Monitorability of Stochastic Dynamical Systems*

A. Prasad Sistla, Miloš Žefran, and Yao Feng

University of Illinois at Chicago
{sistla,mzefran,yfeng9}@uic.edu

Abstract. Monitoring is an important run time correctness checking mechanism. This paper introduces the notions of *monitorability* and *strong monitorability* for partially observable stochastic systems, and gives necessary and sufficient conditions characterizing them. It also presents important decidability and complexity results for checking these properties for finite state systems. Furthermore, it presents general monitoring techniques for the case when systems are modeled as quantized probabilistic hybrid automata, and the properties are specified as safety or liveness automata. Experimental results showing the effectiveness of the methods are given.

1 Introduction

The growing complexity of modern engineered systems and their increased reliance on computation calls for novel approaches to guaranteeing their correct functioning. This is especially important for safety critical systems such as medical devices and transportation systems where a failure can have catastrophic consequences.

One way to ensure correctness of a complex system is to thoroughly test and/or verify it. While testing can increase confidence in a component, it can not guarantee correctness. Verification, on the other hand, can guarantee correctness, but it is simply not feasible, for example, for a car with advanced engine controls and numerous networked microprocessors. In other cases, the component might have been verified for correctness on a model which was not accurate. And more importantly, even if through verification a component is found to be defective, we may still want to use it if the incorrect behavior only occurs rarely.

An alternative to testing and verification is to monitor the behavior of the component at run time. The monitor observes the inputs and outputs of the component and checks whether the behavior of the system is consistent with the expected behavior. Monitors can be especially useful if a fail-safe shutdown procedures can be developed, which is true for a broad class of systems. The fundamental advantage of monitors is that they are in principle easy to implement, and they are independent of the design procedures used to develop a component. While wrong assumptions might lead to a faulty design, the monitor is independent of design decisions and can therefore easily detect that the component is failing to perform its function.

In control systems literature, it is commonly assumed that system behavior is stochastic and the state of the system is not directly observable. Furthermore, in digital control systems the state is typically quantized. We thus consider Hidden

* This research was supported in part by NSF grants IIS-0905593, CNS-0910988 and CNS-1035914.

Markov Chains (HMC) to model such discrete state systems. In our earlier work [10,24], we addressed the problem of monitoring a system, modeled as a HMC H, when the correctness specification is given by a deterministic Streett automaton $\mathcal{A}$ on infinite strings. In these works, we considered *external* monitoring where the automaton $\mathcal{A}$ is defined on the outputs generated by the system. There we defined two measures, called *Acceptance Accuracy* (AA) and *Rejection Accuracy* (RA) that capture the effectiveness of the monitor. Monitoring algorithms for achieving arbitrary high values of accuracies were presented when H and $\mathcal{A}$ are finite state systems. The values $(1 - AA)$ and $(1 - RA)$ are measures of false alarms and missed alarms, respectively, and should be kept low.

In this paper, we consider *internal* monitoring where the automaton $\mathcal{A}$ is specified on the states of the system, not on its outputs. Further, we allow both the system H and the automaton $\mathcal{A}$ to have infinite number of states. Since states are not directly observable, internal monitoring is significantly harder than external monitoring. In this setting, we define two notions of monitorability. We say that a system H is *strongly monitorable* with respect to an automaton $\mathcal{A}$, if there is a monitor such that both of its accuracies are 1. We give a necessary and sufficient condition for strong monitorability. We show that, for finite state systems, the problem of deciding whether a system H is strongly monitorable with respect to an automaton $\mathcal{A}$ is PSPACE-complete.

We also define a more realistic notion called *monitorability*. A system H is said to be *monitorable* with respect to an automaton $\mathcal{A}$ if accuracies arbitrarily close to 1 can be achieved, i.e., for every $x \in [0, 1)$, there is a monitor such that both of its accuracies are greater than or equal to x. We present a fundamental result that exactly characterizes monitorability. It states that a system H is monitorable with respect to $\mathcal{A}$ exactly when the probability measure of the set of so-called 0/1-limit sequences is 1. We show that even for finite state systems, determining monitorability is undecidable; more specifically, it is shown to be r.e.-complete. However, we identify some sufficient conditions for monitorability.

Finally, we consider systems specified as quantized probabilistic hybrid automata [11]. We assume that the correctness automaton, called property automaton, is specified by a non-probabilistic quantized hybrid automaton. We present monitoring algorithms for these cases that employ particle filters for state estimation. When the property automaton $\mathcal{A}$ specifies a safety property on the discrete states of the system, then one can achieve arbitrary high levels of accuracies by appropriately specifying threshold probability parameter. When $\mathcal{A}$ also includes a liveness property, one can achieve high levels of accuracies by approximating it using safety automata with timeouts. High accuracies can be achieved by adjusting the time-out parameters. We implemented these techniques in Matlab and evaluated them for an example of a train equipped with electronically controlled pneumatic (ECP) brakes. Experimental results showing the effectiveness of the monitoring algorithms are presented.

In summary the main contributions of the paper are as follows: (1) a Hidden Markov model for modeling infinite state systems and accuracy measures when the property automaton is specified on system states; (2) definitions of monitorability and strong monitorability, and exact characterizations of systems that

satisfy these properties; PSPACE-completeness result for checking strong monitorability and undecidability for checking monitorability for finite state systems; identification of sufficient conditions, that hold in practice, for monitorability; (3) monitoring algorithms when the system and property automata are specified by probabilistic hybrid automata and hybrid automata, respectively; and (4) experimental results showing the effectiveness of our approach. Due to the space limitations, most proofs have been omitted in the current version; for complete details we refer the reader to [26].

2 Related Work

For external monitoring, it is easy to see that every safety property is strongly monitorable and [10] shows that, for finite state systems, all properties including liveness properties are monitorable. There we employ dynamically increasing timeouts to achieve monitorability of liveness properties. The current paper employs completely different techniques and presents some fundamentally new results on monitorability for internal monitoring.

Several authors consider monitoring temporal properties of deterministic systems capable of measuring system state [3,6,8,20,30]. Some of them define monitorability, but it is only with respect to a property. In contrast, we consider partially observable stochastic systems (i.e., HMCs). In this case, the monitoring problem is significantly more difficult and both the system and the property need to be considered when defining monitorability.

A problem that has been extensively studied is monitoring and diagnosis of hybrid automata [2,4,14,17,29], where the aim is to detect when the automaton enters a fail state so that the system can appropriately react. In most cases, these works employ techniques that depend on the specific possible modes of failure. None of the above works addresses the general problem of monitoring system behaviors against specifications given in an expressive formal system such as the hybrid automata. Furthermore, they do not address the problem of monitoring liveness properties.

Control synthesis for stochastic discrete-event systems has been studied in [15, 18]. In contrast to our work, the authors only consider finite-state systems with directly observable state. Similarly, the literature on diagnosability of partially-observable discrete-event systems (e.g. [31]) only considers deterministic finite-state systems.

A method for monitoring and checking quantitative and probabilistic properties of real-time systems has been given in [23], [21] considers monitoring interfaces for faults using game-theoretical framework, and *conservative* run time monitors were proposed in [16,25]; none of these works is intended for monitoring of hybrid systems.

3 Definitions and Notation

Sequences. Let S be a set. Let $\sigma = s_0, s_1, \ldots$ be a possibly infinite sequence over S. For any $i \geq 0$, $\sigma[0, i]$ denotes the prefix of σ up to s_i. If α_1 is a finite sequence and α_2 is either a finite or an ω-sequence then $\alpha_1\alpha_2$ denotes the concatenation of

the two sequences in that order. We let S^*, S^ω denote the set of finite sequences and the set of infinite sequences over S. If $C \subseteq S^\omega$ and $\alpha \in S^*$ then αC denotes the set $\{\alpha\beta : \beta \in C\}$.

Safety Properties. For any $\sigma \in S^\omega$, let prefixes(σ) denote the set of prefixes of σ and for any $C \subseteq S^\omega$, let prefixes(C) $= \cup_{\sigma \in C}$(prefixes(σ)). We say that $C \subseteq S^\omega$ is a *safety* property if the following condition holds: for any $\sigma \in S^\omega$, if prefixes(σ) $\subseteq$ prefixes(C) then $\sigma \in C$. For any $C \subseteq S^\omega$, let $closure(C)$ be the smallest safety property such that $C \subseteq closure(C)$.

Automata. We consider deterministic Streett automata to specify properties over infinite sequences. Each such automaton $\mathcal{A}$ has an input alphabet Σ and defines a language $L(\mathcal{A}) \subseteq \Sigma^\omega$. These automata can have countable number of states. Throughout sections 3 and 4, an automaton refers to a Streett automaton. We also consider Büchi automata, a subclass of Streett automata, and a subclass of Büchi automata called *safety automata* whose language is a safety property.

Markov Chains. We assume that the reader is familiar with basic probability theory, random variables and Markov chains. We consider stochastic systems given as Markov Chains [19] and monitor their computations for satisfaction of a given property specified by an automaton or a temporal formula. A Markov chain $G = (S, R, \phi)$ is a triple satisfying the following: S is a set of countable states; $R \subseteq S \times S$ is a total binary relation (i.e., for every $s \in S$, there exists some $t \in S$ such that $(s,t) \in R$); and $\phi : R \to (0, 1]$ is a probability function such that for each $s \in S$, $\sum_{(s,t)\in R} \phi((s,t)) = 1$. Note that, for every $(s,t) \in R$, $\phi((s,t))$ is non-zero. Intuitively, if at any time the system is in a state $s \in S$, then in one step, it goes to some state t such that $(s,t) \in R$ with probability $\phi((s,t))$. A finite path p of G is a sequence $s_0, s_1, \ldots, s_n$ of states such that $(s_i, s_{i+1}) \in R$ for $0 \leq i < n$. For any such p, if $n > 0$, then let $\phi(p) = \prod_{0\leq i<n} \phi((s_i, s_{i+1}))$; if $n = 0$ then let $\phi(p) = 1$. An infinite path of G is an infinite sequence of states $s_0, s_1, ...$ such that $\forall i \geq 0$, $(s_i, s_{i+1}) \in R$. We let $Paths(G)$ and $Paths(G, s)$ for any $s \in S$, respectively, denote the set of all infinite paths in G and the set of all infinite paths in G starting from s.

For any Markov chain G, as given above, we define a class $\mathcal{E}_G$ of measurable sets of infinite sequences over S. $\mathcal{E}_G$ is the σ-algebra [19] generated by sets of sequences of the form pS^ω where $p \in S^*$. Now, for any system state $r \in S$, we define a probability function $\mathcal{F}_{G,r}$ defined on $\mathcal{E}_G$ as follows. Intuitively, for any $C \in \mathcal{E}_G$, $\mathcal{F}_{G,r}(C)$ denotes the probability that a sequence of states generated from the system state r, is in C. $\mathcal{F}_{G,r}$ is the unique probability measure satisfying all the probability axioms [19], such that for every $p \in S^*$ and $C = pS^\omega$, if p is the empty sequence then $\mathcal{F}_{G,r}(C) = 1$, if p is a finite path starting from state r then $\mathcal{F}_{G,r}(C) = \phi(p)$, otherwise $\mathcal{F}_{G,r}(C) = 0$.

Although, for convenience, we have considered all sequences in S^ω in defining $\mathcal{E}_G$, sequences that are not paths in G do not contribute to the probability of any $C \in \mathcal{E}_G$, as shown below. Since S is a countable set, it is not difficult to see that $Paths(G)$, $Paths(G, r) \in \mathcal{E}_G$. Further more, for any $C \in \mathcal{E}_G$, it can be shown that $\mathcal{F}_{G,r}(C) = \mathcal{F}_{G,r}(C \cap Paths(G)) = \mathcal{F}_{G,r}(C \cap Paths(G, r))$.

For any $D \in \mathcal{E}_G$, we let $\mathcal{F}_{G,r|D}$ denote the conditional probability function given D; formally, for any $C, D \in \mathcal{E}_G$, $\mathcal{F}_{G,r|D}(C) = \frac{\mathcal{F}_{G,r}(C \cap D)}{\mathcal{F}_{G,r}(D)}$. For any $\alpha \in S^*$ and $C = \alpha S^\omega$, we let $\mathcal{F}_{G,r}(\alpha)$ denote the probability $\mathcal{F}_{G,r}(C)$ and $\mathcal{F}_{G,r|\alpha}$ denote the conditional probability function $\mathcal{F}_{G,r|C}$. For a set $C \subseteq S^*$, we let $\mathcal{F}_{G,r}(C)$ denote $\mathcal{F}_{G,r}(CS^\omega)$.

We will use automata to specify properties over sequences of states of a Markov chain G. The input symbols to the automata are states of G, i.e., members of S. It has been shown that, for any automaton $\mathcal{A}$, $L(\mathcal{A})$ is measurable [28]. We will be interested in monitoring sequences of states of a system modeled by G, i.e., computations generated by G, to ensure that it satisfies the property given by an automaton $\mathcal{A}$. However, the monitor can not observe the actual states of the system.

Hidden Markov Chains. A Hidden Markov Chain (HMC) [5] $H = (G, O, r_0)$ is a triple where $G = (S, R, \phi)$ is a Markov chain, $O : S \to \Sigma$ is the output function and $r_0 \in S$ is the initial state. Intuitively, for any $s \in S$, $O(s)$ is the output generated in state s and this output is generated when ever a transition entering state s is taken. The generated symbols become inputs to the monitor. H is called Hidden Markov chain because one only observes the outputs generated in each state but not the actual state[1]. We extend the output function O to paths of G as follows. For any finite or infinite path $p = s_0, s_1, \ldots, s_i, \ldots$ in G, $O(p) = O(s_0), O(s_1), \ldots, O(s_i), \ldots$. For any finite or infinite sequence α in $\Sigma^* \cup \Sigma^\omega$, we let $O^{-1}(\alpha)$ denote the set of $p \in S^* \cup S^\omega$ such that $O(p) = \alpha$. For any $C' \subseteq \Sigma^* \cup \Sigma^\omega$, we let $O^{-1}(C') = \cup_{\alpha \in C'}(O^{-1}(\alpha))$.

For any HMC H as given above, we define a class $\mathcal{E}_H$ of sets of infinite sequences over Σ and for any $r \in S$, we define a probability measure $\mathcal{F}_{H,r}$ on $\mathcal{E}_H$ as follows. $\mathcal{E}_H$ is the σ-algebra generated by the sets $\alpha\Sigma^\omega$ for $\alpha \in \Sigma^*$. For any system state $r \in S$ and $C' \in \mathcal{E}_H$, $\mathcal{F}_{H,r}(C') = \mathcal{F}_{G,r}(O^{-1}(C'))$. Intuitively, $\mathcal{F}_{H,r}(C')$ denotes the probability that an output sequence generated from the system state r, is in C'.

Quantized Probabilistic Hybrid Automata. Quantized probabilistic hybrid automata (QPHA) are probabilistic hybrid automata [11] whose continuous variables are quantized. Their semantics is given by a HMC, but they provide a convenient formalism for specifying systems. A quantized probabilistic hybrid automaton $\mathcal{A}$ is a tuple $(Q, V, \Delta t, \mathcal{E}, \mathcal{T}, c_0)$ where Q is a finite set of *discrete* states (modes); $V = \{x_q\}_{q \in Q} \cup \{y_q\}_{q \in Q} \cup \{n_q\}_{q \in Q}$ is the finite set of real-valued *continuous*, *output* and *noise* variables, respectively, that will be assumed to be quantized ;Δt is the sampling time; $\mathcal{E}$ is a function that with each $q \in Q$ associates a set $\mathcal{E}(q)$ of difference equations describing the evolution of the continuous state and the output at time $t + \Delta t$ as a function of the state at t and the noise variables; $\mathcal{T}$ is a function that assigns to each $q \in Q$ a set of *transition*

[1] In the traditional definition of HMCs considered in literature, the output of a state can be any symbol in Σ generated with a probability distribution that is specific to the state; since Σ is a countable set, it is not difficult to see that by duplicating each state as many times as there are output symbols, such a HMC can be converted into an equivalent HMC consistent with our model.

triplets (p_{qi}, ϕ_i, ψ_i), where the *guard* ϕ_i is a predicate over the set of continuous and discrete variables, p_{qi} is a probability distribution over *transition target discrete states*, and the *reset relation* ψ_i is a set of assignments that update or reset some of the continuous variables; and c_0 denotes the initial discrete and continuous states of the automaton. If no noise variables are present and for each all transition triplets the transition target state set is a singleton, a QPHA becomes a quantized hybrid automaton (QHA) [1]. A property can be specified if an appropriate acceptance condition is defined for a QHA. In fact, in this case the QHA is equivalent to a Streett automaton.

Within each mode q, the evolution of the QPHA is given by the difference equations. Since the continuous, noise and output variables are assumed to be quantized, these transitions can be interpreted as an HMC. When a guard ϕ_i becomes satisfied, a transition takes place from q to some target mode q' according to the probability distribution p_{qi}. The overall evolution of the QPHA can be thus interpreted as the evolution of an appropriate HMC. See [11, 12] for details. We will use QPHA to model systems. A property can be modeled using QHA by defining an appropriate acceptance condition.

Monitors. A monitor $M : \Sigma^* \to \{0, 1\}$ is a function with the property that, for any $\alpha \in \Sigma^*$, if $M(\alpha) = 0$ then $M(\alpha\beta) = 0$ for every $\beta \in \Sigma^*$. For an $\alpha \in \Sigma^*$, we say that M rejects α, if $M(\alpha) = 0$, otherwise we say M accepts α. Thus if M rejects α then it rejects all its extensions. For an infinite sequence $\sigma \in \Sigma^\omega$, we say that M rejects σ iff there exists a prefix α of σ that is rejected by M; we say M accepts σ if it does not reject it. Let $L(M)$ denote the set of infinite sequences accepted by M. It is not difficult to see that $L(M)$ is a safety property and $O^{-1}(L(M))$ is measurable (it is in $\mathcal{E}_G$).

Note that for a monitor to be implementable, M has to be a computable function as observed in [30] for deterministic systems. We will not further consider this issue in this paper.

Accuracy Measures. Let $\mathcal{A}$ be an automaton on states of H. The *acceptance accuracy* of M for $\mathcal{A}$ with respect to the HMC H, denoted by $AA(M, H, \mathcal{A})$, is the probability $\mathcal{F}_{G,r_0|L(\mathcal{A})}(O^{-1}(L(M)))$ where r_0 is the initial state of H. Intuitively, it is the conditional probability that a sequence generated by the system is accepted by M, given that it is in $L(\mathcal{A})$. We define the *rejection accuracy* of M for $\mathcal{A}$ with respect to H, denoted by $RA(M, H, \mathcal{A})$, to be the probability that a sequence generated by the system is rejected by M, given that it is not in $L(\mathcal{A})$; formally, it is the probability $\mathcal{F}_{G,r_0|C}(D)$, where C, D are the complements of $L(\mathcal{A})$ and $O^{-1}(L(M))$ respectively.

Monitorability. We say that a system H is *strongly monitorable* with respect to an automaton $\mathcal{A}$ if there exists a monitor M such that $AA(M, H, \mathcal{A}) = RA(M, H, \mathcal{A}) = 1$, i.e., both of its accuracies are 1. We say that a system H is *monitorable* with respect to an automaton $\mathcal{A}$ if for every $x \in [0, 1)$ there exists a monitor M such that $AA(M, H, \mathcal{A}) \geq x$ and $RA(M, H, \mathcal{A}) \geq x$. Strong monitorability is a property that is difficult to satisfy. In the next section, we give necessary and sufficient conditions for these properties to be satisfied.

It is worth noting that monitorability, while related to the classical notion of observability, is fundamentally different from it. It is not difficult to construct hybrid systems that are not observable or even discrete-state observable but are monitorable.

4 Conditions for Monitorability

In this section we give necessary and sufficient conditions for strong monitorability and monitorability. Let $H = (G, O, r_0)$ be a HMC where $G = (S, R, \phi)$ is the associated Markov chain. Let $\mathcal{A}$ be an automaton with input alphabet S. $H, G, \mathcal{A}$ are fixed throughout this section unless otherwise stated.

4.1 Strong Monitorability

We define a set of infinite paths $OverlapSeq(H, \mathcal{A})$ that intuitively captures non-trivial overlap, based on the generated outputs, between sets of infinite paths of G that are accepted and those that are rejected by $\mathcal{A}$. We say that a finite path p in G is *good* if it starts from r_0 and the set C of infinite paths, accepted by $\mathcal{A}$, having p as a prefix, has non-zero measure, i.e., $\mathcal{F}_{G,r_0}(C) > 0$ where $C = (pS^\omega \cap Paths(G, r_0) \cap L(\mathcal{A}))$. Let $GoodPaths(H, \mathcal{A})$ be the set of infinite paths in G having only good prefixes. Now we define $OverlapSeq(H, \mathcal{A}) = (Paths(G, r_0) - L(\mathcal{A})) \cap O^{-1}(O(GoodPaths(H, \mathcal{A})))$. Intuitively, $OverlapSeq(H, \mathcal{A})$ is the set of $p \in Paths(G, r_0)$ such that p is rejected by $\mathcal{A}$ and each of its prefix generates the same output sequence as some good path in G, i.e., it can not be distinguished from a good path based on the outputs.

The following theorem gives a necessary and sufficient condition for strong monitorability.

Theorem 1. *Let $H = (G, O, r_0)$ be a hidden Markov chain where $G = (S, R, \phi)$ is the associated Markov chain. Let $\mathcal{A}$ be an automaton with input alphabet S. H is strongly monitorable with respect to $\mathcal{A}$ iff $\mathcal{F}_{G,r_0}(OverlapSeq(H, \mathcal{A})) = 0$.*

The lower bound in next theorem is proved by reduction from the non-universality problem for non-deterministic safety automata:

Theorem 2. *Given a finite HMC H and a finite state automaton $\mathcal{A}$, the problem of determining if H is strongly monitorable with respect to $\mathcal{A}$ is PSPACE-complete.*

If H is strongly monitorable with respect to $\mathcal{A}$, using the techniques employed in the proof of the above theorem, we can construct a monitor M' both of whose accuracies equal 1. M' simply constructs a deterministic automaton $\mathcal{C}$ and runs it on the output generated by H. It rejects iff $\mathcal{C}$ rejects. M' does not estimate any probabilities.

4.2 Monitorability

Consider any $\alpha \in \Sigma^*$. According to our notation $\mathcal{F}_{H,r}(\alpha)$ is the probability that an output sequence of length n, generated by H from state r, is α. Let $\alpha \in \Sigma^*$ be such that $\mathcal{F}_{H,r}(\alpha) > 0$. Now, we define a probability measure $AccProb(\alpha)$ which is the conditional probability that an execution of the system H that initially generated the output sequence α is accepted by $\mathcal{A}$. Formally, $AccProb(\alpha) =$

$\mathcal{F}_{H,r_0|C}(L(\mathcal{A}))$ where $C = O^{-1}(\alpha)S^\omega$. Let $RejProb(\alpha) = 1 - AccProb(\alpha)$. Observe that $RejProb(\alpha)$ is the conditional probability that an execution of the system that initially generated the output sequence α is rejected by $\mathcal{A}$.

Recall that for any $\beta \in \Sigma^\omega$ and integer $i \geq 0$, $\beta[0,i]$ denotes the prefix of β of length $i+1$. Now, let $OneSeq(H,\mathcal{A})$ be the set of all $\beta \in \Sigma^\omega$ such that $\lim_{i\to\infty} AccProb(\beta[0,i])$ exists and its value is 1. Similarly, let $ZeroSeq(H,\mathcal{A})$ be the set of all $\beta \in \Sigma^\omega$ such that the above limit exists and is equal to 0. Let $ZeroOneSeq(H,\mathcal{A}) = OneSeq(H,\mathcal{A}) \cup ZeroSeq(H,\mathcal{A})$. The following lemma states that the sets $OneSeq(H,\mathcal{A})$ and $ZeroSeq(H,\mathcal{A})$ are measurable. It also states that the measure of those executions of H that generate output sequences in $OneSeq(H,\mathcal{A})$ (resp., sequences in $ZeroSeq(H,\mathcal{A})$) and that are rejected by $\mathcal{A}$ (respectively, accepted by $\mathcal{A}$), is zero.

Lemma 1. *The sets $OneSeq(H,\mathcal{A})$ and $ZeroSeq(H,\mathcal{A})$ are measurable (both are members of $\mathcal{E}_H$). Furthermore,*

$$\mathcal{F}_{G,r_0}(O^{-1}(OneSeq(H,\mathcal{A})) - L(\mathcal{A})) = 0 \quad \textit{and}$$
$$\mathcal{F}_{G,r_0}(O^{-1}(ZeroSeq(H,\mathcal{A})) \cap L(\mathcal{A})) = 0.$$

The following theorem, a central result of the paper, gives a necessary and sufficient condition for the monitorability of H with respect to $\mathcal{A}$. But in addition to providing a characterization of the monitorability, the theorem also provides a method for constructing monitors as explained in Section 4.3.

Theorem 3. *For any HMC H and deterministic Streett automaton $\mathcal{A}$, H is monitorable with respect to $\mathcal{A}$ iff $\mathcal{F}_{H,r_0}(ZeroOneSeq(H,\mathcal{A})) = 1$.*

Proof. Let $H = (G,O,r_0)$ be a HMC where $G = (S,R,\phi)$ is a Markov chain, and $\mathcal{A}$ be a deterministic Streett automaton with input alphabet S. Assume that H is monitorable with respect to $\mathcal{A}$.

Suppose that $\mathcal{F}_{H,r_0}(ZeroOneSeq(H,\mathcal{A})) < 1$. Let $F = \Sigma^\omega - ZeroOneSeq(H,\mathcal{A})$. Clearly $\mathcal{F}_{H,r_0}(F) > 0$. Consider any $\beta \in F$. It should be easy to see that for some $m > 0$, the following property holds (otherwise β is in $(ZeroOneSeq(H,\mathcal{A}))$):

Property 1. $AccProb(\beta[0,i]) < (1 - \frac{1}{2^m})$ for infinitely many values of i and $RejProb(\beta[0,j]) < (1 - \frac{1}{2^m})$ for infinitely many values of j.

For each $m > 0$, define F_m to be the set of sequences $\beta \in F$ such that m is the smallest integer that satisfies Property 1. It is easy to see that the set $\{F_m : m > 0\}$ is a partition of F. It should also be easy to see that $F_m \in \mathcal{E}_H$, i.e., is measurable, for each $m > 0$. Since $\mathcal{F}_{H,r_0}(F) > 0$, it follows that for some $m > 0$, $\mathcal{F}_{H,r_0}(F_m) > 0$. Fix such an m and let $x = \mathcal{F}_{H,r_0}(F_m)$. From Property 1, it can be shown that for any $C \in \mathcal{E}_H$, such that $C \subseteq F_m$ and $y = \mathcal{F}_{H,r_0}(C) > 0$, the following property holds:

Property 2. $\mathcal{F}_{H,r_0}(O^{-1}(C) \cap L(\mathcal{A})) \geq \frac{y}{2^m}$ and $\mathcal{F}_{H,r_0}(O^{-1}(C) - L(\mathcal{A})) \geq \frac{y}{2^m}$.

Now, consider any monitor M. Recall that $L(M)$ is the set of infinite sequences over Σ that are accepted by M. Since $L(M)$ is a safety property, it is easily seen

that $L(M) \in \mathcal{E}_H$, i.e., it is measurable. Now, since $x = \mathcal{F}_{H,r_0}(F_m)$ and $F_m = (F_m \cap L(M)) \cup (F_m - L(M))$, it is the case that either $\mathcal{F}_{H,r_0}(F_m \cap L(M)) \geq \frac{x}{2}$ or $\mathcal{F}_{H,r_0}(F_m - L(M)) \geq \frac{x}{2}$. In the former case, by taking $C = F_m \cap L(M)$ and using Property 2, we see that the measure of bad executions of the system (i.e., those in $S^\omega - L(\mathcal{A})$) that are accepted by the monitor is $\geq \frac{x}{2^{m+1}}$ and in the latter case, by taking $C = (F_m - L(M))$, the measure of good executions of the system that are rejected is $\geq \frac{x}{2^{m+1}}$. Now, let $z = \max\{\mathcal{F}_{H,r_0}(Paths(G, r_0) \cap L(\mathcal{A})), \mathcal{F}_{H,r_0}(Paths(G, r_0) - L(\mathcal{A}))\}$. From the above arguments, we see that for every monitor M, either $RA(M, H, \mathcal{A}) \leq 1 - \frac{x}{z \cdot 2^{m+1}}$ or $AA(M, H, \mathcal{A}) \leq 1 - \frac{x}{z \cdot 2^{m+1}}$. This contradicts our assumption that H is monitorable with respect to $\mathcal{A}$.

Now, assume that $\mathcal{F}_{H,r_0}(ZeroOneSeq(H, \mathcal{A})) = 1$. Let $z \in (0, 1)$. Let $M_z : \Sigma^* \to \{0, 1\}$ be a function such that for any $\alpha \in \Sigma^*$, $M_z(\alpha) = 0$ iff there exists a prefix α' of α such that $RejProb(\alpha') \geq z$. Clearly M_z is a monitor. When extended to infinite sequences $M_z(\beta) = 0$ for every $\beta \in ZeroSeq(H, \mathcal{A})$, i.e., it rejects all of them. The second part of lemma 3 implies that $\mathcal{F}_{G,r_0}(O^{-1}(ZeroSeq(H, \mathcal{A})) \cap L(\mathcal{A})) = 0$, and it can further be deduced that $\mathcal{F}_{G,r_0}((S^\omega - O^{-1}(ZeroSeq(H, \mathcal{A})) - L(\mathcal{A})) = 0$. From these observations, it follows that $RA(M_z, H, \mathcal{A}) = 1$. It should be easy to see that the measure of good executions of H that are rejected by M_z is $\leq \min\{y, 1 - z\}$ where $y = \mathcal{F}_{G,r_0}(Paths(G, r_0) \cap L(\mathcal{A}))$. Therefore, $AA(M_z, H, \mathcal{A}) \geq 1 - \frac{\min\{y, 1-z\}}{y}$. Now, for any given $x \in (0, 1)$, we can chose a value of z such that $AA(M_z, H, \mathcal{A}) \geq x$ and $RA(M_z, H, \mathcal{A}) = 1$. This implies that H is monitorable with respect to $\mathcal{A}$. □

The following result can be obtained by reducing the non-universality problem for Probabilistic Finite State Automata.

Theorem 4. *The problem of deciding if a finite state HMC is monitorable with respect to a finite state automaton is r.e.-complete and hence is undecidable, where r.e. is the set of recursively enumerable sets.*

Remark 1. Theorem 3 generalizes to the case when we replace $\mathcal{A}$ by an arbitrary measurable set $C \in \mathcal{E}_H$ by defining $AA(M, H, C)$ to be simply $\mathcal{F}_{G,r_0|C}(O^{-1}(L(M)))$.

4.3 Monitoring Algorithms

Although, the problem of determining if a HMC is monitorable with respect to an automaton $\mathcal{A}$ for finite state systems, is undecidable, we can give sufficient conditions that ensure monitorability.

Intuitively, $H = (G, O, r_0)$ is going to be monitorable with respect to an automaton $\mathcal{A}$ if the statistics (i.e., probability distributions) of outputs generated in paths (in $Paths(G, r_0)$) that are accepted by $\mathcal{A}$ is different from those generated in paths that are rejected by $\mathcal{A}$. Many times, this property may be known. For example, consider a system that can fail, i.e., can get into any of a set of failure states and once it gets into these states, it remains in these states. Further more, assume that the probability distributions of outputs generated in failure states is different from that in non-failure states. Such systems are monitorable with respect to properties that hold only on computations without failure states.

Consider the HMC given in Figure 1 with initial state s. Its output symbols are a, b. Let $\mathcal{A}$ be the automaton that accepts all paths in which state v appears infinitely often. This HMC is monitorable with respect to $\mathcal{A}$, but is not strongly monitorable with respect to $\mathcal{A}$. The probabilities of generation of a, b are different in the two strongly connected components.

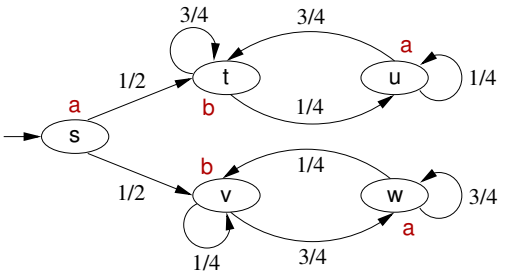

Fig. 1: A HMC that is monitorable.

Assume that H is monitorable with respect to $\mathcal{A}$. Now, we address the problem of constructing accurate monitors for it. The second part of the proof of theorem 4 gives an approach. Here we choose some probability threshold value z. After each output symbol generated by H, if α is the output sequence generated thus far, we compute $RejProb(\alpha)$ and reject (i.e., raise an alarm) if $RejProb(\alpha) \geq z$. Theorem 3 implies that as the threshold probability value z approaches 1[2], we get a sequence of monitors that have accuracies approaching 1. This method requires computation of $RejProb(\alpha)$. Since $\mathcal{A}$ is deterministic, if H and $\mathcal{A}$ are finite state systems then we can construct their product Markov chain and using standard techniques [19] we can compute $RejProb(\alpha)$. For infinite state systems the above approach does not work and it may not be efficient even for finite state systems.

Monitoring Safety Properties. Assume that the system is modeled using a probabilistic hybrid automaton and a property is specified by a safety automaton $\mathcal{A}$. We construct the product of the hybrid automaton model of the system and the automaton $\mathcal{A}$. As the system runs, using an appropriate estimator that uses the product automaton, after each output generated by the actual system, we estimate the probability that the system execution is bad. This is estimated to be the probability that the component denoting the state of $\mathcal{A}$ is an error state in the product automaton, given that it has generated the observed output sequence. If this estimated value is $\geq z$ then we reject. When the system is monitorable with respect to $\mathcal{A}$, we can achieve high accuracies by increasing z.

Monitoring Liveness Properties. Monitoring of properties specified by liveness automata can be achieved using the methods given in [16, 25]. Let $\mathcal{A}$ be a Büchi automaton[3]. We convert $\mathcal{A}$ into a safety automaton $\mathcal{A}'$ by using timeouts. Let T' be positive time out value. $\mathcal{A}$ is modified so that if an accepting state is not reached within T' units from the start or from the last time an accepting

[2] Making $z = 1$ may result in rejection accuracy being 0.
[3] The construction can be easily extended to an arbitrary Streett automaton.

state is reached, then the automaton goes to the error state. It is fairly easy to show that any input sequence that is rejected by $\mathcal{A}$ is also rejected by $\mathcal{A}'$; however $\mathcal{A}'$ rejects more input sequences. Thus, $\mathcal{A}'$ is an approximation of $\mathcal{A}$. Note that we get better approximations by choosing larger values of T'. The above construction can be incorporated by including a *counter* variable in the HA model. The details are straightforward. This approach will be used in the experimental section.

5 Example

Consider the operation of a train with electronically-controlled pneumatic (ECP) brakes [9]. In this case, a braking signal is sent to each of the N cars of the train that subsequently engage their own brakes. We consider the case when the braking systems of individual cars can fail. If this happens to more than a given number of cars ($2N/3$ in our example) the train might not be able to stop and it should start an emergency stopping procedure (for example, engaging the brakes using the traditional pneumatic system). We would therefore like to develop a monitor that can correctly trigger the stopping mechanism in the event when several of the cars have faulty brakes, allowing the train operators to take the advantage of the superior braking performance of the ECP while not sacrificing the safety of the train.

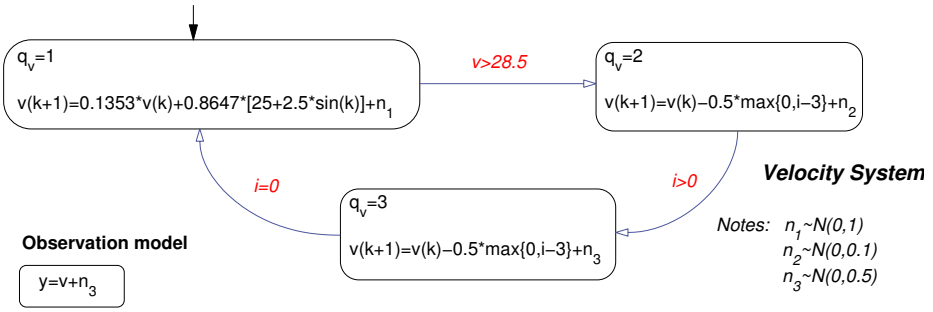

Fig. 2: Velocity subsystem for the train with ECP brakes.

Figure 2 describes how the train velocity v evolves. The train starts in the discrete state $q_v = 1$ and remains in that state until the velocity exceeds a threshold $V_U = 28.5$, when it switches to the discrete state $q_v = 2$. The train remains in the state $q_v = 2$ until one of the brakes engages and it switches to state $q_v = 3$. The velocity in states $q_v = 2$ and $q_v = 3$ depends on the number of brakes that have been engaged through the braking force term $-0.5 \max\{0, i - \lfloor N/3 \rfloor\}$, where i is the number of brakes that have been engaged and N is the number of cars ($N = 10$ for the simulations). Note that in order for the brakes to slow down the train at least $\lfloor N/3 \rfloor$ brakes need to be engaged. When all the brakes disengage, the velocity system switches back to the state $q_v = 1$. When in the state $q_v = 1$, the train accelerates to a constant velocity $V_C = 25$ and oscillates around it with the amplitude 2.5. The measured variable y is assumed to be v, however the measurements are corrupted by a measurement noise n_3. It is

worth noting that the dynamics of the system (and thus statistical properties of output sequences) are different for $q_v = 1$ and $q_v = 3$. The system thus satisfies the sufficient conditions for monitorability given in Section 4.3.

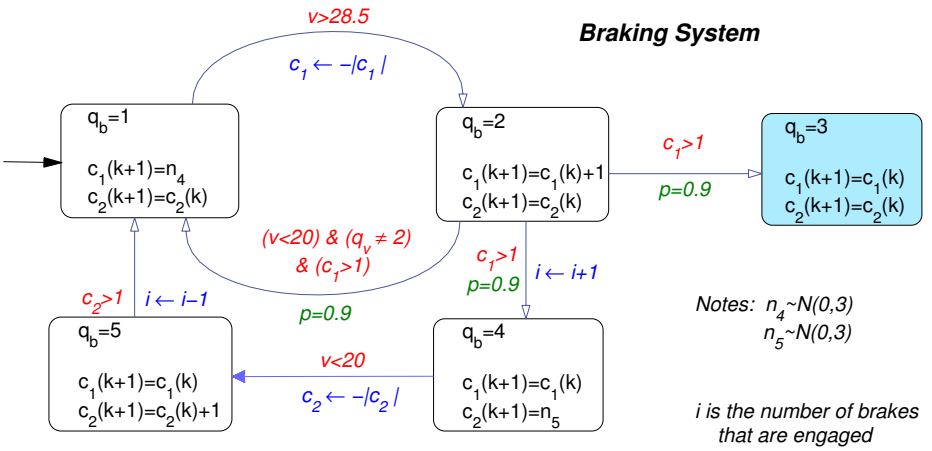

Fig. 3: ECP braking subsystem of each car of the train.

Figure 3 describes the operation of the braking system of each of the cars. The braking system starts in the discrete state $q_b = 1$ and remains in that state until the velocity exceeds a threshold $V_U = 28.5$, when it switches to the discrete state $q_b = 2$. The braking system remains in the state $q_b = 2$ until the timer c_1 reaches $L_1 = 1$ (modeling delays in actuation and computational delays). Note that the initial value of the timer c_1 in the state $q_b = 2$ is not deterministic, so the duration of time the system remains in $q_b = 2$ is a random variable. After the timer reaches L_1, the braking system can fail with a probability $p = 0.1$ and permanently switch to $q_b = 3$. With the probability $p = 0.9$ it either returns to state $q_b = 1$ if the velocity already fell below the threshold $V_L = 20$, or switches to $q_b = 4$ and engages in braking sequence otherwise. When the brake engages the variable i is increased by 1, thereby affecting the velocity of the train as described above. When the velocity falls below $V_L = 20$, the brake disengages after a random amount of time (modeled by the timer c_2 in the state $q_b = 5$), when it switches to the state $q_b = 1$.

Since a braking system is defined for each car, the overall model of the system is roughly a product of N copies of the braking system with the velocity system. For $N = 10$, the number of discrete states of the resulting automaton approaches 30 million. Observe that if the system above is allowed to run forever then all the breaks will eventually fail with probability one. To prevent this, we assume that the brakes can only fail in the first τ units of time. To capture this, we add an additional counter in the breaking subsystem that allows the transition from $q_b = 2$ to the $q_b = 3$ only if this counter is less than τ; this is not shown in the figure. For the simulations, $\tau = 500$.

The desired behavior of the train is given by the following specification: every time the train velocity increases beyond V_U, the train should brake so that the

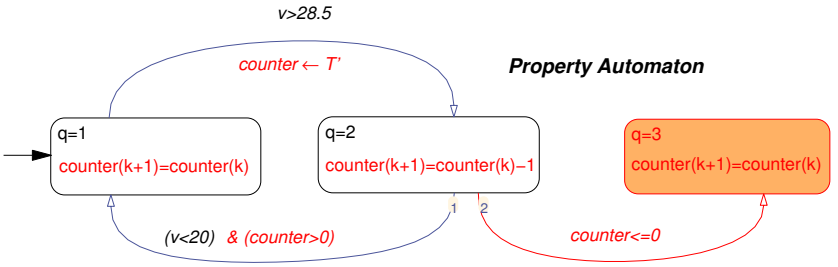

Fig. 4: Property automaton for the train with ECP brakes.

velocity decreases below V_L. This is given by a liveness automaton that has two states $q = 1$ and $q = 2$, and whose initial state as well as the single acceptance state is $q = 1$. This automaton $\mathcal{P}$ is converted to a safety automaton $\mathcal{P}'$ using a static time out T' according to the approach given in Section 4.3. Figure 4 shows the modified automaton $\mathcal{P}'$, where all the modifications of the original liveness automaton are shown in red. Note that $\mathcal{P}'$ has an additional state $q = 3$ which is the error/bad state.

State estimation. Let $\mathcal{S}$ be the system automaton and $\mathcal{P}'$ the modified property automaton given in Figure 4. Note that $\mathcal{P}'$ has a single error/bad state $q = 3$. We construct the product of $\mathcal{S}$ and $\mathcal{P}'$ to obtain the product automaton $\mathcal{S} \times \mathcal{P}'$. Using this product automaton and using the outputs generated by the actual system, our Monitor M estimates the probability that the state component of the property automaton $\mathcal{P}'$ is the bad state. Thus, it becomes necessary to estimate the probability that the property automaton $\mathcal{P}'$ enters the bad state. This can be achieved by propagating the belief (probability distribution over the states of the product automaton) from the current state to the next state, given the new observation [22]. Particle filters were developed as a computationally efficient approximation of the belief propagation [7, 13, 27]. They have been successfully applied in the hybrid system community for state estimation [4,14,17,29]. These methods become impractical for realistic systems with high number of states and several improvements have been suggested in recent years. It is also worth noting that for particle filters, both estimation accuracy and time complexity increase with the number of particles. The exact relationships depend on the structure of the system and transition probabilities and are difficult to characterize. All these issues are beyond the scope of the present paper.

Experimental results. As described in Section 4, the monitor M computes the probability that the property automaton $\mathcal{P}'$ is in a bad state and raises an alarm when this probability surpasses a given threshold P. In order to evaluate the performance of the monitor numerically, the system was run 1000 times. Particle filter was used to estimate the probability of each state of the product automaton $\mathcal{S} \times \mathcal{P}'$. The number of particles for the particle filter was $\eta = 200$. This number was chosen so that particles were not depleted during the transients corresponding to discrete state transitions. The simulation was terminated when either an alarm was raised, or the discrete time (number of steps the system has taken) reached $T_d = 700$. As explained above, the brakes can only fail during the first $\tau = 500$ units of time. For each run, the number of brakes that failed (the

braking system was in $q_b = 3$) was recorded, as well as the state of the monitor. The acceptance and rejection accuracies, respectively, denoted by $AA(M, \mathcal{S}, \mathcal{P}')$ and $RA(M, \mathcal{S}, \mathcal{P}')$ were computed according to:

$$AA(M, \mathcal{S}, \mathcal{P}') = \frac{g_a}{g_a + g_r} \qquad RA(M, \mathcal{S}, \mathcal{P}') = \frac{b_r}{b_a + b_r},$$

where g_a (resp., g_r) is the number of good runs that were accepted (resp., rejected), and b_r (resp., b_a) is the number of bad runs that were rejected (resp., accepted). Note that g_r corresponds to the number of false alarms, and b_a to the number of missed alarms; accuracies approach 1 as these numbers approach 0. A run was considered good if the state of the property automaton at $T_d = 700$ was not equal to 3, and bad otherwise.

z	$T' = 20$					
	g_a	g_r	b_a	b_r	AA	RA
0.050	254	262	0	484	0.492	1.000
0.100	256	222	0	522	0.536	1.000
0.150	256	200	0	544	0.561	1.000
0.300	258	155	0	587	0.625	1.000
0.500	259	117	0	624	0.689	1.000
0.750	259	80	0	661	0.764	1.000
0.875	259	56	0	685	0.822	1.000
0.950	259	43	1	697	0.858	0.999

(a) Monitor outcomes for 1000 runs.

z	$T' = 20$		$T' = 40$		$T' = 60$	
	AA	RA	AA	RA	AA	RA
0.050	0.49	1.00	0.98	1.00	1.00	1.00
0.100	0.54	1.00	0.99	1.00	1.00	1.00
0.150	0.56	1.00	0.99	1.00	1.00	1.00
0.300	0.62	1.00	1.00	1.00	1.00	1.00
0.500	0.69	1.00	1.00	1.00	1.00	1.00
0.750	0.76	1.00	1.00	1.00	1.00	1.00
0.875	0.82	1.00	1.00	1.00	1.00	1.00
0.950	0.86	1.00	1.00	1.00	1.00	1.00

(b) Monitor accuracies.

Table 1: Results of experiments for different values of z and T'.

An example of the monitor performance for $T' = 20$ and different values of the threshold probability z is shown in Table 1a. As expected, if z increases the acceptance accuracy AA increases as it becomes more difficult to reject a run. However, even for $z = 0.95$ the acceptance accuracy does not reach 1 because the small value of T' makes it easy for a run to be rejected. Also, as z increases, the rejection accuracy decreases since it becomes more difficult for the particle filter to estimate that the property automaton entered the fail state with such a high probability.

Table 1b shows accuracy measures of the monitor for different values of the probability threshold z and time out value T'. It can be seen that as z increases the acceptance accuracy increases. Similarly, as T' increases the acceptance accuracy increases. The reason for this is that as z increases, the estimate of the probability that the state of the property automaton $\mathcal{P}'$ is the bad state must be higher before the monitor raises an alarm. Clearly for $z_1 < z_2$, if the monitor with the threshold z_2 would raise an alarm so would the monitor with the threshold z_1, while the reverse is not true. So with lower z, the probability that a false alarm is declared is higher. The same reasoning explains the trends as T' increases. As discussed above, the rejection accuracy may decrease as z increases; similarly, larger T' could lead to more missed alarms. However, these trends can not really be observed in our example due to excellent performance of the particle filter.

6 Conclusion

In this paper we formulated the problem of monitoring both safety and liveness properties for partially observable stochastic systems. Two different notions of monitorability are defined and necessary and sufficient conditions are given for systems to satisfy these properties. Complexity and decidability results for checking these two notions of monitorability for finite state systems are presented. We also presented a general approach for monitoring when the system is specified as a (quantized) probabilistic hybrid automaton and the property is specified as a safety or liveness hybrid automaton. The monitors have been implemented for an example of a train equipped with electronically controlled pneumatic brake. Particle filters were used as state estimators. Experimental results showing the effectiveness of our approach are presented.

References

1. R. Alur, C. Courcoubetis, T. Henzinger, and P. Ho. Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems. In *Hybrid Systems*, volume 736 of *Lecture Notes in Computer Science*, pages 209–229. Springer, 1993.
2. A. Balluchi, L. Benvenuti, M. Di Benedetto, and A. Sangiovanni-Vincentelli. Design of observers for hybrid systems. In *Hybrid Systems: Computation and Control*, volume 2289 of *Lecture Notes in Computer Science*, pages 59–80. Springer, 2002.
3. A. Bauer, M. Leucker, and C. Schallhart. Runtime verification for LTL and TLTL. *ACM Transactions on Software Engineering and Methodology*, 2011.
4. H. Blom and E. Bloem. Particle filtering for stochastic hybrid systems. In *43rd IEEE Conference on Decision and Control, 2004. CDC*, volume 3, 2004.
5. O. Cappe, E. Moulines, and T. Riden. *Inferencing in Hidden Markov Models*. Springer, 2005.
6. M. d'Amorim and G. Roşu. Efficient monitoring of ω-languages. In *Computer Aided Verification*, volume 3576 of *Lecture Notes in Computer Science*, pages 311–318. Springer, 2005.
7. A. Doucet, N. de Freitas, K. Murphy, and S. Russell. Rao-Blackwellised particle filtering for dynamic Bayesian networks. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pages 176–183, 2000.
8. Y. Falcone, J. C. Fernandez, and L. Mounier. Run-time verification of safety-progress properties. In *RV*, volume 5779 of *Lecture Notes in Computer Science*, pages 40–59. Springer, 2009.
9. Federal Railroad Administration. ECP brake system for freight service. `http://www.fra.dot.gov/downloads/safety/ecp_report_20060811.pdf`, 2006.
10. K. Gondi, Y. Patel, and A. Sistla. Monitoring the full range of ω-regular properties of stochastic systems. In *Verification, Model Checking, and Abstract Interpretation*, volume 5403 of *Lecture Notes in Computer Science*, pages 105–119. Springer, 2009.
11. M. Hofbaur and B. Williams. Mode estimation of probabilistic hybrid systems. In *Hybrid Systems: Computation and Control*, volume 2289 of *Lecture Notes in Computer Science*, pages 81–91. Springer, 2002.
12. M. W. Hofbaur. *Hybrid Estimation of Complex Systems*, volume 319 of *Lecture Notes in Control and Information Sciences*. Springer, 2005.
13. M. Isard and A. Blake. Condensation–conditional density propagation for visual tracking. *International journal of computer vision*, 29(1):5–28, 1998.

14. X. Koutsoukos, J. Kurien, and F. Zhao. Estimation of distributed hybrid systems using particle filtering methods. In *Hybrid Systems: Computation and Control*, volume 2623 of *Lecture Notes in Computer Science*, pages 298–313. Springer, 2003.
15. R. Kumar and V. Garg. Control of stochastic discrete event systems modeled by probabilistic languages. *IEEE Transactions on Automatic Control*, 46(4):593–606, 2001.
16. T. Margaria, A. Sistla, B. Steffen, and L. Zuck. Taming interface specifications. In *CONCUR 2005 Concurrency Theory*, volume 3653 of *Lecture Notes in Computer Science*, pages 548–561. Springer, 2005.
17. S. McIlraith, G. Biswas, D. Clancy, and V. Gupta. Hybrid systems diagnosis. In *Hybrid Systems: Computation and Control*, volume 1790 of *Lecture Notes in Computer Science*, pages 282–295. Springer, 2000.
18. V. Pantelic, S. Postma, and M. Lawford. Probabilistic Supervisory Control of Probabilistic Discrete Event Systems. *IEEE Transactions on Automatic Control*, 54(8):2013–2018, 2009.
19. A. Papoulis and S. U. Pillai. *Probability, Random Variables and Stochastic Processes*. McGrawHill, NewYork, 2002.
20. A. Pnueli and A. Zaks. PSL model checking and run-time verification via testers. In *FM*, volume 4085 of *Lecture Notes in Computer Science*, pages 573–586. Springer, 2006.
21. A. Pnueli, A. Zaks, and L. D. Zuck. Monitoring interfaces for faults. In *Proceedings of the 5^{th} Workshop on Runtime Verification (RV'05)*, 2005. To appear in a special issue of ENTCS.
22. S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, second edition, December 2002.
23. U. Sammapun, I. Lee, and O. Sokolsky. Rt-mac:runtime monitoring and checking of quantitative and probabilistic properties. In *Proc. of 11th IEEE International Conference on Embedded and Real-time Computing Systems and Applications (RTCSA 2005)*, pages 147–153, 2005.
24. A. Sistla and A. Srinivas. Monitoring temporal properties of stochastic systems. In *Verification, Model Checking, and Abstract Interpretation*, volume 4905 of *Lecture Notes in Computer Science*, pages 294–308. Springer, 2008.
25. A. Sistla, M. Zhou, and L. Zuck. Monitoring off-the-shelf components. In *Verification, Model Checking, and Abstract Interpretation*, volume 3855 of *Lecture Notes in Computer Science*, pages 222–236. Springer, 2006.
26. A. P. Sistla, M. Žefran, and Y. Feng. Monitorability of stochastic dynamical systems. Technical Report CVRL-2011-01, Computer Vision and Robotics Laboratory, University of Illinois at Chicago, Chicago, IL, 2011. `http://www.cvrl.cs.uic.edu/~milos/publications/papers/cav2011_long.pdf`.
27. S. Thrun, D. Fox, W. Burgard, and F. Dellaert. Robust Monte Carlo localization for mobile robots. *Artificial Intelligence*, 128(1-2):99–141, 2001.
28. M. Vardi. Automatic verification of probabilistic concurrent systems. In *26th annual Symposium on Foundations of Computer Science*, pages 327–338. IEEE Computer Society Press, 1985.
29. V. Verma, G. Gordon, R. Simmons, and S. Thrun. Real-time fault diagnosis. *IEEE Robotics & Automation Magazine*, 11(2):56–66, 2004.
30. M. Viswanathan and M. Kim. Foundations for runtime monitoring of reactive systems - fundamentals of the mac language. In *ICTAC*, volume 3407 of *Lecture Notes in Computer Science*, pages 543–556. Springer, 2004.
31. T. Yoo and S. Lafortune. Polynomial-time verification of diagnosability of partially observed discrete-event systems. *IEEE Transactions on Automatic Control*, 47(9):1491–1495, 2002.