

Minimization of Communication of Event Occurrences in Acyclic Discrete Event Systems

Weilin Wang, Stéphane Lafortune, and Feng Lin

Abstract—The problem of minimizing communication of event occurrences in systems modeled by finite-state automata is considered. There are n communicating agents observing the behavior of the system for purposes of control or diagnosis. A set of communication policies for the agents is said to be minimal if communications of event occurrences cannot be removed without affecting the correctness of the solution. Under an assumption on the absence of cycles (other than self-loops) in the system model, an algorithm that computes a set of minimal communication policies in polynomial time in the number of states of the system is presented.

I. INTRODUCTION

There is a long-standing interest in minimizing communication among nodes in distributed dynamic systems for reasons of power, bandwidth, security, or network topology. A survey of work on minimization of communication in the context of diagnosis or control of logical (untimed) models of discrete event systems where the information is decentralized can be found in [1]; recent work includes [2]–[5]. Our approach considers a general problem of minimization of communication of event occurrences and it can be used in both control and diagnosis applications. Our contribution is to propose a computationally-efficient algorithm for solving minimum communication problems for a special class of systems.

The system is modeled by an automaton G , with n agents observing the behavior of G using their own sets of sensors. The agents are able to communicate among each other with negligible delay as compared with the dynamics of the system. Agents communicate *event occurrences* to each other. The communication structure allows agent j to immediately relay to agent k information it just received from agent i about an event occurrence. The agents are working as a team to accomplish some unspecified task (monitoring, diagnosis, or control). For this purpose, they need to be able to distinguish unambiguously among certain pairs of states in the state space of G ; the pairs that need to be distinguished are assumed specified at the outset. This requirement is called the *local specification condition* and it necessitates the exchange of information about event occurrences in real time among agents. Moreover, for the purpose of realizing the communication policy to be designed for each agent, the agent will need to distinguish further pairs of states if these have different communication decisions for some common events associated with them; this latter requirement is called *feasibility* of the

communication policy. The objective is to find a *minimal* set of communication policies among the agents. The notion of minimality is a logical one: a communication policy is minimal if removing one or more communications of event occurrences in the dynamic evolution of the system renders a correct solution incorrect in terms of either the local specification condition or the feasibility condition.

Our problem formulation is different from that in [2]–[5]. We borrow some of the features of the problem treated in [6], [7], including a similar notion of minimality of communication. However, our treatment is different in several respects: number of agents (more than two), communication structure (relaying of information allowed), local specification condition (more general than those in [6], [7]), and algorithmic procedure. In order to obtain a more computationally-efficient solution procedure than that of [6], [7], an additional assumption is made regarding the absence of cycles other than self-loops in G . An important benefit of this assumption is that it is now possible to synthesize communication policies in *polynomial-time complexity in the size of the state space of G* . This result is one of the main contributions of this paper and it is in contrast to the work in [6], [7] as well as to the other relevant works reviewed in [1]. Our results were first reported in preliminary form in [8].

II. DESCRIPTION OF PROBLEM

A. System Model

We model the untimed discrete event system as a deterministic finite-state automaton $G = (X, E, f, \Gamma, x_0)$ where X is the finite set of states, E is the finite set of events, $f : X \times E \rightarrow X$ is the 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. The transition function f is extended to domain $X \times E^*$ in the usual manner. $\mathcal{L}(G)$ is the language generated by G . The set of transitions $TR(G)$ of G is

$$TR(G) := \{(x, e) \in X \times E : e \in \Gamma(x)\}. \quad (1)$$

We define $x \in X$ to be a *leaf* state of G if $(\forall (x, e) \in TR(G)) f(x, e) = x$.

Assume that there are n agents that observe the behavior of G . The set of agents is denoted by $\mathcal{A} = \{1, 2, \dots, n\}$. The set of events that can be observed by agent i is denoted by E_i , $i \in \mathcal{A}$. We assume that every event is observed by at least one agent: $E = \bigcup_{i=1}^n E_i$.

This research was supported in part by NSF grants CCR-0325571, ECS-0624821, and ECS-0624828.

Weilin Wang and Stéphane Lafortune are with the University of Michigan, Ann Arbor, MI 48109, {weilinw, stephane}@umich.edu. Feng Lin is with Wayne State University, Detroit, MI 48202, flin@ece.eng.wayne.edu

B. Communication Model

For the sake of generality, we start by presenting a language-based model of communication during the evolution of G . Then we show how to restrict this general model to one that is based on the state space of G . Communications of event occurrences occur from one agent to another. For agents $i, j \in \mathcal{A}$, $i \neq j$, the event occurrences communicated from agent i to agent j after each string in $\mathcal{L}(G)$ are described by the function $com_{ij} : \mathcal{L}(G) \rightarrow 2^E$. By adding $com_{ii} : \mathcal{L}(G) \rightarrow \{\emptyset\}$ with $com_{ii}(s) = \emptyset$ for all $s \in \mathcal{L}(G)$, we define the $n \times n$ matrix $\mathcal{C}\mathcal{O}\mathcal{M} = [com_{ij}]$, where $i, j \in \mathcal{A}$. Given communication matrix $\mathcal{C}\mathcal{O}\mathcal{M}$, we use mutual induction to define the information mapping $\theta_i : \mathcal{L}(G) \rightarrow E^*$, $i \in \mathcal{A}$, as follows. For the empty string ε , $\theta_i(\varepsilon) = \varepsilon$, and for all $se \in \mathcal{L}(G)$,

$$\theta_i(se) = \begin{cases} \theta_i(s)e & \text{if } e \in E_i \cup (\cup_{j \in \mathcal{A}, j \neq i} com_{ji}(s)) \\ \theta_i(s) & \text{otherwise.} \end{cases} \quad (2)$$

In words, after the occurrence of s , the next event e is *known* to agent i iff it is either directly observed by agent i or communicated to it by some other agent. We form the vector of information mapping $\Theta = [\theta_i : i = 1, 2, \dots, n]$.

Given $\mathcal{C}\mathcal{O}\mathcal{M}$, event $e \in E$, and string $se \in \mathcal{L}(G)$, we construct a graph where nodes are agents and arc (i, j) is defined if $e \in com_{ij}(s)$. The set of all chains in this graph is denoted by $\zeta(\mathcal{C}\mathcal{O}\mathcal{M}, s, e)$. For $c \in \zeta(\mathcal{C}\mathcal{O}\mathcal{M}, s, e)$, $c(i, k)$ means that chain c starts at node i and ends at node k . We note that $c(i, i)$ is a trivial chain of node i with no arc.

Not all arbitrary communication matrices $\mathcal{C}\mathcal{O}\mathcal{M}$ will be “feasible” based on the information available to agents. First, an agent can only send out information about the occurrence of an event if it has such information as indicated in Eqn. (3) below. Second, for each communication, the original information source must come from direct observation at an agent. Moreover, to guarantee feasibility, it is required that any two sequences of events that are indistinguishable to an agent must be followed by the same communications for events “known” by the agent. Namely, communication matrix $\mathcal{C}\mathcal{O}\mathcal{M}$ must be “compatible” with the vector of information mapping Θ that is built from it as indicated in Eqn. (4) below. In all, for all $i \in \mathcal{A}$, com_{ij} is said to be *feasible* if

$$(\forall se \in \mathcal{L}(G)) e \in com_{ij}(s) \Rightarrow [\theta_i(se) = \theta_i(s)e] \wedge [(\exists c(k, i) \in \zeta(\mathcal{C}\mathcal{O}\mathcal{M}, s, e)) e \in E_k] \quad (3)$$

and

$$(\forall se, s'e \in \mathcal{L}(G)) \theta_i(se) = \theta_i(s'e) = \theta_i(s)e = \theta_i(s')e \Rightarrow [e \in com_{ij}(s) \Leftrightarrow e \in com_{ij}(s')]. \quad (4)$$

To check feasibility, we first calculate Θ from $\mathcal{C}\mathcal{O}\mathcal{M}$, and then check if Eqns. (3) and (4) hold.

The above definitions of $\mathcal{C}\mathcal{O}\mathcal{M}$ and Θ are language-based. For their implementation on automaton G , we will define their state-based counterparts. For this purpose, we introduce a so-called “implementability condition” that restricts the richness of communication policies to the state structure of G . This condition is reminiscent of the implementability condition in [6].

Implementability Condition on G :

$$(\forall i, j \in \mathcal{A})(\forall s, s' \in \mathcal{L}(G)) f(x_0, s) = f(x_0, s') \Rightarrow [e \in com_{ij}(s) \Leftrightarrow e \in com_{ij}(s')]. \quad (5)$$

More specifically, for implementability, the event occurrences to be communicated from agent i to agent j are described by

$$COM_{ij} \subseteq TR(G), \quad (6)$$

where $(x, e) \in COM_{ij}$ means that

$$((\forall s \in \mathcal{L}(G)) f(x_0, s) = x) e \in com_{ij}(s). \quad (7)$$

The implementability condition (5) is imposed in order to restrict the space of communication policies over which the optimization problem will be solved. The next step is to specialize the notion of feasibility to COM_{ij} .

The transitions observed by or communicated to agent i are described by the index function $\mathcal{I}_i : TR(G) \rightarrow \{0, 1\}$ defined as

$$\mathcal{I}_i(x, e) = \begin{cases} 1 & \text{if } e \in E_i \text{ or } (x, e) \in (\cup_{j \in \mathcal{A}, j \neq i} COM_{ji}) \\ 0 & \text{otherwise} \end{cases}. \quad (8)$$

By adding $COM_{ii} = \emptyset$, we define the $n \times n$ state-based communication matrix $COM = [COM_{ij}]$. Given COM , we can retrieve the corresponding $\mathcal{C}\mathcal{O}\mathcal{M}$ as follows:

$$com_{ij}(s) = \{e \in E : (f(x_0, s), e) \in COM_{ij}\}. \quad (9)$$

We say that COM is feasible if the corresponding $\mathcal{C}\mathcal{O}\mathcal{M}$ is feasible. For each transition $(x, e) \in TR(G)$, we construct a graph where nodes are agents and arc (i, j) is defined if $(x, e) \in COM_{ij}$. The set of all chains in this graph is denoted by $\zeta(COM, (x, e))$. For chain $c \in \zeta(COM, (x, e))$, $c(i, k)$ means that c starts at node i and ends at node k . From Eqns. (3), (4), (5), (6) and (8), the notion of feasibility is specialized to COM as follows. For all $i, j \in \mathcal{A}$, COM_{ij} is *feasible* iff

$$(\forall (x, e) \in TR(G)) (x, e) \in COM_{ij} \Rightarrow [\mathcal{I}_i(x, e) = 1] \wedge [(\exists c(k, i) \in \zeta(COM, (x, e))) e \in E_k] \quad (10)$$

and

$$(\forall se, s'e \in \mathcal{L}(G)) \theta_i(se) = \theta_i(s'e) = \theta_i(s)e = \theta_i(s')e \Rightarrow [(f(x_0, s), e) \in COM_{ij} \Leftrightarrow (f(x_0, s'), e) \in COM_{ij}]. \quad (11)$$

Overall, COM is feasible if

$$\forall i, j \in \mathcal{A}, COM_{ij} \text{ is feasible.} \quad (12)$$

Here, for given COM , the test of Eqn. (10) is straightforward and the test of Eqn. (11) is treated in Section III-C.

We allow the possibility that two agents may not be directly connected with each other. But we require that any two agents can communicate with each other at least indirectly, i.e., the connectivity graph of the agents is strongly connected. To specify connectivity, we use, for each agent $i \in \mathcal{A}$, a set of agents $A_i^{nc} \subseteq \mathcal{A}$ to denote that agent i cannot directly communicate with agent $j \in A_i^{nc}$, i.e.,

$$(\forall j \in A_i^{nc}) COM_{ij} = \emptyset. \quad (13)$$

We form the vector A^{nc} as $A^{nc} = [A_i^{nc} : i = 1, \dots, n]$. Given an A^{nc} , we construct a graph where nodes are agents and arc (i, j) is defined if $j \notin A_i^{nc}$. The set of all chains in this graph is denoted by $\zeta(A^{nc})$. For chain $c \in \zeta(A^{nc})$, $c(i, k)$ means that c starts at node i and ends at node k . Then the strong connectivity requirement is that

$$(\forall i, j \in \mathcal{A}) \exists c(i, j) \in \zeta(A^{nc}). \quad (14)$$

Under Eqn. (14), it can be verified that the communication policy $COM = [COM_{ij}]$, where

$$COM_{ij} = \begin{cases} \{(x, e) : e \in \Gamma(x)\} & j \notin A_i^{nc} \text{ and } j \neq i \\ \emptyset & \text{otherwise,} \end{cases} \quad (15)$$

is feasible, and

$$(\forall i \in \mathcal{A})(\forall (x, e) \in TR(G)) \mathcal{S}_i(x, e) = 1, \quad (16)$$

i.e., under this specific policy, all agents can perfectly observe the system G .

Remark: Notation – For given matrices of sets $B = [b_{ij}]$ and $B' = [b'_{ij}]$ for $i = 1, \dots, l$ and $j = 1, \dots, m$, (an example of such a matrix is COM), B is a subset (superset) of B' , denoted by $B \subseteq (\supseteq) B'$, if

$$[(\forall i, j) b_{ij} \subseteq (\supseteq) b'_{ij}] \wedge [(\exists i, j) b_{ij} \subset (\supset) b'_{ij}]. \quad (17)$$

The notation $B \subseteq (\supseteq) B'$ is defined similarly in the obvious way. For given matrices of sets $B = [b_{ij}]$ and $B' = [b'_{ij}]$, we define $B \setminus B' = [b_{ij} \setminus b'_{ij}]$.

C. Objective for Agents

Besides the feasibility requirements specified by Eqns. (10) and (11), we require that each agent be able to distinguish certain pairs of states of G for its own control, diagnosis, or other purpose. Formally, for each agent $i \in \mathcal{A}$, we specify a relation $T_{i,spec} \subseteq X \times X$, which is called the *local specification condition*. We require that no state pair $(x, y) \in T_{i,spec}$ be indistinguishable from the viewpoint of agent i , i.e., for all $s, s' \in \mathcal{L}(G)$,

$$\theta_i(s) = \theta_i(s') \Rightarrow (f(x_0, s), f(x_0, s')) \notin T_{i,spec}. \quad (18)$$

Other agents must communicate sufficient information to agent i so that the above requirement is satisfied.

Using $T_{i,spec}$ to describe the requirement of agent i is a rather general approach. In supervisory control for instance, key properties are controllability and observability. One can deal with controllability first by computing the supremal controllable sublanguage of the desired (specification) language. Second, one can transform G so that the supremal controllable sublanguage of the specification is generated by a subautomaton of G . In the last step, the requirement of observability can be translated into a state disambiguation problem, as described in [9]. All of the above operations can be done in polynomial complexity in the state space of G and of the specification language.

D. Problem Statement

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

Input: System $G = (X, E, f, \Gamma, x_0)$, set of strongly connected agents \mathcal{A} , and set of local specification conditions $T_{i,spec}$, $i \in \mathcal{A}$, for those agents.

Goal: Calculate a minimal communication $COM^* = [COM^*_{ij}]$, where $COM^*_{ij} \subseteq TR(G)$ such that:

- C1. COM^* is feasible.
- C2. The local specification condition $T_{i,spec}$ is satisfied for each i , i.e., for all $s, s' \in \mathcal{L}(G)$

$$\theta_i^*(s) = \theta_i^*(s') \Rightarrow (f(x_0, s), f(x_0, s')) \notin T_{i,spec} \quad (19)$$

where θ_i^* is the information map obtained from COM^* .

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

E. Obtaining a Computationally Efficient Algorithm

Communications of event occurrences among agents are interdependent. That is, in general, the determination by agent i of when and whom to communicate to depends upon its own “observation” of the system. But communications from agent i affect the “observation” of the system of other agents, and, consequently, the communication policies of these agents to agent i . Thus it affects the “observation” of agent i . Due in part to such interdependency, which is well-documented in [1] and the references therein, it is fair to say that computational obstacles are *the* major hurdle in any of the solution procedures that have been proposed so far for the different types of minimization of communication problems considered in the discrete event systems literature. For the problem posed in the preceding section, we can always get a solution by exhaustive search since the solution space has finite cardinality. Therefore, our objective is to find subclasses of systems for which computationally efficient algorithms exist. For this purpose, we make the following assumption for the remainder of this paper (except Section III-C):

Assumption (A1): The graph of G has no cycles other than self-loops, i.e., for all $x \in X$, for all $e \in E$ and $t \in E^*$, $[f(x, et)$ is defined $\wedge f(x, e) \neq x \Rightarrow f(x, et) \neq x$.

There are classes of systems with repetitive behavior where the only loops are from marked (aka final) states to the initial state. This is often the case in modeling automated manufacturing systems; it is also the case in systems that model “missions” that must be performed by a set of communicating agents. For such systems, Assumption (A1) is not restrictive as these loops can be opened and the resulting problem can be solved using the algorithm to be presented in the next section. In this way, we can compute a minimal communication policy for one “run” of the system.

For the purpose of solving the problem of Section II-D, we ask the following question: Can we remove communications for transitions in $TR(G)$ *one by one* until we get a minimal set satisfying conditions (C1) and (C2)? Unfortunately, the answer is “no”: removing communications of event occurrences one by one will not work in general. The reason is a so-called “lack of monotonicity” of the set of distinguishable states with respect to the set of observed event occurrences. Namely, is it possible that two states of G that are indistinguishable under a feasible policy COM will become distinguishable under a strict subset of COM that is still feasible. The reader is referred to [10] for an example. Such lack of monotonicity does not occur when the entire set of observable events is the parameter, as opposed to $TR(G)$. (Note that other instances of lack of monotonicity have been reported in the literature; for example, it is shown in [11] that the “observer property” of projections [12] is not monotone with respect to the set of observable events.)

The preceding discussion shows that when removing communications of event occurrences, we need to consider *sets*

of transitions. A naive approach is to consider all possible $COM = [COM_{ij}]$, where $COM_{ij} \subseteq TR(G)$. For each COM , we check conditions (C1) and (C2). This can be done, for example, by building an observer based on COM (e.g., as is done in [6]) and then check (C1) and (C2) by examining observer states. If COM does not satisfy (C1) and (C2), we assign 0 to it; otherwise we assign 1 to it. We can always find a minimal COM with assigned value 1. Any minimal COM will be a solution to our problem. The number of iterations in the above approach is linear with respect to the cardinality of the set of all possible COM . In the case of n agents where any two agents can communicate with each other, there are $n(n-1)$ COM_{ij} 's that need to be considered in COM . For each COM_{ij} , agent i needs to make decisions for each transition in $TR(G)$, which in the worst case has cardinality $|X \times E|$. Considering all possible combinations, the set of all possible COM is in the order of $2^{n(n-1)|X \times E|}$. Moreover, in each test, the construction of the observer has a worst-case complexity in the order of $|E| \times 2^{|X|}$. Totally, we have a worst-case complexity of $\mathcal{O}(|E|2^{(n(n-1)|E|+1)|X|})$ for this exhaustive search approach. (Note that in most cases, the number of states is far larger than the number of events or the number of agents.)

One of the main contributions of this paper is to make the number of iterations *linear* with respect to $|X|$ by exploiting Assumption (A1). This assumption allows to decompose the problem by ordering the states in a topological sort where there exists a monotonicity property for the removal of communications. The specifics of the algorithm are presented in the next section. However, linear iterations does not mean polynomial complexity in state space within each iteration. To overcome the exponential complexity in the number of states in X at each iteration, we will not construct an observer, but rather, we will show in Section III-C how to construct in polynomial time in both $|X|$ and $|E|$ the list of all state pairs that are indistinguishable under a given COM .

III. PROBLEM SOLUTION

A. Algorithm MIN-COM-GEN

We present the main algorithm, called Algorithm MIN-COM-GEN, for finding a minimal communication COM^* .

Algorithm MIN-COM-GEN:

Step 1: Initialization.

- 1.1 Define $\mathcal{C} : X \rightarrow \{0, 1\}$, and set $\mathcal{C}(x) = 0, \forall x \in X$.
- 1.2 Start with communication¹ $COM = [COM_{ij}]$, where COM satisfies conditions (C1) and (C2) in Section II-D and in addition

$$(\forall i \in \mathcal{A})(\forall (x, e) \in TR(G)) \mathcal{I}_i(x, e) = 1 \quad (20)$$

where \mathcal{I}_i is the index function corresponding to COM .

Step 2: Find a state $x_l \in X$ that satisfies the following conditions²:

- i. $\mathcal{C}(x_l) = 0$, and

- ii. $(\forall e \in \Gamma(x_l)) (f(x_l, e) = x_l) \vee (f(x_l, e) = x_k \Rightarrow \mathcal{C}(x_k) = 1)$.

Step 3: Remove communications of event occurrences at state x_l as follows.

Consider the matrix of potentially removable communications $PRC(x_l) = [PRC_{ij}(x_l)]$, where $PRC_{ij}(x_l)$ is the set of communications from agent i to agent j at state x_l , which are specified by Step 1.2³. Find an $RC^*(x_l) \subseteq PRC(x_l)$, such that under

$$COM \leftarrow COM \setminus RC^*(x_l)$$

conditions I - III below are satisfied:

- I. COM is feasible;
- II. The local specification conditions are satisfied;
- III. There does not exist $RC'(x_l) \subseteq PRC(x_l)$ s.t. $RC'(x_l)$ satisfies I and II and $RC'(x_l) \supset RC^*(x_l)$.

Step 4: Set $\mathcal{C}(x_l) = 1$.

Step 5: If there exists $x \in X$ such that $\mathcal{C}(x) = 0$, repeat Steps 2 to 4.

Otherwise, set $COM^* \leftarrow COM$. \square

Algorithm MIN-COM-GEN works as follows. After the necessary initialization in Step 1, each state of G is examined in an order determined by a topological sort of the graph as specified by Step 2. The major subroutine of the algorithm is in Step 3, where a matrix $RC^*(x_l)$ of maximal removable communications at state x_l is calculated. MIN-COM-GEN is a “general” algorithm because the specific method employed for searching over all candidate $RC^*(x_l)$ is not specified. The testing of conditions I and II of Step 3 can be done in polynomial time in the state space of G , according to the methods to be presented in Section III-C. Finding $RC^*(x_l)$ by exhaustive search over all the subsets of the active event set at each state of G will of course work. In special problem instances, where more structural assumptions about the system model are made, more efficient ways of searching over these subsets can be devised, as in the case considered in [13].

B. Properties of Algorithm MIN-COM-GEN

The main element of the proof of correctness of Algorithm MIN-COM-GEN is the minimality of the solution that it returns. This proof requires several intermediate results. Due to space limitations, these results are stated without proofs.⁴

For $x \in X$ and $V \subseteq TR(G)$, we construct the subautomaton $\hat{G}(x, V)$ of G by defining x to be the initial state and by only keeping the part of G that is accessible under the transitions in V . We use the notation \hat{f} for the transition function of $\hat{G}(x, V)$ and $\hat{X}(x, V)$ for its set of states. We also define $\hat{G}(x) = \hat{G}(x, TR(G))$ and $\hat{X}(x) = \hat{X}(x, TR(G))$.

Lemma 1 states that for any nonempty subset Y of states, there exists at least one state in the subset where all strings starting at that state never visit any other states in Y , i.e., once a string leaves that state, it leaves Y .

³In case such initialization is given by Eqn. (15), $PRC_{ii} = \emptyset$ and for $i \neq j$, $PRC_{ij}(x_l) = \{(x_l, e) \in TR(G) : e \in \Gamma(x_l)\}$.

⁴All proofs, as well as other intermediate results and examples of the application of MIN-COM-GEN, can be found in the technical report at URL www.eecs.umich.edu/umdes/publications.cgi.

¹An example of such COM is given by Eqn. (15).

²In case of $\mathcal{C}(x) = 0$ for all $x \in X$, Step 2 returns a leaf state.

Lemma 1: Let $L_{ns}(x_l) = \mathcal{L}(\hat{G}(x_l, TR(G))) \setminus E_{sl}^*(x_l)$, where $E_{sl}(x_l)$ is the set of events that self-loop at state x_l . For any $Y \subseteq X$, $Y \neq \emptyset$, there exists $x_l \in Y$ such that⁵ $(\forall t \in L_{ns}(x_l)) f(x_l, t) \notin Y$.

Lemma 2 shows that if the states in X have not all been examined, then Step 2 of Algorithm MIN-COM-GEN will find the next state to be examined.

Lemma 2: If there exists $x \in X$ with $\mathcal{C}(x) = 0$, then we can find an x_l in Step 2 of Algorithm MIN-COM-GEN.

Lemma 3 shows that once a state is picked by Step 2 of Algorithm MIN-COM-GEN, the communication policies of all its reachable states remain fixed afterwards.

Lemma 3: By Step 2 of Algorithm MIN-COM-GEN, at the time when Algorithm MIN-COM-GEN examines state x_k , all other states $x \in \hat{X}(x_k) \setminus \{x_k\}$ have been examined and their communications will remain unchanged afterwards.

The next two results are key to the proof of Theorem 1 below. Given automata G and G' with G' a subautomaton of G , let $COM = [COM_{ij}]$ be the communication matrix for G . Then, $COM' = [COM'_{ij}]$ is the restriction of COM to G' .

Lemma 4: For a given COM , if the feasibility and local specification are satisfied for G , then they are also satisfied for $\hat{G}(x)$, for all $x \in X$.

Proposition 1: In Step 3 of Algorithm MIN-COM-GEN, suppose that we are examining $RC(x_l)$. Let $COM' = COM \setminus RC(x_l)$ be the corresponding communication. Then the following two conditions are satisfied for G iff they are satisfied for $\hat{G}(x_l)$.

- I. COM' is feasible.
- II. The local specification conditions are satisfied.

We are now ready to prove the correctness of Algorithm MIN-COM-GEN.

Theorem 1: Algorithm MIN-COM-GEN gives a solution to the minimum communication problem formulated in Section II-D.

Proof: By contradiction, suppose that Algorithm MIN-COM-GEN gives a solution COM^* , but there exists a better solution COM^{**} with $COM^{**} \subset COM^*$. By Lemma 2, after the execution of Algorithm MIN-COM-GEN, all states $x \in X$ have been examined for removing communications. Let $RC^*(x)$ and $RC^{**}(x)$ be the communications removed at state x corresponding to COM^* and COM^{**} respectively. Since $COM^{**} \subset COM^*$, we have $RC^*(x) \subseteq RC^{**}(x)$ for all $x \in X$. Let $Z = \{x \in X : RC^*(x) \subset RC^{**}(x)\}$. Since $COM^{**} \subset COM^*$, we have $Z \neq \emptyset$. By Lemma 1, we can find an $x_k \in Z$, such that for all $t \in \mathcal{L}(\hat{G}(x_k)) \setminus E_{sl}(x_k)^*$, $f(x_k, t) \notin Z$, where $E_{sl}(x_k)$ is the set of events that self-loop at state x_k .

Let COM be the matrix of communications just before Algorithm MIN-COM-GEN examined state x_k . Let $COM'' = COM \setminus RC^{**}(x_l)$ and $COM' = COM \setminus RC^*(x_l)$. Using the results developed previously, we have:

COM^{**} is a solution to the minimal communication problem
 $\Rightarrow COM^{**}$ satisfies (I) and (II) for G
 $\Rightarrow COM^{**}$ satisfies (I) and (II) for $\hat{G}(x_k)$ (by Lemma 4)
 $\Rightarrow COM''$ satisfies (I) and (II) for $\hat{G}(x_k)$
 (by Lemma 3, communications remain unchanged in

$\hat{X}(x_k) \setminus \{x_k\}$)

$\Rightarrow COM''$ satisfies (I) and (II) for G (by Proposition 1).

On the other hand, since COM^* is the solution obtained by Algorithm MIN-COM-GEN, $RC^*(x)$ is a maximum set whose corresponding communications COM' satisfy (I) and (II). This leads to a contradiction to Step 3 of Algorithm MIN-COM-GEN because $RC^{**}(x) \supset RC^*(x)$, and the corresponding COM'' satisfies (I) and (II) for $\hat{G}(x_k)$. \square

By using Algorithm MIN-COM-GEN together with observers for the tests required in Step 3, we can solve the minimization of communication problem formulated in Section II-D for acyclic systems in a worst-case computational effort of $\mathcal{O}(|E||X|2^{n(n-1)|E|+|X|})$. To achieve polynomial complexity in terms of the size of the state space of G , we need a different testing method for Step 3 than the construction of observers. This is the topic of the next section.

C. Tests in Step 3 of Algorithm MIN-COM-GEN

We now discuss how to efficiently implement the testing of conditions (I) and (II) at each agent for each candidate subset $RC^{cand} \subseteq PRC(x_l)$ in Step 3 of Algorithm MIN-COM-GEN. This is equivalent to testing all equations that specify these two conditions. Testing of Eqn. (10) is straightforward and can be easily resolved by the corresponding definitions. For tests of Eqns. (11) and (18), it is important, at each agent k , to construct the list of state pairs in $X \times X$ that are “confused” under the information mapping θ_k^{cand} that results from RC^{cand} . We say that $(x, y) \in X \times X$ is a *confused pair* for agent k if

$$\begin{aligned} & (\exists s, s' \in \mathcal{L}(G)) [f(x_0, s) = x, f(x_0, s') = y] \\ & \wedge [\theta_k^{cand}(s) = \theta_k^{cand}(s')]. \end{aligned} \quad (21)$$

Observers could be used to identify all confused state pairs at each agent under a given communication policy, but the worst-case computational complexity of their construction is exponential in $|X|$. Instead, we propose two methods that can identify all confused pairs in polynomial time in both $|X|$ and $|E|$. The first method is called the “table technique” and it can be found in [9]. The second method that we have developed is called the “product automaton technique.” Inspired by the approach in [14], we construct a nondeterministic product automaton from G (actually, $\hat{G}(x_l)$ suffices) and a given communication policy with corresponding index function \mathcal{I}_k for agent k (defined in Eqn. (8)). Denote the product automaton by G_k^P ; its initial state is (x_0, x_0) , where x_0 is the initial state of G . Its nondeterministic transition function $f_k^P((x, y), e)$ is defined as follows:

$$f_k^P((x, y), e) = \begin{cases} (f(x, e), f(y, e)) & \text{if } \mathcal{I}_k(x, e) = \mathcal{I}_k(y, e) = 1 \\ (f(x, e), y) & \text{if } \mathcal{I}_k(x, e) = \mathcal{I}_k(y, e) = 0 \\ (x, f(y, e)) & \text{if } \mathcal{I}_k(x, e) = \mathcal{I}_k(y, e) = 0 \\ (f(x, e), y) & \text{if } (\mathcal{I}_k(y, e) = 1 \vee (y, e) \notin TR(G)) \\ & \wedge \mathcal{I}_k(x, e) = 0 \\ (x, f(y, e)) & \text{if } (\mathcal{I}_k(x, e) = 1 \vee (x, e) \notin TR(G)) \\ & \wedge \mathcal{I}_k(y, e) = 0 \\ \text{undefined} & \text{otherwise.} \end{cases} \quad (22)$$

Let the accessible state space for G_k^P under the above transition function f_k^P be X_k^P . For agent k and $x, y \in X$, the fact that state

⁵In case $Y = X$, all leaf states satisfy Lemma 1.

x can be confused with state y is equivalent to the existence of state $(x, y) \in X_k^P$ corresponding to \mathcal{S}_k . (Proofs of these properties are straightforward and omitted.) Clearly, Eqn. (11) for feasibility holds iff

$$\begin{aligned} & (\nexists (x, y) \in X_i^P) [(x, e), (y, e) \in TR(G)] \\ & \wedge [\mathcal{S}_i(x, e) = \mathcal{S}_i(y, e) = 1] \\ & \wedge [(x, e) \in COM_{ij} \wedge (y, e) \notin COM_{ij}] \end{aligned} \quad (23)$$

for all $i, j \in \mathcal{A}$. It is also clear that Eqn. (18) for the local specification condition holds iff

$$(\forall i \in \mathcal{A}) X_i^P \cap T_{i,spec} = \emptyset \quad (24)$$

The computational effort of the above methods is determined by the complexity of the product automaton (or of the table of indistinguishable state pairs) plus the complexity of testing Eqns. (11) and (18). The computational effort for using the product automaton is upper bounded by $|X|^2 \times |E|$; this is of the same order as testing Eqns. (11) and (18). (The complexity of the table technique is also of the same order.) Therefore, testing of C1 and C2 in Step 3 of Algorithm MIN-COM-GEN can be done in an order of $\mathcal{O}(|X|^2 \times |E|)$. Note that the results in this section still apply even if Assumption (A1) does not hold.

D. Complexity of Algorithm MIN-COM-GEN

As a consequence of the iterative procedure in Algorithm MIN-COM-GEN and of the strategy described in Section III-C for the implementation of Step 3, we have the following result.

Theorem 2: The problem formulated in Section II-D can be solved with polynomial time complexity with respect to the cardinality of the state space of G .

IV. CONCLUSION

Algorithm MIN-COM-GEN is the first of its kind to achieve polynomial time complexity in the state space of the system for minimization of communications of event occurrences. In the context of Algorithm MIN-COM-GEN, the specific details on how to efficiently implement the search over candidate solutions in Step 3 is left unspecified. We have identified a special case, involving a set of agents each communicating with a central station, where the search in Step 3 can be organized in a way that is of polynomial complexity in the set of *agents* and in the set of *active events* at each state. These results are presented in [13]. It would be of interest to find other special cases than the one in [13] where Step 3 of Algorithm MIN-COM-GEN is amenable to efficient implementations.

The correctness proof of Algorithm MIN-COM-GEN is centered on the “decomposition” result in Proposition 1, which is itself dependent on the acyclic assumption (A1). It remains an open problem to identify other ways to circumvent the lack of monotonicity of candidate solutions mentioned in Section II-E and obtain computationally-tractable algorithms for solving communication problems in more general classes of systems.

REFERENCES

- [1] J. H. van Schuppen, “Decentralized control with communication between controllers,” in *Unsolved Problems in Mathematical Systems and Control Theory*, V. D. Blondel and A. Megretski, Eds. Princeton: Princeton University Press, 2004, pp. 144–150.
- [2] S. Tripakis, “Decentralized control of discrete-event systems with bounded or unbounded delay communication,” *IEEE Trans. Automatic Control*, vol. 49, no. 9, pp. 1489–1501, Sept. 2004.
- [3] S. Ricker and B. Caillaud, “Mind the gap: Expanding communication options in decentralized discrete-event control,” *46th IEEE Conference on Decision and Control*, pp. 5924–5929, Dec. 2007.
- [4] A. Mannani and P. Gohari, “Decentralized supervisory control of discrete-event systems over communication networks,” *IEEE Trans. Automatic Control*, vol. 53, no. 2, pp. 547–559, March 2008.
- [5] S. Ricker, “Asymptotic minimal communication for decentralized discrete-event control,” in *Proceedings of the 9th International Workshop on Discrete Event Systems*, 2008, pp. 486–491.
- [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, “Minimization of communication in distributed discrete event systems,” in *Proceedings of the 2007 European Control Conference*, 2007.
- [9] —, “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.
- [10] S. Lafortune, “On decentralized and distributed control of partially-observed discrete event systems,” in *Advances in Control Theory and Applications*, edited by C. Bonivento, A. Isidori, L. Marconi, and C. Rossi, in *Lecture Notes in Control and Information Sciences*, Vol. 353. Springer, 2007, pp. 171–184.
- [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] K. Wong and W. Wonham, “Hierarchical control of discrete-event systems,” *Discrete Event Dynamic Systems: Theory and Applications*, vol. 6, no. 3, pp. 241–273, July 1996.
- [13] 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.
- [14] J. N. Tsitsiklis, “On the control of discrete-event dynamical systems,” *Math. Control Signals Systems*, vol. 2, pp. 95–107, 1989.