in turn produced incorrect diagnoses.

To address the problems of computational complexity and reliability, artificial intelligence and knowledge based techniques were applied to building diagnostic systems. An important characteristic of early AI work was the use of heuristics and judgmental knowledge of human experts to improve system performance. For example, the MYCIN system ([31]) encoded expert knowledge in the form of heuristic production rules with certainty factors. Other approaches used causal models (e.g., CASNET ([38]) and *set covering* techniques that allowed for simultaneous diagnosis of multiple diseases (e.g., [26]). As discussed earlier, these systems exhibited performance problems that could be attributed to: (i) incompleteness and inconsistencies in the knowledge of human experts, (ii) the difficulty in extracting heuristic information from human experts, and (iii) the task dependency of the derived knowledge. These limitations motivated the development of *model-based* systems.

The key to the model-based approach to diagnosis of a device (or system) is the availability of knowledge about the structure and behavior of the *correctly* functioning device, and a means for explicit representation of that knowledge (e.g., via measured variable values). Based on this knowledge, the behavior of the device or system can be predicted through derivation and simulation (e.g., GDE ([10]), Sherlock ([11]), GDE+ ([34]). The General Diagnostic Engine, GDE, a landmark system developed by de Kleer and Williams ([10]) performed multiple fault diagnosis using an Assumption-Based Truth Maintenance System (ATMS). For purposes of efficiency, diagnosis systems also store, as part of the model, nominal behavior of the device for standard input sets (e.g., [15] [19]). Model-based approaches have also employed directed causal relations between system variables for diagnostic analysis ([24]). Qualitative simulation methods have also formed the basis for monitoring and diagnosis (e.g., [12]). TEXSYS ([16]) used the GDE-like methodology for monitoring and diagnosis of a complex thermal system. DEDALE ([7]) employed *order of magnitude* relations. This work was extended to deal with more continuous analog systems (CATS ([21])). Gallanti, et al. ([15]) combined a Boolean model for candidate generation with a simplified quantitative (difference equation) model for candidate verification. Their work focused on control issues and did not relate faulty parameters to individual component failures. More recently, there has been work that employs a temporal causal graph and higher order derivatives to model system dynamics, and represent faulty situations as dynamic transients ([21]). A progressive monitoring scheme compares observations to the predicted transients to isolate the true faults.

To improve the diagnostic efficiency, diagnostic systems have also incorporated fault models (e.g., Sherlock([11]), GDE+ ([34])). Sherlock used fault models to rank candidates. GDE+ used fault models to eliminate spurious candidates (i.e., candidates that are inconsistent with the set of known faults) and MIMIC ([12]) used fault models to hypothesize candidates.

Techniques have been developed that combine different approaches to diagnosis. Fink and Lusth's IDM system ([14]) combines the use of associational and functional reasoning for diagnosis of complex electro-mechanical systems. IN-ATE ([5]) uses associational rules to generate initial candidates (i.e., one or a set of suspects) and then uses a model-based approach to further test and discriminate among the candidates. ABEL ([25]), a system developed for medical diagnosis combines the use of multi-level causal models that describes domain phenomena at different levels of granularity for effective problem solving. More recently, many hybrid systems have been developed that integrate different (usually two) modules, such as neural network nad model-based systems ([29]), neural network and associational systems ([34]), constraint satisfaction and case-based systems, (e.g., [18] [30]), and model-based with case-based (e.g., [27] [31]). The modules these systems integrate are different from MDS : (i) the individual modules they use (none of them combine model-based with associational rule base), and (ii) the way the two module cooperate to generate better performance. However, their goal of achieving efficiency and completeness are similar to ours.

3. Architecture of MDS

The architecture of MDS system is illustrated in Figure 1. The associational module uses rules that are direct associations between symptoms and hypotheses. Since it is created by human experts and is based on the vocabulary and tasks that the aircraft mechanics use to perform their day to day operations, it helps define the right level of detail at which diagnostic analysis needs to be performed for particular tasks. The knowledge for this module was extracted largely from troubleshooting documents such as the TAFI (Turn Around Fault Isolation) manuals[2], and human experts (both mechanics and engineers). This module also provides the appropriate vocabulary used in the system's interaction with the users (i.e., the mechanics) in the form of partial decision tree which most mechanics are trained to use. The MBD module incorporates

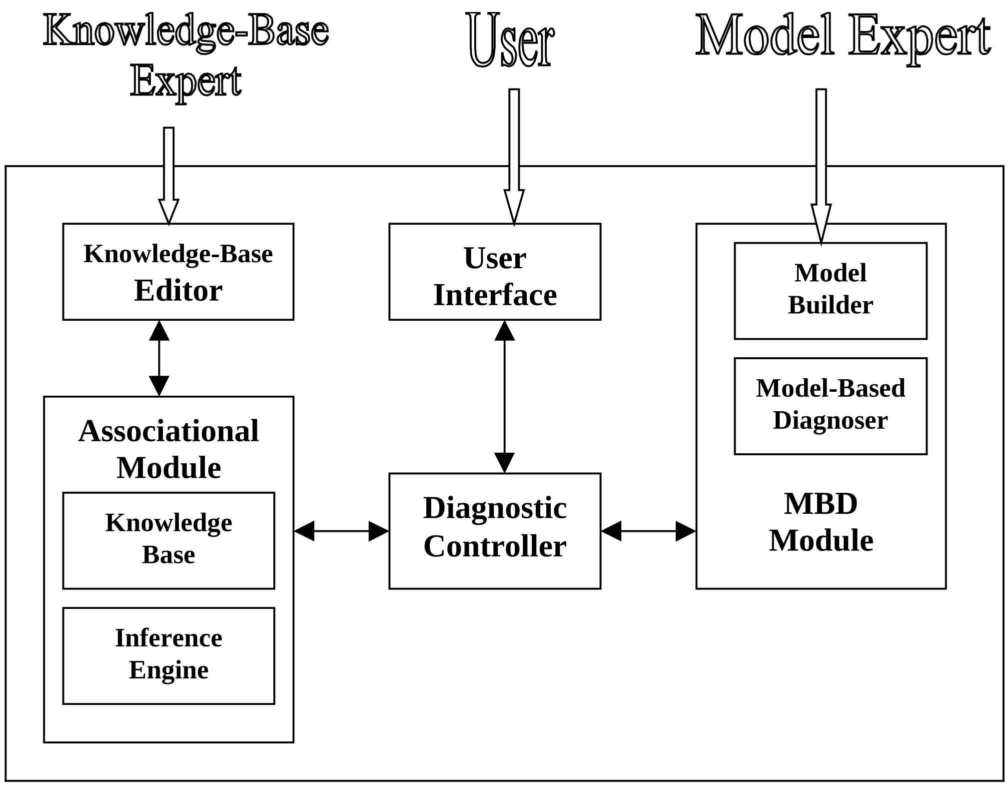


Figure 1: Architecture of MDS

[2] The TAFI manual is a troubleshooting manual for DC-10 aircraft, developed by McDonnell Douglas Corporation.

schematic, functionality, and behavioral models of the ([2]). Every effort was made during the design phase to make it complete and consistent. It is mainly used for diagnosis when the associational module fails to identify faults in a given situation. Furthermore, it is used to identify errors in the associational module and assist engineers and expert mechanics in updating the associational module. However, when updating the associational module, we only add faults that have occurred multiple times, and include tests that are suggested by engineers and mechanics. In this way, we maintain a relative small knowledge base in the associational module for efficient reasoning.

To achieve the integrated capabilities, the *diagnostic controller* coordinates the diagnostic activities between the associational and MBD modules. The idea is to derive typical faults quickly using symptom-cause associations and drop down to the MBD module only for unusual cases. During diagnosis, the associational module is invoked first. If a conclusion is derived, then the MBD module is used to verify the conclusion. If the result is correct, then the problem is solved. Otherwise, the MBD module goes back and deduces the solution from the initial symptoms. When the associational module cannot derive a definite conclusion, the MBD module takes over. Information obtained by the associational module is transferred to the MBD module by the diagnostic controller. Conclusions as well as intermediate results derived by the two modules are used by the diagnostic controller to identify problems within the associational module.

4. The Pneumatic System of the DC-10 Aircraft

Our modeling and diagnosis tasks focus on the part of the pneumatic system (Figure 2) that regulates air pressure and temperature drawn from one of three engines before it is delivered through the manifold system to different subsystems of the aircraft that constitute loads (e.g., the wing de-icing system).

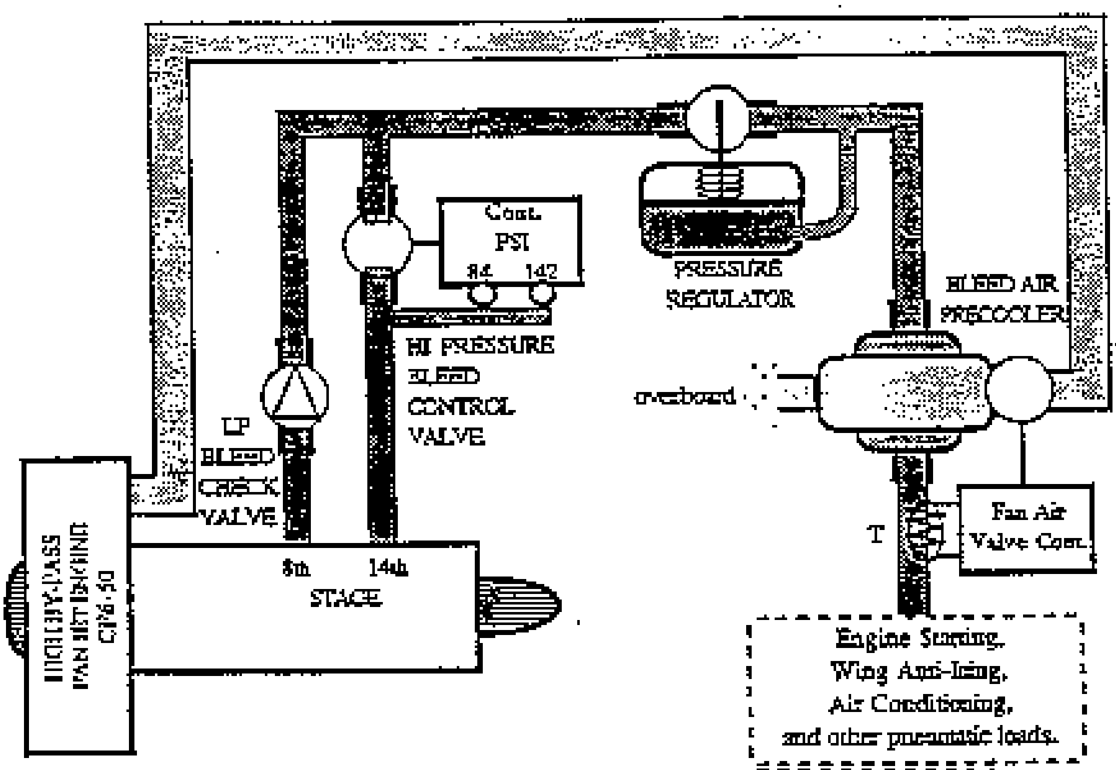


Figure 2: The Pneumatic System

The pneumatic pressure is regulated by a *pressure regulator subsystem*. The pressure regulator valve is modeled as a first-order system, where the opening of the regulating valve is determined by the changes in pressure at the regulator output. The temperature is controlled by a *pre-cooler subsystem*, whose primary component, a heat exchanger, draws cool air from a second source to cool the bleed air from the engine. Feedback mechanisms sense the temperature at the pre-cooler output. This information is fed back to the valve controller that fixes the opening of the valve to control the amount of cold air input to the heat exchanger, using the power obtained from the hot air transmitted through the sense line. For the diagnosis model, both the pressure regulator and pre-cooler subsystem are modeled in more detail in terms of primitive components. For

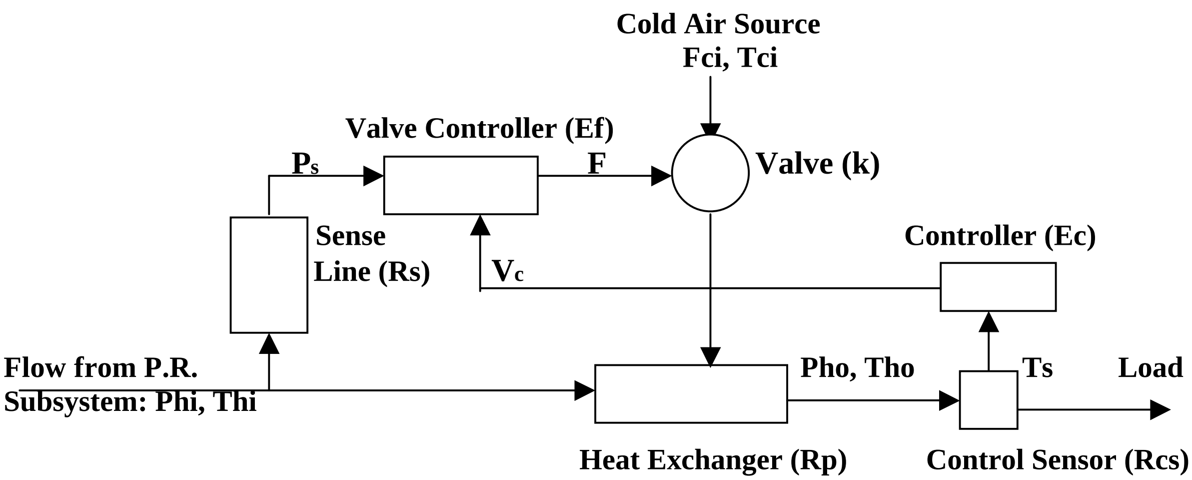


Figure 3: The Pre-cooler Subsystem

example, the pre-cooler subsystem is modeled in terms of six primitive components (Figure 3): (i) the heat exchanger, (ii) the feedback controller, (iii) the valve, (iv) the valve controller, (v) the temperature control sensor, and (vi) the sense line.

5. The Diagnosis Modules

This section provides an overview of the associational and model-based diagnosis modules. Details of the design and implementation of these modules are presented in [2] [3] [39].

5.1. The Associational Module

The associational module adopts a conventional expert system architecture with the focus on developing methods for generating partial decision trees that indicate to the mechanics suggested sequences of tests for fault isolation. The system has three primary components: the *knowledge base*, the *inference engine*, and the *decision tree generator*.

Information in the knowledge base is derived directly from human experts and existing troubleshooting manuals (in this case, the TAFI manual), and is represented as: (i) *production rules* that link symptoms to hypotheses, and (ii) *a list of available tests*. Production rules are used to derive plausible hypotheses from available symptoms. A typical rule has the form (LHS BF RHS BV), where LHS (Left Hand Side) specifies one or more symptoms and situations that need to be true for the rule to be fired. RHS (Right Hand Side) contains the conclusions of the rule i.e., possible hypotheses that explain the given symptoms. BV (Belief Value) specifies the confidence level associated with each hypothesis in the conclusion on a scale of 0 to 1. BF (Belief Function) specifies the confidence level of the rule itself. The Dempster-Schafer probability model is used to combine various belief values during the inferencing process ([1]). An example of a rule used in the current system is show below:

```
(|rule121|     ((((system)(pneumatic)) ((temp_light) (on)) ((stage_light) (normal))
                 ((pneu_temp) (high)) ((pneu_pressure) (normal)) 0.7)
               ((((hyp) (heat_exchanger))) 0.7) (((hyp)(valve))) 0.6)))
```

The above rule states that if the system is pneumatic, the temperature light is on, the stage light is normal, the pneumatic temperature is high, and the pneumatic pressure is normal, then the two possible fault candidates are the hear exchanger and the valve, with belief values of 0.7 and 0.6, respectively. The overall confidence level for this rule is 0.7. An interpretation of the later value is that the importance of this fact in the fault isolation process is 0.7.

The test list contains relevant tests that either support individual hypotheses or help to further discriminate among the individual hypotheses in the possible faults list. Each test in the list contains the following information:

- Rankings of the tests in terms of the three criterion: (i) cost of the test, (ii) time required to perform the test, and (iii) the predictability of the test,
- How the result of that test may affect other tests, and
- Conclusions or intermediate results that may be derived directly from the result of the test, when one or more hypotheses are verified. (Otherwise, the result of a test is considered a new symptom that can be used to update the hypothesis list.)

The inference engine controls the diagnostic activity within the associational module. It assumes an iterative *hypothesize, test, and refine* control structure ([1] [39]). The system first derives initial symptoms from preliminary information collected by mechanics. A set of hypotheses (possible malfunctioning components) is then generated from a initial set of symptoms using a forward chaining reasoning mechanism. Given the set of hypotheses, a backward chaining reasoning mechanism is used to select tests that either *discriminate* among candidate hypotheses, or *verify* that a potential candidate is actually faulty. This basic forward and backward chaining reasoning structure is adopted from the MIDST expert system shell ([1]). The tests are then ordered and presented as a partial decision tree. In the current system, the tests are ordered based on a value computed from the three criteria listed above. The weighting scheme for the criteria is determined by mechanics ([39]) based on the particular situation. The test with the highest rank becomes the root of the decision tree. For each possible result of that test, a new set of tests is generated, and their scores are calculated. The test with the highest score on each branch becomes the root of that sub-tree. This process continues recursively until either a leaf node is encountered (i.e., the child of a test is a conclusion) or the tree has reached a predetermined number of levels (see figure 5 for an example tree).

The user can study the decision tree presented, and conduct one or more tests[3]. Results of the additional tests performed are fed back into the system, and they may determine that a particular hypothesis has been verified. This would complete the current diagnostic session. Otherwise, if no conclusion has been reached, the system goes back to derive new symptoms based on the test results, and uses these symptoms to update and prune the list of possible hypotheses. A new set of tests are then selected, and the diagnostic session continues. The system repeats the *symptoms*-hypotheses-test-decision tree loop until the faulty components are identified.

It should be pointed out that the reasoning process of associational module is designed to make the testing process efficient and flexible for the users. Rather than asking a mechanic to conduct one test at a time, it generates a partial decision tree dynamically, so that a set of related tests can be conducted before the user interacts with the system for additional information and help. The decision tree approach provides the user both flexibility (test orders are determined dynamically) and efficiency (the results of one test will change the importance of some other tests). From our interaction with mechanics, we found that they prefer the decision tree approach. On the other hand, just providing a complete, static decision tree annoys them, because they felt tied to a pattern that they cannot alter, even if their experience indicates otherwise.

[3] Users are not restricted to perform only the tests that appear on the decision tree. They may choose to make additional and/or alternative tests and report results that they consider relevant.

5.2. The Model-Based Diagnostic Module

The Model-Based Diagnostic (MBD) module has two main components: the *model builder* and the *model-based diagnoser*. Given a description of the physical system, (e.g., system schematics and design documents that describe the functions of the components of the system and their interconnections) the model builder assists the expert in developing equation models of the physical systems that can be used effectively for diagnosis. Based on the model of the system, the diagnostic engine isolates faulty components using an iterative sequence of candidate generation and testing, and measurement selection.

5.2.1. Equation Model of the Pre-cooler System

The model builder assist the human expert in developing model of a physical system in the form of a set of output equations that relate measurable parameters to parameters that are linked to individual components. These output equations are then used by the MBD module to perform model-based diagnosis. Since the modeling process is not the focus of this paper and is discussed in detail elsewhere ([2] [39]), we will only present the output equations here. However, it should be pointed out that the MBD module is independent of the modeling scheme used to create the equations. It works with output equations that are either generated by the model builder ([2]) or provided by human experts ([3]).

For the pre-cooler system shown in figure 2 and 3, the following two sets of output equation are generated for output parameters T_{ho} (equations 1 through 6) and P_{ho} (equations 7 and 8), the output temperature and pressure of the bleed air at the load, respectively.

$$T_{ho} = T_{hi} - C_h (T_{hi} - T_{ci})(\frac{v}{l} + \frac{l}{C_c R_c D}) \tag{1}$$

$$D = C_c C_h R_p^2 \frac{l^2}{v^2} + \frac{cR_p}{R_h R_c l}(C_c R_c (R_h + R_p) + C_h R_h (R_c + R_p)) + \frac{R_p^2}{R_c Rh}(R_c + R_h + R_p) \tag{2}$$

$$C_c = Vc\rho = \Delta t F_c c\rho = \frac{l}{v} F_c c\rho \tag{3}$$

$$F_c = \sqrt{\frac{P_3}{C_3}}(X_{\max} - X) \tag{4}$$

$$X = \frac{A}{k}(P_{ro} - R_s F_s)E \tag{5}$$

$$E = E_f + E_c (T_{set} - R_{cs} Q_r) \tag{6}$$

$$P_{ho} = P_{in}(1 - \frac{R_p}{R_p + \frac{C_3}{x_h}}) \tag{7}$$

$$X_h = X_{set} - A_2 (P_{in} - R_{pt} f_{pt}) \frac{E_p}{K_p} \tag{8}$$

Parameters in the above equations are by illustrated in the table below.

Parameter	Their meaning in the Pneumatic System
T_{hi}	Initial temperatures of the hot air
T_{ci}	Initial temperatures of the cold air
V	Velocity of the hot air flow

L	Length of the path the air masses traverse in the pre-cooler
R_c	Resistance of the pipe line from the cold air source to heat exchanger
R_h	Resistance of the pine line from the hot air source to heat exchanger
V	Volume of the cold air in the heat exchanger unit
ρ	Density of the air
C	Unit thermal capacitor of the air
F_c	Flow rate of the cold air
R_s	Resistance of the sense line to liquid flow
R_p	Heat flow resistance of the heat exchanger
E_f	Effort transfer ratio of the valve controller
E_c	Effort transfer ratio of the controller
R_{cs}	Heat flow resistance of the control sensor
K	Spring constant of the cold air valve
X_{max}	Maximum length of opening of the cold air valve
P_3	Pressure difference over the cold air valve
C_3	Cold air valve constant
T_{set}	Desired temperature
P_{ro}	Pressure of the flow from the pressure regulator subsystem
A	Area of the opening of the sense line
P_{in}	Input pressure of the flow to the pneumatic system
X_{set}	Default opening of the pressure regulator
$A2$	Area of the opening of the pipe to the pressure regulator
R_{pt}	Liquid flow resistance of the pipe line from the hot air source to the
F_{pt}	Liquid flow constant
E_p	Effort transfer ratio of the feedback controller
K_p	Spring constant of the pressure regulator valve

For diagnosis purpose, parameters of a system are divided into four categories.

1. Output parameters – measurable parameters at the output of the system being diagnosed, such as T_{ho} and P_{ho}.
2. Input parameters – parameters associated with subsystems at the upstream, such as P_{in}, T_{hi}, and T_{ci}.
3. Component parameters – parameters that represent the functions of individual components in the system. For example, R_p represent the resistance of the heat exchanger to heat flow. Other

component parameters include: R_{s}, E_{f}, k, R_{s}, E_{c}, R_{cs}, k_{p}, and R_{pt}. The set of component parameters define the set of possible single fault hypotheses that are considered for diagnostic analysis.

4. Co-component parameters – intermediate parameters that combine equations from different subsystems and domains to establish direct links between output parameters to component parameters. For example, C_c represents the thermal capacitance of the heat exchanger. This parameter helps to connect the thermal and fluid parts of the system. They can be derived from component parameters and other system variables (e.g., see equation (3)), and, therefore, are not included as possible fault hypotheses. Other co-component parameters include: D, F_c, X, E, P, X_h. Note that a single equation can be generated for each output parameter by repeatedly substituting co-component parameters. However, this often produces complicated forms that result in less efficient analysis by our automated algorithm.
5. Constants – domain and system parameters whose value do not change within the subsystem, such as ρ, l and C_3.

5.2.2. Model-based Diagnosis

Given the set of measurements made on the system and the system description which include parameter definitions and the current set of output equations, the diagnostic algorithm is summarized as follows:

Repeat until one candidate is confirmed or all candidates are eliminated

1. Generate partial explanations, one for each new measurement.
2. Generate/update candidate set using best first search.
3. Perform test(s) selected using a decision theoretic scheme.

The diagnosis algorithm is based on the assumptions: (a) the system was operating in steady-state, and the measurement sampling rate is high enough to catch deviations soon after they manifest, and (b) components degrade in a continuous manner and there are no abrupt structural changes in the system. Diagnosis is initiated when observed system behavior deviates from a steady state. Again, the pneumatic system is used to illustrate the process. Details of step 1 and 2 can be found in ([3]).

Step 1. Generate *partial explanations* by performing qualitative causal analysis on the set of output equations. For each output parameter and a new measurement, first determine the sign of PDC(P_k,$W_{i(+|-)}$) (Possible Direction of Change of P_k caused by change in W_i) for all output parameter P_k and component parameter W_i pairs by computing the partial derivative

$$\frac{\partial p_k}{\partial w_i} = \frac{\partial p_k}{\partial s_1}\frac{\partial s_1}{\partial s_2}\cdots\frac{\partial s_{n-1}}{\partial s_n}\frac{\partial s_n}{\partial w_i}$$

Where $s_1 \ldots s_i$ are co-component parameters that link P_k to W_i. For example, the relation between R_s, and T_{ho} can be determined using equations (1)-(5):

$$\frac{\partial T_{ho}}{\partial R_s} = \frac{\partial T_{ho}}{\partial C_c}\frac{\partial C_c}{\partial F_c}\frac{\partial F_c}{\partial X}\frac{\partial X}{\partial R_s} = (-C_h(T_{hi} - C_{hi})\frac{l}{C_c^2 R_c D}) \bullet (\frac{l}{v}c\rho) \bullet (-\sqrt{\tfrac{P_3}{C_3}}) \bullet (-\frac{A}{k}F_s E)$$

Since the qualitative values of variables A, k, E, F_s, C_c, R_c, c, ρ, D, l, C_h, C_3, P_3, and v are all known to be + and our steady-state assumption implies that T_{hi}-T_{ci} > 0 (i.e., the initial temperature difference between hot and cold air may fluctuate but is always positive), the partial derivative evaluates to -, therefore, PDC(T_{ho}, R_s+) is -, and PDC(T_{ho}, R_s-) is +. Next, partial explanations for each measurable parameter is generated based on whether it is normal (within 2% of the normal value), above-normal (+), or below-normal (-), as described below:

- For each deviant output parameter Y, form a proposition formula: $X_{1(+|-)} \vee X_{2(+|-)} \vee \ldots \vee X_{n(+|-)}$, where $X_{i(+|-)}$'s are changes in the component parameters X_i's that are consistent with the observed deviation of Y, i.e., $PDC(Y, X_{i(+|-)})$ is equal to the deviation in Y. In our example, suppose the observed deviation for T_{ho} is +, the following partial explanation for T_{ho+} would be generated:

$$F(T_{ho}+) = R_p+ \vee E_f+ \vee K- \vee R_s- \vee E_c \vee R_{cs}-.$$

Notice that, based on previously established value of $PDC(T_{ho}, R_s-)$ (+) and $PDC(T_{ho}, R_s+)$ (-) R_s- is included in the formula while R_s+ is not. In other words, only a decrease in the resistance of the sense line is consistent with above-normal temperature. Further notice that although R_s- is included in the explanation, the low probability associated with the event (decreased resistance) will result in it being dropped from the candidate list for further analysis.

- For each measurement Y that is reported to be normal, form a proposition formula:

$$(\neg X_1 \wedge \neg X_2 \wedge \ldots \wedge \neg X_n) \vee (X_{1(+|-)} \wedge X_{2(+|-)}) \vee (X_{2(+|-)} \wedge X_{3(+|-)}) \vee \ldots \vee (X_{n-1(+|-)} \wedge X_{n(+|-)})$$

Each pair of (X_i, X_j) is included if their influence on Y are complementary (i.e., if $PDC(Y, X_{i(+|-)})$ is +, then $PDC(Y, X_{j(+|-)})$ is -, and vice versa.). This formula suggests that the X_i's are either all normal or at least two of them are deviant and their combined effect on Y is null. In our example, assuming the output parameter P_{ho} is normal, the following formula will be generated:

$$F(P_{ho}) = (\neg K_p \wedge \neg R_p \wedge \neg E_p \wedge \neg R_{pt}) \vee (K_p+ \wedge E_p+) \vee (K_p- \wedge E_p-) \vee \ldots$$

Note that each $\neg X_i$ implies both $\neg X_i+$ and $\neg X_i-$.

Step 2. Generate the *candidate/update candidate set* based on the current set of partial explanations. Our system uses a best-first search algorithm that generates candidates in order of their prior probabilities until either (i) the number of candidates reaches a preset threshold *k1*, or (ii) the prior probability of the best candidate is *k2* times greater than the prior probability of the last candidate. The following table shows the prior probabilities of the components in the pneumatic system:

E_f+	E_f-	E_c+	E_c-	K_p+	K_p-	K+	K-	R_p+
0.2	0.1	0.18	0.01	0.15	0.18	0.15	0.15	0.2
E_p+	E_p-	R_{pt}+	R_{pt}-	R_{cs}+	R_{cs}-	R_s+	R_s-	R_p-
0.1	0.2	0.2	0.01	0.1	0.05	0.2	0.001	0.001

For each component parameter, two probability values are used, one indicating the likelihood of its increasing, the other indicating the possibility of its decreasing. This use of prior probabilities allows the system to take into account that components are more likely to fail in one direction than the other. For example, the resistance of a pipe used for fluid transfer may increase over time but almost never decreases. In our example, given that T_{ho} is above normal and all other measurable parameters are normal, the following candidates are generated (*k1* and *k2* are both set to 25):

$$Cand = ((E_f+), (E_c+), (K-), (R_{cs}-), (R_p+, R_{pt}+), (R_p-, E_p-), (K_p+, R_p+), (K_p-, R_{pt}+),$$
$$(K_p-, E_p-), (E_p+, R_{pt}+), (E_p+, K_p+))$$

Notice that *Rs*- (the resistance of a sense line used for fluid transfer decreasing) is not included in the initial set even though it is consistent with the current observation because its extremely low probability.

Step 3. Perform measurement selection based on the established relationship between measurable parameters and individual components (the set of output equations) using an information-theoretic method similar to the one used in GDE ([10]). For each possible measurement O_i, we calculate $\Delta H_e(O_i)$ to evaluate the expected changes in the entropy of the system if the measurement O_i is made, using the formula:

$$\Delta\,\mathrm{He(Oi)} = \sum_{k=1}^{m} p(O_i = V_{ik}) \log p(O_i = V_{ik}) + p(U_i) \log p(U_i) - \frac{np(U_i)}{m} \log \frac{p(U_i)}{m}\ .$$

In the above formula, each V_{ik} ($1 \le k \le m$) is a possible outcome of measurement O_i, U_i is the set of candidates that do not predict a value for O_i. In the qualitative framework, a measurement can take one of three values: above-normal (+), normal (0), and below-normal (-). Give the current set of candidates C, the probability $P(O_i=V_{ik})$ can be calculated using the formula:

$$\mathrm{P(Oi = Vik)} = \sum_{C_l \in C} p(O_i = V_{ik} \mid C_l)\, p(C_l)\ ,$$

where $p(C_l)$ is the current probability for the candidate C_l. $P(O_i=V_{ik}|C_l)$ (the probability that parameter O_i will have value V_{ik} given that the candidate is C_l) can be easily calculated using qualitative analysis on the equations for O_i. When V_{ik} is either + or -, $P(O_i=V_{ik}|C_l)$ has value 1 if C_l contains only components that are in the corresponding partial explanation of O_l. It has value 1/3 when C_l contains components in partial explanations for both $O_{i}+$ and Oi-. It has value 0 when C_l does not contain any components in either of the partial explanations. When V_{ik} is 0, $P(O_i=V_{ik}|C_l)$ has value 1 if C_l does not contain any components in either of the two lists. It has value 0 when C_l only contains components in one of the partial explanations. It has value 1/3 when C_l contains components in both partial explanations. For example, given the two possible partial explanations for $P_{ro}+$ and $P_{ro}-$:

$F(P_{ro}+) = K_p+ \vee E_p- \vee R_{pt}+ \vee R_p-$ $\qquad$ $F(P_{ro}-) = K_p- \vee E_p+ \vee R_{pt}-$

Since E_p- is only in F(Pro+) and K_p- is only in F(P_{ro}-), (E_p-,K_p-) supports, with equal probabily, P_{ro}+ (with E_p-), P_{ro}- (with K_p-), P_{ro} normal (using both E_p- and K_p-). Therefore, we get p(P_{ro}=+ | (E_p-,K_p-)) = 1/3 Similarly, p(P_{ro}=+ | E_{cs}-) = 0 because E_{cs}- is not included in the partial explanations for P_{ro}.

After ΔHe(O_i) is calculated for all the remaining possible measurements O_i, the one with the smallest ΔHe(O_i) value is selected. In our example, the following ΔHe's are computed:

Test	P_{vc}	F_c	V_c	P_{ro}	P_{hv}	P_{prc}	T_{cs}	O_p
ΔHe	-0.909	-0.890	-0.727	-0.676	-0.457	-0.457	-0.362	-0.175

Based on this information, *P_{vc}* (The output power at the *valve controller*) was chosen as the next measurement. The initial probability of a candidate *CL* is calculated from the prior probability of its components using the formula:

$$p(Cl) = \prod_{c \in C_l} p(c) \prod_{c \notin C_l} (1 - p(c)) .$$

After each measurement, the probability of a candidate is updated using Bayes rule:

$$p(C_l \mid O_i = V_{ik}) = \frac{p(O_i = V_{ik} \mid C_l)\, p(C_l)}{p(O_i = V_{ik})} .$$

We do not compute p(O_i=V_{ik}) because it is the same for all candidates, and, therefore, would not affect their relative order. Continuing with our example, given the new measurement P_{vc} was recorded as being above normal, indicates that the fault is in the pre-cooler subsystem. As a result, the new candidate set is *Cand = ((E_f+), (E_c+), (R_{cs}-)).* By calculating ΔHe, the system identified that the measurement *V_c* (the voltage signal sent by the controller) would provide the most information. The result from *V_c* was reported to be normal, and that left *E_f+* as the only single candidate. However, the system also noticed that a double fault involving $R_{cs}+$ and E_c+ could also explain the symptoms observed so far. As a result, ($R_{cs}+$, E_c+) was also generated as a candidate. A measurement at T_{cs} was then suggested, and since it was normal, the system concluded that the actual faulty component is E_f (the valve controller).

6. The Diagnostic Controller

The diagnostic controller directs diagnostic activities within the system, and communicates information from the MBD module to the associational module whenever the associational module fails. This involves two primary tasks: (i) controlling the interaction among the two modules to solve particular diagnosis problems, and (ii) updating and refining the associational module from results derived from the MBD module when the associational module is found to be incomplete or in error. Specifically, the task of the diagnostic controller includes:

1. *Diagnostic control*: invoking the appropriate module at the appropriate time with the right information,
2. *Problem characterization*: identifying and characterizing the problem in the associational module when it produces an incorrect conclusion or fails to draw a conclusion, and
3. *Knowledge base refinement*: updating the knowledge base of the Associational module depending on the type of problem identified.

6.1. Diagnostic Control

A high level description of the overall control algorithm is shown below:

```
Step 1:   Diagnose using the Associational Module;
Step 2:   If faulty components are found
          Then {Invoke MBD module to verify the conclusion;
                If conclusion is verified, then done;
                Else {Diagnose using the MBD module;
                      Update the associational module; }}
Step 3:  Else {Invoke MBD module for further diagnosis;
```

Update the associational module; }

A diagnostic session begins with the associational module, and terminates in one of the following situations:

1. A confirmed candidate (single or multiple faults) is found.
2. All candidates generated initially are eliminated (or no candidate is generated at all).
3. More than one candidate still remains and cannot be further discriminated.

In situation 1, the MBD module is invoked to verify the conclusion, if it has not been verified before. This can often be done offline, especially in situations when the mechanic faces a busy schedule. If the conclusion is found to be correct, the diagnostic session is completed, and the faulty component(s) is (are) reported. However, if the conclusion cannot be verified to be correct, the system resumes diagnosis in an attempt to derive the faulty components using the MBD module.[4] If the MBD module is successful, the entire diagnostic record is then sent back to the diagnostic controller to further identify and characterize problems in the associational module. Situations 2 and 3 can be caused by incomplete and/or inconsistent knowledge in the associational module. In both cases, the MBD module is invoked to perform diagnosis. As in situation 1, the results from the MBD module are sent back to the diagnostic controller for problem identification. When control switches to the MBD module, the information is passed from the associational module in the form of:

- Results of the tests conducted and partial diagnosis conclusions derived by the associational module are passed to the MBD module. Tests conducted in the associational module fall into one of the following categories: (i) those that directly establish the status of individual components

[4] This can be done with a different mechanics since all diagnostic records were kept intact

(working/faulty), (ii) those that can be interpreted as directly measurable parameters (e.g., pressure, temperature) on individual components in the MBD module, and (iii) those that use heuristics to confirm or eliminate candidate components. In the MBD framework, tests in category (i) and (ii) are considered reliable, whereas test results in category (iii) are considered ad hoc, and therefore, not used to eliminate any candidates.

- Partial diagnosis results derived by the associational module are passed to the MBD module. Partial results occur when the associational module has narrowed down the problem to a smaller set of candidates but could not derive more specific results. This information is used by the MBD module to rank candidate when the measurements indicate that multiple solutions are possible. Candidates flagged by the associational module are given additional weights during the ranking process, while candidates that were eliminated by ad hoc tests (category (c)) will have their prior probability reduced for this particular diagnosis.

6.2. Problem Characterization and Knowledge Base Refinement

As discussed in the previous sections, knowledge in the associational module is represented in two forms: (i) production rules of the form (LHS BF RHS BV), and (ii) the list of available tests. Problems in the associational knowledge base can be attributed to inconsistency, which can be classified into:

1. Incorrect test specifications: the implications of a test are incorrectly specified.
2. Incorrect rules: either the LHS or the RHS of a rule is incorrect.

or incompleteness, similarly categorized as:

3. missing tests.
4. missing rules.
5. under-specified tests: possible conclusions are missing from the implications of a test.

A problem is identified when the associational module fails to reach a definite conclusion. Whenever this happens, the diagnostic controller characterizes the problem in terms of one of the five categories listed above. This is based on a step by step comparison of the candidate sets at each step of the problem solving process, the measurements made, tests conducted, and the final conclusions drawn. The characterization process is summarized below:

Case (i): the associational module eliminated all candidates.

- If the actual faulty component(s) identified by the MBD module were initially generated as candidates by the associational module and then eliminated, the problem is *incorrect test specifications*. The tests that caused this elimination can be identified by comparing the two different candidate sets generated by the associational module at various steps.
- If the candidates were never generated, then there are *missing rules* in the knowledge base.

Case (ii): multiple candidate sets remain, but no further tests are applicable.

- If the candidates that remain unverified are also generated by MBD module, and were retained after candidate testing, then identify the set of measurements that confirmed or eliminated them during the candidate discrimination phase. Check if these measurements were in the test list of the associational module. If they are, *under-specified tests* is the cause. Otherwise, the cause is *missing tests*.

- If these candidates were not generated by the MBD module, or were initially generated but later discarded during candidate testing, the associational module has generated spurious candidates from the initial symptoms. The rules that caused the generation of these candidates are *incorrect rules*.

Case (iii): the components declared faulty by the associational module are verified to be normal by the MBD module. This can be looked upon as a combination of cases (i) and (ii) since the real component(s) was (were) not confirmed, and candidate(s) that should be eliminated remained. After the problem is characterized, the next step is to update the knowledge base.

6.3. Knowledge Base Refinement

Knowledge acquisition and refinement are difficult problems. Recently, many methods have been developed for the verification and validation of existing knowledge bases (e.g., [17] [29]). Knowledge acquisition and refinement tools have been developed for general applications (e.g., [6] [23] [35]), and particular expert systems ([22]). These tools often guide the user in their modifying efforts. Methods have also been developed for automated knowledge acquisition that use induction of decision trees (e.g., [28]), rule induction methods (e.g., [20]), and rough set theory (e.g., [37]) to extract knowledge from databases.

As a step toward automating the knowledge base refinement task, our current system focuses on improving the completeness of an existing knowledge base by analyzing diagnostic cases where the MBD module performed better than the associational module. Specifically, the diagnostic controller starts the refinement process when the associational module fails with a problem that is later solved by the MBD module, indicating an incompleteness problem within the associational module. After characterization, the diagnostic controller deals with each one of the incompleteness problems as follows:

1. Missing tests: When the problem of missing tests (or measurements) are identified, new tests need to be added to the knowledge base. The effect of these tests, in terms of the candidates they confirm or eliminate, can be identified by comparing the two candidate sets that the MBD module maintained before and after the tests were made. Information about the subjective evaluative criteria for the tests as well as how the tests should be presented are obtained from domain experts. Therefore, addition of the new tests involves the joint efforts by the system and the domain expert.

2. Under-specified tests: When under-specified tests are found, these tests are updated automatically. Effects of the tests on additional candidates are added to the specifications of these tests. Again, these effects can be found by comparing the different candidate sets the MBD module generated during problem solving. The evaluation information on tests remain unchanged by default, but the domain expert may change them on his own initiative.

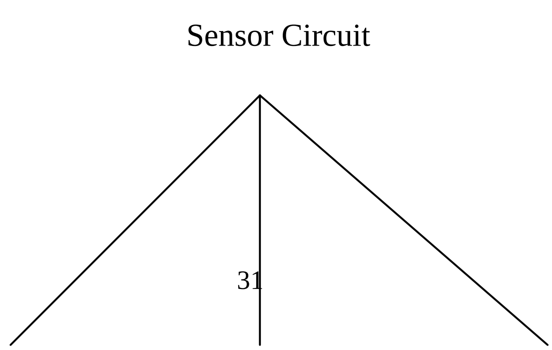
3. Missing rules: When a missing rule situation occurs, new rules are added to the associational module. These rules link initial symptoms to candidates that were generated

by the MBD module, but were missing from the candidate set of the associational module.

Inconsistent rules and tests are reported to the experts, along with detailed descriptions of the exact problem encountered by the system. For each incorrect test specification, the diagnostic controller reports the name of the test, the candidates it had incorrectly confirmed or eliminated, and the test results that are related to it. As for incorrect rules, the diagnostic controller reports the rule number and the candidates it had incorrectly included in its RHS. In both cases, the burden of actually correcting the knowledge base is left to a domain expert.

6.4. Example

We illustrate the working of the overall diagnosis system with an example. Consider the pneumatic system with a malfunctioning *controller*. The initial symptoms reported were: output temperature *Tout* above normal, input temperature *Tin*, input pressure *Pin*, and output pressure *Pout* normal. Since the problem appears to be in the temperature regulation part, the diagnosis system focus was on the pre-cooler subsystem. The associational module was invoked first, and it generated five candidates: (i) *sense line*, (ii) *heat exchanger*, (iii) *valve controller*, (iv) *valve*, and (v) *control sensor circuit wire*. Note that the *controller* was not listed; we are probably dealing with an incomplete associational module. The system generated the following *partial* decision tree (the leaf nodes are exclosed in <>):



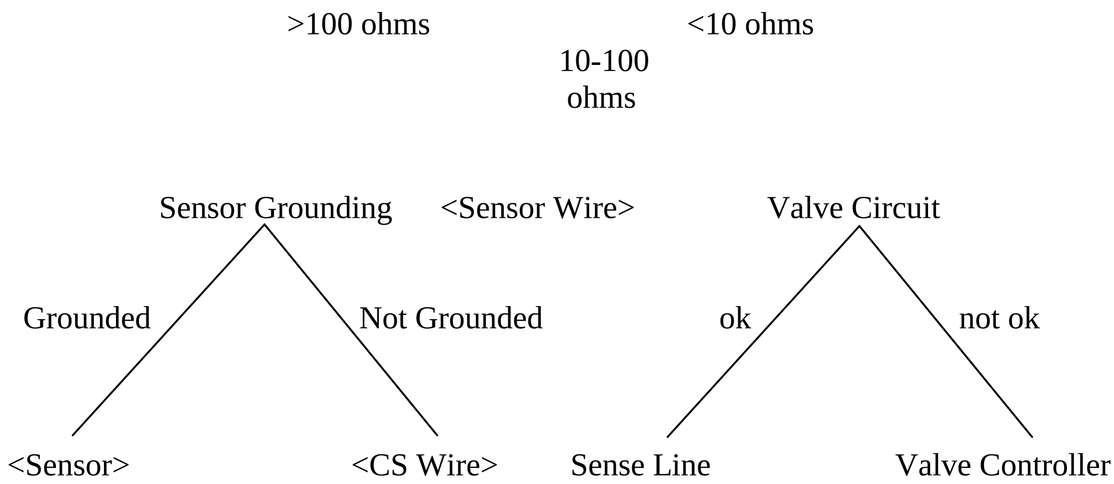


Figure 5: A Partial Decision Tree Used in the Example

The user can choose tests from the tree, or follow his or her intuitions. In this case, the user followed the tree and reported the following test results: (i) Sensor circuit -- the resistance difference between the control sensor circuit and the indicating sensor circuit was less then 10 ohms, (ii) Valve circuit (this include the valve and the valve controller) was found to be operating normally, (iii) Sense line -- no leakage or blockage detected, and (iv) Heat Exchanger (a child node of *sense line* in the initial decision tree and was therefore not shown in the partial tree)-- tested to be normal. Given these results, the associational module eliminated all candidates and was not able to isolate the problem. At this point, the MBD module was invoked to continue the diagnosis. Results of the tests were also transferred to the MBD module, with the following implications: (i) the *sense line* and *heat exchanger* were considered normal since they were tested directly, (ii) the correctness of the resistance of the sensor circuit was interpreted as an indication that the temperature from the control sensor (*Ts*) is normal since the temperature reading is a function of the resistance, and (iii) The *valve circuit* test checks the current at Pin 49 of the circuit and consider the circuit ok when it is properly grounded. However, this interpretation by the associational module was considered ad hoc. Consequently, components of the valve circuit were not eliminated. However, the test did indicate that both the *valve* and *valve controller* would be less likely candidates than what its prior probability indicated. Given this information, the MBD module generated three candidates: {*controller*}, {*valve-controller*} and {*valve*}, with the {*controller*} ranked as the most likely candidate. It then suggested that the voltage signal *VC* sent to the *valve controller* be measured. Since the output temperature is higher than normal, this voltage should be below normal. When that measurement did not concur with expectations, the MBD module concluded that the controller was actually the faulty component.

At this point, the diagnostic controller took over again to initiate the knowledge refinement process. Since the faulty component was not generated initially as a candidate by the associational module, a *missing rule* situation was identified, and the diagnostic controller added the following new rule to the associational knowledge base:

```
(|rule200|      ((((system)(pneumatic))((temp_light) (off))
                  ((stage_light) (normal)) ((pneu_temp) (high))
                  ((pneu_pressure) (normal)) ((pressure_light) (off))
                  ((sensor_circuit) (<10)) ((valve_circuit) (normal))
                  ((sense_line) (normal)) ((heat_exchanger) (normal))
                  ((valve_controller_valtage) (low)) 0.9)
                ((((hyp) (controller))) 0.9)))
```

The missing test *Vc* was also reported and the appropriate information about the test was requested from the domain experts.

6.5. Implementation

The Associational module is implemented in Common Lisp and C, and runs on Unix-based workstations. The current knowledge base contains about 130 rules and 30 tests for troubleshooting the pneumatic system of the DC-10 aircraft. Rules and tests can be entered into the system using the *KB Editor* that was extended from the *Rule Editor* used in the MIDST system. (MIDST is an expert system shell used for geological exploration. Details about MIDST system can be found in ([1]). A Graphical User Interface that displays partial decision trees, tests information, and conclusion(s) has also been implemented. This software runs in parallel with the inference mechanism of the associational module, and is implemented in C using X-Windows. A model builder has been implemented in X-Windows and C with a simple graphics and menu-based interface. It provides users with tools for creating equation models of physical systems ([2] [39]). Models created can be stored as subsystems (e.g., the pre-cooler or the pneumatic system) or as mechanisms. The MBD Module and the Diagnostic Controller were both implemented in C using X-Window. The MBD module also calls subroutines from Mathematica to perform symbolic manipulation and calculate partial derivatives in order to generate partial explanations.

7. Discussion and Conclusions

In this paper, we have discussed the design and implementation of an integrated diagnosis system MDS that combines associational and model-based approaches to diagnosis. The design and implementation of the associational module is tailored towards achieving efficiency in routine diagnostic problem solving, and for satisfying the needs of the users that this system was developed for (i.e., aircraft mechanics). The MBD module is developed to achieve completeness and consistency in performing diagnosis on complex systems, and to avoid the brittleness that often occurs in associational approaches, especially when dealing with unusual and novel faults. It employs qualitative causal analysis of the system equations. By analyzing the effects of changes in component parameters on a particular output parameter and comparing the results with the observed deviation of the output parameter, our method can effectively eliminate a large number of candidates during the initial candidate generation phase and produce a more focused result. Coupled with this, we use information theory-based methods (e.g., de Kleer and Williams ([10])) to compute the information *gain* for each possible additional measurement, and, therefore, are able to focus the measurement selection task for further refinement of candidate components.

An important issue addressed in our integrated diagnosis framework is how to use the "deeper" knowledge from system models to improve the completeness and consistency of the "shallow" (or associational) knowledge base. Our hybrid scheme develops a methodology for automated knowledge refinement. The MBD module is used to identify incompleteness and inconsistencies in the associational module, and through the diagnostic controller perform knowledge refinement to update the associational knowledge base.

There are similarities as well as differences between MDS and other existing integrated approaches to diagnosis (e.g., [14] [18] [27] [30] [31] [36]). Major differences include: (i) the type of individual modules used ([18] [30] [36]), (ii) how the modules interact ([14]), and (iii) the amount and type of information exchange among the modules ([27] [33]). Perhaps our system is most similar to the systems by Portinale [27] and Someren ([33]). In all three systems, the associational component is designed and used to "speed up" the diagnostic process and the Model-Based Reasoning (MBR) module is used for completeness. However, the MBR components in their system are not used to help improve its associational component (in their case, a case-based reasoning module) as our system does. Our knowledge acquisition method also differs from many existing approaches. Instead of deriving new knowledge and rules from databases using induction methods ([20] [28] [37]), our approach extracts knowledge from the model-based subsystem by analyzing new diagnostic cases, either automatically (when new rules are added), or semi-automatically (when it guides the human expert to modify existing rules and tests). Currently, we are working on the premise that the existing model-based diagnosis module is accurate and complete. For complex systems, this may not always be the case. In the future, we will be looking into developing techniques where unusual diagnostic cases may be employed to refine both model-based and associational modules. Updating of the system model will necessarily be performed with the help of human experts.

Bibiography

[1] G. Biswas and T.S. Anand. "Using the Dempster Shafer scheme in a mixed-initiative expert system shell", *Uncertainty in Artificial Intelligence*, L. Kand, T.S. Levitt, and J. Lenamer (Editors), Elsevier Science, pp. 223-239, 1989.

[2] G. Biswas and X. Yu, "A Formal Modeling Scheme for Continuous Systems: Focus on Diagnosis", *Proceedings of IJCAI*, Chambery, France, pp. 1474-1479, 1993.

[3] G. Biswas, R. Kapadia, X. W. Yu, "Combined Qualitative- Quantitative Steady State Diagnosis of Continuous-valued Systems", *IEEE Transactions on Systems, Man, and Cybernetics*, July 1997.

[4] T. Bylander and B. Chandrasekaran, "Generic tasks for knowledge-based reasoning: the 'right' level of abstraction for knowledge acquisition", *International Journal of Man-Machine Studies*, vol. 26, pp. 231-243, 1987.

[5] R. Canton, F. Pipitone, W. Lander, and M. Marrone. "Model-based probabilistic reasoning for electronics troubleshooting", *Proceedings of 8th IJCAI*, pp. 207-211, 1983.

[6] S. Craw and R. Boswell, "Representing Problem-Solving for Knowledge refinement", *Proceedings of the Sixteenth National Conference on Artificial Intelligence Eleventh Innovative Applications of AI Conference,* 1999.

[7] P. Dague, O. Raiman, and P. Deves. "Troubleshooting: when modeling is the difficulty", Proceedings of 6th National Conference on Artificial Intelligence, pp. 600-605, Seattle, WA, August, 1987.

[8] R. Davis and W.C. Hamscher, "Model-Based Reasoning: Troubleshooting", in *Exploring Artificial Intelligence: Survey Talks from the National Conferences on AI*, H.E. Shrobe, ed., pp. 297-346, Morgan Kaufmann, San Mateo, CA, 1988.

[9] R. Davis. "Form and Content in Model Based Reasoning", *Proceedings of Workshop on Model Based Reasoning, IJCAI-89*, Detroit, MI, pp. 11-33, 1989.

[10] J. de Kleer and B.C. Williams. "Diagnosing multiple faults", *Artificial Intelligence*, 32, pp. 97-130, 1987.

[11] J. deKleer, "Focusing on Probable Diagnoses", *Proceedings of the 9th AAAI*, pp. 842-848, 1991.

[12] D. Dvorak and B. Kuipers. "Model-based monitoring of dynamic systems", *Proceedings of 11th IJCAI*, pp. 1238-1243, 1989.

[13] B. Falkenhainer and K. Forbus, "Compositional Modeling: Finding the right model for the job", *AI Journal*, vol. 51, pp. 95-143, 1991.

[14] P.K. Fink and J.C. Lusth, "Expert Systems and Diagnostic Expertise in the Mechanical and Electrical Domains", *IEEE Trans. on Systems, Man, and Cybernetics*, vol. SMC-17, pp. 340-349, 1987.

[15] M. Gallanti, M. Roncato, R. Stefanini, and G. Tornielli. "A diagnostic algorithm based on models at different levels of abstraction", *Proceedings of 11th IJCAI*, pp. 1350-1355, 1989.

[16] B. J. Glass, W.K. Erickson, and K. J. Swanson, "TEXSYS: A Large-scale demonstration of model-based real-time control of a space stations subsystem", *Proceedings of the Seventh Conference on Artificial Intelligence Applications*, pp. 378-384. 1991.

[17] G. Guida and G. Mauri, "Evaluatiing Performance and Quality of Knowledge-Based Systems: Foundation and Methodology", *IEEE Transaction on Knowledge and Data Engineering*, vol. 5, no. 2, pp. 204-224, April, 1993.

[18] Y. Huang and R. Miles, "Using Case-Based Techniques to Enhance Constraint Satisfaction Problem Solving", Applied Artificial Intelligence, an International Journal, vol. 10, no. 4 1996.

[19] F. Lackinger and W. Nejdl, "Diamon: A Model-Based Troubleshooter Based on Qualitative Reasoning", *IEEE Expert*, 8(1):33-40, February 1993.

[20] R. S. Michalski, I. Mozetic, J. Hong, and N. Lavrac, "Multi-Purpose Incremental Learning System AQ15 and its Testing Application to Three Medical Domains", *Proceedings of the fifth National Conference on Artificial Intelligence*, pp. 1041-1045, 1986.

[21] P.J. Mosterman and G. Biswas, "Diagnosis of Continuous Valued Systems in Transient Operating Regions", *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 29, no. 6, pp. 554-565, 1999.

[22] P. M. Murphy and M. J. Pazzani, "Revision of Production System Rule-bases", *Machine Learning: Proceedings of the 11th International Conference*, pp. 199-207, 1994.

[23] D. Ourston and R. J. Mooney, "Theory Refinement Combining Analytical and empirical Methods", *Artificial Intelligence*, Vol. 66, pp. 273-309, 1994

[24] L. Palowitch, "Fault Diagnosis of Process Plants using Causal Models". *Ph.D. Thesis*, Massachussets Inst. of Technology, 1996.

[25] R.S. Patil, "Causal representation of patient illness for electrolyte and acid-based diagnosis", *Technical Report*, no. 267, Massachussetts Institute of Technology, Laboratory for Computer Science, Cambridge, MA, 1981.

[26] Y. Peng and J. A. Reggia. "Plausibility of diagnostic hypothesis: the nature of simplicity", *AAAI-86*, pp. 140-145, 1986.

[27] L. Portinale and P. Torasso, ADAPtER: An Integrated Diagnostic System Combining Case-Based and Abductive Reasoning", *Topics in Case Based Reasoning, Proceedings of the First International Conference on Case-Based Reasoning*, pp. 277-288, 1995.

[28] J. R. Quinlan *C4.5 – Programs for Machine Learning*, Morgan Kaufmann, CA, 1993.

[29] G. W. Rosenwald and C. Liu, "Rule-Based System validation through Automatic Identification of Equivalence Classes", *IEEE Transaction on Knowledge and Data Engineering*, Vol. 9, No. 1, pp. 24-31, 1997.

[30] R.C. Rosenberg, and D.C. Karnopp, *Introduction to Physical System Dynamics*, McGraw-Hill, 1983.

[31] E. H. Shortliffe, *Computer-based medical consultations: MYCIN*, New York, 1976.

[32] M.H. Sqalli and E.C. Feuder, "Diagnosing InterOperability Problems by Enhancing Constraint Satisfaction with case-Based Reasoning", *Ninth International Work on Principles of Diagnosis*, pp. 266-273, 1998.

[33] M. Someren, J. Surma, and P. Torasso, "A Utility-based Approach to Learning in a Mixed Case-based and Model-Based Architecture", *Proceedings of the Second International Conference on Case Based Reasoning*, 1997.

[34] P. Struss and O. Dressler, "Physical Negation: Integrating Fault Models into the General Diagnostic Engine", *Proceedings of IJCAI-89*, pp. 1318-1323, Detroit, MI, 1989.

[35] M. Tallis and Y. Gil, "Designing Scripts to Guide Users in Modifying Knowledge-based Systems", *Proceedings of the Sixteenth National Conference on Artificial Intelligence Eleventh Innovative Applications of AI Conference*, 1999.

[36] C. Tudorie, C. Segal, and E. Pecheanu, "Human Operator Observations Elicitation in a Hybrid (Neuro-Symbolical) Diagnosis", Proceedings of the Ninth International Workshop on Principles of Diagnosis, pp. 282-285. 1998.

[37] S. Tsumoto and H. Tanaka, "Automated Discovery of Medical Expert System Rules from clinical Database on Rough Sets", *Proceedings of conference on Data mining Applications*, pp. 63-69, 1997.

[38] S. Weiss, C. Kulikowsi, and A. Safir. "A model-based consultation system for the long-term management of glaucoma", *Proceedings of the 5th IJCAI*, pp. 826-832, 1977.

[39] X. W. Yu, "Multi-level Reasoning and Diagnosis", *Ph.D. Dissertation*, Computer Science Department, Vanderbilt University, December, 1992.