2005 American Control Conference
June 8-10, 2005. Portland, OR, USA

Prioritized Synchronization under Mask for Interaction/Control of Partially Observed Discrete Event Systems

Changyan Zhou and Ratnesh Kumar
Department of Electrical & Computer Engineering
Iowa State University, Ames, IA 50014

***Abstract*— This paper extends the formalism of prioritized synchronous composition (PSC), introduced by Heymann, for modeling interaction (and control) of discrete event systems to incorporate partial observation. PSC based control helps remove the control-compatibility requirement of a supervisor. In order to also remove the observation-compatibility requirement of a supervisor, there have been attempts to generalize PSC to account for partial observation. First such attempt was the notion of masked composition (MC), and later the notion of masked-PSC (MPSC) was introduced. Under MPSC the condition for existence of supervisor is *normality together with controllability*, as opposed to the usual weaker condition of *observability together with controllability*. This motivates the introduction of the notion of prioritized synchronous composition under mask (PSCM). We show that when PSCM is adopted as a mechanism of interaction, not only the control & observation-compatibility requirements are removed of a supervisor, the existence condition is given by *achievability* that is weaker than controllability and observability combined. (The weaker condition is required since we allow supervisors to be nondeterministic.) This suggests that the notion of PSCM, presented in the paper is an appropriate generalization of PSC to account for partial observation.**

Keywords: Discrete event systems, supervisory control, prioritized synchronous composition under mask, achievability, partial observation

I. INTRODUCTION

Most work on supervisory control of discrete event systems (DESs), such as [14], [13], [7], use strict synchronous composition (SSC) of the plant and supervisor as a mechanism of control. In SSC, it is required that the common events occur synchronously, which is a restriction. For example, there is no a priori reason for a supervisor to synchronously execute all the uncontrollable events that a plant executes.

Heymann [3] proposed a type of interaction, called prioritized synchronous composition (PSC), which relaxed such synchronization requirements. PSC delegates the effects of control limitations from logic part (implemented as automata) to interface part (implemented as PSC), and thereby, removes the requirement of "completeness" [7] or "Σ_u-compatibility" [7] of a supervisor. In PSC, each system possesses an event priority set specifying a set of events whose execution require its participation. The systems which do not have priority over the event also participate in its execution if they can, and otherwise the event takes place without the participation of such systems. Thus, a system with no priority over an event cannot block its execution. Supervisory control of DESs using PSC has been studied in [5], [11], [1], [2], [4], [15], [10], [12].

PSC models the interaction among systems when all the events are completely observable. When systems interact under partial observation (modeled as non-identity observation masks), their interaction through PSC requires that the systems be observation compatible with respect to their masks [16]. So it is meaningful to generalize the notion of PSC to allow interaction of systems possessing non-identity observation mask. This then will allow us to model interactions of systems without the need to ensure that they are control or observation compatible.

An effort to generalize PSC in such a direction was presented in [16], and the generalization was called masked compsition (MC). In MC, each system was associated with two types of mask function: a control mask that identified events from the control perspective, and an observation mask that identified events from the observation perspective. The main difficulty with that work is the underlying modeling formalism of process objects that contains "virtual transitions" besides "real" ones, and modeling of practical systems as such process objects is not clear.

Another generalization of PSC to describe prioritized synchronization of systems via interfaces, *masked prioritized synchronous composition (MPSC)*, was introduced by Kumar-Heymann in [8] and latter used for control with "driven" events in [6]. MPSC retains the basic concept of PSC in that each system has its own event priority set, i.e., the set of events in which it must participate in order for them to occur in the composition. In MPSC, each system is allowed to interact with its environment via interfaces that are modeled as event mask functions. When two or more systems interact at a common interface, they can synchronize on events that are mapped to common interface events.

The research was supported in part by the National Science Foundation under the grants NSF-ECS-9709796, NSF-ECS-0099851, NSF-ECS-0218207, NSF-ECS-0244732, NSF-EPNES-0323379, and NSF-0424048, and a DoD-EPSCoR grant through the Office of Naval Research under the grant N000140110621.

MPSC is appropriate for systems interacting via common interfaces. When MPSC is employed for control the condition for existence of a supervisor is *normality together with controllability*, as opposed to the usual weaker condition of *observability together with controllability*. This suggests that MPSC imposes certain stringent interface constraints. This motivated us to introduce the notion of prioritized synchronous composition under mask (PSCM) in this paper.

Through the introduction of PSCM we are able to relax the restriction on control that MPSC imposes. We show that when PSCM is adopted as a mechanism of interaction, not only the control & observation-compatibility requirements are removed of a supervisor, the existence condition is given by *achievability* [9] that is weaker than controllability and observability combined. (The weaker condition is required since we allow supervisors to be nondeterministic, whereas the conditions of controllability and observability combined are required for the existence of deterministic supervisor.) This suggests that the notion of PSCM, presented in the paper, is an appropriate generalization of PSC to account for partial observations. The results on PSCM-based control presented in the paper have benefitted from the past work of our group [9] that laid the foundation of nondeterministic control and introduced the notion of *achievability* as a condition for existence of a nondeterministic supervisor under partial observation.

II. Notations and Preliminaries

In this paper nondeterministic state machines (NSMs) are used to model discrete event systems. A NSM G is a four tuple: $G := (X, \Sigma, \alpha, X_0)$, where X is its set of states, Σ is its set of events, $\alpha : X \times (\Sigma \cup \{\epsilon\}) \to 2^X$ is its state transition function, and $X_0 \subseteq X$ is its set of initial states. For an event set Σ, we use $\overline{\Sigma}$ to denote $\Sigma \cup \{\epsilon\}$. A triple $(x, \sigma, x') \in X \times \overline{\Sigma} \times X$ is called a *transition* if $x' \in \alpha(x, \sigma)$; if $\sigma = \epsilon$, the transition is called an ϵ-transition.

Given an event set Σ, Σ^* denotes the set of all finite-length sequences of events from Σ, including the trace of zero length, denoted ϵ. For $x \in X$, we use $\Sigma(x) := \{\sigma \in \overline{\Sigma} \mid \alpha(x, \sigma) \neq \emptyset\}$ to denote the set of labels in $\overline{\Sigma}$ defined at x. The *generated* languages of G, denoted $L(G)$, is defined as $L(G) := \{s \in \Sigma^* \mid \alpha(X_0, s) \neq \emptyset\}$. Letting $pr(\cdot)$, denote the prefix closure operation, $L(G) = pr(L(G))$.

One way to model control interaction between plant and supervisor is via the *strict synchronous composition (SSC)* of their state machine representations. The SSC of two state machines $G_i := (X_i, \Sigma, \alpha_i, X_{0i})$ is the NSM $G_1 \| G_2 := (X_1 \times X_2, \Sigma, \alpha, X_{01} \times X_{02})$, where for $x_1 \in X_1$, $x_2 \in X_2$, and $\sigma \in \overline{\Sigma}$:

$$\alpha((x_1, x_2), \sigma) := \begin{cases} \alpha_1(x_1, \sigma) \times \alpha_2(x_2, \sigma) & \text{if } \sigma \in \Sigma \\ (\alpha_1(x_1, \epsilon) \times \{x_2\}) \cup (\{x_1\} \times \alpha_2(x_2, \epsilon)) & \text{if } \sigma = \epsilon \end{cases}$$

It is easy to see that $L(G_1 \| G_2) = L(G_1) \cap L(G_2)$.

In order to relax the strict synchronization requirement of SSC, Heymann [3] proposed *prioritized synchronous composition (PSC)*. In PSC, each system has an event priority set. An event can occur as long as all systems having the priority over the event can participate.

For $i = 1, 2$, consider NSM $G_i = (X_i, \Sigma, \alpha_i, X_{0i})$ with its event priority set A_i. Then the prioritized synchronous composition of G_1 and G_2 is given by

$$G_{1A_1} \|_{A_2} G_2 = (X_1 \times X_2, \Sigma, \alpha, X_{01} \times X_{02}),$$

where for $x_1 \in X_1, x_2 \in X_2$ and $\sigma \in \Sigma$,

$$\alpha((x_1, x_2), \sigma) := \begin{cases} \alpha_1(x_1, \sigma) \times \alpha_2(x_2, \sigma), \text{ if} \begin{cases} \alpha_1(x_1, \sigma) \neq \emptyset, \\ \alpha_2(x_2, \sigma) \neq \emptyset \end{cases} \\ \alpha_1(x_1, \sigma) \times \{x_2\}, \text{ if} \begin{cases} \alpha_1(x_1, \sigma) \neq \emptyset, \\ \alpha_2(x_2, \sigma) = \emptyset, \sigma \notin A_2 \end{cases} \\ \{x_1\} \times \alpha_2(x_2, \sigma), \text{ if} \begin{cases} \alpha_2(x_2, \sigma) \neq \emptyset, \\ \alpha_1(x_1, \sigma) = \emptyset, \sigma \notin A_1 \end{cases} \\ \emptyset, \text{ otherwise} \end{cases}$$

$$\alpha((x_1, x_2), \epsilon) := (\alpha_1(x_1, \epsilon) \times \{x_2\}) \cup (\{x_1\} \times \alpha_2(x_2, \epsilon)).$$

The event priority set of $G_{1A_1} \|_{A_2} G_2$ is given by $A_1 \cup A_2$. In the special case when the event priority sets of the two systems exhaust the entire event set Σ, PSC can be transformed to SSC using the method of augmentation introduced in [3].

The events executed by a system are partially observed by other systems owing to the particular event-sensors used. Such a partial observation induces a partition of $\overline{\Sigma}$, with each partition representing a set of observation indistinguishable events. *For $\sigma \in \overline{\Sigma}$, we use $M(\sigma) \subseteq \overline{\Sigma}$ to denote the set of σ-indistinguishable events. $\sigma \in \Sigma$ is said to be unobservable if $\sigma \in M(\epsilon)$; σ is said to be completely observable if* $M(\sigma) = \{\sigma\}$. The set of unobservable events is denoted Σ_{uo}, and set of completely observable events is denoted Σ_o.

III. Prioritized Synchronization under Mask

In this section, we formalize the notion of prioritized synchronous composition under mask (PSCM) and study its properties. PSCM generalizes the prioritized synchronization of systems to incorporate partial observation. In PSCM, each system posseses an event priority set and an observation mask. In this section, we show that when certain constraints are imposed on the priority sets and observation masks, the PSCM of two systems can alternatively be obtained by first "augmenting" each of the systems, and next computing the PSC of augmented systems.

In PSCM, an event is "locally" enabled at a certain state of a system if it is executable at that state or is a non-priority event. An event is enabled in the composition (i.e., "globally" enabled) if and only if it is enabled by all interacting systems (i.e., locally enabled by all). A globally enabled event that is executable by one of the systems, can occur in the composed system upon "initiation" by a system that can execute it. Then other systems track it by executing an observation indistinguishable event. If the event is unobservable to or if no observationally indistinguishable events are defined in one of the systems, then that system does not participate in tracking. The transition in the composed system is labeled by the initiating event (and not by the tracking event).

Since any executable event is automatically enabled, and since a system cannot block events outside its priority set (meaning they always remain enabled), the set of enabled events at a state x_i of G_i is $\Sigma(x_i) \cup A_i^c$, the set of executable events at x_i together with the non-priority events. We denote this as, $\Sigma_e(x_i) := \Sigma(x_i) \cup A_i^c$.

An event is enabled at a state $(x_1, \cdots, x_n)$ of PSCM composed systems $\{G_i, i \leq n\}$ if and only if it is enabled at state x_i of G_i. In other words, the set of enabled events at state $(x_1, \cdots, x_n)$ of the composition is given by, $\Sigma_e((x_1, \cdots, x_n)) := \bigcap_{i=1}^n \Sigma_e(x_i) = \bigcap_{i=1}^n [\Sigma(x_i) \cup A_i^c]$.

Next we formally define the notion of PSCM.

Definition 1: For $i = 1, 2$, consider system $G_i = (X_i, \Sigma, \alpha_i, X_{0i})$, possessing event priority set A_i, and observation mask M_i. Then the *prioritized synchronous composition under mask (PSCM)* of G_1 and G_2 is given by

$$G_1 {}_{A_1}^{M_1}\|_{A_2}^{M_2} G_2 = (X_1 \times X_2, \Sigma, \alpha, X_{01} \times X_{02}),$$

where for $x_1 \in X_1, x_2 \in X_2$ and $\sigma \in \Sigma$,

$$\alpha((x_1, x_2), \sigma) :=$$

$$\begin{cases} \alpha_1(x_1, \sigma) \times \alpha_2(x_2, \sigma'), \text{ if } \begin{cases} \sigma \in \Sigma_e((x_1, x_2)), \\ \alpha_1(x_1, \sigma) \neq \emptyset, \\ \alpha_2(x_2, \sigma') \neq \emptyset, \\ \sigma' \in M_2(\sigma) \neq M_2(\epsilon) \end{cases} \\ \alpha_1(x_1, \sigma') \times \alpha_2(x_2, \sigma), \text{ if } \begin{cases} \sigma \in \Sigma_e((x_1, x_2)), \\ \alpha_1(x_1, \sigma') \neq \emptyset, \\ \alpha_2(x_2, \sigma) \neq \emptyset, \\ \sigma' \in M_1(\sigma) \neq M_1(\epsilon) \end{cases} \\ \alpha_1(x_1, \sigma) \times \{x_2\}, \text{ if } \begin{cases} \sigma \in \Sigma_e((x_1, x_2)), \\ \alpha_1(x_1, \sigma) \neq \emptyset, \\ [\epsilon \in M_2(\sigma) \bigvee \\ \alpha_2(x_2, M_2(\sigma)) = \emptyset] \end{cases} \\ \{x_1\} \times \alpha_2(x_2, \sigma), \text{ if } \begin{cases} \sigma \in \Sigma_e((x_1, x_2)), \\ \alpha_2(x_2, \sigma) \neq \emptyset, \\ [\epsilon \in M_1(\sigma) \bigvee \\ \alpha_1(x_1, M_1(\sigma)) = \emptyset] \end{cases} \\ \emptyset, \text{ otherwise} \end{cases}$$

$$\alpha((x_1, x_2), \epsilon) :=$$
$$(\alpha_1(x_1, M_1(\epsilon)) \times \{x_2\}) \cup (\{x_1\} \times \alpha_2(x_2, M_2(\epsilon))).$$

Note in all clauses, the executable event σ is also enabled in the composition ($\sigma \in \Sigma_e((x_1, x_2))$). In the first clause, σ is executable at x_1 ($\alpha_1(x_1, \sigma) \neq \emptyset$). G_1 initiates σ by transitioning to a state in $\alpha_1(x_1, \sigma)$, and G_2 tracks by executing an M_2-indistinguishable event $\sigma' \in M_2(\sigma) \neq M_2(\epsilon)$ that is defined at state x_2 ($\alpha_2(x_2, \sigma') \neq \emptyset$). The second clause is similar to the first clause, except here G_2 initiates σ and G_1 tracks by executing $\sigma' \in M_1(\sigma) \neq M_1(\epsilon)$. In the third clause, σ is executable in G_1 and either it is unobservable to G_2 ($\sigma \in M_2(\epsilon)$) or there is no M_2-indistinguishable event defined at state x_2 ($\alpha_2(x_2, M_2(\sigma)) = \emptyset$). So, σ occurs asynchronously in G_1. (Note that G_2 does not block it since σ is enabled by both G_1 and G_2 by virtue of its membership in $\Sigma_e((x_1, x_2)) = \Sigma_e(x_1) \cap \Sigma_e(x_2)$.) The fourth clause can be understood in a similar way as the third clause. Finally, an ϵ-transition in the composition corresponds to an asynchronous execution of a label in $M_i(\epsilon)$, $i = 1, 2$, in which case only G_i participates.

The event priority set of $G_1 {}_{A_1}^{M_1}\|_{A_2}^{M_2} G_2$ is given by $A := A_1 \cup A_2$. The class of observationally indistinguishable events for an event σ in $G_1 {}_{A_1}^{M_1}\|_{A_2}^{M_2} G_2$ is given by $M(\sigma) := M_1(\sigma) \cap M_2(\sigma)$.

Remark 1: In the special case when both systems have identity mask (Id) functions, the PSCM reduces to the PSC. I.e., $G_1 {}_{A_1}^{Id}\|_{A_2}^{Id} G_2 = G_1 {}_{A_1}\|_{A_2} G_2$.

The following example illustrates the concept of PSCM.

Example 1: Consider G_1 and G_2 shown in Figure 1, with

$$A_1 = \{a\}, M_1(a) = M_1(b) = \{a, b\}, M_1(c) = \{\epsilon, c\},$$
$$M_1(d) = \{d\}; A_2 = \{b\}, M_2(a) = M_2(d) = \{a, d\},$$
$$M_2(b) = \{b\}, M_2(c) = \{c\}.$$

$G_1 {}_{A_1}^{M_1}\|_{A_2}^{M_2} G_2$ is drawn in Figure 2, where for simplicity a

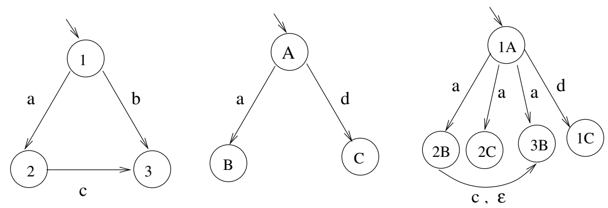

Fig. 1. G_1 (left), G_2 (middle), and $G_1 {}_{A_1}^{M_1}\|_{A_2}^{M_2} G_2$ (right)

state (x_1, x_2) of the composition is written as $x_1 x_2$.

At state $1A$,

$$\begin{aligned} \Sigma_e(1A) &= [\{a, b\} \cup \{b, c, d\}] \cap [\{a, d\} \cup \{a, c, d\}] \\ &= \{a, c, d\}. \end{aligned}$$

Since for $a \in \Sigma_e(1A)$, $\alpha_1(1, a) = \{2\}$, $a, d \in M_2(a)$, $\alpha_2(A, a) = \{B\}$, and $\alpha_2(A, d) = \{C\}$, by clause 1, we have the transitions $(1A, a, 2B)$ and $(1A, a, 2C)$ in $G_1 {}_{A_1}^{M_1}\|_{A_2}^{M_2} G_2$.

Similarly, for $a \in \Sigma_e(1A)$, $\alpha_2(A, a) = \{B\}$, $a, b \in M_1(a)$, $\alpha_1(1, a) = \{2\}$, and $\alpha_1(1, b) = \{3\}$. By clause 2, we have the transitions $(1A, a, 2B)$ and $(1A, a, 3B)$ in $G_1 {}_{A_1}^{M_1}\|_{A_2}^{M_2} G_2$.

Similarly, for $d \in \Sigma_e(1A)$, $\alpha_2(A, d) = \{C\}$, $d \in M_1(d)$ and $\alpha_1(1, d) = \emptyset$. By clause 4, we have the transition $(1A, d, 1C)$ in $G_1 {}_{A_1}^{M_1}\|_{A_2}^{M_2} G_2$.

Note that for $c \in \Sigma_e(1A)$, $\alpha_1(1, c) = \emptyset$ and $\alpha_2(A, c) = \emptyset$. Thus, transition on c at state $1A$ in $G_1 {}_{A_1}^{M_1}\|_{A_2}^{M_2} G_2$ does not exist.

At state $2B$,

$$\Sigma_e(2B) = [\{c\} \cup \{b, c, d\}] \cap [\emptyset \cup \{a, c, d\}] = \{c, d\}.$$

Since for $c \in \Sigma_e(2B)$, $\alpha_1(2, c) = \{3\}$, $c \in M_2(c)$ and $\alpha_2(B, c) = \emptyset$. By clause 3, we have the transition $(2B, c, 3B)$ in $G_1 {}_{A_1}^{M_1}\|_{A_2}^{M_2} G_2$.

Also, since $c \in M_1(\epsilon)$, by clause 5, transition $(2B, \epsilon, 3B)$ is defined in $G_1 {}_{A_1}^{M_1}\|_{A_2}^{M_2} G_2$.

IV. Augmentation for Conversion to PSC/SSC

The main feature of PSCM (when compared to PSC) is that execution of an event enabled in the composition by a system can be tracked by another system by synchronously executing an indistinguishable event. We call such synchronous execution M-synchronous executions, or simply M-synchronizations. One purpose of augmentation is to introduce new transitions that let such M-synchronizations be computed as ordinary synchronizations, where the synchronizing transitions carry the same label. Another purpose is to also capture asynchronous executions also as ordinary synchronous executions by introducing appropriate self-loop transitions in systems that are non-participants, and for unobservable events appropriate ϵ-transtitions in the systems where unobservable events are executable. It is clear that augmentation in G_j for an asynchronous execution of G_i will be a self-loop, whereas the augmentation for a transition in G_i that G_j tracks, will be along side the tracking transition, and also augmentation on ϵ-transition in G_j will be along side an unobservable event transition executable in G_j. Care must be taken so that it is always the case that an augmented transition of G_j synchronizes with an existing transition of G_i, i.e., an augmented transition of G_j should not synchronize with an augmented transition of G_i, since such a transition is not possible in the original composition.

In other words, in order to perform augmentation on $\sigma \in \Sigma$ at state x_i of G_i, either (i) σ is a priority event of G_i, is executable at x_i, and is not completely observable by G_i, or (ii) σ is a non-priority event of G_i but a priority event of some system. In either case there must exist a system for which σ is a priority event and is completely observable. Allowing for the possibility of augmentation on any priority event, we make the following requirement assumption.

Assumption 1: For $i \leq n$, consider NSM G_i possessing event priority set A_i and observation mask M_i. Suppose $\forall \sigma \in A = \cup_i A_i$, $\exists j$ such that $\sigma \in A_j \cap \Sigma_{oj}$.

Under Assumption 1, all events in $A = \cup_i A_i$ are candidates for augmentation. Moreover the label ϵ can be used for augmentation whenever an unobservable event is locally defined. Letting $Aug_i(x_i) \subseteq \overline{\Sigma}$ denote the set of labels in $\overline{\Sigma}$ with which state x_i in system G_i can be augmented.

Algorithm 1: For $i \leq n$, consider NSM G_i possessing event priority set A_i and observation mask M_i. The following algorithm computes augmented G_i, $G_i^{Aug_i} := (X_i, \Sigma, \alpha_i^{Aug}, X_{0i})$.

1) For each state x_i of G_i,
$$Aug_i(x_i) := \begin{cases} [A_i \cap \Sigma(x_i) - \Sigma_{oi}] \cup [A - A_i] & \text{if } \Sigma(x_i) \cap A \cap M_i(\epsilon) = \emptyset \\ [A_i \cap \Sigma(x_i) - \Sigma_{oi}] \cup [A - A_i] \cup \{\epsilon\} & \text{otherwise} \end{cases}$$
2) For $\sigma \in Aug_i(x_i) - M_i(\epsilon)$, if $\alpha_i(x_i, M_i(\sigma)) \neq \emptyset$, add transitions on σ from x_i to all states in this set, otherwise add self-loop on σ at x_i;
3) For $\sigma \in Aug_i(x_i) \cap M_i(\epsilon)$, add self-loop on σ at x_i. Further if $\alpha_i(x_i, \sigma) \neq \emptyset$, add transitions on ϵ from x_i to all states in this set.

Since we do not augment with events in $[\Sigma - A]$, through the augmentation we are able to simulate only those asynchronous or M-synchronous executions as synchronous executions that occur on events outside $[\Sigma - A]$, i.e., on events in A. So after the augmentation, the priority sets of both the systems can only be enlarged to the set A, where all events will occur synchronously. This is summarized in the following theorem. For space consideration, only sketched proof is given.

Theorem 1: Under the Assumption 1, $G_1 {}_{A_1}^{M_1}\|_{A_2}^{M_2} G_2 = G_1^{Aug_1} {}_{A}^{M_1}\|_{A}^{M_2} G_2^{Aug_2}$.

Sketched Proof: Since the state sets, the event sets, and the initial states of the two NSM's are all identical, we only need to show that they also have the identical set of transitions. Let the set of events defined (resp., enabled) at a state x_i of $G_i^{Aug_i}$ is denoted by $\Sigma_{Aug}(x_i)$ (resp., $\Sigma_{Aug,e}(x_i)$). Since the priority set of $G_i^{Aug_i}$ is A, it follows that $\Sigma_{Aug,e}(x_i) = \Sigma_{Aug}(x_i) \cup A^c \cup \{\epsilon\}$. Then by Algorithm 1, we have $\Sigma_e((x_1, x_2)) = \Sigma_{Aug,e}((x_1, x_2))$. Since G_i is a subautomaton of $G_i^{Aug_i}$, it follows that each transition of the left NSM is also a transition of the right NSM. For the converse, since we do not augment an event outside A, a transition of the right NSM on an event outside A is also a transition of the left NSM. For a transition on an event $\sigma \in A$, by Assumption 1, we assume $\sigma \in A_2 \cap \Sigma_{o2}$. By Definition 1, for a transition on σ to occur in the composed system, $\sigma \in \Sigma_{Aug,e}((x_1, x_2))$. Then the following cases exist: (i) $\sigma \in \Sigma(x_1) \cap \Sigma_{o1}$, there is no newly introduced transition on σ in the right NSM for this case. (ii) $\sigma \in ([\Sigma(x_1) - \Sigma_{o1}] \cup A_1^c) - M_1(\epsilon)$. (iii) $\sigma \in ([\Sigma(x_1) - \Sigma_{o1}] \cup A_1^c) \cap M_1(\epsilon)$. By Algorithm 1 and Definition 1, a transition on σ for cases (ii) and (iii) of the right NSM is also a transition of the left NSM. ■

In the special case, when the event priority sets exhaust the entire event set, i.e., when $A = \Sigma$, all M-synchronous executions can be simulated as ordinary synchronous executions. Then no M-synchronizations are needed and all masks can be treated as identity mask. We state this formally below.

Assumption 2: For $i \leq n$, consider NSM G_i possessing event priority set A_i such that $A = \cup_i A_i = \Sigma$.

Theorem 2: Under the Assumptions 1 and 2,
$$G_1 {}_{A_1}^{M_1}\|_{A_2}^{M_2} G_2 = G_1^{Aug_1} {}_{\Sigma}^{M_1}\|_{\Sigma}^{M_2} G_2^{Aug_2} = G_1^{Aug_1} {}_{\Sigma}^{Id}\|_{\Sigma}^{Id} G_2^{Aug_2}.$$
Sketched Proof: By Theorem 1, it suffices to prove that $G_1^{Aug_1} {}_{\Sigma}^{M_1}\|_{\Sigma}^{M_2} G_2^{Aug_2} = G_1^{Aug_1} {}_{\Sigma}^{Id}\|_{\Sigma}^{Id} G_2^{Aug_2}$. It is easily to see that the right NSM is a subautomaton of the left NSM. For the converse, the analysis carried out in proof of Theorem 1 is also valid here since Theorem 1 applies under one less assumption. Notice that each M-synchronization or asynchronous execution in the left NSM is duplicated. Then, by replacing non-identity masks M_1 and M_2 by the identity mask, only certain duplicate transitions on $\sigma \in A$

are avoided, but no σ-transition of the left NSM is removed in the right NSM. ■

The following corollary follows from Theorem 2 and Remark 1.

Corollary 1: Under the Assumptions 1 and 2,

$$\begin{aligned} G_1{}^{M_1}_{A_1}\|^{M_2}_{A_2} G_2 &= G_1^{Aug_1}{}^{Id}_{\Sigma}\|^{Id}_{\Sigma} G_2^{Aug_2} \\ &= G_1^{Aug_1}{}_{\Sigma}\|_{\Sigma} G_2^{Aug_2} \\ &= G_1^{Aug_1} \| G_2^{Aug_2}. \end{aligned}$$

The result in the above corollary can be read as follows. Under Assumptions 1 and 2,

$$\begin{aligned} & PSCM((G_1, A_1, M_1), (G_2, A_2, M_2)) \\ = \; & PSC((G_1^{Aug_1}, A), (G_2^{Aug_2}, A)) \\ = \; & SSC(G_1^{Aug_1}, G_2^{Aug_2}). \end{aligned}$$

Note that Assumptions 1 and 2 automatically hold when one of systems can observe every event completely (has identity mask) and has priority over every event (priority set = Σ). This yields the following corollary, which is useful in supervisory control setting, for a plant can "observe" every event and has priority over every event.

Corollary 2: $G_1{}^{Id}_{\Sigma}\|^{M}_{A_2} G_2 = G_1{}_{\Sigma}\|_{\Sigma} G_2^{Aug_2} = G_1 \| G_2^{Aug_2}$.

Note that no assumptions are needed in Corollary 2.

V. Control of Discrete Event Systems via PSCM

In this section, we extend the theory of supervisory control under partial observation to the present setting where control is exercised by means of interaction via PSCM (under the restriction that the event priority set of the supervisor is a subset of the event priority set of the plant, i.e., there are no "driven" events). The plant is modeled by a NSM $G = (X, \Sigma, \alpha, X_0)$ having event priority set Σ and identity observation mask. The supervisor is modeled as another NSM $S = (Y, \Sigma, \beta, Y_0)$ having event priority set Σ_c, the set of controllable events, and an observation mask M. The control specification is given by a deterministic state machine $R = (Q, \Sigma, \delta, q_0)$. The control task is to design a supervisor S such that the behavior of the controlled plant equals the specification language, i.e., $L(G^{Id}_{\Sigma}\|^{M}_{\Sigma_c} S) = L(R)$. We show that both the existence and synthesis of a supervisor can be determined polynomially.

In [9], SSC based control under partial observation was studied, allowing the supervisor to be nondeterministic. Due to the use of SSC, supervisor was required to be (Σ_u, M)-compatible [9]. It was shown that a necessary and sufficient condition for the existence of a (Σ_u, M)-compatible supervisor such that the behavior of the controlled system equals the specification language is *achievability*.

Definition 2: [9] Let $K \subseteq \Sigma^*$, then (i) K is said to be Σ_u-*controllable* with respect to Σ^* if $\forall s \in pr(K)$ and $\forall a \in \Sigma_u$, $sa \in pr(K)$. (ii) K is said to be M-*recognizable* with respect to Σ^* if $\forall s, t \in \Sigma^*$ and $\forall a \in \Sigma$ with $M(a) = M(\epsilon)$, $sat \in pr(K) \Rightarrow sa^*t \subseteq pr(K)$. (iii) K is said to be (Σ_u, M)-*achievable* with respect to Σ^* ((Σ^*, Σ_u, M)-achievable for short) if K is Σ_u-controllable and M-recognizable with respect to Σ^*, and $\forall s, t \in \Sigma^*$, $\forall a \in \overline{\Sigma}$, $b \in \Sigma_u$ with $M(a) = M(b)$, $sat \in pr(K) \Rightarrow \{sbt\} \subseteq pr(K)$. (iv) $K \subseteq L(G)$ is $(L(G), \Sigma_u, M)$-*achievable* if exists (Σ^*, Σ_u, M)-achievable K' such that $pr(K) = pr(K') \cap L(G)$.

We obtain the following necessary and sufficient condition for the existence of a PSCM-based supervisor enforcing a given specification.

Theorem 3: Given G, R, the set of controllable events Σ_c and a mask M, there exists a supervisor S such that $L(G^{Id}_{\Sigma}\|^{M}_{\Sigma_c} S) = L(R)$ if and only if $L(R) \subseteq L(G)$ and $L(R)$ is $(L(G), \Sigma_u, M)$-achievable.

The following algorithm constructs a supervisor from a specification.

Algorithm 2: Consider a state machine $R = (Q, \Sigma, \delta, Q_0)$.

1) For every transition (q, b, q_1) with either $M(b) = M(\epsilon)$ or $\exists (q, b', q_2)$ such that $M(b) = M(b') \neq M(\epsilon)$, replace (q, b, q_1) by a pair of transitions (q, ϵ, q') and (q', b, q_1), where q' is a newly added state. Denote the resulting state machine as $R' = (Q', \Sigma, \delta', Q'_0)$.
2) For every state q' of R', every event $b \in \Sigma_u \cap (\Sigma - M(\epsilon))$ such that $M(b) \cap \Sigma(q') = \emptyset$, add a transition $(q', b, dump)$, where $dump$ is an added state. Denote the resulting state machine as $\overline{R} := (\overline{Q}, \Sigma, \overline{\delta}, \overline{Q}_0)$.

Remark 2: It follows from Theorem 3 that the existing tests for achievability, which is of complexity $O(|G||R|^2)$, can be applied to verify the existence of a supervisor. Further if the existence condition is satisfied, a supervisor can be obtained by applying Algorithm 2 to a generator of the specification language, implying the synthesis of a supervisor is of complexity $O(|R|)$. Note that in general the state machine obtained by Algorithm 2 is not (Σ_u, M)-compatible. Thus, by using the control mechanism of PSCM, the requirement of (Σ_u, M)-compatibility of a supervisor has been removed, as desired.

VI. An Illustrative Example

We illustrate Theorem 3 through a manufacturing example, taken from [9]. The manufacturing system consists of one robot, two workstations and two storage-stations. The robot moves among the workstations and storage-stations on a guide rail. Initially, the robot departs from workstation 1 (event a). Then it picks up a part either from storage-station 1 (event b_1) or storage-station 2 (event b_2), and delivers the part to workstation 2 for processing (event c). After the processing, robot returns the part to a storage-station. After returning parts, robot goes back to workstation 1 and can repeat the whole process. A state machine model G of the system is drawn in Figure 2.

Not returning the part to its original storage-station is not desirable. The specification R, also shown in Figure 2, gives the desired behavior. According to the specification, after processing, the robot returns the part to its original storage-station. The rest of the behavior is the same as the one feasible in the system.

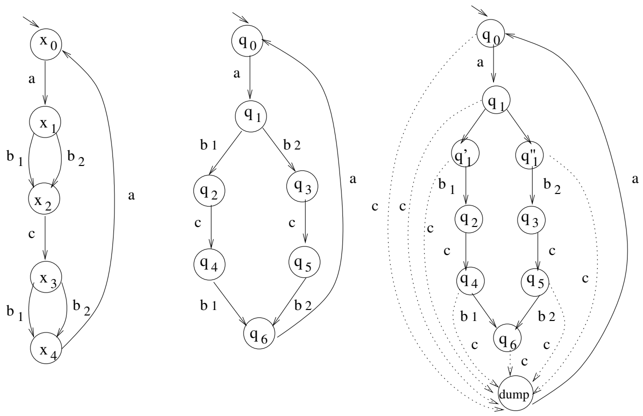

Fig. 2. G (left), R (middle), and $\overline{R}$ (right)

We require that the part must be delivered to workstation 2 for processing, i.e., the event c is uncontrollable. Also, only events a and c are completely observable. Events b_1 and b_2 are observationally indistinguishable. Thus, we have $\Sigma = \{a, b_1, b_2, c\}$, $\Sigma_c = \{a, b_1, b_2\}$, and the observation mask M is given by, $M(a) = \{a\}$, $M(b_1) = M(b_2) = \{b_1, b_2\}$ and $M(c) = \{c\}$. It can be verified that $L(R)$ is not observable, i.e., a deterministic supervisor does not exist. However, $L(R)$ is $(L(G), \Sigma_u, M)$-achievable and so from Theorem 3, a PSCM-based supervisor does exist. (Thanks to PSCM-based control formalism developed here that allows supervisor to be nondeterministic.)

We use Algorithm 2 to construct $\overline{R}$ acts as a supervisor.

At state q_1 of R, $M(b_1) = M(b_2)$, by step 1 of Algorithm 2, we replace transition (q_1, b_1, q_2) (resp., (q_1, b_2, q_3)) by a pair of transitions (q_1, ϵ, q_1') and (q_1', b_1, q_2) (resp., (q_1, ϵ, q_2'') and (q_2'', b_2, q_3)). Next by step 2 of Algorithm 2, since $c \in \Sigma_u - (\Sigma - M(\epsilon))$ and $M(c) = \{c\}$, we add transitions on c from states $q_0, q_1, q_1', q_1'', q_4, q_5$, and q_6 to the newly added "$dump$" state.

We obtain the state machine $\overline{R}$ drawn in Figure 2. One can verify that $L(G_\Sigma^{Id} \|_{\Sigma_c}^{M} \overline{R}) = L(R)$, i.e., $\overline{R}$ serves as a desired supervisor. It should be noted that $\overline{R}$ is not (Σ_u, M)-compatible. As an example, the uncontrollable event c is undefined at the $dump$ state.

VII. Conclusion

In this paper we introduced the notion of prioritized synchronous composition under mask (PSCM) to model interaction of discrete event systems under partial observation. We showed that when PSCM is adopted as a mechanism of control, not only the control & observation-compatibility requirements are removed of a supervisor, the existence condition is given by achievability that is weaker than controllability and observability combined. (The weaker condition is required since the supervisor is allowed to be nondeteministic.) This suggests that the notion of PSCM, introduced in the paper is an appropriate generalization of PSC to account for partial observation. We also showed that both existence and synthesis of a PSCM-based supervisor is polynomially solvable.

References

[1] S. Balemi. Input/output discrete event processes and communication delays. *Discrete Event Dynamical Systems: Theory and Applications*, 4(1):41–85, 1994.

[2] M. Fabian. *On Object Oriented Nondeterministic Supervisory Control*. PhD thesis, Control Engineering Laboratory, Chalmers University, Goteberg, 1995.

[3] M. Heymann. Concurrency and discrete event control. *IEEE Control Systems Magazine*, 10(4):103–112, 1990.

[4] M. Heymann and F. Lin. Discrete-event control of nondeterministic systems. *IEEE Transactions on Automatic Control*, 43(1):3–17, January 1998.

[5] M. Heymann and G. Meyer. Algebra of discrete event processes. Technical Report NASA 102848, NASA Ames Research Center, Moffett Field, CA, June 1991.

[6] S. Jiang and R. Kumar. Supervisory control of nondeterministic discrete event systems with driven events via masked prioritized synchronization. *IEEE Transactions on Automatic Control*, 47(9):1438–1449, 2002.

[7] R. Kumar, V. K. Garg, and S. I. Marcus. On controllability and normality of discrete event dynamical systems. *Systems and Control Letters*, 17(3):157–168, 1991.

[8] R. Kumar and M. Heymann. Masked prioritized synchronization for interaction and control of discrete event systems. *IEEE Transactions on Automatic Control*, 45(11):1970–1982, 2000.

[9] R. Kumar, S. Jiang, C. Zhou, and W. Qiu. Control using nondeterministic superviors for partially observed discrete event systems. In *Proceedings of 2004 American Control Conference*, Boston, MA, June 2004. 4472-4476.

[10] R. Kumar and M. A. Shayman. Non-blocking supervisory control of nondeterministic systems via prioritized synchronization. *IEEE Transactions on Automatic Control*, 41(8):1160–1175, August 1996.

[11] R. Kumar and M. A. Shayman. Supervisory control of real-time systems using prioritized synchronization. In R. Alur, T. Henzinger, and E. Sontag, editors, *Hybrid Systems III*, volume 1066 of *Lecture Notes in Computer Science*. Springer-Verlag, New York, 1996.

[12] R. Kumar and M. A. Shayman. Centralized and decentralized supervisory control of nondeterministic systems under partial observation. *SIAM Journal of Control and Optimization*, 35(2):363–383, March 1997.

[13] P. J. Ramadge and W. M. Wonham. Supervisory control of a class of discrete event processes. *SIAM Journal of Control and Optimization*, 25(1):206–230, 1987.

[14] P. J. Ramadge and W. M. Wonham. The control of discrete event systems. *Proceedings of IEEE: Special Issue on Discrete Event Systems*, 77:81–98, 1989.

[15] M. Shayman and R. Kumar. Supervisory control of nondeterministic systems with driven events via prioritized synchronization and trajectory models. *SIAM Journal of Control and Optimization*, 33(2):469–497, March 1995.

[16] M. A. Shayman and R. Kumar. Process objects/masked composition: An object oriented approach for modeling and control of discrete event systems. *IEEE Transactions on Automatic Control*, 44(10):1864–1869, 1999.