Systems & Control Letters 84 (2015) 21–26

Contents lists available at ScienceDirect

Systems & Control Letters

journal homepage: www.elsevier.com/locate/sysconle

Detectability in stochastic discrete event systems[☆]

Christoforos Keroglou, Christoforos N. Hadjicostis [*]

Department of Electrical and Computer Engineering, University of Cyprus, Cyprus

ARTICLE INFO

Article history:
Received 10 August 2014
Received in revised form
23 June 2015
Accepted 15 July 2015
Available online 24 August 2015

Keywords:
Stochastic discrete event systems
Detectability
Stochastic detectability

ABSTRACT

A discrete event system possesses the property of detectability if it allows an observer to perfectly estimate the current state of the system after a finite number of observed symbols, i.e., detectability captures the ability of an observer to eventually perfectly estimate the system state. In this paper we analyze detectability in stochastic discrete event systems (SDES) that can be modeled as probabilistic finite automata. More specifically, we define the notion of A-detectability, which characterizes our ability to estimate the current state of a given SDES with increasing certainty as we observe more output symbols. The notion of A-detectability is differentiated from previous notions for detectability in SDES because it takes into account the probability of problematic observation sequences (that do not allow us to perfectly deduce the system state), whereas previous notions for detectability in SDES considered each observation sequence that can be generated by the underlying system. We discuss observer-based techniques that can be used to verify A-detectability, and provide associated necessary and sufficient conditions. We also prove that A-detectability is a PSPACE-hard problem.

1. Introduction and motivation

Early instances of state estimation problems in discrete event systems appear in [1,2], both of which formulate the observability problem that requires perfect knowledge of the current state of the system. The state estimation problem is key in many applications involving complex systems. For example, opacity [3,4] requires that a given set of states (with certain properties of interest) remain opaque (non-identifiable) based on the generated sequence of observations, regardless of the underlying activity in the system. Another related application is fault diagnosis [5–7] which requires discrimination (within a finite time interval following the occurrence of a fault) between the set of normal states (states that are possible under normal behavior) and the set of faulty states (states that are possible under faulty behavior), for every possible trace that can be executed in the system; disambiguation between these two sets of states requires state estimation techniques. A similar problem in stochastic discrete event systems (probabilistic finite automata) is the classification between two given models (hidden Markov models or probabilistic finite automata) [8,9]. Classification is closely related to diagnosability if we can treat these models separately (i.e., no transition takes place from a state in one set (which can be thought of as the set of normal states) to a state in the other set (which can be thought of as the set of faulty states, in cases where the fault occurs at system initialization), or vice-versa).

An important task associated with state estimation is that of accurate characterization of the possible (compatible) current states following a (possibly long) observation sequence generated by the underlying discrete event system. In deterministic settings, a key concept is the notion of detectability which was introduced by [10]. In particular, the notion of *strong detectability* holds if all observation sequences lead to an accurate estimate of the current state (perfect knowledge of the system state) after a finite number of observations. Thus, the notion of detectability is primarily determined by finite observation sequences generated by the underlying discrete event system. Extensions of detectability to stochastic discrete event systems were considered in [11] and are discussed later, once we have the opportunity to introduce relevant terminology.

In this paper we are interested in exploring state estimation techniques in stochastic discrete event systems (SDES) that can be modeled by probabilistic finite automata (PFAs) under particular observation models. The authors of [10,11] introduced notions of

☆ This work was partially funded by the Cyprus Research Promotion Foundation (CRPF) Framework Programme for Research, Technological Development and Innovation 2009–2010 (CRPF's FP 2009–2010), co-funded by the Republic of Cyprus and the European Regional Development Fund, under Grant $T\Pi E/OPIZO/0609(BE)/08$. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of CRPF.

* Corresponding author.
E-mail addresses: keroglou.christoforos@ucy.ac.cy (C. Keroglou), chadjic@ucy.ac.cy (C.N. Hadjicostis).

http://dx.doi.org/10.1016/j.sysconle.2015.07.005
0167-6911/

detectability in nondeterministic and stochastic settings respectively. In the approach for detectability in nondeterministic finite automata in [10], the problematic system behavior corresponds to sequences of observations that do not lead to exact state estimation (i.e., they do not lead to perfect state estimation with no uncertainty). In the approach for detectability in PFA's in [11], the problematic behavior is associated with sequences of observations that do not allow us to estimate the exact state with increasing certainty. More specifically, the notion of stochastic detectability in [11] takes into account all possible observation sequences (infinite sequences) and declares the system *not* stochastically detectable when such problematic sequences are present.

The major contribution of this paper is the introduction and verification of the notion of A-detectability. Specifically, we provide necessary and sufficient conditions for A-detectability along with a proof that A-detectability is a PSPACE-hard problem. A-detectability concentrates on highly probable system behavior and characterizes the given system's detectability accordingly. By considering only observation sequences that belong to the recurrent behavior of the system, A-detectability does not take into account observation sequences that are treated as problematic in previous notions of stochastic detectability. Since the approach in the paper also makes connections to notions used in related problems (such as stochastic diagnosis or classification), it could potentially provide insight for future research in these other areas. For example, we prove in this paper that A-detectability is a PSPACE-hard problem, which likely implies that no polynomial verification algorithm exists. This could provide insight about the verification of A-diagnosability, which is the analogous notion in fault diagnosis, and whose possible verification with a polynomial algorithm remains an open problem.

The paper is organized as follows: in Section 2 we revisit notation on automata (nondeterministic finite automata and probabilistic finite automata), languages and Markov chains, and we recall detectability for discrete event systems. In this section we also discuss the verification of detectability in nondeterministic finite automata using either a deterministic observer construction or a detector construction. In Section 3 we introduce the notion of A-detectability and its associated necessary and sufficient conditions, and we establish that A-detectability is a PSPACE-hard problem. During this development of the material we also provide several examples. We conclude in Section 4 with some directions for future research.

2. Detectability for discrete event systems

2.1. Notation on languages and automata

Let Σ be an alphabet (set of events) and denote by Σ^* the set of all finite-length strings of elements of Σ (sequences of events), including the empty string ε (the length of a string s is denoted by $\|s\|$ with $\|\varepsilon\| = 0$). A language $L \subseteq \Sigma^*$ is a subset of finite-length strings in Σ^* [12] (i.e., sequences of events with the convention that the first event appears on the left). Given strings $s, t \in \Sigma^*$, the string st denotes the concatenation of s and t, i.e., the sequence of events captured by s followed by the sequence of events captured by t. For a string s, $\bar{s}$ denotes the *prefix-closure* of s, and is defined as $\bar{s} = \{t \in \Sigma^* \mid \exists t' \in \Sigma^* \{tt' = s\}\}$.

Definition 1 (*Nondeterministic Finite Automaton (NFA)*)**.** A nondeterministic finite automaton is captured by $G = (X, \Sigma, \delta, X_0)$, where $X = \{x_1, x_2, \ldots, x_{|X|}\}$ is the set of states, Σ is the set of events, $\delta : X \times \Sigma \to 2^X$ is the nondeterministic state transition function, and $X_0 \subseteq X$ is the set of possible initial states.

For a set $Q \subseteq X$ and $\sigma \in \Sigma$, we define $\delta(Q, \sigma) = \cup_{q\in Q} \delta(q, \sigma)$; with this notation at hand, the function δ can be extended from the domain $X \times \Sigma$ to the domain $X \times \Sigma^*$ in a routine recursive manner: $\delta(x, \sigma s) := \delta(\delta(x, \sigma), s)$ for $x \in X$, $s \in \Sigma^*$ and $\sigma \in \Sigma$ (note that $\delta(x, \varepsilon) := \{x\}$). The behavior of G is captured by $L(G) := \{s \in \Sigma^* \mid \exists x_0 \in X_0\{\delta(x_0, s) \neq \emptyset\}\}$. We use $L(G, x)$ to denote the set of all traces that originate from state x of G (so that $L(G) = \bigcup_{x_0\in X_0} L(G, x_0)$).

Definition 2 (*Deterministic Finite Automaton (DFA)*)**.** A deterministic finite automaton is captured by $D = (X, \Sigma, \delta, x_0)$, where $X = \{x_1, x_2, \ldots, x_{|X|}\}$ is the set of states, Σ is the set of events, $\delta : X \times \Sigma \to X$ is the (possibly partially defined) state transition function, and $x_0 \in X$ is the initial state.

The function δ can be extended from the domain $X \times \Sigma$ to the domain $X \times \Sigma^*$ in the routine recursive manner:

$$\delta(x, \sigma s) = \begin{cases} \delta(\delta(x, \sigma), s), & \text{if } \delta(x, \sigma) \text{ is defined,} \\ \text{undefined,} & \text{otherwise,} \end{cases}$$

for $x \in X$, $s \in \Sigma^*$ and $\sigma \in \Sigma$ (note that in this case $\delta(x, \varepsilon) := x$). The behavior of D is captured by $L(D) := \{s \in \Sigma^* \mid \delta(x_0, s)$ is defined$\}$.

In general, only a subset Σ_{obs} ($\Sigma_{obs} \subseteq \Sigma$) of the events can be observed, so that Σ is partitioned into the set of observable events Σ_{obs} and the set of unobservable events $\Sigma_{uo} = \Sigma - \Sigma_{obs}$. The natural projection $P_{\Sigma_{obs}} : \Sigma^* \to \Sigma_{obs}^*$ can be used to map any trace executed in the system to the sequence of observations associated with it. This projection is defined recursively as $P_{\Sigma_{obs}}(\sigma s) = P_{\Sigma_{obs}}(\sigma)P_{\Sigma_{obs}}(s)$, $\sigma \in \Sigma, s \in \Sigma^*$, with

$$P_{\Sigma_{obs}}(\sigma) = \begin{cases} \sigma, & \text{if } \sigma \in \Sigma_{obs}, \\ \varepsilon, & \text{if } \sigma \in \Sigma_{uo} \cup \{\varepsilon\}, \end{cases}$$

where ε represents the empty trace [12]. In the sequel, the subscript Σ_{obs} in $P_{\Sigma_{obs}}$ will be dropped when it is clear from context. We denote an observation sequence of length n as $\omega = \omega_1\omega_2 \ldots \omega_n$, where $\forall i$, $\omega_i \in \Sigma_{obs}$.

Definition 3 (*Possible States Following a Sequence of Observations ($R : 2^{|X|} \times \Sigma_{obs}^* \to 2^{|X|}$)*)**.** Suppose that a nondeterministic automaton $G = (X, \Sigma, \delta, X_0)$ starts from a set of possible states $X' \subseteq X$; the set of all possible states after observing $\omega \in \Sigma_o^*$ is $R(X', \omega) = \{x \in X \mid (\exists x' \in X')(\exists s \in \Sigma^*)\{P(s) = \omega \wedge x \in \delta(x', s)\}\}$.

The projection of the language $L(G)$ of a nondeterministic automaton G is defined as $P(L(G)) = \{P(s) \mid s \in L(G)\}$. Note that using Definition 3, the unobservable reach [12] can be expressed as $UR(X') = R(X', \varepsilon)$.

Definition 4 (*Probabilistic Finite Automaton (PFA)*)**.** A stochastic discrete event system (SDES) is modeled in this paper as a probabilistic finite automaton (PFA) $H = (X, \Sigma, p, \pi_0)$, where $X = \{x_1, x_2, \ldots, x_{|X|}\}$ is the set of states, Σ is the set of events, π_0 is the initial-state probability distribution vector, and $p(x_i, \sigma \mid x_{i'})$ is the state transition probability defined for $x_i, x_{i'} \in X$, and $\sigma \in \Sigma$, as the probability that event σ occurs and the system transitions to state x_i given that the system is in state $x_{i'}$.

We can assign a probability to each trace in Σ^* with the interpretation that this value determines the probability of occurrence of this trace: if $\Pr(x_{i'}, s)$ denotes the probability that s is executed in the system and the end state of the system is state $x_{i'}$, then we can define for $\sigma \in \Sigma$, $s \in \Sigma^*$,

$$\left.\begin{aligned} &\Pr(x_i, \epsilon) = \pi_0(x_i) \\ &\Pr(x_i, s\sigma) = \sum_{x_{i'}\in X} p(x_i, \sigma \mid x_{i'}) \Pr(x_{i'}, s) \\ &\Pr(s\sigma) = \sum_{x_i\in X} \Pr(x_i, s\sigma) \end{aligned}\right\}. \tag{1}$$