Appears in *Working Papers of the 5th International Workshop on Principles of Diagnosis* pp.321-325, New Paltz, NY, October 1994

Multiple-Fault Diagnosis Using General Qualitative Models with Fault Modes

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

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

Abstract

This paper describes an approach to diagnosis of systems described by qualitative differential equations represented as QSIM models. An implemented system QDOCS is described that performs multiple-fault, fault-model based diagnosis, using constraint satisfaction techniques, of qualitative behaviors of systems described by such models. We demonstrate the utility of this system by accurately diagnosing randomly generated faults using simulated behaviors of a portion of the Reaction Control System of the space shuttle.

1 Introduction

Qualitative reasoning is well-established as a useful mode of reasoning for the simulation and monitoring of continuous physical systems. By reasoning in terms of qualitative ranges of variables as opposed to precise numerical values, it is possible to compute information about the behavior of a system with very little information about the system and without doing expensive numerical simulation. A very general language for describing physical systems as qualitative differential equations is given by QSIM (Kuipers, 1994), a program that simulates systems described in this language. Previous approaches to diagnosing faults in QSIM models have either been unable to work with fault models (Ng, 1990) or have made a single-fault assumption (Dvorak, 1992). Most previous work on model-based diagnosis (for example, (Reiter, 1987) or GDE(de Kleer and Williams, 1987)) has concentrated on static systems and is insufficient to diagnose continuous systems. Few of the other approaches to diagnosis of continuous physical systems (for example (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 the work of Reiter or DeKleer.

This work is an initial attempt at building a general, multiple-fault diagnosis system which uses behavioral mode information with *a priori* probabilities. The diagnostic architecture is similar to SHERLOCK (de Kleer and Williams, 1989) and the algorithm builds on the work of (Ng, 1990) in INC-DIAGNOSE. A general constraint satisfaction technique is used to detect faults and trace dependencies. The implemented system introduced here, QDOCS (for Qualitative Diagnosis Of Continuous Systems), is powerful enough to accurately diagnose a number of different fault models for a part of the Space Shuttle's Reaction Control System - a realistic system, modeled with 42 QSIM constraints.

The rest of this paper is organized as follows: in section 2, a simple example which will be used throughout the paper is used to motivate the work. QDOCS's algorithm is presented in section 3 and its application is illustrated on the example from section 2. Section 4 then reports on an experiment done to test QDOCS and presents the results obtained. Section 5 contains a discussion of some of the limitations of the approach presented 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 we use to illustrate our algorithm consists of a simple bathtub with a drain. It is assumed that this 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 represented as 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 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

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

```
(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

```
(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

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)].

3 QDOCS's Diagnostic Approach

Using the standard approach to 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, it 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 of these and uses constraint satisfaction to determine which of these are consistent. The process of finding successors to states and filtering on the constraints continues as QSIM builds a tree of states called an *envisionment*.

QSIM solves the CSP by 1) establishing node consistency, 2) using Waltz filtering (Waltz, 1975) to establish arc consistency, 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 early to avoid unnecessary search.

A model is inconsistent with a given sequence of sensor readings if there is no corresponding behavior in the envisionment. 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 the QSIM envisionment.

The second case is computationally much more expensive to account for. This is because a given set of sensor readings may have hundreds of possible completions each of which must be compared with the hundreds of potential successors in order to determine the causes for the continuity failure. The algorithm developed in this paper accounts for the first case listed above while our current research is aimed at finding ways to obtain diagnostic information from the second kind of failure. It is shown, through experiments on an example, that diagnosing static inconsistency, is, by itself, enough to account for some very realistic faults.

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's tag set of possible values causes a variable's number 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 number of possible tuples in the constraint's tag, the constraint's dependency set is similarly updated with the union. When a variable is left with no possible values, the dependency set it is left with is returned as a conflict set.

The heuristic of filtering on the most restrictive constraints early helps reduce the size of conflict sets but it does not guarantee minimal conflicts. The size reduction comes from the fact that the conflicts chosen initially already have only a small number of tuples in the cross-product of the sets of possible values of the variables they act upon. A constraint with fewer possible tuples to begin with is 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, we've chosen to go with the simpler algorithm discussed here despite the fact that the conflict sets obtained are not guaranteed minimal. Of course, no known technique makes such a guarantee.

3.2 Constructing Diagnoses

The above process is used to compute a single conflict set when the constraints corresponding to a particular set of values for the mode-variables are activated. 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 mode-variables 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 mode-variable in the conflict set in addition to the behavioral modes of the current candidate. An agenda is 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 retuns a conflict set of ((drain normal) (level-sensor normal) (outflow-sensor normal)). Note that 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.

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

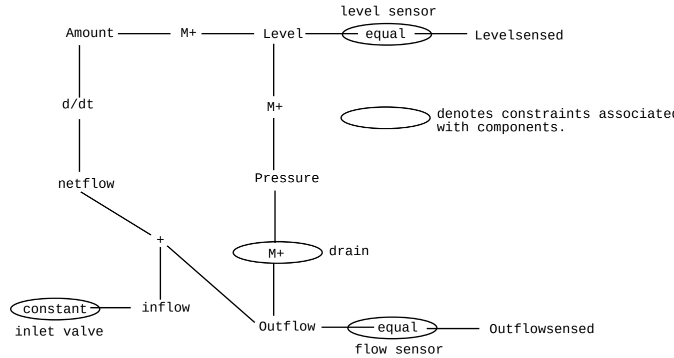

Figure 3: Constraint Network for the Bathtub

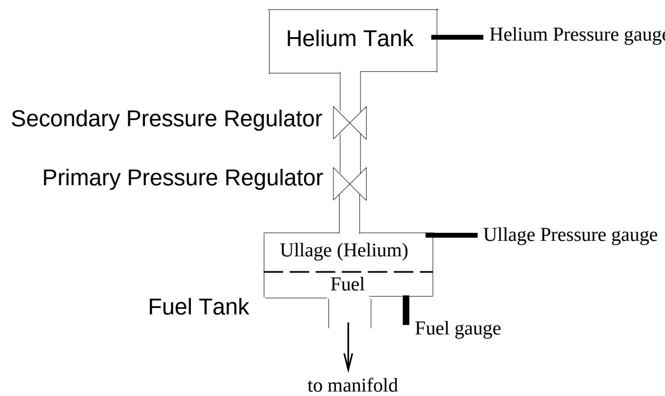

Figure 4: A Part of the Reaction Control System

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. Suppose the first hypothesis is that the outflowsensor is stuck at high while all the other components are behaving normally. This is the first one picked to focus on. However, with this hypothesis, 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 the first constraint to be picked by the constraint satisfier using its restrictive constraint heuristic. 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 declares the hypothesis consistent.

4 An Experiment

We have described an implemented system that performs the task of obtaining conflict sets and generating diagnoses by using static constraint satisfaction on states from the behavior of a system described using QSIM. This section describes an experiment performed to gauge the extent to which this alone would be a useful tool in a realistic system.

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 described qualitatively using QSIM and simulated in various fault modes (Kay, 1992).

For this experiment, we further concentrate on about half of this subsystem consisting of the helium (propellant) tank, the two pressure regulators and the fuel tank. 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.

A model was built which had fault modes for the two pressure regulators (stuck-open and stuck-closed) and for the helium tank (leaking). Three sensors were assumed - pressure gauges for the two tanks and a fuel gauge in the fuel tank. Fault modes for these sensors were also modeled. All these components also have an unknown mode where the variables acted upon are unconstrained. Probabilities were assigned arbitrarily just for testing purposes since we did not have information on the actual probabilities of these faults. The complete model has 42 constraints.

The *a priori* probabilities were used to randomly generate sets of faults in this model. Since four of the components have three modes each and two of the components have four modes each, this made for a space of 1296 possible behavioral modes altogether. 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 with a fixed 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 envisionment graph and passed to QDOCS's diagnostic engine.

QDOCS was run for 100 iterations of the best-first search or until it generated 5 hypotheses 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. These cases were discarded and we concentrate on the rest.

When this experiment was run on 200 randomly generated faults, the correct hypothesis was among the top 5 produced by QDOCS in 86% of the cases. In 64%, it was the top hypothesis. A lot of the cases (64% of the failing cases, or all but 5% of the total cases) where QDOCS did not produce the correct hypothesis were ones where a subset of the faults was enough to account for the behavior, and hence the more likely, smaller, fault set was among the ones generated. The test runs took approximately a minute each on a Sparc 2 running Lucid Common Lisp.

5 Discussion and Future Work

The experiment described in the previous section shows that the diagnostic reasoner described does fairly well in this domain with random sets of faults. Since QDOCS is currently unable to use continuity information between states, what this shows is that in this domain, most behaviors derived from different sets of behavioral modes lead to states that are qualitatively distinct from each other. Our plan is to study some more complex systems, particularly in the domain of chemical reactors, and explore the limits of this mode of diagnostic reasoning.

Certain kinds of faults, however, will lead to states that are not qualitatively distinct from states consistent with other behavioral modes. As an example of a system where faults cannot be detected by static analysis alone, suppose we had a system consisting of a closed tank containing some fluid. Suppose also that the level sensor for the fluid was unable to gauge the direction of change of the fluid's level. From a sequence of states with decreasing qualitative magnitudes for the level of fluid, one should be able to derive a contradiction. The current QDOCS will not detect such a contradiction because it cannot perform an analysis of behaviors across time steps. The fact that successive states of the system must be continuous may be used to derive a contradiction from the behavior. This would involve looking at all possible completions of the state and deciding what components were responsible for the lack of continuity between the state and its successors. This is an approach we plan to explore.

On the computational side, another area we propose to insvestigate is that of efficient caching of possible values for different variables during the constraint satisfaction phase of the algorithm. Traditional truth maintenance systems like the ATMS 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

The two previous systems implemented to perform diagnosis on systems modeled using QSIM both suffer from a number of limitations that QDOCS does not. INC-DIAGNOSE (Ng, 1990) was an implementation 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 very primitive constraint propagator that was not guaranteed to detect all the faults that could be detected within a state. The propagator would only work in cases where all the variables that a constraint acted upon were restricted to just one (or zero) possible values after the constraint was applied. If this was not the case, the propagator would fail and possible conflicts would not be detected. QDOCS, on the other hand, does use behavioral mode information to find a model consistent with the behavior and it uses a complete constraint satisfaction algorithm.

The other previous diagnosis work on QSIM models, MIMIC (Dvorak, 1992) suffers from a number of different problems. 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 faults. 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 the possibly failing components. It thus restricts itself to a smaller set of possible component failures. Third, MIMIC makes a single-fault assumption, while QDOCS doesn't.

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 the multiple-fault fault-based diagnosis that QDOCS performs.

Two other systems that perform diagnosis on dynamical systems include CATS (Dague et al., 1991) and SIDIA (Guckenbiehl and Schafer-Richter, 1990). Both are systems that use numerical methods to obtain conflicts and perform diagnosis on circuits with state but neither one can a) handle fault modes, or b) generalize to qualitative differential equations. Pan's work on predictive analysis (Pan, 1984) uses a qualitative model to perform diagnosis on analog circuits but it requires a complete state transition diagram for all the states of the circuit - something that is obviously combinatorially explosive in the kinds of models that we are considering here.

7 Conclusion

We have described an architecture for the diagnosis of systems described by qualitative differential equations. The technique uses a multiple-fault, fault-model based approach to generating diagnoses. 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 approaches to the problem in that it uses 1) a general modelling framework in 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., 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.

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

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.

Kay, H., and Kuipers, B. J. (1993). Numerical behavior envelopes for qualitative models. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, 606–613. Washington, D.C.

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

Kuipers, B. J., and Berleant, D. (1988). Using incomplete quantitative knowledge in qualitative reasoning. In *Proceedings of the Seventh National Conference on Artificial Intelligence*, 324–329. St. Paul, MN.

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.

Pan, J. (1984). Qualitative reasoning with deep-level mechanism models for diagnoses of mechanism failures. In *Proceedings of the 1st IEEE Conference on AI Applications*, 295–301. Denver, CO.

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

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.