Decentralized Supervisory Control with Communicating Controllers

George Barrett, *Member, IEEE,* and Stéphane Lafortune, *Fellow, IEEE*

***Abstract*—The decentralized control problem for discrete-event systems addressed in this paper is that of several communicating supervisory controllers, each with different information, working in concert to exactly achieve a given legal sublanguage of the uncontrolled system's language model. A novel information structure model is presented for dealing with this class of problems. Existence results are given for the cases of when controllers do and do not anticipate future communications, and a synthesis procedure is given for the case when controllers do not anticipate communications. Several conditions for optimality of communication policies are presented, and it is shown that the synthesis procedure yields solutions, when they exist for this class of controllers, that are optimal with respect to one of these conditions.**

***Index Terms*—Communicating controllers, decentralized control, discrete-event systems.**

I. Introduction

THE DECENTRALIZED nature of information in many large-scale systems, as well as practicality, dictate the need for supervisory control systems that are also decentralized. Decentralized control of discrete-event systems, in the absence of communication, has been well studied in previous work [1]–[11], and many classifications of language structure admitting decentralized control without communication exist, including coobservability, decomposability and strong decomposability.

Control of logical discrete-event systems with communication has been pondered recently in [12]–[15]. The fundamental alteration to the structure of the control systems is that the controllers observe events generated by the system and are allowed to pass messages in order to attempt to resolve ambiguities and determine correct control actions.

Prior work has yet to introduce a framework that is general enough to address certain fundamental questions concerning decentralized supervisory control with communication (paraphrased from [16]).

1) Who should know what and when?
2) Who should communicate with whom?
3) When should controllers communicate?
4) What should controllers communicate?

It is one of the intentions of this paper to present a framework that is general enough to address these questions. The decentralized control problem that we address is that of several communicating supervisory controllers, each with different information, working in concert to exactly achieve a given legal sublanguage of the uncontrolled system's model. There are several contributions of this work.

1. A novel information structure formalism is presented in Section II for dealing with this class of problems. The information structure, based on extended trace models, explicitly represents actions observable by each controller, which controllers communicate to other controllers, what symbols are communicated, when controllers initiate communication, and what information may be inferred by each of the controllers following any sequence of actions. Constraining specific components of the information structure yields several classes of problems.
2. Necessary and sufficient conditions are given in Section III for the existence, under certain assumptions, of solutions to the described decentralized supervisory control problem with communication. These conditions characterize the class of languages achievable by decentralized controllers that anticipate and expect future communications from other controllers.
3. Necessary and sufficient conditions are given that characterize the class of languages achievable by communicating supervisory controllers, termed "myopic," that do not anticipate future communications.
4. By comparing the classes of languages described by the previous two contributions, we elucidate the significance of controllers that anticipate future communications in decentralized supervisory control problems. Using the class of controllers that anticipate future communications has nontrivial implications for synthesis algorithms.
5. Several basic notions for optimum communication policies are given in Section IV.
6. In Section V-A, a finite version of the myopic controller information structure is presented. The controllers in this class maintain and communicate finite-state estimates.
7. A procedure is presented for finding an optimal communication policy, if one exists, for the above-mentioned class of finite myopic controllers. The main part of the procedure deals with the construction of a unique, supremal set which can then be used to determine communication requirements.

Manuscript received November 11, 1998; revised December 1, 1999. Recommended by Associate Editor, L. Dai. This work was supported in part by the DDR&E MURI on Low Energy Electronics Design for Mobile Platforms and managed by ARO under Grant ARO DAAH04-96-1-0377. This work was performed while G. Barrett was attending The University of Michigan.

G. Barrett was attending The University of Michigan. He is now with the System and Information Sciences Group, The Johns Hopkins University Applied Physics Laboratory, Laurel, MD 20723-6099 USA (e-mail: george.barrett@jhuapl.edu).

S. Lafortune is with the Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor, MI 48109-2122 USA (e-mail: stephane@eecs.umich.edu).

Publisher Item Identifier S 0018-9286(00)07498-5.

8. In Section V-D it is shown that, despite the existence of the unique supremal set mentioned above, optimal communication policies are not unique in general.

General knowledge of supervisory control and its most common notation is assumed, and for introductory material the reader is directed to [9],[17]–[19]. The organization of this paper is as follows. Section II introduces and describes a novel information structure formalism to address decentralized supervisory control problems. Given this general framework, the problem examined in this paper is then formally stated. The existence of solutions to the above-mentioned problem is examined in Section III. Two cases for existence are considered: when controllers do and do not anticipate future communications. Optimal communication policies are discussed in Section IV. A constrained class of controllers is examined in Section V, where the finite model is described and a synthesis procedure is given. The synthesis procedure is illustrated by example in Section V-C, and the policies derived from the procedure are proved optimal in Section V-D. Non-uniqueness of optimal solutions is shown in Section V-D, and the paper concludes with a summary in Section VI.

II. A General Framework for Decentralized Supervisory Control Problems

When designing communicating supervisory controllers that cooperate to achieve a desired legal behavior, the three roles of each controller must be considered: estimation, control, and communication. In general, these three roles cannot be separated, and any synthesis procedure must take all into account simultaneously. To design the control and communication policies, five specifications are necessary:

1. the events that each controller can control;
2. the symbols that each controller can communicate;
3. the information available to each controller to be used in its control and communication policies;
4. how the individual decisions of the controllers affect the plant (decision fusion);
5. constraints on the forms of the control and communication policies.

The first, second, fourth and fifth items in this list are generally specified prior to policy synthesis for a given number of controllers acting on a given system. These items are generally dictated by specific physical mechanisms and resources available in the given problem, e.g., memory, processing capabilities, energy available for storage, processing and transmission, etc. The third item refers to the *information structure* [16], [20], [21] of the control system. When the controllers are not allowed to communicate, the information structure of the problem is fixed. However, when a communication policy is to be created, the information structure is altered as the communication policy varies during synthesis.

Consider a discrete-event system modeled by an automaton G with associated language $\mathcal{L}(G)$. Recall that associated with the system is a set of events that can be disabled Σ_c, and there is a set of events, Σ_o, that can be observed by the controllers. The set of all events is denoted by Σ. The sets of uncontrollable and unobservable events are denoted by $\Sigma_{uc} = \Sigma \setminus \Sigma_c$ and $\Sigma_{uo} = \Sigma \setminus \Sigma_o$, respectively. To control the system, there is a finite set of coordinating controllers represented by $Z = \{1, 2, \cdots, n\}$. Each controller $i \in Z$ has an associated set of events $\Sigma_{c,i} \subseteq \Sigma_c$ that it can disable, a set Ξ_i of symbols that it is allowed to communicate to other controllers, and a set of events $\Sigma_{o,i} \subseteq \Sigma_o$ that it can directly observe. The events that are unobservable to each controller are given by $\Sigma_{uo,i} = \Sigma \setminus \Sigma_{o,i}$. To represent the fact that controllers have only partial observations of traces in $\mathcal{L}(G)$, a projection operator $\mathbb{P}_i$: $\Sigma^* \rightarrow \Sigma_{o,i}^*$ is used. Recall that $\mathbb{P}_i(\sigma) = \sigma$ if $\sigma \in \Sigma_{o,i}$ otherwise $\mathbb{P}_i(\sigma) = \varepsilon$, and $(\forall s\sigma \in \Sigma^*)$ $\mathbb{P}_i(s\sigma) = \mathbb{P}_i(s)\mathbb{P}_i(\sigma)$. If a subscript is not given, e.g. $\mathbb{P}$, then it is assumed the codomain is Σ_o^*. The inverse projection of $\mathbb{P}_i$ is the mapping $\mathbb{P}_i^{-1}$: $\Sigma_{o,i}^* \rightarrow 2^{\Sigma^*}$ defined as $\mathbb{P}_i^{-1}(\omega) = \{s \in \Sigma^* | \mathbb{P}_i(s) = \omega\}$.

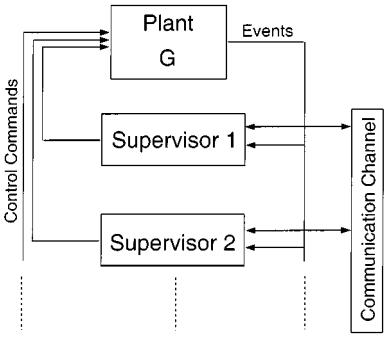

Fig. 1. Multiple controller supervision with communication.

Regarding control, numerous voting schemes can be used to combine the control actions of the controllers in Z [7], [22]. However, regardless of how the control signals are combined, the closed-loop language of the plant, G, under control of the set of supervisors, Z, is denoted by $\mathcal{L}(Z/G)$. The controllers have access to a communication channel which allows the sharing of information. This general system topology is shown in Fig. 1. For the purposes of the present work, it will be assumed that channel access is only restricted in the sense that a controller is not able to "eavesdrop" on communications among other controllers. The distinction of the communication channel from the plant is somewhat artificial, and the two are shown as separate objects in Fig. 1 merely to emphasize the existence of communication between the controllers.

The representation of the dynamics of the closed-loop system with communication is extended from sets of traces over plant events to sets of *extended traces* [23] over plant events and controller communications. These extended traces will be called *trajectories*.[1] The global system trajectories produced by the plant/controller and controller/controller interactions are defined by the set

$$\mathcal{T}(Z/G) \subseteq \left(\Sigma^* \left[\mathcal{C}_\Xi^{n\times n}\right] \left[\mathcal{C}_\Xi^{n\times n}\right]\right)^*. \tag{1}$$

A communication matrix $C_\Xi \in \mathcal{C}_\Xi^{n\times n}$ is a matrix with transmitter and receiver axis and elements which are sets of symbols being communicated. The communication symbols are selected from a set Ξ. The set of all possible $n \times n$-communication matrices, $\mathcal{C}_\Xi^{n\times n}$, is assumed to contain the matrix of empty

[1]The use of "trajectory" here differs from its use in nondeterministic supervisory control literature.

sets $\emptyset^{n\times n}$ (analog of ε). The first communication matrix in (1) represents messages sent at that instant, and the second communication matrix represents messages being received at that instant. When assumptions of zero-delay and lossless communication are used, the two communication matrices can be combined, and the form of the global trajectories reduces to

$$\mathcal{T}(Z/G) \subseteq \left(\Sigma^* \left[\mathcal{C}_\Xi^{n\times n}\right]\right)^*. \tag{2}$$

This is precisely the version we use here. When a communication matrix is completely empty, i.e., no communication takes place and the matrix elements are only empty sets, then the matrix need not be represented in the trajectory: $t\emptyset^{n\times n} = t$. It is sometimes useful to ignore communications and restrict attention to the underlying event sequence of a trajectory, t, denoted by $t|_\Sigma$. This restriction is extended to sets of trajectories, e.g., $\mathcal{T}(Z/G)|_\Sigma$, by restricting each trajectory in the set. The prefix closure of a set of trajectories, T, is defined in the obvious way by

$$\overline{T} = \left\{t \in \left(\Sigma^* \left[\mathcal{C}_\Xi^{n\times n}\right]\right)^* | \exists t' \in \left(\Sigma^* \left[\mathcal{C}_\Xi^{n\times n}\right]\right)^*, tt' \in T\right\} \tag{3}$$

and, due to our convention that $t\emptyset^{n\times n} = t$, prefixes of trajectories can end with either the occurrence of an event or communication. We will always define $\mathcal{T}(Z/G) = \overline{\mathcal{T}(Z/G)}$; hence, $\mathcal{T}(Z/G)|_\Sigma = \mathcal{L}(Z/G)$.

In general, the controllers acting on the plant cannot observe all events that can occur and they cannot observe all communications that occur. Following a trajectory $t \in \mathcal{T}(Z/G)$ the unprocessed data available to a controller, $i \in Z$, for making a decision is represented by an extended trace of the form

$$t_i^k = \sigma_1^i \left[\prod_{j\in Z} \Xi_1^{j,i}\right] \sigma_2^i \left[\prod_{j\in Z} \Xi_2^{j,i}\right] \cdots \sigma_k^i \left[\prod_{j\in Z} \Xi_k^{j,i}\right] \in T_i$$

where $\sigma^i \in \Sigma_{o,i} \cup \{\varepsilon\}$, $\Xi^{j,i}$ is a set of symbols communicated from Controller j to Controller i, and the Ξ products are to be interpreted as column vectors of communicated symbols. To derive the "locally observed" trajectory sets, T_i, from the global set of trajectories a prefix-preserving projection operator, $\mathbb{T}_i$, is given by

$$\mathbb{T}_i\colon \left(\Sigma^* \left[\mathcal{C}_\Xi^{n\times n}\right]\right)^* \to \left(\Sigma_{o,i}^* \left[\mathcal{C}_\Xi^{-i}\right]\right)^* \tag{4}$$

where $\mathcal{C}_\Xi^{-i}$ is the set of ith columns of communication matrices in $\mathcal{C}_\Xi^{n\times n}$ ($i-$ will denote the ith row), and

$$\mathbb{T}_i(\sigma[C_\Xi]) = \begin{cases} \varepsilon\left[\prod_{j\in Z} C_\Xi^{j,i}\right], & \text{if } \sigma \notin \Sigma_{o,i} \text{ and } \exists j \text{ s.t. } C_\Xi^{j,i} \neq \emptyset \\ \varepsilon, & \text{if } \sigma \notin \Sigma_{o,i} \text{ and } \forall j \left(C_\Xi^{j,i} = \emptyset\right) \\ \sigma\left[\prod_{j\in Z} C_\Xi^{j,i}\right], & \text{if } \sigma \in \Sigma_{o,i} \text{ and } \exists j \text{ s.t. } C_\Xi^{j,i} \neq \emptyset \\ \sigma, & \text{if } \sigma \in \Sigma_{o,i} \text{ and } \forall j \left(C_\Xi^{j,i} = \emptyset\right) \end{cases}$$

$$\mathbb{T}_i(t\sigma[C_\Xi]) = \mathbb{T}_i(t)\mathbb{T}_i(\sigma[C_\Xi]).$$

The projection $\mathbb{T}_i(\sigma[C_\Xi])$ removes σ if it is unobservable to Controller i and removes elements of C_Ξ not received by Controller i. The inverse projection of $\mathbb{T}_i$ is the mapping

$$\mathbb{T}_i^{-1}\colon \left(\Sigma_{o,i}^* \left[\mathcal{C}_\Xi^{-i}\right]\right)^* \to 2^{\left(\Sigma^* \left[\mathcal{C}_\Xi^{n\times n}\right]\right)^*} \tag{5}$$

defined by $\mathbb{T}_i^{-1}(\omega) = \{t \in (\Sigma^*[\mathcal{C}_\Xi^{n\times n}])^* | \mathbb{T}_i(t) = \omega\}$. Again, the projection of a set of trajectories is determined by the set of projected trajectories. We then derive the set $T_i = \overline{T}_i = \mathbb{T}_i(\overline{\mathcal{T}(Z/G)})$.

Given their observations, the controllers make communication and control decisions. The control policy is $\gamma = \{\gamma_1, \gamma_2, \cdots, \gamma_n\}$ where each γ_i is a disablement map:

$$\gamma_i\colon \overline{T}_i \to 2^{\Sigma_{c,i}}$$

and the *fusion* of the individual control decisions following a trajectory, t, is denoted by $fus(\gamma, t)$. An event $\sigma \in \Sigma_c$ is disabled following t if $\sigma \in fus(\gamma, t)$. The communication policy is $\theta = \{\theta_1, \theta_2, \cdots, \theta_n\}$ where each θ_i is a communication map:

$$\theta_i\colon \overline{T}_i \times \overline{T}_i \to \prod_{k=1}^{n} 2^{\Xi_i}.$$

Note that the domains of control maps and communication maps differ. The motivation for this difference is that control decisions are *persistent* over all trajectories that map to the same locally observed trajectory, but communications generally only take place when observations change, (cf. Moore versus Mealy-type automata [24]).

Given a control/communication-policy pair that depends on $\mathbb{T} = (\mathbb{T}_1, \cdots, \mathbb{T}_n)$, the generation of the set of trajectories $\mathcal{T}(Z/G)$ is denoted $(\gamma, \theta)_\mathbb{T} \rightharpoonup \mathcal{T}(Z/G)$. An example rule set[2] for recursively generating $\mathcal{T}(Z/G)$ is given by the following constructions:

1. $\varepsilon \in \mathcal{T}(Z/G)$;
2. $[t \in \mathcal{T}(Z/G)] \wedge [t|_\Sigma \sigma \in \mathcal{L}(G)] \wedge [\sigma \notin fus(\gamma, t)] \Rightarrow t\sigma \in \mathcal{T}(Z/G)$;
3. $t, t\sigma \in \mathcal{T}(Z/G) \Rightarrow t\sigma \begin{bmatrix} \theta_1(\mathbb{T}_1(t), \mathbb{T}_1(t\sigma)) \\ \vdots \\ \theta_n(\mathbb{T}_n(t), \mathbb{T}_n(t\sigma)) \end{bmatrix} \in \mathcal{T}(Z/G)$;
4. $t, tC_\Xi \in \mathcal{T}(Z/G) \Rightarrow tC_\Xi \begin{bmatrix} \theta_1(\mathbb{T}_1(t), \mathbb{T}_1(tC_\Xi)) \\ \vdots \\ \theta_n(\mathbb{T}_n(t), \mathbb{T}_n(tC_\Xi)) \end{bmatrix} \in \mathcal{T}(Z/G)$;

where, for constructions 3 and 4, it is understood that $t\emptyset^{n\times n} = t$. The model for generating trajectories presented here does not prevent the use of "bad" communication policies that lead to Zeno[3] behavior. Indeed, communication-policy synthesis procedures should guarantee non-"bad" communication behavior resulting from the use of the generated policies.

[2]We use "example" here, for there appears to be no reason to assume such a rule set is unique.

[3]Zeno behavior is characterized by the potential for an infinite number of occurrences (of communication in this case) in a finite amount of time. The zero-delay model does not prohibit an unbounded number of communications between the occurrence of plant events.

Having addressed the issue of how a set $\mathcal{T}(Z/G)$ is generated using a given (γ, θ) pair, we can now determine when a set $\mathcal{T}(Z/G)$ is feasible given the informational constraints of each controller in the form of the $\mathbb{T}_i$'s.

Definition 2.1: A set of trajectories T, with $T|_\Sigma$ controllable with respect to $\mathcal{L}(G)$ and Σ_{uc}, is informationally consistent with respect to $\mathbb{T} = (\mathbb{T}_1, \cdots, \mathbb{T}_n)$ if there exists a control/communication pair $(\gamma, \theta)_{\mathbb{T}}$ such that $(\gamma, \theta)_{\mathbb{T}} \rightharpoonup T$.

The most simple example of a set of trajectories *not* being informationally consistent is the case where we have *one* controller that is supposed to achieve a language that is not observable. The language in this case is not informationally consistent because there are two traces following which the controller derives the same information set but must make two incompatible control decisions. The analogy of observability for a single controller that must make control decisions extends to informational consistency for multiple controllers that must make both communication and control decisions.

Only informationally consistent trajectory sets will be considered feasible for multiple-controller supervision of logical discrete-event systems. This feasibility requirement is completely consistent with existing literature on centralized and decentralized control of DEDS.

The set $\mathbb{T}_i^{-1}(\mathbb{T}_i(t)) \cap \mathcal{T}(Z/G)$ represents the finest information that can be obtained under the projection $\mathbb{T}_i$. In general, controllers may not have the required resources to determine the finest information sets, and the controllers have access only to the information provided by a map

$$\Psi_i \colon \left(\Sigma_{o,i}^* \prod_{k=1}^{n} 2^{\Xi_k} \right)^* \to 2^{(\Sigma^* [\mathcal{C}_\Xi^{n \times n}])^*} \tag{6}$$

which we will refer to as a controller's *inference map*.[4] We define the information structure of this model in the spirit of those defined in [16] and [21] by

$$\mathcal{I} := \{t, \Psi_i(\mathbb{T}_i(t)) \colon t \in \mathcal{T}(Z/G), i = 1, \cdots, n\}. \tag{7}$$

In the unconstrained case, $\Psi_i(\mathbb{T}_i(t)) = \mathbb{T}_i^{-1}(\mathbb{T}_i(t)) \cap \mathcal{T}(Z/G)$, the maximal information set. We will generally require that the information structure factors the control and communication policies, i.e.,

$$\Psi_i(\mathbb{T}_i(t)) = \Psi_i(\mathbb{T}_i(t')) \Rightarrow \gamma_i(\mathbb{T}_i(t)) = \gamma_i(\mathbb{T}_i(t')) \tag{8}$$

and

$$\left.\begin{array}{l} \Psi_i(\mathbb{T}_i(t)) = \Psi_i(\mathbb{T}_i(t')) \\ \Psi_i(\mathbb{T}_i(t'')) = \Psi_i(\mathbb{T}_i(t''')) \end{array}\right\} \Rightarrow \theta_i(\mathbb{T}_i(t), \mathbb{T}_i(t'')) = \theta_i(\mathbb{T}_i(t'), \mathbb{T}_i(t''')). \tag{9}$$

An example of a constrained inference map is represented by the case where a controller "knows" the desired language but does not "know" *a priori* that it will receive communications, that is, Ψ_i must satisfy the constraint

$$\Psi_i(t)|_\Sigma = \Psi_i(t)|_\Sigma \Sigma_{uo,i}^* \cap \mathcal{L}(H).$$

An interesting feature of this model is that it captures the fact that the inference maps (estimation), the communication maps, and the control maps cannot be designed independently. This lack of separation between the maps and the requirement that trajectory sets be informationally consistent are the principle contributors to the difficulties encountered when attempting to synthesize suitable, optimal, control and communication maps. Problems investigated in the sections that follow are now formally stated.

Problem (P): Given a plant G with generated language $\mathcal{L}(G)$, a desired behavior modeled by an automaton H with language $\mathcal{L}(H) \subseteq \mathcal{L}(G)$, and a set of controllers $Z = \{1, 2, \cdots, n\}$ with informational resources represented by $\mathbb{T} = (\mathbb{T}_1, \cdots, \mathbb{T}_n)$, construct control and communication policies for the controllers, $\gamma = \{\gamma_1, \gamma_2, \cdots, \gamma_n\}$ and $\theta = \{\theta_1, \theta_2, \cdots, \theta_n\}$ respectively, such that $(\gamma, \theta)_{\mathbb{T}} \rightharpoonup \mathcal{T}(Z/G)$ and $\mathcal{T}(Z/G)|_\Sigma = \mathcal{L}(H)$.

Designers may also be interested in solutions to Problem (P) that are optimal in some sense. That is, the control and communication policies chosen must minimize some cost function or maximize some preference relation. We will distinguish problem Problem (P) from this optimization problem, which we will refer to as Problem (P*). Examples of communication optimization include minimizing the communication alphabet Ξ, minimizing the number of trajectories with communication, minimizing the number of controllers that must communicate, etc. Similar optimization efforts regarding only observed events have appeared for centralized supervisory control and monitoring [27] and [28]. However, these problems are significantly different from the current effort due to their lack of controller multiplicity and that, here, the supervisors make both control and communication decisions. A brief formalization of several notions of optimality is the subject of Section IV.

III. Existence of Solutions

The primary purpose of this section is to present necessary and sufficient conditions for the existence of communication policies that support a solution to Problem (P) as stated in Section II. The following results deal with the case in which communication between the controllers is allowed in both directions; e.g., Controller i can communicate to Controller j, and Controller j can communicate with Controller i. We will compare the case of when controllers can anticipate receiving communications along certain traces, and can therefore adjust their estimates based on these anticipated communications, to the case of when controllers are not allowed to utilize the anticipation of future communications to affect their estimates. It is important to note that "anticipate communication" is not intended to imply that a controller knows *exactly* what communications will occur, for this would represent a causality violation. The implied meaning is that controllers have prior knowledge of, and make use of, the communication policy. The inference maps, $\Psi_1 \cdots \Psi_n$, depend upon the communication policy. The assumptions used in the results that follow are:

A.1: The plant is modeled by a finite automaton $G = (X_G, \Sigma_G, \delta_G, x_{G0})$ with associated language $\mathcal{L}(G)$, and the desired behavior is modeled by a finite automaton $H = (X_H, \Sigma_H, \delta_H, x_{H0})$ with language $\mathcal{L}(H) \subseteq \mathcal{L}(G)$. Recall that the automaton representing the product of H and G is denoted by $[H \times G] = (X_{H \times G}, \Sigma_{H \times G}, \delta_{H \times G}, x_{H0,G0})$.

[4] In earlier versions [25], [26] of this model, Ψ_i's co-domain was 2^{Σ^*}.

A.2: Controllers are synchronized on the initial state of the system.

A.3: There are no communication delays.

A.4: There are no communication losses.

A.5: Collectively, the controllers observe all observable events and control all controllable events, i.e., $\bigcup_{j\in Z}\Sigma_{o,j}=\Sigma_o$ and $\bigcup_{j\in Z}\Sigma_{c,j}=\Sigma_c$.

A.6: The control laws for the individual supervisors are Ψ_i-*permissive*; that is, following the observation of t_i, Controller i may only disable an event $\sigma\in\Sigma_{c,i}$ if σ should not be enabled following any trajectory in $\Psi_i(t_i)$. This assumption is implicitly standard for existing work in both centralized supervisory control and decentralized supervisory control without communication.

A.7: The joint action of the controllers on the system is captured by the union of the sets of disabled events.

A.8: The behavior of the controllers is restricted so that they only respond to communications initiated by the observation of an event. This eliminates the potential for unterminated cycles of communications and responses among the controllers. However, it does not limit the amount of information passed among controllers using two-way broadcast.

A.9: Controllers are able to determine from which other controllers they receive communicated symbols.

A.10: Communication between controllers is *two-way broadcast*, that is, whenever Controller i sends a message to Controller j it also sends the message to every other controller. Upon receiving a communication caused by the observation of an event, Controller j responds by sending a message to Controller i and to every other controller.

The first result of this section is formalized by the following theorem on the existence of information structures that support solutions to (P). This theorem addresses the case where controllers can utilize "knowledge" of potential future communications to affect their estimates, i.e., controllers know the communication policy.

Theorem 3.1 (Unconstrained): Let a set of permissive controllers, Z, be given that satisfy the given assumptions. A communication policy exists that supports a solution to Problem (P) iff the following two conditions hold:

1. $\mathcal{L}(H)$ is controllable[5] with respect to $\mathcal{L}(G)$ and Σ_{uc};
2. $\mathcal{L}(H)$ is observable[6] with respect to $\mathcal{L}(G)$, Σ_o and Σ_c.

Furthermore, the information structure has a finite representation and the solution to (P) can be obtained by the communication of controller state-estimates.

Theorem 3.1 implies that Problem (P) has a solution iff the desired language $\mathcal{L}(H)$ can be implemented by a centralized supervisor. In particular, a communication policy that will allow the generation of $\mathcal{L}(H)$ via decentralized controllers is for the controllers to maintain finite-state estimates, representing the set of states each controller infers the plant may be in at a particular instant, and to communicate these state estimates among all of the controllers following the occurrence of any observable event.

[5]Recall that a language K is controllable w.r.t. L and Σ_{uc} iff $\overline{K}\Sigma_{uc}\cap L\subseteq\overline{K}$.

[6]The definition of observability is recalled in Appendix A.

It will be useful to define the following function $\Delta_{H\times G}\colon 2^{X_{H\times G}}\times 2^{\Sigma^*}\to 2^{X_{H\times G}}$ as

$$\Delta_{H\times G}(X_\Delta,\mathcal{L}_\Delta)=\{x'\in X_{H\times G}|\exists\, x\in X_\Delta,\ \exists t\in\mathcal{L}_\Delta \text{ such that } x'=\delta_{H\times G}(x,t)\},$$

that is, $\Delta_{H\times G}(X_\Delta,\mathcal{L}_\Delta)$ is the set of states of $H\times G$ reachable from the states in X_Δ by traces in $\mathcal{L}_\Delta$.

Proof of Theorem 3.1: To prove Theorem 3.1, it suffices to show that there exists a communication policy for controllers that anticipate future communications that allows the "reconstruction" of the same state estimate that a centralized supervisor observing Σ_o would generate. Hence, the control actions of the centralized supervisor can be reconstructed also. A technique often useful for classifying the types of languages achievable by a class of controllers is to assume the controllers are allowed to communicate all of the time. Consider the following communication policy: "All controllers communicate their state estimates via two-way broadcast following the occurrence of every locally observed event." Thus, following the occurrence of any observable event, Assumptions A.5 and A.8, the controllers exchange state estimates. Most importantly, the controllers "expect" to receive a communication following any observable event that they do not directly observe. Here, the controllers do not communicate the actual observable event that occurs, merely their state estimates. The proof given here provides for the general case of n controllers.

At the initial state of the system, i.e., following $\varepsilon\in\mathcal{L}(H\times G)$, the state estimates for the controllers are: $(\forall\, i\in Z)$

$$E_i(\varepsilon)=\Delta_{H\times G}\left(\{x_{H0,G0}\},\,(\Sigma_{uo,i}\setminus\cup_{j\neq i}\Sigma_{o,j})^*\right). \quad (10)$$

Note that in all estimates above $\mathcal{L}_\Delta=(\Sigma_{uo,i}\setminus\cup_{j\neq i}\Sigma_{o,j})^*$ instead of $\mathcal{L}_\Delta=\Sigma^*_{uo,i}$. This is because Controller i will observe events in $\Sigma_{o,i}$, and it will receive a communication following events in $\Sigma_{o,j}$. Thus, by Assumptions A.5 and A.8 and since controllers *expect* to receive these communications, the state estimate for Controller i is updated following the occurrence of any observable event. Substituting the identity, by Assumption A.5, $\Sigma_{uo,i}\setminus\cup_{j\neq i}\Sigma_{o,j}=\Sigma_{uo}$ into (10) yields $(\forall\, i\in Z)$

$$E_i(\varepsilon)=\Delta_{H\times G}\left(\{x_{H0,G0}\},\,\Sigma^*_{uo}\right). \quad (11)$$

That is, each controller starts off with the same information due to the fact that the communication policy is taken into account within the state estimates.

Following $s\sigma\in\mathcal{L}(H\times G)$, there are three possible cases and state-estimate update rules. These rules explicitly represent how state estimates are intersected while using the knowledge of which controller(s) communicated those state estimates. For these update rules, the set of controllers which directly observe σ is denoted by Z_σ, i.e.,

$$Z_\sigma=\{i\in Z|\sigma\in\Sigma_{o,i}\}.$$

Case (i): $Z_\sigma=\emptyset$. By Assumption A.5, $\sigma\notin\Sigma_{o,i},\ \forall\, i\in Z$ implies that $\sigma\in\Sigma_{uo}$ Hence,

$$E_i(s\sigma)=E_i(s).$$

Case (ii): $\emptyset \subset Z_\sigma \subset Z$. For this case, there are two groups of controllers: those that observe σ and those that do not. The update rule is

$$E_i(s\sigma) = \left[\bigcap_{k\in Z_\sigma} \Delta_{H\times G}\left(E_k(s), (\Sigma_{uo,k} \setminus \cup_{j\neq k}\Sigma_{o,j})^*\,\sigma\right)\right] \cap \cdot \left[\bigcap_{k\notin Z_\sigma} \Delta_{H\times G}\left(E_k(s), (\Sigma_{uo,k} \setminus \cup_{j\neq k}\Sigma_{o,j})^* \cdot (\cap_{j\in Z_\sigma}\Sigma_{o,j} \setminus \Sigma_{o,k})\right)\right]. \quad (12)$$

For those controllers that do not observe σ in the above expression, the inference takes place that some event occurred that they did not observe because these controllers receive communicated state estimates from some nonempty set of controllers that did observe an event. By Assumption A.9, the controllers that do not observe σ infer that the event that occurred must have been observed by every controller initiating the communication Therefore, the event must be in the set $\cap_{j\in Z_\sigma}\Sigma_{o,j} \setminus \Sigma_{o,k}$. Recall that the controllers only communicate state estimates and *not* events. Substituting the identity $\Sigma_{uo,k} \setminus \cup_{j\neq k}\Sigma_{o,j} = \Sigma_{uo}$ into (12) yields

$$E_i(s\sigma) = \left[\bigcap_{k\in Z_\sigma} \Delta_{H\times G}\left(E_k(s), \Sigma_{uo}^*\sigma\right)\right] \cap \left[\bigcap_{k\notin Z_\sigma} \Delta_{H\times G}\left(E_k(s), \Sigma_{uo}^*\left(\cap_{j\in Z_\sigma}\Sigma_{o,j} \setminus \Sigma_{o,k}\right)\right)\right]. \quad (13)$$

Case (iii): $Z_\sigma = Z$. In this case, every controller observes σ, and the update rule is simply the first part of (12)

$$E_i(s\sigma) = \bigcap_{k\in Z} \Delta_{H\times G}\left(E_k(s), (\Sigma_{uo,k} \setminus \cup_{j\neq k}\Sigma_{o,j})^*\,\sigma\right). \quad (14)$$

Consider the state estimate generated by a centralized controller that observes all events in Σ_o. This centralized estimate, $E_{H\times G}$, is determined by

$$E_{H\times G}(\varepsilon) = \Delta_{H\times G}\left(\{x_{H0,G0}\}, \Sigma_{uo}^*\right),$$
$$E_{H\times G}(s\sigma) = \begin{cases} \Delta_{H\times G}\left(E_{H\times G}(s), \Sigma_{uo}^*\sigma\right), & \text{if } \sigma \in \Sigma_o, \\ E_{H\times G}(s), & \text{otherwise.} \end{cases}$$

Notice that the update rules represented in cases (ii) and (iii) generate state estimates using $\mathcal{L}_\Delta = \Sigma_{uo}^*\sigma$ and not $\sigma\Sigma_{uo}^*$. This is an important feature of the state-estimate update rules that results from the assumption of instantaneous communication following the observed event σ, i.e., if communication is instantaneous, then no unobservable continuations can occur following the generation of σ and before the initiation of communication. If the communication channel had a characteristic delay associated with it, then this would alter the state estimates by appending to $\mathcal{L}_\Delta$ locally unobservable traces of maximum length corresponding to the delay. All available information concerning the communication channel should, in general, be accounted for in each $\mathcal{L}_\Delta$.

The use of a simple, inductive argument on the length of traces in $\mathcal{L}(H \times G)$ reveals that the decentralized controllers produce the same state estimates as that of the centralized controller:

Basis of Induction: For trace $t = \varepsilon$, $E_i(\varepsilon) = \Delta_{H\times G}(\{x_{H0,G0}\}, \Sigma_{uo}^*) = E_{H\times G}(\varepsilon)$, so the assertion is true for the base case.

Inductive Hypothesis: Assume the assertion is true for $t = s$.

Inductive Step: The three cases are examined.

Case (i): $Z_\sigma = \emptyset$

$$E_i(s\sigma) = E_i(s) = E_{H\times G}(s) = E_{H\times G}(s\sigma).$$

Case (ii): $Z_\sigma \neq \emptyset$

$$\begin{aligned} E_i(s\sigma) &= \left[\bigcap_{k\in Z_\sigma} \Delta_{H\times G}\left(E_k(s), \Sigma_{uo}^*\sigma\right)\right] \cap \cdot \left[\bigcap_{k\notin Z_\sigma} \Delta_{H\times G}\left(E_k(s), \Sigma_{uo}^*\left(\cap_{j\in Z_\sigma}\Sigma_{o,j} \setminus \Sigma_{o,k}\right)\right)\right] \\ E_i(s\sigma) &= \left[\bigcap_{k\in Z_\sigma} \Delta_{H\times G}\left(E_{H\times G}(s), \Sigma_{uo}^*\sigma\right)\right] \cap \cdot \left[\bigcap_{k\notin Z_\sigma} \Delta_{H\times G}\left(E_{H\times G}(s), \Sigma_{uo}^*\left(\cap_{j\in Z_\sigma}\Sigma_{o,j} \setminus \Sigma_{o,k}\right)\right)\right] \\ &= \Delta_{H\times G}\left(E_{H\times G}(s), \Sigma_{uo}^*\sigma\right) \cap \cdot \left[\bigcap_{k\notin Z_\sigma} \Delta_{H\times G}\left(E_{H\times G}(s), \Sigma_{uo}^*\left(\cap_{j\in Z_\sigma}\Sigma_{o,j} \setminus \Sigma_{o,k}\right)\right)\right] \\ &= \Delta_{H\times G}\left(E_{H\times G}(s), \Sigma_{uo}^*\sigma\right) \\ &= E_{H\times G}(s\sigma). \end{aligned}$$

Case (iii) results by similar manipulations as in Case (ii), and the inductive argument is complete.

Following the trace $s\sigma \in \mathcal{L}(H\times G)$, the control decision that a centralized supervisor generates is based on the set of states $\Delta_{H\times G}(E_{H\times G}(s\sigma), \Sigma_{uo}^*)$ (permissive). However, we have that $E_i(s\sigma) = E_{H\times G}(s\sigma)$ for all $i \in Z$, so each controller in the decentralized system is able to synthesize the centralized control decision under the policy of full communication. From this and the well-known fact of supervisory control theory that $\mathcal{L}(H)$ is achievable by a central supervisor iff $\mathcal{L}(H)$ is controllable with respect to $\mathcal{L}(G)$ and Σ_{uc}, and $\mathcal{L}(H)$ is observable with respect to $\mathcal{L}(G)$, Σ_o and Σ_c, the first assertion of the theorem is true. Furthermore, the estimator structures used here for each controller are finite by the boundedness of $2^{X_{H\times G}}$ leading to a finite information structure. The finite state estimates were the only objects communicated in the above policy, so the second assertion of the theorem holds. □

The proof of Theorem 3.1 utilizes the construction of a finite estimator structure and inference maps that utilize the existence of future communications from other controllers along certain trajectories in the system. For the case of full communication used in the proof, these trajectories are easy to determine and are represented by the $\mathcal{L}_\Delta$ arguments of $\Delta_{H\times G}(\cdot,\cdot)$, i.e., $\mathcal{L}_\Delta$

contains all unobservable continuation traces for which communication is *not* expected. If a locally unobservable continuation trace is not in $\mathcal{L}_\Delta$ then either it cannot occur, it will be followed by a locally observable event, or it will be followed by a communication. Memory of prior communications is captured by the state-update rules for the estimator structure. In retrospect, it is to be expected that full communication will allow the centralized controller's information to be reconstructed by decentralized controllers. However, one may not have expected that this could be done by only communicating state-estimates and not the events themselves.

It will be shown that the fundamental property required to support the result of Theorem 3.1 is the anticipation of communication and its effect on the controllers' information. Not only do these controllers base their "state estimates" on every event they observe and every communication they receive, but they also utilize the fact that they can expect future communications from other controllers if certain events occur. As discussed earlier, this expectation is captured by customizing each $\mathcal{L}_\Delta$ used in the state-update rules, and this is, in general, what makes communication-policy synthesis difficult. The next result provides a characterization of the languages achievable by controllers that *do not* anticipate future communications.

Consider permissive controllers that maintain trajectory, not state, estimates. That is, following a trajectory $t \in \mathcal{T}(Z/G)$, each controller $i \in Z$ has an estimate $\Psi_i(\mathbb{T}_i(t))$ of trajectories that Controller i infers could have occurred given its observations of events and communications received. In the case of a centralized controller with total recall, perfect memory of entire observed event sequence, the centralized estimate for $t \in \mathcal{T}(Z/G)$ and $s = t|_\Sigma$ is [assuming $\mathcal{T}(Z/G)|_\Sigma = \mathcal{L}(Z/G) = \mathcal{L}(H)$]:

$$\begin{aligned}\Psi_{central}(\mathbb{T}_{central}(t)) &= \mathbb{T}_{central}^{-1}(\mathbb{T}_{central}(t)) \cap \mathcal{T}(Z/G) \\ &= \mathbb{P}^{-1}(\mathbb{P}(s)) \cap \mathcal{L}(H)\end{aligned}$$

where the last equality holds because, in the centralized case, there is no communication and the trajectories degenerate to event traces, and $\mathbb{T}_{central}$ can be viewed, to some extent, as the standard natural projection $\mathbb{P}$. Because of its dependence solely on s, with no communication, we may denote $\Psi_{central}(\mathbb{T}_{central}(t))|_\Sigma$ by $\tilde{\Psi}_{central}(s)$ for brevity and convenience. Likewise, we will use the notation $\tilde{\Psi}_i(s)$ to represent $\Psi_i(\mathbb{T}_i(t))|_\Sigma$ when $t|_\Sigma = s$, and the following definition will be used to characterize a particular inference map constraint.

Definition 3.1: Controller i is called *myopic* if Ψ_i does not take into account future communications between the controllers, that is,

$$(\forall t_i \in T_i)\, \Psi_i(t_i)|_\Sigma = \Psi_i(t_i)|_\Sigma \Sigma_{uo,i}^* \cap \mathcal{L}(H). \tag{15}$$

The trace estimates of myopic controllers with *unbounded* memory and communication capacities are characterized in the following lemma.

Lemma 3.1: Let $\tilde{\Psi}_i(s) := \tilde{\Psi}_i(t|_\Sigma) := \Psi_i(\mathbb{T}_i(t))|_\Sigma$, $i = 1, \cdots, n$, be trace estimates generated by myopic controllers that communicate their trace estimates via two-way broadcast following every locally observed event. Then for $s \in \mathcal{L}(H)$

$$\begin{aligned}\tilde{\Psi}_i(s) &= \tilde{\Psi}_{central}(s)\Sigma_{uo,i}^* \cap \mathcal{L}(H) \\ &= \mathbb{P}^{-1}(\mathbb{P}(s))\Sigma_{uo,i}^* \cap \mathcal{L}(H)\end{aligned}$$

that is, the trace estimates of each controller in the decentralized system are equal to the trace estimate that a centralized controller would have, appended with the set of locally unobservable continuation traces.

Proof of Lemma 3.1 (By induction): For $t = \varepsilon$

$$\begin{aligned}&\tilde{\Psi}_{central}(\varepsilon)\Sigma_{uo,i}^* \cap \mathcal{L}(H) \\ &\quad = \left[\mathbb{P}^{-1}(\mathbb{P}(\varepsilon)) \cap \mathcal{L}(H)\right] \Sigma_{uo,i}^* \cap \mathcal{L}(H) \\ &\quad = \Sigma_{uo,i}^* \cap \mathcal{L}(H) \\ &\quad = \tilde{\Psi}_i(\varepsilon)\end{aligned}$$

hence the assertion holds, by construction, for the base case. Assume the assertion holds for $t = s$, and we must show that it holds for $t = s\sigma$. Let $Z_\sigma \subseteq Z$ denote the set of controllers $i \in Z$ for which $\sigma \in \Sigma_{o,i}$. There are three cases: $Z_\sigma = \emptyset$, $\emptyset \subset Z_\sigma \subset Z$, and $Z_\sigma = Z$.

Case (i): $Z_\sigma = \emptyset$. $\tilde{\Psi}_i(s\sigma) = \tilde{\Psi}_i(s) = \tilde{\Psi}_{central}(s)\Sigma_{uo,i}^* \cap \mathcal{L}(H)$. By the assumption that $\Sigma_o = \cup_{i \in Z}\Sigma_{o,i}$, we have $Z_\sigma = \emptyset \Rightarrow \sigma \in \Sigma_{uo}$ and $\tilde{\Psi}_{central}(s\sigma) = \tilde{\Psi}_{central}(s)$; hence,

$$\tilde{\Psi}_i(s\sigma) = \tilde{\Psi}_{central}(s\sigma)\Sigma_{uo,i}^* \cap \mathcal{L}(H).$$

Case (ii): $\emptyset \subset Z_\sigma \subset Z$. The general rule used here for updating trace estimates following communication is given by

$$\tilde{\Psi}_i(s\sigma) = \left[\left[\bigcap_{j \in Z_\sigma} \tilde{\Psi}_j(s)\sigma\right] \cap \left[\bigcap_{j \notin Z_\sigma} \tilde{\Psi}_j(s)\right]\right] \Sigma_{uo,i}^* \cap \mathcal{L}(H). \tag{16}$$

Substituting from the induction hypothesis

$$\begin{aligned}\tilde{\Psi}_i(s\sigma) = \Bigg[&\bigcap_{j \in Z_\sigma} \left[\tilde{\Psi}_{central}(s)\Sigma_{uo,j}^* \cap \mathcal{L}(H)\right]\sigma \cap \\ &\cdot \bigcap_{j \notin Z_\sigma} \tilde{\Psi}_{central}(s)\Sigma_{uo,j}^* \cap \mathcal{L}(H)\Bigg] \Sigma_{uo,i}^* \cap \mathcal{L}(H).\end{aligned} \tag{17}$$

Using the fact that $(\forall j \notin Z_\sigma)\ \sigma \in \Sigma_{uo,j}$

$$\begin{aligned}&\tilde{\Psi}_i(s\sigma) \\ &\quad = \left[\left[\bigcap_{j \in Z} \tilde{\Psi}_{central}(s)\Sigma_{uo,j}^*\sigma\right] \cap \mathcal{L}(H)\sigma \cap \mathcal{L}(H)\right] \\ &\qquad \cdot \Sigma_{uo,i}^* \cap \mathcal{L}(H) \\ &\quad = \left[\tilde{\Psi}_{central}(s)\Sigma_{uo}^*\sigma \cap \mathcal{L}(H)\sigma \cap \mathcal{L}(H)\right] \Sigma_{uo,i}^* \cap \mathcal{L}(H) \\ &\quad = \left[\left[\tilde{\Psi}_{central}(s)\Sigma_{uo}^* \cap \mathcal{L}(H)\right]\sigma \cap \mathcal{L}(H)\right] \Sigma_{uo,i}^* \cap \mathcal{L}(H) \\ &\quad = \left[\tilde{\Psi}_{central}(s)\Sigma_{uo}^* \cap \mathcal{L}(H)\right]\sigma\Sigma_{uo,i}^* \cap \mathcal{L}(H).\end{aligned}$$

Use $\tilde{\Psi}_{central}(s) = \tilde{\Psi}_{central}(s)\Sigma_{uo}^* \cap \mathcal{L}(H)$ to reduce

$$\begin{aligned}\tilde{\Psi}_i(s\sigma) &= \tilde{\Psi}_{central}(s)\sigma\Sigma_{uo,i}^* \cap \mathcal{L}(H) \\ &= \tilde{\Psi}_{central}(s\sigma)\Sigma_{uo,i}^* \cap \mathcal{L}(H) \\ &\qquad \text{(Using Lemma B.1 in Appendix B).}\end{aligned}$$

Case (iii): $Z_\sigma = Z$. All controllers observe σ in this case, and the general rule used here for updating trace estimates following communication is given by

$$\tilde{\Psi}_i(s\sigma) = \left[\bigcap_{j\in Z} \tilde{\Psi}_j(s)\sigma\right] \Sigma^*_{uo,i} \cap \mathcal{L}(H) \tag{18}$$

which reduces as in Case (ii) to

$$\tilde{\Psi}_i(s\sigma) = \tilde{\Psi}_{central}(s\sigma)\Sigma^*_{uo,i} \cap \mathcal{L}(H). \tag{19}$$

For each of the three cases the induction step holds, hence the inductive proof is complete. □

The class of languages achievable using myopic controllers can now be characterized.

Theorem 3.2 (Myopic Ψ_i-Permissive): Let Z be a set of myopic Ψ_i-permissive controllers that maintain trace estimates and communicate their trace estimates via two-way broadcast following every event they observe locally. Then Problem (P) can be solved with these controllers iff:

1. $\mathcal{L}(H)$ is controllable with respect to $\mathcal{L}(G)$ and Σ_{uc};
2. $(\forall s \in \mathcal{L}(H))(\forall \sigma \in \Sigma_c)$:

$$[s\sigma \notin \mathcal{L}(H)] \wedge [s\sigma \in \mathcal{L}(G)] \Rightarrow (\exists i \in Z)\left[\mathbb{P}^{-1}(\mathbb{P}(s))\Sigma^*_{uo,i}\sigma \cap \mathcal{L}(H) = \emptyset\right] \wedge [\sigma \in \Sigma_{c,i}].$$

Proof of Theorem 3.2—Sufficiency: We can construct supervisors in two ways.

(a) having the controllers communicate their trace estimates to every other controller following every observed event resulting in trace estimates derived in Lemma 3.1;

(b) using trace estimates from (a) we assign the permissive disable maps as follows: $(\forall i \in Z)$

$$\gamma_i(\mathbb{T}_i(t)) = \left\{\sigma \in \Sigma_{c,i} | \tilde{\Psi}_i(t|_\Sigma)\sigma \cap \mathcal{L}(H) = \emptyset\right\}. \tag{20}$$

This form of disable map is analogous to the "pass the buck" construction used in [9] where the "buck" that is passed is the decision to *disable* an event. These control maps represent the Ψ_i-permissive property of the controllers: a controller only disables an event if there is no ambiguity to that controller that the event should be disabled over all traces in $\tilde{\Psi}_i(t|_\Sigma)$.[7]

Thus if $s\sigma \in \mathcal{L}(H)$, then no controller disables σ, and if $s\sigma \notin \mathcal{L}(H)$, $\sigma \in \Sigma_c$ and $s\sigma \in \mathcal{L}(G)$, then the constructions in (b) and the second condition of the theorem guarantee at least one controller will disable σ. If $s \in \mathcal{L}(H)$, $s\sigma \in \mathcal{L}(G)$ and $\sigma \in \Sigma_{uc}$ then the first condition implies $s\sigma \in \mathcal{L}(H)$ (controllability), and no controller disables σ; hence, Problem (P) is solvable using these controllers.

Necessity: If Problem (P) is solvable using Z, then it is solvable in the centralized case. Therefore, we have the controllability requirement of the first condition. For the second condition, we will use contradiction. Assume Problem (P) is solvable using the myopic controllers in Z with permissive control maps, but the second condition does not hold, then $\exists s \in \mathcal{L}(H)$ and $\exists \sigma \in \Sigma_c$ such that

$$[s\sigma \notin \mathcal{L}(H)] \wedge [s\sigma \in \mathcal{L}(G)] \wedge \left[(\forall i \in Z)\left[\mathbb{P}^{-1}(\mathbb{P}(s))\Sigma^*_{uo,i}\sigma \cap \mathcal{L}(H) \neq \emptyset\right] \vee [\sigma \notin \Sigma_{c,i}]\right].$$

Then for all $i \in Z$ such that $\sigma \in \Sigma_{c,i}$ we have $\mathbb{P}^{-1}(\mathbb{P}(s))\Sigma^*_{uo,i}\sigma \cap \mathcal{L}(H) \neq \emptyset$. This implies that for each such i there are two traces, s, $s' \in \tilde{\Psi}_i(s)$, for which σ must be disabled following s but enabled following s'; hence, each controller must enable σ by the permissive nature of the control maps. The enabling of σ allows the violation of the language $\mathcal{L}(H)$, a contradiction, so the second condition must hold. □

The proof of Theorem 3.2 proceeds by showing that trace estimates generated as in Lemma 3.1 are sufficiently refined such that permissive control policies can produce any language possessing the property shown in the theorem. The proof also shows that if the desired language does not have the required property, the trace estimates generated by myopic controllers with total recall and full communication are not refined enough for permissive control maps to produce the correct actions required to synthesize the desired behavior.

It is interesting to compare the class of languages achievable by myopic controllers to the class of languages achievable by controllers that anticipate future communications. It can be shown (see Appendix A) that the observability condition in Theorem 3.1 can be rewritten as $(\forall s \in \mathcal{L}(H))(\forall \sigma \in \Sigma_c)$,

$$[s\sigma \notin \mathcal{L}(H)] \wedge [s\sigma \in \mathcal{L}(G)] \Rightarrow (\exists i \in Z)\left[\mathbb{P}^{-1}(\mathbb{P}(s))\sigma \cap \mathcal{L}(H) = \emptyset\right] \wedge [\sigma \in \Sigma_{c,i}].$$

Comparing this with the second condition in Theorem 3.2 reveals that permissive myopic controllers with arbitrarily large memory and communication resources (i.e., maintaining and communicating arbitrarily large trace estimates) are outperformed by permissive controllers that maintain and communicate finite state estimates, *but anticipate future communications*.

We can also use the results in Appendix A to compare permissive myopic controllers to controllers that do not communicate at all. The class of languages achievable by permissive controllers with no communication is characterized by the coobservability property [9] which, disregarding marking, can be written as

$$(\forall s \in \mathcal{L}(H))(\forall \sigma \in \Sigma_c): [s\sigma \notin \mathcal{L}(H)] \wedge [s\sigma \in \mathcal{L}(G)] \Rightarrow (\exists i \in Z)\left[\mathbb{P}^{-1}_{\Sigma_{o,i}}(\mathbb{P}_{\Sigma_{o,i}}(s))\sigma \cap \mathcal{L}(H) = \emptyset\right] \wedge [\sigma \in \Sigma_{c,i}]$$

from which it is apparent that permissive communicating myopic controllers achieve a strictly larger class of languages than coobservable languages. We summarize this comparison with the following corollary.

Corollary 3.1: Let $\mathcal{L}(G)$ be a specified language, and $\Sigma_{o,i}$, $\Sigma_{c,i}$ be given for $i = 1, \cdots, n$. Denote by $\mathcal{C}_{\mathcal{L},obs}$ the class of languages observable with respect to $\mathcal{L}(G)$, $\cup_{i=1\cdots n}\Sigma_{o,i}$, $\cup_{i=1\cdots n}\Sigma_{c,i}$. Denote by $\mathcal{C}_{\mathcal{L},coobs}$ the class of languages coobservable with respect to

[7] There are other "bucks" that can be passed. For example, work is currently being done [22] on *anti-permissive* control maps where each controller attempts to *disable* an event if there is any ambiguity to that controller as to whether the event should be enabled or disabled. The fusion rule for these anti-permissive controllers is the intersection of the disabled events (versus the union of disabled events for permissive controllers). In the centralized case, permissive and anti-permissive control maps are equivalent (there are no "bucks" to be passed).

$\mathcal{L}(G), \Sigma_{o,i}, \Sigma_{c,i}, i = 1, \cdots, n$. Denote by $\mathcal{C}_{\mathcal{L},fsnm}$ the class of languages achievable with permissive finite-state estimate nonmyopic controllers (See Proof of Theorem 3.1) with respect to $\mathcal{L}(G), \Sigma_{o,i}, \Sigma_{c,i}, i = 1, \cdots, n$. Finally, denote by $\mathcal{C}_{\mathcal{L},myopic}$ the class of languages satisfying the properties given in Theorem 3.2 with respect to $\mathcal{L}(G), \Sigma_{o,i}, \Sigma_{c,i}, i = 1, \cdots, n$, i.e., languages achievable by permissive myopic controllers with total recall. Then $\mathcal{C}_{\mathcal{L},coobs} \subseteq \mathcal{C}_{\mathcal{L},myopic} \subseteq \mathcal{C}_{\mathcal{L},fsnm} = \mathcal{C}_{\mathcal{L},obs}$.

From a communication-policy synthesis viewpoint, the results of this section have at least one major implication. This implication is exactly analogous to the case in stochastic control where there is no separation between estimation and control. Here, for Problem (P), the lack of separation is between estimation and communication. In order to determine when communication is required (synthesis of the communication policy), the estimation policy needs to be known. For controllers that anticipate future communications, however, the synthesis of the estimation policy depends on knowledge of the communication policy. Thus, in the general case, the communication policies and inference maps must be synthesized simultaneously.

IV. Optimality of Communication Policies

When designing the communication policy, it may be desirable to have the controllers communicate as little as possible while collectively guaranteeing that a desired behavior is synthesized. It is difficult to agree on definitions of "optimality" or "minimality" for discrete-event system policies that are useful for all scenarios. The most significant reason for this difficulty is the lack of measure on the system states or events. [8] Optimality in discrete-event systems generally involves the coarseness of a partition [30] or the supremality/infimality of a particular set. For example, supremal controllable languages contain as many desired traces as possible. Thus, one might wish to attempt to characterize an optimal communication policy using similar concepts. Most of these characterizations will probably have the tendency to be good for some systems while bad for others due to the existence of transition loops that indicate that a communication may occur an arbitrarily large number of times regardless of the partition refinement or any partial ordering of the set resulting from the optimization process. There are, however, certain conditions that we would like any definition of optimality to possess, and it is therefore useful to attempt to capture the intuitive aspects of these conditions with a mathematical characterization.

Let $\mathcal{C}_\Theta^P$ be the set of all communication policies that will allow Problem (P) to be solved, i.e., for each $\theta \in \mathcal{C}_\Theta^P$ there is a corresponding control policy γ such that (γ, θ) solves Problem (P). Each $\theta \in \mathcal{C}_\Theta^P$ has a corresponding language $\mathcal{L}_{com}^\theta$ that represents the set of all traces $s \in \mathcal{L}(G)$ for which a communication occurs immediately following s. $\mathcal{L}_{com}^\theta$ is generally not a prefix-closed language unless there is constant communication.

One condition we would like for any optimal communication policy θ^* to possess is that no trace in $\mathcal{L}_{com}^{\theta^*}$ can be removed without some other trace (or set of traces) being added. This leads us to the first optimality condition:

(C1) $\forall \theta \in \mathcal{C}_\Theta^P \setminus \{\theta^*\}$:

$$\mathcal{L}_{com}^\theta \not\subset \mathcal{L}_{com}^{\theta^*}.$$

There appears to be nothing "wrong" about only requiring C1 for optimality. Although it does not indicate structures of policies that we should look for or how to search the θ-space, it is consistent with the common usage of language inclusion for optimality in logical-DES control theory.

We may also want additional structure in optimal communication policies. For example, we may want optimal policies to "postpone communication for as long as possible." In some sense, the idea behind this notion of optimality is identical to that used for the supremal controllable sublanguage. Optimal control policies postpone the disablement of a set of events for as long as possible. Of course, the lack of a measure on the events or states themselves makes even this notion of optimality open to debate. There are many ways to define the idea of postponing communication in DES. We give two here, where the first condition, C2, is actually a special case of the second, C3:

(C2) $(\forall \theta \in \mathcal{C}_\Theta^P \setminus \{\theta^*\})(\forall t \in \mathcal{L}_{com}^{\theta^*})$:

$$\mathcal{L}_{com}^\theta \not\subset \left[\mathcal{L}_{com}^{\theta^*} \setminus \{t\}\right] \cup \{t\}\Sigma^+$$

where it is standard to define $\Sigma^+ = \Sigma^* \setminus \{\varepsilon\}$. Condition C2 indicates that no single trace in $\mathcal{L}_{com}^{\theta^*}$ can be lengthened (hence postponing communication longer). There may, however, be communication structures satisfying C2 that would allow, for example, two traces in $\mathcal{L}_{com}^{\theta^*}$ to be simultaneously lengthened; hence, we have the following condition.

(C3) $(\forall \theta \in \mathcal{C}_\Theta^P \setminus \{\theta^*\})(\forall \mathcal{L}_{pp} \subseteq \mathcal{L}_{com}^{\theta^*})$:

$$\mathcal{L}_{com}^\theta \not\subset \left[\mathcal{L}_{com}^{\theta^*} \setminus \mathcal{L}_{pp}\right] \cup \mathcal{L}_{pp}\Sigma^+.$$

This condition says that communication cannot be postponed along any subset of traces in $\mathcal{L}_{com}^{\theta^*}$.[9] Evidently, if a communication policy is C3-optimal, then it is also optimal with regard to C1 ($\mathcal{L}_{pp} = \emptyset$) and C2 ($|\mathcal{L}_{pp}| = 1$). There are, of course, many conditions that a designer could specify for optimality However, it is not our intention to exhaust the possibilities here, and in the follwoing we will restrict attention to C1–C3.

V. Finite-State Estimate Myopic Controller Information Structure and Synthesis Procedure

In this section, we present an example of how the general information structure $\mathcal{I}$ of Section II can be constrained *a priori* as a means of side-stepping the difficulties associated with exactly optimal nonseparable solutions to Problem (P*). Instead of searching for an exact solution to the original problem, we will search for optimal solutions within a constrained class of controllers. That is, the problem is constrained to yield communication policies that are more amenable to synthesis.

A. Finite-State Estimates for Myopic Controllers

The specific decentralized supervisory control problem investigated here is that of having two controllers with a constrained

[8] Work has been done in [29] where this type of measure is added to the system model to derive an optimal centralized control for discrete-event systems.

[9] The idea of postponing communication "for as long as possible" seems intuitive. However, this may have significantly undesirable effects on robustness—an issue yet to be investigated.

information structure. However, the discussion naturally generalizes to n controllers. The information structure is constrained such that the inference maps satisfy (15). Furthermore, the inference maps of these controllers will be based on a *finite* structure, making the solution physically implementable. Additional assumptions used in this section are as follows.

A.11: Controllers maintain and communicate estimates of the state of the plant.

A.12: The inference maps are based on a finite structure as described below.

To construct the controller inference maps, we will first define a finite estimator structure E which is a finite-state machine with augmented state information. For the case of two controllers, an estimator-structure state is a quadruple

$$(\sigma, E_1, E_2, x)$$

where $\sigma \in \Sigma$, $E_1 \in 2^{X_{H\times G}}$, $E_2 \in 2^{X_{H\times G}}$, and $x \in X_{H\times G}$. E_1 and E_2 are the finite-state estimates of Controllers 1 and 2, respectively. Two estimator-structure states (σ, E_1, E_2, x) and $(\tilde{\sigma}, \tilde{E}_1, \tilde{E}_2, \tilde{x})$ will be considered equivalent if $\sigma = \tilde{\sigma}$, $E_1 = \tilde{E}_1, E_2 = \tilde{E}_2$, and $x = \tilde{x}$. Denote by $X_E \subset \Sigma \times 2^{X_{H\times G}} \times 2^{X_{H\times G}} \times X_{H\times G}$ the set of all (σ, E_1, E_2, x)-tuples for which $x \in E_1$ and $x \in E_2$. The initial state of the estimator structure is

$$x_{E0} = \left(\epsilon, \Delta_{H\times G}\left(\{x_{H0,G0}\}, \Sigma^*_{uo,1}\right) \Delta_{H\times G}\left(\{x_{H0,G0}\}, \Sigma^*_{uo,2}\right), x_{H0,G0}\right).$$

The transition function of the estimator structure is defined from the transitions in $\delta_{H\times G}$ as follows:

$$\delta_E\left((\sigma, E_1, E_2, x), \sigma'\right) = (\sigma', E_1', E_2', x') \tag{21}$$

where

$$\begin{aligned} E_1' &= \Delta_{H\times G}\left(E_1, \Sigma^*_{uo,1}\mathbb{P}_1(\sigma')\right) \\ E_2' &= \Delta_{H\times G}\left(E_2, \Sigma^*_{uo,2}\mathbb{P}_2(\sigma')\right) \\ x' &= \delta_{H\times G}(x, \sigma'). \end{aligned}$$

As stated in Assumption A.11, the controllers are constrained *a priori* to communicate their respective estimates E_i. The update rule for state estimates following the assumed two-way communication (Assumption A.10) is determined by the operator com in (22):

$$com(\sigma, E_1', E_2', x) = (\sigma, E_1'', E_2'', x) \tag{22}$$

where

$$E_i'' = \begin{cases} E_1' \cap E_2', \\ \quad \text{if } \sigma \in \Sigma_{o,1} \cap \Sigma_{o,2}, \\ E_1' \cap \Delta_{H\times G}\left(E_2', \Sigma^*_{uo,2}\left(\Sigma_{o,1} \setminus \Sigma_{o,2}\right)\right), \\ \quad \text{if } \sigma \in \Sigma_{o,1} \setminus \Sigma_{o,2}, \\ \Delta_{H\times G}\left(E_1', \Sigma^*_{uo,1}\left(\Sigma_{o,2} \setminus \Sigma_{o,1}\right)\right) \cap E_2', \\ \quad \text{if } \sigma \in \Sigma_{o,2} \setminus \Sigma_{o,1}, \\ E_i', \\ \text{otherwise.} \end{cases} \tag{23}$$

The com operator of (22) can be interpreted in the following way.

(i) If both controllers observe an event, then the new information or "innovation" derived by the communication is simply the intersection of both controller state estimates.

(ii) If Controller A does not observe an event but receives a communication from Controller B, then Controller A "knows" that Controller B observed an event that was not observable by Controller A, and this "knowledge" is incorporated into the state-update rule. Note that Controller B, as in the Proof of Theorem 3.1, does not explicitly communicate which event was observed.

(iii) The fourth implication of (22) is for notational consistency with the fact that controllers communicate following observable events, and it allows com to be completely defined.

The complete state-estimate update rule for individual controllers is as follows.

1. following an event use (21) to derive E_i';
2. following a two-way communication use (22) to derive E_i''.

Based on the finite estimator structure defined above, it is now possible to specify the inference maps for the controllers and thus complete the specification of the form of this myopic information structure. Let $(\sigma, E_1(s\sigma), E_2(s\sigma), x(s\sigma))$ be the estimator-structure state reached following $t \in \mathcal{T}(Z/G)$ where $s\sigma = t|_\Sigma$, and let $t_i = \mathbb{T}_i(t)$ be the trajectory observed by Controller i along t. The inference map Ψ_i that will be used for this section is

$$\Psi_i(t_i) = \{t' \in \mathcal{T}(Z/G) | \delta_{H\times G}(x_{H0,GO}, t'|_\Sigma) \in \Delta_{H\times G}\left(E_i(s\sigma), \Sigma^*_{uo,i}\right)\}. \tag{24}$$

The myopic nature of this type of state-estimate update is evidenced by the use of $\Sigma^*_{uo,i}$ in the $\mathcal{L}_\Delta$ language arguments of the $\Delta_{H\times G}$ operation. Controllers that anticipate future communications would have specific traces removed from the individual $\mathcal{L}_\Delta$ languages at each state in the estimator structure. The traces removed from $\mathcal{L}_\Delta$ would correspond to those indicated by the anticipation of future communications. Clearly, the com operator described in (22) is not the only possible choice for indicating how communication affects state-estimates.

The inference map of (24) is not only myopic as described above, but, due to its dependence on finite state estimates, it is also *forgetful* in the sense that the trajectory t_i is not identified (the set of trajectories leading to the same state estimate in the structure is identified). Certainly, more complex inference maps can be constructed, e.g., constructing the E_i to be nonmyopic, basing the state-update rules on some finite-length "most recent" record of t_i, more complex com operator,[10] etc. However, it is not our intention to exhaust those possibilities here.

B. Control and Communication

The control policy $\gamma = \{\gamma_1, \gamma_2\}$ presented here is based on the permissive "pass the buck" constructs in [9], that is, following trajectory t_i. Controller i is allowed to disable an event $\sigma_c \in \Sigma_{c,i}$ only if the event should be disabled or does not exist in $\mathcal{L}(G)$ following every trace in $\Psi_i(t_i)$. Formally, for $t_i \in T_i$:

$$\gamma_i\left(\Psi_i(t_i)\right) = \{\sigma_c \in \Sigma_{c,i} | \forall s \in \Psi_i(t_i)|_\Sigma,$$

[10] In general, com should be based on the global set of trajectories, $\mathcal{T}(Z/G)$, to generate maximal information sets and not just the "local" information available at each controller as is done in (22); however, synthesis would then suffer from a lack of separation as discussed in Section III.

$$[s\sigma_c \in \mathcal{L}(G) \Rightarrow s\sigma_c \notin \mathcal{L}(H \times G)]\}. \quad (25)$$

To begin the synthesis of a communication policy, it must be determined *why* communication is needed. To formalize this, the notion of a *conflict state* is introduced.

Definition 5.1: Let E be the estimator structure for a two-controller system as described in Section V-A. A conflict state of estimator structure E is a state (σ, E_1, E_2, x) for which there exists $\sigma_c \in \Sigma_c$ that must be disabled at state $x \in X_{H\times G}$ (for the sake of legality) and $\forall i \in Z$ for which $\sigma_c \in \Sigma_{c,i}$, $\exists \tilde{x} \in \Delta_{H\times G}(E_i, \Sigma^*_{uo,i})$ such that $\delta_{H\times G}(\tilde{x}, \sigma_c)$ is defined (enabled).

The implication of Definition 5.1 is that a conflict state is one in which an event must be disabled in the global system. However, none of the controllers has enough information to determine that the event must be disabled, therefore all of the controllers enable the event by default. The avoidance of conflict states is the motivation for communication between controllers. Note that if $\mathcal{L}(H)$ is coobservable with respect to $\mathcal{L}(G)$, all $\Sigma_{o,i}$, and all $\Sigma_{c,i}$, then E has no conflict states.

Let X_{conf} be the set of *all* conflict states in X_E, and define the subset of estimator structure states X^0_Ω as:

$$X^0_\Omega = \{(\sigma, E_1, E_2, x) \in X_E | com(\sigma, E_1, E_2, x) \in X_{conf}\}.$$

Notice that if $com(\sigma, E_1, E_2, x)$ is a conflict state, then (σ, E_1, E_2, x) is also a conflict state. Intuitively, this makes sense because there is more information available following a communication than following no communication. By Assumption A.3, instantaneous communication upon reaching a potential conflict state is an allowable means of avoiding a true conflict state, i.e., communication can be used to attempt to immediately resolve a potential conflict. The set X^0_Ω is the set of all states for which instantaneous communication fails to resolve potential conflicts. Define the sequence $\{X^k_\Omega\}$ such that

$$X^k_\Omega = \{(\sigma, E_1, E_2, x) \in X_E | (\sigma, E_1, E_2, x) \in X^{k-1}_\Omega, \text{ or } \exists \sigma' \in \Sigma \text{ such that } \delta_E(com(\sigma, E_1, E_2, x), \sigma') \in X^{k-1}_\Omega\}.$$

The set X^k_Ω is first constructed by including all estimator states that are in X^{k-1}_Ω. Second, all estimator-structure states are included in X^k_Ω for which communication at those states cannot guarantee that the system will not transition, in one step, to a state in X^{k-1}_Ω. The sequence $\{X^k_\Omega\}$ converges by the monotonicity of the cardinality of X^k_Ω and the boundedness of X_E Furthermore, the limiting set is unique. Denote the limit of the sequence by X_Ω. This set characterizes all states of the estimator structure (not necessarily reachable from x_{E0}) for which there exists a path of states that leads to a conflict state and the controllers communicate at all possible moments along this path. *Hence, if the system enters X_Ω, there exists a continuation trace which violates the desired behavior regardless of the amount of communication along that trace.*

For a given automaton $A = (X_A, \Sigma_A, \delta_A, x_{A0})$, define the *preimage* of a set of states $X \in X_A$ by

$$Pre_A(X) = \{x \in X_A | \exists \sigma \in \Sigma_A \text{ such that } \delta_A(x, \sigma) \in X\}.$$

We define the *boundary* of X_Ω, denoted $Bdry(X_\Omega)$, as the set of states (σ, E_1, E_2, x) which have a transition into X_Ω but for which $com(\sigma, E_1, E_2, x)$ does not $(Bdry(X_\Omega) = Pre_E(X_\Omega) \setminus X_\Omega)$. *These states represent the last possible moment for which communication of state estimates as described above can be utilized to avoid all conflict states in X^0_Ω.* We will refer to the procedure just described for constructing X_Ω as the "X_Ω-Procedure" in the sequel.

Note that, due to fourth implicant of (22), every estimator-structure state in $Bdry(X_\Omega)$ has the property

$$(\sigma, E_1, E_2, x) \in Bdry(X_\Omega) \Rightarrow \sigma \in \Sigma_o.$$

This property is important because we would like at least one controller to be able to initiate communication on the boundary of X_Ω. Because $\sigma \in \Sigma_o$ is guaranteed on the boundary of X_Ω, by Assumption A.5 we can allow the controllers to "wait" until the boundary of X_Ω is reached. Then, at the boundary, at least one controller will observe the transition to the boundary and communicating state estimates will guarantee that X_Ω will be avoided. The definition of X_Ω can be used to show that $x_{E0} \notin X_\Omega$ is necessary and sufficient for $\mathcal{L}(H)$ to be exactly achievable under the finite-state estimate myopic scheme described. This test for $x_{E0} \notin X_\Omega$ provides a very crude characterization of the set of languages achievable by these finite-state estimate myopic controllers.

We now give two rules for constructing the communication maps based on $Bdry(X_\Omega)$. Comparison of these two rules will help elucidate the difficulties associated with Problem (P*) even with the use of these myopic controllers.

Rule (R1) For $t \in \mathcal{T}(Z/G)$ with $s = t|_\Sigma$, and $\sigma \in \Sigma_o$:

$$\theta_1(\Psi_1(\mathbb{T}_1(t))\sigma) = \begin{cases} E_1(s\sigma), & \text{if } \sigma \in \Sigma_{o,1} \text{ and } \exists E_2, x \text{ such that} \\ & (\sigma, E_1(s\sigma), E_2, x) \in Bdry(X_\Omega), \\ & \text{or } (\sigma, E_1(s\sigma), E_2, x) \in X_{conf} \setminus X^0_\Omega \\ & \text{or a communication is received from} \\ & \text{other controller (A.10);} \\ \emptyset & \text{otherwise} \end{cases} \quad (26)$$

$$\theta_2(\Psi_2(\mathbb{T}_2(t))\sigma) = \begin{cases} E_2(s\sigma), & \text{if } \sigma \in \Sigma_{o,2} \text{ and } \exists E_1, x \text{ such that} \\ & (\sigma, E_1, E_2(s\sigma), x) \in Bdry(X_\Omega), \\ & \text{or } (\sigma, E_1, E_2(s\sigma), x) \in X_{conf} \setminus X^0_\Omega \\ & \text{or a communication is received from} \\ & \text{other controller (A.10)} \\ \emptyset, & \text{otherwise.} \end{cases} \quad (27)$$

Note that Rule R1 utilizes the two-way broadcast assumption mentioned in Section III. The notation "$\Psi_i(\cdot)\sigma$" in the argument of each θ_i means that given a controller's previous estimate $\Psi_i(\cdot)$ and the new observation σ, the controller decides instantly whether to initiate a communication or not. Note also that, technically, θ_i must be defined inductively starting with the empty trace. This is because of the use of trajectories t in the argument of θ_i, i.e., a trajectory of length k can only be defined if each θ_i has been defined for all prefixes of that trajectory with lengths less than k. In general, this would present

a problem since $\Psi_i(\mathbb{T}_i(t))$ will usually depend on trajectories which are continuations of t which cannot be known without first knowing θ_i along those continuations. The use of myopic controllers solves this problem by removing the dependence on continuations of t at the cost, of course, of limiting controller performance. It can be seen that Rule R1 generalizes easily to n controllers and provides an easy method of generating communication policies based on the finite-estimator structure; furthermore, given the set X_Ω there is a unique communication policy associated with Rule R1. The uniqueness of the communication policy results from the use of only the unique sets $Bdry(X_\Omega)$ and X_{conf} in the decision to communicate or not.

Now, consider the following rule.

Rule (R1*) For $t \in \mathcal{T}(Z/G)$ with $s = t|_\Sigma$, and $\sigma \in \Sigma_o$

$$\theta_1(\Psi_1(\mathbb{T}_1(t))\sigma) = \begin{cases} E_1(s\sigma), & \text{if } \sigma \in \Sigma_{o,1} \text{ and } [\theta_1(\Psi_1(\mathbb{T}_1(t))\sigma) = \emptyset] \\ & \Rightarrow \exists E_2, x \text{ such that } [(\sigma, E_1(s\sigma), E_2, x) \\ & \in \Delta_E(\{x_{E0}\}, \Psi_1(\mathbb{T}_1(t))\sigma) \cap Bdry(X_\Omega)] \\ & \vee [(\sigma, E_1(s\sigma), E_2, x) \in \Delta_E \\ & (\{x_{E0}\}, \Psi_1(\mathbb{T}_1(t))\sigma) \cap X_{conf} \setminus X_\Omega^0] \\ & \text{or a communication is received from} \\ & \text{other controller (A.10)} \\ \emptyset, & \text{otherwise} \end{cases} \tag{28}$$

$$\theta_2(\Psi_2(\mathbb{T}_2(t))\sigma) = \begin{cases} E_2(s\sigma), & \text{if } \sigma \in \Sigma_{o,2} \text{ and } [\theta_2(\Psi_2(\mathbb{T}_2(t))\sigma) = \emptyset] \\ & \Rightarrow \exists E_1, x \text{ such that } [(\sigma, E_1, E_2(s\sigma), x) \\ & \in \Delta_E(\{x_{E0}\}, \Psi_2(\mathbb{T}_2(t))\sigma) \cap Bdry(X_\Omega)] \\ & \vee [(\sigma, E_1, E_2(s\sigma), x) \in \Delta_E \\ & (\{x_{E0}\}, \Psi_2(\mathbb{T}_2(t))\sigma) \cap X_{conf} \setminus X_\Omega^0] \\ & \text{or a communication is received from} \\ & \text{other controller (A.10)} \\ \emptyset, & \text{otherwise} \end{cases} \tag{29}$$

The difference between Rules R1 and R1* is that Rule R1* checks the reachability of states that may be either in $Bdry(X_\Omega)$ or in X_{conf} before determining that a communication is required. Rule R1 does not perform this reachability test, and so it may produce communications based on indistinguishability of the current state from a state in X_E that is not even reachable. Clearly, Rule R1* should outperform Rule R1 in some sense. However, this performance comes at a price. Reachability in the estimator structure cannot be determined *a priori* to the synthesis of the θ_i maps, and reachability may change during the synthesis of θ_i maps. Thus, for Rule R1* in general, an iterative procedure must be used to synthesize each θ_i, and the resulting maps are not necessarily unique despite the uniqueness of X_Ω. It is unclear whether such iterative procedures necessarily terminate. However, it is a relatively simple matter to check whether or not a proposed communication policy correctly implements Rule R1*. Despite the difficulties associated with synthesizing communication policies that are consistent with Rule R1*, the performance of Rule R1* is discussed in more detail in Section V-D.

This completes the construction of the decentralized supervisory control system with communicating finite-state myopic controllers. An illustrative example is given in the following subsection.

C. Example

The example presented here utilizes the myopic inference maps of (24) based on the finite estimator structure E to solve the decentralized supervisory control problem with communication. The desired behavior of the controlled system is depicted in Fig. 2 where it is assumed that the event γ_1 is possible in the plant at state 7, but is disabled in the desired behavior. It is assumed that $\Sigma_{o,1} = \{\alpha_1, \beta_1, \gamma_1\}, \Sigma_{c,1} = \{\gamma_1\}$, $\Sigma_{o,2} = \Sigma_{c,2} = \{\alpha_2\}$. The key feature of this example is the interleaving of α_1 and α_2. To begin the solution procedure, the finite estimator structure is constructed without communication. This facilitates the identification of reachable conflict states, the avoidance of which motivates communication. The estimator structure constructed using (21) is shown in Fig. 3. The states of the estimator structure are named **a**, **b**, **c**, **d**, **e**, **f**, **g**, and **h** for reference. Examining the estimator structure constructed with no communication between the controllers, it becomes apparent that state **g** is a conflict state. The conflict arises because the event γ_1 must be disabled at the system's state 7 (estimator-structure state **g**) However, Controller 1 is uncertain as to whether the system is at state 6 or state 7. State 6 requires that γ_1 be enabled. Hence, Controller 1 enables γ_1, by default, at estimator-structure state **g**.

In accordance with the X_Ω-Procedure, the set X_Ω^0 of all conflict states that cannot be immediately resolved using communication needs to be determined. Because we are only interested in reachable conflict states, our attention is focused on state **g** in the estimator-structure. Communication at state **g** follows the event β_1 which is observable only to Controller 1 Therefore, the second implication of (22) applies. Controller 2, upon receiving a communication from Controller 1, recognizes that an observable event has occurred (by Assumption A.10) that was not observed by itself, and Controller 2 utilizes the second implication of the com operator to determine its new state estimate

$$\begin{aligned} E_2'' &= E_1' \cap \Delta_{H\times G}\left(E_2', \Sigma_{uo,2}^*(\Sigma_{o,1} \setminus \Sigma_{o,2})\right) \\ &= E_1' \cap \Delta_{H\times G}\left(\{2, 4, 5, 6, 7, 8\}, \Sigma_{uo,2}^*\{\alpha_1, \beta_1, \gamma_1\}\right) \\ &= E_1' \cap \{4, 6, 7, 8\} \\ &= \{6, 7\}. \end{aligned}$$

Controller 1 carries out a similar calculation arriving at $E_1'' = \{6, 7\}$. Hence, Controller 1 still enables event γ_1. Thus, estimator-structure state **g** is a member of X_Ω^0. Given that the conflict at state **g** cannot be immediately resolved using communication, the preimage of **g** is examined for inclusion in the set X_Ω^1. The state estimates for the controllers following communication at state **e**, found in a similar manner as above, are $E_1'' = \{5\}$ and $E_2'' = \{5\}$; hence, at state **e**

$$com(\alpha_2, \{3, 4, 5\}, \{2, 5\}, 5) = (\alpha_2, \{5\}, \{5\}, 5)$$

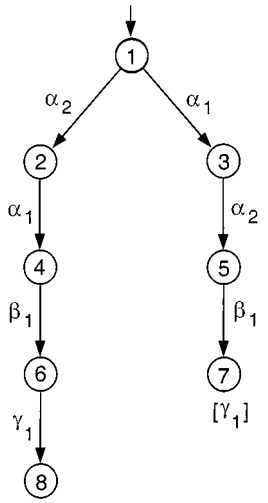

Fig. 2. Desired behavior to be decentrally implemented.

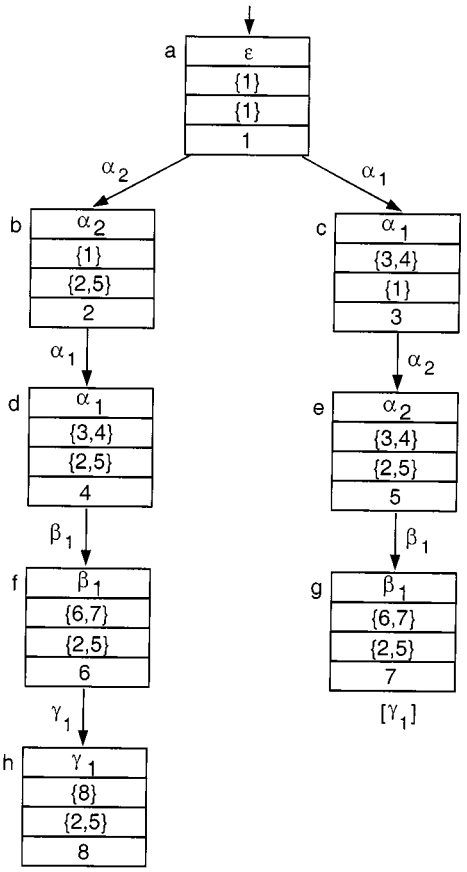

Fig. 3. Estimator structure without communication.

which, following the event β_1, leads to the new state $(\beta_1, \{7\}, \{5, 7\}, 7)$. The state $(\beta_1, \{7\}, \{5, 7\}, 7)$ is not a conflict state. Hence estimator-structure state **e** is the last point at which the controllers can communicate to avoid conflict states (namely, conflict state **g**).

Controller 2 needs to initiate the communication at state **e** following the observation of α_2 However, state **b** is indistinguishable from state **e** (modulo Controller 2's inference map) indicating that Controller 2 must communicate at state **b** also. The estimator structure determined by this informal procedure represents a communication policy that is consistent with Rule R1* and is shown in Fig. 4 where doubled state boxes indicate the occurrence of communication (left = before communication, right = after communication). There are no conflict states in Fig. 4, and the set of system trajectories represented by Fig. 4 is given in Table I.

D. Myopic Controllers, Optimality and Non-Uniqueness

Here, we show that any communication policy consistent with Rule R1* for the finite-state myopic controllers is, in fact, optimal over the class of described finite-state estimate myopic controllers with respect to Condition C1.

To facilitate the proof and to relate informational consistency to state estimates, define the following set:

$$E^{\theta}_{i,com} = \big\{(\sigma, E_i(s\sigma)) \in \Sigma_{o,i} \times 2^{X_{H\times G}} \big| s\sigma \in \mathcal{L}(Z/G), \exists E_1, \cdots, E_{i-1}, E_{i+1} \cdots, E_n, x \text{ such that Controller } i \text{ initiates communication at } (\sigma, E_1, \cdots, E_i(s\sigma), \cdots, E_n, x) \text{ under policy } \theta.\big\}$$

So, $E^{\theta}_{i,com}$ is the set of event/state-estimate pairs where Controller i initiates communication under policy θ.

Lemma 5.1: Let $\mathcal{L}^{\theta}_{com} \subset \mathcal{L}^{\theta^*}_{com}$ where $s\sigma \in \mathcal{L}^{\theta^*}_{com} \setminus \mathcal{L}^{\theta}_{com}$, then $(\forall s'\sigma \in \mathcal{L}(Z/G))(\forall i \in Z)$

$$E_i(s\sigma) = E_i(s'\sigma) \Rightarrow (\sigma, E_i(s'\sigma)) \notin E^{\theta}_{i,com} \quad (30)$$

$$\Rightarrow s'\sigma \notin \mathcal{L}^{\theta}_{com}. \quad (31)$$

Proof of Lemma 5.1: Implication (30) follows by contradiction: if $\exists i \in Z$ such that $E_i(s\sigma) = E_i(s'\sigma)$ and $(\sigma, E_i(s'\sigma)) \in E^{\theta}_{i,com}$, then $s\sigma \in \mathcal{L}^{\theta}_{com}$ which contradicts $s\sigma \in \mathcal{L}^{\theta^*}_{com} \setminus \mathcal{L}^{\theta}_{com}$. Implication (31) follows from the definition of $E^{\theta}_{i,com}$. □

Lemma 5.1 simply states that if $s\sigma$ does not lead to a communication and $s'\sigma$ does lead to a communication, then $s'\sigma$ does not "look like" $s\sigma$ to any controller that initiates communication following $s'\sigma$. Given this lemma, we have the following optimality property of communication policies satisfying Rule R1*.

Theorem 5.1: Let $\theta^* = \{\theta^*_1, \theta^*_2, \cdots, \theta^*_n\}$ be consistent with the X_Ω-Procedure and Rule R1*; then θ^* is C1-optimal over the class of myopic controllers that maintain and communicate finite state estimates as described in (21) and (22).

Proof of Theorem 5.1 (By contradiction): Suppose $\exists \theta \in \mathcal{C}^P_\Theta$ (defined in Section IV) such that $\mathcal{L}^{\theta}_{com} \subset \mathcal{L}^{\theta^*}_{com}$ where $s\sigma \in \mathcal{L}^{\theta^*}_{com}$ but $s\sigma \notin \mathcal{L}^{\theta}_{com}$, then one of two cases holds:

(i): $s\sigma$ leads to a state of X_E in the set $Bdry(X_\Omega) \cup [X_{conf} \setminus X^0_\Omega]$; but $s\sigma \notin \mathcal{L}^{\theta}_{com}$, so the system state is in $Bdry(X_\Omega) \cup [X_{conf} \setminus X^0_\Omega]$ but no communication occurs. By the construction of X_Ω, without communication at states in $Bdry(X_\Omega)$ it cannot be guaranteed that the system does not reach a state in X_Ω and hence from reaching a conflict state that cannot be resolved by communication. By definition of $X_{conf} \setminus X^0_\Omega$, not communicating when the system is at a state

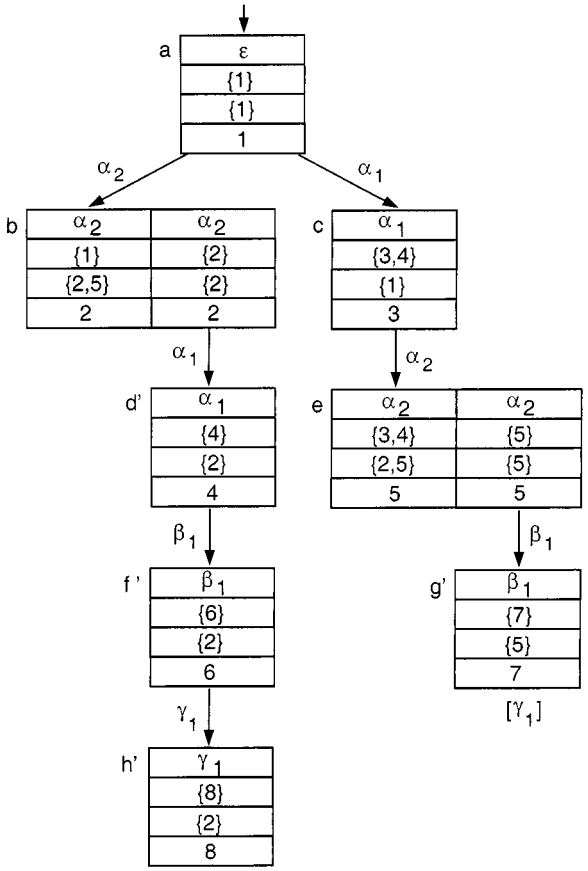

Fig. 4. Estimator structure with communication.

TABLE I
TRAJECTORIES REPRESENTED BY FIG. 4

T	$\mathbb{T}_1(T)$	$\mathbb{T}_2(T)$
ε	ε	ε
$\alpha_2 \begin{bmatrix} -- & \{1,2\} \\ \{2,5\} & -- \end{bmatrix}$	$\varepsilon \begin{bmatrix} -- \\ \{2,5\} \end{bmatrix}$	$\alpha_2 \begin{bmatrix} \{1,2\} \\ -- \end{bmatrix}$
$\alpha_2 \begin{bmatrix} -- & \{1,2\} \\ \{2,5\} & -- \end{bmatrix} \alpha_1$	$\varepsilon \begin{bmatrix} -- \\ \{2,5\} \end{bmatrix} \alpha_1$	$\alpha_2 \begin{bmatrix} \{1,2\} \\ -- \end{bmatrix}$
$\alpha_2 \begin{bmatrix} -- & \{1,2\} \\ \{2,5\} & -- \end{bmatrix} \alpha_1\beta_1$	$\varepsilon \begin{bmatrix} -- \\ \{2,5\} \end{bmatrix} \alpha_1\beta_1$	$\alpha_2 \begin{bmatrix} \{1,2\} \\ -- \end{bmatrix}$
$\alpha_2 \begin{bmatrix} -- & \{1,2\} \\ \{2,5\} & -- \end{bmatrix} \alpha_1\beta_1\gamma_1$	$\varepsilon \begin{bmatrix} -- \\ \{2,5\} \end{bmatrix} \alpha_1\beta_1\gamma_1$	$\alpha_2 \begin{bmatrix} \{1,2\} \\ -- \end{bmatrix}$
α_1	α_1	ε
$\alpha_1\alpha_2 \begin{bmatrix} -- & \{3,4,5\} \\ \{2,5\} & -- \end{bmatrix}$	$\alpha_1 \begin{bmatrix} -- \\ \{2,5\} \end{bmatrix}$	$\alpha_2 \begin{bmatrix} \{3,4,5\} \\ -- \end{bmatrix}$
$\alpha_1\alpha_2 \begin{bmatrix} -- & \{3,4,5\} \\ \{2,5\} & -- \end{bmatrix} \beta_1$	$\alpha_1 \begin{bmatrix} -- \\ \{2,5\} \end{bmatrix} \beta_1$	$\alpha_2 \begin{bmatrix} \{3,4,5\} \\ -- \end{bmatrix}$

in $X_{conf} \setminus X_\Omega^0$ implies that the correct control decision is not guaranteed. It follows that θ cannot solve Problem (P).

(ii): $s\sigma$ does not lead to a state in the set $Bdry(X_\Omega) \cup [X_{conf} \setminus X_\Omega^0]$ but is indistinguishable from $s'\sigma$ which does lead to a state in $Bdry(X_\Omega) \cup [X_{conf} \setminus X_\Omega^0]$, and by Lemma 5.1 $(\forall i \in Z) E_i(s\sigma) = E_i(s'\sigma) \Rightarrow s'\sigma \notin \mathcal{L}_{com}^\theta$. Since $s'\sigma$ leads to a state in $Bdry(X_\Omega) \cup [X_{conf} \setminus X_\Omega^0]$ and $s'\sigma \notin \mathcal{L}_{com}^\theta$ we may follow the same reasoning as Case (i) above to conclude that the policy θ cannot solve Problem (P).

Cases (i) and (ii) both contradict the assertion that $\theta \in \mathcal{C}_\Theta^P$. □

It is for Case (ii) of the proof above that Rule R1* differs most significantly from Rule R1. Rule R1* is based on there being an $s'\sigma$ that leads to a state in $Bdry(X_\Omega) \cup [X_{conf} \setminus X_\Omega^0]$ which is indistinguishable from the current state, while Rule R1 is based only on the fact that there is a state in $Bdry(X_\Omega) \cup [X_{conf} \setminus X_\Omega^0]$ that is indistinguishable from the current state regardless of whether that state is actually reachable. The qualification of using (21) and (22) for updating information is for comparison of myopic controllers with equivalent information processing techniques.

The simplicity of the proof of Theorem 5.1 is derived primarily from the structure of the sets $Bdry(X_\Omega)$ and X_Ω. Communication policies based on Rule R1* and $Bdry(X_\Omega) \cup [X_{conf} \setminus X_\Omega^0]$ are structured such that communication is only initiated when a controller obtains a state estimate that is identical to an estimate for that controller at a system state in the reachable part of the set $Bdry(X_\Omega) \cup [X_{conf} \setminus X_\Omega^0]$ or if not communicating makes such a system state reachable. As the proof relates, removing any of the communications due to this state-estimate based inference implies removing a communication at the very last point where communication would be useful in avoiding or resolving conflict states.

Although X_Ω, $Bdry(X_\Omega)$ and X_{conf} are unique sets, it is important to note that Rule R1* is not the sole, unique, rule for determining optimal communication policies based on X_Ω and $Bdry(X_\Omega) \cup [X_{conf} \setminus X_\Omega^0]$. Indeed, R1* itself is not necessarily associated to a unique communication policy. In general there may be many optimal communication policies as stated in the following theorem.

Theorem 5.2 (Non-Uniqueness): If $\mathcal{L}(H) \subseteq \mathcal{L}(G)$ is controllable with respect to $\mathcal{L}(G)$ and Σ_{uc} and observable with respect to $\mathcal{L}(G)$, Σ_o and Σ_c, then C1–C3-optimal communication policies that support solutions to Problem (P) are not unique, in general.

Proof of Theorem 5.2: The proof of Theorem 5.2 is easily performed by example, that is, we need only give a counter-example to a uniqueness hypothesis. Consider the desired behavior shown in Fig. 5 where, again, $[\gamma_i]$ implies that the event γ_i is defined in the plant but must be disabled. Assume two controllers will be used with $\Sigma_{o,1} = \{\alpha_1, \beta_1, \gamma_1\}$, $\Sigma_{c,1} = \{\gamma_1\}$, $\Sigma_{o,2} = \{\alpha_2, \gamma_2\}$ and $\Sigma_{c,2} = \{\gamma_2\}$. For the desired behavior shown in Fig. 5 the class of myopic controllers supports multiple optimal solutions to Problem (P). Two C1–C3-optimal communication policies for the desired behavior of Fig. 5 are shown in Fig. 6. For one myopic controller solution, Controller 1 communicates its state estimate following α_1. Similarly, the other optimal solution has Controller 2 communicating following α_2. These two solutions can be interpreted as first determining the set X_Ω as discussed earlier, then the communication maps are generated in a specific order. The first optimal communication policy in Fig. 6 results from building θ_1 first, and because the reachable subset of $Bdry(X_\Omega) \cup [X_{conf} \setminus X_\Omega^0]$ is altered, θ_2 is then designed to only respond to Controller 1's communications. Thus, the synthesis of θ_2 is conditioned on the design of θ_1. Vice versa for the second optimal communication map. Both policies in Fig. 6 satisfy R1*, and for comparison, the policy de-

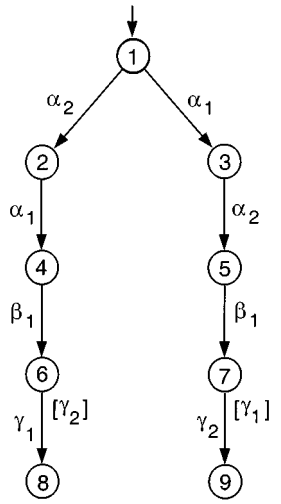

Fig. 5. Desired behavior with nonunique optimal decentralized implementation.

termined by Rule R1 is shown in Fig. 7. Thus, the fact that the set X_Ω is unique does not imply there are unique C1–C3-optimal communication policies. Note that for this example, if the controllers were allowed to be nonmyopic, then the communication policies represented in Fig. 6 would still be C1–C3-optimal; however, the state-estimates within each box would be more refined due to nonmyopia. □

E. Remarks on Myopic Controllers

It was discussed in Sections II and III that the synthesis of communication and control policies is difficult due to the lack of separation between estimation, communication and control. In Sections V-A and V-B, some of the difficulties in synthesizing communication policies were bypassed by the use of myopic controllers where a separation between communication and estimation is forced as a constraint on the information structure. The separation between communication and estimation for myopic controllers occurs because each $\mathcal{L}_\Delta$ is known *a priori* to be $\Sigma^*_{uo,i}$; thus, ignorance of future potential communications is forced into the structure of the controllers so that a simple synthesis algorithm (X_Ω-Procedure and Rule R1) may be used. Unfortunately, the synthesis of optimal communication policies (such as those consistent with R1*) is complicated by the fact that the information used to determine if a communication is required, e.g., the reachable part of $Bdry(X_\Omega) \cup [X_{conf} \setminus X^0_\Omega]$, is affected by both the decision to communicate at the present state and by similar decisions at other states in the system. The fundamental idea for this section is revealed: very simplifying assumptions were made, yet optimal policies, even in a very constrained class of controllers, are difficult to determine.

Of course, a decentralized supervisory control scheme using the myopic and state-estimate based inference maps presented in Section V-B will not, in general, allow arbitrary languages $\mathcal{L}(H)$ to be achievable under control with communication. For these permissive finite-structure myopic controllers, the conditions of Theorem 3.2 are necessary but not sufficient due to the use of only state estimates. This is an interesting departure from centralized supervisory control where observability and controllability are necessary and sufficient regardless of whether trace estimates or state estimates are used for implementation. Upon generating X_Ω, if $x_{E0} \in X_\Omega$, then the initial state of the system is such that no amount of communication using the described Ψ,

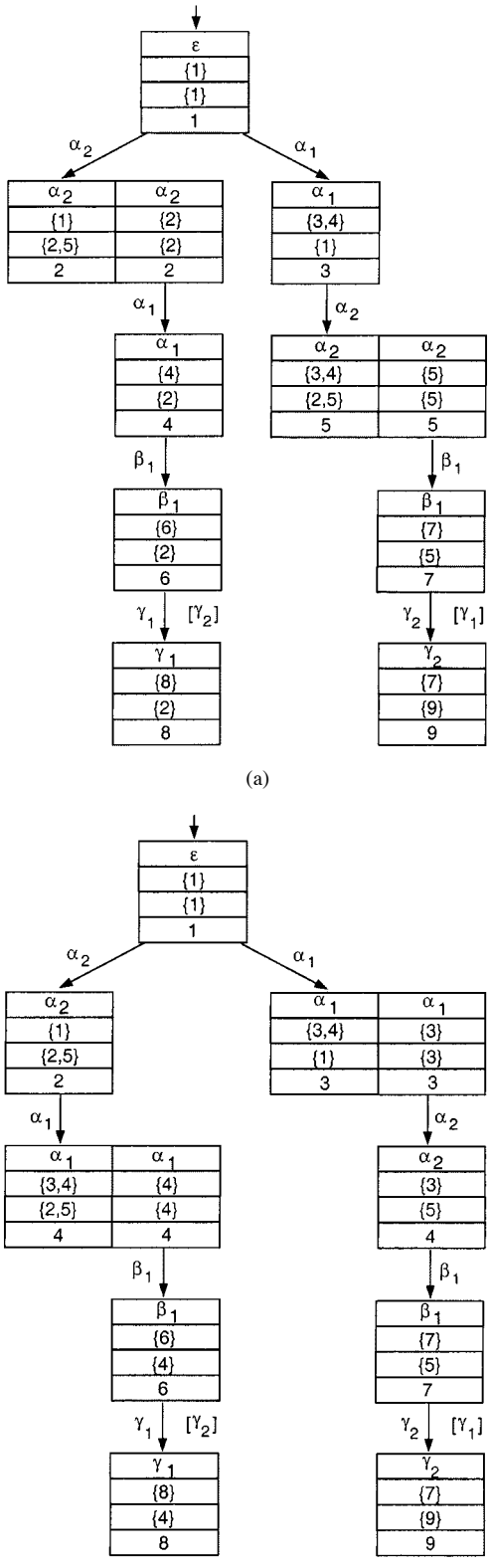

Fig. 6. Two C1–C3-optimal communication policies for Fig. 5.

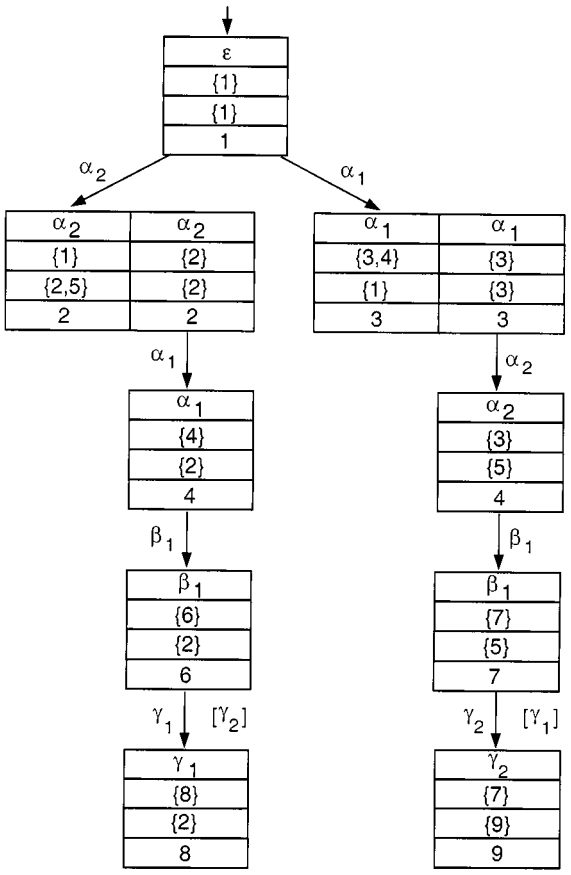

Fig. 7. Communication policy for Fig. 5 resulting from Rule R1.

γ and θ can prevent the system from reaching a conflict state. The definition of X_Ω can be used to show that $x_{E0} \notin X_\Omega$ is necessary and sufficient for $\mathcal{L}(H)$ to be exactly achievable under the permissive finite-state estimate myopic scheme described; thus, this brute-force test provides a very crude characterization of the set of languages achievable by these finite-state estimate myopic controllers.

The X_Ω-Procedure described above was presented as generating the set X_Ω by examining the *entire* state-space X_E; however, this manner of presentation was to ease the need for proofs of convergence and uniqueness. In practice, only a subset of X_Ω would be generated, as needed, using efficient algorithms; certainly, it is not desirable to examine unreachable states. However, as discussed earlier, the problem remains that reachability can not, in general, be determined *a priori*.

The utility of the X_Ω-Procedure described above applies not only to the control of a discrete-event system, but by changing the definition of "conflict state" many predicates on the states of the system can be enforced also. Examples of such predicates could include: having the completion of a task (represented by marking, for example) "known" to at least one of the controllers at all times, having diagnostic information about the plant "known" to at least one of the controllers, etc. These extensions are beyond the intended scope of this paper.

The myopic scheme presented, whatever disadvantages or shortcomings it may have, does have the benefit of a well defined and unique set $Bdry(X_\Omega) \cup [X_{conf} \setminus X_\Omega^0]$ that indicates communication requirements. As already mentioned, for arbitrarily constrained information structures, no such uniqueness property for optimal communication policies (even those resulting from the unique X_Ω) holds as indicated in Theorem 5.2.

F. Remarks on Computational Complexity

At this point, it may be useful to describe the computational complexity of some of the ideas mentioned in this paper. Decentralized control is notorious for its difficulty and associated high computational complexity. Given a choice, there seem to be few cases where decentralized control is preferred to centralized control. The motivation for this paper is that there are many cases where the designer *does not* have a choice.

The X_Ω-Procedure, as described in this section, is a good indicator of the daunting computation complexity of decentralized controller policy synthesis. The potential generation of the entire space of $(\sigma, E_1, \cdots, E_n, x)$-tuples indicates a worst-case complexity of $O(|\Sigma| \cdot 2^{|Z||X|} \cdot |X|)$. Of course, there are questions regarding decentralized supervisory control that can be answered in polynomial time. For example, testing the conditions of Theorem 3.1, specifically the controllability and observability conditions, has been shown to be polynomial [31]. The algorithm proposed in [31] was extended in [8] for the case of testing coobservability while maintaining polynomial complexity. The conjecture, given here without proof, is that a modified version of the observability and coobservability testing algorithms can be applied to the Theorem 3.2 resulting in a test with polynomial computational complexity. It is unclear as to what insight or utility would result from a proof of this conjecture.

VI. Conclusion

In this paper, the problem of achieving a given desired language using decentralized supervisory control with communication was addressed. A novel framework was presented for analysis and synthesis issues in decentralized supervisory control with communication. We characterized the classes of languages achievable for the two cases of when controllers do and do not anticipate future communications. Anticipation of future (potential) communications is described as a condition on the Ψ_i components of the information structure in the general model. Comparing the two classes of achievable languages reveals the fact that communicating controllers with finite memory and communication resources and that anticipate future communications outperform controllers with unbounded memory and communication resources that *do not* anticipate future communications.

It was shown how the general information structure can be constrained to finite objects to permit physically implementable solutions to the above-mentioned problem, and a synthesis procedure was presented for the class of finite state-estimate controllers that do not anticipate future communications. An example was given to demonstrate the use of this class of constrained information structure controllers to produce a solution to the decentralized supervisory control problem (when a solution exists for this class of controllers). It was shown that optimal communication policies exist, and a rule was given describing a

class of policies that are optimal in the sense that no communication instance can be removed from the communication policy. Other types of optimality were presented, and the nonuniqueness of optimal solutions was demonstrated.

APPENDIX A
ALTERNATIVE EXPRESIONS OF OBSERVABILITY AND COOBSERVABILITY

The material here represents an attempt to simplify the descriptions and comparisons of language observability and coobservability to other language classifications. The definitions given below are not new; they are simply the "classical" definitions re-written for simplification. Yet another way of expressing coobservability can be found in [8, Proposition 3].

Proposition A.1: Let H and G be the specification and plant automata, respectively. The language $\mathcal{L}(H)$ is *observable* with respect to $\mathcal{L}(G)$, Σ_o and Σ_c iff $(\forall\, s \in \mathcal{L}(H))(\forall\, \sigma \in \Sigma_c)$:

$$[s\sigma \notin \mathcal{L}(H)] \wedge [s\sigma \in \mathcal{L}(G)] \Rightarrow \left[\mathbb{P}^{-1}(\mathbb{P}(s))\sigma \cap \mathcal{L}(H) = \emptyset\right]. \tag{32}$$

The idea behind observability and its representation in the above proposition is the following. Following the occurrence of a trace s, if an event σ must be disabled (i.e., the trace $s\sigma$ is not in $\mathcal{L}(H)$ but is in $\mathcal{L}(G)$) then none of the traces in the supervisor's maximal information set $\mathbb{P}^{-1}(\mathbb{P}(s)) \cap \mathcal{L}(H)$ can be followed by the event σ.

Proof of Proposition A.1: Recall the "classical" definition of observability: a language $\mathcal{L}(H)$ is observable with respect to $\mathcal{L}(G)$, Σ_o, and Σ_c if $(\forall\, s,\, s' \in \Sigma^*)(\forall\, \sigma \in \Sigma_c)$:

$$\begin{aligned}\mathbb{P}(s) = &\mathbb{P}(s') \\ &\Rightarrow [(\forall\, \sigma \in \Sigma_c)\,[s'\sigma \in \mathcal{L}(H)] \wedge [s \in \mathcal{L}(H)] \\ &\quad \wedge [s\sigma \in \mathcal{L}(G)] \Rightarrow s\sigma \in \mathcal{L}(H)]. \end{aligned} \tag{33}$$

Note that the "$nextact_K$" relation that is generally associated with the definition of observability is included in the expression of (33). Without loss of generality, we may restrict attention to $s,\, s' \in \mathcal{L}(H) \subseteq \mathcal{L}(G)$, so we rewrite (33) as
$(\forall\, s,\, s' \in \mathcal{L}(H))(\forall\, \sigma \in \Sigma_c)$:

$$\begin{aligned}\mathbb{P}(s) = &\mathbb{P}(s') \\ &\Rightarrow [[s'\sigma \in \mathcal{L}(H)] \wedge [s\sigma \in \mathcal{L}(G)] \Rightarrow s\sigma \in \mathcal{L}(H)] \end{aligned} \tag{34}$$

which may be expanded to yield
$(\forall\, s,\, s' \in \mathcal{L}(H))(\forall\, \sigma \in \Sigma_c)$:

$$[s\sigma \in \mathcal{L}(H)] \vee [s\sigma \notin \mathcal{L}(G)] \vee [\mathbb{P}(s) \neq \mathbb{P}(s')] \vee [s'\sigma \notin \mathcal{L}(H)]. \tag{35}$$

Re-writing (35) yields:
$(\forall\, s,\, s' \in \mathcal{L}(H))(\forall\, \sigma \in \Sigma_c)$:

$$\begin{aligned}&[s\sigma \notin \mathcal{L}(H)] \wedge [s\sigma \in \mathcal{L}(G)] \\ &\quad \Rightarrow \neg\,[[\mathbb{P}(s) = \mathbb{P}(s')] \wedge [s'\sigma \in \mathcal{L}(H)]] \end{aligned} \tag{36}$$

which is directly equivalent to the alternative definition:
$(\forall\, s \in \mathcal{L}(H))(\forall\, \sigma \in \Sigma_c)$:

$$[s\sigma \notin \mathcal{L}(H)] \wedge [s\sigma \in \mathcal{L}(G)] \Rightarrow \left[\mathbb{P}^{-1}(\mathbb{P}(s))\sigma \cap \mathcal{L}(H) = \emptyset\right]. \tag{37}$$

□

For decentralized systems, note that under the assumption that $\cup_{i \in Z} \Sigma_{c,\,i} = \Sigma_c$, the alternative expression for observability is also equivalent to:
$(\forall\, s \in \mathcal{L}(H))(\forall\, \sigma \in \Sigma_c)$:

$$\begin{aligned}&[s\sigma \notin \mathcal{L}(H)] \wedge [s\sigma \in \mathcal{L}(G)] \\ &\quad \Rightarrow (\exists\, i \in Z)\left[\mathbb{P}^{-1}(\mathbb{P}(s))\sigma \cap \mathcal{L}(H) = \emptyset\right] \wedge [\sigma \in \Sigma_{c,\,i}]. \end{aligned} \tag{38}$$

Proposition A.2: Let $Z = \{1, \cdots, n\}$ be a set of supervisory controllers with respective sets of observable and controllable events. Let H and G be the specification and plant automata, respectively. The language $\mathcal{L}(H)$ is *coobservable* with respect to $\mathcal{L}(G), \Sigma_{o,\,1} \cdots, \Sigma_{o,\,n}$ and $\Sigma_{c,\,1}, \cdots, \Sigma_{c,\,n}$ iff
$(\forall\, s \in \mathcal{L}(H))(\forall\, \sigma \in \Sigma_c = \cup_{i \in Z} \Sigma_{c,\,i})$:

$$\begin{aligned}&[s\sigma \notin \mathcal{L}(H)] \wedge [s\sigma \in \mathcal{L}(G)] \\ &\quad \Rightarrow (\exists\, i \in Z)\left[\mathbb{P}_i^{-1}(\mathbb{P}_i(s))\sigma \cap \mathcal{L}(H) = \emptyset\right] \wedge [\sigma \in \Sigma_{c,\,i}]. \end{aligned}$$

Proof of Proposition A.2: Recall the definition of the $nextact_K$ relation:
$(\forall\, \sigma \in \Sigma)(\forall\, s,\, s' \in \Sigma^*), (s,\, \sigma,\, s') \in nextact_K$ if

$$\left[s'\sigma \in \overline{K}\right] \wedge \left[s \in \overline{K}\right] \wedge [s\sigma \in \mathcal{L}(G)] \Rightarrow \left[s\sigma \in \overline{K}\right]. \tag{39}$$

We will immediately rewrite (39) in the following way (without a loss of generality for our purposes): $(\forall\, \sigma \in \Sigma_c)(\forall\, s,\, s' \in \overline{K})$, $(s,\, \sigma,\, s') \in nextact_K$ if

$$\left[s\sigma \notin \overline{K}\right] \wedge [s\sigma \in \mathcal{L}(G)] \Rightarrow \left[s'\sigma \notin \overline{K}\right]. \tag{40}$$

Given sets of observable and controllable events for each supervisory controller in $Z = \{1, \cdots, 2\}$, a language $\mathcal{L}(H) \subset \mathcal{L}(G)$ is coobservable with respect to $\mathcal{L}(G), \Sigma_{o,\,1}, \Sigma_{o,\,2}, \Sigma_{c,\,1}, \Sigma_{c,\,1}$ if [9]
$(\forall\, s,\, s',\, s'' \in \Sigma^*)$

$$\begin{aligned}\mathbb{P}_1(s) = &\mathbb{P}_1(s') \wedge \mathbb{P}_2(s) \\ &= \mathbb{P}_2(s'') \\ &\Rightarrow (\forall\, \sigma \in \Sigma_{c,\,1} \cap \Sigma_{c,\,2})\left[(s,\, \sigma,\, s') \in nextact_{\mathcal{L}(H)}\right. \\ &\quad \left. \vee\, (s,\, \sigma,\, s'') \in nextact_{\mathcal{L}(H)}\right] \\ &\quad \wedge (\forall\, \sigma \in \Sigma_{c,\,1} \setminus \Sigma_{c,\,2})\left[(s,\, \sigma,\, s') \in nextact_{\mathcal{L}(H)}\right] \\ &\quad \wedge (\forall\, \sigma \in \Sigma_{c,\,2} \setminus \Sigma_{c,\,1})\left[(s,\, \sigma,\, s'') \in nextact_{\mathcal{L}(H)}\right] \end{aligned} \tag{41}$$

where *marking* will be considered separately. Note that while coobservability naturally generalizes to n supervisory controllers, the form of (41) does not allow the generalized definition of coobservability to be easily written. The difficulty results from having a separate "test" string, e.g., s', s'', for each

controller and from having to write out the conjuncts for every possible set of intersections of controllable events that the event σ could exist. Indeed, even the form of coobservability found in [8, Proposition 3] does not generalize easily for the same reasons. The "new" form of the definition of coobservability does not suffer this difficulty.

(41) can be written as

$$(\forall s, s', s'' \in \mathcal{L}(H))(\forall \sigma \in \Sigma_c = \cup_{i \in Z} \Sigma_{c,i})$$

$$\begin{aligned}
&\mathbb{P}_1(s) \\
&\quad = \mathbb{P}_1(s') \wedge \mathbb{P}_2(s) \\
&\quad = \mathbb{P}_2(s'') \\
&\quad \Rightarrow \left[\left[(s, \sigma, s') \in nextact_{\mathcal{L}(H)} \vee (s, \sigma, s'') \in nextact_{\mathcal{L}(H)}\right]\right. \\
&\qquad \left.\wedge [\sigma \in \Sigma_{c,1} \cap \Sigma_{c,2}]\right] \\
&\qquad \vee \left[\left[(s, \sigma, s') \in nextact_{\mathcal{L}(H)}\right] \wedge [\sigma \in \Sigma_{c,1} \setminus \Sigma_{c,2}]\right] \\
&\qquad \vee \left[\left[(s, \sigma, s'') \in nextact_{\mathcal{L}(H)}\right] \wedge [\sigma \in \Sigma_{c,2} \setminus \Sigma_{c,1}]\right]
\end{aligned} \tag{42}$$

which further simplifies to

$$(\forall s, s', s'' \in \mathcal{L}(H))(\forall \sigma \in \Sigma_c)$$

$$\begin{aligned}
\mathbb{P}_1(s) &= \mathbb{P}_1(s') \wedge \mathbb{P}_2(s) \\
&= \mathbb{P}_2(s'') \\
&\Rightarrow \left[\left[(s, \sigma, s') \in nextact_{\mathcal{L}(H)}\right] \wedge [\sigma \in \Sigma_{c,1}]\right] \\
&\qquad \vee \left[\left[(s, \sigma, s'') \in nextact_{\mathcal{L}(H)}\right] \wedge [\sigma \in \Sigma_{c,2}]\right].
\end{aligned} \tag{43}$$

Expanding (43) using (40) eliminates explicit mention of $nextact$ and yields

$$(\forall s, s', s'' \in \mathcal{L}(H))(\forall \sigma \in \Sigma_c)$$

$$\begin{aligned}
&[\mathbb{P}_1(s) \neq \mathbb{P}_1(s')] \\
&\qquad \vee [[s\sigma \in \mathcal{L}(H) \vee s\sigma \notin \mathcal{L}(G) \vee s'\sigma \notin \mathcal{L}(H)] \\
&\qquad \wedge [\sigma \in \Sigma_{c,1}]] \\
&\qquad \vee [\mathbb{P}_2(s) \neq \mathbb{P}_2(s')] \\
&\qquad \vee [[s\sigma \in \mathcal{L}(H) \vee s\sigma \notin \mathcal{L}(G) \vee s'\sigma \notin \mathcal{L}(H)] \\
&\qquad \wedge [\sigma \in \Sigma_{c,2}]]
\end{aligned} \tag{44}$$

which simplifies to

$$(\forall s \in \mathcal{L}(H))(\forall \sigma \in \Sigma_c)$$

$$\begin{aligned}
&[s\sigma \notin \mathcal{L}(H)] \wedge [s\sigma \in \mathcal{L}(G)] \\
&\quad \Rightarrow \left[\left[\mathbb{P}_1^{-1}(\mathbb{P}_1(s))\,\sigma \cap \mathcal{L}(H) = \emptyset\right] \wedge [\sigma \in \Sigma_{c,1}]\right] \\
&\qquad \vee \left[\left[\mathbb{P}_2^{-1}(\mathbb{P}_2(s))\,\sigma \cap \mathcal{L}(H) = \emptyset\right] \wedge [\sigma \in \Sigma_{c,2}]\right],
\end{aligned} \tag{45}$$

and (45) is directly equivalent to the alternative definition given in the proposition.

If one is interested in the marking action of the controllers, then similar manipulations as above can be used to show the following condition is required for coobservability

$$(\forall s \in \mathcal{L}(H))$$

$$[s \notin \mathcal{L}_m(H)] \Rightarrow (\exists i \in Z)\left[\mathbb{P}_i^{-1}(\mathbb{P}_i(s)) \cap \mathcal{L}_m(H) = \emptyset\right]. \tag{46}$$

□

Appendix B
Simple Fact For Catenative Information Sets

The following provides some assistance in proving certain claims. Definitions of sets and operations are found within the main body of the text.

Lemma B.1: Let $\sigma \in \Sigma_o$, and let $\Sigma_{uo} \subseteq \Sigma_{uo,i}$ and $\mathcal{L}(H) = \overline{\mathcal{L}(H)}$ be even, then

$$\begin{aligned}
&\left[\mathbb{P}^{-1}(\mathbb{P}(s)) \cap \mathcal{L}(H)\right] \sigma \Sigma_{uo,i}^* \cap \mathcal{L}(H) \\
&\quad = \left[\mathbb{P}^{-1}(\mathbb{P}(s\sigma)) \cap \mathcal{L}(H)\right] \Sigma_{uo,i}^* \cap \mathcal{L}(H).
\end{aligned}$$

Proof of Lemma B.1: First, we have

$$\begin{aligned}
\mathbb{P}^{-1}(\mathbb{P}(s))\sigma\Sigma_{uo,i}^* &= \mathbb{P}^{-1}(\mathbb{P}(s))\Sigma_{uo}^*\sigma\Sigma_{uo}^*\Sigma_{uo,i}^* \\
&= \mathbb{P}^{-1}(\mathbb{P}(s\Sigma_{uo}^*\sigma\Sigma_{uo}^*))\Sigma_{uo,i}^* \\
&= \mathbb{P}^{-1}(\mathbb{P}(s\sigma))\Sigma_{uo,i}^*.
\end{aligned}$$

Hence, what remains is to show that

$$\begin{aligned}
&\mathbb{P}^{-1}(\mathbb{P}(s))\sigma\Sigma_{uo,i}^* \cap \mathcal{L}(H)\sigma\Sigma_{uo,i}^* \cap \mathcal{L}(H) \\
&\quad = \mathbb{P}^{-1}(\mathbb{P}(s\sigma))\Sigma_{uo,i}^* \cap \mathcal{L}(H)\Sigma_{uo,i}^* \cap \mathcal{L}(H) \\
&\quad = \mathbb{P}^{-1}(\mathbb{P}(s))\sigma\Sigma_{uo,i}^* \cap \mathcal{L}(H).
\end{aligned}$$

It is always the case that "$\subseteq$" holds, so we need only show that

$$\begin{aligned}
&\mathbb{P}^{-1}(\mathbb{P}(s))\sigma\Sigma_{uo,i}^* \cap \mathcal{L}(H)\sigma\Sigma_{uo,i}^* \cap \mathcal{L}(H) \\
&\quad \supseteq \mathbb{P}^{-1}(\mathbb{P}(s))\sigma\Sigma_{uo,i}^* \cap \mathcal{L}(H).
\end{aligned}$$

Let $t\sigma u \in \mathbb{P}^{-1}(\mathbb{P}(s))\sigma\Sigma_{uo,i}^* \cap \mathcal{L}(H)$ where $u \in \Sigma_{uo,i}^*$, then $t \in \mathcal{L}(H)$ which implies that $t\sigma u \in \mathcal{L}(H)\sigma\Sigma_{uo,i}^*$. □

Acknowledgment

The authors wish to thank F. Lin, A. Overkamp, K. Rudie, J. H. van Schuppen, and D. Teneketzis for stimulating discussions on the topics of decentralized information and decentralized control with communication. The authors would also like to acknowledge the useful comments of the anonymous reviewers. Special thanks go to K. Rudie for suggesting that a proof accompany the alternate expression of coobservability in Appendix A.

References

[1] R. Cieslak, C. Desclaux, A. Fawaz, and P. Varaiya, "Supervisory control of discrete-event processes with partial observations," *IEEE Trans. Automa. Contr.*, vol. 33, pp. 249–260, Mar. 1988.

[2] K. Inan, "An algebraic approach to supervisory control," *Math. Contr., Signals, and Syst.*, vol. 5, no. 2, pp. 151–164, 1992.

[3] P. Kozák and W. M. Wonham, "Fully decentralized solutions of supervisory control problems," *IEEE Trans. Automatic. Contr.*, vol. 40, pp. 2094–2097, Dec. 1995.

[4] F. Lin and W. Wonham, "Decentralized supervisory control of discrete-event systems," *Information. Sci.*, vol. 44, pp. 199–224, 1988.
[5] ——, "Decentralized control and coordination of discrete-event systems with partial observations," *IEEE Trans. Automat. Contr.*, vol. 35, pp. 1330–1337, Dec. 1990.
[6] A. Overkamp and J. H. van Schuppen, "A characterization of maximal solutions for decentralized discrete event control problems," in *Proc. Int. Workshop on Discrete Event Syst.*, Edinburgh, UK, Aug. 1996, pp. 278–283.
[7] J. H. Prosser, M. Kam, and H. G. Kwatny, "Decision fusion and supervisor synthesis in decentralized discrete-event systems," in *Proc. 1997 Amer. Contr. Conf.*, June 1997, pp. 2251–2255.
[8] K. Rudie and J. C. Willems, "The computational complexity of decentralized discrete-event control problems," *IEEE Trans. Automat. Contr.*, vol. 40, pp. 1313–1318, July 1995.
[9] K. Rudie and W. Wonham, "Think globally, act locally: Decentralized supervisory control," *IEEE Trans. Automat. Contr.*, vol. 37, pp. 1692–1708, Nov. 1992.
[10] S. Takai, "On the language generated under fully decentralized supervision," *IEEE Trans. Automat. Contr.*, vol. 43, pp. 1253–1255, Sept. 1998.
[11] Y. Willner and M. Heymann, "Supervisory control of concurrent discrete-event systems," *Int. J. Contr.*, vol. 54, no. 5, pp. 1143–1169, 1991.
[12] S. L. Ricker and K. Rudie, "Know means no: Incorporating knowledge into decentralized discrete-event control," in *Proc. 1997 Amer. Contr. Conf.*, June 1997, pp. 2348–2353.
[13] K. Rudie, S. Lafortune, and F. Lin, "Minimal communication in a distributed discrete-event control system," in *Proc. 1999 IEEE ACC*, 1999, pp. 1965–1970.
[14] J. H. van Schuppen, "Decentralized supervisory control with information structures," in *Proc. Int. Workshop on Discrete Event Syst.*, Cagliari, Italy, Aug. 1998, pp. 36–41.
[15] K. C. Wong and J. H. van Schuppen, "Decentralized supervisory control of discrete event systems with communication," in *Proc. Int. Workshop on Discrete Event Syst.*, Edinburgh, UK, Aug. 1996, pp. 284–289.
[16] D. Teneketzis, "On information structures and nonsequential stochastic control," *CWI Quarterly*, vol. 9, no. 3, pp. 241–260, 1996.
[17] C. G. Cassandras and S. Lafortune, *Intro. Discrete Event Syst.*. Norwell, MA: Kluwer, 1999.
[18] R. Kumar and V. K. Garg, *Modeling and Contr. Logical Discrete Event Syst.*. Norwell, MA: Kluwer, 1995.
[19] P. Ramadge and W. Wonham, "The control of discrete event systems," *Proc. IEEE*, vol. 77, pp. 81–98, Jan. 1989.
[20] H. Witsenhausen, "Separation of estimation and control for discrete time systems," *Proc. IEEE*, vol. 59, no. 11, pp. 1557–1566, 1971.
[21] H. Witsenhausen, "Equivalent stochastic control problems," in *Math. Contr., Signals, and Syst.*: Springer-Verlag, 1988, vol. 1, pp. 3–11.
[22] T.-S. Yoo and S. Lafortune, *A New Architecture for Decentralized Supervisory Contr.*, 1999.
[23] A. Arnold, *Finite Trans. Syst.*. Englewood Cliffs, NJ: Prentice Hall, 1994.
[24] Z. Kohavi, *Switching and Finite Automata Theory*, 2nd ed. New York: McGraw-Hill, 1978.
[25] G. Barrett and S. Lafortune, "A novel framework for decentralized supervisory control with communication," in *Proc. IEEE I. Conf. Syst., Man, Cybern.*, San Diego, CA, 1998, pp. 617–620.
[26] ——, "On the synthesis of communicating controllers with decentralized information structures for discrete-event systems," in *Proc. 37th IEEE Conf. Dec. Contr.*, Tampa, FL, 1998, pp. 3281–3286.
[27] R. Debouk, S. Lafortune, and D. Teneketzis, "On an optimization problem in sensor selection for failure diagnosis," Department of Electrical Engineering and Computer Science, University of Michigan, http://www.eecs.umich.edu/umdes, 1999.
[28] A. Haji-Valizadeh and K. A. Loparo, "Minimizing the cardinality of an events set for supervisors of discrete-event dynamical systems," *IEEE Trans. Automat. Contr.*, vol. 41, pp. 1579–1593, Nov. 1996.
[29] R. Sengupta and S. Lafortune, "An optimal control theory for discrete event systems," *SIAM J. Contr. Optim.*, vol. 46, pp. 488–541, Mar. 1998.
[30] G. Barrett and S. Lafortune, "Bisimulation, the supervisory control problem and strong model matching for finite state machines," *Discrete Event Dyn. Syst.: Theory and Appl.*, vol. 8, pp. 377–429, Dec. 1998.
[31] J. N. Tsitsiklis, "On the control of discrete event dynamical systems," *Math. Contr. Signals and Syst.*, vol. 2, no. 2, pp. 95–107, 1989.

George Barrett (S'90-M'99) received two M.S. degrees from North Carolina State University in 1994 and a Ph.D. degree in electrical engineering systems from the University of Michigan, Ann Arbor, in 1999.

He currently holds a position of Senior Research Scientist in the System and Information Sciences Group (RSI) at the Johns Hopkins University Applied Physics Laboratory. In 1998, he won the Student Best Paper Contest at the 37th IEEE Conference on Decision and Control. He has been a consultant to industry, and he holds a U.S. Patent. His current research interests are fundamental limitations and applications of resource allocation, communication and coordination in decentralized and distributed systems.

Stéphane Lafortune (M'86-SM'97-F'99) received the B.Eng. degree from École Polytechnique de Montréal in 1980, the M.Eng. degree from McGill University in 1982, and the Ph.D. degree from the University of California at Berkel in 1986, all in electrical engineering.

Since September 1986, he has been with the University of Michigan, Ann Arbor, where he is a Professor of electrical engineering and computer science. His research interests are in discrete event systems and in intelligent transportation systems. He co-authored, with C. Cassandras, the textbook *Introduction to Discrete Event Systems* (Norwell, MA: Kluwer, 1999). Recent publications are available at the website www.eecs.umich.edu/umdes.