Daniel Dvorak and Benjamin Kuipers
Process monitoring and diagnosis: a model-based approach.
IEEE Expert 6(3): 67-74, June 1991

Abstract

This article describes a method for monitoring and diagnosis of process systems based on three foundational technologies: semi-quantitative simulation, measurement interpretation (tracking), and model-based diagnosis. Compared to existing methods based on fixed-threshold alarms, fault dictionaries, decision trees, and expert systems, several advantages accrue:

- the physical system is represented in a semi-quantitative model which, unlike a pure numeric model, predicts all possible behaviors that are consistent with the incomplete/imprecise knowledge of the system's devices and processes, ensuring, for example, that a hazardous-but-infrequent behavior will not be overlooked;
- imprecise knowledge of parameter values and functional relationships (both linear and non-linear) can be expressed in the semi-quantitative model and used during simulation, producing a valid range for each variable;
- incremental simulation of the model in step with incoming sensor readings, with subsequent comparison of observations to predictions, permits earlier fault detection than with fixed thresholds;
- by using a structural model of the plant and tracing upstream from the site of unexpected readings, model-based diagnosis permits efficient generation of fault candidates without resort to pre-compiled (and often incomplete) symptom–fault patterns;
- by injecting a hypothesized fault into the model and tracking its predictions against observations, the dynamic behavior of the plant is exploited to corroborate or refute hypotheses;
- by simulating ahead in time from the current state, an operator can be forewarned of nearby undesirable states that the plant might enter.

Process Monitoring and Diagnosis: A Model-Based Approach

Daniel Dvorak
AT&T Bell Laboratories

Benjamin Kuipers
The University of Texas at Austin

January 2, 1996

In a book with the curious title of *Normal Accidents*, author Charles Perrow [1] examines several of the most notable accidents involving complex systems of the modern industrial world — accidents such as the 1979 Three Mile Island nuclear power accident, the 1977 New York City blackout, and the 1969 Texas City explosion of a butadiene refining unit. Perrow highlights the difficult job of plant operators who are responsible for physical systems having complex interactions and tight coupling. With current monitoring technology, alarms are triggered whenever fixed thresholds are exceeded. A nuclear power plant, for example, can have over a thousand distinct alarms, and hundreds of them can be activated within a minute, as in a loss-of-coolant accident. In such situations, process operators tend to overlook relevant information, respond too slowly, and panic when the rate of information flow is too great.

Not surprisingly, operator advisory systems have become an important area of application for expert systems. Systems such as Escort [2] (an expert system for complex operations in real-time) and Realm [3] (a reactor emergency action level monitor) are two of many expert systems developed for process industries. For surveys of this work, see [4] and [5]. These systems aim to reduce the cognitive load on operators, usually by helping to diagnose the cause of alarms and possibly to suggest corrective actions. Most of these expert systems get their knowledge of symptoms, faults, and corrective actions through the usual process of codifying human expertise in rules or decision trees. The problem, as with all expert systems, is reliability. As Denning observes, "the trial-and-error process by which knowledge is elicited, programmed, and tested is likely to produce inconsistent and incomplete databases; hence, an expert system may exhibit important gaps in knowledge at unexpected times" [6]. Obviously, these "gaps in knowledge" can have serious consequences in some process industries.

An alternate approach — one which is not based on an expert's imperfect recall of symptoms and faults — is to use a model of the process to predict its behavior or at least check consistency among some observed variables. When observations disagree with the model's predictions, some diagnostic technique is initiated to identify the fault candidates. These *model-based* approaches to diagnosis have emerged from two different communities. In the engineering community, *fault detection and isolation* (FDI) techniques generally rely on a precise mathematical model of the process and on pre-enumerated symptom–fault patterns ("fault signatures"). See [7] for an excellent collection of state-of-the-art work in FDI. In the computer science/AI community, *model-based reasoning* (MBR), as applied to diagnosis, relies on models of structure and behavior. Given symptoms of misbehavior (as detected with the behavioral model), fault candidates are identified by following a dependency chain back from a violated prediction to each component and parameter that contributed to that prediction. See [8] for an excellent survey of model-based troubleshooting.

The model-based approach described in this article has evolved within the AI community, but some similar ideas have appeared independently in the FDI literature, such as that of Isermann [9]. In all the model-based approaches it's important for the reader to look closely at the type of model used since that determines many of the capabilities and limitations of the specific method. The variety of model types is evident in a short sampling of the literature — dynamic quantitative mathematical models [9], dynamic qualitative mathematical models [10], the extended signed directed graph [11], causal models and confluence equations [12], fuzzy qualitative models [13], and the semi-quantitative model in this article.

This article focuses on process monitoring and diagnosis — basic elements of an operator advisory system. In this setting several conditions hold that challenge diagnostic methods: the plant is a continuous-variable dynamic system with feedback loops and state, diagnosis must be performed while the system operates, many system quantities are not sensed, and measurements are unreliable due to sensor failures. We begin our presentation with an intuitive description of our design for process monitoring and diagnosis. As we describe later, the design is based on three foundational technologies: semi-quantitative simulation, measurement interpretation (tracking), and model-based diagnosis. We then walk through an example of the system at work, detecting and diagnosing a fault. Finally, we list some limitations of the method and discuss related work in the field.

The Basic Idea: Mimicry

The key cognitive skill for process operators is the formation of a mental model that not only accounts for current observations but also enables them to predict near-term behavior and predict the effect of possible control actions. This observation underlies our framework for process monitoring, named Mimic. The basic idea is quite simple: *mimic* the physical system with a predictive model, and when the system changes behavior due to a fault or repair, change the model accordingly so that it continues to give accurate predictions of expected behavior. Intuitively, Mimic incrementally simulates a model of the physical system in step with incoming observations, making the state of the model track the state of the physical system. When the observations disagree with predictions, model-based diagnosis is employed to determine the possible fault(s). When a fault is identified, it is injected into the model so that the model's predictions continue to track observations. The key benefit is that we can use the model as our window into what's happening inside the physical system. Specifically, the model can be used to:

- detect early deviations from expected behavior, much more quickly than with fixed-threshold alarms (an extreme form of *analytical redundancy*[1]);

[1] The term *analytical redundancy*, also called *functional redundancy*, refers to the fault detection method of using known analytical relationships among sets of signals, such as outputs from dissimilar sensors, to check for mutual consistency. The method (and the phrase) emerged as an alternative to the earlier practice of *hardware redundancy*, wherein

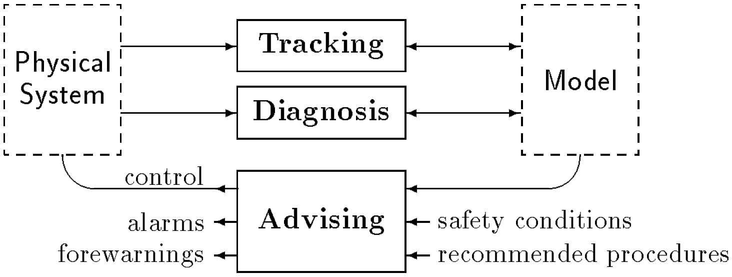

Figure 1: The 3 tasks of process monitoring.

- predict the values of unobserved variables (*signal reconstruction*) to permit alarms or other inferences on unseen variables, and to assist the operator's understanding of process conditions;
- predict ahead in time, thus forewarning of undesirable or hazardous conditions;
- predict the effect of proposed control actions to test if the control action will have the desired effect — a valuable capability in complex systems.

Figure 1 depicts this framework where a predictive model mimics the physical system. Two tasks maintain the model. The *tracking* task advances the state of the model in step with observations from the physical system. The *diagnosis* task, upon identifying a particular fault, injects that fault into the current model so that the predictions of the model will continue to agree with actual observations. To be precise, Mimic maintains a *set* of candidate models since a given behavior might be caused by one of several faults. Each candidate model represents a possible condition of the system (*i.e.*, its state and faults).

The end purpose of monitoring and diagnosis is *advice* — advice to the operator about what's happening and what to do about it. The role of the *advising* task is to apply the expert knowledge of safety conditions, recommended operating procedures, and performance objectives to produce advice in the form of alarms, forewarnings, and recommended actions. Although it is not further discussed in this paper, the advising task is a major beneficiary of the model-based approach in that the candidate models (and their

3 or 4 identical sensors and voting logic are used for fault tolerance.

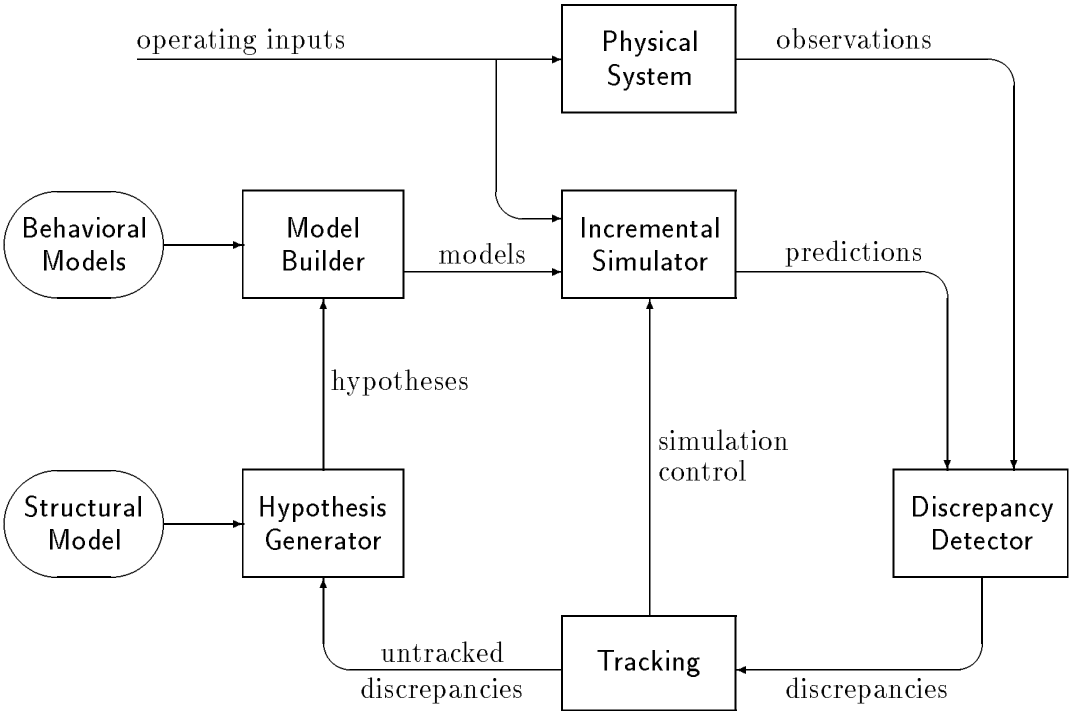

Figure 2: Architecture of Mimic.

tracked states) can provide a testbed for generating forewarnings and for testing proposed control actions.

1 Three Key Technologies

Mimic is built upon three technologies that have emerged from research in recent years: semi-quantitative simulation, measurement interpretation (tracking), and model-based diagnosis. These technologies are joined together in a *hypothesize-build-simulate-match* cycle, as shown in Figure 2. This figure gives a more detailed view of the tracking and diagnosis tasks of Figure 1.

1.1 Semi-Quantitative Simulation

Industrial process plants (such as chemical refineries and nuclear power plants) are examples of continuous-variable dynamic systems. In modern control theory these dynamic systems are modeled with a set of coupled first-order differential equations consisting of the balance equations, physical-chemical state equations, and phenomenological laws. Given a set of initial values, a numerical simulation of the equations yields precise predictions of values for each variable over time.

Oddly enough, standard numeric simulation is too precise and too narrow for our purposes. In reality, there is much imprecision in process systems. Sensors, actuators, and functional units operate within certain tolerances, parameter values are known approximately, and some functional relations are not known with precision. One approach, of course, is to do precise numerical simulation using average values, and then use some form of approximate matching of simulation results to observations. There are two main problems with this approach:

1. Given initial conditions, numerical simulation predicts only one behavior from a model even though more than one may be possible, given the real imprecision. For example, a tiny difference in one parameter can determine whether or not a rocket achieves escape velocity. In effect, numerical simulation makes an inappropriate commitment to a single behavior whereas qualitative simulation guarantees that all possible behaviors will be predicted. This capability is especially important in testing a fault hypothesis, which may exhibit several qualitatively distinct behaviors.

2. The second problem is the *approximate-matching* problem — how do you decide, in a principled way, when a difference between prediction and observation is due to imprecision and when it is due to a fault? What we *really* want is to explicitly express the imprecise knowledge as part of the model and have the simulator use it, producing valid ranges for each variable, permitting direct matching of observations to predictions. Semi-quantitative simulation provides this capability. Furthermore, when the observations match the predictions of two (or more) distinct behaviors, we want to track *both* hypotheses until they

diverge, which Mimic does.

Qualitative simulation [14], the foundation for semi-quantitative simulation, has two important characteristics for our application. First, it uses a qualitative level of description that permits imprecise knowledge to be expressed. This purely qualitative description uses no numbers (but can take advantage of quantitative information when available, as we shall soon see). Second, qualitative simulation generates *all* of the qualitatively distinct behaviors attainable from a starting state, consistent with the given imprecise knowledge. This property is essential in monitoring a physical system, whether healthy or faulty.

Quantitative refinement of qualitative simulation [15] takes advantage of quantitative knowledge when it is available, which is always the case in process plants. This knowledge consists of *numeric ranges* for landmark values (*e.g.*, the pressure-relief valve opens at 200–210 psi) plus *envelope functions* that define the limits of monotonic relationships (*e.g.*, an approximate relationship between the volume of fluid in a tank and its height). Quantitative values are expressed as numeric ranges and simulated with a modified interval arithmetic. Interval arithmetic is normally subject to an uncertainty explosion, but this problem is avoided because all reasoning takes place with respect to the fixed set of landmarks provided by the qualitative behavior. The resulting semi-quantitative simulation retains all the properties of qualitative simulation, but with two additional benefits: (1) behaviors that are qualitatively possible but quantitatively invalid are eliminated, and (2) numeric range predictions are generated for each variable and can be directly compared to numeric sensor readings. This form of semi-quantitative simulation is provided in Qsim, which we have used in this research.

1.2 Measurement Interpretation (Tracking)

Mimic seeks to maintain a model whose state and whose faults (if any) reflects the current state of the physical system. More precisely, it maintains a *set* of models, each in a state consistent with the most recent observations. This set is called the "tracking set", and each model in the set embodies different fault hypotheses and therefore represents an alternate interpretation of the system. Models are added to the tracking set during diagnosis as

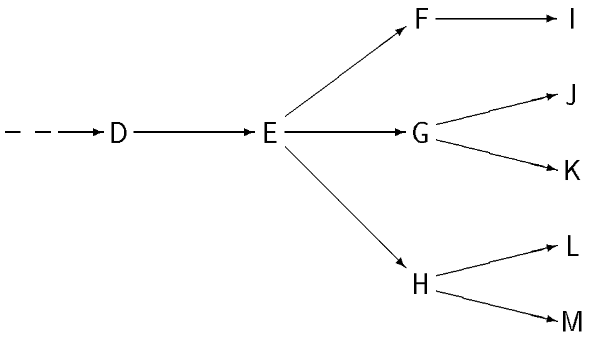

Figure 3: Tracking through a behavior graph.

fault/repair hypotheses are generated. Models are deleted from the tracking set during tracking, when a model's predictions do not track observations.

Qualitative simulation generates a behavior graph, which is a directed graph of the possible qualitative states of the system and the transitions among them. A *behavior* is a path through the directed graph and consists of a sequence of states alternating between states that represent an instant of time and states that represent an interval of time. *Tracking*, also called measurement interpretation [16, 17], is the process of using the observations to follow a path (a behavior) through the behavior graph. Using the fragment of behavior graph in Figure 3, we describe several details of the process:

- If a model is currently in state E, then each new set of observations is compared to the values of state E. If the observations match the predictions, (*i.e.,* if each observation is within the predicted range), then the model remains in state E. Note that no *ad hoc* approximate matching is needed. Because the semi-quantitative model predicts a valid numeric range for each variable, observations are compared directly to the predicted ranges of each simulated state.[2]

- When an observation does not match the current state E, *incremental simulation* is used to generate the immediate successor states F, G, H.[3]

[2] Usual noise filtering of sensor readings should still be performed.

[3] Recall that the reason why a state in a semi-quantitative simulation can have more than one possible successor state is because of the imprecise knowledge expressed in the model.

If, say, the observations match state G, then the model is retained with its state now set to G.

Incremental simulation refers to the control that the tracking task exerts over the simulator. When triggered, the simulator generates only the immediate qualitative successor state(s) to the current state. Thus, the simulation is advanced only as needed; it is never "run to completion".

- If the observations do not match any of the immediate successor states *(F,G,H)*, then incremental simulation is repeated and the observations compared to the second-generation successor states *(I, J, K, L, M)*. This limited-distance look-ahead is needed to jump over instantaneous states that fall between consecutive observations.

- Observations may include independent variables, *i.e.*, exogenous variables whose values cannot be predicted. When an independent variable changes value, tracking must reinitialize the models in the tracking set using the current observations and most recent predictions for integrated quantities. Thus, the simulation, just like the physical system, is made to react to changes in independent variables.

- Through progressive step-size refinement of the semi-quantitative simulation, a desired precision can be attained for the quantitative predictions [18]. Specifically, the time interval between qualitative time points can be used to refine the step-size of the quantitative simulation, inserting new quantitative states having time points within that interval. These new quantitative states more precisely bound the predicted behavior.

- It's important to note that MIMIC never has to generate the full behavior graph (an "envisionment") — a computation that can be prohibitively expensive for complex systems because of the intractable branching problem of qualitative simulation. Instead, MIMIC performs incremental simulation to generate only the states in the immediate vicinity of the last tracked state, abandoning any branches that do not track the observations. In effect, the observations act as a filter that eliminates irrelevant branches in the behavior graph.

1.3 Model-Based Diagnosis

Process systems are designed for continuous operation, and are therefore somewhat fault-tolerant. The economic pressures to keep the plant in production often mean that the system will continue running with multiple faults. Thus, single-fault diagnosis is inadequate. However, complete multiple-fault diagnosis is combinatorially explosive and therefore unrealistic for real-time monitoring. As a middle approach, MIMIC uses a method for incrementally constructing and testing multiple-fault hypotheses. Specifically, since the periodic sensor measurements are frequent, it is assumed that only a single new fault (or a single repair) can occur between successive measurements. Thus, MIMIC *can* construct multiple-fault hypotheses, one hypothesis at a time.

Let's now examine when and how diagnosis occurs. The tracking task discards a model when there is a discrepancy between predictions and observations. However, before the model is discarded, an attempt is made to modify it to bring its predictions into agreement with the observations. This is similar in intent to the *Debug* phase of the Generate-Test-Debug (GTD) paradigm [19], though GTD and MIMIC differ in many other ways. Using the *structural* model (the model of components, connections, and parameters), the algorithm traces upstream from the site of the discrepancies to identify all components and parameters that could have contributed to the discrepancy (*dependency tracing*). Under the assumption that the discrepancies are due to a single new fault or a single new repair, the only suspects to consider are those that can account for *all* discrepancies. These suspects are further checked for global consistency through constraint-suspension; if there is no assignment of values that is consistent with all the symptoms, then the suspect is exonerated. For each remaining suspect, each of its other operating modes is tested for compatibility with the observations. Whatever model variations survive this test are added to the tracking set. For a more detailed description of model-based diagnosis, see [8].

Unlike many diagnostic methods, model-based diagnosis does *not* rely on a set of symptom–fault patterns. The problem with relying on such patterns is that they are often incomplete since it is difficult for an expert to anticipate all possible faults and their symptoms, especially the symptoms of interacting faults. Even if the symptom–fault patterns are collected from exhaustive fault-model simulations, it is not necessarily more efficient to use such patterns, as Davis has argued [20].

Another important property of model-based diagnosis is that it handles failed sensors and missing data in a natural way, not as a special case. A sensor is just another component that affects an observation; dependency tracing will identify it as a suspect in the usual way. As Scarl [21] observes, model-based reasoning avoids combinatoric problems in handling failed sensors and unavailable data because it matches against predictions rather than symptomatic patterns. If a sensor is bad and thus gives readings different than predicted, the sensor becomes a suspect simply because it is upstream of the discrepancy. If a datum is unavailable, then it is not compared with predictions and therefore cannot cause discrepancies.

2 Example

To illustrate Mimic at work, consider the electric water heater shown in Figure 4, which has been modeled and tested with Mimic. The water heater has a single thermostat which controls whether or not power is applied to the two heating elements (on-off control). Raw sensor information comes from a temperature sensor near the thermostat, from a flow-rate sensor on the cold-water inlet, and from a voltage sensor on the heating elements. In a real monitoring situation we would want to diagnose a variety of possible faults such as defective heating elements, a stuck thermostat, a faulty flow-rate sensor, and loss of electrical power. However, to keep this example simple, we consider only the possibility of defective heating elements.

The water heater is modeled as a 2-compartment model in which two masses of water (upper and lower) are connected with thermal flow and mass flow between them, as shown in Figure 5. Each compartment is treated as well-mixed (the temperature is the same everywhere within the compartment). The temperature in each compartment is affected by five heat flows: heat gain from the heating element, heat loss to the room through the insulating jacket, heat gain due to water inflow, heat loss due to water outflow, and heat transfer through thermal contact with the other compartment. The semi-quantitative Qsim model of the water heater contains the usual equations that relate mass, mass flow, heat, heat flow, thermal resistance, and

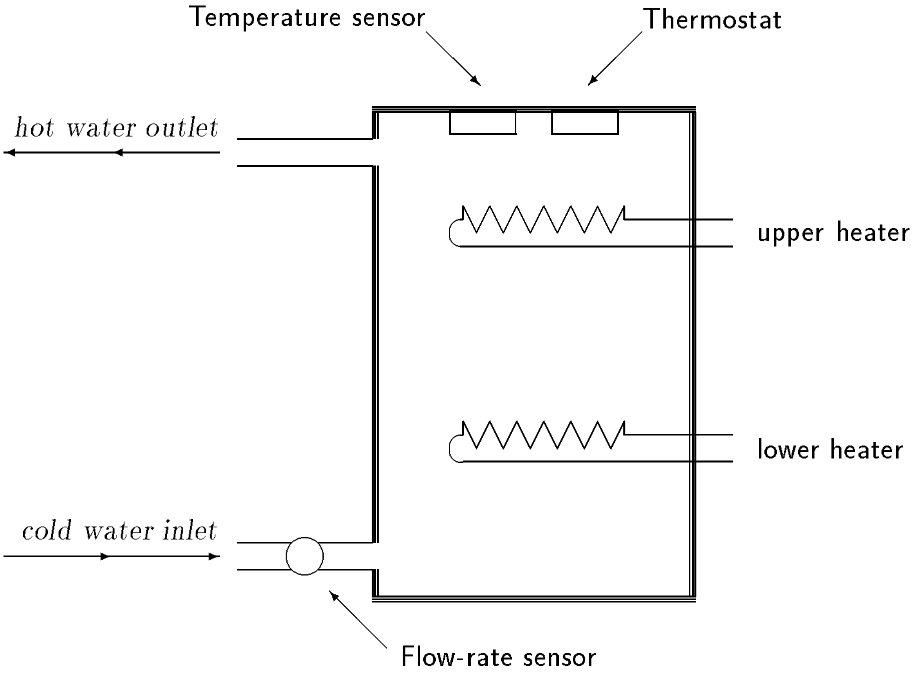

Figure 4: Electric water heater.

temperature. In addition, it contains numeric ranges for landmark values such as room temperature, inlet water temperature, nominal heating rate, and thermal resistance of the insulation. No envelope functions are needed in this model because there are no imprecise monotonic relationships.

In the normal (fault-free) model all components of the water heater (tank, heating elements, thermostat, temperature sensor, flow-rate sensor, voltage sensor, voltage supply) operate according to their intended purpose. In a fault model, a faulty component operates according to a failure mode, such as a heating element that generates no heat when power is applied.

Table 1 summarizes an example of monitoring the water heater, showing how monitoring progresses over eight moments in a series of observations.[4] For each moment, the table shows the quantitative sensor readings and three sets maintained inside Mimic. The water heater begins in a state where the water in the tank is hot, the heating elements are off, no water is flowing,

[4] The numeric values shown in Table 1 are from a numeric simulation of the water heater in which the lower heater produces no heat.

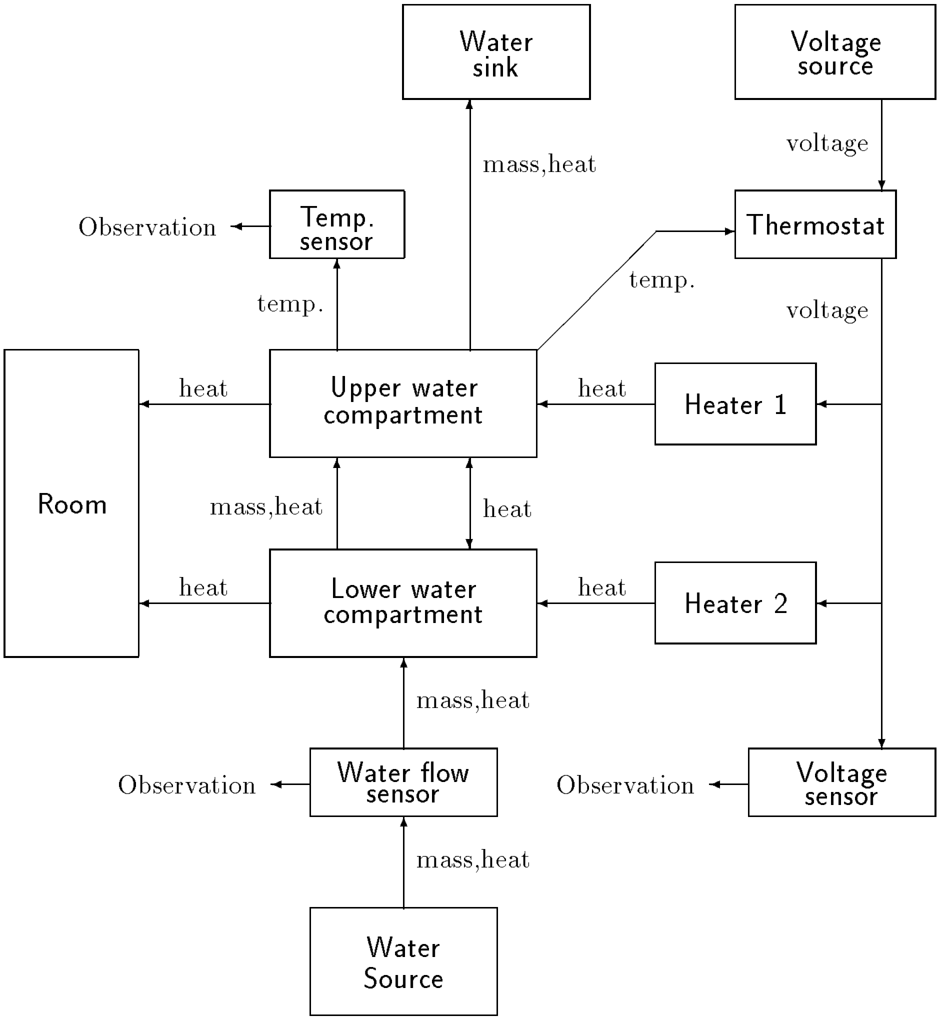

Figure 5: Structural model of water heater.

Moment	0	1	2	3	4	5	6	7
Synopsis	temp hot	flow starts	temp dropping	heater on	temp still dropping	flow stops	temp rising	heater off
Time (min.)	0.0	1.0	2.0	2.4	2.7	3.0	13.0	27.7
Flow (liters/min.)	0	30	30	30	30	0	0	0
Temp. (deg. C)	64.9	64.9	61.4	58.9	57.1	55.9	60.0	66.0
Power (on or off)	off	off	off	on	on	on	on	off
New fault hypotheses	none	none	none	none	bad H1 bad H2	none	none	none
New fault model(s)	none	none	none	none	bad H1 bad H2	none	none	none
Tracked model(s)	normal	normal	normal	normal	bad H1 bad H2	bad H1 bad H2	bad H2	bad H2

Table 1: Diagnosing the water heater from dynamic behavior.

and there is a slow temperature loss. These readings are consistent with the normal model. Now, someone starts to draw water for a bath. A high flow rate is measured but all other readings remain the same. Since water flow is an independent variable, Mimic reinitializes every tracked model (just the normal model in this case) to reflect the change. Since the normal model is consistent with the new values, it is retained.

As time continues, the temperature inside the tank drops because of the cooler inlet water. These readings are consistent with the current state of the normal model, so no change occurs to the tracking set. At moment 3 the temperature drops to the point where the heating elements turn on, as observed by the voltage sensor. These readings are also consistent with the predictions of the normal model, so the model is retained.

At moment 4 the temperature continues to drop. Although this observation is *qualitatively* consistent with the normal model, it is inconsistent with the associated *quantitative* ranges. In effect, the model is saying that for this flow rate, tank capacity, heating rate, and inlet temperature, the water temperature should not be dropping so fast. Thus, the tracking task discards the normal model. At the same time, this discrepancy triggers dependency tracing which identifies two possible faults — a bad upper heating element or a bad lower heating element (denoted bad-H1 and bad-H2)[5]. This causes two fault models to be built. Both models are successfully initialized, so MIMIC is now tracking two models.

The water flow stops at moment 5 (somebody turned off the faucet). With this change in an independent variable, MIMIC reinitializes the two models. At moment 6, the temperature is observed to be rising. The observed temperature is then compared to the quantitative predictions of the two models. Because the observed temperature exceeds the range predicted by the bad-H1 model, that model is discarded. The predictions of the one remaining model, bad-H2, are compatible with the observations, so the model is retained. This model continues to track future readings, thus emerging as the sole fault hypothesis.

3 Discussion

The water heater example shows how, with few observable variables, MIMIC can diagnose a system by observing its dynamic behavior. In general, the speed at which a diagnosis can be narrowed depends on the number of monitored variables and the dynamic activity of the system. With more monitored variables and more system activity, there are more opportunities to refute incorrect hypotheses.

Diagnostic systems often rank competing hypotheses by probability, based on the *a priori* fault probabilities of the components. Because MIMIC does continuous monitoring, it can also rank hypotheses by *age*. The longer that a hypothesis survives, continuing to track the changing observations, the stronger the evidence supporting that hypothesis. This age-ranking also desirably focuses attention on hypotheses that account for the *earliest* manifestations of a fault, before numerous other manifestations (and corresponding

[5] Due to the one-fault-at-a-time assumption, the double-fault bad-H1, bad-H2 is not hypothesized. In a more detailed example, other hypotheses would also be proposed, such as a faulty temperature sensor, faulty flow sensor, and faulty thermostat, since all are upstream of the temperature discrepancy.

hypotheses) appear. In short, the natural time delays in the system help in identifying the correct hypothesis.

Alarms are treated in a new way in Mimic in that they are based primarily on the predictions of the model(s) in the tracking set. This has several nice consequences: alarm thresholds can be dynamic rather than fixed, thus allowing earlier alerting; alarms can be based on unobserved variables, permitting more freedom in alarm design; alarms can reveal *any* mutually inconsistent readings (extreme analytical redundancy); alarms, called *forewarnings*, can be based on near-future predicted states; and false alarms due to operating-mode changes (*e.g.,* startup/shutdown) should not occur if the model faithfully predicts such dynamic behavior.

4 Limitations

If Mimic cannot quickly refute invalid hypotheses, the tracking set will grow and Mimic will slow down correspondingly. Mimic refutes hypotheses through tracking, which means that there must be an observed discrepancy between the model's predictions and the sensor readings. In practice, this means that the model's predictions must be reasonably well-bounded and that there must be an adequate number of well-placed sensors.

Mimic assumes that faults occur one-at-a-time. More precisely, it assumes that the manifestations of different faults appear at different times with respect to its sampling rate. This assumption may be violated in the case of a catastrophic event (such as an explosion) or cascading faults, where discrepancies due to more than one fault may appear simultaneously.

Qualitative simulation can predict spurious behaviors, *i.e.*, behaviors that do not occur in the physical system. This means, for example, that a real fault could go undetected if its behavior happened to correspond to a spurious behavior. Prediction of spurious behaviors is due to an interaction between the qualitative level of description and the local state-to-state perspective of the simulation algorithm. However, this problem has been substantially reduced in qualitative simulation by the introduction of several *global constraints* — constraints that eliminate spurious behaviors through global consistency checks, such as the non-intersection constraint applied to trajectories in qualitative phase space [22], the automatic derivation of energy constraints by recognizing conservative and non-conservative forces [23], and the use of higher-order derivative constraints [24].

The QSIM algorithm guarantees that all behaviors are predicted, and only under a qualitative level of description does this give a tractable set of possibilities. In simple cases (such as the water heater) this is tractable in practice as well as in theory. For more complex systems, controlling the size of the hypothesis set is still a potential problem.

5 Related Work

Kay [25] has demonstrated the MIMIC approach in monitoring the pump-down phase of a vacuum system for semiconductor fabrication, where ultra-high vacuums are required (10^{-9} Torr). Since there is no practical theory for the sorption of gases, it is difficult to model the process numerically. Kay's semi-quantitative model, with dynamic envelopes that bounded the expected observations, permitted reasoning with uncertainties and still achieved detection of faults early in the pump-down phase.

Abbott's approach to monitoring and diagnosis [32], like sc Mimic, takes advantage of the *sequence* in which symptoms appear, although the mechanisms are somewhat different. DRAPHYS detects symptoms (discrepancies) by comparing sensor readings to expected values computed from a numerical simulation model of the fault-free system. Fault hypotheses are then generated by tracing *upstream* from the symptom through a graph model of the paths of interaction among components (dependency-tracing through a structural model). As new symptoms appear, DRAPHYS tests each existing hypothesis to see if propagating its effects further *downstream* in the graph model covers the new symptoms. This latter step is akin to MIMIC's tracking, but at a more abstract level. Specifically, the graph model in DRAPHYS represents only that a fault in one component may affect another component; there is no information about whether the affected sensor reading should be high or low, and there is no information about time delays in fault propagation. Such information can be used to refute some hypotheses, which MIMIC can do because the semi-quantitative model that it uses during tracking provides such information.

Isermann's model-based approach to process fault diagnosis [9], like Mimic, uses dynamic mathematical models and measurable input and output signals to allow estimation of unmeasurable internal quantities, which can then be used for fault detection. Unlike Mimic, however, Isermann's models are purely quantitative and are expected to "describe the process behavior precisely." The resulting approximate-matching problem (to determine if an observation is "normal") is handled with a Bayes decision algorithm. After a symptom is recognized (through Bayes decision), the fault is classified by comparison with fault signatures, which have been established a priori. Although Isermann's work uses different methods for simulation and measurement interpretation and diagnosis, he reaches a similar conclusion that we share: *"dynamic process behavior yields considerably more information on process faults than can be achieved in the static case."*

A number of expert systems have been built which share the same operational goal as Mimic — that of relieving some of the burden of monitoring from process operators [4]. Mimic focuses primarily on determining the state of the physical system, but most of these expert systems have the broader scope of trying to advise the operator on corrective actions. Escort [2], an exemplar of this group, gets its knowledge of faults and anomalies and corrective actions through the usual process of codifying human expertise in rules; Escort does not encode a predictive model of the physical system as Mimic does.

The version of Mimic described in this article has evolved from an earlier design presented in [26]. In particular, hypothesis generation is now based on dependency tracing rather than on a decision tree, and a more sophisticated form of semi-quantitative simulation is used.

6 Summary

This article has described a method for monitoring and diagnosis of process systems based on three foundational technologies: semi-quantitative simulation, measurement interpretation (tracking), and model-based diagnosis. These technologies have been joined together in a hypothesize-build-simulate-match architecture (Figure 2). Compared to existing methods based on fixed-threshold alarms, fault dictionaries, decision trees, and expert systems, several advantages accrue:

- The physical system is represented in a semi-quantitative model which, unlike a pure numeric model, predicts all possible behaviors that are consistent with the incomplete/imprecise knowledge of the system's devices and processes. This ensures, for example, that a hazardous-but-infrequent behavior will not be overlooked.

- Imprecise knowledge of parameter values and functional relationships (both linear and non-linear) can be expressed in the semi-quantitative model and used during simulation. This produces a valid range for each variable and eliminates the need for approximate-matching of observations to predictions.

- Incremental simulation of the model in step with incoming sensor readings, with subsequent comparison of observations to predictions, permits earlier fault detection than with fixed-threshold alarms.

- By using a structural model of the plant and tracing upstream from the site of unmatched observations, model-based diagnosis permits efficient generation of fault candidates without resort to pre-compiled (and often incomplete) symptom–fault patterns.

- By injecting a hypothesized fault into the model and tracking its predictions against observations, the dynamic behavior of the plant is exploited to corroborate or refute hypotheses.

- By simulating ahead in time from the current state, an operator can be forewarned of nearby undesirable states that the plant might enter. Similarly, the effects of proposed control actions can be determined by simulating from the current state.

The three technologies that MIMIC builds upon continue to be active areas of research, and MIMIC stands to inherit the benefits of this research. For example, recent research by Berleant and Kuipers [18] and Kay and Kuipers [27] have improved quantitative reasoning mechanisms to provider tighter bounds on predictions of semi-quantitative models. Also, research by Fouché and Kuipers [23] on reasoning about energy has eliminated an important source of spurious behaviors in qualitative simulation.

An important task, not discussed in this article, is the *model-building* task. Model-based reasoning can, and should, be decomposed into a *model-building* task, which creates the semi-quantitative differential equations from a higher-level description of a physical system, and a *simulation* task, which takes the equations and predicts the possible behaviors. Although the example model in this paper was described at the level of semi-quantitative differential equations used in QSIM, it is usually more convenient to describe a model at a higher level of abstraction. For example, the *device ontology* [28] views a system as a collection of interconnected devices (such as tanks, pumps, and pipes), and the *process ontology* [29] views a system as a set of processes (such as liquid flow and heat flow) plus the preconditions that enable each process. Both of these popular ontologies can be compiled into the qualitative mathematics of QSIM [30, 31], but additional work on automated model-building is needed to add partial quantitative information and permit automatic injection of faults.

7 Acknowledgements

We thank the guest editors and anonymous referees for pointing out several shortcomings in the earlier version of this article. Daniel Dvorak has been supported by the AT&T Doctoral Support Program. Benjamin Kuipers has been supported in part by NSF grants IRI-8602665, IRI-8905494, and IRI-8904454, by NASA grants NAG 2-507 and NAG 9-200, and by the Texas Advanced Research Program under grant no. 003658175.

References

[1] Charles Perrow. *Normal Accidents.* Basic Books, Inc., New York, 1984.

[2] Paul A. Sachs, Andy M. Paterson, and Michael H. M. Turner. Escort – an expert system for complex operations in real time. *Expert Systems*, 3(1):22–29, January 1986.

[3] Robert A. Touchton and Mike Casella. Reactor emergency action level monitor: a real-time expert system. In *Instrument Society of America Convention*, October 1986.

[4] Daniel Dvorak. Expert systems for monitoring and control. Technical Report AI87-55, Department of Computer Sciences, The University of Texas at Austin, May 1987.

[5] Thomas J. Laffey, Preston A. Cox, James L. Schmidt, Simon M. Kao, and Jackson Y. Read. Real-Time Knowledge-Based Systems. *AI Magazine*, **9**(1), 1988.

[6] Peter J. Denning. Towards a science of expert systems. *IEEE Expert*, 1(2):80–83, Summer 1986.

[7] Ron Patton, Paul Frank and Robert Clark (editors). *Fault Diagnosis in Dynamic Systems: Theory and Applications.* Prentice Hall, Englewood Cliffs, NJ, 1989.

[8] Randall Davis and Walter Hamscher. Model-based Reasoning: Troubleshooting. Chapter 8 of *Exploring Artificial Intelligence: Survey Talks from the National Conferences on Artificial Intelligence*, Howard E. Schrobe (editor). Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1988.

[9] Rolf Isermann. Process fault diagnosis based on dynamic models and parameter estimation methods, chapter 7 of *Fault Diagnosis in Dynamic Systems: Theory and Applications.* Prentice Hall, Englewood Cliffs, NJ, 1989.

[10] David Dalle Molle. Qualitative simulation of dynamic chemical processes. Doctoral dissertation, The University of Texas at Austin, 1989.

[11] Olayiwola O. Oyeleye and Mark A. Kramer. Qualitative simulation of Chemical Process Systems: Steady-State Analysis. *AIChE Journal*, Vol. 34, No. 9, September 1988.

[12] V. Venkatasubramanian and S. H. Rich. An object-oriented two-tier architecture for integrating compiled and deep-level knowledge for process diagnosis. *Comput. chem. Engng.* Vol. 12, No. 9/10, pp. 903–921, 1988.

[13] Qiang Shen and Roy Leitch. Synchronized qualitative simulation in diagnosis. To appear in the working papers of the *1991 Workshop on Qualitative Reasoning*, Austin, Texas, May 1991.

[14] Benjamin Kuipers. Qualitative Simulation. *Artificial Intelligence*, 29(3):289–338, September 1986.

[15] Benjamin Kuipers and Daniel Berleant. Using incomplete quantitative knowledge in qualitative reasoning. In *Proceedings of the Seventh National Conference on Artificial Intelligence (AAAI-88)*, pages 324–329, August 1988.

[16] Kenneth D. Forbus. Interpreting measurements of physical systems. In *Proceedings of the Fifth National Conference on Artificial Intelligence (AAAI-86)*, pages 113–117. August 1986.

[17] Dennis DeCoste. Dynamic across-time measurement interpretation. *Proceedings Eighth National Conference on Artificial Intelligence* (AAAI-90), pp 373–379, 1990.

[18] Daniel Berleant and Benjamin Kuipers. Bridging the gap from qualitative to numerical simulation. Submitted to *Artificial Intelligence*, 1991.

[19] Reid Simmons and Randall Davis. Generate, test and debug: combining associational rules and causal models. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence (IJCAI-87)*, pages 1071–1078.

[20] Randall Davis. Form and content in model based reasoning. In the working papers of the *1989 Workshop on Model Based Reasoning*, AAAI, August 1989.

[21] Ethan Scarl. Sensor Failure and Missing Data: further inducements for reasoning with models. In the working papers of *1989 Workshop on Model Based Reasoning*, AAAI, August 1989.

[22] Wood W Lee and Benjamin J Kuipers. "Non-intersection of trajectories in qualitative phase space: a global constraint for qualitative simulation", in *Proceedings of the Seventh National Conference on Artificial Intelligence (AAAI-88)*, pages 286–290.

[23] Pierre Fouché and Benjamin Kuipers. Reasoning about energy in qualitative simulation. To appear in *IEEE Transaction on Systems, Man, and Cybernetics*, 1991.

[24] Benjamin Kuipers, Charles Chiu, David Dalle Molle, and David Throop. Higher-order derivative constraints in qualitative simulation. To appear in *Artificial Intelligence*, 1991. Also available as *Technical Report AI-TR-90-116, The University of Texas at Austin, 1990.*

[25] Herbert Kay. Monitoring and diagnosis of multi-tank flows using qualitative reasoning. Master's thesis, The University of Texas at Austin; Austin, Texas; July 1990.

[26] Daniel Dvorak and Benjamin Kuipers. Model-based monitoring of dynamic systems. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence (IJCAI-89)*, pp. 1238–1243, Detroit, MI, August 1989.

[27] Herbert Kay and Benjamin Kuipers. Numerical behavior envelopes for qualitative models. Submitted to AAAI-91.

[28] Johan de Kleer and John Seely Brown. A Qualitative Physics based on Confluences, in *Qualitative Reasoning about Physical Systems*, Daniel G. Bobrow, ed., (The MIT Press, Cambridge, MA 1985), pp. 7–83. Reprinted from *Artificial Intelligence*, Vol. 24, 1984.

[29] Qualitative Process Theory, in *Qualitative Reasoning about Physical Systems*, Daniel G. Bobrow, ed., (The MIT Press, Cambridge, MA 1985), pp. 85–168. Reprinted from *Artificial Intelligence*, Vol. 24, 1984.

[30] David Franke, Daniel Dvorak, and Benjamin Kuipers. A device ontology for QSIM. Submitted to *Artificial Intelligence.*

[31] James Crawford, Adam Farquhar, and Benjamin Kuipers. QPC: A compiler from physical models into qualitative differential equations. *Proceedings of the Eighth National Conference on Artificial Intelligence*, AAAI-90, pp. 365–372, 1990.

[32] Kathy Abbott. Robust fault diagnosis of physical systems in operation. PhD thesis, Rutgers University, 1990.