# Know Means No: Incorporating Knowledge into Discrete-Event Control Systems

S. Laurie Ricker, *Member, IEEE* and Karen Rudie, *Member, IEEE*

***Abstract*—Modal logic is introduced into the modeling of discrete-event systems. Analysis within this framework includes formal reasoning about what supervisors know or do not know about a given system. This model can be used to develop control strategies that solve decentralized discrete-event control problems. When a problem cannot be solved using fully decentralized supervisors, reasoning about knowledge may provide guidelines for incorporating communication and pooled information into the model.**

***Index Terms*—Automata, discrete-event systems, modal logic.**

## I. Introduction

A DISCRETE-EVENT system (DES) is a set of sequences of events that describes the behavior of a physical process. A change in the system state of the process is precipitated by the occurrence of an action or event and is not time-driven. A discrete-event control problem arises when we want to restrict the system to performing a specified subset of overall system behavior. A solution to a discrete-event control problem exists when we can construct an overseer (or *supervisor*) to achieve the desirable (or *legal*) behavior by either preventing some events from taking place (*disabling* an event) or allowing—but not forcing—others to occur (*enabling* an event).

Decentralized discrete-event problems originate when more than one supervisor is required to ensure that the system avoids illegal behavior. In this class of problems, no one supervisor has a complete view of the system behavior. The supervisors must coordinate, without communication, the disabling and enabling of events to realize the legal behavior. In other words, each supervisor must know enough of what the system is doing to make correct decisions to turn events off or on.

Decentralized control problems arise naturally in distributed systems. For instance, in telecommunication problems a sender and receiver are physically separated by a communication channel, so that neither participant sees everything that occurs. The framework for decentralized discrete-event control we adopt for this work is taken from [1] and [2] and is based on the theory of formal languages. A discrete-event system is viewed as a generator of a formal language and establishing control for the system amounts to determining which sequences in the language should be recognized by each supervisor. Intrinsic to the study of these processes is the informal argument that as long as at least one supervisor *knows* the correct control action to take in preventing illegal behavior of the system, an overall control strategy may be synthesized.

In most formulations of decentralized discrete-event control problems, decisions are based solely on what each supervisor observes. A control solution cannot be constructed if, after observing some sequence of events, there is no supervisor that "knows enough" to disable a particular event. When such a stalemate is reached, it means that, in isolation, a supervisor lacks appropriate information to make the correct control decision. However, if supervisors could access their collective knowledge about the situation, thereby eliminating some of the uncertainty in making the correct control decision, it may be the case that a control strategy can be formulated.

We are interested in recasting this discrete-event control framework into knowledge theory [3] for two reasons: 1) to explain, in more intuitive terms, the structure of the system and the behavior of controllers that solve decentralized DES problems and 2) to formally reason about what supervisors need to know to solve control problems. As noted above, informal reasoning about knowledge is already an integral part of analyzing decentralized supervisory control problems. Thus it seems natural to formally consider what it means for each supervisor to "know enough."

Halpern and Moses [3] formulated a model of knowledge (based on modal logic) to analyze distributed systems. This model is based on the concept of *possible worlds*. The idea is that an *agent* (equivalent to the notion of an overseer) has only a partial view of the distributed system and may be unable to distinguish different system states from the true state of the system. An agent's knowledge of the system depends only on its local view of the system behavior. As an agent acquires more knowledge, there are fewer worlds or system states the agent considers as possible. When an agent has insufficient knowledge, the model allows us to consider the knowledge of *groups* of agents. We consider this aspect of the knowledge model to determine if combining the knowledge of two DES supervisors will produce "enough" information to reach a control solution.

The knowledge ascribed to supervisors is, by itself, insufficient to adequately capture overall system behavior since a supervisor only uses knowledge to take a particular action. The interaction between knowledge and action is expressed in a knowledge-based protocol [4]. In situations where a DES supervisor has insufficient knowledge to make a control

Manuscript received February 27, 1998; revised November 24, 1998 and October 12, 1999. Recommended by Associate Editor, L. Dai. This work was supported in part by NSERC under Grant OGP0138887.

S. L. Ricker is with CWI, Amsterdam, 1090 GB, The Netherlands.

K. Rudie is with the Department of Electrical and Computer Engineering, Queen's University, Kingston, ON K7L 1N6, Canada (e-mail: rudie@ee.queensu.ca).

Publisher Item Identifier S 0018-9286(00)06321-2.

decision, we envision actions being taken on the basis of the knowledge acquired through communication.

Reasoning about knowledge has been part of the analysis of a variety of applications in the areas of economics [5], [6], computer security [7], distributed database systems [8], robotics [9], and communication problems [10]. Temporal logic has been applied to the study of supervisory control problems [11], [12] and modal logic has been used as the basis for a computer language that simulates discrete-event processes [13]; however, a formal model of knowledge has yet to be incorporated into the study of discrete-event control problems.

In this paper we establish that standard decentralized DES concepts can be equivalently expressed in knowledge theory. In the DES framework of Ramadge and Wonham [1], supervisors determine whether any given sequence is a member of the legal language. In the knowledge theory setting, the basic variables correspond to answers to questions such as "is fact $p$ true?" or "does an agent know that fact $p$ is true?" When we apply knowledge theory to decentralized DES, supervisors determine whether they have the knowledge to disable an event. A fact $p$ in the knowledge model for DES corresponds to whether event $\sigma$ is legal at a given state of the system. We restrict our attention in this paper to reasoning about whether an agent knows to disable event $\sigma$ (based on what an agent directly observes). However, the knowledge formalism gives us a means of expressing more complex statements that could guide decision-making in reaching a control solution. For instance, we can say "agent 1 knows that agent 2 does not know to disable event $\sigma$." By characterizing the nature of knowledge in a discrete-event control system in this manner, we can investigate what additional information a supervisor needs when it does not know whether to take a disable action. Our long-term research goal is to understand how and when the dissemination of knowledge among supervisors leads to control solutions for a class of control problems that decentralized supervisory control theory does not presently address. For those problems, communication between supervisors could be a means of improving the knowledge each supervisor possesses. In addition to presenting the decentralized DES framework recast in knowledge theory, we also suggest some of the features of knowledge logic that we hope to exploit in establishing what information needs to be communicated to provide DES supervisors with enough knowledge to solve control problems.

Recently, there has been some work on incorporating communication into decentralized control: an algebraic approach [14] and models that use information structures from stochastic control [15], [16]. In addition there has been some work on the role communication plays in a distributed diagnoser [17] and an algorithm for minimal communication in distributed systems [18].

We begin by providing a brief background of DES theory and of knowledge logic. Section III introduces the knowledge model we apply to DES. Section IV contains the knowledge-based protocol that we propose for guiding the actions of decentralized agents. We then present several examples that illustrate how to introduce knowledge into the analysis of decentralized DES problems.

## II. Background and Notation

One of the difficulties in bringing together the notation from two established fields is addressing the overlap of symbols used to represent distinctly different concepts. We have tried to accommodate the more serious notational discrepancies, but also include references that can be consulted for further clarification.

### A. Discrete-Event Systems

This work adopts the framework for discrete-event systems as developed by Ramadge and Wonham [1]. A brief review of essential notation is provided in this section. More comprehensive introductions to discrete-event control theory include [19], [1], [20], [21]. References for decentralized control include [22]–[25], [2], [26], [27].

The discrete-event control theory of Ramadge and Wonham entails the system requiring control (the *plant*) to be described as a generator of a formal language (i.e., a state machine). The behavior of the plant is represented by sequences constructed from a non-empty set of symbols called an *alphabet*. The alphabet represents the set of all possible *events* that can occur within the system. Transitions from one system state to another do not depend on the passage of time, but rather, on the occurrence of an event. Also specified is a second generator: one that describes the desirable or legal behavior of the plant and therefore generates the *legal language*. The goal is to develop a control strategy for a supervisor that will constrain the behavior of the plant to that of the legal language. The supervisor averts undesirable behavior of the plant by disabling those events whose occurrence would lead to an illegal sequence.

More formally, the plant is modeled by an automaton $G = (Q^G, \Sigma, \delta^G, q_0^G)$, where $Q^G$ is a set of *states*; $\Sigma$ is the alphabet; $\delta^G$ is the *transition function*, a partial function $\delta^G$: $\Sigma \times Q^G \rightarrow Q^G$; and $q_0^G \in Q^G$ is the *initial state*. The set $\Sigma^*$ contains all possible finite strings (i.e., sequences) over $\Sigma$ plus the null string $\varepsilon$. The language generated by the plant $G$, denoted $L(G)$, is a subset of $\Sigma^*$ and is defined as follows: $L(G) := \{t | t \in \Sigma^*$ and $\delta^G(t, q_o^G)$ is defined $\}$.

For any strings $t$ and $u \in \Sigma^*$, we say that $u$ is a *prefix* of $t$ if $\exists\, v \in \Sigma^*$ such that $t = uv$. Thus every string $t \in \Sigma^*$ (where $t \neq \varepsilon$) has at least two prefixes: $\varepsilon$ and $t$. If $L \subseteq \Sigma^*$, the *prefix-closure* of $L$ is a language, denoted by $\overline{L}$, consisting of all prefixes of strings of $L$:$\overline{L} := \{u \in \Sigma^*$: $u$ is a prefix of $t\}$. Because every string is a prefix of itself, $L \subseteq \overline{L}$. A language is said to be *prefix-closed* if $L = \overline{L}$. By definition, $L(G)$ is prefix-closed.

We also describe the legal behavior of the plant as an automaton $E = (Q^E, \Sigma, \delta^E, q_0^E)$ and the legal language is denoted $L(E)$. We assume that $E$ is a subautomaton of $G$ as described in the context of supervisory control in [28] and [29]. That is, $Q^E \subseteq Q^G$, $q_0^E = q_0^G$ and $\delta^E(t, q_0^G) = \delta^G(t, q_0^G)$ for all $t \in L(E)$.

A plant is represented by a finite-state machine or a directed graph, as shown in Fig. 1, where the nodes of the graph are the states in $Q$, the arcs of the graph are the transitions defined by the function $\delta^G$, labels for the arcs are the events in $\Sigma$, and the initial state is identified by a small entry arrow. Thus for any event $\sigma \in \Sigma$ and state $q \in Q^G$, $\delta^G(\sigma, q)$ is defined (written

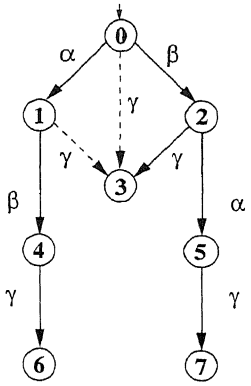


Fig. 1. A plant $G$ and its legal automaton $E$. Illegal transitions are indicated with a dashed line.

$\delta^G(\sigma, q)!$) if there is an arc labeled by $\sigma$ from $q$ to some other state. The definition for $\delta^G$ can be extended to a partial function for $\Sigma^* \times Q^G$, so that $\delta^G(t, q) = q'$, for $t \in L(G)$, means that state $q'$ can be reached from state $q$ via the sequence $t$. In Fig. 1, we can say both $\delta^G(\gamma, 4) = 6$ and $\delta^G(\alpha\beta\gamma, 0) = 6$ because event $\gamma$ is defined as a transition from state 4 to state 6 and because there is some path from the initial state 0 to state 6 for sequence $\alpha\beta\gamma$.

Formally, a supervisor $\mathcal{S}$ is a pair $(T, \psi)$ in which $T$ is an automaton $T = (X, \Sigma, \xi, x_o)$, where $X$ is a set of *states* for the supervisor; $\Sigma$ is the alphabet used by $G$; $\xi$ is the *transition function*, a partial function $\xi$: $\Sigma \times X \rightarrow X$; $x_o$ is the *initial state* for the supervisor; and $\psi$, called a *feedback map*, is given by $\psi$: $\Sigma \times X \rightarrow \{disable, enable\}$ satisfying $\psi(\sigma, x) = enable$ if $\sigma \in \Sigma_{uc}$, $x \in X$, and $\psi(\sigma, x) \in \{disable, enable\}$ if $\sigma \in \Sigma_c$, $x \in X$. That is, $\psi$ is interpreted as a rule for disablement and ensures that uncontrollable events are never disabled. The automaton $T$ monitors the behavior of $G$ and changes state according to the events generated by $G$. The control rule $\psi(\sigma, x)$ indicates whether $\sigma$ should be enabled or disabled at the corresponding state in $G$. The behavior of $G$ when it is constrained by $\mathcal{S}$ is described by the automaton $\mathcal{S}/G$, called a *supervised discrete-event system*:

$$\mathcal{S}/G = (\Sigma, Q \times X, (\delta \times \xi)^{\psi}, (q_o, x_o)).$$

The behavior of $\mathcal{S}/G$ is described by $L(\mathcal{S}/G)$. The modified transition function $(\delta \times \xi)^{\psi}$ is defined as a mapping $\Sigma \times Q \times X \rightarrow Q \times X$:

$$(\delta \times \xi)^{\psi}(\sigma, (q, x)) := \begin{cases} (\delta(\sigma, q), \xi(\sigma, x)), & \text{if } \delta(\sigma, q)! \\ & \wedge \xi(\sigma, x)! \\ & \wedge \psi(\sigma, x) = \\ & \quad enable; \\ undefined, & \text{otherwise.} \end{cases}$$

There are some systems of which not all events can be seen by the supervisor. In this case, a supervisor has only a partial view of the system. When the application requires more than one such supervisor to achieve the desirable behavior, this leads to a decentralized discrete-event problem.

Now that a supervisor may no longer see every event in a sequence $t$, we need a formal way to represent the events that it does see. Suppose a supervisor $i$ can only observe events in some set $\Sigma_{i,o} \subseteq \Sigma$. For decentralized control problems with $n$ supervisors, a *projection operator* $P_i$ is defined for each supervisor and is a mapping from $\Sigma^*$ to $\Sigma_{i,o}^*$, for $i = 1, \cdots, n$. This operator effectively "erases" those elements $\sigma$ from a string $t$ that are not found in the set of observable events $\Sigma_{i,o}$

$$\begin{aligned} P_i(\varepsilon) &= \varepsilon \\ P_i(\sigma) &= \varepsilon, \qquad \sigma \in \Sigma \setminus \Sigma_{i,o} \\ P_i(\sigma) &= \sigma, \qquad \sigma \in \Sigma_{i,o} \\ P_i(t\sigma) &= P_i(t)P_i(\sigma), \qquad t \in \Sigma^*, \ \sigma \in \Sigma. \end{aligned} \tag{1}$$

Thus if the plant generates sequence $t$, then $P_i(t)$ indicates the sequence of events observed by supervisor $i$.

Informally, a supervisor is an agent that has the ability to control some events based on a partial view of the plant's behavior. To establish such supervision on $G$, we partition the set of events $\Sigma$ into the disjoint sets $\Sigma_c$, *controllable* events, and $\Sigma_{uc}$, *uncontrollable* events. Controllable events are those events whose occurrence is preventable (i.e., may be disabled). For decentralized control problems, the set of events each supervisor controls is denoted by $\Sigma_{i,c}$ where we assume $\Sigma_{i,c} \subseteq \Sigma_c$. Uncontrollable events are those events which cannot be prevented and are deemed permanently enabled.

Consider two local supervisors acting on $G$ to be

$$\mathcal{S}_1 = (T_1, \phi) \quad \text{and} \quad \mathcal{S}_2 = (T_2, \psi)$$

where $T_1 = (X, \Sigma, \xi, x_0)$ and $T_2 = (Y, \Sigma, \eta, y_0)$. The *conjunction* of $\mathcal{S}_1$ and $\mathcal{S}_2$ is the supervisor

$$\mathcal{S}_1 \wedge \mathcal{S}_2 := (T_1 \times T_2, \phi * \psi)$$

where

$$T_1 \times T_2 := (X \times Y, \Sigma, \xi \times \eta, (x_0, y_0))$$

and $\sigma \in \Sigma, x \in X, y \in Y \Rightarrow$

$$(\xi \times \eta)(\sigma, x, y) := \begin{cases} (\xi(\sigma, x), \eta(\sigma, y)), & \text{if } \xi(\sigma, x)! \\ & \wedge \eta(\sigma, y)! \\ undefined, & \text{otherwise} \end{cases}$$

$$(\phi * \psi)(\sigma, x, y) := \begin{cases} disable, & \text{if } \phi(\sigma, x) = disable \\ & \vee \psi(\sigma, y) = disable \\ enable, & \text{otherwise.} \end{cases}$$

That is, the composite supervisor $\mathcal{S}_1 \wedge \mathcal{S}_2$ disables an event if either $\mathcal{S}_1$ or $\mathcal{S}_2$ issues a disablement command. When a supervisor $\mathcal{S}$ is the result of a conjunction of two supervisors $\mathcal{S}_1$ and $\mathcal{S}_2$, we write $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$.

It is often convenient in the case of partial observability, to define a supervisor $\mathcal{S}_i$ only in terms of events in $\Sigma_{i,c}$ and $\Sigma_{i,o}$. In this case $\mathcal{S}_i$ can be extended to a supervisor $\tilde{\mathcal{S}}_i$. The *local* supervisor $\mathcal{S}_i$ acts only on events in $\Sigma_{i,c} \subseteq \Sigma$ and observes events in $\Sigma_{i,o} \subseteq \Sigma$ while $\tilde{\mathcal{S}}_i$ takes the same control action as $\mathcal{S}_i$ on $\Sigma_{i,c}$, enables all events in $\Sigma \setminus \Sigma_{i,c}$, makes the same transitions as $\mathcal{S}_i$ on $\Sigma_{i,o}$ and stays at the same state for events in $\Sigma \setminus \Sigma_{i,o}$. A supervisor $\tilde{\mathcal{S}}_i$ that acts on all of $\Sigma$ and mirrors the control actions

of a supervisor $\mathcal{S}_i$ that observes and controls only a subset of $\Sigma$ is called the *global extension* of $\mathcal{S}_i$.

The decentralized problem we consider is described in [2]:

*Given a plant $G$ over an alphabet $\Sigma$ (with controllable events $\Sigma_{1,c}$, $\Sigma_{2,c} \subseteq \Sigma$ and observable events $\Sigma_{1,o}$, $\Sigma_{2,o} \subseteq \Sigma$), and an automaton $E$, where $L(E)$ represents legal sequences, $L(E) \subseteq L(G)$ and $L(E) \neq \emptyset$, find local supervisors $\mathcal{S}_1$ and $\mathcal{S}_2$ such that $\tilde{\mathcal{S}}_1 \wedge \tilde{\mathcal{S}}_2$ is a supervisor for $G$ and such that*

$$L(\tilde{\mathcal{S}}_1 \wedge \tilde{\mathcal{S}}_2/G) = L(E). \tag{2}$$

*Here, for $i = 1, 2$, local supervisor $\mathcal{S}_i$ can observe only events in $\Sigma_{i,o}$ and can control only events in $\Sigma_{i,c}$ and $\tilde{\mathcal{S}}_i$ is the global extension of $\mathcal{S}_i$. The set of uncontrollable events, $\Sigma_{uc}$, is understood to be $\Sigma \setminus (\Sigma_{1,c} \bigcup \Sigma_{2,c})$.*

To describe a solution to the above problem, it is convenient to use the notion of *controllability* [1]. Given $G$ over an alphabet $\Sigma$, for a language $K \subseteq L(G)$, $K$ is *controllable* with respect to $G$ if

$$\overline{K}\Sigma_{uc} \cap L(G) \subseteq \overline{K} \tag{3}$$

where $\overline{K}\Sigma_{uc} := \{t\sigma | t \in \overline{K} \text{ and } \sigma \in \Sigma_{uc}\}$. If we think of $K$ as a set of "legal" sequences, then we want to know when it will be impossible to stop an illegal sequence from happening. It must be that the introduction of an uncontrollable event into a legal sequence results in another legal sequence. Therefore, to solve (2), it is necessary that $L(E)$ be controllable. If $L(E)$ is not controllable, a supremal controllable sublanguage of $L(E)$, possibly $\emptyset$, denoted $\sup \underline{C}(L(E), G)$, can always be found [1]. The standard solution to the centralized control problem with full observation produces a supervisor that acts on $G$ to generate $\sup \underline{C}(L(E), G)$. The important point to note is that such a solution is said to be "minimally restrictive," in that the supervisor disables events in $G$ only when absolutely necessary to prevent an illegal sequence from occurring. That is, the largest possible subset of legal sequences is generated.

A necessary and sufficient condition for the solution to the above decentralized problem can be found using the notion of *co-observability*. Given $G$ over an alphabet $\Sigma$, sets $\Sigma_{1,c}, \Sigma_{2,c}, \Sigma_{1,o}, \Sigma_{2,o} \subseteq \Sigma$, projections $P_1$: $\Sigma^* \rightarrow \Sigma_{1,o}^*$, $P_2$: $\Sigma^* \rightarrow \Sigma_{2,o}^*$, a prefix-closed language $K \subseteq L(G)$ is *co-observable* with respect to $G$, $P_1$, $P_2$ if

$$\begin{aligned}
&\forall t, t', t'' \in \Sigma^*,\ P_1(t) = P_1(t'),\ P_2(t) = P_2(t'') \Rightarrow \\
&\quad (\forall \sigma \in \Sigma_{1,c} \cap \Sigma_{2,c}) t \in \overline{K} \wedge s\sigma \in L(G) \wedge t'\sigma, t''\sigma \in \overline{K} \\
&\quad \Rightarrow t\sigma \in \overline{K} \qquad \textbf{conjunct 1} \\
&\wedge \quad (\forall \sigma \in \Sigma_{1,c} \setminus \Sigma_{2,c}) t \in \overline{K} \wedge t\sigma \in L(G) \wedge t'\sigma \in \overline{K} \\
&\quad \Rightarrow t\sigma \in \overline{K} \qquad \textbf{conjunct 2} \\
&\wedge \quad (\forall \sigma \in \Sigma_{2,c} \setminus \Sigma_{1,c}) t \in \overline{K} \wedge t\sigma \in L(G) \wedge t''\sigma \in \overline{K} \\
&\quad \Rightarrow t\sigma \in \overline{K} \qquad \textbf{conjunct 3.}
\end{aligned}$$

We would like a decentralized supervisor's view of a string to be enough for it to take the correct control action. If both supervisors can control the event in question (i.e., conjunct 1), then we just need one of the supervisors to be able to have an unambiguous view of the strings $t$, $t'$, $t''$ to make the correct control decision regarding $\sigma$. However, when an event is controlled by only one supervisor (i.e., conjuncts 2 and 3), then that supervisor's view of $t$, $t'$, $t''$ must be sufficient to decide on the control action for $\sigma$.

It is now possible to discuss the existence of a solution to the decentralized problem. The following theorem (along with its proof) appears as Theorem 4.1 in [2]:

*Theorem 1:* There exist supervisors $\tilde{\mathcal{S}}_1$ and $\tilde{\mathcal{S}}_2$ that solve the above decentralized supervisory control problem if and only if $L(E)$ is controllable with respect to $G$ and co-observable with respect to $G$, $P_1$, $P_2$.

Thus we can find decentralized controllers that synthesize $L(E)$ provided that the legal language satisfies the properties of controllability and co-observability. While it is possible to find the supremal controllable sublanguage of $L(E)$, if $L(E)$ is not co-observable there is no unique supremal co-observable sublanguage of $L(E)$.

### B. A Model for Knowledge

The framework for the modeling knowledge used is based on a knowledge logic for distributed systems [3], where multiple agents reason about their knowledge of the world. An agent could be a human, a machine (e.g., a robot) or even a component of a machine (e.g., an electrical circuit). Unless otherwise indicated, the definitions and results in this section are adopted from [30]. The model assumes that if an agent does not have complete knowledge of the true state of the world, it assumes a number of worlds are possible. The world is described in terms of a non-empty set $\Phi$ of facts or *primitive propositions*. More complicated formulas are constructed using expressions from propositional calculus: $\neg$ for negation and $\wedge$ for conjunction In addition, $\varphi \vee \psi$ represents $\neg(\neg\varphi \wedge \neg\psi)$.

The system model is conceptually divided into two components: the agents and the environment. The latter captures the relevant aspects of the system that are not part of the description of agent behavior. We assume that there is a set of agents $\mathtt{G} = \{1, \cdots, n\}$ to which we ascribe knowledge about the system.

The system behavior is captured by a *global* state. A global state is an $(n + 1)$-tuple, denoted $w$, that records the state of the environment and the *local* state—an agent's set of possible worlds—for each of the $n$ agents. Formally $w = (w_e, w_1, \cdots, w_n)$. We can further refer to individual components of $w$: $w_e$ and $w_i$ represent the local state of the environment and agent $i$ (for $i \in \{1, \cdots, n\}$), respectively.

We will reason about what an agent knows about the truth of facts in the system at global states. Knowledge of a fact is expressed using modal operators (one for each agent) $K_1, \cdots, K_n$. Thus $K_1 p$, where $p \in \Phi$, is interpreted as "agent 1 knows $p$."

The semantics of the possible-worlds model is formalized using *Kripke structures*. A Kripke structure $M$ is an $(n + 2)$-tuple containing a set of worlds (e.g., global states), an *interpretation* function $\pi$ that assigns truth values at each world $w$ to the primitive propositions in $\Phi$ (e.g., $\pi(w)(p) = \textbf{false}$), and *possibility relations*, one for each agent, that define binary relations on the set of worlds. That is, the relation defines the set of worlds that look alike to an agent at any given world in the system. The possibility relation is typically not defined

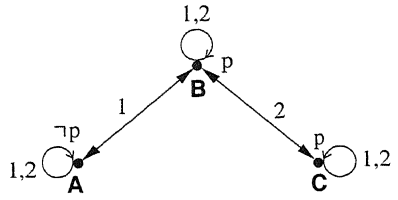


Fig. 2. A simple Kripke structure.

for the environment since we are not interested in ascribing knowledge to the environment.

A Kripke structure is also expressible as a labeled graph. In particular, nodes are worlds and edge labels (sets of agents) capture the possibility relation. For instance, worlds that look alike to agent $i$ are joined by an edge with a label "$i$." Each world is also labeled with the truth values of all primitive propositions $p \in \Phi$, where we use the notation "$\neg p$" to indicate that the truth value of $p$ is **false** and "$p$" corresponds to a value of **true.**

The following example illustrates a simple Kripke structure and is adapted from [30]. The graphical representation of this system is shown in Fig. 2. Suppose $\Phi = \{p\}$ and $n = 2$. Let the set of worlds be $\{\mathtt{A}, \mathtt{B}, \mathtt{C}\}$ and the interpretation function be defined such that proposition $p$ is true at worlds B and C but false at state A (i.e., $\pi(\mathtt{A})(p) = \mathbf{false}$, $\pi(\mathtt{B})(p) = \pi(\mathtt{C})(p) = \mathbf{true}$). The possibility relations for the agents are defined as follows: agent 1 cannot tell the difference between A and B while worlds B and C look alike to Agent 2. These relations are captured in Fig. 2 by the edge label of "1" joining worlds A and B and the edge label "2" joining worlds B and C. The self-loops at all three worlds with edge label "1,2" indicate that an agent cannot distinguish a given state from itself. For example, in addition to state B looking like A to agent 1, state A also looks like state A. For purposes of this discussion, the possibility relation is always an equivalence relation and therefore, it is always the case that reflexivity and symmetry hold. Therefore, from now on, self-loops and arrows will be omitted from diagrams of Kripke structures.

We now have all the components we need to reason about knowledge: a set of worlds describing the behavior of the system and an interpretation $\pi$ to analyze truth values of the propositions at each world in the system. Together the set of worlds and $\pi$ define an *interpreted system*, denoted by $\mathcal{I}$.

To discuss knowledge in an interpreted system, we assume that the possibility relation is defined as follows. Let $w$, $w'$ be two worlds in $\mathcal{I}$. We say $w$ and $w'$ are *indistinguishable to agent* $i$ if the local state according to agent $i$ is the same at both worlds:

$$w \sim_i w' if w_i = w'_i. \tag{4}$$

To discuss what it means for a fact $p$ to be true at a particular world in $\mathcal{I}$, we use the notation $(\mathcal{I}, w) \models p$, which can be read as "$p$ *is true at* $(\mathcal{I}, w)$" or "$p$ *holds at* $(\mathcal{I}, w)$". A fact $p$ holds at a state $w$ if the truth value as defined by $\pi$ is true at $w$. For example, at world B in Fig. 2, we can say $p$ is true because $\pi(\mathtt{B})(p) = true$. More formally:

$$(\mathcal{I}, w) \models p \qquad (\text{for } p \in \Phi) \quad \text{iff} \quad \pi(w)(p) = \mathbf{true}.$$

The clause for negation indicates that $\neg p$ is true at world $w$ exactly if $p$ is not true:

$$(\mathcal{I}, w) \models \neg p \quad \text{iff} \quad (\mathcal{I}, w) \not\models p.$$

In Fig. 2, we can say at world A, $\neg p$ holds because $p$ is not true at A.

We can consider more than one fact holding at a world:

$$(\mathcal{I}, w) \models p_1 \wedge p_2 \quad \text{iff} \quad (\mathcal{I}, w) \models p_1 \quad \text{and} \quad (\mathcal{I}, w) \models p_2.$$

Thus, the conjunction of two propositions holds at $w$ if it is the case that each proposition is true at $w$.

What does it mean for an agent to know facts in its world? An agent knows a fact $p$ at $w$ if $p$ holds at all worlds that the agent cannot distinguish from $w$:

$$(\mathcal{I}, w) \models K_i p \quad \text{iff} \quad (\mathcal{I}, w') \models p, \quad \forall w' \text{ such that } w \sim_i w'. \tag{5}$$

Referring to Fig. 2 again, we can now describe the knowledge of agents at any world in the system: e.g., at world B, the formula $\neg K_1 p \wedge K_2 p$ holds. That is, at world B agent 1 does not know whether $p$ is true, while at world B agent 2 knows that $p$ is true. At world B, agent 1 considers the existence of two possible worlds: A and B. It considers both $p$ and $\neg p$ to be possible because $p$ is false at world A while $p$ is true at world B. Agent 2 also considers the existence of two possible worlds: B and C. However, since $p$ holds at both those worlds, at B agent 2 knows that $p$ is true.

It follows that if at $w$ an agent knows $p$, it also knows $p$ at all other worlds it considers possible at $w$:

$$(\mathcal{I}, w) \models K_i p \quad \text{iff} \quad (\mathcal{I}, w') \models K_i p \tag{6}$$

for all $w'$ such that $w \sim_i w'$. For instance, agent 2 considers that worlds B and C "look alike." Since $p$ is true at both these worlds, we can say that at world B, agent 2 knows $p$. Similarly, we can say that at world C agent 2 also knows $p$.

Finally, we note a property, called the *Knowledge Axiom,* that states if an agent knows a fact, then the fact is true:

$$(\mathcal{I}, w) \models K_i p \Rightarrow (\mathcal{I}, w) \models p. \tag{7}$$

Note that since $K_i p$ holds at some world $w$, we know by (6) that $K_i p$ holds at all worlds that agent $i$ cannot distinguish from $w$. Since $p$ is true in all worlds that agent $i$ considers possible, in particular, $p$ is true at $w$.

## III. A Knowledge Model for DES

In this section we describe how to recast decentralized supervisory control problems as interpreted systems. We do not claim that the reformulation of this problem provides a more efficient solution but, rather, suggest that knowledge theory provides a more natural way of thinking about discrete-event control problems.

### A. The Interpreted System $\mathcal{I}^{\mathrm{DES}}$

We denote our "sequence-based" interpreted system for describing decentralized discrete-event systems as $\mathcal{I}^{\mathrm{DES}}(G, E)$.

Like local supervisors in the decentralized DES formulation of [22], [2], the agents in this interpreted system make control decisions based on their partial view of the *sequences* generated by the DES plant $G$. The *environment* of our interpreted system is the language generated by the plant and the *agents* play a role equivalent to that of decentralized DES supervisors.

A global state for $n$ agents in $\mathcal{I}^{\mathrm{DES}}(G, E)$ captures a "snapshot" of the sequence generated by the plant language $L(G)$. The set of states for the environment is the set of sequences in $L(G)$, while the set of local states for the agents is the set of sequences each agent observes according to the projection operation of (1). More formally, a global state is defined as $w = (w_e, w_1, \cdots, w_n) = (t, P_1(t), \cdots, P_n(t))$ for $t \in L(G)$. In this paper, we will assume that $n = 2$ so that the group of agents is $\mathtt{G} = \{1, 2\}$.

The interpretation $\pi^{\mathrm{DES}}$ associated with our interpreted system captures the notion of whether or not an event in $\Sigma$ is permissible as sequences evolve in the plant. To form $\Phi$, the set of primitive propositions for $\mathcal{I}^{\mathrm{DES}}(G, E)$, we want to associate with each $\sigma \in \Sigma$ two distinct propositions: $\sigma_G$ to represent the fact that at a particular state in the plant the event is defined (i.e., is possible), and $\sigma_E$ to represent the fact that at the corresponding state in the legal automaton the event is defined. The propositions are defined in terms of events because we want to reason about the knowledge an agent has of the occurrence of an event, instead of, for instance, a certain sequence. If $\Sigma$ is finite (i.e., $|\Sigma| = N$); it can be written as $\Sigma = \{\sigma_1, \sigma_2, \cdots, \sigma_N\}$. Define the set of propositions $\Phi_G = \{\sigma_i^G | i = 1, \cdots, N\}$ and the set of propositions $\Phi_E = \{\sigma_i^E | i = 1, \cdots, N\}$ where $\Phi_G$ and $\Phi_E$ contain $N$ symbols. Let $\Phi = \Phi_G \cup \Phi_E$. Because we will frequently want to associate $\sigma_i^G$ with its counterpart $\sigma_i^E$, we define the relation $\mathtt{R}_\Sigma$ such that $\mathtt{R}_\Sigma \subseteq \Phi_G \times \Phi_E$ and $\mathtt{R}_\Sigma := \{(\sigma_G, \sigma_E) | \exists\, \sigma_i \in \Sigma$ where $\sigma_G = \sigma_i^G, \sigma_E = \sigma_i^E\}$. For convenience, we will use the notation $\sigma_G$ (respectively, $\sigma_E$) without explicit reference to $\mathtt{R}_\Sigma$ when we mean $\sigma_i^G$ (respectively, $\sigma_i^E$). The proposition $\sigma_G$ is "event $\sigma$ can occur" and $\sigma_E$ is "event $\sigma$ is legal." For convenience, we will refer to $\Phi_{uc}$ when we need to identify those propositions in $\Phi$ which represent events in $\Sigma_{uc}$ (as defined in Section II-A).

The interpretation for the propositions in $\Phi$ is defined for all $\sigma \in \Sigma$

$$\pi^{\mathrm{DES}}(w)(\sigma_G) := \begin{cases} \textbf{true}, & \text{if } \delta^G(w_e\sigma, q_0^G)! \\ \textbf{false}, & \text{otherwise.} \end{cases}$$

$$\pi^{\mathrm{DES}}(w)(\sigma_E) := \begin{cases} \textbf{true}, & \text{if } \delta^E(w_e\sigma, q_0^E)! \\ \textbf{false}, & \text{otherwise.} \end{cases} \quad (8)$$

In other words, a proposition $\sigma_G$ is true at a global state $w$ if the event $\sigma$ happens at the actual plant state reached by $w_e$. A proposition $\sigma_G$ is false (denoted $\neg\sigma_G$) at world $w$ if the event $\sigma$ is not defined directly following the event sequence $w_e$. Similarly, a proposition $\sigma_E$ is true at world $w$ if the event $\sigma$ happens directly following the event sequence $w_e$ and $w_e\sigma$ is part of the legal behavior of the plant. A proposition $\sigma_E$ is false (denoted $\neg\sigma_E$) at a global state $w$ if either $\sigma_G$ is false or if the sequence $w_e\sigma$ is part of the illegal behavior of the plant.

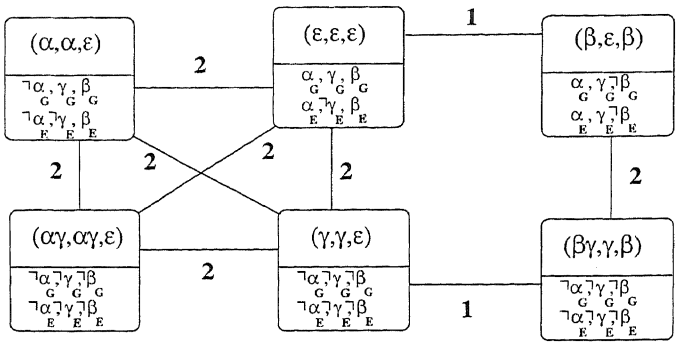


Fig. 3. A portion of the Kripke structure for $G$ in Fig. 1.

Because the set of worlds in $\mathcal{I}^{\mathrm{DES}}(G, E)$ and the truth values assigned by $\pi^{\mathrm{DES}}$ to the primitive propositions in $\Phi$ are derived from the legal automaton $E$ and the plant $G$, we can consider that $\mathcal{I}^{\mathrm{DES}}$ has implicit parameters $G$ and $E$. Thus, for convenience, we drop these arguments for the rest of the paper.

We illustrate the knowledge model by constructing a Kripke structure for the plant and legal automaton in Fig. 1. In this example, suppose that agent 1 sees and controls events $\alpha$ and $\gamma$ while agent 2 sees event $\beta$ and controls events $\beta$ and $\gamma$.

The complete Kripke structure contains ten states, corresponding to the ten sequences in $L(G)$. We show a representative portion of the structure in Fig. 3. The set of propositions is $\Phi = \{\alpha_G, \alpha_E, \beta_G, \beta_E, \gamma_G, \gamma_E\}$ and the truth assignments for $\pi^{\mathrm{DES}}$ are made according to (8). For example, at global state $(\alpha, \alpha, \varepsilon)$, proposition $\gamma_G$ is assigned a value of **true** because there is a transition of $\gamma$ from sequence $\alpha$ in the plant; however, $\gamma_E$ has a truth assignment of **false** because in Fig. 1, the sequence $\alpha\gamma$ is not defined in the legal automaton. The possibility relations describe how agents view the world. In $\mathcal{I}^{\mathrm{DES}}$, the possibility relation for each agent is defined using the indistinguishability relation $\sim_i$ of (4). That is, two global states look alike to an agent $i$ if the global states have the same local state according to agent $i$. For instance, the possibility relation for agent 1 would contain the pair of states $((\gamma, \gamma, \varepsilon), (\beta\gamma, \gamma, \beta))$ because these states have the same local state according to agent 1, namely $\gamma$.

The top half of a node in Fig. 3 contains one of the global states in the system and the bottom half of the node shows the truth values for the primitive propositions at that global state. The diagram exactly describes the states of the plant where we want to impose control (as constrained by the legal language) and what each agent believes are the possible worlds of this system. By following the edges connecting the nodes, we can also determine what each agent believes are its possible worlds. For instance, the node labeled $(\beta, \varepsilon, \beta)$ looks the same to agent 2 as the node labeled $(\beta\gamma, \gamma, \beta)$ because the local state of agent 2 is $\beta$ in both the global states. This is indicated by an edge labeled "2" joining these nodes.

We can describe the knowledge of each agent at a particular state in the interpreted system. For example, at $w = (\varepsilon, \varepsilon, \varepsilon)$

$$(\mathcal{I}^{\mathrm{DES}}, w) \models K_1(\alpha_G \wedge \alpha_E \wedge \gamma_G) \wedge K_2\neg\gamma_E.$$

At $w$ the set of worlds that agent 1 considers possible is $\{(\varepsilon, \varepsilon, \varepsilon), (\beta, \varepsilon, \beta)\}$. At both these states the truth values

of $\alpha_G$, $\alpha_E$ and $\gamma_G$ are true. Therefore we say that "Agent 1 knows $\alpha$ can happen and is legal at $(\varepsilon, \varepsilon, \varepsilon)$ and that $\gamma$ can happen at $(\varepsilon, \varepsilon, \varepsilon)$." Similarly, we say that "Agent 2 knows that it is not the case that $\gamma$ is legal at $(\varepsilon, \varepsilon, \varepsilon)$" because at all states indistinguishable from $(\varepsilon, \varepsilon, \varepsilon)$ to agent 2—namely $(\varepsilon, \varepsilon, \varepsilon)$, $(\alpha, \alpha, \varepsilon)$, $(\alpha\gamma, \alpha\gamma, \varepsilon)$ and $(\alpha, \alpha, \varepsilon)$—the formula $\neg\gamma_E$ is true. We can also make note of the lack of knowledge an agent has: at $w$ agent 1 does not know whether $\gamma$ is legal because at $(\varepsilon, \varepsilon, \varepsilon)$ the formula $\neg\gamma_E$ is true while at $(\beta, \varepsilon, \beta)$ the formula $\gamma_E$ is true. This is denoted as $(\mathcal{I}^{\mathrm{DES}}, r, m) \models \neg K_1\gamma_E$ and is read "Agent 1 does not know whether $\gamma$ is legal at $(\varepsilon, \varepsilon, \varepsilon)$."

## IV. Knowledge-Based Protocols

The interpreted system $\mathcal{I}^{\mathrm{DES}}$ describes the knowledge that each agent has concerning the validity of a particular sequence. We need to associate actions with an agent's knowledge. For instance, if an agent knows that a particular proposition is possible but not legal for a set of possible worlds, then we want it to disable the corresponding event. A knowledge-based protocol [4] is a strategy that links actions and knowledge for agents and the environment. We believe that it is natural to think of a supervisor basing its control actions on what the supervisor "knows" about the present state of the system. Even though we depart from some of the specifics of the formalism for knowledge-based protocols in [4], we incorporate the idea that there ought to be a connection between action and knowledge.

We examine knowledge-based protocols where actions to disable or enable an event are based only on local state information and where an uncontrollable event cannot be disabled. We describe protocols for agents but not for the environment, which we view as incapable of taking control actions. For the decentralized DES we consider, when we say that a supervisor $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ solves a problem, we mean that when $G$ is under the control of $\mathcal{S}$ (i.e., when $\mathcal{S}$ disables or enables events of $G$), the resultant language generated, namely $L((\mathcal{S}_1 \wedge \mathcal{S}_2)/G)$, equals the legal language: $L((\mathcal{S}_1 \wedge \mathcal{S}_2)/G) = L(E)$. Solving the decentralized problem with a knowledge-based protocol amounts to constructing a protocol that will ensure that all legal sequences and only legal sequences are generated.

### A. A Knowledge-Based Protocol For Decentralized Control

A local supervisor in a decentralized DES system disables controllable event $\sigma$ if the supervisor is able to determine that the occurrence of $\sigma$ will lead to illegal behavior; otherwise event $\sigma$ is enabled. An event $\sigma$ is disabled if at least one local agent takes a "disable $\sigma$" action.

The actions that drive the global state changes of the system are performed according to a selection rule or *protocol.* A *knowledge-based* protocol is a protocol where actions are taken on the basis of the local knowledge of an agent. We define a knowledge-based protocol as a mapping that characterizes which events are disabled $\mathcal{KP}_i$: $\mathcal{L}_i \times \Sigma \to \{enable, disable\}$, where $\mathcal{L}_i$ is the set of local states for agent $i$. Since the knowledge-based protocol is defined on the local view of an agent, the actions of agent $i$ at $w$ are applied at all $w'$ that are indistinguishable to agent $i$ at $w$. Just as a local decentralized DES supervisor makes control decisions based on its partial view of a sequence, we want an agent to use knowledge and its local states to determine if a given event should be disabled. A *joint* knowledge-based protocol is the collection of the knowledge-based protocols for all agents in $\mathtt{G}$.

We identify the group of agents that can control a given event as $\mathtt{G}_\sigma := \{i\colon \sigma \in \Sigma_{i,c}\}$, i.e., for all $\sigma \in \Sigma_{1,c} \cap \Sigma_{2,c}$, $\mathtt{G}_\sigma = \{1, 2\}$; for all $\sigma \in \Sigma_{1,c} \setminus \Sigma_{2,c}$, $\mathtt{G}_\sigma = \{1\}$; for all $\sigma \in \Sigma_{2,c} \setminus \Sigma_{1,c}$, $\mathtt{G}_\sigma = \{2\}$, and for all $\sigma \in \Sigma_{uc}$, $\mathtt{G}_\sigma = \emptyset$.

A joint knowledge-based protocol $\mathcal{KP} = (\mathcal{KP}_1, \mathcal{KP}_2)$ solves the decentralized problem if for all $w \in \mathcal{I}^{\mathrm{DES}}$ and all $(\sigma_G, \sigma_E) \in \mathtt{R}_\Sigma$:

$$\text{i)}\quad (\mathcal{I}^{\mathrm{DES}}, w) \models \sigma_G \wedge \neg\sigma_E \Rightarrow (\exists\, i \in \mathtt{G}_\sigma)\mathcal{KP}_i(w_i, \sigma) = disable$$

$$\text{ii)}\quad (\mathcal{I}^{\mathrm{DES}}, w) \models \sigma_G \wedge \sigma_E \Rightarrow (\nexists\, i \in \mathtt{G})\mathcal{KP}_i(w_i, \sigma) = disable.$$

That is, solving the problem amounts to allowing only legal sequences and all legal sequences to occur. Note that since we assume that $E$ is a subautomaton of $G$, it will never be the case that $(\mathcal{I}^{\mathrm{DES}}, w) \models \neg\sigma_G \wedge \sigma_E$.

To solve the decentralized control problem using knowledge-based protocols we must formalize what it means for agents to "know enough." We describe several conditions that $\mathcal{I}^{\mathrm{DES}}$ must satisfy before a solution can be achieved. In particular, we present a necessary and sufficient condition so that our joint knowledge-based protocol admits only (and all) the legal sequences in $L(E)$.

We define a property analogous to co-observability [2] and controllability [1] that characterizes the nature of knowledge an interpreted system requires to yield a decentralized solution to the DES control problem.

*Definition 1:* An interpreted system $\mathcal{I}^{\mathrm{DES}}$ is Kripke-observable if:

$$\forall\, w \in \mathcal{I}^{\mathrm{DES}},\quad \forall\, (\sigma_G, \sigma_E) \in \mathtt{R}_\Sigma,\quad (\mathcal{I}^{\mathrm{DES}}, w) \models \neg\sigma_G \vee \sigma_E \vee (\exists\, i \in \mathtt{G}_\sigma) \quad \text{such that } (\mathcal{I}^{\mathrm{DES}}, w) \models K_i\neg\sigma_E. \tag{9}$$

That is, if an illegal event $\sigma$ is about to occur, at least one agent that can control $\sigma$ knows that it should be disabled. We note here that co-observability is a condition on set membership and set containment for sets of sequences while Kripke-observability involves logic tests on propositions.

The condition we were initially trying to capture in the definition of Kripke-observability was that for every event that *can* occur, at least one agent knows whether or not that event is legal. Our intuition led us to the following logic formulation:

$$\forall\, w \in \mathcal{I}^{\mathrm{DES}},\quad \forall\, (\sigma_G, \sigma_E) \in \mathtt{R}_\Sigma,\quad (\exists\, i \in \mathtt{G}_\sigma) \quad \text{such that} \quad (\mathcal{I}^{\mathrm{DES}}, w) \models K_i\sigma_E \vee K_i\neg\sigma_E. \tag{10}$$

However, this is actually too strong a condition as the following example will illustrate.

The plant and legal automaton in Fig. 4 represent a co-observable language (with supervisors $\mathcal{S}_1$ and $\mathcal{S}_2$) if $\Sigma_{1,o} = \{\alpha\} = \Sigma_{1,c}$, $\Sigma_{2,o} = \{\beta\}$, and $\Sigma_{2,c} = \{\alpha, \beta\}$. Thus a control strategy for this decentralized problem is as follows: $\mathcal{S}_1$ disables $\alpha$ after seeing $\alpha$ and $\mathcal{S}_2$ disables $\alpha$ after seeing $\beta\beta$. If we were to use the condition in (10), then the condition would fail at $w = (\varepsilon, \varepsilon, \varepsilon)$. At this state, the possible worlds of agent 1 are $(\varepsilon, \varepsilon, \varepsilon)$, $(\beta, \varepsilon, \beta)$, and $(\beta\beta, \varepsilon, \beta\beta)$. Both disjuncts of (10) fail for event $\alpha$ at $w$ since $(\mathcal{I}^{\mathrm{DES}}, w) \models \alpha_E$, whereas at $w'$ and $w''$, when $w' = (\beta, \varepsilon, \beta)$ and $w'' = (\beta\beta, \varepsilon, \beta\beta)$, we have $(\mathcal{I}^{\mathrm{DES}}, w') \models \neg\alpha_E$ and $(\mathcal{I}^{\mathrm{DES}}, r'', m'') \models \neg\alpha_E$. The possible worlds for agent 2 at $w = (\varepsilon, \varepsilon, \varepsilon)$ are states $w' = (\alpha, \alpha, \varepsilon)$, $w'' = (\alpha\alpha, \alpha\alpha, \varepsilon)$ and $w$ itself. As it did for agent 1, both disjuncts fail on event $\alpha$ at all states indistinguishable from $w$ since $(\mathcal{I}^{\mathrm{DES}}, w) \models \alpha_E$ while $(\mathcal{I}^{\mathrm{DES}}, w') \models \neg\alpha_E$ and $(\mathcal{I}^{\mathrm{DES}}, w'') \models \neg\alpha_E$. In fact, the failure to meet the condition of (10) was because we required an agent to know when an event should be enabled.

This observation led to a revised definition:

$$\begin{aligned} &\forall w \in \mathcal{I}^{\mathrm{DES}}, \qquad \forall (\sigma_G, \sigma_E) \in \mathrm{R}_{\Sigma}, \\ &\quad (\mathcal{I}^{\mathrm{DES}}, w) \models \sigma_E \\ &\quad \vee \qquad (\exists i \in \mathrm{G}_\sigma) \quad \text{such that } (\mathcal{I}^{\mathrm{DES}}, w) \models K_i \neg\sigma_E. \end{aligned} \tag{11}$$

Now that the condition in (11) does not insist on knowledge if the event is legal, this updated condition is satisfied for event $\alpha$ at all states in Fig. 4. However, Fig. 5 shows another co-observable system where (11) also fails. Let $\Sigma_{1,o} = \{\alpha\}$, $\Sigma_{1,c} = \{\alpha, \beta\}$, $\Sigma_{2,o} = \{\mu\}$, $\Sigma_{2,c} = \{\mu\}$. There is a problem with event $\mu$, even though this event never needs to be disabled. At state $w = (\alpha, \alpha, \varepsilon)$, agent 2 neither knows that $\mu$ should be disabled nor that it should not be disabled since $(\mathcal{I}^{\mathrm{DES}}, w) \models \neg\mu_E$, while at $w' = (\varepsilon, \varepsilon, \varepsilon)$, and therefore $w \sim_2 w'$, it is the case that $(\mathcal{I}^{\mathrm{DES}}, w') \models \mu_E$. Therefore, at $w$ neither disjunct of condition (11) is satisfied. Since $\mu$ cannot actually happen after $\alpha$ occurs, it is too strong to require than an agent possess any knowledge about $\mu$ at $w$.

If we use the definition of (9), the system of Fig. 5 is Kripke-observable. In particular, at state $w = (\alpha, \alpha, \varepsilon)$, the first disjunct for event $\mu$ is now satisfied: $(\mathcal{I}^{\mathrm{DES}}, w) \models \neg\mu_G$.

Note that even though the definitions in (10) and (11) fail on this example system, there is still a solution to the problem. This is because the default action is to enable an event when no agent knows whether or not to disable that event. The requirement in the first disjunct of (10) to *know* that the event should be enabled is too strong. Similarly (11) fails as we neglected to notice that we can omit knowledge tests on "do not care" states, that is, global states where events are not even defined in the corresponding DES plant states. Hence this knowledge condition can be relaxed so that a test for knowledge is only performed when an event $\sigma$ is possible but is not legal (i.e., when the disjunct $\neg\sigma_G \vee \sigma_E$ does not hold).

*Theorem 2:* Given $G$, $E$, there exists a joint knowledge-based protocol $\mathcal{KP} = (\mathcal{KP}_1, \mathcal{KP}_2)$ that solves the decentralized problem iff $\mathcal{I}^{\mathrm{DES}}(G, E)$ is Kripke-observable.

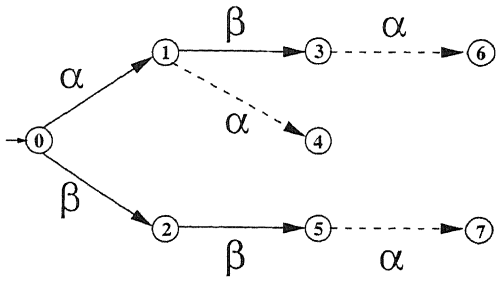


Fig. 4. A plant $G$ and legal automaton $E$. (Illegal transitions are marked with a dashed line.)

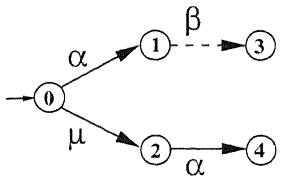


Fig. 5. A plant $G$ and legal automaton $E$. (Illegal transitions are marked with a dashed line.)

*Proof:* ($\Leftarrow$) Suppose $\mathcal{I}^{\mathrm{DES}}$ is Kripke-observable. Define the following knowledge-based protocol:

$$(\forall \ell \in \mathcal{L}_i)(\forall \sigma \in \Sigma)$$

$$(\forall i \in \mathrm{G}_\sigma)\mathcal{KP}_i(\ell, \sigma) = \begin{cases} \mathit{disable}, & \text{if } \exists w \text{ such that } \ell = w_i \wedge (\mathcal{I}^{\mathrm{DES}}, w) \models K_i\neg\sigma_E, \\ \mathit{enable}, & \text{otherwise.} \end{cases}$$

$$(\forall j \notin \mathrm{G}_\sigma)\mathcal{KP}_j(\ell, \sigma) = \mathit{enable}.$$

If an agent knows than an event is illegal, it will disable the event. Therefore, unless an agent knows that an event is illegal, the event will be enabled. That is, if an agent knows that an event is illegal, it will disable the event—hence the motto "know means no." Note that for an event controllable by agent $i$, the definition of $\mathcal{KP}_i$ in (12) is robust to the choice of $w$ in (12) (i.e., if a different $w'$ were chosen such that $\ell = w'_i$, then by (6), $(\mathcal{I}^{\mathrm{DES}}, w) \models K_i\neg\sigma_E$ iff $(\mathcal{I}^{\mathrm{DES}}, w') \models K_i\neg\sigma_E$).

We want to show that $\mathcal{KP} = (\mathcal{KP}_1, \mathcal{KP}_2)$ solves the decentralized problem.

i) Suppose $(\mathcal{I}^{\mathrm{DES}}, w) \models (\sigma_G \wedge \neg\sigma_E)$. We want to show that this implies $(\exists i \in \mathrm{G}_\sigma)\ \mathcal{KP}_i(w_i, \sigma) = \mathit{disable}$.

Since $(\mathcal{I}^{\mathrm{DES}}, w) \models \sigma_G \wedge \neg\sigma_E$, we have $(\mathcal{I}^{\mathrm{DES}}, w) \not\models \neg\sigma_G \vee \sigma_E$. Thus, since Kripke-observability holds, it must be the case that $\exists i \in \mathrm{G}_\sigma$ such that $(\mathcal{I}^{\mathrm{DES}}, w) \models K_i\neg\sigma_E$. Therefore either $\mathcal{KP}_1(w_i, \sigma) = \mathit{disable}$ or $\mathcal{KP}_2(w_i, \sigma) = \mathit{disable}$.

ii) Suppose $(\mathcal{I}^{\mathrm{DES}}, w) \models \sigma_G \wedge \sigma_E$. We want to show that this implies $(\nexists i \in \mathrm{G})\ \mathcal{KP}_i(w_i, \sigma) = \mathit{disable}$.

The requirement for $\mathcal{KP}_i(w_i, \sigma) = \mathit{disable}$ for some $i \in \mathrm{G}_\sigma$ is that $(\mathcal{I}^{\mathrm{DES}}, w) \models K_i\neg\sigma_E$. However, since $(\mathcal{I}^{\mathrm{DES}}, w) \models \sigma_E$, we cannot have $(\mathcal{I}^{\mathrm{DES}}, w) \models K_i\neg\sigma_E$ for any $i$ [by (7)]. Therefore $\mathcal{KP}_i(w_i, \sigma) = \mathit{enable}$ for all $i \notin \mathrm{G}_\sigma$. By (13), for all $i \notin \mathrm{G}_\sigma$, $\mathcal{KP}_i(w_i, \sigma) = \mathit{enable}$.

TABLE I
CHECKING KRIPKE-OBSERVABILITY

| $w$ | Disjunct of Kripke-observability satisfied | | |
|---|---|---|---|
| $(\varepsilon,\varepsilon,\varepsilon)$ | $(\mathcal{I}^{DES},w)\models K_2\neg\gamma_E$ | $(\mathcal{I}^{DES},w)\models \alpha_E$ | $(\mathcal{I}^{DES},w)\models \beta_E$ |
| $(\beta,\varepsilon,\beta)$ | $(\mathcal{I}^{DES},w)\models \gamma_E$ | $(\mathcal{I}^{DES},w)\models \alpha_E$ | $(\mathcal{I}^{DES},w)\models \neg\beta_G$ |
| $(\gamma,\gamma,\varepsilon)$ | $(\mathcal{I}^{DES},w)\models K_2\neg\gamma_E$ | $(\mathcal{I}^{DES},w)\models \neg\alpha_G$ | $(\mathcal{I}^{DES},w)\models \neg\beta_G$ |
| $(\alpha,\alpha,\varepsilon)$ | $(\mathcal{I}^{DES},w)\models K_2\neg\gamma_E$ | $(\mathcal{I}^{DES},w)\models \neg\alpha_G$ | $(\mathcal{I}^{DES},w)\models \beta_E$ |
| $(\beta\gamma,\gamma,\beta)$ | $(\mathcal{I}^{DES},w)\models \neg\gamma_G$ | $(\mathcal{I}^{DES},w)\models \neg\alpha_G$ | $(\mathcal{I}^{DES},w)\models \neg\beta_G$ |
| $(\alpha\gamma,\alpha\gamma,\varepsilon)$ | $(\mathcal{I}^{DES},w)\models K_2\neg\gamma_E$ | $(\mathcal{I}^{DES},w)\models \neg\alpha_G$ | $(\mathcal{I}^{DES},w)\models \neg\beta_G$ |
| $(\alpha\beta,\alpha,\beta)$ | $(\mathcal{I}^{DES},w)\models \gamma_E$ | $(\mathcal{I}^{DES},w)\models \neg\alpha_G$ | $(\mathcal{I}^{DES},w)\models \neg\beta_G$ |
| $(\beta\alpha,\alpha,\beta)$ | $(\mathcal{I}^{DES},w)\models \gamma_E$ | $(\mathcal{I}^{DES},w)\models \neg\alpha_G$ | $(\mathcal{I}^{DES},w)\models \neg\beta_G$ |
| $(\alpha\beta\gamma,\alpha\gamma,\beta)$ | $(\mathcal{I}^{DES},w)\models \neg\gamma_G$ | $(\mathcal{I}^{DES},w)\models \neg\alpha_G$ | $(\mathcal{I}^{DES},w)\models \neg\beta_G$ |
| $(\beta\alpha\gamma,\alpha\gamma,\beta)$ | $(\mathcal{I}^{DES},w)\models \neg\gamma_G$ | $(\mathcal{I}^{DES},w)\models \neg\alpha_G$ | $(\mathcal{I}^{DES},w)\models \neg\beta_G$ |

($\Rightarrow$) We need to show that if $\mathcal{I}^{\mathrm{DES}}$ is not Kripke-observable then there is no joint knowledge-based protocol $\mathcal{KP}$ that solves the decentralized problem.

Suppose that some $\mathcal{KP}$ solves the decentralized problem. Since $\mathcal{I}^{\mathrm{DES}}$ is not Kripke-observable, there must exist $w \in \mathcal{I}^{\mathrm{DES}}$ such that $\exists(\sigma_G,\sigma_E) \in \mathrm{R}_\Sigma$ where $(\forall i \in \mathrm{G}_\sigma)$ $(\mathcal{I}^{\mathrm{DES}}, w) \not\models \neg\sigma_G \vee \sigma_E \vee K_i\neg\sigma_E$. That is

$$(\mathcal{I}^{\mathrm{DES}}, w) \not\models \neg\sigma_G \vee \sigma_E \qquad (12);$$

and for all $i \in \mathrm{G}_\sigma$

$$(\mathcal{I}^{\mathrm{DES}}, w) \not\models K_i\neg\sigma_E. \qquad (13)$$

The expression in (14) implies that $(\mathcal{I}^{\mathrm{DES}}, w) \models (\sigma_G \wedge \neg\sigma_E)$. Since $\mathcal{KP}$ solves the decentralized problem, and since $(\mathcal{I}^{\mathrm{DES}}, w) \models (\sigma_G \wedge \neg\sigma_E)$, it must be the case that $\exists\, j$ such that $\mathcal{KP}_j(w_j, \sigma) = \mathit{disable}$. Note that (15) holds for agent $j$ and implies that $\exists\, w'$ such that $(\mathcal{I}^{\mathrm{DES}}, w') \models \sigma_E$ and $w \sim_j w'$. Since $\mathcal{KP}$ solves the decentralized problem, $\mathcal{KP}_j(w'_j, \sigma) \neq \mathit{disable}$. However, since $\mathcal{KP}_j(w_j,\sigma) = \mathit{disable}$, this means that $\mathcal{KP}_j(w'_j, \sigma) = \mathit{disable}$ (since $w \sim_j w'$ means that $w_j = w'_j$), which leads to a contradiction. ■

We want to ensure that the actions taken by agents as a result of executing the joint knowledge protocol exactly permit the legal language of the DES plant. Therefore, we make precise the set of sequences that are generated by the supervised interpreted system.

*Definition 2:* The language that contains all sequences determined by the actions of the joint knowledge-based protocol is $\mathcal{KL}(\mathcal{KP}, G)$, defined as follows:

$$\begin{aligned} &\varepsilon \in \mathcal{KL}, \\ &w_e\sigma \in \mathcal{KL} \quad \text{if } w_e \in \mathcal{KL} \wedge w_e\sigma \in L(G) \\ &\qquad \wedge \mathcal{KP}_i(w_i, \sigma) = \mathit{enable}, \qquad (i = 1, 2). \end{aligned}$$

Thus a sequence is in $\mathcal{KL}$ if its prefix is already in $\mathcal{KL}$, the sequence is actually generated by the plant and the control action at the corresponding global state is to allow $\sigma$ to happen. If a "disable $\sigma$" action is taken at global state $w$, it is because at least one agent knows that $w_e\sigma$ is not part of the legal language and is therefore, by the above definition, not included in $\mathcal{KL}$.

### B. Example: A Kripke-Observable System

We return to the plant and legal automaton of Fig. 1 where agent 1 sees and controls events $\alpha$ and $\gamma$ while agent 2 sees $\beta$ but controls both $\beta$ and $\gamma$. Part of the Kripke structure associated with the plant is shown in Fig. 3.

We want to show that $\mathcal{I}^{\mathrm{DES}}$ is Kripke-observable. Let $w = (\varepsilon, \varepsilon, \varepsilon)$. We want to ascertain that at this state either agent 1 or agent 2 knows to disable $\gamma$:

$$(\mathcal{I}^{\mathrm{DES}}, w) \not\models \neg\gamma_G \vee \gamma_E. \qquad (14)$$

Therefore, for Kripke-observability, we must find an agent $i$ such that $(\mathcal{I}^{\mathrm{DES}}, w) \models K_i\neg\gamma_E$.

We first check to see if agent 1 has the appropriate knowledge about event $\gamma$. At $w = (\varepsilon, \varepsilon, \varepsilon)$, agent 1 considers one other world to be possible: $(\beta, \varepsilon, \beta)$.

Recall that knowledge of a fact at $w$ requires that the fact hold at all states indistinguishable from $w$. Thus if agent 1 has knowledge that event $\gamma$ is not legal at this part of the plant, it must be the case that $\neg\gamma_E$ holds in the two worlds noted above. Agent 1 fails to have the required knowledge at state $w' = (\beta, \varepsilon, \beta)$, since $(\mathcal{I}^{\mathrm{DES}}, w') \models \gamma_E$. Thus, $(\mathcal{I}^{\mathrm{DES}}, w) \models \neg K_1\neg\gamma_E$ and we must check to see if Kripke-observability is satisfied by agent 2's knowledge at this state.

There are three other possible worlds agent 2 cannot distinguish from $(\varepsilon, \varepsilon, \varepsilon)$: $(\alpha, \alpha, \varepsilon)$, $(\gamma, \gamma, \varepsilon)$ and $(\alpha\gamma, \alpha\gamma, \varepsilon)$. As was the case for agent 1, we need to determine that agent 2 knows $\neg\gamma_E$ holds at $w$. This means that $\neg\gamma_E$ must hold in all worlds that look like $(\varepsilon,\varepsilon,\varepsilon)$ to agent 2. Note that because $\neg\gamma_E$ holds at all four possible worlds, $(\mathcal{I}^{\mathrm{DES}}, w) \models K_2\neg\gamma_E$.

A similar check can be performed at every other state in $\mathcal{I}^{\mathrm{DES}}$ to show that this system is Kripke-observable (summarized in Table I).

The joint knowledge-based protocol for events $\alpha$ and $\beta$ in this system is straightforward: both events are enabled by both agents at every state in $\mathcal{I}^{\mathrm{DES}}$. To realize the legal language, however, the control decisions for $\gamma$ must ensure that $\gamma$ is disabled before either agent sees any event occur and that $\gamma$ is disabled after $\alpha$ is generated by the plant. At $w = (\varepsilon, \varepsilon, \varepsilon)$ agent 2 does know to disable $\gamma$ and thus $\mathcal{KP}_2(\varepsilon, \gamma) = \mathit{disable}$. This action occurs at all global states where $w_2 = \varepsilon$, namely $(\alpha, \alpha, \varepsilon)$—which makes certain that $\alpha\gamma$ will not occur—and

TABLE II
A JOINT KNOWLEDGE-BASED PROTOCOL FOR $G$ AND $E$ AND EVENT $\gamma$ IN FIG. 1

| $w$ | $\mathcal{KP}_1(w_1, \gamma)$ | $\mathcal{KP}_2(w_2, \gamma)$ |
|---|---|---|
| $(\varepsilon, \varepsilon, \varepsilon)$ | *enable* | *disable* |
| $(\beta, \varepsilon, \beta)$ | *enable* | *enable* |
| $(\gamma, \gamma, \varepsilon)$ | *enable* | *disable* |
| $(\alpha, \alpha, \varepsilon)$ | *enable* | *disable* |
| $(\beta\gamma, \gamma, \beta)$ | *enable* | *enable* |
| $(\alpha\gamma, \alpha\gamma, \varepsilon)$ | *enable* | *disable* |
| $(\alpha\beta, \alpha, \beta)$ | *enable* | *enable* |
| $(\beta\alpha, \alpha, \beta)$ | *enable* | *enable* |
| $(\alpha\beta\gamma, \alpha\gamma, \beta)$ | *enable* | *enable* |
| $(\beta\alpha\gamma, \alpha\gamma, \beta)$ | *enable* | *enable* |

at $(\gamma, \varepsilon, \varepsilon)$ and $(\alpha\gamma, \alpha, \varepsilon)$. The "disable $\gamma$" action at the latter two states is irrelevant since the previous disablement actions will guarantee that we never reach these states. The complete set of control actions for event $\gamma$ is summarized in Table II. Note that as long as agent $i$ takes a "disable" action for some event at $w_i$, this action takes precedence over any other agent's "enable" action for the same event at any global states $w' \sim_i w$, thereby ensuring that the event is disabled in all possible worlds of agent $i$ at $w$.

## V. DISTRIBUTED OBSERVABILITY

Previously we considered what it means for an agent to know a fact; however, what does it mean for a group of agents to know a fact? To find a joint knowledge-based protocol that solves the decentralized control problem, we require that the interpreted system be Kripke-observable. But even if the system is not Kripke-observable, it may be the case that the group of agents has the *combined* knowledge to generate the correct control strategy. We call this notion of successfully pooling information to generate a control decision *distributed observability*.

Distributed observability is based on the concept of *distributed knowledge* (taken from [30]). Distributed knowledge is the weakest form of group knowledge: in essence, a group has distributed knowledge of $p$ if after combining all the knowledge of the group, $p$ holds. This amounts to taking the intersection of all sets of worlds each agent in the group considers possible at a given state in the system.

*Definition 3:* A group G of agents has distributed knowledge of $p \in \Phi$ at state $w$, denoted $(\mathcal{I}, w) \models D_{\mathrm{G}}p$, iff $(\mathcal{I}, w') \models p$ for all $w'$ where, for all agents $i$ in a group G, $w_i = w'_i$.

The modal operator $D_{\mathrm{G}}$ means "it is distributed knowledge among the agents in G" [3]. It could be the case that no individual agent knows $p$, but after combining their possible worlds (i.e., take the intersection of the possible worlds for the agents) the group of agents knows $p$—only if $p$ holds in all the remaining possible worlds of the "intersection."

Stronger assertions about group knowledge include "everyone in the group knows $p$" and common knowledge, where "everyone in the group knows $p$, everyone in the group knows that everyone in the group knows $p$" etc. We do not consider these states of knowledge here, but merely note that there exists a hierarchy of states of group knowledge for distributed systems.

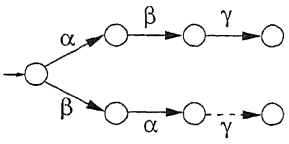


Fig. 6. A plant $G$ and legal automaton $E$. (Illegal transitions are marked with a dashed line.)

Distributed knowledge is the key to a concept we introduce, called *distributed observability*:

*Definition 4:* An interpreted system $\mathcal{I}^{\mathrm{DES}}$ has distributed observability with respect to a group of agents G if

$$\forall w \in \mathcal{I}^{\mathrm{DES}}, \quad \forall(\sigma_G, \sigma_E) \in \mathrm{R}_{\Sigma} \\ (\mathcal{I}^{\mathrm{DES}}, w) \models \neg\sigma_G \vee \sigma_E \vee D_G\neg\sigma_E.$$

That is, at all states in the interpreted system where an event would need to be disabled, there is distributed knowledge about whether to disable that event. Note that if $\mathcal{I}^{\mathrm{DES}}$ is Kripke-observable then by definition $\mathcal{I}^{\mathrm{DES}}$ has distributed observability since at every state, for each event in $\Sigma_c$, at least one agent (even before pooling knowledge) will know the correct control decision to make.

Intuitively, to solve a decentralized problem, even with communication, it would have to be the case that what one agent lacks in knowledge or information, the other can supply. Consider the case of sequences $t$ and $t'$ which look alike to both agents where $t$ is legal and $t'$ is illegal. If agent 1 were to communicate to agent 2 that it (agent 1) knows that one of $t$ or $t'$ has occurred, or if agent 2 were to communicate similar information to agent 1, communication will not help the agents make a control decision.

Using the knowledge framework we can exploit the possible-worlds model to identify system states that are indistinguishable to both agents (and where, therefore, information pooling would be of no help). Further, we can identify states where an agent's ability to make control decisions would be improved by communication.

We present two examples where the two agents have partial observation of a system: one where the pooling of possible worlds is not enough to achieve control and one where we believe that combined knowledge can achieve control.

### A. Example: When Pooling Knowledge Is Not Enough

In Fig. 6 the language generated by the legal automaton is not Kripke-observable[1] if $\Sigma_{1,o} = \{\alpha\}$, $\Sigma_{1,c} = \{\alpha, \gamma\}$, $\Sigma_{2,o} = \{\beta\}$, and $\Sigma_{2,c} = \{\beta\}$. When agent 1 sees $\alpha$ (equivalently, agent 2 sees $\beta$), it does not know whether or not $\alpha\beta$ or $\beta\alpha$ has occurred. Thus a control decision about $\gamma$ cannot be reached. The Kripke structure in Fig. 7 shows that $\mathcal{I}^{\mathrm{DES}}$ is not Kripke-observable. To see this, suppose the system were Kripke-observable. Then when $w = (\beta\alpha, \alpha, \beta)$, it must be that agent 1 knows $\neg\gamma_E$

[1]This example arose from discussions K. Rudie had with S. Lafortune, F. Lin, A. Overkamp and D. Teneketzis.

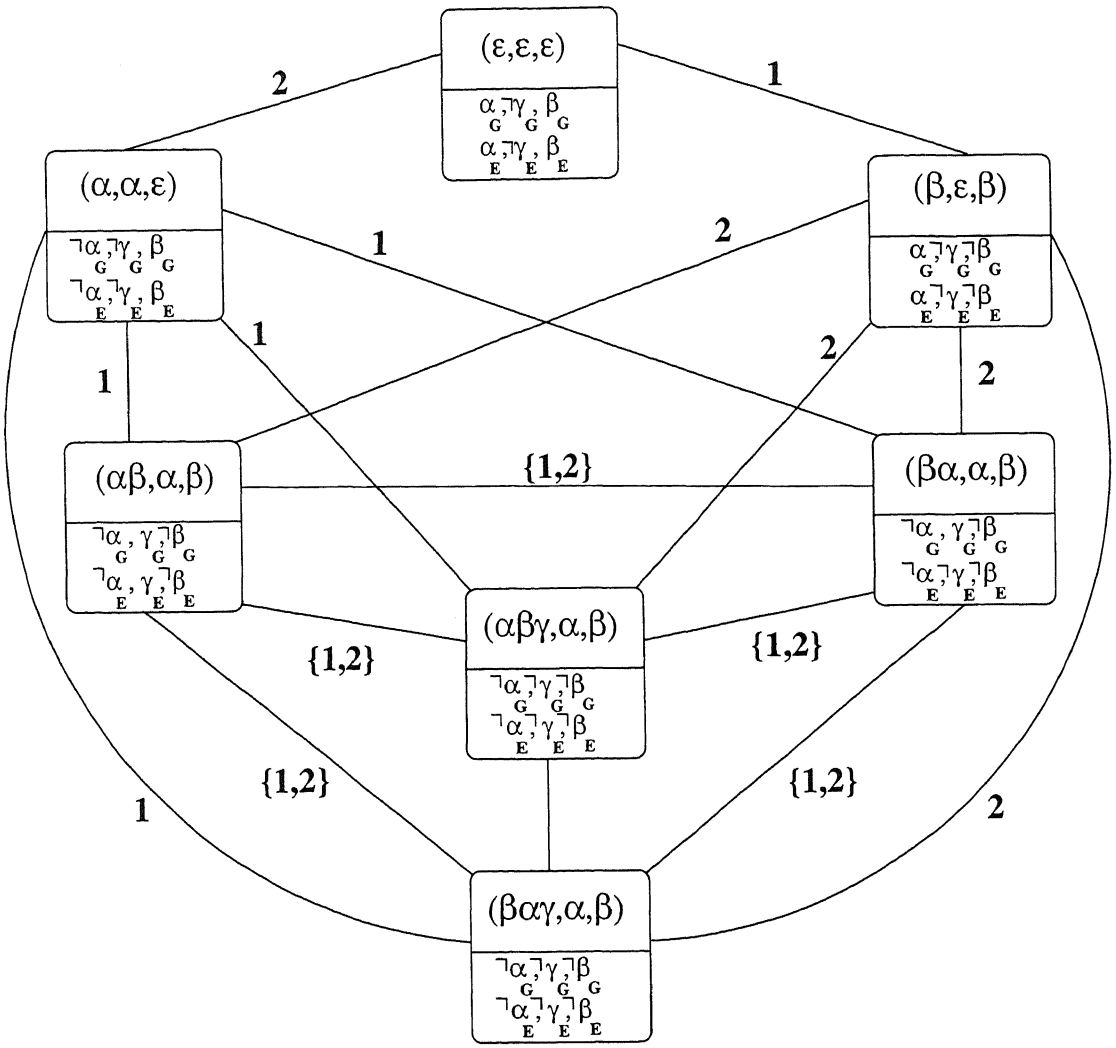


Fig. 7. The Kripke structure for the plant in Fig. 6.

in its set of possible worlds at $w$—since $(\mathcal{I}^{\mathrm{DES}}, w) \not\models \neg\gamma_G \vee \gamma_E$. In fact, this is not the case because $(\mathcal{I}^{\mathrm{DES}}, (\alpha\beta, \alpha, \beta)) \models \gamma_E$.

If both agents were to pool their knowledge at a state where agent 1 sees $\alpha$ and agent 2 sees $\beta$, so that the resulting possible worlds are $(\alpha\beta, \alpha, \beta)$, $(\beta\alpha, \alpha, \beta)$, and $(\alpha\beta\gamma, \alpha, \beta)$, the Kripke structure indicates that still there is no distributed knowledge about $\gamma_E$ for the same reasons that the system is not Kripke-observable: the conflicting truth value of $\gamma_E$ at states $(\alpha\beta, \alpha, \beta)$ and $(\beta\alpha, \alpha, \beta)$. That is, distributed observability is not satisfied.

When distributed observability is not satisfied this tells us that at some place, pooling information does not help. In this example, by the time agent 1 sees $\alpha$ and agent 2 sees $\beta$, the relative ordering of $\alpha$ and $\beta$ has been lost, i.e., even if at that point agent 2 were to tell agent 1 that it has seen $\beta$, that would not convey to agent 1 whether $\alpha\beta$ or $\beta\alpha$ had occurred. This would suggest that the agents must communicate prior to agent 1 seeing $\alpha$ and agent 2 seeing $\beta$.

One possible communication protocol could assume that an agent sends a query as soon as it is uncertain about whether to disable an event. Unfortunately, such a strategy is highly sensitive to small communication delays. In this example, because agent 2 sees and controls only $\beta$, there is never a situation when agent 2 is confused about its control decisions. So it never sends a query to agent 1. Agent 1 would need to submit a query when it sees $\alpha$. If only $\alpha$ has happened and agent 1 sends a query to agent 2, then it would appear that a decision about $\gamma$ can be made since the pooled information would indicate that the only possible world is $(\alpha, \alpha, \varepsilon)$. However, if $\beta$ had taken place before agent 2 receives the query, agent 1 would not know whether or not to disable $\gamma$ since it would not know if $\beta$ had occurred just before $\alpha$ or just after $\alpha$ happened. That is, the usefulness of pooled information depends on whether $\beta$ can occur after $\alpha$ has occurred but before agent 2 receives the query. In other words, even if a query results in a response, the solution is sensitive to the precise moment the query is received.

### B. Example: When Pooling Knowledge Is Enough

The legal language corresponding to the legal automaton illustrated in Fig. 8 is not Kripke-observable. Let $\Sigma_{1,o} = \{\alpha\}$,

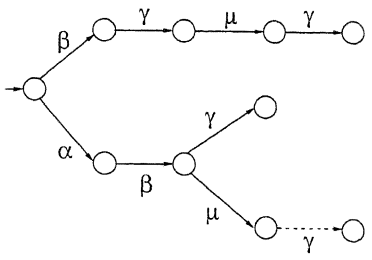


Fig. 8. A plant $G$ and legal automaton $E$. (Illegal transitions are marked with a dashed line.)

$\Sigma_{1,c} = \{\alpha, \gamma\}$, $\Sigma_{2,o} = \{\beta, \mu\}$, and $\Sigma_{2,c} = \{\beta, \mu, \gamma\}$. Upon observing $\alpha$, agent 1 (which sees only $\alpha$) does not know if $\alpha\beta$ or $\alpha\beta\mu$ has occurred and hence does not know whether or not to disable $\gamma$. Similarly by observing $\beta\mu$, agent 2 would not know whether $\alpha\beta\mu$ or $\beta\gamma\mu$ had occurred and could make no decision about disabling $\gamma$. However, we can check the Kripke structure and see that distributed observability is satisfied. That is, when an agent is unable to make a control decision, pooling information will help. In fact, an agent has more flexibility: to submit a query every time it is stuck is possibly unnecessary. If each agent has available to it a record of the history of its queries then it may be possible to deduce further information based on queries it has not received from the other agent. We illustrate this below. In addition, there remains the issue of when information should be pooled. There may be several states where pooling is beneficial and it may be possible to ascertain whether communication should be delayed to the last possible moment or should occur as early as possible.

In this example, as soon as agent 1 sees $\alpha$ it does not know whether or not to disable $\gamma$. At what state should it communicate or query agent 2 so that they can pool their knowledge? We can assume here that communication between agents occurs instantaneously so that as soon as one agent cannot continue, the other agent receives a query to pool knowledge. In this case, agent 2 can continue making control decisions until it sees $\beta\mu$ at which point it must submit a query to pool knowledge. On the other hand, if agent 2 sees $\beta\mu$ but has already received a query from agent 1 (after agent 1 sees $\alpha$), then agent 2 no longer needs to query as it knows that previously agent 1 did not know what to do about $\gamma$. Therefore agent 2 can deduce that $\alpha$ must have occurred.

## VI. Discussion

An interesting point to consider is how robust the structure of a problem is with respect to communication and hence a control strategy. For instance, in the example of Section V-A, the timing of the communication is critical for a solution: agent 1 must assume that if it sends a query to agent 2 immediately upon seeing $\alpha$ and the resulting possible worlds indicate that agent 2 has seen $\beta$, then $\beta$ has occurred before $\alpha$. In contrast with the example of Section V-B, once agent 1 has seen $\alpha$, communication after $\alpha$ occurs or after $\beta$ occurs or after $\mu$ occurs would all yield a solution to the problem. In this case, there may be characteristics of the problem structure (e.g., graph-theoretic properties) that make the timing of communication less critical for generating a solution.

Earlier we discussed what it meant for a joint knowledge-based protocol to solve a decentralized problem where $\mathcal{I}^{\mathrm{DES}}$ was Kripke-observable. If $\mathcal{I}^{\mathrm{DES}}$ is not Kripke-observable but has distributed observability, then the joint knowledge-based protocol $\mathcal{KP}$ is inadequate for solving the problem since communication or querying of other agents now must factor into a solution. Previously, control actions were based strictly on the knowledge acquired by observing traces of the plant. A revised definition of a knowledge-based protocol that exploits distributed observability must allow an agent to base its control decisions not only on its views of traces but also on the results of queries to other agents. A more "intelligent" protocol could be constructed by having each agent record all queries it receives. This information tells the queried agent what the other agent does not know, which, in turn, could allow the agent to draw logical inferences.

One of the disadvantages to the "sequence-based" version of the Kripke structure is that it is possible to generate an infinite-state Kripke structure. This difficulty can be circumvented in two ways: use a "limited-lookahead" strategy or develop a state-based knowledge model where a possible world is now a set of plant states an agent considers the system could be in. Another extension to the model would include systems with more than two agents.

The notion of distributed observability for interpreted systems provides a starting point from which we begin our understanding of how agents might communicate to solve a particular class of decentralized control problems. Introducing knowledge into the analysis of discrete-event systems provides a natural way to reason about what agents need to know to cooperatively solve a problem. To that end, we believe that using knowledge to determine when information should be pooled will lead to the development of knowledge-based protocols with communication for decentralized control problems.

## Acknowledgment

The authors gratefully acknowledge J. Halpern for discussions that clarified aspects of formal reasoning about knowledge, and in particular, for his observation that the original condition for Kripke observability was too strong. They also acknowledge interesting discussions with S. Lafortune that helped to fine-tune some technical details. Finally, the authors would like to thank the anonymous reviewers for their helpful suggestions.

## References

[1] P. J. Ramadge and W. M. Wonham, "Supervisory control of a class of discrete event processes," *SIAM J. Contr. Optimiz.*, vol. 25, no. 1, pp. 206–230, 1987.

[2] K. Rudie and W. M. Wonham, "Think globally, act locally: Decentralized supervisory control," *IEEE Trans. Automat. Contr.*, vol. 37, pp. 1692–1708, Nov. 1992.

[3] J. Y. Halpern and Y. Moses, "Knowledge and common knowledge in a distributed environment," *J. ACM*, vol. 37, no. 3, pp. 549–587, 1990.

[4] J. Y. Halpern and R. Fagin, "Modelling knowledge and action in distributed systems," *Distributed Computing*, vol. 3, pp. 159–177, 1989.

[5] R. J. Aumann, "Agreeing to disagree," *Annals Stat.*, vol. 4, no. 6, pp. 1236–1239, 1976.

[6] B. L. Lipman, "An axiomatic approach to the logical omniscience problem," in *Theoretical Aspects of Reasoning about Knowledge: Proc. Fifth Conf.*, R. Fagin, Ed., 1994, pp. 182–196.

[7] M. Burrows, M. Abadi, and R. Needham, "A logic of authentication," *ACM Trans. Comp. Syst.*, vol. 8, no. 1, pp. 18–36, 1990.
[8] V. Hadzilacos, "A knowledge-theoretic analysis of atomic commitment protocols," in *Proc. 6th ACM Symp. on Principles of Database Syst.*, 1987, pp. 129–134.
[9] R. I. Brafman and Y. Shoham, "Knowledge considerations in robotics and distribution of robotic tasks," in *Proc. 14th Int. Joint Conf on Artificial Intelligence*, 1995.
[10] J. Y. Halpern and L. D. Zuck, "A little knowledge goes a long way: Knowledge-based derivations and correctness proofs for a family of protocols," *J. ACM*, vol. 39, no. 3, 1992.
[11] J. S. Ostroff and W. M. Wonham, "A temporal logic approach to real time control," in *Proc. 24th Conf. Dec. and Contr.*, 1985, pp. 656–657.
[12] J. G. Thistle and W. M. Wonham, "Control problems in a temporal logic framework," *Int. J. Contr.*, vol. 44, pp. 943–976, 1986.
[13] A. Radiya and R. Sargent, "A logic-based foundation of discrete event modeling and simulation," *ACM Trans. Modeling Comp. Simul.*, vol. 4, no. 1, pp. 3–51, 1994.
[14] K. C. Wong and J. H. van Schuppen, "Decentralized supervisory control of discrete-event systems with communication," in *Proc. Int. Workshop on Discrete Event Syst.*, 1996, pp. 284–289.
[15] G. Barrett and S. Lafortune, "On the synthesis of communicating controllers with decentralized information structures for discrete-event systems," in *Proc. IEEE Conf. Dec. Contr.*, 1998, pp. 3281–3286.
[16] J. H. van Schuppen, "Decentralized supervisory control with information structures," in *Proc. of the Int. Workshop on Discrete Event Syst.*, 1998, pp. 36–41.
[17] R. Sengupta, "Diagnosis and communication in distributed systems," in *Proc. of the Int. Workshop on Discrete Event Syst.*, 1998, pp. 144–151.
[18] K. Rudie, S. Lafortune, and F. Lin, "Minimal communication in a distributed discrete-event control system," in *Proc. Amer Contr Conf.*, 1999, pp. 1965–1970.
[19] C. G. Cassandras, S. Lafortune, and G. J. Olsder, "Introduction to the modelling, control and optimization of discrete-event systems," *Trends in Contr.: A European Perspec.*, pp. 217–291, 1995.
[20] P. J. Ramadge and W. M. Wonham, "The control of discrete-event systems," *Proc. of the IEEE*, vol. 77, no. 1, pp. 81–98, 1989.
[21] J. G. Thistle, "Supervisory control of discrete event systems," *Math. Comp. Modelling*, vol. 11/12, no. 23, pp. 25–53, 1996.
[22] R. Cieslak, C. Desclaux, A. S. Fawaz, and P. Varaiya, "Supervisory control of discrete-event processes with partial observations," *IEEE Trans. Automat. Contr.*, vol. 33, no. 3, pp. 249–260, 1988.
[23] F. Lin and W. M. Wonham, "On observability of discrete-event systems," *Info. Sci.*, vol. 44, pp. 173–198, 1988.
[24] ——, "Decentralized control and coordination of discrete-event systems with partial observation," *IEEE Trans. Automat. Contr.*, vol. 35, pp. 1330–1337, Dec. 1990.
[25] K. Rudie and J. C. Willems, "The computational complexity of decentralized discrete-event control problems," *IEEE Trans. Automat. Contr.*, vol. 40, pp. 1313–1319, July 1995.
[26] S. Takai and S. Kodama, "Decentralized state feedback control of discrete event systems," *Syst. Contr. Lett.*, vol. 22, no. 5, pp. 369–375, 1994.
[27] Y. Willner and M. Heymann, "Supervisory control of concurrent discrete-event systems," *Int. J. Contr.*, vol. 54, no. 5, pp. 1143–1169, 1991.
[28] H. Cho and S. I. Marcus, "Supremal and maximal sublanguages arising in supervisor synthesis problems with partial observations," *Math. Syst. Theory*, vol. 22, pp. 177–211, 1989.
[29] S. Lafortune and E. Chen, "The infimal closed controllable superlanguage and its application in supervisory control," *IEEE Trans. Automat. Contr.*, vol. 35, pp. 398–405, Apr. 1990.
[30] R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi, *Reasoning About Knowledge*. Cambridge, MA: MIT Press, 1995.

**S. Laurie Ricker** received the B.Sc. degree in mathematics from Mount Allison University, Sackville, NB, Canada in 1987, the M.Sc. and Ph.D. degrees in computing and information science from Queen's University, Kingston, ON, Canada in 1993 and 2000, respectively.

Since September 1999 she has been an ERCIM fellow at INRIA-IRISA in Rennes, France, and at CWI in Amsterdam, The Netherlands. Her current research interests include failure diagnosis, discrete-event systems and verification of concurrent software.

**Karen Rudie** received the B.Sc. degree in mathematics and engineering from Queen's University, Kingston, Ontario, Canada in 1985, and the M.A.Sc. and Ph.D. degrees in electrical engineering from the University of Toronto, Ontario, Canada in 1988 and 1992, respectively.

From 1992 to 1993, she was a postdoctoral researcher at the Institute for Mathematics and its Applications, Minnesota. Since 1993, she has been with the Department of Electrical and Computer Engineering at Queen's University, where she is currently an Associate Professor. In 1999–2000, she was a visiting professor in Electrical Engineering and Computer Science at the University of Michigan. Her research interests include control of discrete-event systems and hybrid systems.

From 1996–1999, she served as an Associate Editor of IEEE TRANSACTIONS ON AUTOMATIC CONTROL.