Covering-Based Supervisory Control of Partially Observed Discrete Event Systems for State Avoidance

R. C. Hill, D. M. Tilbury, and S. Lafortune

Abstract— **In this work we present a new polynomial complexity approach to state avoidance for nondeterministic and partially observed discrete event systems. Our approach generates control based on a covering of the system state space that identifies overlapping sets of indistinguishable states. This approach is shown to be more permissive than existing state-feedback control techniques.**

I. Introduction

State-feedback control approaches base their applied control on the system's current state, rather than on the string of events that took the system to that state. These approaches to control restrict the behavior of a system to a subset of states, thereby solving a state-avoidance problem. These types of approaches have received somewhat limited attention in the discrete event system (DES) literature recently, but deserve renewed attention due to the reduced computational complexity they require for nondeterministic and partially observed systems. Nondeterministic and partially observed systems are becoming more prevalent in the literature as new types of abstraction are employed to reduce model complexity [1] [2]. Also, new results with polynomial complexity have recently been developed for identifying indistinguishable system states [3]. In this paper, we will leverage these results to present a new covering-based approach to state avoidance that generates a less restrictive control law than existing state-feedback approaches to control, while still maintaining polynomial complexity.

Most existing results on the supervisory control of nondeterministic or partially observed DES construct event-feedback controllers using techniques that require exponential complexity in the number of system states [4] [5] [6] [7]. The exponential complexity of these approaches negates much of the advantage originally sought from the use of abstraction in the first place. An alternative approach, therefore, is to employ techniques that construct state-feedback control laws [8] [9]; this is not an exhaustive list and we will in particular focus on the work producing the least restrictive control [9], which we will improve upon. The idea with a state-feedback law is that the control applied at a given state must always be the same, regardless of the actual string of events that took the system there. It is well understood that a static feedback law like this will be more restrictive than a dynamic one in the case of partial observation [10], however, this sacrifice is often worth the associated reduction in computational complexity.

The existing works [8] [9] on state-feedback control under partial observation rely on viewing the state space of a system through a mask that effectively partitions the state space into disjoint sets of equivalent states. Since each state in a given partition cannot be distinguished when observed through the associated mask, the control applied at each of these states must be consistent. In state-feedback control, if two states are respectively reached by two strings with the same observation, then they are always considered indistinguishable, even if other strings exist for which the states can be distinguished. This illustrates how a state-feedback law achieves a reduction in complexity in exchange for more restrictive control as compared to an event-feedback law.

We propose a control law that is based on a covering of the state space, rather than a partitioning. This covering represents overlapping sets of indistinguishable states and has the advantage that if two states are indistinguishable from a third state, they do not necessarily have to be indistinguishable from one another. We go on to prove that our approach generates a less restrictive feedback law than existing works on state-feedback control. Another improvement provided by our approach is that it applies to nondeterministic models. Exponential complexity is avoided by using the recent results of [3] for determining indistinguishable states with polynomial complexity, and by implementing the covering-based control law on-line.

The outline of the rest of this paper is as follows. Section II introduces notation and required concepts. Section III recalls some results from existing research on state-feedback control where a mask is employed. Section IV then introduces our covering-based approach to control and proves that it generates a less restrictive control law. Section V illustrates our approach through a simple example and Section VI summarizes the contributions of this paper.

II. Preliminaries

We will consider DES modeled by possibly nondeterministic automata represented by the four-tuple $G = (X, \Sigma, \delta, x_0)$, where X is the set of states, Σ is the set of events, $\delta : X \times \Sigma \to 2^X$ is the state transition function, and $x_0 \in X$ is the initial state. Let Σ^* be the set of all finite strings of elements of Σ, including the empty string ε. The function δ is extended to $\delta : X \times \Sigma^* \to 2^X$ in the natural way. The notation $\delta(x, s)!$ for any $x \in X$ and any $s \in \Sigma^*$ denotes that $\delta(x, s)$ is nonempty. The notation $\Sigma_G(x)$ will

This work was supported in part by NSF grants CMS-05-28287 and EECS-0624821. All authors are with the University of Michigan, Ann Arbor, MI 48109-2125, USA (rchill@umich.edu; tilbury@umich.edu; stephane@umich.edu).

represent the set of *active events* of state x in automaton G, that is, those events $\sigma \in \Sigma$ for which $\delta(x,\sigma)!$.

In supervisory control of DES, the event set is partitioned into controllable and uncontrollable events $\Sigma = \Sigma_c \dot{\cup} \Sigma_u$ where controllable events can be disabled and uncontrollable events cannot. A *state-feedback supervisory controller* is then a function $f : X \rightarrow 2^{\Sigma}$ that determines the set of events to be enabled based on the current state of the system under control. We will define the closed-loop system f/G as a subautomaton of G that includes those transitions not disabled by f [11]:

$$f/G = (X, \Sigma, \delta_f, x_0) \tag{1}$$

where the function $\delta_f : X \times \Sigma \rightarrow 2^X$ is defined

$$\delta_f(x,\sigma) := \begin{cases} \delta(x,\sigma) & \text{if } \sigma \in f(x) \text{ and } \delta(x,\sigma)! \\ \emptyset & \text{otherwise} \end{cases}$$

Note in the above definition that f/G has the same state set as G, though not all states are reachable in f/G.

It is our goal to find a controller that can keep the behavior of G within a set of "good" states represented by a predicate Q on the state space X. Specifically, $Q : X \rightarrow \{0,1\}$, where a state x that satisfies the predicate is denoted $Q(x) = 1$. Also, let $Q \in \mathbf{P}$, where $\mathbf{P} = \{0,1\}^X$ represents the set of all possible predicates. We also define a partial order on predicates, where $Q_1 \leq Q_2$ if $Q_1(x) \leq Q_2(x)$ for all $x \in X$.

Existence of a state-feedback controller f that satisfies a predicate Q requires a controllability-type property called Σ_u*-invariance*. We will define this property using the notion of the *weakest liberal precondition*, wlp_σ, of a given predicate Q introduced in [11] and defined below. The weakest liberal precondition is a predicate transformer that accepts a state x for a given event σ if either σ is not defined at x, or if all instances of the event σ at x lead to states which are accepted by the predicate Q. The definition presented here is modified to account for the fact that the automaton G is possibly nondeterministic, that is, $\delta(x,\sigma)$ may have more than one element.

$$\begin{aligned} D_\sigma(x) &:= \begin{cases} 1 & \text{if } \delta(x,\sigma)! \\ 0 & \text{otherwise} \end{cases} \\ wp_\sigma(Q)(x) &:= \begin{cases} 1 & \text{if } [D_\sigma(x) = 1] \wedge \\ & \quad [Q(x') = 1, \forall x' \in \delta(x,\sigma)] \\ 0 & \text{otherwise} \end{cases} \\ wlp_\sigma(Q) &:= wp_\sigma(Q) \vee \neg D_\sigma \end{aligned}$$

In the above, $wlp_\sigma(Q)$ is a predicate defined over all $x \in X$. The predicate Q is then defined to be Σ_u-invariant if:

$$Q \leq \bigwedge_{\sigma \in \Sigma_u} wlp_\sigma(Q) \tag{2}$$

In words, Σ_u-invariance means that there are no uncontrollable events that lead from a state that satisfies Q to a state that does not satisfy Q.

If the state space of G is not fully observable, then additional considerations must be addressed. In existing work on state-based control under partial observation, the concept of a "mask" is employed [8] [9]. A mask M is defined as a function $M : X \rightarrow Y$ that maps elements from the state space X to the observation space Y. The idea is that under partial observation two states x and x' might not be distinguishable, that is, $M(x) = M(x') = y$. It is then necessary that the state-feedback control $f(x)$ be determined based on $M(x)$:

$$\text{For any } x, x' \in X, M(x) = M(x') \Rightarrow f(x) = f(x') \tag{3}$$

The set of state-feedback controllers which satisfy (3) is denoted F_o. In existing state-based work it is assumed that the mask M is given. For the purposes of comparison with the approach of this paper, we will assume that the mask is constructed based on the set of observable events. We, therefore, partition Σ into observable and unobservable events, $\Sigma = \Sigma_o \dot{\cup} \Sigma_{uo}$. Under the natural projection $P : \Sigma^* \rightarrow \Sigma_o^*$, unobservable events cannot be "seen," that is, they map to the empty string $\varepsilon = P(\sigma),\ \forall \sigma \in \Sigma_{uo}$. Therefore, the mask M is constructed to satisfy the following constraint:

$$\begin{aligned} &M : X \rightarrow Y \text{ is defined such that if} \\ &\exists s, s' \in \Sigma^* \text{ with } x \in \delta(x_0, s),\ x' \in \delta(x_0, s'), \\ &\quad \text{and } P(s) = P(s'), \text{ then } M(x) = M(x'). \end{aligned} \tag{4}$$

In the above, states $x \in \delta(x_0, s)$ and $x' \in \delta(x_0, s')$ for which $P(s) = P(s')$ are defined to be *indistinguishable*. Therefore, the mask M is constructed such that all indistinguishable states map to the same observation.

III. State-Based Feedback Control

In this section we present results from [9] [12] [13] that relate to the construction of a state-feedback law f that will satisfy a given predicate Q under partial observation such that the requirement of (3) is satisfied. While this review of prior results is necessary for the presentation of our new approach to state-feedback control in Section IV, we also extend this prior work to handle nondeterministic system models.

A. Supervisor Existence

In [12], conditions for supervisor existence under partial observation are presented. These conditions are based in part on sets $A_Q(x) \subseteq \Sigma_c$ that define which events must be disabled at state x given a set of allowable states represented by the predicate Q.

$$\begin{aligned} A_Q(x) = \{&\sigma \in \Sigma_c \mid \exists x' \in X : [M(x) = M(x')] \wedge \\ &[Q(x') = 1] \wedge [wlp_\sigma(Q)(x') = 0]\} \end{aligned} \tag{5}$$

Based on the definition of $A_Q(x)$, a controllable event σ must be disabled at state x if there exists an indistinguishable state x' that satisfies the given predicate Q but not the weakest liberal precondition for σ. This logic also holds for the case $x' = x$, since a state is always considered indistinguishable from itself, that is, $M(x) = M(x)$.

We next define the predicate transformation $R : \mathbf{P} \rightarrow \mathbf{P}$ in the manner of [12]. Let $Q \in \mathbf{P}$ be a predicate. If $Q(x_0) = 0$, then $R(Q)(x) = 0$ for all $x \in X$; otherwise, $R(Q)$ is constructed iteratively as follows:

Algorithm 1: $R(Q)$ Construction

Step 1: Let $R(Q)(x_0) = 1$.
Step 2: If $R(Q)(x) = 1$ and $wp_\sigma(Q)(x) = 1$ for some $\sigma \in \Sigma - A_Q(x)$, then define $R(Q)(x') = 1$ for all $x' \in \delta(x, \sigma)$.
Step 3: Repeat Step 2 iteratively until $R(Q)(x)$ has been set equal to 1 for all states reachable via states that satisfy $R(Q)$. Once all these reachable states have been addressed, set $R(Q)(x) = 0$ for any remaining states. $\diamond$

The predicate transformation R captures those states that satisfy a given predicate and are reachable by transitions not prohibited by the sets $A_Q(x)$. With R defined, we can now introduce the property called *M-controllability* [12].

Definition 1: A predicate $Q \in \mathbf{P}$ is said to be M-controllable if the following two conditions hold:
$C1)$ Q is Σ_u-invariant.
$C2)$ $Q \leq R(Q)$. $\diamond$

The property of M-controllability can then be used for determining existence of a state-feedback controller. Specifically, if we define $Re(f/G) \in \mathbf{P}$ to be the predicate that is true only at all reachable states of f/G, then we have the following theorem from [12]:

Theorem 1: [12] Let $Q \in \mathbf{P}$ be a predicate with $Q(x_0) = 1$. Then there exists a state-feedback controller $f \in F_o$ such that $Re(f/G) = Q$ if and only if Q is M-controllable.

One possible state-feedback law is given by:

$$f(x) = \Sigma - A_Q(x) \tag{6}$$

It is demonstrated in [12] that the state-feedback law given in (6) is the unique maximally permissive control law in F_o for which $Re(f/G) = Q$ when Q is M-controllable. This result and Theorem 1 were originally presented for deterministic models, but they hold for nondeterministic models also.

B. Supervisor Construction

If a given predicate Q is not M-controllable, then [9] and [13] prescribe some approaches for finding a subpredicate of Q that can be achieved by state feedback. A difficulty that arises is that the set of M-controllable subpredicates does not in general possess a supremal element [13]. The work of [13], therefore, defines a new property denoted *strong M-controllability* for which a supremal subpredicate does exist. The supremal subpredicate of Q is denoted $\sup \mathcal{SC}(Q)$ and can be constructed by the algorithm of [13]. In [13] it is further demonstrated that $\sup \mathcal{SC}(Q)$ is larger than the supremal controllable and normal subpredicate of Q presented in [14] and denoted $\sup \mathcal{CN}(Q)$. More precisely,

$$\sup \mathcal{CN}(Q) \leq \sup \mathcal{SC}(Q)$$

The property of normality is defined in Remark 4 of Section V.

In [13] it is demonstrated that $\sup \mathcal{SC}(Q)$ is not in general a maximal M-controllable subpredicate of Q. Conditions are also provided for which an M-controllable subpredicate is maximal, though construction techniques are not presented. Maximal here is defined in terms of the partial order on predicates introduced in Section II.

The work of [9] then presents an approach for constructing an M-controllable subpredicate that while not necessarily maximal, is larger than $\sup \mathcal{SC}(Q)$. Specifically, it is shown:

$$\sup \mathcal{SC}(Q) \leq R(Q^\uparrow)$$

In the above, $Q^\uparrow$ represents the supremal Σ_u-invariant subpredicate of Q constructed according to [11]. Therefore, the feedback law of (6) with Q replaced by $Q^\uparrow$ is able to achieve the subpredicate $R(Q^\uparrow)$.

Taking the above results together, $R(Q^\uparrow)$ represents the largest subpredicate of Q for which a state-feedback law construction algorithm exists in the literature.

IV. Covering-Based Feedback Control

In this section, we propose a new covering-based approach to control that will be shown to be more permissive than those state-based approaches that currently exist in the literature. This advantage will specifically derive from the fact that the requirement of (3) is stronger than necessary. This follows from the character of the mask M. The fact that M is a function implies that when the state space is observed through this mask it is effectively partitioned into disjoint sets of states that have the same observation. For example, if x and x' have the same observation $M(x) = M(x')$, and x' and x'' have the same observation $M(x') = M(x'')$, it then follows that x and x'' must have the same observation $M(x) = M(x'')$. Therefore, all three states x, x', and x'' must be in the same partition of the state space. It may not be necessary, however, that the same control action be applied at x and x'' if they are not indistinguishable. In other words, if the observed string that reaches both x and x' is different than the observed string that reaches both x' and x'', then the control applied at x and x'' may be allowed to be different. Figure 1 illustrates this situation where σ is disabled at x'', but need not be disabled at x since these states are not both reached by the same observed string. In essence, we would like to base our control law on a covering of the state space rather than a partition. If the event σ was possible at state x', then σ would need to be disabled at all three states for our covering-based approach as well.

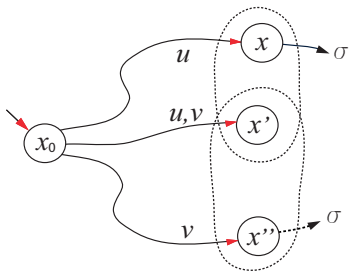

Fig. 1. Example of a covering for indistinguishable states

In order to present our covering-based approach, we will employ the mapping I_Q defined as follows:

Definition 2: Let $I_Q : X \rightarrow 2^X$ be a mapping defined $\forall x, x' \in X$ with $Q(x) = Q(x') = 1$ as follows: $x' \in I_Q(x)$ if x and x' are indistinguishable, that is, if $\exists s, s' \in \Sigma^*$ such

that $x \in \delta(x_0, s)$, $x' \in \delta(x_0, s')$, and $P(s) = P(s')$ where $P : \Sigma^* \to \Sigma_o^*$. $\diamond$

In the above, it is always the case that a state x is considered indistinguishable from itself, that is, $x \in I_Q(x)$.

We can then define new sets of prohibited events, $A'_Q(x) \subseteq \Sigma_c$. At a state x accepted by Q, a controllable transition σ that is defined in the uncontrolled plant G is prohibited if it leads to a state that is not accepted by Q or if it is prohibited at a state x' for which $x' \in I_Q(x)$.

Since the definition of prohibited events for a state x, $A'_Q(x)$, depends on the prohibited events of other states, each set $A'_Q(x)$ is constructed iteratively. In words, if there is a string of indistinguishable states defined:

$$x \in I_Q(x'),\ x' \in I_Q(x''), \ldots,\ x^{(m-1)} \in I_Q(x^{(m)})$$

each with σ possible and such that σ at $x^{(m)}$ leads to a state not accepted by Q, then σ is again added to $A'_Q(x)$. This construction indicates a transitivity similar to that imposed by M, *except that here the transitivity is limited to indistinguishable states where σ is possible*. Assuming the mapping I_Q is given, $A'_Q(x)$ can be constructed as prescribed in Algorithm 2. The mapping I_Q can be constructed with polynomial complexity using results from [3].

Algorithm 2: Prohibited Events Determination
Input: automaton G, predicate Q and mapping I_Q
For each $x \in X$
  For each transition $\sigma \in \Sigma_G(x) \cap \Sigma_c$
    If $[(Q \wedge \neg wlp_\sigma(Q))(x) = 1]$ then
      add σ to the set of prohibited events at x,
      $A'_Q(x) \leftarrow \{\sigma\} \cup A'_Q(x)$.
    End if
    If $\sigma \in A'_Q(x)$ then
      define a set of states T that is initialized with state x, $T \leftarrow \{x\}$. Also let $\mathcal{M} : X \to \{0, 1\}$ be a partial function marking whether or not states in set T have been addressed yet. Set $\mathcal{M}(x) = 0$.
      For each $x' \in T$ with $\mathcal{M}(x') = 0$
        For each $x'' \in I_Q(x')$ that is not in T
          If $\delta(x'', \sigma)!$ then
            add state x'' to set T, $T \leftarrow \{x''\} \cup T$, and add event σ to the set of prohibited events at x'', $A'_Q(x'') \leftarrow \{\sigma\} \cup A'_Q(x'')$. Set $\mathcal{M}(x'') = 0$.
          End if
        End for
        Set $\mathcal{M}(x') = 1$.
      End for
      Clear T and $\mathcal{M}$.
    End if
  End for
End for
Output: the sets $A'_Q(x)$ $\diamond$

Algorithm 2 has complexity $\mathcal{O}(mn^2)$ where m is the cardinality of the event set Σ and n is the cardinality of the state space X. Assuming the automaton G has a finite number of transitions, this algorithm will terminate in finite time. The sets $A'_Q(x)$ defined by Algorithm 2 satisfy the following equation by construction:

$$A'_Q(x) = \{\sigma \in \Sigma_c \mid \delta(x, \sigma)! \wedge (\exists x' \in X) : [x' \in I_Q(x)] \wedge \{[wlp_\sigma(Q)(x') = 0] \vee [\sigma \in A'_Q(x')]\}\} \quad (7)$$

In order to explicitly compare the sets $A_Q(x)$ and $A'_Q(x)$, we now define the following mapping J_Q that reflects the partition implicitly imposed on the state set X by a mask M satisfying (4):

Definition 3: Let $J_Q : X \to 2^X$ be a mapping defined $\forall x, x' \in X$ with $Q(x) = Q(x') = 1$ as follows: $x' \in J_Q(x)$ if $M(x) = M(x')$. $\diamond$

The definition of $A_Q(x)$ given in (5) can then be rewritten in terms of the mapping J_Q as follows:

$$A_Q(x) = \{\sigma \in \Sigma_c \mid (\exists x' \in X) : [x' \in J_Q(x)] \wedge [Q(x') = 1] \wedge [wlp_\sigma(Q)(x') = 0]\} \quad (8)$$

Examining (7) and (8), σ can be an element of $A_Q(x)$ when $\delta(x, \sigma)$ is empty, while it cannot be an element of $A'_Q(x)$. Furthermore, since the mapping J_Q imparts a partition on the state space X, $A_Q(x) = A_Q(x')$ if $x' \in J_Q(x)$. These observations along with the fact that $I_Q(x) \subseteq J_Q(x)$ then implies that $A'_Q(x) \subseteq A_Q(x)$.

The new $A'_Q(x)$ can then be employed to generate a new predicate transformation R'. Let $Q \in \mathbf{P}$ be a predicate. If $Q(x_0) = 0$, then $R'(Q)(x) = 0$ for all $x \in X$; otherwise $R'(Q)$ is constructed iteratively in the same manner as R in Algorithm 1.

Algorithm 3: $R'(Q)$ Construction
Step 1: Let $R'(Q)(x_0) = 1$.
Step 2: If $R'(Q)(x) = 1$ and $wp_\sigma(Q)(x) = 1$ for some $\sigma \in \Sigma - A'_Q(x)$, then define $R'(Q)(x') = 1$ for all $x' \in \delta(x, \sigma)$.
Step 3: Repeat Step 2 iteratively until $R'(Q)(x)$ has been set equal to 1 for all states reachable via states that satisfy $R'(Q)$. Once all these reachable states have been addressed, set $R'(Q)(x) = 0$ for any remaining states. $\diamond$

From the above algorithm we can see that all states that satisfy $R'(Q)$ can be reached from x_0 by going through states that satisfy Q without the occurrence of any prohibited transitions, that is, for any $x \in X - \{x_0\}$ with $R'(Q)(x) = 1$, there exist $x_1, x_2, \ldots, x_m \in X$ and $\sigma_0, \sigma_1, \ldots, \sigma_{m-1} \in \Sigma$ satisfying the following conditions:

$C3)$ $x_{i+1} \in \delta(x_i, \sigma_i)$ for $i = 0, 1, \ldots, m-1$.
$C4)$ $Q(x_i) = 1$ for $i = 0, 1, \ldots, m$.
$C5)$ $\sigma_i \in \Sigma - A'_Q(x_i)$ for $i = 0, 1, \ldots, m-1$.
$C6)$ $x_m = x$.

The following result that employs the logic of Lemma 1 in [13] can now be presented.

Theorem 2: For any predicate $Q \in \mathbf{P}$

$$R(Q) \leq R'(Q) \quad (9)$$

Proof: If $Q(x_0) = 0$, then $R(Q)(x) = R'(Q)(x) = 0$ for all $x \in X$. Consider the case that $Q(x_0) = 1$. Since $A'_Q(x) \subseteq A_Q(x)$ for any $x \in X$, we have $R(Q) \leq R'(Q)$. ■

Although the subpredicate $R(Q^{\uparrow})$ represents a larger achievable state set than can be achieved by any prior state-feedback work, Theorem 2 demonstrates that we can generate a potentially larger subpredicate $R'(Q^{\uparrow}) \geq R(Q^{\uparrow})$. This together with the example of Section V shows that our covering-based approach is less restrictive than existing state-feedback approaches.

The only thing left to do is to define a covering-based control law that will achieve the subpredicate $R'(Q^{\uparrow})$. We first propose the following state-feedback control law f' that we will employ in the construction of our covering-based law.

$$f'(x) = \Sigma - A'_{Q^{\uparrow}}(x) \tag{10}$$

For this control law, the closed-loop automaton $f'/G = (X, \Sigma, \delta_{f'}, x_0)$ is defined in the same manner as (1). The notation $Re(f'/G) \in \mathbf{P}$ again represents the predicate that is true only at all reachable states of f'/G. For any $x \in X - \{x_0\}$ with $Re(f'/G)(x) = 1$, there exist $x_1, x_2, \ldots, x_m \in X$ and $\sigma_0, \sigma_1, \ldots, \sigma_{m-1} \in \Sigma$ satisfying the following conditions [8]:

$C7)$ $x_{i+1} \in \delta(x_i, \sigma_i)$ for $i = 0, 1, \ldots, m-1$

$C8)$ $\sigma_i \in f'(x_i)$ for $i = 0, 1, \ldots, m-1$

$C9)$ $x_m = x$

The control law f' achieves the behavior specified by the predicate $R'(Q^{\uparrow})$. This fact is mathematically represented $Re(f'/G) = R'(Q^{\uparrow})$ and can now be proven using logic similar to that employed in Theorem 1 of Section III (proved in [12]).

Theorem 3: Let $Q \in \mathbf{P}$ be a predicate. If $Q^{\uparrow}(x_0) = 1$ and f' is given by (10), then $Re(f'/G) = R'(Q^{\uparrow})$.

Proof: Since f'/G has the same state set as G and x_0 is the initial state of G, x_0 is reachable in f'/G. That is, $Re(f'/G)(x_0) = 1$. By assumption, we also have that $Q^{\uparrow}(x_0) = 1$. Step 1 of Algorithm 3 then provides that $R'(Q^{\uparrow})(x_0) = 1$.

$(\leq)$ We will next show that $Re(f'/G) \leq R'(Q^{\uparrow})$ by induction. For any $x \in X - \{x_0\}$ with $Re(f'/G)(x) = 1$, there exist $x_1, x_2, \ldots, x_m \in X$ and $\sigma_0, \sigma_1, \ldots, \sigma_{m-1} \in \Sigma$ satisfying conditions $C7 - C9$. For the basis step, we already have that $Re(f'/G)(x_0) = R'(Q^{\uparrow})(x_0) = 1$. For the induction step, suppose that $R'(Q^{\uparrow})(x_k) = 1$. We now want to show that $R'(Q^{\uparrow})(x_{k+1}) = 1$ where $x_{k+1} \in \delta(x_k, \sigma_k)$ by $C7$. Consider two cases:

1) If $\sigma_k \in \Sigma_u$, we then have that $Q^{\uparrow}(x_{k+1}) = 1$ since $Q^{\uparrow}(x_k) = 1$ and $Q^{\uparrow} \leq wlp_{\sigma_k}(Q^{\uparrow})$. Furthermore, since $\sigma_k \in \Sigma_u$ we have that $\sigma_k \notin A'_{Q^{\uparrow}}(x_k)$. This and the fact that $\delta(x_k, \sigma_k)!$ provides that $R'(Q^{\uparrow})(x_{k+1}) = 1$ by Step 2 of Algorithm 3.
2) If $\sigma_k \in \Sigma_c$, then by condition $C8$ and (10), we have $\sigma_k \in f'(x^k) = \Sigma - A'_{Q^{\uparrow}}(x_k)$. Therefore, by Step 2 of Algorithm 3 we again have that $R'(Q^{\uparrow})(x_{k+1}) = 1$.

This completes the induction.

$(\geq)$ We will now show that $Re(f'/G) \geq R'(Q^{\uparrow})$. For any $x \in X - \{x_0\}$ with $R'(Q^{\uparrow})(x) = 1$ there exist $x_1, x_2, \ldots, x_m \in X$ and $\sigma_0, \sigma_1, \ldots, \sigma_{m-1} \in \Sigma$ satisfying conditions analogous to $C3 - C6$ but for $Q^{\uparrow}$ instead of Q. To show that $Re(f'/G)(x) = 1$, it is sufficient to prove that $\sigma_i \in f'(x_i) (i = 0, 1, \ldots, m-1)$. By condition $C3$ and (10), we have $\sigma_i \in \Sigma - A'_{Q^{\uparrow}}(x_i) = f'(x_i)$. ■

Based on the definition of the sets $A'_{Q^{\uparrow}}(x)$ employed by f', it can also be seen that the following relation is implied:

$$\text{For any } x, x' \in X \text{ with } x' \in I_{Q^{\uparrow}}(x), \quad \sigma \in f'(x) \cap \Sigma_G(x) \Rightarrow \sigma \in f'(x') \tag{11}$$

The above then leads to a result that is similar to (3):

$$\text{For any } x, x' \in X,\ x' \in I_{Q^{\uparrow}}(x) \Rightarrow \quad f'(x) \cap \Sigma_G(x) \cap \Sigma_G(x') = f'(x') \cap \Sigma_G(x) \cap \Sigma_G(x') \tag{12}$$

The above expression captures that the control applied by f' is consistent between states that are indistinguishable for those active events that are shared between the states. The limitation of consistency to those active events shared between states demonstrates the limited transitivity that provides the improvement of our covering-based approach.

Under full observation, we could just apply the state-feedback law f'. However, since some states cannot be distinguished for certain strings, we will need to define our covering-based control law as a function of observed strings, $\mathcal{S} : P(\mathcal{L}(G)) \to 2^{\Sigma}$.

$$\mathcal{S}(s) = \bigcap_{x \in \delta_{f'}(x_0, t)} f'(x), \forall t \in P^{-1}(s) \tag{13}$$

The expression in (11) indicates that the control law $\mathcal{S}$ working under partial observation will produce the same behavior as f' working under full observation. This fact is captured by the following proposition where $Re(\mathcal{S}/G) \in \mathbf{P}$ represents the predicate that is true only at states of G that are reachable under the control $\mathcal{S}$.

Proposition 1: Let G be an automaton. For the state-feedback law f' given by (10) and the covering-based feedback law $\mathcal{S}$ given by (13), $Re(f'/G) = Re(\mathcal{S}/G)$.

Proof: Follows from (11). ■

The covering-based control law $\mathcal{S}$ can be implemented on-line from the automaton that results from the state-feedback law f' under full observation, f'/G, which is calculated off-line. Specifically, following the observation of a string s we keep track of the possible set of states reached by strings with same observation s; those events that are active at these states then must be enabled.

A summary of the construction of the resulting controlled subpredicate $Re(f'/G) = R'(Q^{\uparrow})$ is given below.

Algorithm 4: $R'(Q^{\uparrow})$ Construction

Input: plant automaton G and specification predicate Q

Step 1: Find $Q^{\uparrow}$, the supremal Σ_u-invariant subpredicate. The algorithm of [11] can be employed.

Step 2: Construct the mapping $I_{Q^{\uparrow}}$ of indistinguishable states for the predicate $Q^{\uparrow}$. The algorithm of [3] can be used for this purpose and has polynomial complexity in the number of events and states.

Step 3: Construct the sets of prohibited events $A'_{Q^{\uparrow}}(x)$ to satisfy (7). Algorithm 2 can be employed.

Step 4: Follow Algorithm 3 to construct the transformed predicate $R'(Q^{\uparrow})$.
Output: $R'(Q^{\uparrow})$ $\diamond$

In the above, Step 2 and Step 3 could be addressed simultaneously by a single algorithm. Note that each step of the above algorithm has polynomial complexity in the number of events and states of G.

V. Subpredicate Construction Example

In this section we present an example that helps to illustrate our subpredicate construction procedure introduced in Algorithm 4. Consider the partially observed nondeterministic plant automaton G pictured in Fig. 2 and let Q be the specification predicate given below:

$$Q(x) = \begin{cases} 1 & \text{if } x \in \{0,1,2,3,4,6,7,8\} \\ 0 & \text{if } x \in \{5,9\} \end{cases}$$

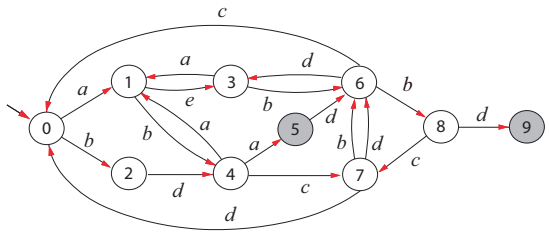

Fig. 2. Plant automaton G

Also let the event set of G be partitioned into controllable and uncontrollable events as follows, $\Sigma_c = \{a, b, c, e\}$ and $\Sigma_u = \{d\}$. By inspection, the predicate Q is not Σ_u-invariant because state 9 is not accepted by Q, yet state 9 is reached by the uncontrollable event d from state 8 that is accepted by Q. Therefore, according to Step 1 of Algorithm 4, the supremal Σ_u-invariant subpredicate $Q^{\uparrow}$ is constructed:

$$Q^{\uparrow}(x) = \begin{cases} 1 & \text{if } x \in \{0,1,2,3,4,6,7\} \\ 0 & \text{if } x \in \{5,8,9\} \end{cases}$$

Now consider that the event set of G is also partitioned into observable and unobservable events as follows, $\Sigma_o = \{a, b, c, d\}$ and $\Sigma_{uo} = \{e\}$. Since G is partially observed and nondeterministic, some of its states are indistinguishable. Following Step 2 of Algorithm 4, the mapping $I_{Q^{\uparrow}}$ representing which states are indistinguishable is then constructed. For the subpredicate $Q^{\uparrow}$ we will represent $I_{Q^{\uparrow}}$ as Table I. Specifically, the lefthand column enumerates each state $x \in X$ for which $Q^{\uparrow}(x) = 1$ and the center column lists the corresponding set of indistinguishable states $I_{Q^{\uparrow}}(x)$.

Step 3 of Algorithm 4 then constructs the sets of prohibited events $A'_{Q^{\uparrow}}(x)$. Examining state 6 of G, since the b event leads to a state not accepted by the predicate $Q^{\uparrow}$, $wlp_b(Q^{\uparrow})(6) = 0$ and b must be in the set $A'_{Q^{\uparrow}}(6)$. It then follows that b is also in the set $A'_{Q^{\uparrow}}(0)$ since state 0 is accepted by $Q^{\uparrow}$, is indistinguishable from state 6, and the b event is possible at state 0. Following this string of logic, b is then in the set $A'_{Q^{\uparrow}}(7)$ since state 7 is indistinguishable from state 0 while being accepted by $Q^{\uparrow}$ and having b possible. Since there are no further states in $I_{Q^{\uparrow}}(0)$, $I_{Q^{\uparrow}}(6)$, or $I_{Q^{\uparrow}}(7)$ for which a b event is defined, the transitivity is terminated. Further examination of G shows that $a \in A'_{Q^{\uparrow}}(4)$ since an a event leads from state 4 which is accepted by $Q^{\uparrow}$ to state 5 which is not. It then follows that a is also in the set $A'_{Q^{\uparrow}}(3)$ since state 3 is indistinguishable from state 6, 3 is accepted by $Q^{\uparrow}$, and a is possible at state 3. Since there are no other states in $I_{Q^{\uparrow}}(3)$ or $I_{Q^{\uparrow}}(4)$ for which an a event is defined, and since there are no other events required to be actively disabled by $Q^{\uparrow}$, all the sets $A'_{Q^{\uparrow}}(x)$ are now completely defined.

TABLE I
Table representing the map $I_{Q^{\uparrow}}$

x	$I_{Q^{\uparrow}}(x)$	$A'_{Q^{\uparrow}}(x)$
0	0, 6, 7	b
1	1, 3	
2	2, 6	
3	3, 1, 4	a
4	4, 3, 6	a
6	6, 2, 4, 0	b
7	7, 0	b

Step 4 of Algorithm 4 then applies the transformation R'. Since state 2 is reached only by the event b originating at state 0 and $b \in A'_{Q^{\uparrow}}(0)$, state 2 will not be allowed by the transformed predicate $R'(Q^{\uparrow})$. All other states, however, are reachable via transitions that are not prohibited by the sets $A'_{Q^{\uparrow}}(x)$. The end result of Algorithm 4 for this example is thus:

$$R'(Q^{\uparrow})(x) = \begin{cases} 1 & \text{if } x \in \{0,1,3,4,6,7\} \\ 0 & \text{if } x \in \{2,5,8,9\} \end{cases}$$

The plant G under the control of the associated feedback law $\mathcal{S}$ then satisfies the resulting subpredicate $R'(Q^{\uparrow})$. Figure 3 shows the portion of G reachable under the control of $\mathcal{S}$. The short dashed lines indicate disabled transitions.

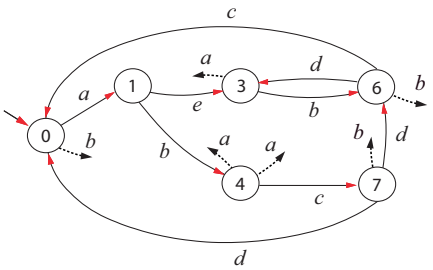

Fig. 3. Automaton representing the reachable portion of G under the control law $\mathcal{S}$

Remark 1: In traditional state-feedback control employing a mask M, states 0 and 4 would be in the same partition since state 0 is indistinguishable from state 6 and state 6 is indistinguishable from state 4. Therefore, traditional

approaches would have disabled the a event at state 0 also. This example along with (9) demonstrates the advantage of our approach over the state-feedback control approaches of [9] [14].

Remark 2: Our covering-based approach, however, still produces a static control law. For example, if for some reason the b event at state 6 needed to be disabled following observation of the string $ab = P(aeb)$, but not following observation of the string $abcd$, our control law would not be able to make that distinction.

Remark 3: States 3 and 4 are indistinguishable in G because they are both reached by strings with the observation $abcbd$. If we had recalculated the mapping $I_{Q^\uparrow}$ during the construction of $R'(Q^\uparrow)$ for an intermediate subpredicate that does not accept state 2, then states 3 and 4 would no longer be indistinguishable. This new mapping would then not have led to a prohibited event set that required that the a event at state 3 be disabled. If we allowed the $I_{Q^\uparrow}$ mapping to change within the calculation of the transformation R', then the resulting subpredicate would depend on the order in which the states were addressed. This dependence is an issue common also to event-feedback approaches to control under partial observation.

Remark 4: We have shown that our approach is more permissive than the approach proposed by [9]. The approach of [9] in turn has been shown to provide more permissive control than the construction of the supremal controllable and normal subpredicate presented in [14]. A predicate is normal if all elements of $M^{-1}(M(x))$ satisfy the predicate when the state x satisfies the predicate, where

$$M^{-1}(M(x)) = \{x' \mid M(x) = M(x')\}$$

The example presented in this section, therefore, also shows how the approach of [14] is more restrictive than our approach. Since in our example states 1 and 5 are both reached by the same string aba, they both have the same observation under M, but state 1 is accepted by the predicate $R'(Q^\uparrow)$ while state 5 is not. Therefore, the subpredicate $R'(Q^\uparrow)$ violates normality. Construction of the supremal normal and controllable subpredicate would then not accept state 1.

Remark 5: Assuming G additionally has a set of marked states, if it were desired that the closed-loop system be nonblocking, we would need to transform the predicate so that states that were blocking under the control law $\mathcal{S}$ were not accepted by the predicate. This transformation of the predicate, however, could result in the loss of the Σ_u-invariance property or failure of (12) (assuming f' reflects the new subpredicate). Therefore, it would then be necessary to iterate Algorithm 4 until the resulting control law did not lead to any blocking states. This approach was taken in [15] where we applied the covering-based control approach of this paper.

VI. Conclusions

In this work we have presented a new covering-based approach to state avoidance under partial observation that is implemented on-line and avoids exponential computational complexity. This approach allows more permissive behavior than can be achieved by existing state-feedback approaches. Existing approaches to state-feedback control under partial observation require that the control be consistently applied at states in the same observation partition. We rather generate a covering of the state space that allows for the application of a less restrictive control law.

The results of this paper can be applied to generating control based on abstracted models. In particular, our approach could be used in conjunction with abstraction techniques based on observation equivalence [16] and conflict equivalence [17]. We have applied the approach of this paper to generate coordinators that resolve conflict between modular supervisors using conflict-equivalent abstractions [15].

VII. Acknowledgements

The authors gratefully acknowledge the anonymous reviewers for their helpful and detailed comments.

References

[1] R. Su and J. Thistle, "A distributed supervisor synthesis approach based on weak bisimulation," in *Proc. Int. Workshop on Discrete Event Systems (WODES)*, Ann Arbor, USA, 2006, pp. 64–69.

[2] H. Flordal and R. Malik, "Modular nonblocking verification using conflict equivalence," in *Proc. Int. Workshop on Discrete Event Systems (WODES)*, Ann Arbor, USA, 2006, pp. 100–106.

[3] W. Wang, S. Lafortune, and F. Lin, "An algorithm for calculating indistinguishible states and clusters in finite state automata with partially observable transitions," *Systems Control Letters*, vol. 56, pp. 656–661, 2007.

[4] F. Lin and W. Wonham, "On observability of discrete event systems," *Information Sciences*, vol. 44, pp. 173–198, 1988.

[5] A. Overkamp, "Supervisory control using failure semantics and partial specification," *IEEE Trans. Automat. Contr.*, vol. 42, pp. 498–510, April 1997.

[6] M. Heymann and F. Lin, "Discrete-event control of nondeterministic systems," *IEEE Trans. Automat. Contr.*, vol. 43, 1998.

[7] C. Zhou, R. Kumar, and S. Jiang, "Control of nondeterministic discrete-event systems for bisimulation equivalence," *IEEE Trans. Automat. Contr.*, vol. 51, no. 5, pp. 754–765, May 2006.

[8] Y. Li and W. Wonham, "Control of vector discrete event systems-part I: The base model," *IEEE Trans. Automat. Contr.*, vol. 38, no. 8, pp. 1215–1227, 1993.

[9] S. Takai and S. Kodama, "Characterization of all M-controllable subpredicates of a given predicate," *Int. J. of Control*, vol. 70, no. 4, pp. 541–549, 1998.

[10] R. Kumar, V. Garg, and S. Marcus, "Predicates and predicate transformers for supervisory control of discrete event dynamical systems," *IEEE Trans. Automat. Contr.*, vol. 38, pp. 232–247, 1993.

[11] P. J. G. Ramadge and W. M. Wonham, "Modular feedback logic for discrete event systems," *SIAM Journal of Control and Optimization*, vol. 25, no. 5, pp. 1202–1218, 1987.

[12] S. Takai, T. Ushio, and S. Kodama, "Static-state feedback control of discrete-event systems under partial observation," *IEEE Trans. Automat. Contr.*, vol. 40, no. 11, pp. 1950–1954, November 1995.

[13] S. Takai and S. Kodama, "M-controllable subpredicates arising in state feedback control of discrete event systems," *Int. J. of Control*, vol. 67, no. 4, pp. 553–566, 1997.

[14] Y. Li, "Control of vector discrete-event systems," Ph.D. dissertation, University of Toronto, Toronto, Canada, 1991.

[15] R. Hill, "Modular verification and supervisory controller design for discrete-event systems using abstraction and incremental construction," Ph.D. dissertation, University of Michigan, Ann Arbor, USA, 2008.

[16] R. Milner, *Communication and Concurrency*. London: Prentice-Hall, Inc, 1989.

[17] R. Malik, D. Streader, and S. Reeves, "Conflicts and fair testing," *International Journal of Foundations of Computer Science*, vol. 17, no. 4, pp. 797–813, 2006.