Decentralized Supervisory Control with Communication

by

George Barrett and Stéphane Lafortune


Last Revision: 51198 MJD

ALL RIGHTS RESERVED

THE UNIVERSITY OF MICHIGAN

SYSTEM SCIENCE AND ENGINEERING DIVISION
DEPARTMENT OF ELECTRICAL ENGINEERING AND
COMPUTER SCIENCE
Room 4234, EECS Building
Ann Arbor, MI 48109-2122
USA
Decentralized Supervisory Control with Communication

George Barrett and Stéphane Lafortune
Department of Electrical Engineering and Computer Science
The University of Michigan
1301 Beal Avenue
Ann Arbor MI 48109-2122

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.

Keywords: discrete-event systems, decentralized control, communicating controllers

1 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 and many classifications of language structure admitting decentralized control without communication exist including co-observability, decomposability and strong decomposability.

Control of logical discrete-event systems with communication has been pondered recently in . 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 for example (paraphrased from ):

(Q1) Who should know what and when?
(Q2) Who should communicate with whom?
(Q3) When should controllers communicate?
(Q4) 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 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 model. The contributions of this work are:

1. A novel information structure formalism is presented in Section 2 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 3 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 (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 non-trivial implications for synthesis algorithms.

5. Several basic notions for optimality of communication policies are given in Section 4.

6. In Section 5.1 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.

8. In Section 5.4 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 [3Γ7Γ14Γ18]. The organization of this paper is as follows. Section 2 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 3. Two cases for existence are considered: when controllers do and do not anticipate future communications. Communication policy optimality is discussed in Section 4. A constrained class of controllers are examined in Section 5 where the finite model is described and a synthesis procedure is given. The synthesis procedure is illustrated by example in Section 5.3 and the policies derived from the procedure are proved optimal in Section 5.4. Non-uniqueness of optimal solutions is shown in Section 5.4 and the paper concludes with a summary in Section 6.
2 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, it is necessary to specify:

- the events that each controller can control
- the symbols that each controller can communicate
- the information available to each controller to be used in its control and communication policies
- constraints on the forms of the control and communication policies.

The first item in this list is generally specified a priori for a given number of controllers acting on a given system. The second and fourth items are generally dictated by 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 [21] of the control system and is now described.

Consider a discrete-event system modeled by an automaton $G$ with associated language $\mathcal{L}(G)$. Associated with the system is a set of events that can be disabled $\Sigma_G$ and a set of observable events $\Sigma_o$. The set of all events is denoted by $\Sigma = \Sigma_G \cup \Sigma_o$, and it will be assumed that $\Sigma$ is equivalent to the set of events of $G$. 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 $Z = \{1, 2, \ldots, n\}$. Each controller $i \in Z$ has an associated set of events $\Sigma_{i,o} \subseteq \Sigma_o$ that it can disable, a set $\Xi_i$ of symbols that it is allowed to communicate to other controllers, and a set of events $\Sigma_{i,u} \subseteq \Sigma_{uc}$ that it can directly observe. The events that are unobservable to each controller are given by $\Sigma_{uo,i} = \Sigma \setminus \Sigma_{i,o}$. To represent the fact that controllers have only partial observations of traces in $\mathcal{L}(G)$, a projection operator $\mathcal{P}_{\Sigma_{i,o}} : \Sigma^* \rightarrow \Sigma^*_{i,o}$ is defined as follows:

$$\mathcal{P}_{\Sigma_{i,o}}(\sigma) = \sigma$$ if $\sigma \in \Sigma_{i,o}$, otherwise $\mathcal{P}_{\Sigma_{i,o}}(\sigma) = \varepsilon$ ($\varepsilon$ indicates the empty trace). The assignment is $\mathcal{P}_{\Sigma_{i,o}}(s) = \mathcal{P}_{\Sigma_{i,o}}(s) \mathcal{P}_{\Sigma_{i,o}}(\sigma)$. If the subscript alphabet is not given, e.g., $\mathcal{P}_i$ then it is assumed the subscript is the set of all observable events $\Sigma_o$. The inverse projection of $\mathcal{P}_{\Sigma_{i,o}}$ is the mapping $\mathcal{P}_{\Sigma_{i,o}}^{-1} : \Sigma^*_{i,o} \rightarrow 2^\Sigma^*$ defined as $\mathcal{P}_{\Sigma_{i,o}}^{-1}(\omega) = \{s \in \Sigma^* | \mathcal{P}_{\Sigma_{i,o}}(s) = \omega\}$.

We will denote the closed-loop language of the plant $\mathcal{L}(Z/G)$ of the set of supervisors $Z$ by $\mathcal{L}(Z/G)$. The precise manner in which the control signals are combined to affect the plant e.g., conjunction, disjunction, prioritization, etc., is a constraint on the general decentralized control problem and must be stated as an assumption for specific problems. The various methods of combining control signals will not be dealt with in this paper (see [13]), but regardless of how the control signals are combined, the result will be denoted by $\mathcal{L}(Z/G)$.

Following a trace $s \in \mathcal{L}(Z/G)$ the unprocessed data available to a controller $i \in Z$ for making a decision is represented by an extended trace [1] of the form:

$$t^k_i = \sigma^i_1 \prod_{j \in Z} \Xi^i_1 \sigma^i_2 \prod_{j \in Z} \Xi^i_2 \ldots \sigma^i_k \prod_{j \in Z} \Xi^i_k \in T_i$$

2 With a slight abuse of notation, we will use "s\sigma" for catenation instead of "\{s\}\{\sigma\}".
where $\sigma^i \in \Sigma_{o,i} \cup \{\varepsilon\} \Xi^{j,i}$ is a (possibly empty) set of symbols communicated from Controller $j$ to Controller $i$ and we must have that $s \in \mathcal{P}_{\Xi^{j,i}}^{-1}(\sigma^i \sigma^i \sigma^i \ldots \sigma^i)$. Note that Controller $i$ only "observes" an $\varepsilon$-transition if two communications arrive at controller $i$ with no event in $\Sigma_{o,i}$ occurring between them. It will be understood that if $\Xi^{j,i}$ is empty then no communication occurred from $j$ to $i$. We will denote the set of all possible symbols that can be communicated as $\Xi = \cup_{j \in Z} \Xi^{j,i}$ and the extended traces will be referred to as observation/communication trajectories or more simply trajectories. The trajectories observed by controllers represent the controllers' information states and are projected versions of the global trajectories. The set of all global system trajectories is

$$T \subseteq (\Sigma \left[ C_{\Xi}^{n \times n} \right] \left[ C_{\Xi}^{n \times n} \right])^* \tag{1}$$

where $C_{\Xi}^{n \times n}$ is the set of all $n \times n$-matrices that have elements that are sets $\Xi^{j,i} \subseteq \Xi_{j}^{O} i, j \in Z$ of communication symbols. A matrix $C_{\Xi} \in C_{\Xi}^{n \times n}$ is a communication matrix with one axis indicating which controller is sending a set of symbols and the other axis indicating receivers. The $j, i$-th element of $C_{\Xi} C_{\Xi}^{j,i}$ is the set of symbols $\Xi^{j,i}$ being communicated from Controller $j$ to Controller $i$. The dynamic configuration of the communication channels as captured in this trajectory model is closely related to mobility in Milner’s $\pi$-calculus \cite{10P11}.

The structure of \ref{eq:1} is interpreted as follows: the event set $\Sigma$ corresponds to events that occur in the plant $\Gamma$ the first $C_{\Xi}^{n \times n}$ component corresponds to communications that are transmitted at that instant $\Gamma$ and the last $C_{\Xi}^{n \times n}$ component corresponds to messages received at that instant. In what follows we will restrict attention to the case of zero delay and lossless communications in which the structure of $T$ simplifies greatly to

$$T \subseteq (\Sigma \left[ C_{\Xi}^{n \times n} \right])^* \tag{2}.$$

Without this simplification more assumptions and details must be fixed that do not relate to our present goals.

When examining a single trajectory $\Gamma t \in TT$ it may be of interest to only consider the event components of that trajectory; hence define the mapping $T : (\Sigma \left[ C_{\Xi}^{n \times n} \right])^* \rightarrow \Sigma^*$ that essentially projects trajectories to their underlying event sequences. To derive the trajectory sets $TT_\Gamma$ from the global set of trajectories $TT_\Gamma$ a prefix-preserving projection operator $\pi_i : T \rightarrow T_\Gamma$ is defined as:

$$\pi_i(\sigma \left[ C_{\Xi} \right]) = \begin{cases} 
\varepsilon & \text{if } \sigma \not\in \Sigma_{o,i} \text{ and } \exists j \text{ s.t. } C_{\Xi}^{j,i} \neq \emptyset, \\
\varepsilon & \text{if } \sigma \not\in \Sigma_{o,i} \text{ and } \forall j \text{ s.t. } C_{\Xi}^{j,i} = \emptyset, \\
\sigma & \text{if } \sigma \in \Sigma_{o,i} \text{ and } \exists j \text{ s.t. } C_{\Xi}^{j,i} \neq \emptyset, \\
\sigma & \text{if } \sigma \in \Sigma_{o,i} \text{ and } \forall j \text{ s.t. } C_{\Xi}^{j,i} = \emptyset,
\end{cases}$$

$$\pi_i(t\sigma \left[ C_{\Xi} \right]) = \pi_i(t) \pi_i(\sigma \left[ C_{\Xi} \right]).$$

Given the observation of a trajectory $t_i \in T_i$ by Controller $i$ the inference of what trace (or set of traces) could have actually occurred in $L(Z/G)$ will be determined by the mapping

$$\Psi_i : T_i \rightarrow \rho(L(Z/G))$$

where $\rho(X)$ denotes the power set of $X$. Loosely speaking the inference map $\Psi_i$ reflects the “reasoning” capabilities of controller $i$ given the “knowledge” of the global system behavior including the “reasoning” of other controllers $\Gamma$ all control policies $\Gamma$ and all communication policies.\footnote{Anthropomorphic terminology is common; however, a controller need not be aware of the reasoning involved in its policies.}
The information structure [21Γ24Γ25] presented here for decentralized supervisory control problems is given by

\[ I := \{ T, (\Sigma_c; i, \Xi_i, \Psi_i) : i = 1, \ldots, n \}. \]  

Equation (3) indicates the lack of "separation" of estimation, control, and communication; the design of control and communication policies are dependent on the information structure and the information structure is dependent on the policies to be designed. For example, the inference maps which generate an information basis for the control and communication policies may depend on all of \( TT \) and \( T \) is built up from individual control and communication decisions resulting from information derived by the controllers’ inference maps. Due to the dependence of the information structure on the policies to be synthesized it is not possible in general to specify the complete information structure \textit{a priori} to the synthesis problem. Only the form (represented by resource or functional constraints) of the information structure can be specified before synthesizing inference, control, and communication policies. The information structure just defined to the best of our knowledge is novel in the field of Discrete-Event Systems and we hope that it will provide a useful formalism to analyze and compare different schemes in decentralized supervisory control. The term “information structure” originated in decision analysis literature and can have many interpretations. Recently other work [22] also uses this terminology for discrete-event systems. The information structures presented in [22] resemble the \textit{information patterns} given in [24] while the information structure presented here in Eqn (3) is intended to more closely resemble that of the intrinsic model given in [21Γ25].

To complete the modeling framework of this section two additional components will be formally defined: the control policy represented by a set of disablement maps and the communication policy represented by a set of communication maps. The control policy is \( \Gamma = \{ \Gamma_1, \Gamma_2, \ldots, \Gamma_n \} \) where each \( \Gamma_i \) is a disablement map:

\[ \Gamma_i : \rho(\mathcal{L}(Z/G)) \rightarrow \rho(\Sigma_c; i). \]  

Finally, the communication policy is \( \Theta = \{ \Theta_1, \Theta_2, \ldots, \Theta_n \} \) where each \( \Theta_i \) is a communication map:

\[ \Theta_i : \rho(\mathcal{L}(Z/G)) \rightarrow \prod_{k=1}^{n} \rho(\Xi_i). \]  

The problem investigated in this paper can now be formally stated:

(P) Given a plant \( G \) with generated language \( \mathcal{L}(G) \) a desired behavior modeled by an automaton \( H \) with language \( \mathcal{L}(H) \subseteq \mathcal{L}(G) \Gamma \) and a set of controllers \( Z = \{1, 2, \ldots, n\} \) construct control and communication policies for the controllers \( \Gamma = \{ \Gamma_1, \Gamma_2, \ldots, \Gamma_n \} \) and \( \Theta = \{ \Theta_1, \Theta_2, \ldots, \Theta_n \} \) respectively such that \( \mathcal{L}(Z/G) = \mathcal{L}(H) \).

Control and communication policies that are admissible to the solution of Problem (P) are those which are consistent with Eqn(4) and Eqn(5) i.e. past observations are mapped to decisions on what to control and communicate and two decisions of a controller that are based on the same observations or same inferences must be equivalent.

The term "separation" has not been formally defined for logical-DES policies; we will use the term to imply the concept of a lack of functional dependence between policies. Quotation marks will not be used for later occurrences of the term in this text.
3 Existence of Solutions

The primary purpose of this section is to present necessary and sufficient conditions for the existence of information structures that support a solution to Problem (P) as stated in Section 2. Two cases will be examined: when controllers can anticipate (expect) receiving communications along certain traces and can therefore adjust their estimates based on these anticipated communications and 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 but would represent a causality violation. The implied meaning is that controllers have prior knowledge of (and make use of) the communication policy. The assumptions used in the results that follow are:

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

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 permissive; that is following the observation of \( t_i \) Controller \( i \) may only disable an event \( \sigma \in \Sigma_{c,i} \) if \( \sigma \) should not be enabled following any trace in \( \Psi_i(t_i) \).

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

A.8 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 and 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.

A.9 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 untermined 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.10 Controllers are able to determine from which controllers they receive communicated symbols.

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 (Existence for Case I) An information structure 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 $\Sigma_c$;
2. $\mathcal{L}(H)$ is observable\(^6\) with respect to $\mathcal{L}(G)$, $\Sigma_o$ and $\Sigma_c$.

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 system 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.

It will be useful to define the following function $\Delta_{H \times G} : \rho(X_{H \times G}) \times \rho(\Sigma^*) \rightarrow \rho(X_{H \times G})$ as

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

that is $\Delta_{H \times G}(X_\Delta, L_\Delta)$ is the set of states of $[H \times G]$ reachable from the states in $X_\Delta$ by traces in $L_\Delta$.

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 $\Sigma_o$ 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.9) the controllers exchange state estimates and most importantly the controllers “expect” to receive a communication following any observable event that they do not directly observe. The controllers do not have to communicate the actual observable event that occurs merely their state estimates. For the sake of expositional simplicity we will only prove the theorem for the case of two controllers; however the proof generalizes to $n$ with the use of two-way broadcasting.

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

$$E_1(\varepsilon) = \Delta_{H \times G}(\{x_{H_0,G_0}\}, \Sigma^*_o)$$
$$E_2(\varepsilon) = \Delta_{H \times G}(\{x_{H_0,G_0}\}, \Sigma^*_a).$$

Note that in both estimates above $L_\Delta = \Sigma^*_o$ instead of $L_\Delta = \Sigma^*_o,i$. 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.9 and since controllers expect to receive these communications the state estimate for Controller $i$ is updated following the occurrence of any observable event.

Following $s \sigma \in \mathcal{L}(H \times G)$ there are four possible cases and state-estimate update rules. These rules explicitly represent how state estimates are intersected while using the knowledge of which

---

\(^5\) Recall that a language $K$ is controllable w.r.t. $L$ and $\Sigma_w$ iff $\overline{K} \cap L \subseteq \overline{K}$.

\(^6\) The definition of observability is recalled in Appendix A.
controller(s) communicated those state estimates.

Case (i): \( \sigma \in \Sigma_{o,1} \cap \Sigma_{o,2} \Gamma \)

\[
E_1(s \sigma) = \Delta_{H \times G}(E_1(s), \Sigma_{uo}^*) \cap \Delta_{H \times G}(E_2(s), \Sigma_{uo}^*) \tag{6}
\]

\[
E_2(s \sigma) = \Delta_{H \times G}(E_1(s), \Sigma_{uo}^*) \cap \Delta_{H \times G}(E_2(s), \Sigma_{uo}^*) \tag{7}
\]

Case (ii): \( \sigma \in \Sigma_{o,1} \cap \Sigma_{uo,2} \Gamma \)

\[
E_1(s \sigma) = \Delta_{H \times G}(E_1(s), \Sigma_{uo}^*) \cap \Delta_{H \times G}(E_2(s), \Sigma_{uo}^*) \tag{8}
\]

\[
E_2(s \sigma) = \Delta_{H \times G}(E_1(s), \Sigma_{uo}^*) \cap \Delta_{H \times G}(E_2(s), \Sigma_{uo}^*) \tag{9}
\]

Case (iii): \( \sigma \in \Sigma_{uo,1} \cap \Sigma_{o,2} \Gamma \)

\[
E_1(s \sigma) = \Delta_{H \times G}(E_1(s), (\Sigma_{uo}^*) \cap \Delta_{H \times G}(E_2(s), \Sigma_{uo}^*) \tag{10}
\]

\[
E_2(s \sigma) = \Delta_{H \times G}(E_1(s), \Sigma_{uo}^*) \cap \Delta_{H \times G}(E_2(s), \Sigma_{uo}^*) \tag{11}
\]

Case (iv): \( \sigma \in \Sigma_{uo,1} \cap \Sigma_{uo,2} = \Sigma_{uo} \Gamma \)

\[
E_1(s \sigma) = \Delta_{H \times G}(E_1(s), \Sigma_{uo}^*) \tag{12}
\]

\[
E_2(s \sigma) = \Delta_{H \times G}(E_2(s), \Sigma_{uo}^*) \tag{13}
\]

Consider the state estimate generated by a centralized controller that observes all events in \( \Sigma_o \). This centralized estimate \( E_{H \times G} \Gamma \) is determined by

\[
E_{H \times G}(\varepsilon) = \Delta_{H \times G}(\{x_{H0,00}\}, \Sigma_{uo}^*) \tag{14}
\]

\[
E_{H \times G}(s \sigma) = \Delta_{H \times G}(E_{H \times G}(s), \Sigma_{uo}^* \sigma) \tag{15}
\]

Notice that the update rules represented by Eqns (6)–(11) 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 \( \sigma \Gamma \) i.e., if communication is instantaneous then no unobservable continuations can occur following the generation of \( \sigma \) 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 \( \mathcal{L}_\Delta \).

It will now be shown by the use of induction on the length of traces in \( \mathcal{L}(H \times G) \) that the decentralized controllers produce the same state estimates as that of the centralized controller.

(Basis of induction) For trace \( t = \varepsilon \Gamma \) \( E_1(\varepsilon) = E_2(\varepsilon) = \Delta_{H \times G}(\{x_{H0,00}\}, \Sigma_{uo}^*) = E_{H \times G}(\varepsilon) \) so the assertion is true for the base case.

(Inductive hypothesis) Assume the assertion is true for \( t = s \Gamma \) and prove it for \( t = s \sigma \).

(Inductive step) Each of the four cases is examined.

Case (i): \( \sigma \in \Sigma_{o,1} \cap \Sigma_{o,2} \). Since by the induction hypothesis \( E_1(s) = E_2(s) = E_{H \times G}(s) \) we have

\[
\Delta_{H \times G}(E_1(s), \Sigma_{uo}^*) = \Delta_{H \times G}(E_2(s), \Sigma_{uo}^*) = \Delta_{H \times G}(E_{H \times G}(s), \Sigma_{uo}^*)
\]
hence
\[ E_1(s\sigma) = E_2(s\sigma) = \Delta_{H \times G}(E_1(s), \Sigma_{uo}^* \sigma) \cap \Delta_{H \times G}(E_2(s), \Sigma_{uo}^* \sigma) \]
\begin{align*}
&= \Delta_{H \times G}(E_1(s), \Sigma_{uo}^* \sigma) \\
&= \Delta_{H \times G}(E_2(s), \Sigma_{uo}^* \sigma) \\
&= \Delta_{H \times G}(E_{H \times G}(s), \Sigma_{uo}^* \sigma) \\
&= E_{H \times G}(s\sigma).
\end{align*}

Case (ii): \( \sigma \in \Sigma_{o,1} \cap \Sigma_{uo,2} \). Again by the induction hypothesis \( E_1(s) = E_2(s) = E_{H \times G}(s) \) we have
\[ \Delta_{H \times G}(E_1(s), \Sigma_{uo}^* \sigma) = \Delta_{H \times G}(E_2(s), \Sigma_{uo}^* \sigma) = \Delta_{H \times G}(E_{H \times G}(s), \Sigma_{uo}^* \sigma), \]
from which it follows that
\[ \Delta_{H \times G}(E_1(s), \Sigma_{uo}^* \sigma) \subseteq \Delta_{H \times G}(E_1(s), \Sigma_{uo}^*(\Sigma_{o,1} \setminus \Sigma_{o,2})) \]
\[ = \Delta_{H \times G}(E_2(s), \Sigma_{uo}^*(\Sigma_{o,1} \setminus \Sigma_{o,2})), \]

hence
\[ E_1(s\sigma) = E_2(s\sigma) = \Delta_{H \times G}(E_1(s), \Sigma_{uo}^* \sigma) \cap \Delta_{H \times G}(E_2(s), \Sigma_{uo}^*(\Sigma_{o,1} \setminus \Sigma_{o,2})) \]
\begin{align*}
&= \Delta_{H \times G}(E_1(s), \Sigma_{uo}^* \sigma) \\
&= \Delta_{H \times G}(E_2(s), \Sigma_{uo}^* \sigma) \\
&= \Delta_{H \times G}(E_{H \times G}(s), \Sigma_{uo}^* \sigma) \\
&= E_{H \times G}(s\sigma).
\end{align*}

Case (iii): Same reasoning as Case (ii).

Case (iv): Since \( E_1(s\sigma) = \Delta_{H \times G}(E_1(s), \Sigma_{uo}^* \sigma) = E_1(s) = E_2(s) = E_{H \times G}(s) \) we have
\[ E_1(s\sigma) = E_2(s\sigma) = E_{H \times G}(s\sigma), \]

which completes the induction step.

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 \( \Delta_{H \times G}(E_{H \times G}(s), \Sigma_{uo}^*) \); but we have that \( E_1(s\sigma) = E_2(s\sigma) = E_{H \times G}(s\sigma) \). 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 \( \Sigma_{uo} \Gamma \) and \( \mathcal{L}(H) \) is observable with respect to \( \mathcal{L}(G) \Gamma \Sigma_{o} \) and \( \Sigma_{I} \Gamma \) the first assertion of the theorem is true. Furthermore the estimator structures used here for each controller are finite by the boundedness of \( \rho(X_{H \times G}) \). 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 or it will be followed by a communication. Memory of prior communications is captured by the state-update rules for the estimator structure Eqs (6)–(14). 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 $L_{A}$ 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 that is, the controllers’ estimates do not take into account future communications.

Consider controllers that maintain trace estimates (not state estimates). That is, following a trajectory $t \in T$, each controller $i \in Z$ has an estimate $\Psi_{i}(\pi_{i}(t))$ of traces 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 the centralized estimate for $t \in T$ and $s = T(t)$ is:

$$\Psi_{central}(\pi_{central}(t)) = P^{-1}(P(T(t)) \cap L(H)$$

$$= P^{-1}(P(s)) \cap L(H),$$

where in the centralized case there is no communication and the trajectories degenerate to event traces and $\pi_{central}$ can be viewed to some extent as the standard natural projection $P$. Because of its dependence solely on $s$ (with no communication) we may denote $\Psi_{central}(\pi_{central}(t))$ by $\tilde{\Psi}_{central}(s)$ for brevity and convenience.

In what follows we will say that Controller $i$ is myopic if $\Psi_{i}$ does not take into account future communications between the controllers that is,

$$(\forall t_{i} \in T_{i}) \ \Psi_{i}(t_{i}) = \Psi_{i}(t_{i})^{\sum_{u_{i},i} \cap L(H)}. \quad (16)$$

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

**Lemma 3.1** Let $\tilde{\Psi}_{i}(s) = \tilde{\Psi}_{i}(T(t)) = \Psi_{i}(\pi_{i}(t))$, $i = 1, \ldots, n$, be trace estimates generated by myopic controllers that communicate their trace estimates via two-way broadcast following every locally observed event. Then for $s \in L(H)$

$$\tilde{\Psi}_{i}(s) = \tilde{\Psi}_{central}(s)^{\sum_{u_{i},i} \cap L(H)$$

$$= P^{-1}(P(s))^{\sum_{u_{i},i} \cap L(H),}$$

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 $t = \varepsilon$

$$\tilde{\Psi}_{central}(\varepsilon)^{\sum_{u_{i},i} \cap L(H) = [P^{-1}(P(\varepsilon))]^{\sum_{u_{i},i} \cap L(H)}$$

$$= \sum_{u_{i},i} \cap L(H)$$

$$= \tilde{\Psi}_{i}(\varepsilon),$$

hence the assertion holds for the base case. Assume the assertion holds for $t = s\Gamma$ and we must show that it holds for $t = s\sigma$. Let $Z_{\sigma} \subset Z$ denote the set of controllers $i \in Z$ for which $\sigma \in \sum_{o_{i},i}$. There are two cases: $Z_{\sigma} \neq \emptyset \Gamma$ and $Z_{\sigma} = \emptyset$.
Case (i): $Z_{\sigma} \neq \emptyset$. The general rule used here for updating trace estimates following communication is given by:

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

Substituting from the induction hypothesis:

$$\bar{\Psi}_i(s\sigma) = \left[ \bigcap_{j \in Z_{\sigma}} \left( \bar{\Psi}_{central}(s)^{s_{uo,j}} \Sigma_{uo,j}^{s} \cap \mathcal{L}(H) \right)^{\sigma} \cap \bigcap_{j \notin Z_{\sigma}} \left( \bar{\Psi}_{central}(s)^{s_{uo,j}} \Sigma_{uo,j}^{s} \cap \mathcal{L}(H) \right)^{\sigma} \right] \Sigma_{uo,i}^{s} \cap \mathcal{L}(H).$$

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

$$\bar{\Psi}_i(s\sigma) = \left[ \bigcap_{j \in Z} \left( \bar{\Psi}_{central}(s)^{s_{uo,j}} \Sigma_{uo,j}^{s} \cap \mathcal{L}(H) \right)^{\sigma} \cap \bigcap_{j \notin Z} \left( \bar{\Psi}_{central}(s)^{s_{uo,j}} \Sigma_{uo,j}^{s} \cap \mathcal{L}(H) \right)^{\sigma} \right] \Sigma_{uo,i}^{s} \cap \mathcal{L}(H).$$

Use $\bar{\Psi}_{central}(s) = \bar{\Psi}_{central}(s)^{s_{uo}} \cap \mathcal{L}(H)$ to reduce:

$$\bar{\Psi}_i(s\sigma) = \bar{\Psi}_{central}(s)^{s_{uo,i}} \cap \mathcal{L}(H)$$

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

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

For both 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 (Existence for Case II)** Let $Z$ be a set of myopic 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 $\Sigma_{uc}$.
2. $(\forall s \in \mathcal{L}(H))(\forall \sigma \in \Sigma_{c})$:

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

\[11\]
Proof of Theorem 3.2
Sufficiency: We can construct supervisors by

(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 and

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

\[
\Gamma_i(\Psi_i(s)) = \{\sigma \in \Sigma_{c,i} | \Psi_i(s)\sigma \cap \mathcal{L}(H) = \emptyset\}. \tag{17}
\]

This form of disable map is analogous to the “pass the buck” construction used in [18]; a controller only disables an event if there is no ambiguity to that controller that the event should be disabled over all traces in \( \Psi_i(s) \).

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

Necessity: If Problem (P) is solvable using \( Z\Gamma \) then it is solvable in the centralized case; hence 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 \( \Gamma \) but the second condition does not hold. Then \( \exists s \in \mathcal{L}(H) \) such that

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

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

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

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 controllers with no communication is characterized by the co-observability property [18] which disregarding marking can be written as:

\[(\forall s \in L(H)) (\forall \sigma \in \Sigma_c) : \]

\[ [s \sigma \not\in L(H)] \land [s \sigma \in L(G)] \Rightarrow \]

\[ (\exists i \in Z) \left[ P_{\Sigma_{o,i}}^{-1}(P_{\Sigma_{o,i}}(s)) \cap L(H) = \emptyset \right] \land [\sigma \in \Sigma_{c,i}], \]

from which it is apparent that permissive myopic controllers that communicate achieve strictly more languages than co-observable languages. We summarize this comparison with the following corollary.

**Corollary 3.1** Let \( L(G) \) be a specified language, and \( \Sigma_{o,i}, \Sigma_{c,i} \) be given for \( i = 1, \ldots, n \). Denote by \( C_{L,obs} \) the class of languages observable with respect to \( L(G), \cup_{i=1,\ldots,n} \Sigma_{o,i}, \cup_{i=1,\ldots,n} \Sigma_{c,i} \). Denote by \( C_{L,coops} \) the class of languages co-observable with respect to \( L(G), \Sigma_{o,i}, \Sigma_{c,i}, i = 1, \ldots, n \). Denote by \( C_{L,fsnm} \) the class of languages achievable with permissive finite-state estimate non-myopic controllers (Cf. Proof of Theorem 3.1) with respect to \( L(G), \Sigma_{o,i}, \Sigma_{c,i}, i = 1, \ldots, n \). Finally, denote by \( C_{L,mypic} \) the class of languages satisfying the properties given in Theorem 3.2 with respect to \( L(G), \Sigma_{o,i}, \Sigma_{c,i}, i = 1, \ldots, n \), i.e., languages achievable by permissive myopic controllers with total recall. Then \( C_{L,obs} \subseteq C_{L,mypic} \subseteq C_{L,fsnm} = C_{L,shb} \).

From a communication-policy synthesis viewpoint the results of this section have at least one major implication. This implication is analogous to the case in stochastic control when 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 but for controllers that anticipate future communications 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.

### 4 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.\(^7\) Optimality in discrete-event systems generally involves the coarseness of a partition [2] or the supremality/infimality of a particular set \( \Sigma \) of 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 will 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

---

\(^7\)Work has been done in [19] where this type of measure is added to the system model to derive an optimal control for discrete-event systems.
optimalit\text{y} to possess, and it is therefore useful to attempt to capture the intuitive aspects of these conditions with a mathematical characterization.

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

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

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

$\quad L^\Theta_{com} \not\subseteq L^*_{com}.$

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 $\Theta$-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) $\forall \Theta \in \mathcal{C}_P \setminus \{\Theta^*\} :$

$\quad \forall t \in L^*_{com} :$

$\quad L^\Theta_{com} \not\subseteq \left[ L^\Theta_{com} \setminus \{t\} \right] \cup \{t\} \Sigma^+,$

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

(C3) $\forall \Theta \in \mathcal{C}_P \setminus \{\Theta^*\} :$

$\quad \forall L_{pp} \subseteq L^*_{com} :$

$\quad L^\Theta_{com} \not\subseteq \left[ L^\Theta_{com} \setminus L_{pp} \right] \cup L_{pp} \Sigma^+.$

This condition says that communication cannot be postponed along any subset of traces in $L^*_{com}$.

Evidently, if a communication policy is C3-optimal, then it is also optimal with regard to C1 ($L_{pp} = \emptyset$) and C2 ($|L_{pp}| = 1$). There are of course, many conditions that a designer could specify for optimality. It is not our intention to exhaust the possibilities here, and in what follows we will restrict attention to C1-C3.\footnote{The idea of postponing communication “for as long as possible” seems intuitive; however, this may have significantly undesirable effects on robustness. Robustness of communication policies is beyond the scope of this paper.}

14
5 Finite-State Estimate Myopic Controller Information Structure and Synthesis Procedure

In this section we present an example of how the general information structure $I$ can be constrained a priori to yield specific problem instances and solutions.

5.1 Finite-State Estimates for Myopic Controllers

The specific decentralized supervisory control problem investigated here is that having two controllers with a constrained information structure; however, the discussion generalizes naturally to $n$ controllers. The information structure is constrained such that the inference maps satisfy Eqn (16). 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 communicate state estimates $\Gamma \Xi \subseteq \rho(X_{H \times G})$.

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 $I$ an estimator-structure state is a four-tuple

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

where $\sigma \in \Sigma I E_1 \subseteq \rho(X_{H \times G}) \Gamma E_2 \subseteq \rho(X_{H \times G}) \Gamma$ 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 $(\sigma, E_1, E_2, x)$ and $(\bar{\sigma}, \bar{E}_1, \bar{E}_2, \bar{x})$ will be considered equivalent if $\sigma = \bar{\sigma} \Gamma E_1 = \bar{E}_1 \Gamma E_2 = \bar{E}_2 \Gamma$ and $x = \bar{x}$. Denote by $X_E \subseteq \Sigma \times \rho(X_{H \times G}) \times \rho(X_{H \times G}) \times X_{H \times G}$ the set of all $(\sigma, E_1, E_2, x)$-tuples for which $x \in E_1$ and $x \in E_2$. The initial state of the estimator structure is

$$x_{E0} = (\epsilon, \Delta_{H \times G}\{x_{H0,\emptyset}\}, \Sigma_{\emptyset,1}^*, \Delta_{H \times G}\{x_{H0,\emptyset}\}, \Sigma_{\emptyset,2}^*, x_{H0,\emptyset}).$$

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

$$\delta_E((\sigma, E_1, E_2, x), \sigma') = (\sigma', E'_1, E'_2, x')$$

where

$$
\begin{align*}
E'_1 &= \Delta_{H \times G}(E_1, \Sigma_{\emptyset,1}^* \mathcal{P}_{\Sigma_{\emptyset,1}}(\sigma')) \\
E'_2 &= \Delta_{H \times G}(E_2, \Sigma_{\emptyset,2}^* \mathcal{P}_{\Sigma_{\emptyset,2}}(\sigma')) \\
x' &= \delta_{H \times G}(x, \sigma').
\end{align*}
$$

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.8) is determined by the operator $\text{com}$ in Eqn (19):

$$\text{com}(\sigma, E'_1, E'_2, x) = (\sigma, E''_1, E''_2, x)$$

(19)
where

\[
E_i^\# = \begin{cases} 
E_i \cap E_2' & \text{if } \sigma \in \Sigma_{o,1} \cap \Sigma_{o,2} \\
E_i \cap \Delta_H \times G(E_2', \Sigma_{n,2}(\Sigma_{o,1} \ \Sigma_{o,2})) & \text{if } \sigma \in \Sigma_{o,1} \ \Sigma_{o,2} \\
\Delta_H \times G(E_1', \Sigma_{n,1}(\Sigma_{o,2} \ \Sigma_{o,1})) \cap E_2' & \text{if } \sigma \in \Sigma_{o,2} \ \Sigma_{o,1} \\
E_i' & \text{otherwise.}
\end{cases}
\]

The \textit{com} operator of Eqn(19) 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 implicant of Eqn(19) is for notational consistency with the fact that controllers communicate following observable events and it allows \textit{com} to be completely defined.

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

1. following an event use Eqn(18) to derive \(E_i')\;\text{and}
2. following a two-way communication use Eqn(19) 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 \((\sigma, E_i(\sigma), E_2(\sigma), x(\sigma))\) be the estimator-structure state reached following \(t \in T\) where \(\sigma = T(t)\;\text{and let } t_i = \pi_i(t)\) be the trajectory observed by Controller \(i\) along \(t\). The inference map \(\Psi_i\) that will be used for this section is

\[
\Psi_i(t_i) = \{\hat{s} \in L(H \times G) | \delta_H \times G(x_{H0,GO}, \hat{s}) \in \Delta_H \times G(E_i(\sigma), \Sigma^*_{n,1})\}.
\]

The myopic nature of this type of state-estimate update is evidenced by the use of \(\Sigma^*_{n,1}\) in the \(L_\Delta\) language arguments of the \(\Delta_H \times G\) operation. Controllers that anticipate future communications would have specific traces removed from the individual \(L_\Delta\) languages at each state in the estimator structure; the traces removed from \(L_\Delta\) would correspond to those indicated by the anticipation of future communications. Clearly the \textit{com} operator described in Eqn(19) is not the only possible choice for indicating how communication affects state-estimates.

The inference map of Eqn(20) is not only myopic as described above but due to its dependence on finite state estimates it is also \textit{forgetful} in the sense that the trajectory \(t_i\) is not identified - the set of trajectories leading to the same state estimate is identified. Certainly more complex inference maps can be constructed e.g., constructing the \(E_i\) to be non-myopic by basing the state-update rules on some finite-length “most recent” record of \(t_i\); more complex \textit{com} operator\footnote{In general, \textit{com} should be based on the global set of trajectories, \(T\), and not just the “local” information available at each controller as is done in Eqn(19); however, synthesis would then suffer from a lack of separation as discussed at the end of Section 3.} \textit{etc.}; however it is not our intention to exhaust the possibilities here.

\[
\n\]
5.2 Control and Communication

The control policy \( \Gamma = \{ \Gamma_1, \Gamma_2 \} \) presented here is based on the “pass the buck” constructs in \cite{18} \( \Gamma \) that is following trajectory \( t_i \); Controller \( i \) is allowed to disable an event \( \sigma_i \in \Sigma_i \) 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 \( t_i \in T_i \):

\[
\Gamma_i(\Psi_i(t_i)) = \{ \sigma_i \in \Sigma_i | \forall s \in \Psi_i(t_i), [s \sigma_i \in \mathcal{L}(G) \Rightarrow s \sigma_i \notin \mathcal{L}(H \times G)] \}.
\]  

(21)

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 \( \mathcal{E} \) be the estimator structure for a two-controller system as described in Section 5.1. A conflict state of estimator structure \( \mathcal{E} \) is a state \((\sigma, E_1, E_2, x)\) for which there exists \( \sigma_i \in \Sigma_i \) that must be disabled at state \( x \in \mathcal{X}_{H \times G} \) (for the sake of legality) and \( \forall i \in Z \) for which \( \sigma_i \in \Sigma_i \), \( \exists \tilde{x} \in \Delta_H \times G(E_i, \Sigma_{\text{act}}) \) such that \( \delta_{H \times G}(\tilde{x}, \sigma_i) \) 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 but none of the controllers has enough information to determine that the event must be disabled and so 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 co-observable (with respect to \( \mathcal{L}(G) \) all states and all \( \Sigma_i \)) then \( \mathcal{E} \) has no conflict states.

Let \( X_{\text{conf}} \) be the set of all conflict states in \( \mathcal{E} \) and define the subset of estimator structure states \( X_\Omega \) as:

\[
X_\Omega = \{ (\sigma, E_1, E_2, x) \in \mathcal{E} | \text{com}(\sigma, E_1, E_2, x) \in X_{\text{conf}} \}.
\]

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

\[
X_\Omega^k = \{ (\sigma, E_1, E_2, x) \in \mathcal{E} | (\sigma, E_1, E_2, x) \in X_\Omega^{k-1} \text{ For } \exists \sigma' \in \Sigma \text{ such that } \delta_{\text{com}}(\sigma, E_1, E_2, x), \sigma' \in X_\Omega^{k-1} \}.
\]

The set \( X_\Omega^k \) is constructed by first including all estimator states that are in \( X_\Omega^{k-1} \). Secondly all estimator-structure states are included in \( X_\Omega^k \) for which communication at those states cannot guarantee that the system will not transition in one step to a state in \( X_\Omega^{k-1} \). The sequence \( \{ X_\Omega^k \} \) converges by the monotonicity of the cardinality of \( X_\Omega^k \) and the boundedness of \( X_\Omega \). The set characterizes all states of the estimator structure (not necessarily reachable from \( x_{E_0} \)) 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 \( X_\Omega \), there exists a continuation trace which violates the desired behavior regardless of the amount of communication along that trace.

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

\[
\text{Pre}_A(X) = \{ x \in X_A | \exists \sigma \in \Sigma_A \text{ such that } \delta_A(x, \sigma) \in X \}.
\]
We define the boundary of $X_\Omega \Gamma$ denoted $Bdry(X_\Omega \Gamma)$ as the set of states $(\sigma, E_1, E_2, x)$ which have a transition into $X_\Omega$ 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 $X_\Omega^0$. We will refer to the procedure just described for constructing $X_\Omega$ as the “$X_\Omega$-Procedure” in the sequel.

Note that due to fourth implicit of Eqn(19) Tevery 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 $X_\Omega$. Because $\sigma \in \Sigma_o$ is guaranteed on the boundary of $X_\Omega \Gamma$ by Assumption A.5 we can allow the controllers to “wait” until the boundary of $X_\Omega$ is reached; then at the boundary at least one controller will observe the transition to the boundary and communicating state estimates will guarantee that $X_\Omega$ will be avoided. The definition of $X_\Omega$ can be used to show that $x_{E_0} \notin X_\Omega$ is necessary and sufficient for $L(\Pi)$ to be exactly achievable under the finite-state estimate myopic scheme described. This test for $x_{E_0} \notin X_\Omega$ 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 $t \in T$ with $s = T(t) \Gamma$ and $\sigma \in \Sigma_o$:

$$\Theta_1(\Psi_1(\pi_1(t))\sigma) = \begin{cases} E_1(s\sigma) & \text{if } \sigma \in \Sigma_{o1} \text{ and } \exists E_2, x \text{ such that } (\sigma, E_1(s\sigma), E_2, x) \in Bdry(X_\Omega), \\
& \text{or } (\sigma, E_1(s\sigma), E_2, x) \in X_{conf} \setminus X_\Omega^0, \\
& \text{or a communication is received from other controller (A,8);} \\
& \text{otherwise } \Gamma \end{cases} \quad (22)$$

$$\Theta_2(\Psi_2(\pi_2(t))\sigma) = \begin{cases} E_2(s\sigma) & \text{if } \sigma \in \Sigma_{o2} \text{ and } \exists E_1, x \text{ such that } (\sigma, E_1, E_2(s\sigma), x) \in Bdry(X_\Omega), \\
& \text{or } (\sigma, E_1, E_2(s\sigma), x) \in X_{conf} \setminus X_\Omega^0, \\
& \text{or a communication is received from other controller (A,8);} \\
& \text{otherwise } \Gamma \end{cases} \quad (23)$$

Note that Rule R1 utilizes two-way broadcast mentioned in Section 3. The notation “$\Psi_i(\cdot)\sigma$” in the argument of each $\Theta_i$ means the following: given a controller’s previous estimate $\Psi_i(\cdot)$ and the new observation $\sigma$ the controller decides instantly whether to initiate a communication or not. Note also that technically $\Theta_i$ must be defined inductively starting with the empty trace. This is because of the use of trajectories $t$ in the argument of $\Theta_i \Gamma$ etc. A trajectory of length $k$ cannot only be defined if each $\Theta_i$ has been defined for all prefixes of that trajectory with lengths less than $k$. In general, this would present a problem since $\Psi_i(\pi_i(t))$ will usually depend on trajectories which are continuations of $t$ which cannot be known without first knowing $\Theta_i$ 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 $X_\Omega$ 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 $t \in T$ with $s = T(t)$ and $\sigma \in \Sigma_c$:

$$
\Theta_1(\Psi_1(\pi_1(t))\sigma) = \begin{cases} 
E_1(s\sigma) & \text{if } \sigma \in \Sigma_{o_1} \text{ and } [\Theta_1(\Psi_1(\pi_1(t))\sigma) = \emptyset] \Rightarrow \exists E_2, x \text{ such that } \\
[(\sigma, E_1(s\sigma), E_2, x) \in \Delta_E(\{x_{E_0}\}, \Psi_1(\pi_1(t))\sigma) \cap Bdry(X_\Omega)]' \\
[(\sigma, E_1(s\sigma), E_2, x) \in \Delta_E(\{x_{E_0}\}, \Psi_1(\pi_1(t))\sigma) \cap X_{conf} \setminus X^0_{\Omega}], (24) \\
\emptyset & \text{otherwise} \\
\end{cases}
$$

$$
\Theta_2(\Psi_2(\pi_2(t))\sigma) = \begin{cases} 
E_2(s\sigma) & \text{if } \sigma \in \Sigma_{o_2} \text{ and } [\Theta_2(\Psi_2(\pi_2(t))\sigma) = \emptyset] \Rightarrow \exists E_1, x \text{ such that } \\
[(\sigma, E_1, E_2(s\sigma), x) \in \Delta_E(\{x_{E_0}\}, \Psi_2(\pi_2(t))\sigma) \cap Bdry(X_\Omega)]' \\
[(\sigma, E_1, E_2(s\sigma), x) \in \Delta_E(\{x_{E_0}\}, \Psi_2(\pi_2(t))\sigma) \cap X_{conf} \setminus X^0_{\Omega}], (25) \\
\emptyset & \text{otherwise} \\
\end{cases}
$$

The difference between Rules R1 and R1* is the following. 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 it this performance comes at a price. Reachability in the estimator structure cannot be determined a priori to the synthesis of the $\Theta_i$ maps and reachability may change during the synthesis of $\Theta_i$ maps. Thus for Rule R1* in general an iterative procedure must be used to synthesize each $\Theta_i$ and the resulting maps are not necessarily unique despite the uniqueness of $X_\Omega$. 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 5.4.

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

5.3 Example

The example presented here utilizes the myopic inference maps of Eqn (20) 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. 1 where it is assumed that the event $\gamma_1$ is possible in the plant at state 7 but is disabled in the desired behavior. It is assumed that $\Sigma_{o_1} = \{\alpha_1, \beta_1, \gamma_1\} \Gamma \Sigma_{c_1} = \{\gamma_1\} \Gamma \Sigma_{o_2} = \Sigma_{c_2} = \{\alpha_2\}$. The key feature of this example is the interleaving of $\alpha_1$ and $\alpha_2$. To begin the solution procedure the finite estimator structure is constructed without communication. This facilitates the identification of reachable conflict states $\Gamma$ the avoidance of which motivates communication. The estimator structure constructed using Eqn(18) is shown in Fig. 2. The states of the estimator structure are named $al b f c d e f f g$ and $h$ for reference. Examining the estimator structure constructed with no communication between the controllers it is apparent that state $g$ is a conflict state. The conflict arises because the event $\gamma_1$ 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 $\gamma_1$ be enabled; hence Controller 1 enables $\gamma_1$ by default on estimator-structure state $g$.

In accordance with the $X_\Omega$-Procedure the set $X^0_{\Omega}$ of all conflict states that cannot be immediately resolved using communication needs to be determined. Because we are only interested in
Figure 1: Desired behavior to be centrally implemented.

Figure 2: Estimator structure without communication.
reachable conflict states. Our attention is focused on state $g$ in the estimator-structure. Communication at state $g$ follows the event $\beta_1$ which is observable only to Controller 1; hence the second implication of Eqn(19) applies. Controller 2 upon receiving a communication from Controller 1 recognizes that an observable event has occurred (by Assumption A.8) that was not observed by itself and Controller 2 utilizes the second implication of the $\text{com}$ operator to determine its new state estimate:

\[
E''_2 = E'_1 \cap \Delta H \times G (E'_2, \Sigma^*_{\omega_2} (\Sigma_{\omega_1} \setminus \Sigma_{\omega_2}))
\]

\[
= E'_1 \cap \Delta H \times G (\{2, 4, 5, 6, 7, 8\}, \Sigma^*_{\omega_2} (\alpha_1, \beta_1, \gamma_1))
\]

\[
= E'_1 \cap \{4, 6, 7\}
\]

\[
= \{6, 7\}.
\]

Controller 1 carries out a similar calculation arriving at $E''_1 = \{6, 7\}$; hence Controller 1 still enables event $\gamma_1$. Thus estimator-structure state $g$ is a member of $X^0_{\Theta^*}$. Given that the conflict at state $g$ cannot be immediately resolved using communication, the preimage of $g$ is examined for inclusion in the set $X^1_{\Theta^*}$. 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 $e$

\[
\text{com}(\alpha_2, \{3, 4, 5\}, \{2, 5\}, 5) = (\alpha_2, \{5\}, \{5\}, 5),
\]

which following the event $\beta_1$ 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 communication at state $e$ following the observation of $\alpha_2$; however state $b$ is indistinguishable from state $e$ (modulo Controller 2's inference map) indicating that Controller 2 must initiate communication 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. 3 where doubled state boxes indicate the occurrence of communication (left=before communication; right=after communication). There are no conflict states in Fig. 3 and the set of system trajectories represented by Fig. 3 is given in Table 1.

### 5.4 Myopic Controllers, Optimality and Non-Uniqueness

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

To facilitate the proof, define the following set:

\[
E^\Theta_{i, \text{com}} = \{(\sigma, E_i(s\sigma)) \in \Sigma_{\omega_i} \times \rho(X_{H \times G}) | s\sigma \in \mathcal{L}(Z/G), \exists E_1, \ldots, E_i, E_{i+1}, \ldots, E_n, x \text{ such that } \text{Controller } i \text{ initiates communication at } (\sigma, E_1, \ldots, E_i(s\sigma), \ldots, E_n, x) \text{ under policy } \Theta_i\}
\]

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

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

\[
E_i(s\sigma) = E_i(s'\sigma) \Rightarrow (\sigma, E_i(s'\sigma)) \notin E^\Theta_{i, \text{com}}
\]

\[
\Rightarrow s'\sigma \notin \mathcal{L}^\Theta_{\text{com}}.
\]
Figure 3: Estimator structure with communication.

<table>
<thead>
<tr>
<th>$T$</th>
<th>$\pi_1(T)$</th>
<th>$\pi_2(T)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\varepsilon$</td>
<td>$\varepsilon$</td>
<td>$\varepsilon$</td>
</tr>
<tr>
<td>$\alpha_2$</td>
<td>${1, 2}$</td>
<td>$\varepsilon$</td>
</tr>
<tr>
<td>${2, 5}$</td>
<td>$\varepsilon$</td>
<td>$\alpha_2$</td>
</tr>
<tr>
<td>$\alpha_2$</td>
<td>${1, 2}$</td>
<td>$\alpha_1$</td>
</tr>
<tr>
<td>${2, 5}$</td>
<td>$\varepsilon$</td>
<td>$\alpha_1$</td>
</tr>
<tr>
<td>$\alpha_2$</td>
<td>${1, 2}$</td>
<td>$\alpha_1\beta_1$</td>
</tr>
<tr>
<td>${2, 5}$</td>
<td>$\varepsilon$</td>
<td>$\alpha_1\beta_1$</td>
</tr>
<tr>
<td>$\alpha_1\beta_1\gamma_1$</td>
<td>$\varepsilon$</td>
<td>$\alpha_1\beta_1\gamma_1$</td>
</tr>
<tr>
<td>$\alpha_1$</td>
<td>$\alpha_1$</td>
<td>$\varepsilon$</td>
</tr>
<tr>
<td>$\alpha_1\alpha_2$</td>
<td>${3, 4, 5}$</td>
<td>$\alpha_1$</td>
</tr>
<tr>
<td>${2, 5}$</td>
<td>$\varepsilon$</td>
<td>$\alpha_1$</td>
</tr>
<tr>
<td>$\alpha_1\alpha_2$</td>
<td>${3, 4, 5}$</td>
<td>$\beta_1$</td>
</tr>
<tr>
<td>${2, 5}$</td>
<td>$\varepsilon$</td>
<td>$\beta_1$</td>
</tr>
<tr>
<td>$\alpha_1\alpha_2$</td>
<td>${3, 4, 5}$</td>
<td>$\beta_1$</td>
</tr>
<tr>
<td>${2, 5}$</td>
<td>$\varepsilon$</td>
<td>$\beta_1$</td>
</tr>
<tr>
<td>$\alpha_1\alpha_2$</td>
<td>${3, 4, 5}$</td>
<td>$\varepsilon$</td>
</tr>
<tr>
<td>${2, 5}$</td>
<td>$\varepsilon$</td>
<td>$\varepsilon$</td>
</tr>
</tbody>
</table>

Table 1: Trajectories represented by Fig. 3
Proof of Lemma 5.1 Implication (26) follows by contradiction: if \( \exists i \in Z \) such that \( E_i(s\sigma) = E_i(s'\sigma) \) and \( (\sigma, E_i(s\sigma)) \in E_i^\theta_{com} \), then \( s\sigma \in L^\theta_{com} \) which contradicts \( s\sigma \notin L^\theta_{com} \). Implication (27) follows from the definition of \( E_i^\theta_{com} \).

Lemma 5.1 simply states that if \( s\sigma \) does not lead to a communication \( \Gamma \) then if \( s'\sigma \) does lead to a communication \( s'\sigma \) 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 \( \Theta^* = \{ \Theta_1^*, \Theta_2^*, \ldots, \Theta_n^* \} \) be consistent with the \( X_\Omega \)-Procedure and Rule R1*; then \( \Theta^* \) is C1-optimal over the class of myopic controllers that maintain and communicate finite state estimates as described in Eqn(18) and Eqn(19).

**Proof of Theorem 5.1** (By contradiction) Suppose \( \exists \Theta \in C^P_\Theta \) (defined in Section 4) such that \( L^\theta_{com} \subset L^\theta_{com} \) where \( s\sigma \in L^\theta_{com} \) but \( s\sigma \notin L^\theta_{com} \), 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 \( s\sigma \notin L^\theta_{com} \) so the system state is in \( Bdry(X_\Omega) \cup [X_{conf} \setminus X^0_{\Omega}] \) but no communication occurs. By the construction of \( X_\Omega \) without communication at states in \( Bdry(X_\Omega) \) it cannot be guaranteed that the system does not reach a state in \( X_\Omega \) 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 in \( X_{conf} \setminus X^0_{\Omega} \) implies that the correct control decision is not guaranteed. It follows that \( \Theta \) cannot solve Problem (P).

Cases (i) and (ii) both contradict the assertion that \( \Theta \in C^P_\Theta \).

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 \( Bdry(X_\Omega) \cup [X_{conf} \setminus X^0_{\Omega}] \) which is indistinguishable from the current state \( \Gamma \) while Rule R1 is based only on the fact that there is a state in \( Bdry(X_\Omega) \cup [X_{conf} \setminus X^0_{\Omega}] \) that is indistinguishable from the current state regardless of whether that state is actually reachable. The qualification of using Eqn(18) and Eqn(19) 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 \( X_\Omega \). Communication policies based on Rule R1* and \( Bdry(X_\Omega) \cup [X_{conf} \setminus X^0_{\Omega}] \) 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 \( Bdry(X_\Omega) \cup [X_{conf} \setminus X^0_{\Omega}] \) 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 \( X_\Omega \Gamma Bdry(X_\Omega) \) 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 \( X_\Omega \) and \( Bdry(X_\Omega) \cup [X_{conf} \setminus X^0_{\Omega}] \). Indeed R1* itself is not necessarily associated 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 $L(H) \subseteq L(G)$ is controllable with respect to $L(G)$ and $\Sigma_{uc}$ and observable with respect to $L(G)$, $\Sigma_{o}$ and $\Sigma_{e}$, 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$1$ that is$1$ we need only give a counter-example to a uniqueness hypothesis. Consider the desired behavior shown in Fig. 4 where$1$ again$1$ implies that the event $\gamma_1$ is defined in the plant but must be disabled. Assume two controllers will be used with $\Sigma_{o,1} = \{\alpha_1, \beta_1, \gamma_1\}$ $\Gamma \Sigma_{e,1} = \{\gamma_1\}$ $\Gamma \Sigma_{o,2} = \{\alpha_2, \gamma_2\}$ and $\Sigma_{e,2} = \{\gamma_2\}$. For the desired behavior shown in Fig. 4 the class of myopic controllers supports multiple optimal solutions to Problem (P). Two $C1$-$C3$-optimal communication policies for the desired behavior of Fig. 4 are shown in Fig. 5. For one myopic controller solution$1$ Controller 1 communicates its state estimate following $\alpha_1$. Similarly$1$ the other optimal solution has Controller 2 communicating following $\alpha_2$. These two solutions can be interpreted as first determining the set $X_{\Omega}$ as discussed earlier$1$ then the communication maps are generated in a specific order. The first optimal communication policy in Fig. 5 results from building $\Theta_1$ first$1$ and because the reachable subset of $Bdry(X_{\Omega}) \cup [X_{conf} \setminus X^\prime_{\Omega}]$ is altered$1$ $\Theta_2$ is then designed to only respond to Controller 1’s communications. Thus$1$ the synthesis of $\Theta_2$ is conditioned on the design of $\Theta_1$. Vice versa for the second optimal communication map. Both policies in Fig. 5 satisfy $R1*$ and for comparison$1$ the policy determined by Rule $R1$ is shown in Fig. 6. Thus$1$ the fact that the set $X_{\Omega}$ is unique does not imply there are unique $C1$-$C3$-optimal communication policies. Note that for this example$1$ if the controllers were allowed to be non-myopic$1$ then the communication policies represented in Fig. 5 would still be $C1$-$C3$-optimal; however$1$ the state-estimates within each box would be more refined due to non-myopia.

5.5 Discussion

It was discussed in Sections 2 and 3 that the synthesis of communication and control policies is difficult due to the lack of separation between estimation$1$ communication and control. In Section 5$1$ 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
Figure 5: Two C1-C3-optimal communication policies for Fig. 4

Figure 6: Communication policy for Fig. 4 resulting from Rule R1
controllers occurs because each \( L_\Delta \) is known \textit{a priori} to be \( \Sigma_{n,i} \); thus ignorance of future potential communications is forced into the structure of the controllers so that a simple synthesis algorithm (\( X_\Omega \)-Procedure and Rule \( \text{R1} \)) may be used. Unfortunately the synthesis of optimal communication policies (such as those consistent with \( \text{R1}^* \)) is complicated by the fact that the information used to determine if a communication is required (e.g., the reachable part of \( Bdry(X_\Omega) \cup [X_{conf} \setminus X_\Omega^0] \)) is affected by both the decision to communicate at the present state and by similar decisions at other states in the system. The nature of Problem (\( P \)) itself (where \( L(H) \) is to be matched exactly) provides a separation between estimation and control greatly simplifying synthesis.

Of course a decentralized supervisory control scheme using the myopic and forgetful inference maps presented in Section 5.2 will not in general allow arbitrary languages \( L(H) \) to be achievable under control with communication. For these 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 \( X_\Omega \) if \( x_{EO} \in X_\Omega \) then the initial state of the system is such that no amount of communication using the described \( \Psi \) and \( \Theta \) can prevent the system from reaching a conflict state. The definition of \( X_\Omega \) can be used to show that \( x_{EO} \not\in X_\Omega \) is necessary and sufficient for \( L(H) \) to be exactly achievable under the 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 \( X_\Omega \)-Procedure described above was presented as generating the set \( X_\Omega \) 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 \( X_\Omega \) 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 \textit{a priori}.

The utility of the \( X_\Omega \)-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 short comings 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 communication policies (even those resulting from the unique \( X_\Omega \)) holds as stated in Theorem 5.2.

6 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 \( \Psi \); 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 non-uniqueness of optimal solutions was demonstrated.

**Acknowledgments**

The authors wish to thank Feng Lin, Ard Overkamp, Karen Rudie, Jan H. van Schuppen, and Demosthenis Teneketzis for stimulating discussions on the topics of decentralized information and decentralized control with communication. It should also be noted that Figure 1 (albeit simple) arose from (and generated much) group discussion with many of the above-mentioned individuals. Special thanks go to Karen Rudie for pointing out an error in an earlier version of the alternate expression of co-observability in Appendix A.

**A Alternative Expressions of Observability and Co-Observability**

The material here represents an attempt to simplify the descriptions and comparisons of language observability and co-observability to other language classifications. The definitions given below are not new; they are simply the “classical” definitions rewritten for simplification. These definitions prove useful in Section 3 for comparing languages achievable by different controller types. (Yet another way of expressing co-observability can be found in [17T Proposition 3].)

**Proposition A.1** Let $H$ and $G$ be the specification and plant automata, respectively. The language $L(H)$ is observable with respect to $L(G)$, $\Sigma_o$ and $\Sigma_c$ iff

$$\forall s \in L(H) (\forall \sigma \in \Sigma_c) : [s \sigma \notin L(H)] \land [s \sigma \in L(G)] \Rightarrow [P^{-1}(P(s)) \sigma \cap L(H) = \emptyset].$$

**Proof of Proposition A.1** Recall the “classical” definition of observability:

a language $L(H)$ is observable with respect to $L(G) \Gamma \Sigma_o \Gamma$ and $\Sigma_c$ if $(\forall s, s' \in \Sigma^*) (\forall \sigma \in \Sigma_c) :$

$$P(s) = P(s') \Rightarrow [(\forall \sigma \in \Sigma_c) [s' \sigma \in L(H)] \land [s \in L(H)] \land [s \sigma \in L(G)] \Rightarrow s \sigma \in L(H)].$$

(28)

Note that the “nextactK” relation that is generally associated with the definition of observability is included in the expression of Eqn(28). Without loss of generality we may restrict attention to $s, s' \in L(H) \subseteq L(G) \Gamma$ so we rewrite Eqn(28) as

$$\forall s, s' \in L(H) (\forall \sigma \in \Sigma_c) :$$

$$P(s) = P(s') \Rightarrow [[s' \sigma \in L(H)] \land [s \sigma \in L(G)] \Rightarrow s \sigma \in L(H)],$$

(29)

which may be expanded to yield

$$\forall s, s' \in L(H) (\forall \sigma \in \Sigma_c) :$$

$$[s \sigma \in L(H)] \lor [s \sigma \notin L(G)] \lor [P(s) \neq P(s')] \lor [s' \sigma \notin L(H)].$$

(30)
Given sets of observable and controllable events for each supervisory controller in the following way: for observability in the decentralized sense, at least one controller must have both the “centralized knowledge” that an event must be disabled and the control authority over an event that must be disabled. From this description we have the following re-write of the definition of observability for decentralized systems under the assumption that \( \cup_{i \in Z} \Sigma_{c, i} = \Sigma_c \):

\[
(\forall s \in \mathcal{L}(H))(\forall \sigma \in \Sigma_c) :
[s \sigma \not\in \mathcal{L}(H)] \land [s \sigma \in \mathcal{L}(G)] \Rightarrow [\mathcal{P}(s) = \mathcal{P}(s')] \land [s' \sigma \in \mathcal{L}(H)] ,
\]

(31)

which is directly equivalent to the alternative definition:

\[
(\forall s \in \mathcal{L}(H))(\forall \sigma \in \Sigma_c) :
[s \sigma \not\in \mathcal{L}(H)] \land [s \sigma \in \mathcal{L}(G)] \Rightarrow [\mathcal{P}^{-1}(\mathcal{P}(s)) \sigma \cap \mathcal{L}(H) = \emptyset] .
\]

(32)

In Section 3 observability is used in the characterization of languages achievable by controllers that expect future (potential) communications in decentralized systems. For these decentralized systems the observability condition’s requirement of command authority over event in \( \Sigma_c \) must be distributed over the set of all controllers. Thus for observability in the decentralized sense at least one controller must have both the “centralized knowledge” that an event must be disabled and the control authority over an event that must be disabled. From this description we have the following re-write of the definition of observability for decentralized systems under the assumption that \( \cup_{i \in Z} \Sigma_{c, i} = \Sigma_c \):

\[
(\forall s \in \mathcal{L}(H))(\forall \sigma \in \Sigma_c) :
[s \sigma \not\in \mathcal{L}(H)] \land [s \sigma \in \mathcal{L}(G)] \Rightarrow (\exists i \in Z) [\mathcal{P}^{-1}(\mathcal{P}(s)) \sigma \cap \mathcal{L}(H) = \emptyset] \land [\sigma \in \Sigma_{c, i}] .
\]

(33)

Under the assumption that \( \cup_{i \in Z} \Sigma_{c, i} = \Sigma_c \) Eqn(33) is equivalent to Eqn(32).

**Proposition A.2** Let \( Z = \{1, \ldots, n\} \) 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 co-observable with respect to \( \mathcal{L}(G) \), \( \Sigma_{c, 1}, \ldots, \Sigma_{c, n} \) and \( \Sigma_{c, 1}, \ldots, \Sigma_{c, n} \) iff:

\[
(\forall s \in \mathcal{L}(H))(\forall \sigma \in \Sigma_c) = \cup_{i \in Z} \Sigma_{c, i} : [s \sigma \not\in \mathcal{L}(H)] \land [s \sigma \in \mathcal{L}(G)] \Rightarrow
(\exists i \in Z) [\mathcal{P}^{-1}(\mathcal{P}(s)) \sigma \cap \mathcal{L}(H) = \emptyset] \land [\sigma \in \Sigma_{c, i}] .
\]

**Proof of Proposition A.2** Recall the definition of the nextact relation:

\[
(\forall \sigma \in \Sigma)(\forall s, s' \in \Sigma^*) (s, \sigma, s') \in nextact_K \text{ if }

[s' \sigma \in \overline{K}] \land [s \sigma \in \overline{K}] \land [s \sigma \in \mathcal{L}(G)] \Rightarrow [s \sigma \in \overline{K}] .
\]

(34)

We will immediately rewrite Eqn(34) for the purposes of control to achieve language \( K \) in the following way: \( (\forall \sigma \in \Sigma)(\forall s, s' \in \overline{K}) (s, \sigma, s') \in nextact_K \text{ if }\)

\[
[s \sigma \not\in \overline{K}] \land [s \sigma \in \mathcal{L}(G)] \Rightarrow [s \sigma \not\in \overline{K}] .
\]

(35)

Given sets of observable and controllable events for each supervisory controller in \( Z = \{1, \ldots, 2\} \) a language \( \mathcal{L}(H) \subset \mathcal{L}(G) \) is co-observable with respect to \( \mathcal{L}(G) \) \( \Gamma(\Sigma_{c, 1} \Sigma_{c, 2}) \Gamma(\Sigma_{c, 1} \Sigma_{c, 1}) \) if [18]

\[
(\forall s, s', s'' \in \Sigma^*)
\mathcal{P}_{\Sigma_{c, 1}}(s) = \mathcal{P}_{\Sigma_{c, 1}}(s') \land \mathcal{P}_{\Sigma_{c, 2}}(s) = \mathcal{P}_{\Sigma_{c, 2}}(s'') \Rightarrow
(\forall \sigma \in \Sigma_{c, 1} \cap \Sigma_{c, 2}) \exists (s, \sigma, s') \in nextact_{\mathcal{L}(H)} \land
(\forall \sigma \in \Sigma_{c, 1} \setminus \Sigma_{c, 2}) \exists (s, \sigma, s'') \in nextact_{\mathcal{L}(H)} \land
(\forall \sigma \in \Sigma_{c, 2} \setminus \Sigma_{c, 1}) \exists (s, \sigma, s'') \in nextact_{\mathcal{L}(H)} ,
\]

(36)
where *marking* will be considered separately. Note that while co-observability naturally generalizes to *n* supervisory controllers the form of Eqn(36) does not allow the generalized definition of co-observability to be easily written. The difficulty results from having a separate “test” string I e.g., $\Gamma d'$, $s'H$ for each controller and from having to write out the conjuncts for every possible set of intersections of controllable events that the event $\sigma$ could exist. Indeed the form of co-observability found in [17T Proposition 3] does not generalize easily for the same reasons. The “new” form of the definition of co-observability does not suffer this difficulty.

Equation (36) can be written as 

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

$$\mathcal{P}_{\Sigma_{c,1}}(s) = \mathcal{P}_{\Sigma_{c,1}}(s') \land \mathcal{P}_{\Sigma_{c,2}}(s) = \mathcal{P}_{\Sigma_{c,2}}(s'') \Rightarrow$$

$$\left[ \left[ (s, \sigma, s') \in nextact_L[H] \lor (s, \sigma, s'') \in nextact_L[H] \right] \land [\sigma \in \Sigma_{c,1} \cap \Sigma_{c,2}] \right] \lor$$

$$\left[ (s, \sigma, s') \in nextact_L[H] \land [\sigma \in \Sigma_{c,1} \setminus \Sigma_{c,2}] \right] \lor$$

$$\left[ (s, \sigma, s'') \in nextact_L[H] \land [\sigma \in \Sigma_{c,2} \setminus \Sigma_{c,1}] \right],$$

which further simplifies to 

$$ (\forall s, s', s'' \in L(H)) \left( \forall \sigma \in \Sigma_c \right)$$

$$\mathcal{P}_{\Sigma_{c,1}}(s) = \mathcal{P}_{\Sigma_{c,1}}(s') \land \mathcal{P}_{\Sigma_{c,2}}(s) = \mathcal{P}_{\Sigma_{c,2}}(s'') \Rightarrow$$

$$\left[ (s, \sigma, s') \in nextact_L[H] \land [\sigma \in \Sigma_{c,1}] \right] \lor$$

$$\left[ (s, \sigma, s'') \in nextact_L[H] \land [\sigma \in \Sigma_{c,1}] \right].$$

Expanding Eqn(38) using Eqn(35) eliminates explicit mention of *nextact* and yields: 

$$ (\forall s, s', s'' \in L(H)) \left( \forall \sigma \in \Sigma_c \right)$$

$$\left[ \mathcal{P}_{\Sigma_{c,1}}(s) \neq \mathcal{P}_{\Sigma_{c,1}}(s') \right] \lor \left[ [s \sigma \in L(H) \lor s \sigma \notin L(G) \lor s' \sigma \notin L(H)] \land [\sigma \in \Sigma_{c,1}] \land$$

$$\left[ [s \sigma \in L(H) \lor s \sigma \notin L(G) \lor s' \sigma \notin L(H)] \land [\sigma \in \Sigma_{c,2}] \right],$$

which simplifies using (i) the fact that $\mathcal{P}_{\Sigma_{c,1}}(t) \neq \mathcal{P}_{\Sigma_{c,1}}(t')$ iff $t' \notin \mathcal{P}_{\Sigma_{c,1}}(\mathcal{P}_{\Sigma_{c,1}}(t))$ and (ii) DeMorgan’s laws to

$$ (\forall s \in L(H)) \left( \forall \sigma \in \Sigma_c \right)$$

$$[s \sigma \notin L(H)] \land [s \sigma \in L(G)] \Rightarrow$$

$$\left[ \mathcal{P}_{\Sigma_{c,1}}^{-1}(\mathcal{P}_{\Sigma_{c,1}}(s)) \sigma \in L(H) = \emptyset \right] \land [\sigma \in \Sigma_{c,1}] \lor$$

$$\left[ \mathcal{P}_{\Sigma_{c,2}}^{-1}(\mathcal{P}_{\Sigma_{c,2}}(s)) \sigma \in L(H) = \emptyset \right] \land [\sigma \in \Sigma_{c,2}],$$

and Eqn(40) 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 co-observability:

$$ (\forall s \in L(H))$$

$$[s \notin L_m(H)] \Rightarrow (\exists i \in Z) \left[ \mathcal{P}_{\Sigma_c}^{-1}(\mathcal{P}_{\Sigma_{c,i}}(s)) \cap L_m(H) = \emptyset \right].$$

(41)
References


