Supervisor Localization of Discrete-Event Systems under Partial Observation*

Renyuan Zhang[1], Kai Cai[2], W.M. Wonham[3]

Abstract

Recently we developed *supervisor localization*, a top-down approach to distributed control of discrete-event systems. Its essence is the allocation of monolithic (global) control action among the local control strategies of individual agents. In this paper, we extend supervisor localization by considering partial observation; namely not all events are observable. Specifically, we employ the recently proposed concept of *relative observability* to compute a partial-observation monolithic supervisor, and then design a suitable localization procedure to decompose the supervisor into a set of local controllers. In the resulting local controllers, only observable events can cause state change. Further, to deal with large-scale systems, we combine the partial-observation supervisor localization with an efficient architectural synthesis approach: first compute a heterarchical array of partial-observation decentralized supervisors and coordinators, and then localize each of these supervisors/coordinators into local controllers.

Keywords

Discrete-event systems, supervisory control, supervisor localization, partial observation, automata

I. INTRODUCTION

In [1–5] we developed a top-down approach, called *supervisor localization*, to the distributed control of multi-agent discrete-event systems (DES). This approach first synthesizes a monolithic supervisor (or a heterarchical array of modular supervisors) assuming that *all* events can be observed, and then

decomposes the supervisor into a set of local controllers for the component agents. Localization creates a purely distributed control architecture in which each agent is controlled by its own local controller; this is particularly suitable for applications consisting of many autonomous components, e.g. multi-robot systems. Moreover, localization can significantly improve the comprehensibility of control logic, because the resulting local controllers typically have many fewer states than their parent supervisor. The assumption of full event observation, however, may be too strong in practice, since there often lacks enough sensors to observe every event.

In this paper, we extend supervisor localization to address the issue of partial observation. Our approach is as follows. We first synthesize a partial-observation monolithic supervisor using the concept of *relative observability* in [6]. Relative observability is generally stronger than observability [7, 8], weaker than normality [7, 8], and the supremal relatively observable (and controllable) sublanguage of a given language exists. The supremal sublanguage may be effectively computed [6], and then implemented by a partial-observation (feasible and nonblocking) supervisor [9, Chapter 6]. We then suitably extend the localization procedure in [1] to decompose the supervisor into local controllers for individual agents, and moreover prove that the derived local controlled behavior is equivalent to the monolithic one. We then suitably extend the localization procedure in [1] to decompose the supervisor into local controllers for individual agents, and moreover prove that the derived local controlled behavior is equivalent to the monolithic one.

The main contribution of this work is the novel combination of supervisor localization [1] with relative observability [6], which leads to a systematic approach to distributed control of DES under partial observation. The central concept of supervisor localization is *control cover* [1], which is defined on the state set of the full-observation supervisor. Under partial observation, we propose an extended control cover, which is defined on the state set of the partial-observation supervisor; roughly speaking, the latter corresponds to the *powerset* of the full-observation supervisor's state set. In this way, in the transition structure of the resulting local controllers, only observable events can lead to state changes. We design an extended localization algorithm for computing these local controllers. Moreover, to deal with large-scale systems, we combine the developed localization procedure with an efficient architectural synthesis approach [10]: first compute a heterarchical array of partial-observation decentralized supervisors and coordinators that collectively achieves globally feasible and nonblocking controlled behavior, and then localize each of these supervisors/coordinators into local controllers.

Our proposed localization procedure can in principle be used to construct local controllers from a partial-observation supervisor computed by any synthesis method. In particular, the algorithms in [11, 12] compute a nonblocking (maximally) observable sublanguage that is generally incomparable with the

supremal relatively observable sublanguage. The reason that we adopt relative observability is first of all that its generator-based computation of the supremal sublanguage is better suited for applying our localization algorithm; by contrast [12] uses a different transition structure called "bipartite transition system". Another important reason is that the computation of relative observability has been implemented and tested on a set of benchmark examples. This enables us to study distributed control under partial observation of more realistic systems; by contrast, the examples reported in [11, 12] are limited to academic ones.

We note that in [13–15] partial-observation supervisors are synthesized to enforce properties other than safety (satisfying imposed specifications) and nonblockingness; examples include diagnosability and opacity. Although we focus on safety and nonblockingness, for the sake of consistency with our previous work on full-observation localization that targets multi-agent distributed control problems, our developed localization procedure may be applied in the same way to decompose partial-observation supervisors with other properties. Thus if a partial-observation supervisor enforces specified properties determined only by the synthesized language, then those properties are preserved by localization and achieved collectively by the synthesized local controllers.

The paper is organized as follows. Section II reviews the supervisory control problem of DES under partial observation and formulates the partial-observation supervisor localization problem. Section III develops the partial-observation localization procedure, and Section IV presents the localization algorithm, which is illustrated by a Transfer Line example. Section V outlines a procedure combining the partial-observation localization with an efficient heterarchical supervisor synthesis to address distributed control of large-scale systems. Finally Section VI states our conclusions.

II.   PRELIMINARIES AND PROBLEM FORMULATION

A. Supervisory Control of DES under Partial Observation

A DES plant is given by a generator

$$\mathbf{G} = (Q, \Sigma, \delta, q_0, Q_m) \tag{1}$$

where Q is the finite state set; $q_0 \in Q$ is the initial state; $Q_m \subseteq Q$ is the subset of marker states; Σ is the finite event set; $\delta : Q \times \Sigma \to Q$ is the (partial) state transition function. In the usual way, δ is extended to $\delta : Q \times \Sigma^* \to Q$, and we write $\delta(q, s)!$ to mean that $\delta(q, s)$ is defined. Let Σ^* be the set of all finite strings, including the empty string ϵ. The *closed behavior* of $\mathbf{G}$ is the language

$$L(\mathbf{G}) = \{s \in \Sigma^* | \delta(q_0, s)!\}$$

3

and the *marked behavior* is

$$L_m(\mathbf{G}) = \{s \in L(\mathbf{G})|\delta(q_0, s) \in Q_m\} \subseteq L(\mathbf{G}).$$

A string s_1 is a *prefix* of a string s, written $s_1 \leq s$, if there exists s_2 such that $s_1 s_2 = s$. The *(prefix)* *closure* of $L_m(\mathbf{G})$ is $\overline{L_m(\mathbf{G})} := \{s_1 \in \Sigma^*|(\exists s \in L_m(\mathbf{G})) \ s_1 \leq s\}$. In this paper, we assume that $\overline{L_m(\mathbf{G})} = L(\mathbf{G})$; namely, $\mathbf{G}$ is *nonblocking*.

For supervisory control, the event set Σ is partitioned into Σ_c, the subset of controllable events that can be disabled by an external supervisor, and Σ_{uc}, the subset of uncontrollable events that cannot be prevented from occurring (i.e. $\Sigma = \Sigma_c \dot{\cup} \Sigma_{uc}$). For partial observation, Σ is partitioned into Σ_o, the subset of observable events, and Σ_{uo}, the subset of unobservable events (i.e. $\Sigma = \Sigma_o \dot{\cup} \Sigma_{uo}$). Bring in the *natural projection* $P : \Sigma^* \to \Sigma_o^*$ defined by

$$P(\epsilon) = \epsilon;$$

$$P(\sigma) = \begin{cases} \epsilon, & \text{if } \sigma \notin \Sigma_o, \\ \sigma, & \text{if } \sigma \in \Sigma_o; \end{cases} \tag{2}$$

$$P(s\sigma) = P(s)P(\sigma), \quad s \in \Sigma^*, \sigma \in \Sigma.$$

As usual, P is extended to $P : Pwr(\Sigma^*) \to Pwr(\Sigma_o^*)$, where $Pwr(\cdot)$ denotes powerset. Write $P^{-1} : Pwr(\Sigma_o^*) \to Pwr(\Sigma^*)$ for the *inverse-image function* of P.

For two languages $L_1 \subseteq \Sigma_1^*$ and $L_2 \subseteq \Sigma_2^*$, the *synchronous product* $L_1||L_2 \subseteq (\Sigma_1 \cup \Sigma_2)^*$ is defined according to $L_1||L_2 := P_1^{-1}L_1 \cap P_2^{-1}L_2$, where $P_i : (\Sigma_1 \cup \Sigma_2)^* \to \Sigma_i^*$ $(i = 1, 2)$ are the natural projections as defined in (2). For two generators $\mathbf{G}_i = (Q_i, \Sigma_i, \delta_i, q_{0,i}, Q_{m,i})$, $i = 1, 2$, let $L_m(\mathbf{G}_i)$ and $L(\mathbf{G}_i)$ be the marked and closed behaviors of $\mathbf{G}_i$ respectively; then the *synchronous product* $\mathbf{G} = (Q, \Sigma, \delta, q_0, Q_m)$ of $\mathbf{G}_1$ and $\mathbf{G}_2$, denoted by $\mathbf{G}_1||\mathbf{G}_2$, is constructed [9] to have marked behavior $L_m(\mathbf{G}) = L_m(\mathbf{G}_1)||L_m(\mathbf{G}_2)$ and closed behavior $L(\mathbf{G}) = L(\mathbf{G}_1)||L(\mathbf{G}_2)$. Synchronous product of more than two generators can be constructed similarly.

A *supervisory control* for $\mathbf{G}$ is any map $V : L(\mathbf{G}) \to \Gamma$, where $\Gamma := \{\gamma \subseteq \Sigma|\gamma \supseteq \Sigma_{uc}\}$. Then the closed-loop system is $V/\mathbf{G}$, with closed behavior $L(V/\mathbf{G})$ and marked behavior $L_m(V/\mathbf{G})$ [9]. Under partial observation $P : \Sigma^* \to \Sigma_o^*$, we say that V is *feasible* if

$$(\forall s, s' \in L(\mathbf{G})) \ P(s) = P(s') \Rightarrow V(s) = V(s'), \tag{3}$$

and V is *nonblocking* if $\overline{L_m(V/\mathbf{G})} = L(V/\mathbf{G})$.

It is well-known [7] that under partial observation, a feasible and nonblocking supervisory control V exists which synthesizes a (nonempty) sublanguage $K \subseteq L_m(\mathbf{G})$ if and only if K is both controllable

and observable [9]. When K is not observable, however, there generally does not exist the supremal observable (and controllable) sublanguage of K. Recently in [6], a new concept of *relative observability* is proposed, which is stronger than observability but permits the existence of the supremal relatively observable sublanguage.

Formally, a sublanguage $K \subseteq L_m(\mathbf{G})$ is *controllable* [9] if

$$\overline{K}\Sigma_{uc} \cap L(\mathbf{G}) \subseteq \overline{K}. \tag{4}$$

Let $C \subseteq L_m(\mathbf{G})$. A sublanguage $K \subseteq C$ is *relatively observable* with respect to C (or C-observable) if for every pair of strings $s, s' \in \Sigma^*$ that are lookalike under P, i.e. $P(s) = P(s')$, the following two conditions hold [6]:

$$\text{(i) } (\forall \sigma \in \Sigma)s\sigma \in \overline{K}, s' \in \overline{C}, s'\sigma \in L(\mathbf{G}) \Rightarrow s'\sigma \in \overline{K} \tag{5}$$

$$\text{(ii) } s \in K, s' \in \overline{C} \cap L_m(\mathbf{G}) \Rightarrow s' \in K \tag{6}$$

For $E \subseteq L_m(\mathbf{G})$ write $\mathcal{CO}(E)$ for the family of controllable and C-observable sublanguages of E. Then $\mathcal{CO}(E)$ is nonempty (the empty language $\emptyset$ belongs) and is closed under set union; $\mathcal{CO}(E)$ has a unique supremal element $\sup \mathcal{CO}(E)$ given by

$$\sup \mathcal{CO}(E) = \bigcup \{K | K \in \mathcal{CO}(E)\}$$

which may be effectively computed [6]. Note that since relative observability is weaker than normality [9], $\sup \mathcal{CO}(E)$ is generally larger than the normality counterpart.

B. Formulation of Partial-Observation Localization Problem

Let the plant $\mathbf{G}$ be comprised of N (> 1) component agents

$$\mathbf{G}_k = (Q_k, \Sigma_k, \delta_k, q_{0,k}, Q_{m,k}), \quad k = 1, ..., N.$$

Then $\mathbf{G}$ is the synchronous product of $\mathbf{G}_k$, k in the integer range $\{1, ..., N\}$, denoted as $[1, N]$, i.e. $\mathbf{G} = ||\{\mathbf{G}_k | k \in [1, N]\}$. Here, the Σ_k need not be pairwise disjoint. These agents are implicitly coupled through a specification language $E \subseteq \Sigma^*$ that imposes a constraint on the global behavior of $\mathbf{G}$ (E may itself be the synchronous product of multiple component specifications). For the plant $\mathbf{G}$ and the imposed specification E, let the generator $\mathbf{SUP} = (X, \Sigma, \xi, x_0, X_m)$ be such that

$$L_m(\mathbf{SUP}) := \sup \mathcal{CO}(E \cap L_m(\mathbf{G})). \tag{7}$$

and $L(\mathbf{SUP}) = \overline{L_m(\mathbf{SUP})}$ (i.e. **SUP** is nonblocking). We call **SUP** the *controllable and observable controlled behavior*.[1] To rule out the trivial case, we assume that $L_m(\mathbf{SUP}) \neq \emptyset$.

Now let $\alpha \in \Sigma_c$ be an arbitrary controllable event, which may or may not be observable. We say that a generator

$$\mathbf{LOC}_\alpha = (Y_\alpha, \Sigma_\alpha, \eta_\alpha, y_{0,\alpha}, Y_{m,\alpha}), \ \Sigma_\alpha \subseteq \Sigma_o \cup \{\alpha\}$$

is a *partial-observation local controller* for α if (i) $\mathbf{LOC}_\alpha$ enables/disables the event α (and only α) consistently with **SUP**, and (ii) if α is unobservable, then α-transitions are selfloops in $\mathbf{LOC}_\alpha$, i.e.

$$(\forall y \in Y_\alpha) \ \eta_\alpha(y, \alpha)! \Rightarrow \eta_\alpha(y, \alpha) = y.$$

Condition (i) means that for all $s \in \Sigma^*$ there holds

$$P_\alpha(s)\alpha \in L(\mathbf{LOC}_\alpha), \ s\alpha \in L(\mathbf{G}), \ s \in L(\mathbf{SUP})$$

$$\Leftrightarrow s\alpha \in L(\mathbf{SUP}) \tag{8}$$

where $P_\alpha : \Sigma^* \to \Sigma_\alpha^*$ is the natural projection. Condition (ii) requires that only observable events may cause a state change in $\mathbf{LOC}_\alpha$, i.e.

$$(\forall y, y' \in Y_\alpha, \forall \sigma \in \Sigma_\alpha) \ y' = \eta_\alpha(y, \sigma)!, \ y' \neq y \Rightarrow \sigma \in \Sigma_o. \tag{9}$$

This requirement is a distinguishing feature of a partial-observation local controller as compared to its full-observation counterpart in [1].

Note that the event set Σ_α of $\mathbf{LOC}_\alpha$ in general satisfies

$$\{\alpha\} \subseteq \Sigma_\alpha \subseteq \Sigma_o \cup \{\alpha\};$$

in typical cases, both subset containments are strict. The events in $\Sigma_\alpha \setminus \{\alpha\}$ may be viewed as communication events that are critical to achieve synchronization with other partial-observation local controllers (for other controllable events). The event set Σ_α is not fixed *a priori*, but will be determined as part of the localization result presented in the next section.

We now formulate the *Partial-Observation Supervisor Localization Problem*:

Construct a set of partial-observation local controllers $\{\mathbf{LOC}_\alpha \mid \alpha \in \Sigma_c\}$ such that the collective controlled behavior of these local controllers is equivalent to the controllable and observable controlled

---

[1]Note that **SUP**, defined over the entire event set Σ, is *not* a representation of a partial-observation supervisor. The latter can only have observable events as state transitions, according to the definition in Section III-A, below.

behavior **SUP** in (7) with respect to **G**, i.e.

$$L_m(\mathbf{G}) \cap \Big(\bigcap_{\alpha \in \Sigma_c} P_\alpha^{-1} L_m(\mathbf{LOC}_\alpha) \Big) = L_m(\mathbf{SUP})$$

$$L(\mathbf{G}) \cap \Big(\bigcap_{\alpha \in \Sigma_c} P_\alpha^{-1} L(\mathbf{LOC}_\alpha) \Big) = L(\mathbf{SUP}).$$

Having obtained a set of partial-observation local controllers, one for each controllable event, we can allocate each controller to the agent(s) owning the corresponding controllable event. Thereby we build for a multi-agent DES a nonblocking distributed control architecture under partial observation.

III. PARTIAL-OBSERVATION LOCALIZATION PROCEDURE

A. Uncertainty Set

Let $\mathbf{G} = (Q, \Sigma, \delta, q_0, Q_m)$ be the plant, $\Sigma_o \subseteq \Sigma$ the subset of observable events, and $P : \Sigma^* \to \Sigma_o^*$ the corresponding natural projection. Also let $\mathbf{SUP} = (X, \Sigma, \xi, x_0, X_m)$ be the controllable and observable controlled behavior (as defined in (7)).

Under partial observation, when a string $s \in L(\mathbf{SUP})$ occurs, what is observed is $P(s)$; namely, the events in Σ_{uo} ($= \Sigma \setminus \Sigma_o$) are erased. Hence two different strings s and s' may be lookalike, i.e. $P(s) = P(s')$. For $s \in L(\mathbf{SUP})$, let $U(s)$ be the subset of states that may be reached by some string s' that looks like s, i.e.

$$U(s) = \{x \in X | (\exists s' \in \Sigma^*) P(s) = P(s'), x = \xi(x_0, s')\}.$$

It is always true that the state $\xi(x_0, s) \in U(s)$. We call $U(s)$ the *uncertainty set* of the state $\xi(x_0, s)$ associated with string s. Let

$$\mathcal{U}(X) := \{U(s) \subseteq X | s \in L(\mathbf{SUP})\} \tag{10}$$

i.e. $\mathcal{U}(X)$ is the set of uncertainty sets of all states (associated with strings in $L(\mathbf{SUP})$) in X. The size of $\mathcal{U}(X)$ is $|\mathcal{U}(X)| \leq 2^{|X|}$ in general.

The transition function associated with $\mathcal{U}(X)$ is $\hat{\xi} : \mathcal{U}(X) \times \Sigma_o \to \mathcal{U}(X)$ given by

$$\hat{\xi}(U, \sigma) = \bigcup \{\xi(x, u_1 \sigma u_2) | x \in U, u_1, u_2 \in \Sigma_{uo}^*\}.^2 \tag{11}$$

If there exist $u_1, u_2 \in \Sigma_{uo}^*$ such that $\xi(x, u_1 \sigma u_2)!$, then $\hat{\xi}(U, \sigma)$ is defined, denoted as $\hat{\xi}(U, \sigma)!$. With $\mathcal{U}(X)$ and $\hat{\xi}$, define the *partial-observation monolithic supervisor* [9, 16]

$$\mathbf{SUPO} = (\mathcal{U}(X), \Sigma_o, \hat{\xi}, U_0, \mathcal{U}_m) \tag{12}$$

---

[2]Let $U = U(s)$ for some string $s \in L(\mathbf{SUP})$; then by definition of uncertainty set, $\bigcup\{\xi(x, u_1 \sigma u_2) | x \in U, u_1, u_2 \in \Sigma_{uo}^*\} = \{\xi(x_0, s' u_1 \sigma u_2) | \xi(x_0, s')!, Ps' = Ps, u_1, u_2 \in \Sigma_{uo}^*\} = \{\xi(x_0, s'') | \xi(x_0, s'')!, Ps'' = P(s\sigma)\} = U(s\sigma) \in \mathcal{U}(X)$.
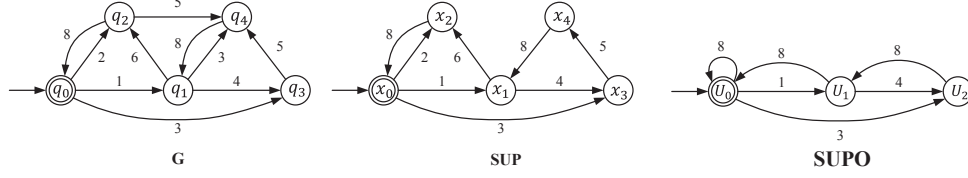
Fig. 1. Plant **G**, controllable and observable controlled behavior **SUP**, partial-observation monolithic supervisor **SUPO**, $\Sigma_c = \{1, 3, 5\}$, and $\Sigma_o = \{1, 3, 4, 8\}$. Inspecting the transition diagram of **SUP**, the uncertainty sets are $U(\epsilon) = \{x_0, x_2\}$, $U(1) = \{x_1, x_2\}$, $U(1.4) = \{x_3, x_4\}$, and for the remainder of strings $s \in L(\mathbf{SUP})$, $U(s)$ equals one of the above; namely, the set $\mathcal{U}(X)$ of uncertainty sets of **SUP** is $\mathcal{U}(X) = \{\{x_0, x_2\}, \{x_1, x_2\}, \{x_3, x_4\}\}$. For later reference, denote $U_0 = \{x_0, x_2\}$, $U_1 = \{x_1, x_2\}$, and $U_2 = \{x_3, x_4\}$. With $\mathcal{U}(X) = \{U_0, U_1, U_2\}$, we find $\hat{\xi} : \mathcal{U}(X) \times \Sigma_o \to \mathcal{U}(X)$, and then the partial-observation supervisor **SUPO** is constructed. Note that in **SUPO**, only observable events lead to state changes. Notation: a circle with $\to$ denotes the initial state, and a double circle denotes a marker state; this notation will be used throughout.

where $U_0 = U(\epsilon)$ and $\mathcal{U}_m = \{U \in \mathcal{U}(X) | U \cap X_m \neq \emptyset\}$. It is known [9, 16] that $L(\mathbf{SUPO}) = P(L(\mathbf{SUP}))$ and $L_m(\mathbf{SUPO}) = P(L_m(\mathbf{SUP}))$.[3] For an example of uncertainty set and partial-observation monolithic supervisor, see Fig. 1.

Now let $U \in \mathcal{U}(X)$, $x \in U$ be any state in **SUP** and $\alpha \in \Sigma_c$ be a controllable event. We say that (1) α is *enabled* at $x \in U$ if

$$\xi(x, \alpha)!;$$

(2) α is *disabled* at $x \in U$ if

$$\neg\xi(x, \alpha)! \text{ and}(\exists s \in \Sigma^*)\xi(x_0, s) = x \ \& \ \hat{\xi}(U_0, Ps) = U \ \& \ \delta(q_0, s\alpha)!;$$

(3) α is *not defined* at $x \in U$ if

$$\neg\xi(x, \alpha)! \text{ and}(\forall s \in \Sigma^*)\xi(x_0, s) = x \ \& \ \hat{\xi}(U_0, Ps) = U \Rightarrow \neg\delta(q_0, s\alpha)!.$$

Under partial observation, the control actions after string $s \in L(\mathbf{SUP})$ do not depend on the individual state $\xi(x_0, s) \in X$, but just on the uncertainty set $U(s) \in \mathcal{U}(X)$ (i.e. the state of **SUPO**). Since the language $L_m(\mathbf{SUP})$ is (relatively) observable, the following is true.

Lemma 1. *Given* **SUP** *in (7), let* $U \in \mathcal{U}(X)$, $x \in U$, *and* $\alpha \in \Sigma_c$. *If* α *is enabled at* x, *then for all* $x' \in U$, *either* α *is also enabled at* x', *or* α *is not defined at* x'. *On the other hand, if* α *is disabled at* x, *then for all* $x' \in U$, *either* α *is also disabled at* x', *or* α *is not defined at* x'.

---

[3] To ensure that the partial-observation supervisor **SUPO** is feasible, it is necessary to selfloop each state U of $\hat{\xi}$ by $\sigma = \Sigma_{uo}$ exactly when there exists $x \in U$ such that $\xi(x, \sigma)!$.

Proof. By $x \in U \in \mathcal{U}(X)$, there exists $s \in L(\mathbf{SUP})$ such that $\xi(x_0, s) = x$ and $U(s) = U$. Suppose that $\sigma \in \Sigma_c$ is enabled at $x \in U$, i.e. $\xi(x, \sigma)!$; it follows that $\xi(x, s\sigma)!$, i.e. $s\sigma \in L(\mathbf{SUP})$. Now let $x' \in U = U(s)$. According to the subset construction algorithm, there must exist $s' \in L(\mathbf{SUP})$ such that $\xi(x_0, s') = x'$ (i.e. $s' \in L(\mathbf{SUP})$) and $Ps' = Ps$. At state x', either (i) $\xi(x', \sigma)!$, or (ii) $\neg\xi(x', \sigma)!$. Case (i) means that σ is enabled at $x' \in U$. In case (ii), we claim that $s'\sigma \notin L(\mathbf{G})$, i.e. σ is not defined at $x' \in U$. To see this, assume on the contrary that $s'\sigma \in L(\mathbf{G})$. Then we have $Ps' = Ps$, $s' \in L(\mathbf{SUP})$, $s'\sigma \in L(\mathbf{G})$, $s'\sigma \notin L(\mathbf{SUP})$, and $s\sigma \in L(\mathbf{SUP})$. This implies that $L_m(\mathbf{SUP})$ is not observable, which is a contradiction to the definition of $L_m(\mathbf{SUP})$ in (5). Therefore, in case (ii), σ is not defined at $x' \in U$ after all.

The second statement can be proved by a similar argument. $\qquad\square$

B. Localization Procedure

The procedure of partial-observation localization proceeds similarly to [1], but is based on the set $\mathcal{U}(X)$ of the uncertainty sets and its associated transition function $\hat{\xi}$, i.e. based on the partial-observation monolithic supervisor $\mathbf{SUPO}$ in (12).

First, consider the following four functions which capture the control and marking information on the uncertainty sets. Fix a controllable event $\alpha \in \Sigma_c$. Define $E_\alpha : \mathcal{U}(X) \to \{0, 1\}$ according to

$$(\forall U \in \mathcal{U}(X)) \ E_\alpha(U) = \begin{cases} 1, & \text{if } (\exists x \in U)\xi(x, \alpha)!, \\ 0, & \text{otherwise.} \end{cases}$$

Thus $E_\alpha(U) = 1$ if event α is enabled at some state $x \in U$. Then by Lemma 1 at any other state $x' \in U$, α is either enabled or not defined. Also define $D_\alpha : \mathcal{U}(X) \to \{0, 1\}$ according to

$$(\forall U \in \mathcal{U}(X))$$

$$D_\alpha(U) = \begin{cases} 1, & \text{if } (\exists x \in U)\neg\xi(x, \alpha)! \ \&(\exists s \in \Sigma^*) \\ & \quad \xi(x_0, s) = x \ \& \ \hat{\xi}(U_0, Ps) = U \ \& \ \delta(q_0, s\alpha)!, \\ 0, & \text{otherwise.} \end{cases}$$

Hence $D_\alpha(U) = 1$ if α is disabled at some state $x \in U$. Again by Lemma 1 at any other state $x' \in U$, α is either disabled or not defined.

Consider the example displayed in Fig. 1. The control actions include (i) enabling events 1, 3 at state x_0, event 5 at state x_3; and (ii) disabling event 3 at state x_1, event 5 at state x_2. For the uncertainty set $U_0 = \{x_0, x_2\}$, $E_3(U_0) = 1$ because event 3 is enabled at state $x_0 \in U_0$; note that event 3 is not

9

defined at the other state $x_2 \in U_0$. For the uncertainty set $U_1 = \{x_1, x_2\}$, $D_3(U_1) = 1$ because event 3 is disabled at state $x_1 \in U_1$; also note that event 3 is not defined at state $x_2 \in U_1$.

Next, define $M : \mathcal{U}(X) \to \{0, 1\}$ according to

$$(\forall U \in \mathcal{U}(X)) \ M(U) = \begin{cases} 1, & \text{if } U \in \mathcal{U}_m, \\ 0, & \text{otherwise.} \end{cases}$$

Thus $M(U) = 1$ if U is marked in **SUPO** (i.e. U contains a marker state of **SUP**). Finally define $T : \mathcal{U}(X) \to \{0, 1\}$ according to

$$(\forall U \in \mathcal{U}(X))$$

$$T(U) = \begin{cases} 1, & \text{if } (\exists s \in \Sigma^*)\xi(x_0, s) \in U \ \& \\ & \hat{\xi}(U_0, Ps) = U \ \& \ \delta(q_0, s) \in Q_m, \\ 0, & \text{otherwise.} \end{cases}$$

So $T(U) = 1$ if U contains some state that corresponds (via a string s) to a marker state of **G**.

With the above four functions capturing control and marking information of the uncertainty sets in $\mathcal{U}(X)$, we define the *control consistency relation* $\mathcal{R}_\alpha \subseteq \mathcal{U}(X) \times \mathcal{U}(X)$ as follows.

Definition 1. For $U, U' \in \mathcal{U}(X)$, we say that U and U' are *control consistent* with respect to α, written $(U, U') \in \mathcal{R}_\alpha$, if

(i) $E_\alpha(U) \cdot D_\alpha(U') = 0 = E_\alpha(U') \cdot D_\alpha(U)$,

(ii) $T(U) = T(U') \Rightarrow M(U) = M(U')$.

Thus a pair of uncertainty sets (U, U') satisfies $(U, U') \in \mathcal{R}_\alpha$ if (i) event α is enabled at at least one state of U, but is not disabled at any state of U', and vice versa; (ii) U, U' both contain marker states of **SUP** (resp. both do not contain) provided that they both contain states corresponding to some marker states of **G** (resp. both do not contain).

For example, in Fig. 1, for event 3 we have:

	E_3	D_3	M	T
U_0	1	0	1	1
U_1	0	1	0	0
U_2	0	0	0	0

Hence $(U_0, U_2) \in \mathcal{R}_3$, $(U_2, U_1) \in \mathcal{R}_3$, and $(U_0, U_1) \notin \mathcal{R}_3$. From this example we see that $\mathcal{R}_\alpha$ is generally not transitive, and thus not an equivalence relation. This fact leads to the following definition of a *partial-observation control cover*.

10

Definition 2. Let I be some index set, and $\mathcal{C}_\alpha = \{\mathcal{U}_i \subseteq \mathcal{U}(X) | i \in I\}$ be a cover on $\mathcal{U}(X)$. We say that $\mathcal{C}_\alpha$ is a *partial-observation control cover* with respect to α if

(i) $(\forall i \in I, \forall U, U' \in \mathcal{U}_i)\ (U, U') \in \mathcal{R}_\alpha,$

(ii) $(\forall i \in I, \forall \sigma \in \Sigma_o)(\exists U \in \mathcal{U}_i)\ \hat{\xi}(U, \sigma)! \Rightarrow \big[(\exists j \in I)$

$$(\forall U' \in \mathcal{U}_i)\ \hat{\xi}(U', \sigma)! \Rightarrow \hat{\xi}(U', \sigma) \in \mathcal{U}_j\big].$$

A partial-observation control cover $\mathcal{C}_\alpha$ lumps the uncertainty sets $U \in \mathcal{U}(X)$ into (possibly overlapping) *cells* $\mathcal{U}_i \in \mathcal{C}_\alpha$, $i \in I$, according to (i) the uncertainty sets U that reside in the same cell $\mathcal{U}_i$ must be pairwise control consistent, and (ii) for every observable event $\sigma \in \Sigma_o$, the uncertainty set that is reached from any uncertainty set $U' \in \mathcal{U}_i$ by a one-step transition σ must be covered by the same cell $\mathcal{U}_j$. Inductively, two uncertainty sets U and U' belong to a common cell of $\mathcal{C}_\alpha$ if and only if U and U' are control consistent, and two future uncertainty sets that can be reached respectively from U and U' by a given observable string are again control consistent.

The partial-observation control cover $\mathcal{C}_\alpha$ differs from its counterpart in [1] in two aspects. First, $\mathcal{C}_\alpha$ is defined on $\mathcal{U}(X)$, not on X; this is due to state uncertainty caused by partial observation. Second, in condition (ii) of $\mathcal{C}_\alpha$ only observable events in Σ_o are considered, not Σ; this is to generate partial-observation local controllers whose state transitions are triggered only by observable events. We call $\mathcal{C}_\alpha$ a *partial-observation control congruence* if $\mathcal{C}_\alpha$ happens to be a partition on $\mathcal{U}(X)$, namely its cells are pairwise disjoint.

Having defined a partial-observation control cover $\mathcal{C}_\alpha$ on $\mathcal{U}(X)$, we construct a generator $\mathbf{J}_\alpha = (I, \Sigma_o, \zeta_\alpha, i_0, I_m)$ defined over Σ_o and a control function $\psi_\alpha : I \to \{0, 1\}$ as follows:

(i) $i_0 \in I$ such that $(\exists U \in \mathcal{U}_{i_0}) x_0 \in U$; \hfill (13)

(ii) $I_m := \{i \in I | (\exists U \in \mathcal{U}_i) X_m \cap U \neq \emptyset\};$ \hfill (14)

(iii) $\zeta_\alpha : I \times \Sigma_o \to I$ with $\zeta_\alpha(i, \sigma) = j$

$\quad$ if $(\exists U \in \mathcal{U}_i)\ \hat{\xi}(U, \sigma) \in \mathcal{U}_j$; \hfill (15)

(iv) $\psi_\alpha(i) = 1$ iff $(\exists U \in \mathcal{U}_i)\ E_\alpha(U) = 1$. \hfill (16)

The control function $\psi_\alpha(i) = 1$ means that event α is enabled at state i of $\mathbf{J}_\alpha$. Note that owing to cell overlapping, the choices of i_0 and ζ_α may not be unique, and consequently $\mathbf{J}_\alpha$ may not be unique. In that case we pick an arbitrary instance of $\mathbf{J}_\alpha$.

Finally we define the *partial-observation local controller* $\mathbf{LOC}_\alpha = (Y_\alpha, \Sigma_\alpha, \eta_\alpha, y_{0,\alpha}, Y_{m,\alpha})$ as follows.

(i) $Y_\alpha = I$, $y_{0,\alpha} = i_0$, and $Y_{m,\alpha} = I_m$. Thus the control function ψ_α is $\psi_\alpha : Y_\alpha \to \{0,1\}$.

(ii) $\Sigma_\alpha = \{\alpha\} \cup \Sigma_{com,\alpha}$, where

$$\Sigma_{com,\alpha} := \{\sigma \in \Sigma_o \setminus \{\alpha\} \mid (\exists i, j \in I) i \neq j, \ \zeta_\alpha(i, \sigma) = j\} \tag{17}$$

Thus $\Sigma_{com,\alpha}$ is the set of observable events that are not merely selfloops in $\mathbf{J}_\alpha$. It holds by definition that $\{\alpha\} \subseteq \Sigma_\alpha \subseteq \Sigma_o \cup \{\alpha\}$, and $\Sigma_{com,\alpha}$ contains the events of other local controllers that need to be communicated to $\mathbf{LOC}_\alpha$.

(iii) If $\alpha \in \Sigma_o$, then $\eta_\alpha := \zeta_\alpha|_{Y_\alpha \times \Sigma_\alpha} : Y_\alpha \times \Sigma_\alpha \to Y_\alpha$, i.e. η_α is the restriction of ζ_α to $Y_\alpha \times \Sigma_\alpha$. If $\alpha \in \Sigma_{uo}$, first obtain $\eta_\alpha := \zeta_\alpha|_{Y_\alpha \times \Sigma_\alpha}$ and then add α-selfloops $\eta_\alpha(y, \alpha) = y$ to those $y \in Y_\alpha$ with $\psi_\alpha(y) = 1$.

Lemma 2. *The generator $\mathbf{LOC}_\alpha$ is a partial-observation local controller for α, i.e. (8) and (9) hold.*

We postpone the proof of Lemma 2 after our main result, Theorem 1, in the next subsection.

Consider again the example displayed in Fig. 1. We construct a partial-observation local controller $\mathbf{LOC}_5$ for the unobservable controllable event 5. For event 5 we have:

	E_5	D_5	M	T
U_0	0	1	1	1
U_1	0	1	0	0
U_2	1	0	0	0

Hence $(U_0, U_1) \in \mathcal{R}_5$, $(U_0, U_2) \notin \mathcal{R}_5$, and $(U_1, U_2) \notin \mathcal{R}_5$. Further, because $\hat{\xi}(U_0, 8) = \hat{\xi}(U_1, 8) = U_0$, U_0 and U_1 can be put in the same cell; but U_0 and U_2 cannot, nor can U_1 and U_2. So we get a partial-observation control cover $\mathcal{C}_5 = \{\{U_0, U_1\}, \{U_2\}\}$. From this control cover, we construct a generator $\mathbf{J}_5$ as shown in Fig. 2, and a control function ψ_5 such that $\psi_5(\{U_0, U_1\}) = 0$ and $\psi_5(\{U_2\}) = 1$ because $E_5(U_2) = 1$. Finally the partial-observation local controller $\mathbf{LOC}_5$ is constructed from the generator $\mathbf{J}_5$ by adding the 5-selfloop at state y_1 because $\psi_5(\{U_2\}) = 1$ and 5 is unobservable, and removing event 1 since it is merely a selfloop in $\mathbf{J}_5$ (see Fig. 2).

C. Main Result

By the same procedure as above, we construct a set of partial-observation local controllers $\mathbf{LOC}_\alpha$, one for each controllable event $\alpha \in \Sigma_c$. We shall verify that these local controllers collectively achieve the same controlled behavior as represented by $\mathbf{SUP}$ in (7).
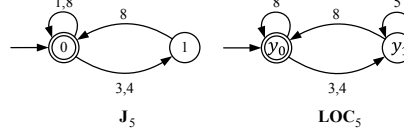
Fig. 2. Generator $\mathbf{J}_5$ and partial-observation local controller $\mathbf{LOC}_5$. In $\mathbf{J}_5$, state 0 corresponds to cell $\{U_0, U_1\}$ of the control cover $\mathcal{C}_5 = \{\{U_0, U_1\}, \{U_2\}\}$ while state 1 corresponds to cell $\{U_2\}$. From $\mathbf{J}_5$ to $\mathbf{LOC}_5$, (i) the 5-selfloop at state y_1 is added because $\psi_5(\{U_2\}) = 1$ and event 5 is unobservable, and (ii) event 1 is removed since it is merely a selfloop in $\mathbf{J}_5$ and thus its occurrences will not affect the enablement/disablement of event 5.

Theorem 1. *The set of partial-observation local controllers $\{\mathbf{LOC}_\alpha | \alpha \in \Sigma_c\}$ is a solution to the Partial-Observation Supervisor Localization Problem, i.e.*

$$L(\mathbf{G}) \cap L(\mathbf{LOC}) = L(\mathbf{SUP}) \tag{18}$$

$$L_m(\mathbf{G}) \cap L_m(\mathbf{LOC}) = L_m(\mathbf{SUP}) \tag{19}$$

where $L(\mathbf{LOC}) = \bigcap\limits_{\alpha \in \Sigma_c} P_\alpha^{-1} L(\mathbf{LOC}_\alpha)$ and $L_m(\mathbf{LOC}) = \bigcap\limits_{\alpha \in \Sigma_c} P_\alpha^{-1} L_m(\mathbf{LOC}_\alpha)$.

Proof: First we show that $L_m(\mathbf{SUP}) \subseteq L_m(\mathbf{G}) \cap L_m(\mathbf{LOC})$. It suffices to show $(\forall \alpha \in \Sigma_c)\, L_m(\mathbf{SUP}) \subseteq P_\alpha^{-1} L_m(\mathbf{LOC}_\alpha)$. Let $\alpha \in \Sigma_c$ and $s \in L_m(\mathbf{SUP})$; we must show $s \in P_\alpha^{-1} L_m(\mathbf{LOC}_\alpha)$. Write $Ps = \sigma_0, ..., \sigma_n$; then $Ps \in PL_m(\mathbf{SUP})$. According to the definition of uncertainty set, there exist $U_0, ..., U_n \in \mathcal{U}(X)$ such that

$$\hat{\xi}(U_j, \sigma_j) = U_{j+1}, j = 0, ..., n - 1.$$

Then by the definition of $\mathcal{C}_\alpha$ and ζ_α, for each $j = 0, ..., n - 1$, there exist $i_j, i_{j+1} \in I$ such that

$$U_j \in \mathcal{U}_{i_j} \;\&\; U_{j+1} \in \mathcal{U}_{i_{j+1}} \;\&\; \zeta_\alpha(i_j, \sigma_j) = i_{j+1}.$$

So $\zeta_\alpha(i_0, \sigma_0...\sigma_n)!$, i.e. $\zeta_\alpha(i_0, Ps)!$. Let $i_n = \zeta_\alpha(i_0, Ps)$; then $U(Ps) \in \mathcal{U}_{i_n}$, and thus $\xi(x_0, s) \in U(Ps) \cap X_m$. So $i_n \in I_m$, i.e. $Ps \in L_m(\mathbf{J}_\alpha)$. Let $P_\alpha' : \Sigma_o^* \to \Sigma_\alpha^*$ be the natural projection as defined in (2); then $P_\alpha(s) = P_\alpha'(Ps) \in P_\alpha' L_m(\mathbf{J}_\alpha) = L_m(\mathbf{LOC}_\alpha)$. Hence $s \in P_\alpha^{-1} L_m(\mathbf{LOC}_\alpha)$.

13

Now that we have shown $L_m(\mathbf{SUP}) \subseteq L_m(\mathbf{G}) \cap L_m(\mathbf{LOC})$, it follows that

$$
\begin{aligned}
L(\mathbf{SUP}) &= \overline{L_m(\mathbf{SUP})} \\
&\subseteq \overline{L_m(\mathbf{G}) \cap L_m(\mathbf{LOC})} \\
&\subseteq \overline{L_m(\mathbf{G})} \cap \overline{L_m(\mathbf{LOC})} \\
&\subseteq L(\mathbf{G}) \cap \bigcap_{\alpha \in \Sigma_c} P_\alpha^{-1} \overline{L_m(\mathbf{LOC}_\alpha)} \\
&\subseteq L(\mathbf{G}) \cap \bigcap_{\alpha \in \Sigma_c} P_\alpha^{-1} L(\mathbf{LOC}_\alpha) \\
&= L(\mathbf{G}) \cap L(\mathbf{LOC})
\end{aligned}
$$

so $L(\mathbf{SUP}) \subseteq L(\mathbf{G}) \cap L(\mathbf{LOC})$.

Next, we prove $L(\mathbf{G}) \cap L(\mathbf{LOC}) \subseteq L(\mathbf{SUP})$, by induction on the length of strings.

For the base case, as it was assumed that $L_m(\mathbf{SUP})$ is nonempty, it follows that the languages $L(\mathbf{G})$, $L(\mathbf{LOC})$ and $L(\mathbf{SUP})$ are all nonempty, and as they are closed, the empty string ϵ belongs to each.

For the inductive step, suppose that $s \in L(\mathbf{G}) \cap L(\mathbf{LOC})$ implies $s \in L(\mathbf{SUP})$, and $s\alpha \in L(\mathbf{G}) \cap L(\mathbf{LOC})$ for an arbitrary event $\alpha \in \Sigma$; we must show that $s\alpha \in L(\mathbf{SUP})$. If $\alpha \in \Sigma_u$, then $s\alpha \in L(\mathbf{SUP})$ because $L_m(\mathbf{SUP})$ is controllable. Otherwise, we have $\alpha \in \Sigma_c$ and there exists a partial-observation local controller $\mathbf{LOC}_\alpha$ for α. It follows from $s\alpha \in L(\mathbf{LOC})$ that $s\alpha \in P_\alpha^{-1} L(\mathbf{LOC}_\alpha)$ and $s \in P_\alpha^{-1} L(\mathbf{LOC}_\alpha)$. So $P_\alpha(s\alpha) \in L(\mathbf{LOC}_\alpha)$ and $P_\alpha(s) \in L(\mathbf{LOC}_\alpha)$, namely, $\eta_\alpha(y_0, P_\alpha(s\alpha))!$ and $\eta_\alpha(y_0, P_\alpha(s))!$. Let $y := \eta_\alpha(y_0, P_\alpha(s))$; then $\eta_\alpha(y, \alpha)!$ (because $\alpha \in \Sigma_\alpha$). Since α may be observable or unobservable, we consider the following two cases.

Case (1) $\alpha \in \Sigma_{uo}$. It follows from the construction (iii) of $\mathbf{LOC}_\alpha$ that $\eta_\alpha(y, \alpha)!$ implies that for the state $i \in I$ of the generator $\mathbf{J}_\alpha$ corresponding to y (i.e. $i = \zeta_\alpha(i_0, P(s))$), there holds $\psi_\alpha(i) = 1$. By the definition of ψ_α in (16), there exists an uncertainty set $U \in \mathcal{U}_i$ such that $E_\alpha(U) = 1$. Let $U' = \hat{\xi}(U_0, Ps)$; by (15) and $i = \zeta_\alpha(i_0, Ps)$, $U' \in \mathcal{U}_i$. According to (11), $\xi(x_0, s) \in U'$. Since U and U' belong to the same cell $\mathcal{U}_i$, by the definition of partial-observation control cover they must be control consistent, i.e. $(U, U') \in \mathcal{R}_\alpha$. Thus $E_\alpha(U) \cdot D_\alpha(U') = 0$, which implies $D_\alpha(U') = 0$. The latter means that for all states $x \in U'$, either (i) $\xi(x, \alpha)!$ or (ii) for all $t \in \Sigma^*$ with $\xi(x_0, t) = x$, $\delta(q_0, t\alpha)$ is not defined. Note that (ii) is impossible for $\xi(x_0, s) \in U'$, because $s\alpha \in L(\mathbf{G})$. Thus by (i), $\xi(\xi(x_0, s), \alpha)!$, and therefore $s\alpha \in L(\mathbf{SUP})$.

Case (2) $\alpha \in \Sigma_o$. In this case, for the state $i \in I$ of the generator $\mathbf{J}_\alpha$ corresponding to y (i.e. $i = \zeta_\alpha(i_0, P(s))$), there holds $\zeta_\alpha(i, \alpha)!$. By the definition of ζ_α in (15), there exists an uncertainty set
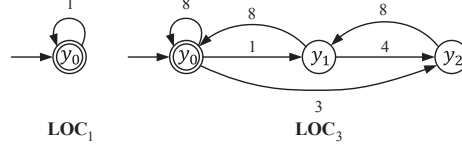
14

Fig. 3. Partial-observation local controllers $\mathbf{LOC}_1$ and $\mathbf{LOC}_3$. For $\mathbf{LOC}_1$, since event 1 is not disabled at any uncertainty set, the partial-observation control cover $\mathcal{C}_1 = \{\{U_0, U_1, U_2\}\}$ and $\mathbf{LOC}_1$ has just one state at which event 1 is enabled. For $\mathbf{LOC}_3$, note that the uncertainty sets U_0 and U_2 (corresponding to states y_0 and y_2 respectively) are not in a common cell of $\mathcal{C}_3$ even though U_0 and U_2 are control consistent; this is because after an 8-transition, they will arrive in U_0 and U_1 respectively, but U_0 and U_1 are not control consistent. Thus $\mathcal{C}_3 = \{\{U_0\}, \{U_1\}, \{U_2\}\}$ and $\mathbf{LOC}_3$ has three states.

$U \in \mathcal{U}_i$ such that $\hat{\xi}(U, \alpha)!$, i.e. $E_\alpha(U) = 1$. The rest of the proof is identical to Case (1) above, and we conclude that $s\alpha \in L(\mathbf{SUP})$ in this case as well.

Finally we show $L_m(\mathbf{G}) \cap L_m(\mathbf{LOC}) \subseteq L_m(\mathbf{SUP})$. Let $s \in L_m(\mathbf{G}) \cap L_m(\mathbf{LOC})$; we must show that $s \in L_m(\mathbf{SUP})$. Since $L_m(\mathbf{G}) \cap L_m(\mathbf{LOC}) \subseteq L(\mathbf{G}) \cap L(\mathbf{LOC}) \subseteq L(\mathbf{SUP})$, we have $s \in L(\mathbf{SUP})$, which implies that for all $\alpha \in \Sigma_c$, $i_n = \zeta_\alpha(i_0, Ps)$ and $U(Ps) \in \mathcal{U}_{i_n}$. In addition, $s \in L_m(\mathbf{LOC})$ implies that $s \in P_\alpha^{-1} L_m(\mathbf{LOC}_\alpha)$; so $P_\alpha s \in L_m(\mathbf{LOC}_\alpha)$, i.e. $\eta_\alpha(y_0, P_\alpha s) \in Y_m$. Since $P_\alpha s = P'_\alpha(Ps)$, $\eta_\alpha(y_0, P_\alpha s)$ corresponds to $\zeta_\alpha(i_0, Ps) = i_n$; so $i_n \in I_m$ (because $Y_m = I_m$). Therefore, there exists $U' \in \mathcal{U}_{i_n}$ such that $X_m \cap U' \neq \emptyset$. Then $M(U') = 1$ and thus $T(U') = 1$. By $s \in L_m(\mathbf{G})$, we have $T(U(Ps)) = 1$. Now we have that both U' and $U(Ps)$ are in $\mathcal{U}_{i_n}$, i.e. $(U', U(Ps)) \in R_\alpha$. Consequently $M(U(Ps)) = M(U') = 1$. Hence $U(Ps) \cap X_m \neq \emptyset$. Let $x = U(Ps) \cap X_m$; then there must exist $t \in L_m(\mathbf{SUP})$ such that $x = \xi(x_0, t)$ and $Pt = Ps$. Now, since $L_m(\mathbf{SUP})$ is observable, by $s \in L_m(\mathbf{G})$ we have $s \in L_m(\mathbf{SUP})$.

$\square$

For the example in Fig. 1, we construct partial-observation local controllers $\mathbf{LOC}_1$ and $\mathbf{LOC}_3$ for the (observable) controllable events 1 and 3 respectively, as displayed in Fig. 3. It is then verified that the collective controlled behavior of these local controllers ($\mathbf{LOC}_1$, $\mathbf{LOC}_3$, and $\mathbf{LOC}_5$) is identical to $\mathbf{SUP}$ (in the sense of (18) and (19)). [4]

Finally, we provide the proof of Lemma 2.

Proof of Lemma 2. We must prove (8) and (9).

---

[4]This can be verified by TCT procedures as follows. First, compute $\mathbf{TEST} = Sync(\mathbf{G}, \mathbf{LOC1}, \mathbf{LOC3}, \mathbf{LOC5})$, i.e. $\mathbf{TEST} = \mathbf{G} \| \mathbf{LOC1} \| \mathbf{LOC3} \| \mathbf{LOC5}$. Then it is verified by $true = Isomorph(\mathbf{TEST}, \mathbf{SUP})$ that $L_m(\mathbf{TEST}) = L_m(\mathbf{SUP})$ and $L(\mathbf{TEST}) = L(\mathbf{SUP})$.

15

First, for ($\Rightarrow$) of Eq. (8), let $P_\alpha(s)\alpha \in L(\mathbf{LOC}_\alpha)$, $s\alpha \in L(\mathbf{G})$ and $s \in L(\mathbf{SUP})$; we must prove that $s\alpha \in L(\mathbf{SUP})$. It is derived from $P_\alpha(s)\alpha \in L(\mathbf{LOC}_\alpha)$, that $P_\alpha(s) \in L(\mathbf{LOC}_\alpha)$, because $L(\mathbf{LOC}_\alpha)$ is prefix-closed. Let $y := \eta_\alpha(y_{0,\alpha}, P_\alpha(s))!$; by $P_\alpha(s)\alpha \in L(\mathbf{LOC}_\alpha)$, $\eta_\alpha(y, \alpha)!$. The rest of the proof is identical to the inductive case of proving ($\subseteq$) of (18), and we conclude that $s\alpha \in L(\mathbf{SUP})$.

Next, for ($\Leftarrow$) of Eq. (8), let $s\alpha \in L(\mathbf{SUP})$; $s \in L(\mathbf{SUP})$ and $s\alpha \in L(\mathbf{G})$ are immediate, and it is left to show that $P_\alpha(s)\alpha \in L(\mathbf{LOC}_\alpha)$. By $s\alpha \in L(\mathbf{SUP})$ and (18), we have for all $\sigma \in \Sigma_c$, $s\alpha \in P_\sigma^{-1}L(\mathbf{LOC}_\sigma)$. Because $\alpha \in \Sigma_c$, we have $s\alpha \in P_\alpha^{-1}L(\mathbf{LOC}_\alpha)$, and thus $P_\alpha(s\alpha) \in L(\mathbf{LOC}_\alpha)$. According to the definition of Σ_α, $\{\alpha\} \subseteq \Sigma_\alpha$. Hence, $P_\alpha(s)\alpha = P_\alpha(s\alpha) \in L(\mathbf{LOC}_\alpha)$.

Finally, to prove (9), let $y, y' \in Y_\alpha$ and $\sigma \in \Sigma_o$ and assume that $y' = \eta_\alpha(y, \sigma)$ and $y \neq y'$; we prove that $\sigma \in \Sigma_o$ by contradiction. Suppose that $\sigma \in \Sigma_{uo}$. According to (15), for all $i \in I$, $\zeta_\alpha(i, \sigma)$ is not defined. Further, according to the rule (iii) of constructing $\mathbf{LOC}_\alpha$, (1) for all $y \in Y$, $\eta_\alpha(y, \sigma)$ is not defined, contradicting the assumption that $y' = \eta_\alpha(y, \alpha)$; (2) the selfloop $\eta_\alpha(y, \alpha) = y$ is added to η_α when $\psi_\alpha(y) = 1$, which, however, contradicts the assumption that $y \neq y'$. So we conclude that $\sigma \in \Sigma_o$.

$\square$

IV. PARTIAL-OBSERVATION LOCALIZATION ALGORITHM AND TRANSFER LINE EXAMPLE

In this section, we adapt the supervisor localization algorithm in [1] to compute the partial-observation local controllers.

Let $\mathbf{SUP} = (X, \Sigma, \xi, x_0, X_m)$ be the controllable and observable controlled behavior (as in (7)), with controllable Σ_c and observable Σ_o. Fix $\alpha \in \Sigma_c$. The algorithm in [1] would construct a control cover on X. Here instead, owing to partial observation, we first find the set $\mathcal{U}(X)$ of all uncertainty sets and label it as

$$\mathcal{U}(X) = \{U_0, U_1, ..., U_{n-1}\}.$$

Also we calculate the transition function $\hat{\xi} : \mathcal{U}(X) \times \Sigma_o^* \to \mathcal{U}(X)$. These steps are done by constructing the partial-observation monolithic supervisor $\mathbf{SUPO}$ as in (12) [9, 16].

Next, we apply the localization algorithm in [1] to construct a partial-observation control cover $\mathcal{C}_\alpha$ on $\mathcal{U}(X)$. Initially $\mathcal{C}_\alpha$ is set to be the singleton partition on $\mathcal{U}(X)$, i.e.

$$\mathcal{C}_\alpha = \{\{U_0\}, \{U_1\}, ..., \{U_{n-1}\}\}.$$

Write $\mathcal{U}_i, \mathcal{U}_j$ for two cells in $\mathcal{C}_\alpha$. Then the algorithm 'merges' $\mathcal{U}_i, \mathcal{U}_j$ into one cell if for every uncertainty set $U_i \in \mathcal{U}_i$ and every $U_j \in \mathcal{U}_j$, U_i and U_j, as well as their corresponding future uncertainty sets reachable
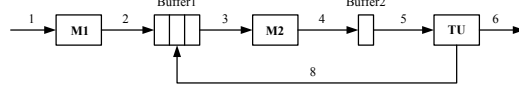
16

Fig. 4. Transfer Line: system configuration, with the set of controllable events $\Sigma_c = \{1, 3, 5\}$
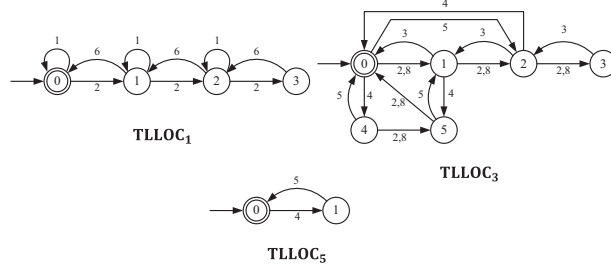

Fig. 5. Transfer Line: local controllers with full observation

by identical strings, are control consistent in terms of $\mathcal{R}_\alpha$. The algorithm loops until all uncertainty sets in $\mathcal{U}(X)$ are checked for control consistency. We call this algorithm the *partial-observation localization algorithm.*

Similar to [1], the algorithm terminates in a finite number of steps and results in a partial-observation control *congruence* $\mathcal{C}_\alpha$ (i.e. with pairwise disjoint cells). The complexity of the algorithm is $O(n^4)$; since the size n of $\mathcal{U}(X)$ is $n \leq 2^{|X|}$ in general, the algorithm is exponential in $|X|$.

In the following, we illustrate the above partial-observation localization algorithm by a Transfer Line system **TL**, as displayed in Fig. 4. **TL** consists of two machines **M1**, **M2** followed by a test unit **TU**; these agents are linked by two buffers (Buffer1, Buffer2) with capacities of three slots and one slot, respectively. We model the synchronous product of **M1**, **M2**, and **TU** as the plant to be controlled; the specification is to protect the two buffers against overflow and underflow.

For comparison purpose, we first present the local controllers under full observation. By [1], these controllers are as displayed in Fig. 5, and their control logic is as follows.

TLLOC$_1$ for agent **M1** ensures that no more than three workpieces can be processed in the material-feedback loop. This is realized by counting the occurrences of event 2 (input a workpiece into the loop) and event 6 (output a workpiece from the loop).

TLLOC$_3$ for agent **M2** guarantees no overflow or underflow of the two buffers. This is realized by counting events 2, 8 (input a workpiece into Buffer1), 3 (output a workpiece from Buffer1), 4 (input a workpiece into Buffer2), and 5 (output a workpiece from Buffer2).
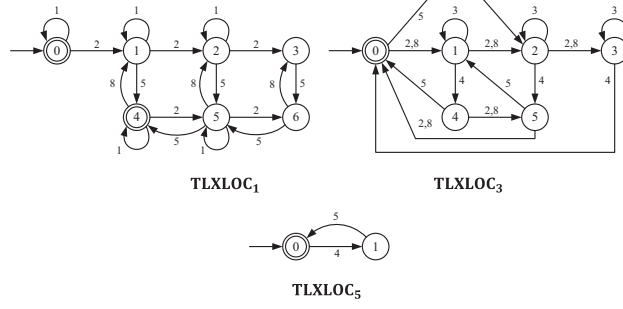
17

Fig. 6.   Transfer Line: local controllers under partial observation P ($\Sigma_{uo} = \{3, 6\}$)

$\mathbf{TLLOC}_5$ for agent $\mathbf{TU}$ guarantees no overflow or underflow of Buffer2. This is realized by counting event 4 (input a workpiece into Buffer2) and event 5 (output a workpiece from Buffer2).

Now consider partial observation. We consider two cases: first with $\Sigma_{uo} = \{3, 6\}$, controlled behavior similar to the full-observation case is achieved but with more complex transition structures; second, with $\Sigma_{uo} = \{1, 3, 5\}$ (i.e. all controllable events are unobservable), the resulting controlled behavior is more restrictive.

Case (i) $\Sigma_{uo} = \{3, 6\}$. We first compute as in (7) the controllable and observable controlled behavior $\mathbf{SUP1}$ which has 39 states. Then we apply the localization algorithm to obtain the partial-observation local controllers. The results are displayed in Fig. 6. It is verified that the collective controlled behavior of these controllers is equivalent to $\mathbf{SUP1}$.

The control logic of $\mathbf{TLXLOC}_1$ for agent $\mathbf{M1}$ is again to ensure that no more than three workpieces can be processed in the loop. But since event 6 is unobservable, the events 5 and 8 instead must be counted so as to infer the occurrences of 6: if 5 followed by 8 is observed, then 6 did not occur, but if 5 is observed and 8 is not observed, 6 may have occurred. As can be seen in Fig. 6, event 6 being unobservable increased the structural complexity of the local controller (as compared to its counterpart in Fig. 5).

The control logic of $\mathbf{TLXLOC}_3$ for agent $\mathbf{M2}$ is again to prevent overflow and underflow of the two buffers. But since event 3 is unobservable, instead the occurrences of event 4 must be observed to infer the decrease of content in Buffer1, and at the same time the increase of content in Buffer2. Also note that since the unobservable controllable event 3 is enabled at states 0, 1, 2, 3, we have selfloops of event 3 at those states. The state size of $\mathbf{TLXLOC}_3$ is the same as its counterpart in Fig. 5.

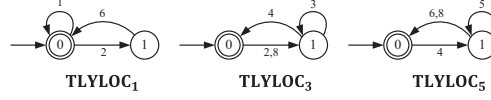$\mathbf{TLXLOC}_5$ for agent $\mathbf{TU}$ is identical to the one in the full-observation case.

18

Fig. 7. Transfer Line: local controllers under partial observation P ($\Sigma_{uo} = \{1,3,5\}$)
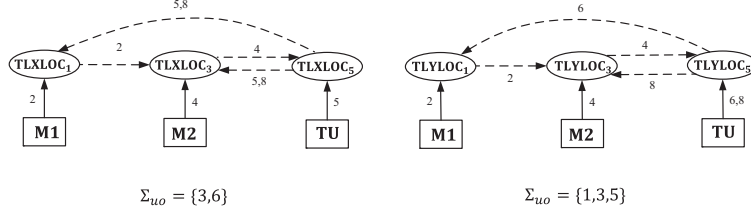

Fig. 8. Transfer Line: communication diagrams of local controllers. The solid lines denote that the corresponding events are directly observed by the local controllers; the dashed lines denote that the corresponding events need to be communicated to the local controllers.

Case (ii) $\Sigma_{uo} = \{1,3,5\}$. We first compute as in (7) the controllable and observable controlled behavior **SUP2** which has only 6 states. Then we apply the localization algorithm to obtain the partial-observation local controllers, as displayed in Fig. 7.

Since all the controllable events are unobservable, the controlled behavior in this case is restrictive: **TLYLOC**$_1$ for agent **M1** allows at most one workpiece to be processed in the loop, and **TLYLOC**$_2$ for agent **M2** allows at most one workpiece to be put in Buffer1 even though Buffer1 has three slots. Also note that in **TLYLOC**$_5$ for agent **TU**, since event 5 is unobservable, events 6 and 8 instead must be observed to infer the occurrence of 5: if either 6 or 8 occurs, event 5 must have previously occurred. In spite of the restrictive controlled behavior, these local controllers collectively achieve equivalent controlled performance to the 6-state **SUP2**.

Finally, we allocate each local controller to the agent owning the corresponding controllable event, and according to the transition diagrams of the local controllers, we obtain two communication diagrams one for each case, as displayed in Fig. 8. A local controller either directly observes an event generated by the agent owning it, as denoted by the solid lines in Fig. 8, or imports an event by communication from other local controllers, as denoted by the dashed lines. Although the communication structures are the same in the two diagrams, owing to different observable event sets Σ_o the observed/communicated events are different.

V. Partial-Observation Localization for Large-Scale Systems

So far we have developed partial-observation supervisor localization assuming that the monolithic supervisor is feasibly computable. This assumption may no longer hold, however, when the system is large-scale and the problem of state explosion arises. In the literature, there have been several architectural approaches proposed to deal with the computational issue based on *model abstraction* [10, 17–19].

Just as in [1], for large-scale system, we propose to combine localization with an efficient heterarchical supervisory synthesis approach [10] in an alternative top-down manner: first synthesize a heterarchical array of partial-observation decentralized supervisors and coordinators that collectively achieves globally feasible and nonblocking controlled behavior; then apply the developed localization algorithm to decompose each supervisor/coordinator into local controllers for the relevant agents.

The procedure of this heterarchical supervisor localization under partial observation is outlined as follows:

Step 1) Partial-observation decentralized supervisor synthesis: For each imposed control specification, collect the relevant component agents (e.g. by event-coupling) and compute as in (12) a partial-observation decentralized supervisor.

Step 2) Subsystem decomposition and coordination: After Step 1, we view the system as comprised of a set of modules, each consisting of a decentralized supervisor with its associated component agents. We decompose the system into smaller-scale subsystems, through grouping the modules based on their interconnection dependencies (e.g. event-coupling or control-flow net [10]).

Having obtained a set of subsystems, we verify the nonblocking property for each of them. If a subsystem happens to be blocking, we design a *coordinator* that removes blocking strings [10, Theorem 4]. The design of the coordinator must respect partial observation; for this reason, we call the coordinator a *partial-observation coordinator*.

Step 3) Subsystem model abstraction: After Step 2, the system consists of a set of nonblocking subsystems. Now we need to verify the nonconflicting property among these subsystems. For this we use model abstraction with the *natural observer* property [10] to obtain an abstracted model of each subsystem.

Step 4) Abstracted subsystem decomposition and coordination: This step is similar to Step 2, but for the abstracted models instead of modules. We group the abstracted models based on their interconnection dependencies, and for each group verify the nonblocking property. If a group turns out to be blocking, we design a partial-observation coordinator that removes blocking strings.

Step 5) Higher-level abstraction: Repeat Steps 3 and 4 until there remains a single group of subsystem

abstractions in Step 4. The heterarchical supervisor/coordinator synthesis terminates at Step 5; the result is a heterarchical array of partial-observation decentralized supervisors and coordinators. Similar to [10], one can establish that these supervisors/coordinators together achieve globally feasible and nonblocking controlled behavior.

Step 6) Partial-observation localization: In this last step, we apply the partial-observation localization algorithm to decompose each of the obtained decentralized supervisors and coordinators into local controllers for their corresponding controllable events. By Theorem 1, the resulting local controllers achieve the same controlled behavior as the decentralized supervisors and coordinators did, namely the globally feasible and nonblocking controlled behavior.

We note that the above procedure extends the full-observation one in [1] by computing partial-observation decentralized supervisors and coordinators in Steps 1-5, and finally in Step 6 applying the partial-observation supervisor localization developed in Section III. In the following we apply the heterarchical localization procedure to study the distributed control of AGV serving a manufacturing workcell under partial observation. As displayed in Fig. 9, the plant consists of five independent AGV

$$\mathbf{A1}, \mathbf{A2}, \mathbf{A3}, \mathbf{A4}, \mathbf{A5}$$

and there are nine imposed control specifications

$$\mathbf{Z1}, \mathbf{Z2}, \mathbf{Z3}, \mathbf{Z3}, \mathbf{WS13}, \mathbf{WS14S}, \mathbf{WS2}, \mathbf{WS3}, \mathbf{IPS}$$

which require no collision of AGV in the shared zones and no overflow or underflow of buffers in the workstations. The generator models of the plant components and the specification are displayed in Figs. 10 and 11 respectively; the detailed system description and the interpretation of the events are referred to [9, Section 4.7]. Consider partial observation and let the unobservable event set be $\Sigma_{uo} = \{13, 23, 31, 42, 53\}$; thus each AGV has an unobservable event. Our control objective is to design for each AGV a set of local strategies such that the overall system behavior satisfies the imposed specifications and is nonblocking.

Step 1) Partial-observation decentralized supervisor synthesis: For each specification displayed in Fig. 11, we group its event-coupled AGV, as displayed in Fig. 12, and synthesize as in (12) a partial-observation decentralized supervisor. The state sizes of these decentralized supervisors are displayed in Table I, in which the supervisors are named correspondingly to the specifications, e.g. **Z1SUP** is the decentralized supervisor corresponding to the specification **Z1**.

Step 2) Subsystem decomposition and coordination: We have nine decentralized supervisors, and thus nine modules (consisting of a decentralized supervisor with associated AGV components). Under full
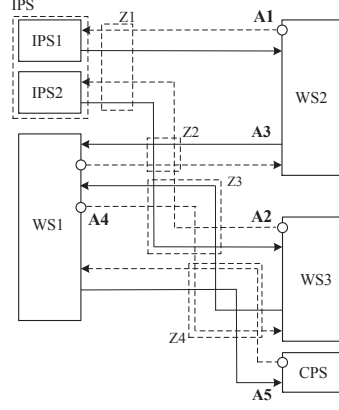
Fig. 9. AGV system configuration. Rectangular dashed boxes represent shared zones of the AGV's traveling routes.
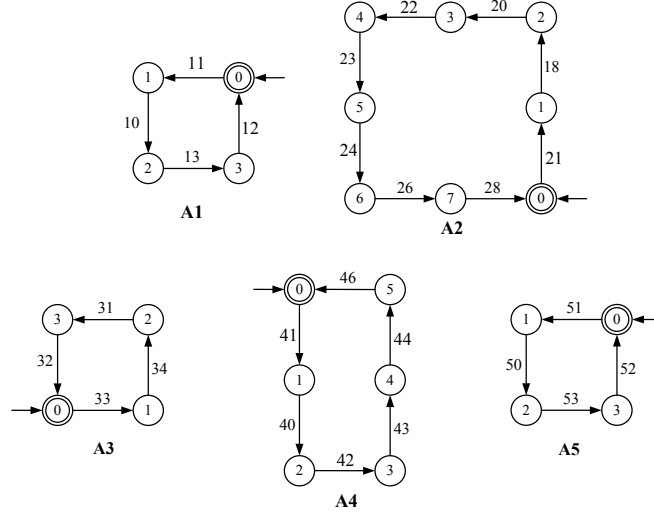

Fig. 10. AGV: Generators of plant components

observation, the decentralized supervisors for the four zones (**Z1SUP**, ..., **Z4SUP**) are *harmless* to the overall nonblocking property [10], and thus can be safely removed from the interconnection structure; then the interconnection structure of these modules are simplified by applying *control-flow net* [10]. Under partial observation, however, the four decentralized supervisors are not harmless to the overall nonblocking property and thus cannot be removed. As displayed in Fig. 13, we decompose the overall

Fig. 11.   AGV: Generators of specifications

Fig. 12.   Event-coupling relations

system into two subsystems:

$$\textbf{SUB1} := \textbf{WS3SUP}||\textbf{WS14SUP}||\textbf{Z3SUP}||\textbf{Z4SUP}$$

$$\textbf{SUB2} := \textbf{WS2SUP}||\textbf{WS13SUP}$$

Between the two subsystems are decentralized supervisors **Z1SUP**, **Z2SUP**, and **IPSSUP**. It is verified

TABLE I.   STATE SIZES OF PARTIAL-OBSERVATION DECENTRALIZED SUPERVISORS

Supervisor	State size	Supervisor	State size
Z1SUP	13	**Z2SUP**	11
Z3SUP	26	**Z4SUP**	9
WS13SUP	15	**WS14SUP**	19
WS2SUP	15	**WS3SUP**	26
IPSSUP	13		

Fig. 13. Subsystem decomposition

that **SUB2** is nonblocking, but **SUB1** is blocking. Hence we design a coordinator **CO1** which makes **SUB1** nonblocking, by

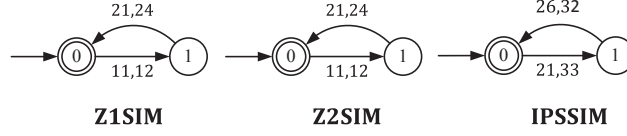$$L_m(\mathbf{CO1}) = \sup \mathcal{CO}(L_m(\mathbf{SUB1}))$$

adapted from [10, Theorem 4]. This coordinator **CO1** has 50 states, and we refer to this nonblocking subsystem **NSUB1**.

Step 3) Subsystem model abstraction: Now we need to verify the nonconflicting property among the nonblocking subsystems **NSUB1**, **SUB2** and the decentralized supervisors **IPSSUP**, **Z1SUP** and **Z2SUP**. First, we determine their shared event set, denoted by Σ_{sub}. Subsystems **NSUB1** and **SUB2** share all events in **A5**: 50, 51, 52 and 53. For **IPSSUP**, **Z1SUP** and **Z2SUP**, we use their reduced generator models **IPSSIM**, **Z1SIM** and **Z2SIM** by supervisor reduction [20], as displayed in Fig. 14. By inspection, **IPSSUP** and **Z1SIM** share events 21 and 24 with **NSUB1**, and events 11 with **SUB2**; **Z2SUP** shares events 24 and 26 with **NSUB1**, and events 32, 33 with **SUB2**. Thus $\Sigma_{sub} = \{11, 12, 21, 24, 26, 32, 33, 50, 51, 52, 53\}$. It is then verified that $P_{sub} : \Sigma^* \to \Sigma_{sub}^*$ satisfies the natural observer property [10]. With P_{sub}, therefore, we obtain the subsystem model abstractions, denoted by **QC_NSUB1** = $P_{sub}(\mathbf{NSUB1})$ and **QC_SUB2** = $P_{sub}(\mathbf{SUB2})$, with state sizes listed in Table II.

Step 4) Abstracted subsystem decomposition and coordination: We treat **QC_NSUB1**, **QC_SUB2**, **IPSSIM**, **Z1SIM** and **Z2SIM** as a single group, and check the nonblocking property. This group turns

24

TABLE II. STATE SIZES OF MODEL ABSTRACTIONS

	NSUB1	QC_NSUB1	SUB2	QC_SUB2
State size	50	19	574	56

Fig. 14. Reduced generator models of decentralized supervisors **Z1SUP**, **Z2SUP** and **IPSSUP**

out to be blocking, and a coordinator **CO2** is then designed by

$$L_m(\mathbf{CO2}) = \sup \mathcal{CO}(L_m(\mathbf{QC\_SUB1}) \| L_m(\mathbf{QC\_SUB2}) \|$$

$$L_m(\mathbf{IPSSIM}) \| L_m(\mathbf{Z1SIM}) \| L_m(\mathbf{Z2SIM}))$$

to make the group nonblocking. This coordinator **CO2** has 160 states.

Step 5) Higher-level abstraction: The modular supervisory control design terminates with the previous Step 4.

We have obtained a hierarchy of nine partial-observation decentralized supervisors and two coordinators. These supervisors and coordinators together achieve globally feasible and nonblocking controlled behavior.

Step 6) Localization: We finally apply the developed supervisor localization procedure to decompose the obtained decentralized supervisors/coordinators into local controllers under partial observation. The generator models of the local controllers are displayed in Fig. 15-19; they are grouped with respect to the individual AGV and their state sizes are listed in Table III. By inspecting the transition structures of the local controllers, only observable events lead to states changes.

Partial observation affects the control logics of the controllers and thus affects the controlled system behavior. For illustration, consider the following case: assuming that event sequence 11.10.13.12.21.18.20.22 has occurred, namely **A1** has loaded a type 1 part to workstation **WS2**, and **A2** has moved to input station **IPS2**. Now, **A2** may load a type 2 part from **IPS2** (namely, event 23 may occur). Since event 24 (**A2** exits Zone 1 and re-enter Zone 2) is uncontrollable, to prevent the specification on Zone 2 (**Z2**) not being violated, AGV **A3** cannot enter Zone 2 if 23 has occurred, i.e. event 33 must be disabled. However, event 33 is eligible to occur if event 23 has occurred. So, under the full observation condition

TABLE III.   STATE SIZES OF PARTIAL-OBSERVATION LOCAL CONTROLLERS

Supervisor/coordinator	Local controller of A1(state size)	Local controller of A2(state size)	Local controller of A3(state size)	Local controller of A4(state size)	Local controller of A5(state size)
Z1SUP	Z1_11(2)	Z1_21(2)			
Z2SUP		Z2_21(2)	Z2_33(2)		
Z3SUP		Z3_21(2),Z3_23(3)		Z3_41(2),Z3_43(3)	
Z4SUP				Z4_41(2)	Z4_51(2)
WS13SUP			WS13_31(2)		WS13_51(2)
WS14SUP				WS14_43(2)	WS14_51(2)
WS2SUP	WS2_13(2)		WS2_33(2)		
WS3SUP		WS3_21(2)		WS3_41(2)	
IPSSUP	IPS_11(2)	IPS_21(2)			
CO1				CO1_41(2)	
CO2	CO2_11(6)		CO2_33(4)		

Fig. 15.   Local controllers for **A1** with controllable events 11 and 13 (the local controllers are named in the format of 'specification_event')
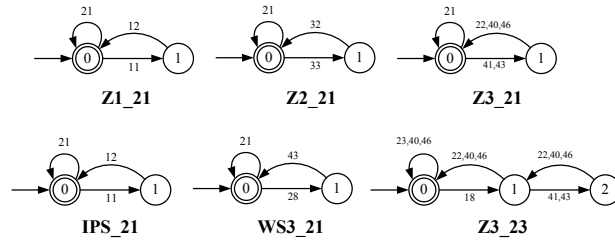

Fig. 16.   Local controllers for **A2** with controllable events 21 and 23

(event 23 is observable) event 33 would occur safely if event 23 has not occurred. However the fact is that event 23 is unobservable; so due to (relative) observability, 33 must also be disabled even if 23 has not occurred, namely the controllers will not know whether or not event 23 has occurred, so it will disabled
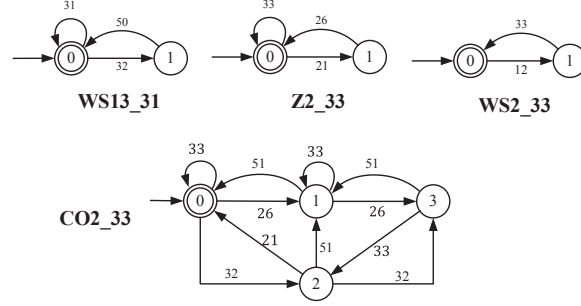
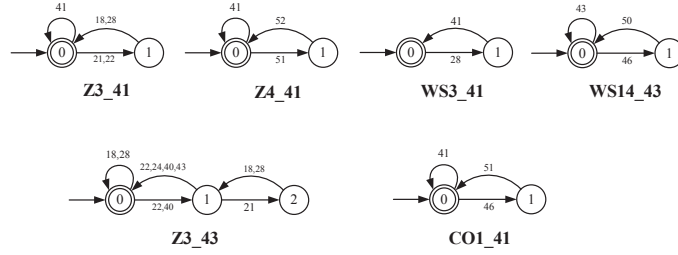Fig. 17. Local controllers for **A3** with controllable events 31 and 33

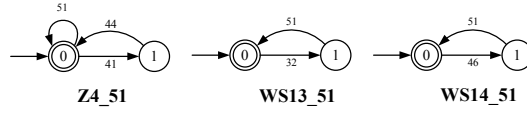Fig. 18. Local controllers for **A4** with controllable events 41 and 43

Fig. 19. Local controllers for **A5** with controllable events 51 and 53 (event 53 is not disabled and thus there is no corresponding local controller)

event 33 in both cases, to prevent the possible illegal behavior. This control strategy coincides with local controller **Z2_33**: event 33 must be disabled if event 21 has occurred, and will not be re-enabled until event 26 has occurred (**A2** exits Zone 2 and re-enter Zone 3).

Finally, the heterarchical supervisor localization has effectively generated a set of partial-observation local controllers with small state sizes (between 2 and 6 states). Grouping these local controllers for the relevant AGV, we obtain a distributed control architecture for the system where each AGV is controlled by its own controllers while observing certain observable events of other AGV; according to the transition diagrams of the local controllers, we obtain a communication diagram, as displayed in Fig. 20, which shows the events to be observed (denoted by solid lines) or communicated (denoted by dashed lines) to local controllers.
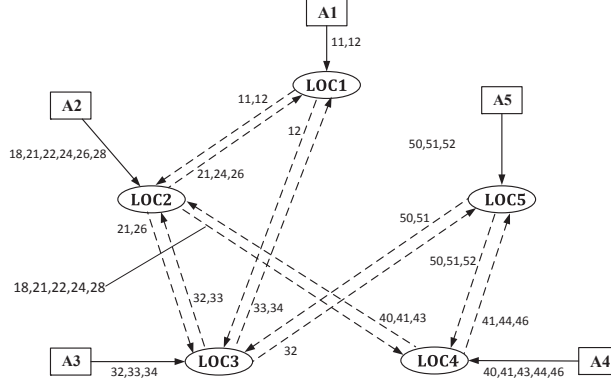
27

Fig. 20. AGV: communication diagram of local controllers. For $i = 1, ..., 5$, **LOCi** represents the local controllers corresponding to **Ai**.

VI. CONCLUSIONS

We have developed partial-observation supervisor localization to solve the distributed control of multi-agent DES under partial observation. This approach first employs relative observability to compute a partial-observation monolithic supervisor, and then decomposes the supervisor into a set of local controllers whose state changes are caused only by observable events. A Transfer Line example is presented for illustration. When the system is large-scale, we have combined the partial-observation supervisor localization with an efficient heterarchical synthesis procedure. In future research we shall extend the partial-observation localization procedure to study distributed control of timed DES.

REFERENCES

[1] K. Cai and W. M. Wonham, "Supervisor localization: a top-down approach to distributed control of discrete-event systems," *IEEE Transactions on Automatic Control*, vol. 55, no. 3, pp. 605–618, 2010.

[2] K. Cai and W. Wonham, "Supervisor localization for large discrete-event systems: case study production cell," *International Journal of Advanced Manufacturing Technology*, vol. 50, no. 9-12, pp. 1189–1202, 2010.

[3] R. Zhang, K. Cai, Y. Gan, Z. Wang, and W. Wonham, "Supervision localization of timed discrete-event systems," *Automatica*, vol. 49, no. 9, pp. 2786–2794, 2013.

[4] K. Cai and W. Wonham, "New results on supervisor localization, with case studies," *Discrete Event Dynamic Systems*, vol. 25, no. 1-2, pp. 203–226, 2015.

[5] K. Cai and W. M. Wonham, *Supervisor Localization: A Top-Down Approach to Distributed Control of Discrete-Event Systems*. Lecture Notes in Control and Information Sciences, vol. 459, Springer, 2015.

[6] K. Cai, R. Zhang, and W. Wonham, "Relative observability of discrete-event systems and its supremal sublanguages," *IEEE Transactions on Automatic Control*, vol. 60, no. 3, pp. 659–670, 2015.

[7] F. Lin and W. Wonham, "On observability of discrete-event systems," *Information Sciences*, vol. 44, no. 3, pp. 173–198, 1988.

[8] R. Cieslak, C. Desclaux, A. Fawaz, and P. Varaiya, "Supervisory control of discrete-event processes with partial observations," *IEEE Transactions on Automatic Control*, vol. 33, no. 3, pp. 249–260, 1988.

[9] W. Wonham, *Supervisory Control of Discrete-Event Systems*. Systems Control Group, ECE Dept, Univ. Toronto, Toronto, ON, Canada, July 2015, available at http://www.control.utoronto.ca/DES.

[10] L. Feng and W. M. Wonham, "Supervisory control architecture for discrete-event systems," *IEEE Transactions on Automatic Control*, vol. 53, no. 6, pp. 1449–1461, 2008.

[11] S. Takai and T. Ushio, "Effective computation of Lm(G)-closed, controllable, and observable sublanguage arising in supervisory control," *Systems & Control Letters*, vol. 49, no. 3, pp. 191–200, 2003.

[12] X. Yin and S. Lafortune, "Synthesis of maximally permissive supervisors for partially-observed discrete-event systems," *IEEE Transactions on Automatic Control*, vol. 61, no. 5, pp. 1239–1254, 2016.

[13] M. Sampath, S. Lafortune, and D. Teneketzis, "Active diagnosis of discrete-event systems," *IEEE Transactions on Automatic Control*, vol. 43, no. 7, pp. 908–929, 1998.

[14] J. Dubreil, P. Darondeau, and H. Marchand, "Supervisor control for opacity," *IEEE Transactions on Automatic Control*, vol. 55, no. 5, pp. 1089–1100, 2010.

[15] X. Yin and S. Lafortune, "A uniform approach for synthesizing property-enforcing supervisors for partially-observed discrete-event systems," *IEEE Transactions on Automatic Control*, 2016, dOI: 10.1109/TAC.2015.2484359.

[16] C. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*, 2nd ed. Springer, 2008.

[17] R. Hill and D. Tilbury, "Modular supervisory control of discrete-event systems with abstraction and incremental hierarchical construction," in *Proc. 8th International Workshop on Discrete Event Systems*, Ann Arbor, MI, July 2006, pp. 399–406.

[18] K. Schmidt and C. Breindl, "Maximally permissive hierarchical control of decentralized discrete event systems," *IEEE Transactions on Automatic Control*, vol. 56, no. 4, pp. 723–737, 2011.

[19] R. Su, J. H. van Schuppen, and J. E. Rooda, "Maximum permissive coordinated distributed supervisory control of nondeterministic discrete-event systems," *Automatica*, vol. 48, no. 7, pp. 1237–1247, 2012.

[20] R. Su and W. Wonham, "Supervisor reduction for discrete-event systems," *Discrete Event Dynamic Systems*, vol. 14, no. 1, pp. 31–53, January 2004.