Appears in *Working Notes of the IJCAI-95 Workshop on Engineering Problems for Qualitative Reasoning,* Monreal, Quebec, Canada, August 1995

Multiple-Fault Diagnosis Using General Qualitative Models with Behavioral Modes

Siddarth Subramanian, Raymond J. Mooney
Dept. of Computer Sciences
University of Texas at Austin
Austin, Texas 78712
siddarth@cs.utexas.edu, mooney@cs.utexas.edu

March 30, 1995

Abstract

This paper describes an approach to performing diagnosis on complex dynamic systems described by qualitative differential equations. An implemented system QDOCS is described that performs multiple-fault diagnosis on systems modeled with QSIM. Constraint satisfaction techniques are used to determine sets of component modes that are inconsistent with observed qualitative behaviors. Most probable diagnoses are computed from the resulting conflicts sets. The utility of the system is illustrated by accurately and efficiently diagnosing randomly generated faults in a model of the Reaction Control System of the space shuttle.

1 Introduction

One class of engineering problems that must be addressed by qualitative reasoning if it is to be useful in real applications is that of diagnosis. The approach to diagnosis described in this paper uses QSIM (Kuipers, 1994) as the modelling language and applies a very general diagnostic technique to these models. Previous approaches to diagnosing faults in QSIM models have been limited in scope and have been unable to work with fault modes (Ng, 1990; Lackinger and Nejdl, 1991) or have made a single-fault assumption (Dvorak, 1992). Most previous work on model-based diagnosis (Reiter, 1987; de Kleer and Williams, 1987) has concentrated on static systems and is generally insufficient to diagnose continuous dynamic systems. Few of the other approaches to diagnosis of continuous systems (Oyeleye et al., 1990; Dague et al., 1991; Guckenbiehl and Schafer-Richter, 1990) have made use of a general modelling language such as that provided by QSIM or used any of the general diagnostic formalisms introduced in by Reiter or DeKleer.

```
(M+ amount level)
(M+ level pressure)
(mode (drain-mode normal) (M+ pressure outflow))
(mode (drain-mode blocked) (ZERO-STD outflow))
(ADD netflow outflow inflow)
(D/DT amount netflow)
(CONSTANT inflow if*)
```

Figure 1: The Combined Bathtub Model

This work is an attempt to build a general, multiple-fault diagnosis system which uses behavioral modes with *a priori* probabilities. The diagnostic architecture is similar to SHERLOCK (de Kleer and Williams, 1989) and the algorithm builds on INC-DIAGNOSE (Ng, 1990). The system uses a general constraint-satisfaction technique to detect faults and trace dependencies in order to generate conflicts and diagnoses. A QSIM-based simulation component is used to verify hypotheses and detect additional inconsistencies. The implemented system, QDOCS (Qualitative Diagnosis Of Continuous Systems), is powerful enough to accurately diagnose a number of different faults in the Space Shuttle's Reaction Control System.

The rest of this paper is organized as follows. In section 2, a simple example is introduced to motivate the work. QDOCS's algorithm is presented in section 3 and its application illustrated on this example. Section 4 reports on an experiment used to test QDOCS on a realistic problem. Section 5 contains a discussion of some of the limitations of the approach and includes some future research directions. The paper ends with a discussion of related work in section 6 and our conclusions in section 7.

2 An Example

An example used to illustrate the algorithm consists of a simple bathtub with a drain. It is assumed that the bathtub is monitored by sensors measuring the amount of water in the tub and the flow rate of the water through the drain. Some of the faults that can be posited about this system include a blocked drain, leaks in the tank, and sensors stuck at various levels.

This system is described using a qualitative differential equation or a QDE. A QDE is a set of constraints, each of which describes the relationship between two or more variables. For instance, an `M+` relation is said to exist between two variables if one is a monotonically increasing function of the other. So, in our normal bathtub model, there is an `M+` relation between the amount and the level of water in the bathtub and also between the the level and pressure, and the pressure and outflow rate. However, in a model of a blocked bathtub, the outflow rate is zero, and it is described by the constraint `ZERO-STD`.

The use of discrete mode variables in QSIM allows us to combine normal and faulty models of a system into a single description as shown in Figure 1. [1] Here, the variable drain-mode

[1] The complete model also has mode variables and fault modes for the level and flow sensors and the inlet

```
(defcomponents bathtub
   (drain drain-mode (normal 0.89) (blocked 0.1) (unknown 0.01))
   (levelsensor levelsensor-mode (normal 0.79) (stuck-at-0 0.1)
                (stuck-at-top 0.1) (unknown 0.01))
   (flowsensor flowsensor-mode (normal 0.79) (stuck-high 0.1)
               (stuck-at-0 0.1) (unknown 0.01))
   (inletvalve inletvalve-mode (normal 0.79) (stuck-closed 0.1)
               (unknown 0.01)))
```

Figure 2: The Bathtub Component Structure

takes on the possible values of *normal*, *blocked*, or *unknown* and the constraints shown above correspond to the two known modes of the bathtub's behavior.

For the purposes of diagnosis, these mode variables can then be associated with components of the system and their different values with behavioral modes of the component. Each of these behavioral modes has an *a priori* probability specified by the model-builder. The component structure used to represent the bathtub is given in Figure 2. Here, each entry consists of the component name (e.g., `drain`), the mode variable (`drain-mode`) and a list of behavioral modes with their *a priori* probabilities (`(normal 0.89) (blocked 0.1) (unknown 0.01)`).

The input to the diagnostic algorithm consists of a behavior, which is a sequence of qualitative values for a subset of the variables corresponding to sensor readings. The output of the algorithm is an assignment of values to the mode variables such that the resulting model is consistent with the observed behavior. A model is considered to be consistent with the behavior if the behavior corresponds to a QSIM simulation of the model.

As an example, suppose QDOCS is given the following single set of sensor readings from a behavior of the bathtub: `(level-sensed (0 top)), (outflow-sensed 0)` (i.e., the level sensed is somewhere between 0 and top and the outflow sensed is 0). This is clearly inconsistent with the normal model of the system which would predict a flow through the drain. Some of the valid diagnoses for this behavior include `[(drain-mode blocked)]`, `[(flowsensor-mode stuck-at-0)]` and `[(drain-mode blocked) (flowsensor-mode stuck-at-0)]`.

The above example motivates an approach of applying QSIM's constraint satisfaction techniques to detect inconsistencies between the sensor readings and the model. However, since the systems under study are dynamic systems that maintain temporal consistency, satisfying the constraints for a given set of sensor readings does not guarantee that the sequence of readings is consistent. The approach we discuss in the next section includes using the continuity checking of QSIM to check this temporal consistency.

---

valve.

3 QDOCS's Diagnostic Approach

Using the standard approach of consistency-based diagnosis, we first determine conflict sets, which are assignments of values to mode variables that are inconsistent with the observed behavior. These conflicts are then used to construct diagnoses.

3.1 Determining Conflict Sets

Most diagnostic systems like GDE (de Kleer and Williams, 1987) use simple constraint propagation to determine conflict sets. However, QSIM requires a more complete constraint satisfaction algorithm since a qualitative constraint typically does not entail a unique value for a remaining variable when all its other variables have been assigned. An earlier attempt to use QSIM to track dependencies for diagnosis (Ng, 1990) only used a simple propagator. Since the propagator alone is not complete, Ng's program, INC-DIAGNOSE is not guaranteed to detect all inconsistencies.

QSIM takes a set of initial qualitative values for some or all of the variables of a model and produces a representation of all the possible behaviors of the system. The inputs to QSIM are 1) a qualitative differential equation (QDE) represented as a set of variables and constraints between them, and 2) an initial state represented by qualitative magnitudes and directions of change for some of these variables. QSIM first completes the state by solving the constraint satisfaction problem (CSP) defined by the initial set of values and the QDE. For each of the completed states satisfying the constraints, QSIM finds qualitative states that are possible successors and uses constraint satisfaction to determine which of these are consistent. The process of finding successors to states and filtering on constraints continues as QSIM builds a tree of states called a behavior tree.

QSIM's approach to solving the CSP is to

1. establish node consistency by ensuring that each constraint is satisfied by the possible values of the variables it acts upon, and

2. using Waltz filtering (Waltz, 1975) to establish arc consistency, by propagating the results of step 1 to other variables and constraints, and, finally,

3. using backtracking to assign values to variables.

The Waltz filtering step is performed incrementally and at each point selects the most restrictive constraint (i.e., the one most likely to fail) to process and propagates its effect on the rest of the network. These heuristics help discover the inconsistency of states quickly and avoid unnecessary search.

A model is inconsistent with a given sequence of sensor readings if there is no corresponding behavior in the behavior tree. There are two possible ways an inconsistency can arise: 1) a particular set of readings may be incompatible with the QDE, or 2) all the sets of readings may be compatible with the QDE but the sequence may not correspond to any particular behavior in a QSIM behavior tree. QDOCS's approach is to first test for consistency between

individual sets of readings and the QDE and then, test to see if the model fits the sequence, i.e., if the sequence of readings corresponds to a behavior generated by QSIM.

QDOCS modifies QSIM's constraint satisfier to keep track of mode-variables whose values played a role in reducing the set of possible values for a variable. Each variable and constraint is associated with an initially empty dependency set of mode variables. Whenever a constraint causes a variable's set of possible values to decrease, the dependency set of the variable is updated with the union of its old dependency set, the dependency set associated with the constraint, and the mode variable, if any, that is associated with the constraint. When a variable reduces the set of possible tuples associated with the constraint, the constraint's dependency set is similarly updated with the union. When a variable is left with no possible values, its current dependency set is returned as a conflict set.

The heuristic of first filtering on the most restrictive constraints helps reduce the size of conflict sets but it does not guarantee minimal conflicts. The most restrictive constraints are those with the least number of initial possible tuples, and therefore are more likely to lead to an inconsistency. In order to fully keep track of the necessary (as opposed to just sufficient) conditions leading to the inconsistency of the state, the constraint satisfaction algorithm would have to keep track of dependencies for each value of each variable that was eliminated. This is clearly combinatorially explosive and therefore, the simpler algorithm is used despite the fact that the conflict sets obtained are not guaranteed minimal. Of course, the constraint propagation techniques used by most model-based diagnosis systems are also not guaranteed to generate minimal conflicts (Forbus and de Kleer, 1993).

For the second part of the algorithm, QDOCS must track a QSIM simulation and match all possible successors at each stage of the simulation with given sensor readings. Whenever the states corresponding to a set of sensor readings fail to have any successors matching the next set of sensor readings, an inconsistency is noted. Dependencies for variables are propagated as above while completing individual successor states and also from states to successor states. Since continuity limits the set of possible values for a variable which is a successor of another state, each variable inherits all the dependencies of its parent.

3.2 Constructing Diagnoses

The above process is used to compute a single conflict set for the set of constraints associated with the current set of behavioral modes. The diagnostic algorithm must then query the constraint satisfier with different sets of values for mode variables to obtain different conflict sets. The approach adopted is similar to SHERLOCK in that candidate generation is focussed on the most probable diagnoses.

Initially, the default candidate (all components initialized to *normal*) is the one used to query the constraint satisfier. The algorithm then tries to complete states for each of the sets of sensor readings in turn, and whenever the constraint satisfier signals a contradiction, a conflict set is generated. The diagnostic reasoner then uses this conflict set and the current candidate to generate new candidates by considering all the other behavioral modes of each component in the conflict set in addition to the behavioral modes of the current candidate. An agenda is

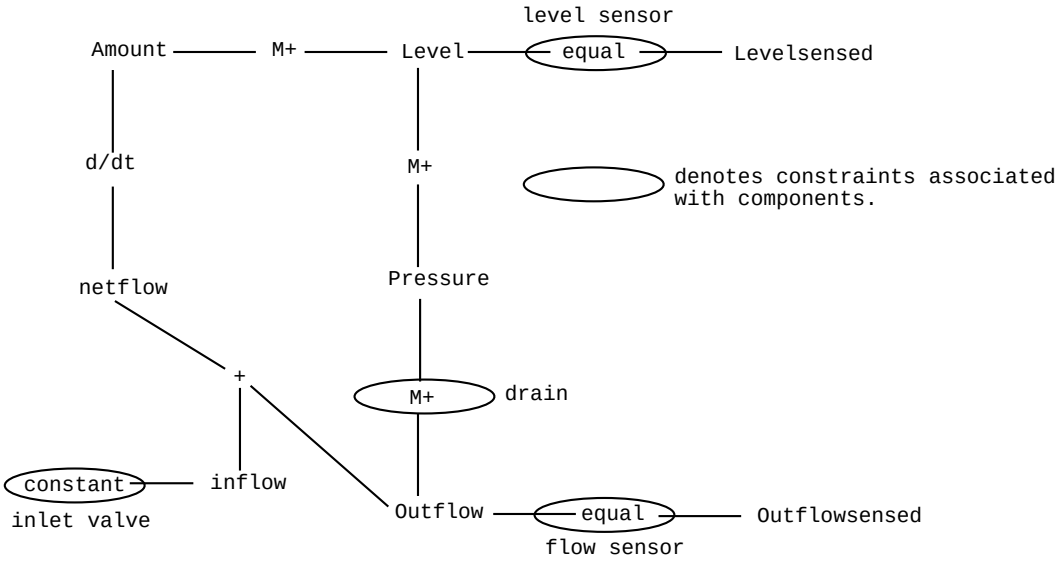

Figure 3: Constraint Network for the Bathtub

built using these candidates listed in order of decreasing probability. [2] The first item on this agenda is the next candidate used to query the diagnostic reasoner. The reasoner continues its best-first search until one or more consistent hypotheses are returned.

As an example, suppose we give the inputs `(levelsensed (0 top)), (outflowsensed 0)` to QDOCS with the bathtub model shown in figure 1. Figure 3 is a representation of the constraint network for the model of the bathtub where all components are assumed to be behaving normally. The first query to the constraint satisfier is with all the mode-variables set to normal. The constraint satisfier returns a conflict set of `((drain normal) (level-sensor normal) (outflow-sensor normal))`. The mode of the inlet valve could have been part of the conflict set given an exhaustive tracking of dependencies or a random order of choosing constraints but with the heuristics from the previous section this gets pruned.

Now, QDOCS places on the agenda all the hypotheses that are derived from changing each of the above mode variables to a different value. These are then ordered according to decreasing probability. For the example, the three most likely hypotheses are 1) the `outflowsensor` is stuck at high, 2) the `drain-mode` is blocked, and 3) the `level-sensor` is `stuck-at-top`, since individually these faults are the most probable. If the hypothesis that the `outflowsensor` is stuck at high is chosen first, the constraint satisfier returns immediately with the conflict set `(outflow-sensor stuck-high)`. This is because this mode directly contradicts the given sensor readings and is thus picked as the most restrictive constraint. The next best item on the agenda is then checked against all conflicts to ensure that it hits all of them and is then passed to the constraint satisfier. Suppose this hypothesis is that the drain is blocked. The constraint satisfier then checks against the observed sensor readings and obtains valid completions of the state. If the observations are part of a sequence, the simulator must verify that

[2] QDOCS calculates the probability of a set of behavioral modes assuming that modes for each component are independent.

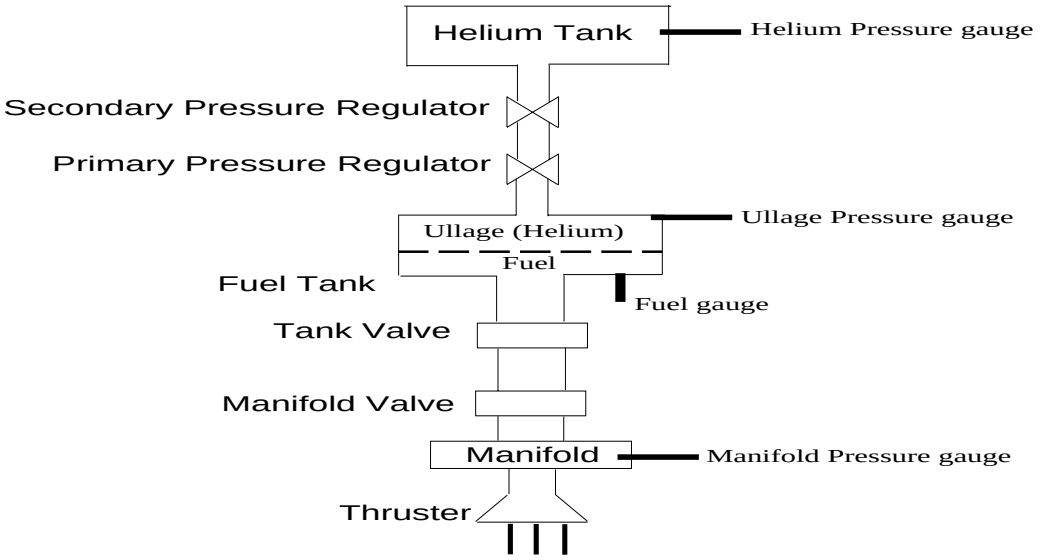

Figure 4: A Part of the Reaction Control System

there is a valid simulation under this hypothesis before declaring the hypothesis consistent.

4 An Experiment

We have described an implemented system that generates conflict sets and diagnoses by using constraint satisfaction and simulation on qualitative sensor values from the behavior of a system described using QSIM. This section describes an experiment performed to test the system on a realistic problem.

Figure 4 shows a portion of the Reaction Control System (RCS) of the Space Shuttle. The RCS consists of sets of jets mounted at three different points (two in the rear and one near the nose) on the space shuttle. These jets help provide control of orientation and velocity while the shuttle is in orbit. Each jet consists of two subsystems that are almost identical – one for the fuel and one for the oxidizer. These subsystems help deliver the fuel and oxidizer into a reaction chamber where they ignite to provide thrust. A model of one of these subsystems has been successfully modelled using QSIM and simulated in various fault modes (Kay, 1992).

We ran two experiments to try and evaluate the usefulness of QDOCS in this domain. In the first experiment, a part of the pictured subsystem of the RCS was modeled. This model was originally used to test an earlier version of QDOCS (Subramanian and Mooney, 1994). The second experiment was on a more complete model of the subsystem modeled by Kay. The two experiments helped to gauge the relationship between the efficiency of the QDOCS system and model size.

The system consists of a helium tank, a pair of pressure regulators (one of which acts to back up the other), a propellant tank containing either the fuel or oxygen, a pair of valves leading from this tank to a manifold and then to the thruster. When this system is working

ideally, the two pressure regulators control the pressure of the helium in the fuel tank at a level that allows for a constant outflow rate of fuel from the tank. When the pressure in the helium tank gets low enough that the regulators can no longer maintain the right pressure in the fuel tank, the regulators merely act as pipes and allow the pressure in the fuel tank to drop with the pressure of the helium tank. The fuel flowing out of the tank then goes through the two valves to the manifold from where it passes to the thruster.

Our model of the RCS has fault modes for the two pressure regulators (stuck-open and stuck-closed) and for the helium tank (leaking). Fault modes are also modeled for the two valves (stuck open/closed), for leaks in the fuel line and in the manifold. The thruster is also a component that may be stuck closed. Four sensors were assumed – pressure gauges for the two tanks and the manifold and a fuel gauge in the fuel tank. Fault modes for these sensors were also modeled. All of these components also have an unknown mode where the variables acted upon are unconstrained. A reasonable probability distribution was assigned to these faults. The complete model has 12 components and 80 constraints.

The smaller RCS model used in (Subramanian and Mooney, 1994) consisted of just the helium tank, the pressure regulators and the fuel tank. This model has 6 components and 42 constraints.

The *a priori* probabilities were used to randomly generate sets of faults for this model. After discarding the nearly 50% of cases where the random generator produced a completely normal set of components, QSIM was used to simulate the sets of faults from a given initial configuration. Given such an input, QSIM produces all the possible behaviors that are qualitatively consistent with the model and the initial configuration. A behavior was chosen (again at random) from this behavior tree and passed to QDOCS's diagnostic engine.

QDOCS was run for 100 iterations of the best-first search or until it generated a hypothesis consistent with all the observations, whichever came first. In some (about a third) of the cases, the behavior used was completely consistent with the behavior of a normal system. This was because some faults simply do not make a difference. For instance, the duplication of pressure regulators is purely for fault tolerance and if the secondary regulator breaks, the behavior of the system is qualitatively no different from the behavior of a normal system. In other cases, some faults (like leaks) may produce behaviors that are entirely qualitatively consistent with normal behavior if the leaks do not cause a qualitative change in the quantities and flow rates of the fluids in the system. These cases were discarded and the experiment concentrated on the rest.

QDOCS was compared against a simple generate and test strategy that used QDOCS as a verifier of hypotheses but otherwise simply tried all possible hypotheses in order of increasing *a priori* probability. Note that because of the nature of this problem, where every variable is potentially dependent on every component, this is the best possible approximation of a structural dependence strategy of the type used in MIMIC(Dvorak, 1992). Figure 5 gives the results of running QDOCS and a simpler generate-and-test algorithm on simulated behaviours from each of the models. The results are averaged over 50 randomly produced faults. The numbers in the first column show the percentage of cases where the given program produced the hypothesis originally used to generate the given behavior. The next column shows the percentage of cases where the diagnosed hypothesis is a subset of the faults in the original

	Correct %	Subset %	Hypotheses tested
Small RCS - QDOCS	56	80	10
Small RCS - generate and test	56	74	14
Large RCS - QDOCS	62	96	7
Large RCS - generate and test	56	86	18

Figure 5: RCS results

hypothesis while the final column shows the number of hypotheses tested by the algorithm in getting a hypothesis.

The results show that overall, QDOCS performs well in producing correct hypotheses in both models. Where it fails to produce the right hypothesis the hypothesis produced is often a smaller subset that is enough to explain the behavior. It is important to note that since QDOCS must verify hypotheses with simulation, the hypotheses it produces are always consistent with the observed behaviors. This means that the diagnoses produced by QDOCS are always "correct" in the sense that they are able to explain the given behavior of the system.

The most significant result is in the improvement shown by QDOCS over the generate and test method on the two models. This is measured by looking at the average number of hypotheses tested by each of the methods.[3] Notice that the improvement of QDOCS in the smaller model is significant but not very large. The improvement in the larger model is much larger as QDOCS tests only 7 hypotheses for 18 tested by the generate and test method. This confirms our belief that QDOCS will show significant improvements over simpler methods on diagnosing larger systems. The average run times for QDOCS on a Sparc 5 workstation for the larger RCS model was 20.2 seconds.

5 Discussion and Future Work

The experiment described in the previous section shows that our diagnostic reasoner does fairly well in this domain with random sets of faults. The combination of using the constraint satisfaction algorithm to generate hypotheses and the QSIM simulator to verify and generate further hypotheses is able to produce consistent diagnoses every time and significantly improves on the generate-and-test strategy.

Since the problems tested contain only a few highly interconnected components, conflict sets frequently contain all of these components. This limits the advantage of consistency-based diagnosis compared to the simpler generate and test strategy. However, as we have shown, as we scale up to larger problems with more components and sensors, the ability of QDOCS to use constraint satisfaction techniques to zero in on inconsistencies will produce significant speedup over other techniques. We plan to look at some other large domains to test our hypothesis.

[3] Since QDOCS was used to verify hypotheses in the generate and test method, a time comparison would not be meaningful.

On the computational side, another area we propose to investigate is that of efficient caching of possible values of different variables during the constraint satisfaction phase of the algorithm. Traditional truth maintenance systems like the ATMS(de Kleer, 1986) are not useful for this purpose since the range of possible values for a variable is rarely narrowed to a single one. Instead, the reasoner must cache sets of possible values derived under different sets of assumptions. We intend to explore ways in which to cache this information and test their utility in improving the efficiency of QDOCS.

6 Related Work

Compared to QDOCS, the previous diagnosis systems for QSIM models all have important limitations. INC-DIAGNOSE (Ng, 1990) was an application of Reiter's theory of diagnosis (Reiter, 1987) to QSIM models. Its main limitations were that first, like Reiter's theory, it was restricted to models where no fault mode information was known, and second, it used a constraint propagator that was not guaranteed to detect all inconsistencies. The propagator only worked in cases where all the variables that a constraint acted upon were restricted to just one (or zero) possible values. If this was not the case, the propagator would not detect some conflicts. QDOCS, on the other hand, uses behavioral-mode information and a complete constraint-satisfaction algorithm. Another system that used the INC-DIAGNOSE approach in the context of a monitoring system is DIAMON(Lackinger and Nejdl, 1991). Again, due to its dependence on the simple constraint propagation in INC-DIAGNOSE, it is only able to detect a small subset of possible faults which QDOCS can diagnose.

The other previous diagnosis work on QSIM models, MIMIC (Dvorak, 1992), has several limitations. First, MIMIC requires the model builder to provide a structural model of the system in addition to the QSIM constraint model. This structural model was fixed and could not change under different fault models. QDOCS does not require this since it uses a constraint-satisfaction algorithm to determine the causes for inconsistencies. Second, MIMIC uses a very simple dependency tracing algorithm to generate potential single-fault diagnoses. This algorithm looks at the structural graph from the point at which the fault is detected and considers all components it finds upstream as possible candidates for failure. QDOCS, on the other hand, uses actual dependencies of the values assigned to variables to identify multiple possibly failing components. It thus restricts itself to a smaller set of possible component failures.

A number of other researchers have looked at diagnosis in the context of monitoring continuous systems (Oyeleye et al., 1990; Doyle and Fayyad, 1991; Abbott, 1988). Each of these systems concentrates on different aspects of the monitoring process, but none performs multiple-fault diagnosis using behavioral modes.

Some recent work by Dressler (Dressler, 1994) performs model-based diagnosis on a dynamical system (a ballast tank system) using a variant of GDE. It first reduces the model to a version suitable for constraint propagation, and then considers only conflicts generated by constraints acting at a particular time. While this is an efficient technique that apparently works well for their application, it is not a general method since some systems may have faults

which can only be detected using information gathered across time.

7 Conclusion

We have described an architecture for diagnosing systems described by qualitative differential equations. It performs multiple-fault diagnosis using behavioral modes. An implemented system, QDOCS, has been shown to be powerful enough to accurately generate diagnoses from qualitative behaviors of a fairly complex system – the Reaction Control System of the Space Shuttle. The approach is more powerful than previous methods in that it uses 1) a general modelling framework (QSIM), 2) a more complete diagnostic architecture and 3) a powerful constraint-satisfaction algorithm as opposed to simple propagation.

References

Abbott, K. H. (1988). Robust operative diagnosis as problem solving in a hypothesis space. In *Proceedings of the Seventh National Conference on Artificial Intelligence*, 369–374. Minneapolis, MN.

Dague, P., Jehl, O., Deves, P., Luciani, P., and Taillibert, P. (1991). When oscillators stop oscillating. In *Proceedings of the Twelfth International Joint Conference on Artificial intelligence*, 1109–1115. Sydney, Australia.

de Kleer, J. (1986). An assumption-based TMS. *Artificial Intelligence*, 28:127–162.

de Kleer, J., and Williams, B. C. (1987). Diagnosing multiple faults. *Artificial Intelligence*, 32:97–130.

de Kleer, J., and Williams, B. C. (1989). Diagnosis with behavioral modes. In *Proceedings of the Eleventh International Joint conference on Artificial intelligence*, 1324–1330. Detroit, MI.

Doyle, R. J., and Fayyad, U. M. (1991). Sensor selection techniques in device monitoring. In *Proceedings of the Second Annual Conference on AI, Simulation and Planning in High Autonomy Systems*, 154–163. IEEE Computer Society Press.

Dressler, O. (1994). Model-based diagnosis on board: Magellan-MT inside. In *Fifth International Workshop on Principles of Diagnosis*, 87–92. New Paltz, NY.

Dvorak, D. (1992). *Monitoring and Diagnosis of Continuous Dynamic Systems Using Semi-quantitative Simulation*. PhD thesis, University of Texas, Austin, TX.

Forbus, K. D., and de Kleer, J. (1993). *Building Problem Solvers*. Cambridge, Massachusetts: MIT Press.

Guckenbiehl, T., and Schafer-Richter, G. (1990). Sidia: Extending prediction-based diagnosis to dynamic models. In *Working Notes of the 1st International Workshop on Principles of Diagnosis*, 74–82. Menlo Park, CA.

Kay, H. (1992). A qualitative model of the space shuttle reaction control system. Technical Report AI92-188, Artificial Intelligence Laboratory, University of Texas, Austin, TX.

Kuipers, B. J. (1994). *Qualitative Reasoning: Modeling and Simulation with Incomplete Knowledge*. MIT Press.

Lackinger, F., and Nejdl, W. (1991). Integrating model-based monitoring and diagnosis of complex dynamic systems. In *Proceedings of the Twelfth International Joint Conference on Artificial intelligence*, 1123–1128. Sydney, Australia.

Ng, H. T. (1990). Model-based, multiple-fault diagnosis of time-varying, continuous physical devices. In *Proceedings of the Sixth Conference on Artificial Intelligence Applications*, 9–15. Santa Barbara, CA.

Oyeleye, O. O., Finch, F. E., and Kramer, M. A. (1990). Qualitative modeling and fault diagnosis of dynamic processes by midas. *Chemical Engineering Communications*, 96:205–228.

Reiter, R. (1987). A theory of diagnosis from first principles. *Artificial Intelligence*, 32:57–95.

Subramanian, S., and Mooney, R. J. (1994). Multiple-fault diagnosis using general qualitative models with fault modes. In *Fifth International Workshop on Principles of Diagnosis*, 321–325. New Paltz, NY.

Waltz, D. (1975). Understanding line drawings of scenes with shadows. In Winston, P. H., editor, *The Psychology of Computer Vision*, 19–91. Cambridge, Mass.: McGraw Hill.