N94- 34051

Discovering operating modes in telemetry data from the Shuttle Reaction Control System

Stefanos Manganaris* Doug Fisher*
Dept. of Computer Science
Vanderbilt University

Deepak Kulkarni
Artificial Intelligence Research Branch
NASA Ames Research Center

August 2, 1993

ABSTRACT

This paper addresses the problem of detecting and diagnosing faults in physical systems, for which suitable system models are not available. We propose an architecture that integrates the on-line acquisition and exploitation of monitoring and diagnostic knowledge. The focus of the paper is on the component of the architecture that discovers classes of behaviors with similar characteristics by observing a system in operation. We investigate a characterization of behaviors based on best fitting approximation models. An experimental prototype has been implemented to test it. We present preliminary results in diagnosing faults of the Reaction Control System of the Space Shuttle. The merits and limitations of the approach are identified and directions for future work are set.

1 INTRODUCTION

One of the tasks that operators of complex systems perform is monitoring: the detection of abnormal system behavior. The identification of the cause of an abnormality, or fault diagnosis, is a second one. Researchers in Artificial Intelligence have been trying to automate both tasks.

Traditional monitoring and diagnosis systems are rule-based: an "expert" encodes faults and associated symptoms in rules. Sophisticated rule-based expert systems can draw inferences based on time histories of data and operate in real-time. However, expert systems suffer in many ways. Acquiring and expressing the required knowledge in usable rules is a difficult task. Strong assumptions in the rules make detecting and diagnosing novel faults

*This research has been supported in part by a grant from NASA Ames (NCC 2-645).

difficult. Finally, maintaining a set of rules is expensive and time consuming.

The model-based approach to monitoring and diagnosis attempts to overcome some of these problems by developing inference engines that can reason using models of systems. Expected system behavior is predicted and then compared with the observed one. Diagnostic inferences are guided by any discrepancies. Model-based systems can handle multiple and novel faults, as long as the model chosen is at the right level of abstraction to explain the faults.

The goal of our research is a general framework for monitoring and fault diagnosis that performs well even in cases where we have neither the required expertise to build a rule-based system, nor sufficiently detailed and otherwise suitable models for a model-based system. By observing a system in operation over time, we attempt to discover patterns in its behavior. Machine learning techniques are used to induce and associate behavior patterns and the conditions under which they were observed. In parallel with knowledge acquisition, monitoring and diagnosis are performed based on the knowledge base built so far. We have experimented with aspects of this general approach using data from the Space Shuttle Reaction Control System (RCS).

2 A TRAINABLE MONITORING AND DIAGNOSIS SYSTEM FOR THE RCS

Operators often detect and diagnose faults by observing how quantities of a system change over time. With experience, they identify patterns over the macroscopic, qualitative, features of behavior and their associated normal or abnormal operating conditions. Macroscopic features refer to the shape of the plot of a quantity over time. Simple features are the average value, the average slope, the average noise level, the period of oscillation, and the frequency spectrum.

Given a physical system, we are interested in discovering the following by observing its behavior over time:

- The *operating modes* or *states* of the system and their *behavior characteristics*. A different underlying model governs the behavior of the system in each state. We are not interested in necessarily discovering that model—some characteristics of the behavior implied are sufficient when they differentiate states.
- The *transitions* from one state to another and the *conditions* associated with them. Conditions may refer to operator commands, system talk-back, quantity values and thresholds, or rule-based combinations of these.

The Trainable Monitoring and Diagnosis System (TMDS) integrates this discovery process with monitoring and diagnosis. A high level description of its architecture is shown in Fig. 1. A data acquisition component is the interface with the monitored system. It preprocesses the monitored signals and sends the results to a knowledge acquisition component and a monitoring & diagnosis component; a knowledge-base is maintained by the first and is

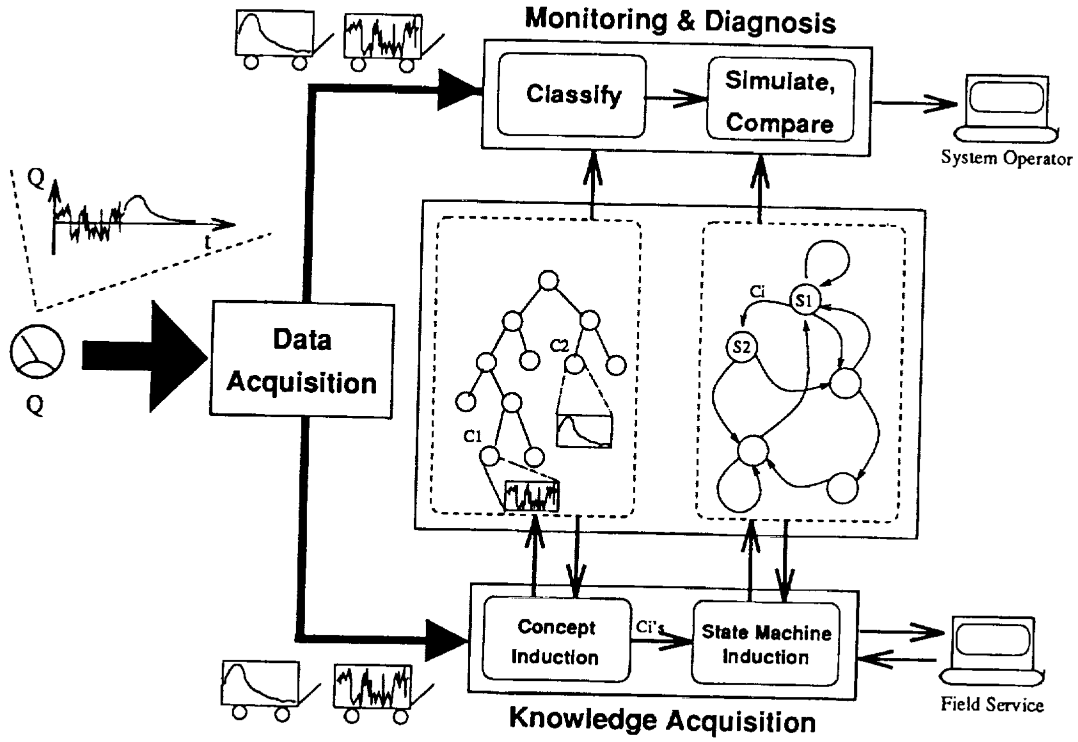


Figure 1: High Level Architecture of the TMDS

used by the second. We demonstrate the current implementation of TMDS with data from the Reaction Control System (RCS) of the Space Shuttle. We experimentally show that behavioral classes discovered by TMDS correspond to a variety of normal and abnormal operating modes of the RCS. We will interleave discussion of general aspects of the TMDS with specific examples in the RCS domain.

The RCS of the Space Shuttle provides propulsive forces from a set of jets to control the motion of the Shuttle (pitch, yaw, roll). It replaces the aerodynamic surfaces, which become ineffective in the upper atmosphere. The RCS is located in three different areas of the orbiter. The forward RCS module is in the fuselage nose area. The aft RCS modules are located in the right and left RCS pods, which are attached to the aft fuselage. Each RCS has two subsystems. One for each propellant: Oxidizer (OX) and Fuel (FU). The OX and FU subsystems are very similar in construction. Each consists of a Helium system, a propellant system (OX or FU), cross-feed and interconnect capabilities, and a jet thruster system. The helium system is used to pressurize the propellant and drive it to the jets. It consists of a Helium tank storing helium under high pressure, two legs in parallel, of two pressure regulators in series, each controlled by a valve. The propellant system consists of a propellant tank and an isolation valve at its output. The jet system consists of a manifold (pipes and valves), and jets. A FU and an OX pipe goes to each jet. A jet fires when OX and FU are allowed contact in its chamber.

Many quantities of the system are transmitted back to earth via a telemetry link. Each RCS has two He pressure sensors at the He tank, one temperature sensor at the He tank, one pressure sensor for the ullage pressure in the propellant tank, one temperature sensor at the propellant tank, and one pressure sensor at the output of the propellant tank. In addition, every valve's position (talk-back) and every command affecting a valve is also transmitted. More information about the system can be found in the RCS training manual.[3]

3 DATA ACQUISITION IN THE TMDS

Signals monitored can be classified as either analog or digital, depending on whether they exist at every instant of time or not. The data acquisition module of TMDS samples any analog signals and digitizes them.

Further processing partitions the continuous stream of data points into intervals of homogeneous behavior characteristics. After an interval has been identified, the behavior of all signals in that duration is characterized. The result is a stream of *behavior summaries*, one for each interval of system-wide homogeneous behavior. A behavior summary contains characterizations for all monitored signals.

The methods used to characterize signals depend on their types. Signals may be deterministic or random. Deterministic signals can be precisely described by a function of time and are thus predictable; random signals cannot. Deterministic signals may be periodic or aperiodic (transient). Random signals can only be described statistically; we can use approximation models and other methods for deterministic signals.

Our work with the RCS system illustrates how we fleshed out these general issues for a particular system. In this case, TMDS is explicitly instructed how to partition the continuous stream of data points into homogeneous intervals by utilizing commands and talk-back present in the telemetry stream. Commands and verification issued through talk-back indicate when operating modes will change. For each interval discovered in this manner, twelve quantities were monitored in addition to the discrete commands and talk-back for detecting configuration changes. Behavior summaries in this experiment consist of the two parameters of the best fitting *linear* approximation models (i.e., slope and intercept) and the squares of the correlation coefficient for *each* of the twelve quantities, which is a measure of the approximation's 'fitness'. RCS behavior in each interval is thus characterized by thirty-six (3×12) numerical attributes. Linear approximation models were used because they were simple and sufficiently informative for the RCS signals. Although most RCS quantities do not behave linearly, linear approximations were deemed satisfactory for short intervals of time. In RCS operation short behavior summaries are typical.

Returning to general issues, the characteristics of a signal vary depending on the operating mode of the system. The data acquisition module partitions the continuous behavior into homogeneous intervals. The TMDS currently relies on a prespecified subset of the monitored signals to perform this partitioning. Signals that are included in the subset are known to be indicators of operating mode changes. This was motivated by our application, where, for example, commands to and talk-back from the system are monitored, and they were deemed sufficiently informative. In other applications, more sophisticated techniques may have to be used. For example, partitioning may be based on the detection of abrupt changes in the spectral behavior of a signal,[2] or of abrupt changes in its mean amplitude level.[1] For deterministic signals, for which approximate models are known, the Minimum Message Length principle may be used to decide when a break would yield a "better" description.[15]

4 DISCOVERING OPERATING MODES

We envision a TMDS architecture that exploits a two-part knowledge-base. The first contains knowledge about *system states*. Each state is associated with a class of behaviors of similar characteristics. The second contains knowledge about *transitions* between states and associated conditions. Both states and transitions are annotated. Annotations indicate whether a behavior or a transition is normal or not, and its cause.

Two machine learning components discover and maintain most of the information in the knowledge base. The first one processes behavior summaries as they are generated by the data acquisition module. It discovers classes of similar behaviors. The current TMDS implementation uses COBWEB/3,[10] a portable implementation of COBWEB[6] that handles numeric attributes. The COBWEB system implements an algorithm for data clustering and incremental concept formation. It can be used to organize objects in classes described by a collection of attributes and their values. COBWEB's approach to classification and learning is known as conceptual clustering.[11,8] Learning in COBWEB is unsupervised: no tutor is necessary to provide feedback. We use an unsupervised, conceptual clustering, learning paradigm, because behavior summaries are unlabeled. Labels that correspond to the particular underlying system states (i.e., operating modes) responsible for a behavior are to be *discovered*. COBWEB organizes concepts in a hierarchy, that is, a partial order according to generality. Each concept node of the hierarchy describes a class of behaviors in terms of the same attributes used to characterize behavior summaries. Learning is incremental: COBWEB processes instances one at a time.

The hierarchy evolves by selecting and applying operators that incorporate each instance into the hierarchy. An operator is applied at each level of the hierarchy, starting at the root. At each cluster the algorithm maintains a probability model for the values of each attribute. This information is used by an evaluation function (*category utility*) to select the operator to apply at a particular level for an instance. Category utility prefers operators that result in hierarchies that maximize intra-class similarities and inter-class differences. In particular, for numeric data, COBWEB/3 defines category utility as

$$\frac{\sum_{k=1}^{K} P(C_k) \sum_i 1/\sigma_{ik} - \sum_i 1/\sigma_i}{4K\sqrt{\pi}}$$

where K is the number of clusters, C_k are the individual clusters, σ_{ik} is the standard deviation of attribute i in cluster k, and σ_i is the population-wide standard deviation of attribute

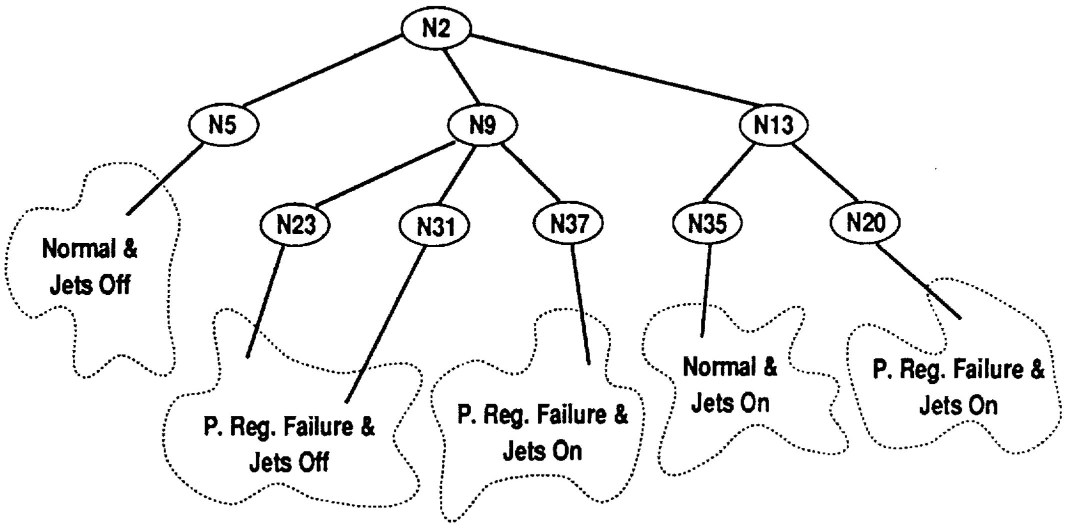
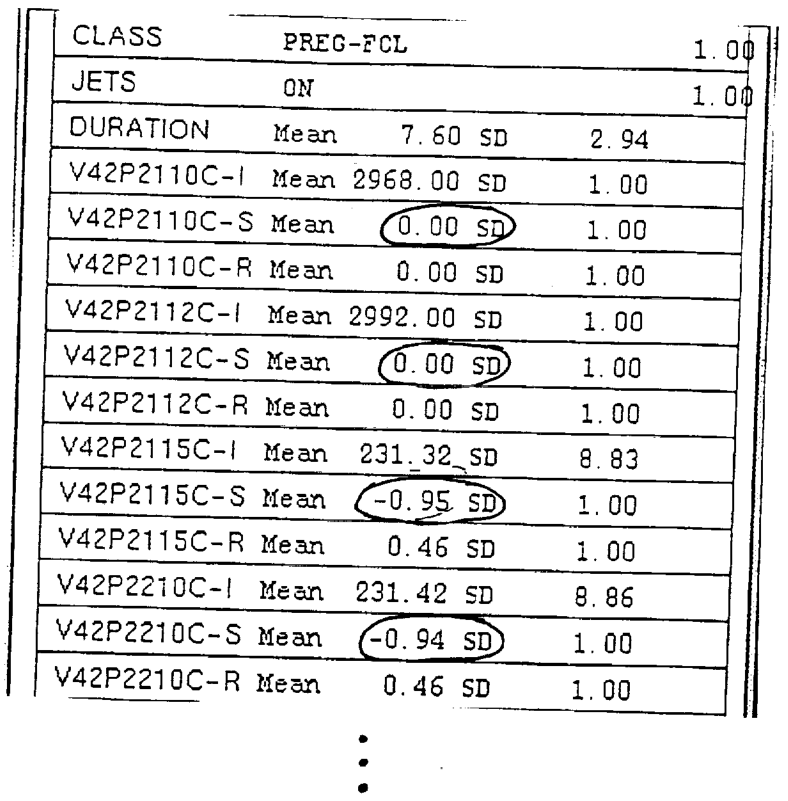

N2
N5
N9
N13
N23
N31
N37
N35
N20
Normal & Jets Off
P. Reg. Failure & Jets Off
P. Reg. Failure & Jets On
Normal & Jets On
P. Reg. Failure & Jets On

Figure 2: A Hierarchy of Behavior Classes Corresponding to RCS's Operating Modes

i. Intuitively, category utility favors clusters that most reduce the standard deviation over the numeric attributes.

Looking again to the RCS domain, Noisy telemetry data, from several minutes of RCS operation and under various conditions, were processed as described in Section 3 to generate fifty-six behavior summaries. Figure 2 shows a hierarchy of classes formed by COBWEB/3 over these 56 behaviors. The leaf nodes correspond to individual behavior summaries. Clusters were manually labeled according to the configuration of the RCS in the corresponding interval. The operating modes of the RCS we have studied can be roughly classified into four categories: normal, with the jets on or off, and abnormal, with a failed pressure regulator and the jets either on or off. The labels were obviously not used for inducing the classes. One can readily notice that the classes identified are meaningful; for example, node N9 corresponds to the class of behaviors when a pressure regulator has failed, and nodes N35 and N5 correspond to the class of normal behaviors. Figure 3 gives an example cluster definition for a class dominated completely by instances of jets-on, pressure regulator failed-closed behavior. The salient aspects of this definition are circled: Helium ullage in the oxidizer (or fuel) tank and the oxidizer out pressure are dropping, and Helium pressure in the Helium tank is steady.

We envision a second machine learning component that induces a state-machine. Its input is the sequence of states (i.e., concepts) found through categorization with the clustering hierarchy. By observing the state transitions and the changes in operating conditions, a finite automaton is constructed showing possible transitions and conditions associated with them. Related work includes Nordhausen and Langley[13] and Dietterich[5] on sequence induction, and Mitchell et al.[12] in search control learning. This component is not yet operational.



CLASS	PREG-FCL				1.00
JETS	ON				1.00
DURATION	Mean	7.60	SD	2.94	
V42P2110C-I	Mean	2968.00	SD	1.00	
V42P2110C-S	Mean	0.00	SD	1.00	
V42P2110C-R	Mean	0.00	SD	1.00	
V42P2112C-I	Mean	2992.00	SD	1.00	
V42P2112C-S	Mean	0.00	SD	1.00	
V42P2112C-R	Mean	0.00	SD	1.00	
V42P2115C-I	Mean	231.32	SD	8.83	
V42P2115C-S	Mean	-0.95	SD	1.00	
V42P2115C-R	Mean	0.46	SD	1.00	
V42P2210C-I	Mean	231.42	SD	8.86	
V42P2210C-S	Mean	-0.94	SD	1.00	
V42P2210C-R	Mean	0.46	SD	1.00	

Figure 3: A cluster definition of jets-on, failed-closed behavior.

5 MONITORING AND DIAGNOSIS WITH THE TMDS

In parallel with the continuous maintenance of the knowledge base, TMDS monitors and diagnoses faults. Behavior summaries, generated by the data acquisition component, are first classified using the classifier constructed by COBWEB. If a behavior summary is very different from all known classes, COBWEB forms a new singleton class. In this way, novel behaviors are detected, and the system operator is warned, even when TMDS has not been trained on them. When a behavior summary is classified to a known faulty class (previously identified by an operator), TMDS can diagnose the fault using information from the annotation of the class. When the classification is to a known normal class, that class is compared to the one predicted by the state machine from the last known state. If the transition at hand is novel, and thus unexpected, or is known to be associated with a malfunction, an appropriate warning may be generated. As we implied earlier, state machine induction is not yet implemented, nor are the diagnostic methods associated with these structures.

However, to test the accuracy of the acquired knowledge from the current system, we ran experiments, where we predicted the class for behaviors TMDS was not trained on. We focused on behaviors where a pressure regulator has failed closed and on nominal behaviors, under different operating conditions: when different jets fire, for different periods, under different temperatures, etc. For each experiment we trained TMDS on fifty behavior summaries, presented in random order, and tested prediction accuracy on six behavior summaries, which were not used for training. The accuracy achieved was 85.5%, averaged over 30 runs. Given the limited amount of data for training, this level of accuracy is promising. The effects of misclassifications to the overall operation of TMDS are yet to be examined.

6 RELATED WORK AND SUMMARY

This work started at NASA Ames as an alternative approach to monitoring and diagnosis of the RCS. Related work at Ames focuses on a mixed quantitative and qualitative model-based framework for diagnosis over time.[16] Other applications of machine learning for inducing diagnostic knowledge are Pearce's AQR[14] and Lee's and Dvorak's DYNALEARN.[9] Both approaches require a model to systematically simulate system behavior. Pearce focuses on static systems with single faults, and, in a propositional framework, induces a rule that covers a set of behavior examples of the same failure. A single rule is induced for each failure in turn. DYNALEARN extends the ideas of AQR to dynamic systems. It induces a decision tree, which is used to predict suitable starting points in model-based tracking of a time-dependent system.

A simple, yet promising approach to combining induction with model-based reasoning was initiated this Summer by the authors, Peter Robinson of NASA Ames, and Julio Ortega of Vanderbilt University. Our perspective is that models in general, and the RCS models at NASA Ames in particular, can be viewed as superb 'feature extractors' for raw (RCS telemetry) data. In particular, initial experiments used recommendations and intermediate state variables computed by the models to augment the raw (simulated) telemetry data from the RCS system. Decision tree induction was then performed over the augmented data instances. Our goal is to identify those portions of the behavior space in which the models seem to be performing well, which is indicated by the use of model recommendations and state variables in portions of an induced decision tree, and those portions of the behavior space in which the models are not performing well, which are indicated by an absense of model variables/recommendations in other portions of the tree. In the long term, we envision a model construction aid that focuses the attention of the model builder on those portions of the behavior space in which the model can be improved.

Smyth and Mellstrom[17] develop a classifier that can reject classes it was not trained on, using a special class of stochastic models and a supervised learner. This feature is fundamental for the classifier in the TMDS. However, we rely on COBWEB, an *unsupervised* learner, in discovering novel classes.

Nordhausen and Langley[13] address the general problem of empirical discovery. Their

IDS system processes a sequence of temporally ordered qualitative descriptions of states and forms a taxonomy of states, discovers relations between pairs of states (e.g., transitions and conditions on those transitions), and numeric relations of quantities. We anticipate that several ideas of IDS can be used in TMDS.

Previous applications of clustering in diagnosis include.[4] Carnes and Fisher cluster individual snapshots of system behavior to learn fault modes for design (in particular, sensor placement) and diagnosis. Static quantitative models are used to generate training data. Training instances are not temporally related.

In this paper, we have shown preliminary results towards a trainable architecture for monitoring and fault diagnosis. A partial implementation of the TMDS architecture was used successfully to discover the characteristics of the behavior of the Reaction Control System under normal and abnormal operating conditions. Key traits of the TMDS system are the following:

- The TMDS is trainable and can be adapted to monitor and diagnose any system. The characteristics of a system's behavior are discovered by observing it in operation. Novel behaviors are identified as such, and knowledge about their cause is elicited from an operator. Knowledge acquisition is driven by the behavior characteristics of the monitored system—*not* by an "expert". TMDS learns from its experiences. Training continues in parallel with monitoring and diagnosis for reinforcement and refinment of its dynamic knowledge-base.
- Monitoring and diagnostic knowledge is in a compiled form, suitable for real-time performance. Knowledge acquisition is guided by TMDS's discoveries, but is a separable task and can be run off-line.
- Diagnosis is based on system behavior over time, not isolated snapshots. Temporal aspects, which often carry crucial diagnostic information, can be captured in state machines and are used in diagnosis.
- TMDS is expected to be robust with respect to sensor failures. A failed sensor results in either erroneous or no information at all about a signal, which is one focus of cited work by Carnes and Fisher. It has been demonstrated that COBWEB's classifier is robust with respect to noisy or missing attributes.[7] Even with some erroneous values COBWEB can still find a good matching class based on the remaining ones.

Although the proposed approach could be the only choice for new or very complex systems, we believe it could also be the right choice for systems for which a rule- or model-based approach may be used. A rule-based system is bound to be static: its knowledge base can only be modified at great expense. A model-based system's understanding of faults is limited to those that can exist in the model's approximation of reality. A trainable monitoring and diagnosis system would, given sufficient training, be able to handle any fault, subject only to the limitations of the expressiveness of its representation of behaviors.

7 ACKNOWLEDGEMENTS

The first two authors thank the AI branch of the NASA Ames Research Center and Peter Friedland for sponsoring visits at Ames during the past two summers. Special thanks to Pat Langley for contributing important ideas, and to Kevin Thompson for his help with COBWEB/3. Many thanks to Peter Robinson and Matt Barry for providing the telemetry data of the RCS, and to Peter Cheesman and Wray Buntine for suggesting techniques to address some of the issues introduced.

8 REFERENCES

[1] Michele Basseville and Albert Benveniste. Design and comparative study of some sequential jump detection algorithms for digital signals. *IEEE Trans. Acous., Speech, Signal Processing*, ASSP-31:521–534, June 1983.

[2] Michele Basseville and Albert Benveniste. Sequential detection of abrupt changes in spectral characteristics of digital systems. *IEEE Transactions on Information Theory*, IT-29(5):709–724, September 1983.

[3] Richard Bush. *Reaction Control System Training Manual.* NASA, Mission Operations Directorate, Training Division, Flight Training Branch, April 1987. RCS 2102.

[4] Ray Carnes and Douglas H. Fisher. Learning fault modes for design and diagnosis though clustering. Technical Report CS-91-06, Vanderbilt University, Computer Science Dept., Nashville, TN, 1991.

[5] Thomas G. Dietterich. Learning about systems that contain state variables. In *Proc. of the Fourth National Conf. on Artificial Intelligence (AAAI-84)*, pages 96–100, Austin, TX, August 21-26 1984.

[6] Douglas H. Fisher. Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, 2:139–172, 1987.

[7] Douglas H. Fisher. Noise-tolerant conceptual clustering. In *Proc. of the Eleventh Intl. Joint Conf. on Artificial Intelligence*, pages 825-830, 1989.

[8] Douglas H. Fisher and Pat Langley. Methods of conceptual clustering and their relation to numerical taxonomy. In W. Gale, editor, *Artificial Intelligence and Statistics*. Addison Wesley, 1986.

[9] W. Lee and D. Dvorak. Learning diagnostic knowledge for continuous-variable dynamic systems. In *Working Notes of the Model Based Reasoning Workshop*, pages 129–133, Detroit, MI, 1989.

[10] Kathleen McKusick and Kevin Thompson. COBWEB/3: A portable implementation. Technical Report FIA-90-6-18-2, NASA Ames Research Center, Artificial Intelligence Research Branch, CA, 94035, June 1990.

[11] R. S. Michalski and R. Stepp. Learning from observation: Conceptual clustering. In R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, editors, *Machine Learning: An Artificial Intelligence Approach.* Morgan Kaufmann, San Mateo, CA, 1983.

[12] T. M. Mitchell, P. E. Utgoff, and R. Banerji. Learning problem solving heuristics by experimentation. In R. S. Michalski, T. M. Mitchell, and J. G. Carbonell, editors, *Machine Learning: An Artificial Intelligence Approach.* Morgan Kaufmann, San Mateo, CA, 1983.

[13] Bernd Nordhausen and Pat Langley. An integrated framework for empirical discovery. *Machine Learning*, 1992. (To appear in the special discovery issue).

[14] D. A. Pearce. The induction of fault diagnosis systems from qualitative models. In *Proc. of the Seventh National Conf. on Artificial Intelligence (AAAI-88)*, pages 353-357, Saint Paul, MN, August 21-26 1988.

[15] R. Bharat Rao and Stephen C-Y. Lu. Learning engineering models with the minimum description length pinciple. In *Proc. of the Tenth National Conf. on Artificial Intelligence*, 1992.

[16] Peter Robinson. Automated fault diagnosis algorithms for the reaction control system of the space shuttle. Technical Report FIA-93-05, NASA Ames Research Center, 1993. (Publication p4).

[17] Padhraic Smyth and Jeff Mellstrom. Detecting novel classes with applications to fault diagnosis. In Derek Sleeman and Peter Edwards, editors, *Proc. Ninth Intl. Conf. Machine Learning*, pages 416–425, 1992.