Proceedings of the
47th IEEE Conference on Decision and Control
Cancun, Mexico, Dec. 9-11, 2008

Optimal Sensor Activation in Controlled Discrete Event Systems

Weilin Wang, Stéphane Lafortune, and Feng Lin

***Abstract*— The problem of sensor activation in a controlled discrete event system is considered. Sensors are assumed to be costly and can be turned on/off during the operation of the system. The agent activates sensors as needed in order to observe the trajectories of the system and correctly implement the given feedback control law. Different policies of sensor activation can be used by the agent. A policy is said to be minimal if any strictly less activation prevents the correct implementation of the control law. A systematic formulation of the sensor activation problem is obtained and a solution procedure is proposed. An algorithm that computes minimal sensor activation policies is presented. The algorithm is of polynomial complexity in both the number of states and the number of events in the system.**

***Index Terms*— Discrete event systems, sensor activation, supervisory control, observability**

I. Introduction

In a control system, the control or diagnosis decisions of an agent often depend on its observation of the system's trajectory. (We use the word "agent" to represent the generic entity responsible for control and/or diagnosis tasks in the control system.) However, online observation of the system is usually limited and can be costly for one or more of the following reasons: life span and availability of sensors, limited power of batteries, available computation and communication resources, or security. To reduce sensor-related costs, the agent may not want to activate the sensors continuously. Therefore, the problem of minimization of *sensor activation* is of great interest in the design of cost-efficient control and diagnostic systems. Our usage of the word "activation" in this paper includes all functionalities associated with the sensing devices, including communication with these devices. The problem of optimal sensor activation has been studied in the context of discrete event systems (DES) recently, where the common objective is to minimize the activation of sensors during the operation of the system [1]–[5]. There is also closely related work concerned with minimizing communications of event occurrences in decentralized-information systems, for purposes of control or diagnosis [6]–[9].

In the initial works on sensor selection in DES, the optimization objective is to minimize the *set* of events to be observed by the agent [10]–[13]. In its general form, this problem is known to be NP-complete (see, e.g., [13]). Research on effective approximation methods to solve such problems remains active; see, e.g., [14]. In these problem settings, it is assumed that, for each event, the agent either always or never activates the sensor.

This paper is concerned with the problem where the agent may want to activate the sensor only some of the time, depending on the trajectory of the system. This flexibility becomes important when the system is operated in an environment where the available sensing resources are limited or costly. For example, in a wireless sensor network, transmitting data or making measurements may involve using limited battery power or limited bandwidth. Also, the life span of a sensor often depends on the frequency of the measurements it takes. In other instances, security concerns may motivate the need to minimize communications with sensing devices. The intricate part of the sensor activation problem is that the decision of activating or not a sensor at a given point in the evolution of the system depends not only on the current trajectory of the system, but also on the agent's observation of that trajectory, which in turn depends on how sensors have been activated so far. The work in [1]–[3] considers the sensor activation problem (primarily for the purpose of event diagnosis) from an optimal control viewpoint and it captures the requirements associated with the definition of sensor activation policies by defining a suitable *information state*. For acyclic systems, i.e., for finite languages, an appropriate filtration of σ-fields of information states is identified, and an optimal policy is then found by dynamic programming. For cyclic systems, i.e., for infinite regular languages, the set of string-based information states is reduced to the set of diagnoser states. The formulation to the sensor activation problem in [4], [5] is different and based on safety 2-player games and weighted automata; the goal is to ensure diagnosability.

This paper is specifically concerned with the problem of minimizing sensor activation to preserve the property of *observability* in a controlled DES. Observability is the necessary and sufficient condition dealing with sensing limitations for supervisor existence in partially-observed supervisory control problems (controllability addresses actuation limitations). We are interested in the generalized notion of observability in the context of observable event occurrences (or transitions) in [15], which extends the basic notion of observability (in the context of observable events) in [16]. The discussion in [17] shows that the problem of guaranteeing both notions of observability can be transformed into a problem of state disambiguation. We borrow some notions from the communication problems described in [8], [9] for the sensor activation problem of this paper. The notion of *feasibility* ensures the consistency between the agent's

W. Wang and S. Lafortune are with the Department of EECS, University of Michigan, Ann Arbor, MI 48109, {weilinw, stephane}@umich.edu. Their research is supported in part by NSF grants CCR-0325571 and ECS-0624821. F. Lin is with the Department of ECE, Wayne State University, Detroit, MI 48202, flin@ece.eng.wayne.edu. His research is supported in part by NSF grants INT-0213651 and ECS-0624828.

observation of the trajectories of the system and its decisions on sensor activation. The notion of *implementability* restricts the solution space to the state space of the system for the sake of computational efficiency. The notion of *minimality* is a logical one: a sensor activation policy is minimal if removing one or more activations of event occurrences in the dynamic evolution of the system renders a correct solution incorrect.

Our approach is different from the work in [1]–[5] in many respects: (i) our notion of optimality is logical as opposed to numerical; (ii) our solution framework covers finite and infinite regular languages in the same manner (in contrast to the work in [3]); (iii) with the requirement of implementability, our solution space is restricted to the subsets of the set of transitions of the system. This last point means that optimal solutions to our problem can be computed in polynomial time in the state space of the system. This is in contrast to the more general numerical optimization problem in [3], where a dynamic program must be solved over a state space doubly-exponential with respect to the maximum length of system strings (acyclic case) or exponential with respect to the state space of the system (cyclic case). In [4], [5], the authors use a safety 2-player game-theoretic argument that involves the construction of the so-called "most-permissive observer," a step of exponential complexity in the state space of the system and doubly-exponential in the number of observable events.

An example in [8] shows that knowing more occurrences of events can cause some distinguishable strings to become indistinguishable. This result is sometimes called the "lack of monotonicity problem" in decentralized-information systems with communication. We demonstrate in this paper that the lack of monotonicity problem does not arise for *feasible sensor activation policies*. In other words, within the class of feasible sensor activation policies, more frequent sensor activations do not cause distinguishable strings to become indistinguishable. This key result allows us to derive polynomial-time algorithms in the state space of the system and in the number of events for calculating minimal sensor activation policies without any structural assumptions on the system.

This paper is organized as follows. Section II presents a systematic description of our problem. Then, we show the existence of a maximum feasible subpolicy for a given policy in Section III. In Section IV, Algorithm MIN-SEN-ACT, together with its subroutine Algorithm MAX-FEA-SUB, are presented for solving the sensor activation problem. An illustrative example and a complexity analysis follow. Section V provides concluding remarks. Proofs have been omitted due to space limitations; they are available from the authors.

II. DESCRIPTION OF PROBLEM

A. System Model

We assume basic knowledge of DES and common notation. We model an untimed DES as a deterministic finite-state automaton

$$G = (X, E, f, \Gamma, x_0) \tag{1}$$

where X is the finite set of states, E is the finite set of events, $f : X \times E \to X$ is the partial transition function where $f(x,e) = y$ means that there is a transition labelled by event e from state x to state y, Γ is the active event function where $\Gamma(x)$ is the set of all events e for which $f(x,e)$ is defined (called the active event set of G at x), and x_0 is the initial state. f is extended to $X \times E^*$ in the usual way. We use $\mathscr{L}(G)$ to denote the language generated by G. We define the set of transitions of G as

$$TR(G) := \{(x,e) \in X \times E : e \in \Gamma(x)\}. \tag{2}$$

The set of events that are observable by the agent is denoted by E_o, and the set of events that can never observed by the agent is denoted by E_{uo}. An event is observable if there exists a sensor associated with the event that can be activated.

B. Sensor Activation Model

We first present a general language-based model of sensor activation. We then restrict this model to one that is based on the state space of G. Sensors are activated by the agent; in the context of this paper, the agent will be the supervisory controller for G. When to activate sensors is described by the following sensor activation mapping

$$\omega : \mathscr{L}(G) \to 2^{E_o}. \tag{3}$$

In other words, for a trajectory $s \in \mathscr{L}(G)$, $\omega(s)$ is a subset of observable events corresponding to the sensors that are activated by the agent after s.

Given a sensor activation mapping ω, we use induction to define the corresponding information mapping $\theta^\omega : \mathscr{L}(G) \to E_o^*$ as follows. For the empty string ε, $\theta^\omega(\varepsilon) = \varepsilon$, and for all $se \in \mathscr{L}(G)$,

$$\theta^\omega(se) = \begin{cases} \theta^\omega(s)e & \text{if } e \in \omega(s) \\ \theta^\omega(s) & \text{otherwise.} \end{cases} \tag{4}$$

After the occurrence of s, the next event e is *seen* or *observed* by the agent when it occurs after s if and only if the agent activates the sensor for e after the occurrence of s.

The set of confusable string pairs, denoted by $\mathscr{T}_{conf}(\omega)$, is defined as

$$\mathscr{T}_{conf}(\omega) = \{(s,t) \in \mathscr{L}(G) \times \mathscr{L}(G) : \theta^\omega(s) = \theta^\omega(t)\}. \tag{5}$$

Note that for all $s \in \mathscr{L}(G)$, we have $(s,s) \in \mathscr{T}_{conf}(\omega)$.

It is important to note that not all arbitrary sensor activation policies ω will be "feasible" based on the information available to the agent. To guarantee feasibility, it is required that any two strings of events that are indistinguishable to the agent must be followed by the same activation decision for every event. Namely, an activation policy ω must be "compatible" with the information mapping θ^ω that is built from it. Formally, ω is said to be *feasible* if

$$\begin{aligned} &(\forall\, e \in E)(\forall\, se, s'e \in \mathscr{L}(G))\ \theta^\omega(s) = \theta^\omega(s') \\ &\Rightarrow\ [e \in \omega(s) \Leftrightarrow e \in \omega(s')]. \end{aligned} \tag{6}$$

In principle, to check feasibility, we first calculate θ^{ω} from ω, and then check if equation (6) holds.

For the purpose of computational efficiency, we introduce a so-called "implementability" condition that restricts the class of sensor activation policies to the state space structure of G. This condition is reminiscent of the implementability condition in [6], [8], [9]. Formally, ω is said to be *implementable* with respect to G if

$$(\forall\, e \in E)\ (\forall\, s, s' \in \mathscr{L}(G)) f(x_0, s) = f(x_0, s') \Rightarrow [e \in \omega(s) \ \Leftrightarrow\ e \in \omega(s')]. \tag{7}$$

In words, implementability requires that any two strings of events that lead to the same state in G must be followed by the same activation decision for every event. Clearly, the solution space for the problem of sensor activation can be refined by refining the state space of automaton G, at the cost of increased computations; we assume hereafter that the designer has chosen the desired structure of G on the basis of available computing power.

When the implementability condition is satisfied, we can associate the activation of sensors with the transition in G: the event associated with each transition in $TR(G)$ is either sensed (activated) by the agent or not. The set of transitions whose event labels are sensed by the agent is denoted by

$$\Omega \subseteq TR(G), \tag{8}$$

where $(x, e) \in \Omega$ means that

$$(\forall\, s \in \mathscr{L}(G))\ f(x_0, s) = x \ \Rightarrow\ e \in \omega(s). \tag{9}$$

We call Ω a sensor activation policy. We shall refer hereafter to the elements of Ω as the *activated transitions*; note that the state name is never observed, only the event label of the activated transition. Ω is defined only if the implementability condition is satisfied. When this is the case, there is a one-to-one correspondence between ω and Ω. In particular, ω can be obtained from Ω as follows.

$$\omega(s) = \{e \in E_o : (f(x_0, s), e) \in \Omega\}. \tag{10}$$

We denote by θ^{Ω} the function θ^{ω} of equation (4) where ω comes from a given Ω according to equation (10). We say that Ω is feasible if the corresponding ω is feasible. It is not difficult to see that Ω is *feasible* if and only if

$$(\forall\, e \in E)(\forall\, se, s'e \in \mathscr{L}(G))\theta^{\Omega}(s) = \theta^{\Omega}(s') \Rightarrow [(f(x_0, s), e) \in \Omega \Leftrightarrow (f(x_0, s'), e) \in \Omega]. \tag{11}$$

In prior works on the sensor selection problem [10]–[13], a sensor is either activated all the time, or not at all. That corresponds to the sensor activation policy

$$\Omega^{on} = \{(x, e) \in TR(G) : x \in X, e \in E_o'\},$$

where $E_o' \subseteq E_o$ is the set of sensors that are (always) activated. It is easy to verify that such a sensor activation policy is necessarily feasible.

The set of confusable state pairs, denoted by $T_{conf}(\Omega)$, is defined as

$$T_{conf}(\Omega) = \{(x, y) \in X \times X :\ (\exists\, s, s' \in \mathscr{L}(G))\ x = f(x_0, s) \wedge y = f(x_0, s') \wedge \theta^{\Omega}(s) = \theta^{\Omega}(s')\}. \tag{12}$$

Note that for all $x \in X$, $(x, x) \in T_{conf}(\Omega)$.

C. Objective for the Agent

In addition to the feasibility requirement specified by equation (11), we may further require that the agent be able to distinguish certain pairs of states of G for its own control purposes. Formally, we specify a relation $T_{spec} \subseteq X \times X$, which is called the *specification* condition. We require that no state pair $(x, y) \in T_{spec}$ be indistinguishable from the viewpoint of the agent, that is,

$$(\forall s, s' \in \mathscr{L}(G))\ \theta^{\Omega}(s) = \theta^{\Omega}(s') \Rightarrow (f(x_0, s), f(x_0, s')) \notin T_{spec}. \tag{13}$$

The agents must activate a proper set of transitions to obtain sufficient information so that the above requirement is satisfied.

T_{spec} is user-defined and problem-dependent. For example, in supervisory control, we may want to find a minimal sensor activation policy so that the *observability* property holds. This property is defined in [15] in the context of observable event occurrences; it is an extension of the original definition in [16] for natural projections on E_o. Obtaining T_{spec} from the observability requirement is discussed in Section 5.1 in [17]; details are omitted here. Given T_{spec} obtained from the observability requirement and given a sensor activation policy Ω, the language $\mathscr{L}(G)$ is observable if and only if equation (13) holds, or, equivalently, $T_{conf}(\Omega) \cap T_{spec} = \emptyset$.

The following example shows how to construct T_{spec}. (For conciseness, when listing elements of sets of states T_{spec} and T_{conf} hereafter, we only list pair (x, y); by definition, pair (y, x) is also in the set even if not explicitly listed.)

Example 1: Consider the system modeled by the automaton A shown in Fig. 1. Suppose the desired behavior is represent by subautomaton G, which is obtained by removing state 3. Let the set of controllable event be $E_c = \{e\}$. Event e is defined at states 0, 1, and 4 of A, but it is not defined at state 1 of G. Therefore, to ensure observability of the language generated by G with respect to that generated by A, we need to disambiguate states 1 and 4, and also states 1 and 0. In other words, $T_{spec} = \{(1, 4), (1, 0)\}$.

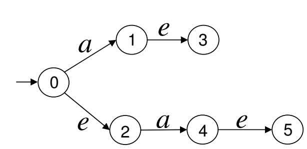

Fig. 1. Observability and state disambiguation in Example 1

We are now ready to formally state the problem to be solved.

D. Problem Statement

Given a system $G = (X, E, f, \Gamma, x_0)$, a specification T_{spec}, and a set of observable events $E_o \subseteq E$. Assume that if the agent activates all sensors all the time, that is, under $\Omega^{all} = \{(x, e) : x \in X, e \in E_o\}$, the specification T_{spec} is satisfied:

$$(\forall s, s' \in \mathscr{L}(G))\theta^{all}(s) = \theta^{all}(s') \Rightarrow (f(x_0, s), f(x_0, s')) \notin T_{spec}. \tag{14}$$

We would like to solve the following problem. Find a sensor activation policy $\Omega^* \subseteq TR(G)$ such that:

C1. Ω^* is feasible.

C2. The specification T_{spec} is satisfied:

$$(\forall s, s' \in \mathscr{L}(G))\theta^*(s) = \theta^*(s') \Rightarrow (f(x_0, s), f(x_0, s')) \notin T_{spec}. \quad (15)$$

where θ^* is the information map obtained from Ω^*.

C3. Ω^* is minimal, i.e., there is no other $\Omega' \subset \Omega^*$ that satisfies (C1) and (C2).

This optimization problem is different from the ones considered in [1]–[5]. The above notion of minimality is particularly well-suited for problems where communications from the sensors to the agent are costly. Moreover, this formulation admits nice properties as will be seen in Section III.

E. About Lack of Monotonicity

In a more general problem setting involving several communication agents, observing the event labels of more transitions does not mean that the agent can distinguish more pairs of states; see, e.g., Example 2 in [8]. Fortunately, for the sensor activation problem discussed in this paper, the agent activates sensors based on its own observation of the system. Because of the feasibility requirement of the sensor activation problem, activating more often the sensors does help the agent to distinguish more pairs of strings. We will formally prove this in Theorem 1 in Section III-B.

III. Analysis of Properties

We first present some definitions and then formally investigate properties of the sensor activation problem.

A. Notation and Definitions

The prefix-closure of a string $s \in \mathscr{L}(G)$, denoted by $PC(s)$, is defined as $PC(s) = \{u \in E^* : (\exists\, v \in E^*)\, uv = s\}$. We say that $\omega' \subseteq \omega''$ if $(\forall\, s \in \mathscr{L}(G))\, \omega'(s) \subseteq \omega''(s)$. We say that $\omega' \subset \omega''$ if $\omega' \subseteq \omega''$ and $(\exists\, s \in \mathscr{L}(G))\, \omega'(s) \subset \omega''(s)$.

For a given sensor activation policy Ω, we define the "unobserved reach" of state $x \in X$ under Ω, denoted by $UR(x, \Omega) \subseteq X$, to be the set of states that can be reached from x via "unobserved" transitions, namely, unactivated transitions or transitions labeled by unobservable events.

B. Properties of Sensor Activation Problem

The relation between the feasibility of Ω and the set of confusable state pairs $T_{conf}(\Omega)$ is given by the following lemma, whose proof follows directly from the definitions of T_{conf} and feasibility.

Lemma 1: Ω is feasible if for any $e \in E$ and $(x,e), (y,e) \in TR(G)$,

$$(x,y) \in T_{conf}(\Omega) \Rightarrow ((x,e) \in \Omega \Leftrightarrow (y,e) \in \Omega). \quad (16)$$

Lemma 2 states that if two strings have the same information mapping, then for each prefix of one string, there exists at least one prefix of the other string having the same information mapping. The proof follows from a recursive application of the definition of the function θ.

Lemma 2: Let $s, t \in \mathscr{L}(G)$. Then $\theta(s) = \theta(t)$ if and only if

$$[(\forall\, u \in PC(s))\, (\exists\, v \in PC(t))\, \theta(u) = \theta(v)] \wedge [(\forall\, v \in PC(t))\, (\exists\, u \in PC(s))\, \theta(v) = \theta(u)]. \quad (17)$$

The following important theorem establishes the monotonicity of feasible sensor activation policies.

Theorem 1: Consider a prefix-closed language L and two sensor activation policies ω' and ω'' for it. If ω' and ω'' are both feasible, i.e., they both satisfy equation (6), then

$$\omega' \supseteq \omega'' \Rightarrow \mathscr{T}_{conf}(\omega') \subseteq \mathscr{T}_{conf}(\omega''). \quad (18)$$

The following corollary is a state-based version of Theorem 1.

Corollary 1: Given system G and two sensor activation policies Ω' and Ω'' for it. If Ω' and Ω'' are both feasible, i.e., they both satisfy equation (11), then

$$\Omega' \supseteq \Omega'' \Rightarrow T_{conf}(\Omega') \subseteq T_{conf}(\Omega''). \quad (19)$$

The following theorem says that the union of two feasible policies is also feasible.

Theorem 2: Consider language $\mathscr{L}(G)$ and two feasible sensor activation policies ω' and ω'' for it. Then, $\omega = \omega' \cup \omega''$ is also feasible.

For any given sensor activation policy, the following theorem states that, among all of its feasible sub-policies, there is a unique maximum feasible sub-policy. This result plays an important role in the development of the algorithmic solution of the sensor activation problem presented in Section IV.

Theorem 3: For a given system G and a sensor activation policy Ω for G, there exists a maximum feasible sensor activation policy $\Omega^{\uparrow F}$ such that $\Omega^{\uparrow F} \subseteq \Omega$, i.e., for all feasible Ω' with $\Omega' \subseteq \Omega$, we have $\Omega' \subseteq \Omega^{\uparrow F}$. Let Ω_i, $i = 1, \ldots, m$ be all feasible sensor activation polices such that $\Omega_i \subseteq \Omega$. Then, we have $\Omega^{\uparrow F} = \cup_{i=1}^m \Omega_i$. Furthermore, its corresponding set of confusable state pairs is given by $T_{conf}(\Omega^{\uparrow F}) = \cap_{i=1}^m T_{conf}(\Omega_i)$.

IV. Algorithms for Minimization of Sensor Activation

In this section, we present the main algorithm, called Algorithm Min-Sen-Act, for finding a minimal sensor activation policy Ω^*. In the algorithm, we use a subroutine, called Algorithm Max-Fea-Sub, to find the maximum feasible sensor activation policy $\Omega^{\uparrow F}$ of a given sensor activation policy Ω. An illustrative example is also presented.

A. Algorithms

Algorithm Min-Sen-Act**:**

Input: A system G, a set of observable events E_o, and a specification T_{spec}.

Step 0: Initialization. Set $\Omega = \{(x,e) \in TR(G) : e \in E_o\}$ and $D = \emptyset$.

Step 1: Pick a transition $(x,e) \in \Omega$ but $(x,e) \notin D$. Let $\Omega^T \leftarrow \Omega \setminus \{(x,e)\}$.

Step 2: Call Algorithm Max-Fea-Sub to calculate the maximum feasible sensor activation policy $\Omega^{\uparrow F}$ of all feasible sub-policies of Ω^T and its corresponding set of confusable

state pairs $T_{conf}(\Omega^{\uparrow F})$ as $(\Omega^{\uparrow F}, T_{conf}(\Omega^{\uparrow F}))$ = MAX-FEA-SUB(Ω^{T}).
Step 3: Test $T_{conf}(\Omega^{\uparrow F}) \cap T_{spec} = \emptyset$. If it is true, set $\Omega \leftarrow \Omega^{\uparrow F}$ and $T_{conf}(\Omega) \leftarrow T_{conf}(\Omega^{\uparrow F})$. Otherwise, set $D \leftarrow D \cup \{(x,e)\}$.
Step 4: If $\Omega \neq D$, go to Step 1. Otherwise set $\Omega^* \leftarrow \Omega$, $T_{conf}(\Omega^*) \leftarrow T_{conf}(\Omega)$, and stop.
OUTPUT: Minimal sensor activation policy Ω^* and its set of confusable state pairs $T_{conf}(\Omega^*)$. ∎

Theorem 4: The output Ω^* of Algorithm MIN-SEN-ACT is a solution to the sensor activation problem formulated in Section II-D.

Given a sensor activation policy Ω, the maximum feasible sensor activation policy $\Omega^{\uparrow F}$ can be found by the following algorithm.

Algorithm MAX-FEA-SUB**:**
INPUT: Sensor activation policy Ω, G, and E_o.
Step 0: Initially, set $\overline{\Omega} \leftarrow \Omega$ and $T \leftarrow \{(x,x) \in X \times X\}$.
Step 1: Recursively, set $T \leftarrow T \cup \{(x,y) \in X \times X : (\exists\, (w,z) \in T)\; x \in UR(w,\overline{\Omega}) \wedge y \in UR(z,\overline{\Omega})\}$.
Step 2: Recursively, set $T \leftarrow T \cup \{(x,y) \in X \times X : (\exists\, (w,e'),(z,e') \in TR(G))(w,z) \in T \wedge f(w,e') = x \wedge f(z,e') = y\}$.
Step 3: Iterate Steps 1 and 2 until there are no further changes to T.
Step 4: Recursively, set $\overline{\Omega} \leftarrow \overline{\Omega} \setminus \{(y,e) \in \overline{\Omega} : (\exists\; (x,e) \in TR(G) \setminus \overline{\Omega}\;)(x,y) \in T\}$.
Step 5: Repeat Steps 1 to 4 until there are no further changes to $\overline{\Omega}$. Then, set $\Omega^{\uparrow F} \leftarrow \overline{\Omega}$ and $T_{conf}(\Omega^{\uparrow F}) \leftarrow T$.
OUTPUT: $\Omega^{\uparrow F}$ and the corresponding set of confusable state pairs $T_{spec}(\Omega^{\uparrow F})$. ∎

Theorem 5: The output $\Omega^{\uparrow F}$ of Algorithm MAX-FEA-SUB satisfies Theorem 3, i.e., it is the maximum feasible sensor activation policy of Ω. Furthermore, the output $T_{conf}(\Omega^{\uparrow F})$ is the set of confusable state pairs under $\Omega^{\uparrow F}$.

B. Illustrative Example

In this Section, we illustrate how Algorithm MIN-SEN-ACT and Algorithm MAX-FEA-SUB proceed. To specify Ω and D for each iteration, we use square brackets to denote that a transition has been removed and parentheses to show that a transition cannot be removed. The subscripts outside square brackets and parentheses are used to mark the order in which the transitions are examined. Suppose we are examining the n^{th} transition; then the current D is the set of all transitions within parentheses that have a subscript less than n, and the current Ω is all transitions in $TR(G)$ but not within square brackets that have a subscript less than n.

Example 2: The system is modeled by automaton A shown in Fig. 2. Let $E_o = \{a_1,a_2,e_1,e_2\}$ and $E_c = \{e_1,e_2\}$. Suppose the purpose of control is to prevent deadlock at state 5. Therefore, we need to disable event e_1 at state 4 and the automaton G describing the desired behavior is given by deleting state 5 together with transition $(4,e_1)$ from A. Since e_1 is also defined at states 0 and 1 where it should not be disabled, we have that $T_{spec} = \{(4,1),(4,0)\}$.

By Step 0 of Algorithm MIN-SEN-ACT, set $\Omega = TR(G)$ and $D = \emptyset$. Go to Step 1; suppose we try to remove $(0,a_1)$

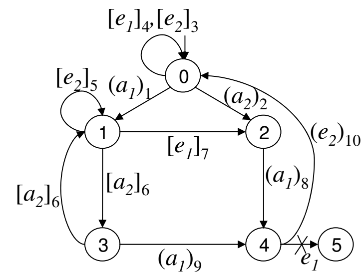

Fig. 2. Minimal solution computed in Example 2

in the first iteration. Set $\Omega^T \leftarrow \Omega \setminus \{(0,a_1)\}$. Go to Step 2 and proceed with Algorithm MAX-FEA-SUB as follows. By Step 0, set $\overline{\Omega} \leftarrow \Omega^T$ and $T = \{(x,x) \in X \times X\}$. By Step 1, set $T \leftarrow T \cup \{(0,1)\}$. By Step 2, since $(0,1) \in T$, $f(0,a_2) = 2$, $f(1,a_2) = 3$, set $T \leftarrow T \cup \{(2,3)\}$. Since $f(0,e_1) = 0$, $f(1,e_1) = 2$, set $T \leftarrow T \cup \{(2,0)\}$. Since $(2,0) \in T$, $f(2,a_1) = 4$, $f(0,a_1) = 1$, set $T \leftarrow T \cup \{(4,1)\}$. Completing Algorithm MAX-FEA-SUB, we have $(4,1) \in T_{conf}(\Omega^{\uparrow F})$. Therefore, $T_{conf}(\Omega^{\uparrow F}) \cap T_{spec} \supseteq \{(4,1)\} \neq \emptyset$ in Step 3 of Algorithm MIN-SEN-ACT. Set $D \leftarrow D \cup \{(0,a_1)\}$. Go to Step 4, since $\Omega \neq D$, go back to Step 1.

In the second iteration, we try to remove $(0,a_2)$. By Step 2, set $\Omega^T \leftarrow \Omega \setminus \{(0,a_2)\}$ and proceed to Algorithm MAX-FEA-SUB. By Step 1, we have $(2,0) \in T$. By Step 2, since $f(2,a_1) = 4$ and $f(0,a_1) = 1$, we have $(4,1) \in T$. Therefore, when performing Step 3 of Algorithm MAX-FEA-SUB, we have $T_{conf}(\Omega^{\uparrow F}) \cap T_{spec} \supseteq \{(4,1)\} \neq \emptyset$. Set $D \leftarrow D \cup \{(0,a_2)\}$. Go to Step 4; since $\Omega \neq D$, go back to Step 1.

In the third iteration, we try to remove $(0,e_1)$. It is easy to verify that $\Omega^{\uparrow F} = \Omega \setminus \{(0,e_1)\}$ with corresponding $T_{conf}(\Omega^{\uparrow F}) = \{(x,x) \in X \times X\}$. For the same reason as for $(0,e_1)$, we can remove $(0,e_2)$ and $(1,e_2)$ in the fourth and fifth iterations.

In the sixth iteration, we try to remove $(1,a_2)$. Set $\Omega^T \leftarrow \Omega \setminus \{(1,a_2)\}$. Go to Step 2 and proceed to Algorithm MAX-FEA-SUB as follows. By Step 0, set $\overline{\Omega} \leftarrow \Omega^T$ and $T = \{(x,x) \in X \times X\}$. By Step 1, we have $T \leftarrow T \cup \{(3,1)\}$. In Step 2, T does not change. Go to Step 4. Since $(1,3) \in T$, $(1,a_2) \notin \overline{\Omega}$, and $(3,a_2) \in TR(G)$, set $\overline{\Omega} \leftarrow \overline{\Omega} \setminus \{(3,a_2)\}$. Go back to Steps 1 and 2; but there is no further change of T. Thus, set $\Omega^{\uparrow F} \leftarrow \overline{\Omega}$, $T_{conf}(\Omega^{\uparrow F}) \leftarrow T$, and go to Step 3 of Algorithm MIN-SEN-ACT. Since $(3,1) \notin T_{spec}$, we have $T_{conf}(\Omega^{\uparrow f}) \cap T_{spec} = \emptyset$. Set $\Omega \leftarrow \Omega^{\uparrow F}$ and $T_{conf}(\Omega) \leftarrow T_{conf}(\Omega^{\uparrow F})$. Since $D \neq \Omega$, go back to Step 1.

In the seventh iteration, we try to remove $(1,e_1)$. Going through the algorithms, it can be verified that $(1,e_1)$ can be removed. Thus, we have $\Omega \leftarrow \Omega \setminus \{(1,e_1)\}$.

In the eighth and ninth iterations, we try to remove $(2,a_1)$ and $(3,a_1)$ respectively. In both cases, by Step 1 of Algorithm MAX-FEA-SUB, we have state 4 in the unobserved reach of state 1, i.e., we have $T_{conf}(\Omega^{\uparrow F}) \cap T_{spec} \supseteq \{(4,1)\} \neq \emptyset$. Thus, we have $D \leftarrow D \cup \{(2,a_1),(3,a_1)\}$.

In the tenth (and final) iteration, we try to remove $(4,e_2)$, but it causes state 0 to be in the unobserved reach of state 4. By Step 1 of Algorithm MAX-FEA-SUB, we have $(4,0) \in T$. Thus, we have $T_{conf}(\Omega^{\uparrow F}) \cap T_{spec} \supseteq \{(4,0)\} \neq \emptyset$. Set $D \leftarrow D \cup \{(4,e_2)\}$. Go to Step 4; we have $D = \Omega$ and $T_{conf}(\Omega^*) = \{(0,0),(1,1),(2,2),(3,3),(4,4),(1,2),(1,3),(2,3)\}$.

Note that the final result depends on the order in which the transitions are examined in Algorithm MIN-SEN-ACT. This can be seen from Fig. 3, which gives an alternative minimal solution for this problem. ■

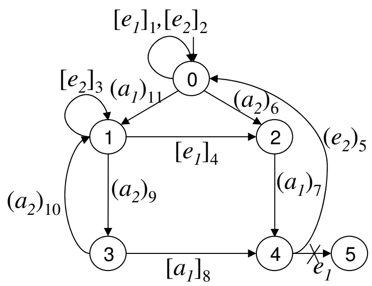

Fig. 3. Another minimal solution for Example 2

C. *Complexity of Algorithm* MIN-SEN-ACT

As a consequence of the iterative procedure in Algorithm MIN-SEN-ACT and of Algorithm MAX-FEA-SUB for the implementation of Step 2, we have the following result.

Theorem 6: The complexity of solving the problem formulated in Section II-D by using the algorithms in this section is upper bounded by $\|X \times E\|(\|X \times X\| + \|X \times E\|)$.

V. CONCLUSION

We have considered a different approach to the dynamic sensor activation problem than what has been studied so far in the literature. We have adopted a logical notion of optimality (set inclusion) and have restricted the solution space for the sake of computational efficiency. In this context, we have proved that the sensor activation problem possesses a monotonicity property; we have also proved the existence of the maximum feasible sub-policy of a given policy. Using these results, we have developed Algorithm MIN-SEN-ACT for the optimization of dynamic sensor activation in the context of controlled DES. It uses Algorithm MAX-FEA-SUB as a subroutine. These two algorithms are of polynomial-time complexity with respect to the cardinality of both the state space and the event set of the system.

While MAX-FEA-SUB returns a unique solution, the maximum feasible sub-policy of the input policy, the solution returned by MIN-SEN-ACT will depend on the order in which the transitions are considered for removal. This is a consequence of the existence of several minimal solutions for the criterion of optimality.

In principle, one could repeat MIN-SEN-ACT with different orders for the transitions in Step 1; note that in this case, calculations made by MAX-FEA-SUB could be saved and reused. Selection of an appropriate order for considering transitions for removal and comparison of different minimal solutions are application-specific. An interesting avenue for future research is to embed MIN-SEN-ACT as a part of a larger quantitative optimization problem. The computational efficiency of MIN-SEN-ACT makes such an approach attractive.

REFERENCES

[1] D. Thorsley and D. Teneketzis, "Active acquisition of information for diagnosis of discrete event systems," in *Proceedings of the Allerton Conference on Control, Communication, and Computing*, Sept. 2004.

[2] ——, "Diagnosis of cyclic discrete-event systems using active acquisition of information," in *8th International Workshop on Discrete Event Systems (WODES'06)*, Ann Arbor, 2006, pp. 248–255.

[3] ——, "Active acquisition of information for diagnosis and supervisory control of discrete event systems," *Discrete Event Dynamic Systems: Theory and Applications*, vol. 17, no. 4, pp. 531–586, Dec. 2007.

[4] F. Cassez, S. Tripakis, and K. Altisen, "Synthesis of optimal-cost dynamic observers for fault diagnosis of discrete-event systems," *First Joint IEEE/IFIP Symposium on Theoretical Aspects of Software Engineering*, pp. 316–325, 2007.

[5] ——, "Sensor minimization problems with static or dynamic observers for fault diagnosis," *Seventh International Conference on Application of Concurrency to System Design*, pp. 90–99, 2007.

[6] K. Rudie, S. Lafortune, and F. Lin, "Minimal communication in a distributed discrete-event system," *IEEE Trans. Automatic Control*, vol. 48, no. 6, pp. 957–975, June 2003.

[7] F. Lin, K. Rudie, and S. Lafortune, "Minimal communication for essential transitions in a distributed discrete-event system," *IEEE Trans. Automatic Control*, vol. 52, no. 8, pp. 1495–1502, Aug. 2007.

[8] W. Wang, S. Lafortune, and F. Lin, "On the minimization of communication in networked systems with a central station," *Discrete Event Dynamic Systems: Theory and Applications*, vol. 18, no. 3, pp. 415–443, Sept. 2008.

[9] ——, "Minimization of communication of event occurrences in acyclic discrete event systems," *IEEE Trans. Automatic Control*, vol. 53, no. 11, Nov. 2008.

[10] A. Haji-Valizadeh and K. A. Loparo, "Minimizing the cardinality of an event set for supervisors of discrete-event dynamical systems," *IEEE Trans. Automatic Control*, vol. 41, no. 11, pp. 1579–1593, 1996.

[11] S. Jiang, R. Kumar, and H. Garcia, "Optimal sensor selection for discrete-event systems with partial observation," *IEEE Trans. Automatic Control*, vol. 48, no. 3, pp. 369–381, Mar. 2003.

[12] R. Debouk, S. Lafortune, and D. Teneketzis, "On an optimization problem in sensor selection," *Discrete Event Dynamic Systems: Theory and Applications*, vol. 12, no. 4, pp. 417–445, Oct. 2002.

[13] T.-S. Yoo and S. Lafortune, "NP-completeness of sensor selection problems arising in partially-observed discrete event systems," *IEEE Trans. Automatic Control*, vol. 47, no. 9, pp. 1495–1499, Sept. 2002.

[14] K. Rohloff, S. Khuller, and G. Kortsarz, "Approximating the minimal sensor selection for supervisory control," *Discrete Event Dynamic Systems: Theory and Applications*, vol. 16, no. 1, pp. 143 – 170, Jan. 2006.

[15] Y. Huang, K. Rudie, and F. Lin, "Decentralized control of discrete-event systems when supervisors observe particular event occurrences," *IEEE Trans. Automatic Control*, vol. 53, no. 1, pp. 384–388, Feb. 2008.

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

[17] W. Wang, S. Lafortune, and F. Lin, "An algorithm for calculating indistinguishable states and clusters in finite-state automata with partially observable transitions," *Systems & Control Letters*, vol. 59, no. 9-10, pp. 656–661, September-October 2007.