Bayesian Fault Detection and Diagnosis in Dynamic Systems

Uri Lerner
Computer Science Dept.
Stanford University
uri@cs.stanford.edu

Ronald Parr
Computer Science Dept.
Stanford University
parr@cs.stanford.edu

Daphne Koller
Computer Science Dept.
Stanford University
koller@cs.stanford.edu

Gautam Biswas
Department of EECS
Vanderbilt University
biswas@vuse.vanderbilt.edu

Abstract

This paper addresses the problem of tracking and diagnosing complex systems with mixtures of discrete and continuous variables. This problem is a difficult one, particularly when the system dynamics are nondeterministic, not all aspects of the system are directly observed, and the sensors are subject to noise. In this paper, we propose a new approach to this task, based on the framework of *hybrid dynamic Bayesian networks (DBN)*. These models contain both continuous variables representing the state of the system and discrete variables representing discrete changes such as failures; they can model a variety of faults, including burst faults, measurement errors, and gradual drifts. We present a novel algorithm for tracking in hybrid DBNs, that deals with the challenges posed by this difficult problem. We demonstrate how the resulting algorithm can be used to detect faults in a complex system.

1 Introduction

The complexity and sophistication of the current generation of industrial processes, and the growing need for autonomous agents that control physical systems, motivate the need for robust online monitoring and diagnosis of complex hybrid systems (e.g., (Isermann 1997) and (McIlraith *et al.* 2000)). We want to monitor the state of the system, reliably detect abnormal behavior, and diagnose the failure. Several approaches have been used for dealing with this problem, but each has its limitations. The traditional model-based schemes for diagnosis and control suffer from computational intractability and numerical convergence problems. The qualitative reasoning mechanisms that dominate this work in the AI community mitigate some of these problems; however, the lack of precision in the representation, and the ambiguities introduced by the reasoning framework can lead them to perform poorly when applied to complex system with continuous dynamics (Hamscher, Console, & de Kleer (eds.) 1992).

In this paper, we propose a different approach to this problem, where we model a complex hybrid system as a *dynamic Bayesian network (DBN)*. This model implicitly defines a probability distribution over projected trajectories of the system state over time. In this sense, it is similar to the very successful *Kalman filter* (Kalman 1960). For systems with linear dynamics and Gaussian noise, the Kalman filter provides an excellent means for tracking system state. Unfortunately, real-life systems are rife with nonlinearities, many of which are expressed as discrete failure modes that can produce discontinuous jumps in system behavior. Hybrid DBNs accommodate a much greater range of problems, including nonlinear dynamics and discrete failure modes that influence system evolution. They can directly represent the noise associated with the system evolution and measurements, as well as the probabilities of faults and their effects.

We first show that many interesting aspects of diagnostic models can be represented in the DBN framework. In particular, we show that they allow a natural encoding of the representation of higher-order system dynamics used in the *temporal causal graph (TCG)* framework of Mosterman and Biswas (1997). In fact, a TCG can be used to provide the skeleton for an appropriate DBN model. We also show that many interesting types of failures can be modeled naturally in the DBN, including *burst faults*, parameter drift, and measurement errors.

There are several advantages to the use of general probabilistic models, such as DBNs, for fault detection and diagnosis. A DBN is complete model of the system. Using this model, the state of the system, including its failure modes, is tracked by maintaining a probability distribution over possible system states given all of the measurements so far. This *belief state* distribution is an exact representation of our best possible beliefs given all of the available evidence. It includes within it the likelihood of different types of failures, as well as a distribution over the relevant system parameters. In principle, many of the issues that have challenged traditional approaches to diagnosis — ranking possible failures, handling of multiple simultaneous failures, and robustness to parameter drift — can be addressed within a probabilistic tracking framework.

Of course, the inference task that is required for maintaining this belief state is a difficult one. Unlike the case of a simple Kalman filter, tracking such systems is generally intractable since the number of modes of these systems grows exponentially over time. We present a novel algorithm that tames this intractability using a combination of several different techniques. We show that this algorithm succeeds in tracking a very difficult scenario on a fairly large system (one involving five tanks). We believe that our approach will scale well to substantially more complex systems.

2 The framework

2.1 Diagnosis of hybrid systems

To ground our discussion, we will focus on the diagnosis task for a class of problems typical of chemical manufac-

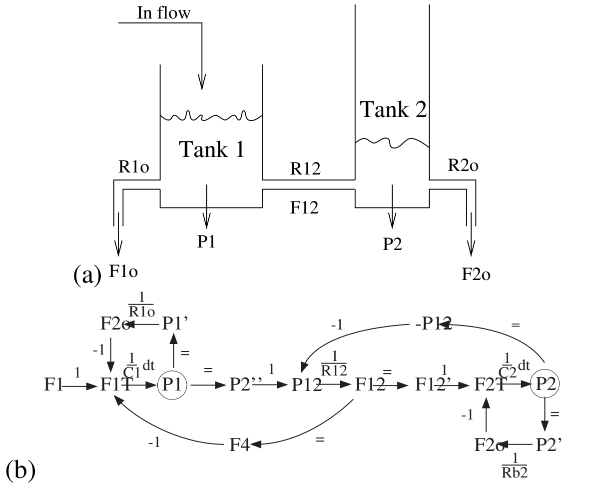

Figure 1: The two-tank system. F indicates flow; P indicates pressure; R indicates Resistance.

turing processes, which involve the transport of materials (mostly fluids) into and between tanks. Such domains are well-represented using bond graph formalism (Rosenberg & Karnopp 1983), where the dynamic behavior of the system is defined by fluid pressures and fluid flow-rates. Consider, for example, a simple two-tank model, shown in Figure 1(a). The model represents a system with two tanks that can hold fluids, an inlet pipe into tank 1, two outlet pipes, and a connecting pipe between the tanks. The storage tanks are *capacitive elements* and the connecting pipes are *resistive elements*. This system is a second order system with natural feedback mechanisms.

The *temporal causal graph (TCG)* framework of (Mosterman & Biswas 1997) is a topological representation that captures local dynamic relations between variables, and provides a more explicit representation of the relation between system parameters and the behavior variables. The TCG for the two-tank example is shown in Figure 1(b). Here, the variables are the pressure and flow-rate variables associated with the tanks and the pipes in the system. Causal edges in a TCG are labeled with component parameter values and temporal information derived from the characteristics of the related components. Resistive and junction components introduce algebraic relations among the system variables, and therefore, define instantaneous temporal relations such as a direct or inverse proportionality between the variables (denoted by ± 1). On the other hand, energy storage elements, like capacitive and inductive elements, introduce integral relations between system variables (labeled with a dt). For example, capacitive relations from the flow-rate variable to the pressure variable are labeled with a $\frac{1}{C}dt$; this implies a temporal relation, i.e., the flow-rate affects the derivative of the corresponding pressure variable.

Many systems have the property that they behave nearly deterministically in the absence of a fault. The deterministic trajectory of the system is often called its *nominal* trajectory. In such cases, faults are sometimes defined implicitly as any abrupt change in a parameter that causes a deviation from the nominal trajectory. Since the temporal causal graph defines a set of qualitative constraints on the system it can be used to predict the effects of sudden discontinuous changes in parameters, e.g., *burst faults*. By contrast, TCGs generally are not used identify *parameter drift* failures, which are the result of gradual changes in system parameters that accumulate over time.

2.2 Dynamic Bayesian Networks

A *dynamic Bayesian network (DBN)* is a temporal stochastic model for a dynamic system. It assumes that the system state can be represented by a set of variables, denoted $\mathbf{Z}$. Each of these variables Z_i can be real-valued or discrete. We use $\mathbf{D}^t \subseteq \mathbf{Z}^t$ to denote the discrete variables in the state. We partition the continuous variables into two subsets: the subset $\mathbf{Y} \subseteq \mathbf{Z}$ are variables that are measurements, i.e., their value is known to us; the remaining subset $\mathbf{X}$ are unobserved.

The system is modeled as evolving in discrete time steps. Thus, each system variable Z has an instantiation Z^t for each *time slice* t. A DBN is a compact graphical representation for the *two-time-slice conditional probability distribution* $P(\mathbf{Z}^{t+1} \mid \mathbf{Z}^t)$. It encompasses both the transition model and the observation model. More formally, a DBN is a directed acyclic graph, whose nodes are random variables in two consecutive time slices: $\mathbf{Z}^t$ and $\mathbf{Z}^{t+1}$. The edges in the graph represent the direct dependence of a time $t+1$ variable Z_i^{t+1} on its immediate causes $\mathrm{Par}(Z_i^{t+1})$. Each such node is annotated with a *conditional probability distribution (CPD)*, that defines the local probability model $P(Z_i^{t+1} \mid \mathrm{Par}(Z_i^{t+1}))$. The DBN model is a compact representation for the two-time-slice distribution via the *chain rule*: $P(\mathbf{Z}^{t+1} \mid \mathbf{Z}^t) = \prod_i P(Z_i^{t+1} \mid \mathrm{Par}(Z_i^{t+1}))$. We note that the transition probabilities for any variable are determined completely by the value of the variables in the current and previous time step. This *Markov* assumption requires us to model explicitly any variables, such as failures, that induce long-term correlations on the system state. We return to this issue below.

For the diagnostic tasks that we focus on, we can restrict attention to a very natural subclass of hybrid DBNs — the *conditional linear Gaussian (CLG)* models. Here, we we require that discrete nodes cannot have continuous parents. We also require that the CPD for a continuous variable be a conditional linear Gaussian. Roughly speaking, in a linear Gaussian dependence, the node is a linear function of its parents with Gaussian noise, where the parameters of the linear dependence can depend on the discrete parents. More precisely, if a node X has continuous parents $Y_1, \ldots, Y_k$ and discrete parents $\mathbf{U}$, we parameterize its CPD using parameters $a_{\mathbf{u},0}, \ldots, a_{\mathbf{u},k}$ and $\sigma^2_{\mathbf{u}}$ for every instantiation $\mathbf{u}$ to the discrete parents $\mathbf{U}$. Then $P(X \mid \mathbf{y}, \mathbf{u})$ is a Gaussian distribution with a mean $a_{\mathbf{u},0} + \sum_{i=1}^k a_{\mathbf{u},i} y_i$ and variance $\sigma^2_{\mathbf{u}}$.

It is important to note that, without discrete variables in the network, this type of DBN defines standard linear Gaussian dynamics. Hence, in this case, the DBN is simply a graphical representation of the standard dynamics used in a Kalman filter, albeit one that makes certain independence

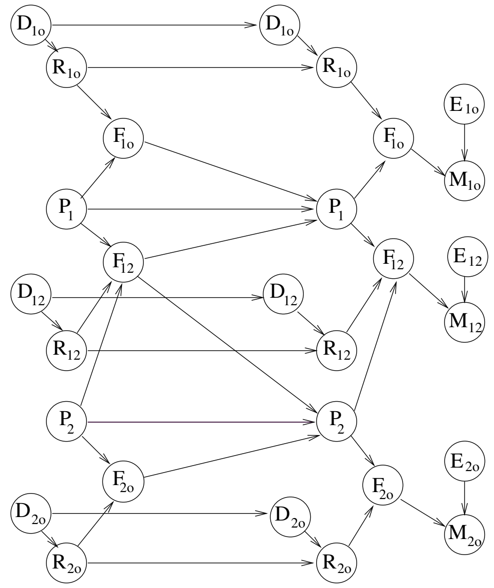

Figure 2: The two-tank DBN.

assumptions explicit. In the presence of discrete parents, the model represents a mixture of linear models, with the mixtures determined by the discrete variables.

2.3 DBNs for diagnosis

Our goal is to represent a diagnostic system, of the type described above, as a DBN. It turns out that we can use a TCG for a system as a "blueprint" for the skeleton of the DBN. We can think of a TCG as a schema for a system of equations describing the continuous system dynamics. We distinguish two types of arcs in a TCG: *temporal* arcs are annotated with a dt, whereas *non-temporal* arcs are not. For any variable X with no incoming temporal arcs, the TCG expresses an instantaneous constraint on X as a function of its predecessors. For a variable X with at least one incoming temporal arc, the TCG expresses a temporal constraint.

We generate a DBN structure from a TCG as follows: For each node X_i in the temporal causal graph, we create X_i^t and X_i^{t+1} to denote the state of the variable at two consecutive time points. (In practice, we will merge nodes that are connected by equality constraints in the TCG.) Let X_j be a node in the TCG which is a direct predecessor of X_i. If the arc from X_j to X_i is non-temporal, we add an arc from X_j^t to X_i^t and an arc fromX_j^{t+1} to X_i^{t+1}. If the arc is temporal, we add an arc only from X_j^t to X_i^{t+1}. This process suffices to generate the structure for a DBN that models the nominal behavior the system.

We then want to add variables that model our observations and represent the failure modes of the system. Our framework accommodates for a wide variety of failure modes. In our presentation, we focus on three important types: burst failures, measurement failures and parameter drift failures. To accommodate these, we need to make two important additions to the TCG induced DBN. Since any parameter that can change must be modeled in the DBN, we add nodes to model the resistance variables. In our implementation, these were conductances and not resistances, since we preferred to use a multiplicative model. We also need to add nodes corresponding to presence of burst failures and the presence of measurement failures.

Figure 2 shows a DBN created by this process. The nodes F_{1t} and F_{2t} simply add incoming flows and this function has been subsumed by the CPDs for P_1 and P_2. The nodes labeled with M correspond to our measurements of the flow parameters in the system and the discrete nodes labeled with E indicate the presence of measurement failures. For example, we define the CPD of M_{12} to be a normal distribution around F_{12} with small variance when E_{12} is false, but with a much larger variance when the E_{12} is true. The R variables model the conductances in the system. These have discrete parents, D, that indicate the presence of faults. Unlike the measurement fault variables, these fault variables have parents in the previous DBN time slice. This is necessary to model persistent events such as drifts. Each conductance fault variable takes on four values: *stable*,*fault*, *buildup* and *leak*. When the system is stable, the CPD of the corresponding R has low noise. When a fault occurs, there is a sharp increase in the variance of the corresponding R. The two drift faults produce a small drift, defined as a percentage of the parameter's previous value. We need the temporal connection between the D nodes to reflect the fact that drifts persist; once a buildup starts in a pipe it tends to continue.

3 Inference

In this section, we propose an inference procedure for fault diagnosis and detection in models represented as DBNs. As we have mentioned, we can view DBNs as a structured representation and extension of traditional Kalman filters. We therefore build our algorithm starting from the classical Kalman filter algorithm. Typical extensions to this algorithm maintain multiple candidate hypotheses about the state of the system. At each time step they update a set of candidate hypotheses and prune out unlikely ones based upon evidence. If the correct hypothesis remains in the candidate set, these algorithms will track the state of the system correctly.

The problem with this type of approach is that it is very difficult to determine which hypotheses to keep for complex systems: there are too many possible new hypotheses at each time, and the information needed to prune away bad hypotheses often is not manifested until several time steps after the hypotheses are generated. We present a novel approach that collapses similar hypotheses into a single hypothesis, then present a novel approximate smoothing algorithm that we use to improve our ability to effectively reduce the number of hypotheses. This approach allows us to deal with complex failure modes and sequences involving many failures. But it does not scale to complex systems that involve many possible failures in different components. We address this problem by combining our techniques with a decomposition method based on the algorithm of Boyen and Koller (1998) that allows the tracking of very large systems.

3.1 Tracking and smoothing

Our dynamic Bayesian network represents the complete state of the system at each time step; it includes variables for the various aspects of the continuous state of the system such as pressures or conductances, as well as discrete variables representing possible failures. This complete model allows us to reduce the problems of fault detection and diagnosis to the task of *tracking* (or *filtering*) a stochastic dynamic system. The tracking problem is defined as follows. As the system evolves, we get observations $\mathbf{y}^1, \mathbf{y}^2, \ldots$. At time t, our most informed evaluation of the state of the system is our posterior distribution $P(\mathbf{Z}^t \mid \mathbf{y}^1, \ldots, \mathbf{y}^t)$ about the current system state given all of our observations so far. We call this posterior distribution our *time t belief state*, and denote it using σ^t. The probability of a discrete fault variable in this belief state takes into consideration all of the evidence up to the present to determine the probability that this fault has occurred.

In principle, tracking is a very easy task, which can be accomplished by the following propagation formula:

$$\sigma^{t+1}(\mathbf{z}^{t+1}) = \alpha P(\mathbf{y}^{t+1} \mid \mathbf{z}^{t+1}) \int \sigma^t(\mathbf{z}^t) P(\mathbf{z}^{t+1} \mid \mathbf{z}^t) d\mathbf{z}^t$$

where α is a normalizing constant. This process is known as a *forward pass*.

Forward tracking gives the best estimate of the likelihood of a fault given the evidence so far. It cannot, however, deal with cases where a fault is momentary, but whose direct effects are unobservable so that its effects become visible to our sensor only later on. The reason is that, at the time that the evidence indicates the presence of a previous failure, there is no longer a variable in the belief state that represents the occurrence of that failure. There is a variable denoting this event at an earlier time slice, but the forward pass only maintains beliefs about variables in the current time step. To explicitly discover faults of this type, we need to also reason backwards in time, from our current evidence to the time slice where the fault took place. This process is known as *smoothing*. Given evidence $\mathbf{y}^1, \ldots, \mathbf{y}^{t+\ell}$, we compute $P(\mathbf{Z}^t \mid \mathbf{y}^1, \ldots, \mathbf{y}^t, \ldots, \mathbf{y}^{t+\ell})$. The smoothing process involves a *backward pass* where evidence from $t + \ell$ is transmitted backwards over the intervening time slices, updating each of them. We omit details for lack of space.

One case of enormous practical importance is the case of *linear systems*. These systems are fully continuous, with linear Gaussian CPDs. In this case, $\mathbf{Z}^{t+1}$ is a linear function of $\mathbf{Z}^t$ and $\mathbf{Y}^{t+1}$ is a linear function of $\mathbf{Z}^{t+1}$, both with some added Gaussian noise. In this case, the belief state can be represented exactly as a multivariate Gaussian distribution over $\mathbf{Z}^t$. This is the basis for an elegant tracking algorithm called the Kalman filter (Kalman 1960) which maintains this belief state in closed form as the system evolves.

3.2 Nonlinearities

Unfortunately, often we cannot apply the Kalman filter directly to real-life problems, since many real-life systems are not linear systems. The continuous relationships between variables are often nonlinear and the failure modes of the system are often discrete, introducing discontinuous changes in system parameters. When the system is nonlinear, the belief state is no longer a multivariate Gaussian, and rarely has a compact closed form representation.

Consider our simple two-tank model. Here, we have a product of two random variables: the flow F is the product of the pressure P and the conductance $\frac{1}{R}$. A standard solution to this type of problem is to approximate the nonlinear dynamics with linear dynamics, and then use a standard linear Gaussian model. Thus, we try to get the best approximation for the first and second moments, and ignore the rest. The classical method of linearizing is called the *Extended Kalman Filter* (Bar-Shalom & Fortmann 1988); it approximates the nonlinear function using its second order Taylor series expansion. In our case, the nonlinear function is a product, which is fairly simple, thus we can compute its first and second order moments in closed form.

A far more problematic type of nonlinearity is caused by discrete state changes that influence the continuous system dynamics. For example, a fault might drastically change the conditional mean or variance of a continuous variable such as the conductance. This type of situation is represented in our model via the dependence of the continuous variables $\mathbf{X}$ on the discrete fault variables $\mathbf{D}$.

This type of model creates substantial difficulties for a tracking algorithm. To understand the difficulties, let $\mathbf{d}^1, \ldots, \mathbf{d}^t$ be some particular instantiation of the discrete variables at time $1, \ldots, t$. Given this instantiation, the dynamics of the continuous variables are, once again, linear Gaussian. Hence, the time t belief state, conditioned on $\mathbf{d}^1, \ldots, \mathbf{d}^t$, is a multivariate Gaussian over $\mathbf{X}^t$. The difficulty is that we have one such Gaussian for every single instantiation $\mathbf{d}^1, \ldots, \mathbf{d}^t$. Thus, in order to do exact tracking, we need to maintain a separate hypothesis for every combination of the discrete variables at all times. The number of such hypotheses grows exponentially with the length of the sequence, which is clearly unacceptable.

A classical tracking algorithm which deals with this problem is described in (Bar-Shalom & Fortmann 1988). The main idea is to maintain our belief state as a smaller set of hypotheses, each of which corresponds to a single multivariate Gaussian. The algorithm, applied to our setting, is as follows. It is convenient to introduce the random variable H^t, each of whose values corresponds to one hypothesis. The distribution of H^t corresponds to the likelihood of the hypothesis. When the algorithm does the forward pass, it considers all the combinations of values of H^t and $\mathbf{D}^{t+1}$. The result is a mixture with $K \times |\mathbf{D}|$ components. Each of these new hypotheses is conditioned on the new measurements $\mathbf{Y}^{t+1}$, and using Bayesian conditioning we adjust both the mixture weights and the parameters of the multivariate Gaussians. The algorithm them *prunes* the hypotheses that have low probability, and selects only the most likely ones to be part of the time $t + 1$ belief state.

I our setting, we also wish to maintain values for the persistent discrete state variables, since the state of the system at time $t + 1$ depends on these values at time t. We therefore represent the belief state using a simple graphical model of the form $\mathbf{D}^t \leftarrow H^t \rightarrow \mathbf{X}^t$. Formally, we represent our time

t belief state σ^t as a mixture $\sigma^t(h^t)$ of K hypotheses, each of which is associated with a single multivariate Gaussian $\sigma^t(\mathbf{X}^t \mid h^t)$ and a discrete distribution $\sigma^t(\mathbf{D}^t \mid h^t)$

The deficiency of this algorithm is that it selects some hypotheses exactly, while entirely ignoring others. In many cases, the hypotheses that are maintained all correspond to scenarios that are all close to nominal behavior, and are therefore qualitatively quite similar. By contrast, the pruned hypotheses often correspond to a priori unlikely faults, that can lead to very different behaviors. We therefore propose a new approach where similar hypotheses are collapsed.

Like the pruning algorithm, we start by performing the forward propagation step, defining a set of possible hypotheses $(H^t, \mathbf{D}^{t+1})$; let $\tilde{H}^{t+1}$ be random variable whose values correspond to this larger set of $K \cdot |\mathbf{D}|$ hypotheses. Next, the measurements are introduced, and the result is a mixture distribution τ^{t+1} over $H^t, \mathbf{D}^{t+1}$ and $\mathbf{X}^{t+1}$. Our task is to generate the $t+1$ hypotheses from this mixture.

We define a new set of mixture components H^{t+1}, each of which aggregates several of the values of $\tilde{H}^{t+1}$. The algorithm thereby defines a *collapsing matrix* that is essentially a deterministic CPD $P(H^{t+1} \mid H^t, \mathbf{D}^{t+1})$. This collapsing matrix is used to define the belief state σ^{t+1}, as a weighted average of the mixture components:

$$\sigma^{t+1}(H^{t+1}) = \sum_{\tilde{H}^{t+1}} P(H^{t+1} \mid \tilde{H}^{t+1})\tau^{t+1}(\tilde{H}^{t+1})$$

$$\sigma^{t+1}(\mathbf{D}^{t+1} \mid H^{t+1}) = \sum_{\tilde{H}^{t+1}} P(\tilde{H}^{t+1} \mid H^{t+1})\tau^{t+1}(\mathbf{D}^{t+1} \mid \tilde{H}^{t+1})$$

Finally, we define $\sigma^{t+1}(\mathbf{X}^{t+1} \mid H^t, \mathbf{D}^{t+1})$ to be the closest Gaussian approximation (i.e., the Gaussian that has the same mean and covariances as the mixture) to $\sum_{\tilde{H}^{t+1}} P(\tilde{H}^{t+1} \mid H^{t+1})P(\mathbf{X}^{t+1} \mid \tilde{H}^{t+1})$.

The main remaining question is the choice of which hypotheses to collapse. We use a greedy approach, that takes into consideration both the likelihood of the different hypotheses and their similarity to each other. We sort the different hypotheses by their likelihood.[1] Then, starting from the most likely hypothesis, we find the closest hypothesis to it, and merge the two. Note that the merged hypothesis will have higher probability, so will remain at the top of the list. When there are no hypotheses that are "close enough", we move to the next most likely hypothesis in our list. When we have filled our quota of hypotheses, we collapse all the remaining hypotheses into one, regardless of how close they are. As our distance measure, we use the sum of the two *relative entropies (KL-distances)* (Cover & Thomas 1991) between the Gaussians associated with the hypotheses. We note that we deliberately do not use the weights in determining the distance between hypotheses; otherwise, we would invariably collapse unlikely hypotheses into likely ones, even if they are qualitatively very different.

[1]To reduce CPU time in our implementation, we first removed all hypotheses with extremely low probability (10^{-8}), and then use the merging approach to collapse the rest.

3.3 Smoothing

Both hypothesis collapsing and pruning are myopic methods; they only use evidence observed up to time t. As discussed above, the effects of some failures have a delay, so a failure at time t may not manifest itself in evidence up to time t. Since *a priori* failure probabilities are typically quite low, failures could have very weak support in our belief state. Thus, by the time the data necessary to diagnose the failure are available, the failure track may be lost. The obvious solution to the problem is to pick the likely hypotheses based not only on past and present evidence but also on future evidence; i.e., we want to use weights obtained after some amount of smoothing. However, smoothing requires that we first propagate a belief state forward in time, and this is the very problem we are trying to solve. We break this cycle by using a slightly different method of collapsing hypotheses. Instead of sorting the hypotheses by likelihoods we always collapse the two most similar Gaussians until our mixture is small enough. This may lead to a more aggressive collapsing since we do not have a bound on the maximal KL-distance between two Gaussians that we collapse. We can afford to be more aggressive here since we will not use the results of smoothing to update our continuous variables, but only to guide our hypothesis reduction method.

It remains to show how we do the backward propagation process required for smoothing. The primary difficulty is the correct handling of the continuous part of our belief state approximation. The reason is that after collapsing a mixture of Gaussians, updating the distribution of each component based only on evidence relative to the result of the collapse is a non-trivial problem. Fortunately, we are primarily interested in getting a more informed posterior for the hypothesis variable, since our main goal is simply to identify the most likely hypotheses. The continuous parts will typically track correctly if we identify the correct hypotheses. Therefore, we execute smoothing only for the discrete variables.

The process is now easy; assume that we use a lookahead window of ℓ time slices (thus, the last observation we get to see is $t+\ell+1$). The backward message to time step $t+\ell'$ is simply the probability of $\mathbf{y}^{t+\ell'+1}, \ldots, \mathbf{y}^{t+\ell+1}$ given $H^{t+\ell'}$. This message defines a posterior distribution over $H^{t+\ell'}$, which can be computed using standard methods. We now use our collapsing matrix to compute the effect of this new information on $\mathbf{D}^{t+l'}$ and $H^{t+l'-1}$. In particular, the probabilities of all the components which were collapsed into some $h^{t+\ell'}$ are multiplied by the change in the probability of $h^{t+\ell'}$. This is also intuitively appealing — since all the collapsed components were similar, we should change their probabilities by the same factor. The result is a message to time step $t+\ell'-1$, which is propagated in the same way.

When the process terminates at time step t, we have the probability $P(H^t \mid \mathbf{y}^1, \ldots, \mathbf{y}^{t+\ell+1})$, which we can then use to better guide which hypotheses to eliminate, as well as our collapsing algorithm. We note, however, that the results of smoothing should be used *only* for guiding the approximation. In order to avoid double-counting evidence, it is very important to continue our tracking using our unsmoothed hypothesis weights $\sigma^t(H^t)$.

3.4 Subsystem decomposition

One of the underlying assumptions of the algorithm is that it is feasible to enumerate all the possible instantiations of the discrete variables $\mathbf{D}^t$. Unfortunately, for non-trivial systems, this assumption is often unrealistic. The number of possible instantiations of the discrete variables $\mathbf{D}^t$ grows exponentially with the number of discrete variables in the system. To deal with this problem, we take an approach introduced for discrete systems in (Boyen & Koller 1998). The crucial idea is to make use of the fact that large systems are typically composed of subsystems, and that, while the subsystems are correlated, the interaction between them is often not so strong. Therefore, it might be reasonable to approximate our beliefs about the system using separate beliefs about the subsystems, i.e., using a belief state where they are independent. Note that this approximation is very different from one that ignores the interactions between the subsystems. As we do the propagation, the state of one subsystem can influence the state of another; but we then decouple the correlations resulting from this interaction when we maintain our beliefs about the current system state.

More precisely, we partition the system variables into n disjoint sets, corresponding to the different subsystems. Let $\mathbf{D}_i$ and $\mathbf{X}_i$ be the discrete and continuous variables in subsystem i, respectively. As for the case of a single system, we represent the belief state for each subsystem i as a mixture, represented using a hypothesis variable H_i. We also associate with each subsystem a set of observation variables $\mathbf{Y}_i$, which are the ones that are most relevant to the subsystem.

Our goal is to get a belief state σ_i^t over each subsystem i. Since subsystem i may be influenced directly by some other subsystems, we cannot perform the inference completely in isolation inside subsystem i. Instead, we consider the *extended subsystem* which includes subsystem i, and all the variables from other subsystems which influence it.

Given our belief state representation, it is possible to describe the distribution over the extended subsystem as a mixture of Gaussians. As in the single subsystem case, we can introduce evidence which changes our probability distribution over discrete variables as well as over continuous variables. Note that different extended subsystems may overlap, and after introducing different measurements into these subsystems we may have a different distribution over the shared variables. We synchronize these probabilities using a message-passing algorithm called *calibration* (Lauritzen & Spiegelhalter 1988). As in backward propagation, we only update the discrete variables, not the continuous ones. As a result of this phase, all the discrete variables are updated using all the measurement information. This is important, as outside evidence can be important in determining the likelihood of the different hypotheses.

It is also possible to modify the smoothing algorithm to use the decomposed representation of the belief state. The collapsing is done independently in every subsystem using the same algorithm (and giving a collapsing matrix for every subsystem). The backward messages are used to update the hypothesis variables of each one of the subsystems. The information can be propagated backwards with the collapsing matrices. The only difference is that after this propagation,

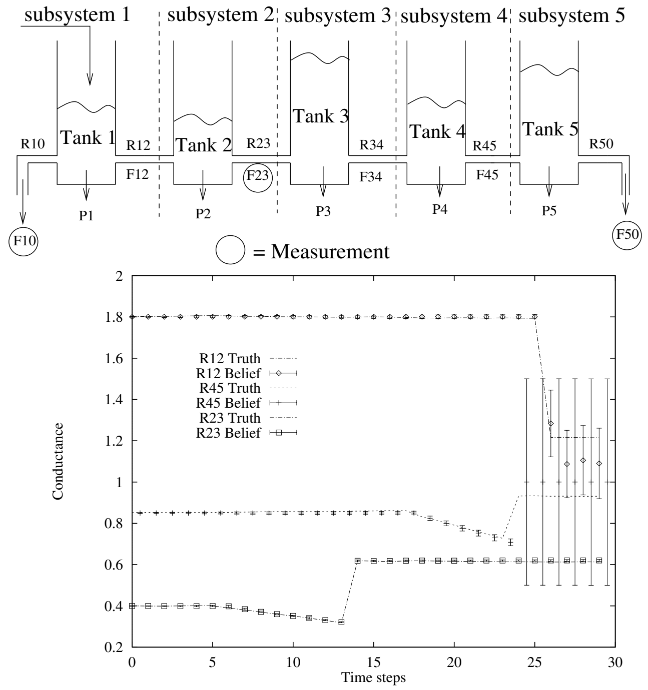

Figure 3: Five tank system and results

we need to calibrate the discrete variables of the subsystems, just like in the forward pass.

4 Experimental Results

We tested our algorithm on a system which contains five water tanks, shown in Figure 3. The system contains six conductances and five pressures, which are all free parameters, but only three measurements, making it a challenging system to track. In addition, the system contains the three types of failures described in Section 2: drifts, bursts and measurement errors, each occurring with probability 0.001. Thus, at every time step every conductance has 4 possible failure modes (stable, fault, buildup, leak) and each measurement has 2 possible failure modes. Counting all the possible failures at time $t + 1$ and the persistent failures from time t, the system has 2^{27} possible failure modes at any point in time, eliminating any hope of using inference without some decomposition of the system.

In our experiments we decomposed the system into five subsystems, since decompositions into less subsystems demanded too much memory. Each tank was considered to be a subsystem (see Figure 3). We tracked five hypotheses per subsystem, with a lookahead of two steps when doing smoothing. We tested our algorithm on a complicated sequence:

- At $t = 5$, R_{23} starts to experience a negative drift.
- At $t = 10$, we introduce two simultaneous measurement errors in the measurements of F_{23} and F_{5o}.
- At $t = 13$, R_{23} bursts, and then returns to a steady state.
- At $t = 17$ R_{45} starts a negative drift.
- At $t = 23$ R_{45} bursts and then returns to normal.
- At $t = 25$ R_{12} bursts.

The graph in Figure 3 shows the results of tracking R_{23},

R_{45} and R_{12}. Initially, at $t = 5$ the effect of the drift in R_{23} was negligible. The corresponding hypothesis had a probability of 0.012%, but after smoothing the probability went up to 7.43%. As a result our algorithm considered this a likely hypothesis, and kept it in the belief state. At $t = 6$ the probability of a negative drift went from 71.7% to 99.9% after smoothing. At this point our algorithm correctly detected the negative drift, and maintained a very high probability for this event until $t = 13$. At this point, before smoothing, our algorithm considered two hypotheses: a burst in R_{23} (probability 57%) or the persistence of the negative drift and a measurement failure (probability 43%). Smoothing raised the probability of a burst (the correct hypothesis) to 100%. The actual values of R_{23} were tracked with high accuracy.

The measurement of F_{23} made the tracking of R_{23} a relatively simple problem. Things are much more complicated for R_{45}. Not only is there no direct measurement of F_{45}, there is no measurement at all in subsystem 4! Therefore, tracking R_{45} is a real challenge. Due to lack of space we omit the actual numbers, but in this run our algorithm detected the drift as soon as it began. (In other runs the detection was sometimes delayed by 2–3 steps.) It is also interesting to see the behavior of R_{45} during the burst. Our algorithm detected the burst, but since no evidence is used in subsystem 4 it could not track the true value of the burst correctly. We plan to address this problem in future work by propagating continuous information between the subsystems as well as discrete information.

For the measurements failures at $t = 10$, our algorithm behaved in almost the same way for the two measurements, so we report on M_{23} only. Before smoothing, our algorithm considered two hypotheses — a burst in R_{23} (probability 81.8%) or a measurement fault and a persistent negative drift in R_{23} (probability 18.2%). After smoothing the probability of the correct hypothesis went up to almost 100%.

We feel that these results demonstrate the power of our algorithm, and its ability to correctly diagnose and track even a complex system with a small number of measurements.

5 Conclusions and future work

In this paper, we presented a new approach for monitoring and diagnosis of hybrid systems. We model these systems as DBNs, thus reducing the problem of diagnosis to the problem of tracking. It is not a surprise that tracking hybrid systems is also a difficult problem. In this paper we focus on a special class of hybrid systems: ones that given some particular assignment to the discrete variables have linear dynamics (or can be linearized with a satisfactory precision). Furthermore, we focus on systems that are composed of several weakly interacting subsystems. We believe that many real-life physical systems belong to this class of systems.

We present a novel tracking algorithm for this class of systems. First, we collapse similar hypotheses instead of just choosing the most likely ones. This technique allows us to use a bounded window look-ahead into the future. We use future observations to assist us in determining which hypotheses are the likely candidates and should be kept relative unchanged, and which are less likely and can be collapsed more aggressively. Our final contribution is introducing a way to avoid the exponential blowup, caused by many discrete variables within a time slice. We do this by reasoning separately about the different subsystems, while still propagating correlations between them.

Our initial experiments with this approach are very encouraging. We have tested it on a very large system (one with 2^{27} different discrete states per time slice), with a particularly difficult scenario. Our algorithm found most of the faults, showing that it can be used to provide reliable tracking and diagnosis even for very hard problems. Of course, we plan to conduct further experiments in other domains.

We are currently working on extending the calibration algorithm to allow us to propagate information between subsystems not only for the discrete variables but for continuous variables as well. We believe that this new feature will significantly improve our tracking capabilities, especially on long sequences with many events.

We are also looking for ways to extend the algorithm beyond the family of conditional linear systems (or systems which can be approximated as such). In particular, we hope to be able to handle discrete children of continuous parents and highly non-linear evidence models.

Finally, we hope to apply our algorithm on real-life applications and not just on synthetic data. We are exploring possible applications in the diagnosis domain, such as monitoring the performance of an engine, as well as application in other domains, such as visual tracking.

Acknowledgments. This research was supported by an ONR Young Investigator Award grant number N00014-99-1-0464 and by ARO under the MURI program, "Integrated Approach to Intelligent Systems," grant number DAAH04-96-1-0341, and by the Terman Foundation.

References

Bar-Shalom, Y., and Fortmann, T. E. 1988. *Tracking and Data Association*. Academic Press.

Boyen, X., and Koller, D. 1998. Tractable inference for complex stochastic processes. In *Proc. UAI*.

Cover, T., and Thomas, J. 1991. *Elements of Information Theory*. Wiley.

Hamscher, W.; Console, L.; and de Kleer (eds.), J. 1992. *Readings in Model-Based Diagnosis*. Morgan Kaufmann.

Isermann, R. 1997. Supervision, fault-detection and fault-diagnosis methods - an introduction. *Control Engineering Practice* 5(5):639–652.

Kalman, R. 1960. A new approach to linear filtering and prediction problems. *J. of Basic Engineering* 82:34–45.

Lauritzen, S., and Spiegelhalter, D. 1988. Local computations with probabilities on graphical structures and their application to expert systems. *J. Roy. Stat. Soc.* B 50.

McIlraith, S.; Biswas, G.; Clancy, D.; and Gupta, V. 2000. Hybrid systems diagnosis. In *Proceedings of Hybrid Systems: Computation and Control*, Lecture Notes in Computer Science. Berlin Heidelberg New York: Springer-Verlag. 282–295.

Mosterman, P. J., and Biswas, G. 1997. Monitoring, prediction, and fault isolation in dynamic physical systems. In *Proc. AAAI-97*, 100–105.

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