Department of Computing and Software

Faculty of Engineering — McMaster University

Decentralized Control Using the Hierarchical Interface-based Supervisory Control Approach

v 1.1

by

Huailiang Liu, Ryan J. Leduc, and S. L. Ricker

CAS Report Series **CAS-14-10-RL**

Department of Computing and Software **December 2015**
Information Technology Building
McMaster University
1280 Main Street West Hamilton, Ontario, Canada L8S 4K1

Copyright © 2015

Decentralized Control Using the Hierarchical Interface-based Supervisory Control Approach

v 1.1

Huailiang Liu[1], Ryan J. Leduc[1], and S. L. Ricker[2]

[1] Department of Computing and Software, Faculty of Engineering,
McMaster University, Hamilton, Ontario, Canada
[2] Department of Mathematics and Computer Science
Mount Allison University, Sackville, NB E4L 1E6, Canada

Technical Report CAS-14-10-RL
Department of Computing and Software
McMaster University

January 28, 2016

Abstract

In decentralized control, agents have only a partial view and partial control of the system and must cooperate to achieve the control objective. To synthesize a decentralized control solution, a specification must satisfy the co-observability property. Existing co-observability verification methods require the (possibly intractable) construction of the complete system.

To increase the scalability of decentralized control, we introduce the Hierarchical Interface-Based Decentralized Supervisory Control (HIDSC) framework that extends the existing Hierarchical Interface-Based Supervisory Control (HISC) approach.

To adapt co-observability for HIDSC, we propose a per-component definition of co-observability along with a verification strategy that requires examination of only a single component at a time. Finally, we provide and prove the necessary and sufficient conditions for supervisory control existence in the HIDSC framework and illustrate our approach with an example. As the entire system model never needs to be constructed, HIDSC can provide significant computational savings.

Keywords: discrete-event systems, supervisory control, decentralized control, Hierarchical Interface-based Supervisory Control

Contents

1 Introduction

One of the main challenges in the control of discrete-event systems (DES) [CL08, RW87, WR87, Won14] is the combinatorial explosion of the product state space. The Hierarchical Interface-Based Supervisory Control (HISC) framework proposed in [HJDQ$^+$10, Led02, Led09, LBLW05, LLD06, LLW05] can help alleviate the state-space explosion problem. HISC provides a set of local properties that can be used to verify global properties, such as nonblocking and controllability, so that the complete system model never needs to be constructed. The sufficient conditions of HISC allow the independent design and verification of different levels, ensuring that a change to one level of the hierarchy will not impact the others.

However, the current HISC framework does not support decentralized control problems that arise naturally through the investigation of a large variety of distributed systems, such as communication networks, integrated sensor networks, networked control systems and automated guided vehicular systems. Decentralized control of DES focuses on problems where multiple agents each control and observe some events in a system and must together achieve some prescribed goal. A decentralized approach is used when the physical system is such that controllers implemented at different locations would naturally only be able to see and affect events occurring in their local vicinity, and have no access to other events. The goal is to be able to implement a set of decentralized controllers that produce the same control actions as a centralized controller with full view and control of the system.

The synthesis of decentralized supervisors requires that the specification satisfies a decentralized property called co-observability [RW92]. Nevertheless, when the system is very large and composed of many sub-systems, checking co-observability using the existing monolithic method [RW95] requires the construction of the complete system model, which may be intractable due to the state-space explosion problem.

In supervisory control of DES, the computation and complexity for many of the control solutions entail only polynomial effort in the model's state size. The computation and complexity is worse in the case of control with partial observations: some problems with full observation are polynomial; however, they are exponential when the situation is partially observable [RYL03, RW95, TL09, Tri04, Tsi89, YL02].

In [Liu15, LLMR14], we introduced an incremental approach for checking co-observability based on the work of Brandin et al. [BMM04] for verifying controllability. Our method greatly increased the size of systems that could be verified, but it also struggled as the state size and number of decentralized controllers increased.

To address the above issue, we propose an approach called the Hierarchical Interface-Based Decentralized Supervisory Control (HIDSC) framework that extends HISC to manage decentralized control problems. We introduce a per-component co-observability definition which does not require the synchronization of all components. We then prove that if a system satisfies the per-component co-observability definition, it is globally co-observable. Further, we provide and prove the necessary and sufficient conditions for supervisor existence in HIDSC. Most of the material in this paper first appeared in our conference paper [LLR15]. The current paper adds full proofs, a more complicated example as well as experimental results.

To the best of our knowledge, there is no existing work where HISC supports decentralized discrete-event control architecture. Although there is some literature proposing hierarchical control of "decentralized" discrete-event systems [SB11, SM06, SMP08], these approaches assume full observation and thus they are "decentralized" in terms of architecture, not in terms of observations.

This paper is organized as follows. In Section 2 we review the relevant definitions and results from supervisory control theory. Section 3 reviews the HISC architecture. In Section 4, we introduce our new HIDSC framework. In Section 5, we illustrate our HIDSC approach with an example. We then present conclusions and future work in Section 6.

2 Preliminaries

This section provides a brief review of the key DES concepts used in this paper. For more details please refer to [CL08, Won14].

2.1 Languages and DES

Event sequences and languages are simple ways to describe DES behaviour. Let Σ be a finite set of distinct symbols (*events*), and let Σ^+ be the set of all finite nonempty sequences of events. Let $\Sigma^* := \Sigma^+ \cup \{\epsilon\}$ be the set of all finite sequences of events plus ϵ, the *empty string*. A language L over Σ is any subset $L \subseteq \Sigma^*$.

The *concatenation* of two strings $s,\ t \in \Sigma^*$, is written as st. Languages and alphabets can also be concatenated. For $L \subseteq \Sigma^*$ and $\Sigma' \subseteq \Sigma$, the concatenation of language L and event set Σ' is defined as $L\Sigma' := \{s\sigma | s \in L,\ \sigma \in \Sigma'\}$.

For strings $s,\ t \in \Sigma^*$, we say that t is a *prefix* of s (written $t \leq s$) if $s = tu$, for some $u \in \Sigma^*$. We also say that t can be *extended* to s. The *prefix closure* $\overline{L}$ of a language $L \subseteq \Sigma^*$ is defined as follows: $\overline{L} := \{t \in \Sigma^* | t \leq s \text{ for some } s \in L\}$. A language L is said to be *prefix-closed* if $L = \overline{L}$.

Let $\text{Pwr}(\Sigma)$ denote the power set of Σ (i.e., the set of all subsets of Σ). For language L, the eligibility operator $\text{Elig}_L : \Sigma^* \to \text{Pwr}(\Sigma)$ is given by $\text{Elig}_L(s) := \{\sigma \in \Sigma\, | s\sigma \in L\}$ for $s \in \Sigma^*$.

Let $\Sigma = \Sigma_1 \cup \Sigma_2$, $L_1 \subseteq \Sigma_1^*$, and $L_2 \subseteq \Sigma_2^*$. Let $i \in \{1, 2\}$, $s \in \Sigma^*$, and $\sigma \in \Sigma$. To capture the notion of partial observation, we define the *natural projection* $P_i :\ \Sigma^* \to \Sigma_i^*$ according to:

$$\begin{aligned} P_i(\epsilon) &= \epsilon, \qquad\qquad P_i(\sigma) = \begin{cases} \epsilon, & \text{if } \sigma \notin \Sigma_i; \\ \sigma, & \text{if } \sigma \in \Sigma_i \end{cases} \\ P_i(s\sigma) &= P_i(s)P_i(\sigma) \end{aligned}$$

Given any language $L \subseteq \Sigma^*$, the *natural projection of a language* L is $P_i(L) := \{P_i(s) \mid s \in L\}$. This is sometimes abbreviated to $P_i\, L$.

The *inverse projection* P_i^{-1}: $Pwr(\Sigma_i^*) \to Pwr(\Sigma^*)$ is defined over subsets of languages. Given any $L \subseteq \Sigma_i^*$, the inverse projection of L is defined as: $P_i^{-1}(L) := \{s \mid P_i(s) \in L\}$.

A DES is represented as a tuple $\mathbf{G} := (Q,\ \Sigma,\ \delta,\ q_0,\ Q_m)$, with state set Q, alphabet set Σ, partial transition function $\delta :\ Q \times \Sigma \to Q$, initial state q_0, and set of marker states Q_m. We use $\delta(q,\ \sigma)!$ to represent that δ is defined for $\sigma \in \Sigma$ at state $q \in Q$. Function δ can be extended to Σ^* by defining $\delta(q,\ \epsilon) := q$ and $\delta(q,\ s\sigma) := \delta(\delta(q,\ s),\ \sigma)$, provided that $q' = \delta(q,\ s)!$ and $\delta(q',\ \sigma)!$, for $s \in \Sigma^*$ and $q \in Q$. We will always assume that a DES has a finite state and event set, and is deterministic. By deterministic, we mean the DES has a single initial state and at most one transition defined at a given state for any $\sigma \in \Sigma$.

For DES $\mathbf{G}$, its *closed behaviour* is denoted by $L(\mathbf{G}) := \{s \in \Sigma^* | \delta(q_0,\ s)!\}$ and its *marked behaviour* by $L_m(\mathbf{G}) := \{s \in L(\mathbf{G}) |\ \delta(q_o, s) \in Q_m\}$.

Definition 2.1. *A DES* $\mathbf{G}$ *is said to be* nonblocking *if*

$$\overline{L_m(\mathbf{G})} = L(\mathbf{G}).$$

The above is a simple form of deadlock checking.

Definition 2.2. *Let $K \subseteq L_m(\mathbf{G}) \subseteq \Sigma^*$. We say that the language K is $L_m(\mathbf{G})$-*closed *if*

$$K = \overline{K} \cap L_m(\mathbf{G}).$$

We note that K being $L_m(\mathbf{G})$-closed means it contains all of its prefixes that belong to $L_m(\mathbf{G})$.

The *synchronous product of languages* L_1 and L_2, denoted by $L_1||L_2$, is defined to be $L_1||L_2 := P_1^{-1}(L_1) \cap P_2^{-1}(L_2)$. If both L_1 and L_2 are over the same event set Σ, then $L_1||L_2 = L_1 \cap L_2$.

Definition 2.3. *Let $\mathbf{G}_i = (Q_i,\ \Sigma_i,\ \delta_i,\ q_{0,i},\ Q_{mi})$, $i = 1, 2$. We define the* synchronous product *of $\mathbf{G}_1$ and $\mathbf{G}_2$ as:*

$$\mathbf{G}_1||\mathbf{G}_2 = (Q_1 \times Q_2,\ \Sigma_1 \cup \Sigma_2,\ \delta,\ (q_{0,1},\ q_{0,2}),\ Q_{m1} \times Q_{m2}),$$

where $\delta((q_1, q_2),\ \sigma)$ is defined as:

$$\begin{cases} (\delta_1(q_1,\sigma), \delta_2(q_2,\sigma)), \textit{if} \sigma \in \Sigma_1 \cap \Sigma_2, \delta_1(q_1,\sigma)!, \delta_2(q_2,\sigma)!; \\ (\delta_1(q_1,\sigma), q_2),\ \textit{if}\ \sigma \in \Sigma_1 \backslash \Sigma_2\ \textit{and}\ \delta_1(q_1,\sigma)!; \\ (q_1, \delta_2(q_2,\sigma)),\ \textit{if}\ \sigma \in \Sigma_2 \backslash \Sigma_1\ \textit{and}\ \delta_2(q_2,\sigma)!; \\ \textit{undefined, otherwise.} \end{cases}$$

We thus have for DES $\mathbf{G} = \mathbf{G}_1||\mathbf{G}_2$ that $L_m(\mathbf{G}) = L_m(\mathbf{G}_1)||L_m(\mathbf{G}_2)$, and $L(\mathbf{G}) = L(\mathbf{G}_1)||L(\mathbf{G}_2)$.

In supervisory control, the event set Σ is partitioned into two disjoint sets: the *controllable* event set Σ_c and the *uncontrollable* event set Σ_{uc}. Controllable events can be prevented from happening (disabled) by a supervisor, while uncontrollable events cannot be disabled. The following definition checks to see if we can ensure our system stays within the behavior specified by $\overline{K}$.

Definition 2.4. *Let K and $L = \overline{L}$ be languages over event set Σ. K is said to be* controllable *with respect to L and Σ_{uc} if*

$$\overline{K}\Sigma_{uc} \cap L \subseteq \overline{K}.$$

We will define a supervisory control which is an abstract way to describe control behavior. First we need to define control patterns.

Definition 2.5. *A* control pattern *is a subset of Σ that contains all uncontrollable events. It represents the events to be currently enabled. The set of all control patterns is:*

$$\Gamma := \{\gamma \in Pwr(\Sigma) |\ \gamma \supseteq \Sigma_{uc}\}.$$

Definition 2.6. *A* supervisory control *for plant $\mathbf{G}$ is any map*

$$V : L(\mathbf{G}) \rightarrow \Gamma.$$

2.2 Decentralized Control

For *decentralized control,* there is an index set of $N > 1$ decentralized controllers, $D = \{1, ..., N\}$. These controllers have only a partial view of the system behaviour and control only a subset of the controllable events. To describe events that each *decentralized controller* $i \in D$ controls, we use the notation $\Sigma_{c,i} \subseteq \Sigma_c$, where $\cup_{i=1}^{N} \Sigma_{c,i} = \Sigma_c$. We refer to the set of controllers that control $\sigma \in \Sigma_c$ as $D_c(\sigma) := \{i \in D \mid \sigma \in \Sigma_{c,i}\}$.

To describe events that each decentralized controller $i \in D$ observes, we use the notation $\Sigma_{o,i} \subseteq \Sigma_o$, where $\cup_{i=1}^{N} \Sigma_{o,i} = \Sigma_o$. We refer to the set of controllers that observe $\sigma \in \Sigma_o$ by $D_o(\sigma) := \{i \in D \mid \sigma \in \Sigma_{o,i}\}$. Correspondingly, the natural projection describing the partial view of each controller is denoted by $P_i : \Sigma^* \rightarrow \Sigma_{o,i}^*$, for $i \in D$.

For decentralized control with a *conjunctive architecture* [RW92], the fusion rule is the conjunction of all *local control decisions*, i.e., an event is globally enabled if not locally disabled. We use the conjunctive architecture in this paper.

For decentralized control, we define a group of local partial-observation decentralized supervisors to exercise control over the plant. The following definition describes the decision rules of individual decentralized supervisors and the conjunction rule that combines their local control decisions into global control decisions.

Definition 2.7. *Let $K \subseteq \Sigma^*$ be the desired behavior and let $i \in D$. Then the decision rule for a* local partial-observation decentralized supervisor for $\mathbf{G}$ *is a map $\mathcal{S}_{P_i} : L(\mathbf{G}) \rightarrow \Gamma$ defined for $t \in L(\mathbf{G})$ as $\mathcal{S}_{P_i}(t) := (\Sigma \backslash \Sigma_{c,i}) \cup \{\sigma \in \Sigma_{c,i} \mid P_i^{-1}[P_i(t)]\sigma \cap \overline{K} \cap L(\mathbf{G}) \neq \emptyset\}$. The* conjunction *of the $\mathcal{S}_{P_i}$, denoted by $\mathcal{S}_{Con} : L(\mathbf{G}) \rightarrow \Gamma$, is defined as: $\mathcal{S}_{Con}(t) := \cap_{i=1}^{N} \mathcal{S}_{P_i}(t)$.*

We note that $\mathcal{S}_{Con}$ is a supervisory control for $\mathbf{G}$ and that $\mathcal{S}_{P_i}(t) = \mathcal{S}_{P_i}(P_i(t))$ as the natural projection is idempotent, i.e., $P_i(t) = P_i(P_i(t))$.

We now define the closed behaviour for the closed-loop system of $\mathbf{G}$ under the control of $\mathcal{S}_{Con}$.

Definition 2.8. *Given $\mathbf{G}$ and $\mathcal{S}_{Con}$, the resulting* closed-loop system *is denoted by $\mathcal{S}_{Con}/\mathbf{G}$. The system's* closed behaviour *$L(\mathcal{S}_{Con}/\mathbf{G})$, is recursively defined as follows:*

I) $\epsilon \in L(\mathcal{S}_{Con}/\mathbf{G})$

II) $t \in L(\mathcal{S}_{Con}/\mathbf{G})$, $\sigma \in \mathcal{S}_{Con}(t)$, and $t\sigma \in L(\mathbf{G})$ if and only if $t\sigma \in L(\mathcal{S}_{Con}/\mathbf{G})$.

The following is the definition of decentralized supervisory control.

Definition 2.9. *Given $\mathbf{G}$ and $\mathcal{S}_{Con}$, we say $\mathcal{S}_{Con}$ is a* decentralized supervisory control for $\mathbf{G}$ *if the decision rule is defined as in Definition 2.7, and the resulting closed-loop system and closed behaviour is defined as in Definition 2.8.*

The following is the definition of nonblocking decentralized supervisory control. It states that the marked language of the closed-loop system is the set of strings marked by $\mathbf{G}$ and still possible in the system's closed-loop behaviour.

Definition 2.10. *We say that $\mathcal{S}_{Con}$ is a* nonblocking decentralized supervisory control (NDSC) *for $\mathbf{G}$ if $\overline{L_m(\mathcal{S}_{Con}/\mathbf{G})} = L(\mathcal{S}_{Con}/\mathbf{G})$ where $L_m(\mathcal{S}_{Con}/\mathbf{G}) := L(\mathcal{S}_{Con}/\mathbf{G}) \cap L_m(\mathbf{G})$.*

We now state the co-observability property which was introduced in [RW92]. The following is the definition of co-observability adapted from [BL00, RW92]. As we will see in Theorem 2.1, co-observability is a key property to ensure that we can synthesize decentralized controllers that cooperate to ensure that the supervised system generates the behaviour specified by language K.

Definition 2.11. *Let K, $L = \overline{L}$ be languages over event set Σ. Let $D = \{1, ..., N\}$ be an index set. Let $\Sigma_{c,i} \subseteq \Sigma$ and $\Sigma_{o,i} \subseteq \Sigma$ be sets of controllable and observable events, respectively, for $i \in D$, where $\Sigma_c = \cup_{i=1}^{N} \Sigma_{c,i}$ and $D_c(\sigma) := \{i \in D \mid \sigma \in \Sigma_{c,i}\}$. Let $P_i : \Sigma^* \to \Sigma_{o,i}^*$ be natural projections. A language K is said to be co-observable with respect to L, $\Sigma_{o,i}$, $\Sigma_{c,i}$, $i \in D$, if*

$$(\forall t \in \overline{K} \cap L)\ (\forall \sigma \in \Sigma_c)\ t\sigma \in L \backslash \overline{K} \Rightarrow (\exists i \in D_c(\sigma))\ P_i^{-1}[P_i(t)]\sigma \cap \overline{K} \cap L = \emptyset.$$

In essence, co-observability states that if $\overline{K}$ disables event $\sigma \in \Sigma_c$ which is possible in L after string t, there must exist at least one decentralized controller that can disable σ and do so unambiguously. Note that in the definition of co-observability, when there is only one controller, i.e., $D = \{1\}$, the property is called *observability* [LW88]. Since the specification K is not necessarily a subset of L, unlike the original definition, we do not require that $K \subseteq L$. Instead of checking all strings in $\overline{K}$, we check all strings in $\overline{K} \cap L$.

In the following sections, when there is no ambiguity, instead of saying that K is co-observable with respect to L, $\Sigma_{o,i}$, $\Sigma_{c,i}$, $i \in D$, we will say that K is co-observable w.r.t. L.

Theorem 2.1 states the standard nonblocking decentralized supervisory control existence theorem that requires that K be $L_m(\mathbf{G})$-closed.

Theorem 2.1 ([CL08]). *Consider DES $\mathbf{G} = (Q,\ \Sigma,\ \delta,\ q_0,\ Q_m)$, where $\Sigma_{uc} \subseteq \Sigma$ is the set of uncontrollable events, $\Sigma_c = \Sigma \backslash \Sigma_{uc}$ is the set of controllable events, and $\Sigma_o \subseteq \Sigma$ is the set of observable events. For each site i, where $i = 1, ..., N$ consider the set of controllable events $\Sigma_{c,i}$ and the set of observable events $\Sigma_{o,i}$; overall, $\bigcup_{i=1}^{N} \Sigma_{c,i} = \Sigma_c$ and $\bigcup_{i=1}^{N} \Sigma_{o,i} = \Sigma_o$. Let P_i be the natural projection from Σ^* to $\Sigma_{o,i}^*$, where $i = 1, ..., N$. Consider also the language $K \subseteq L_m(G)$, where $K \neq \emptyset$. There exists a nonblocking decentralized supervisor S_{con} for $\mathbf{G}$ such that $L_m(S_{con}/G) = K$ and $L(S_{con}/G) = \overline{K}$ if and only if the following three conditions hold:*

1. *K is controllable with respect to $L(G)$ and Σ_{uc};*
2. *K is co-observable with respect to $L(G)$, $\Sigma_{o,i}$, and $\Sigma_{c,i}$, $i = 1, ..., N$;*
3. *K is $L_m(\mathbf{G})$-closed.*

2.3 Decentralized Control with Marking

We will now extend the existing work by introducing a generalization of NDSC in which the supervisory action also includes marking as well as control. This will allow supervisors to add marking information which makes them more expressive. The marked language of the closed-loop system is now defined to be the set of strings marked by $K \subseteq L_m(\mathbf{G})$ that are still possible in the system's closed-loop behaviour. This new definition will allow us to later introduce a decentralized supervisory control existence result which does not require that K be $L_m(\mathbf{G})$-closed.

Definition 2.12. *Let $K \subseteq L_m(\mathbf{G})$. We say that $\mathcal{S}_{con}$ is a* marking nonblocking decentralized supervisory control (MNDSC) *for $(K,\ \mathbf{G})$ if $\overline{L_m(\mathcal{S}_{Con}/\mathbf{G})} = L(\mathcal{S}_{Con}/\mathbf{G})$ where $L_m(\mathcal{S}_{con}/\mathbf{G}) := L(\mathcal{S}_{con}/\mathbf{G}) \cap K$.*

The next definition creates an equivalence between theoretical decentralized supervisory controls and DES supervisors. The idea is that the marking and control information of $\mathcal{S}_{Con}$ is represented by specification $\mathbf{H}$, and that the closed-loop behaviour of $\mathbf{H}||\mathbf{G}$ is equivalent to $\mathcal{S}_{con}/\mathbf{G}$.

Definition 2.13. *Let $\mathcal{S}_{Con}$ be a MNDSC for plant $\mathbf{G} = (Q, \Sigma, \delta, q_0, Q_m)$ and $K \subseteq L_m(\mathbf{G})$, with $L_m(\mathcal{S}_{Con}/\mathbf{G}) = L(\mathcal{S}_{con}/\mathbf{G}) \cap K$ and $L(\mathcal{S}_{Con}/\mathbf{G}) = \overline{K}$. Let $\mathbf{H} = (X, \Sigma, \xi, x_0, X_m)$ be a specification automaton. We say that $\mathbf{H}||\mathbf{G}$ has* equivalent MNDSC behaviour *with $\mathcal{S}_{Con}/\mathbf{G}$, if $K = L_m(\mathbf{H}) \cap L_m(\mathbf{G})$ and $\overline{K} = L(\mathbf{H}) \cap L(\mathbf{G})$. Alternatively, we say that $\mathbf{H}$ is an* equivalent theoretical implementation of MNDSC $\mathcal{S}_{Con}$ for $\mathbf{G}$.

In this paper, we will focus on MNDSC. In particular, it will allow us to later introduce a decentralized supervisor existence result that relies on the closed-loop system $\mathbf{H}||\mathbf{G}$ to be nonblocking, instead of the existing results that require K to be $L_m(\mathbf{G})$-closed. This is essential to adapting decentralized control to the HISC approach as HISC provides a scalable method to verify nonblocking, but not $L_m(\mathbf{G})$-closure.

We note that in decentralized control, there is no real implementation of the centralized supervisor $\mathbf{H}$. The above MNDSC $\mathcal{S}_{Con}$, defined as the control policy of the conjunction of a group of decentralized supervisors, is the real supervisor. We also note that for an HISC system, $\mathbf{H}$ will correspond to the theoretical flat supervisor of the system defined in Section 3, and will be used to determine if the flat system is nonblocking.

3 HISC Architecture

The HISC [Led02, Led09, LBLW05, LLD06, LLW05] approach decomposes a system into a high-level subsystem which communicates with $n \geq 1$ parallel low-level subsystems through separate *interfaces* that restrict the interaction of the subsystems. The high-level subsystem communicates with each low-level subsystem through a separate interface.

In HISC there is a master-slave relationship. A high-level subsystem sends a command to a particular low-level subsystem, which then performs the indicated task and returns a response (answer). Figure 1 shows conceptually the structure and information flow of the system. The overall structure of the system is shown in Figure 2. This style of interaction is enforced by an interface that mediates communication between the two subsystems. All system components, including the interfaces, are modeled as automata.

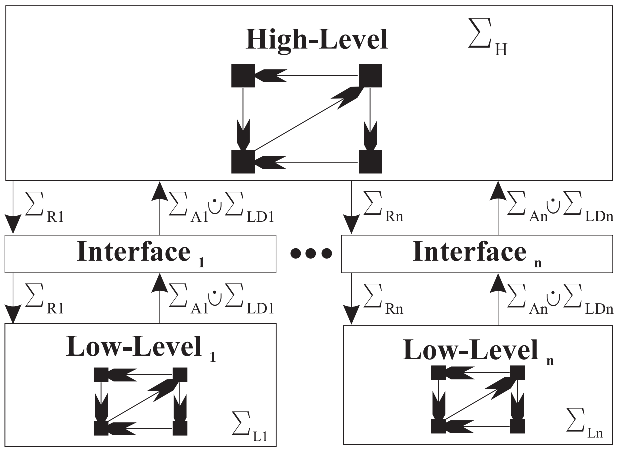


Figure 1: Interface Block Diagram with Low Data Events.

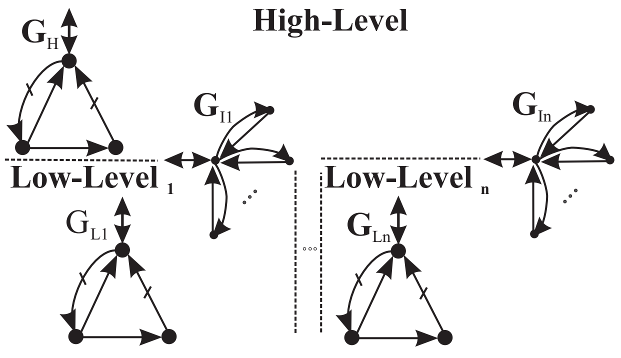


Figure 2: Two Tiered Structure of Parallel System

To restrict information flow and decouple the subsystems, the system alphabet is partitioned into pairwise disjoint alphabets:

$$\Sigma := \Sigma_H \;\dot{\cup} \dot{\bigcup_{j=1,\ldots,n}} [\Sigma_{L_j} \dot{\cup} \Sigma_{R_j} \dot{\cup} \Sigma_{A_j} \dot{\cup} \Sigma_{LD_j}] \tag{1}$$

where we use $\dot{\cup}$ to represent disjoint union.

The events in Σ_H are called *high-level events* and the events in Σ_{L_j} are the j^{th} *low-level events* ($j = 1, \ldots, n$) as these events appear only in the high level and j^{th} low-level subsystem models, $\mathbf{G}_H$ and $\mathbf{G}_{L_j}$ respectively. We then have $\mathbf{G}_H$ defined over event set $\Sigma_H \dot{\cup} (\dot{\cup}_{j \in \{1,\ldots,n\}} [\Sigma_{R_j} \dot{\cup} \Sigma_{A_j} \dot{\cup} \Sigma_{LD_j}])$ and $\mathbf{G}_{L_j}$ defined over event set $\Sigma_{L_j} \dot{\cup} \Sigma_{R_j} \dot{\cup} \Sigma_{A_j} \dot{\cup} \Sigma_{LD_j}$. We model the j^{th} interface by DES $\mathbf{G}_{I_j}$, which is defined over event set $\Sigma_{R_j} \dot{\cup} \Sigma_{A_j} \dot{\cup} \Sigma_{LD_j}$. For the remainder of this paper, we assume $j \in \{1, \ldots, n\}$.

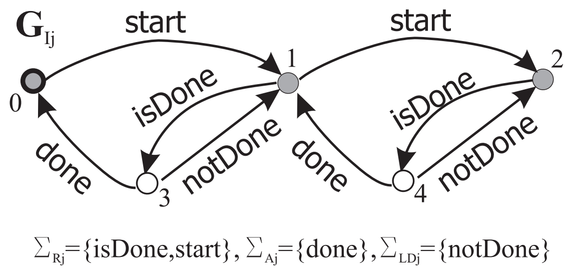
The events in Σ_{R_j}, called *request events*, represent commands sent from the high-level subsystem to the j^{th} low-level subsystem. The events in Σ_{A_j} are *answer events* and represent the low-level subsystem's responses to the request events. The events in Σ_{LD_j} are called *low data events* which provide a means for a low level to send information (data) through the interface. Request, answer, and low data events are collectively known as the set of LD *interface events*, defined as $\Sigma_I := \dot{\cup}_{k \in \{1,\ldots,n\}} [\Sigma_{R_k} \dot{\cup} \Sigma_{A_k} \dot{\cup} \Sigma_{LD_k}]$, and $\mathbf{G}_{I_j}$is an LD interface as defined below.

Definition 3.1. *The j^{th} interface DES $\mathbf{G_{I_j}} = (X_j, \Sigma_{I_j}, \xi_j, x_{o_j}, X_{m_j})$ is an* LD interface *if the following properties are satisfied:*

1. $x_{o_j} \in X_{m_j}$
2. $(\forall x \in X_{m_j})(\forall \sigma \in \Sigma_{I_j})\, \xi_j(x, \sigma)! \Rightarrow [\sigma \in \Sigma_{R_j}] \;\vee\; [\sigma \in \Sigma_{LD_j} \wedge \xi_j(x, \sigma) \in X_{m_j}]$
3. $(\forall x \in X_j - X_{m_j})(\forall \sigma \in \Sigma_{I_j})\, \xi_j(x, \sigma)! \Rightarrow$
   $[\sigma \in \Sigma_{A_j} \wedge \xi_j(x, \sigma) \in X_{m_j}] \vee [\sigma \in \Sigma_{LD_j}]$

Figure 3 shows an example of an LD interface. It could correspond to a machine at the low level with an effective internal buffer of two. In this diagram, the initial state can be recognized by a thick outline, and marked states are filled.

To simplify notation in our exposition, we bring in the following event sets, natural projections, and languages.



$\mathbf{G}_{Ij}$
start
start
0
1
2
3
4
isDone
isDone
notDone
notDone
done
done
Σ_{Rj}={isDone,start}, Σ_{Aj}={done}, Σ_{LDj}={notDone}

Figure 3: Example LD Interface

$$\begin{aligned}
\Sigma_{I_j} &:= \Sigma_{R_j} \dot\cup \Sigma_{A_j} \dot\cup \Sigma_{LD_j}, & P_{I_j} &: \Sigma^* \to \Sigma^*_{I_j} \\
\Sigma_{IL_j} &:= \Sigma_{L_j} \cup \Sigma_{I_j}, & P_{IL_j} &: \Sigma^* \to \Sigma^*_{IL_j} \\
\Sigma_{IH} &:= \Sigma_H \ \cup \bigcup_{k\in\{1,\dots,n\}} \Sigma_{I_k}, & P_{IH} &: \Sigma^* \to \Sigma^*_{IH} \\
\mathcal{H} &:= P_{IH}^{-1}(L(\mathbf{G}_H)), & \mathcal{H}_m &:= P_{IH}^{-1}(L_m(\mathbf{G}_H)) \subseteq \Sigma^* \\
\mathcal{L}_j &:= P_{IL_j}^{-1}(L(\mathbf{G}_{L_j})), & \mathcal{L}_{m_j} &:= P_{IL_j}^{-1}(L_m(\mathbf{G}_{L_j})) \subseteq \Sigma^* \\
\mathcal{I}_j &:= P_{I_j}^{-1}(L(\mathbf{G}_{I_j})), & \mathcal{I}_{m_j} &:= P_{I_j}^{-1}(L_m(\mathbf{G}_{I_j})) \subseteq \Sigma^* \\
\mathcal{I} &:= \cap_{k\in\{1,\dots,n\}} \mathcal{I}_k, & \mathcal{I}_m &:= \cap_{k\in\{1,\dots,n\}} \mathcal{I}_{m_k} \\
\Sigma_{LD} &:= \bigcup_{k\in\{1,\dots,n\}} \Sigma_{LD_k}
\end{aligned}$$

We define our *flat system* to be $\mathbf{G} = \mathbf{G}_H || \mathbf{G}_{I_1} || \mathbf{G}_{L_1} || \ \dots || \mathbf{G}_{I_n} || \mathbf{G}_{L_n}$. By flat system we mean the equivalent DES if we ignored the interface structure.

We now present the properties that an HISC system must satisfy to ensure that it interacts with the interfaces correctly.

Definition 3.2. *The* n^{th} *degree* ($n \geq 1$) *interface system composed of DES* $\mathbf{G}_H, \mathbf{G}_{I_1}, \mathbf{G}_{L_1}, \dots, \mathbf{G}_{I_n}, \mathbf{G}_{L_n}$, *is* LD interface consistent *with respect to the alphabet partition given by (1), if for all* $j \in \{1,\dots,n\}$, *the following conditions are satisfied:*
Multi-level Properties

1. *The event set of* $\mathbf{G}_H$ *is* Σ_{IH}, *and the event set of* $\mathbf{G}_{L_j}$ *is* Σ_{IL_j}.

2. $\mathbf{G}_{I_j}$ *is a LD interface.*

High-Level Property

3. $(\forall s \in \mathcal{H} \cap \mathcal{I})\ Elig_{\mathcal{I}_j}(s) \ \cap \ (\Sigma_{A_j} \dot\cup \Sigma_{LD_j}) \subseteq Elig_{\mathcal{H}}(s)$

Low-Level Properties

4. $(\forall s \in \mathcal{L}_j \cap \mathcal{I}_j)\ Elig_{\mathcal{I}_j}(s) \ \cap \ \Sigma_{R_j} \subseteq Elig_{\mathcal{L}_j}(s)$

5. $(\forall s \in \Sigma^*.\Sigma_{R_j} \cap \mathcal{L}_j \cap \mathcal{I}_j)$
$Elig_{\mathcal{L}_j \cap \mathcal{I}_j}(s\Sigma^*_{L_j}) \cap \Sigma_{A_j} = Elig_{\mathcal{I}_j}(s) \cap \Sigma_{A_j}$ *where*

$$Elig_{\mathcal{L}_j \cap \mathcal{I}_j}(s\Sigma^*_{L_j}) := \bigcup_{l \in \Sigma^*_{L_j}} Elig_{\mathcal{L}_j \cap \mathcal{I}_j}(sl)$$

6. $(\forall s \in \mathcal{L}_j \cap \mathcal{I}_j)$
$s \in \mathcal{I}_{m_j} \Rightarrow (\exists l \in \Sigma^*_{L_j})\ sl \in \mathcal{L}_{m_j} \cap \mathcal{I}_{m_j}$.

We now provide an additional set of properties that the system must satisfy if the flat system $\mathbf{G}$ is to be nonblocking.

Definition 3.3. *The* n^{th} *degree* $(n \geq 1)$ *interface system composed of DES* $\mathbf{G}_H, \mathbf{G}_{I_1}, \mathbf{G}_{L_1}, \ldots, \mathbf{G}_{I_n}, \mathbf{G}_{L_n}$*, is said to be* LD level-wise nonblocking *if the following conditions are satisfied:*

(I) LD nonblocking at the high level*:*
$(\forall s \in \mathcal{H} \cap \mathcal{I})(\exists s' \in (\Sigma - \Sigma_{LD})^*)$
$ss' \in \mathcal{H}_m \cap \mathcal{I}_m$

(II) nonblocking at the low level*:*
$(\forall j \in \{1, \ldots, n\})\ \overline{\mathcal{L}_{m_j} \cap \mathcal{I}_{m_j}} = \mathcal{L}_j \cap \mathcal{I}_j$

The theorem bellow states that verifying the LD level-wise nonblocking and LD interface consistent conditions is sufficient to verify that our flat system is nonblocking.

Theorem 3.1 ([Led09])**.** *If the* n^{th} *degree* $(n \geq 1)$ *interface system composed of DES* $\mathbf{G}_H, \mathbf{G}_{I_1}, \mathbf{G}_{L_1}, \ldots, \mathbf{G}_{I_n}, \mathbf{G}_{L_n}$*, is LD level-wise nonblocking and LD interface consistent with respect to the alphabet partition given by (1), then*

$$L(G) = \overline{L_m(G)} \text{ where } G = G_H || G_{L_1} || G_{I_1} || \ldots || G_{L_n} || G_{I_n}.$$

Since checking that the LD level-wise nonblocking and LD interface consistent conditions only require a single component at a time, we note that we can evaluate each level independently. This means we do not need to construct the entire system model.

For controllability, we need to separate the subsystems into their plant and supervisor sub-components (see Figure 4). We define the *high-level plant* to be $\mathbf{G}^p_H$, and the *high-level supervisor* to be $\mathbf{S}_H$ (defined over event set Σ_{IH}). We define the j^{th} *low-level plant* and *supervisor* to be $\mathbf{G}^p_{L_j}$ and $\mathbf{S}_{L_j}$ (defined over Σ_{IL_j}) respectively. We next define the high-level subsystem to be $\mathbf{G}_H := \mathbf{G}^p_H || \mathbf{S}_H$, and define the j^{th} low-level subsystem to be $\mathbf{G}_{L_j} := \mathbf{G}^p_{L_j} || \mathbf{S}_{L_j}$. We note that in HISC systems, interfaces are always supervisors.

We can now define our *flat supervisor* and *plant* as well as some other languages as follows:
Plant $:= \mathbf{G}^p_H || \mathbf{G}^p_{L_1} || \ldots || \mathbf{G}^p_{L_n}$,
Sup $:= \mathbf{S}_H || \mathbf{S}_{L_1} || \ldots || \mathbf{S}_{L_n} || \mathbf{G}_{I_1} || \ldots || \mathbf{G}_{I_n}$,
$\mathcal{H}^p := P^{-1}_{IH} L(\mathbf{G}^p_H)$, $\mathcal{S}_H := P^{-1}_{IH} L(\mathbf{S}_H) \subseteq \Sigma^*$,
$\mathcal{L}^p_j := P^{-1}_{IL_j} L(\mathbf{G}^p_{L_j})$, $\mathcal{S}_{L_j} := P^{-1}_{IL_j} L(\mathbf{S}_{L_j}) \subseteq \Sigma^*$.
The controllability requirements that each level must satisfy are given in the following definition.

Definition 3.4. *The* n^{th} *degree* $(n \geq 1)$ *interface system composed of DES* $\mathbf{G}^p_H, \mathbf{S}_H, \mathbf{G}^p_{L_1}, \mathbf{S}_{L_1}, \mathbf{G}_{I_1}, \ldots, \mathbf{G}^p_{L_n}, \mathbf{S}_{L_n}, \mathbf{G}_{I_n}$*, is* LD level-wise controllable *with respect to the alphabet partition given by (1), if for all* $j \in \{1, \ldots, n\}$ *the following conditions hold:*

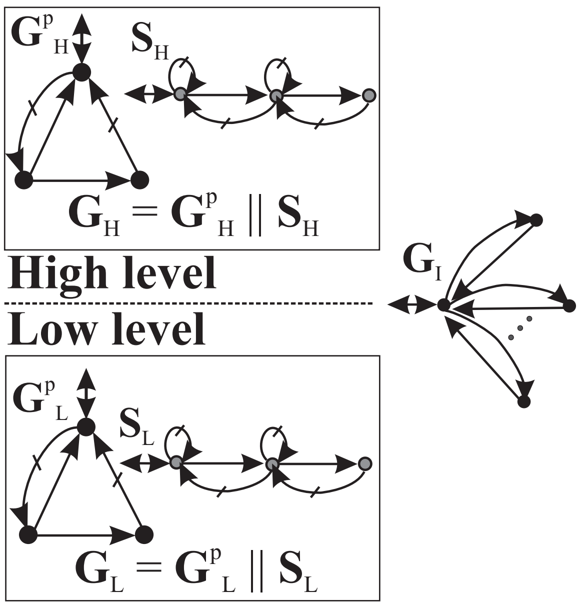


Figure 4: Plant and Supervisor Subplant Decomposition

(I) The alphabet of $\mathbf{G}^p_H$ *and* $\mathbf{S}_H$ *is* Σ_{IH}*, the alphabet of* $\mathbf{G}^p_{L_j}$ *and* $\mathbf{S}_{L_j}$ *is* Σ_{IL_j}*, and the alphabet of* $\mathbf{G}_{I_j}$ *is* Σ_{I_j}

(II) $(\forall s \in \mathcal{L}^p_j \cap \mathcal{S}_{L_j} \cap \mathcal{I}_j) Elig_{\mathcal{L}^p_j}(s) \cap \Sigma_u \subseteq Elig_{\mathcal{S}_{L_j} \cap \mathcal{I}_j}(s)$

(III) $(\forall s \in \mathcal{H}^p \cap \mathcal{I} \cap \mathcal{S}_H) Elig_{\mathcal{H}^p \cap \mathcal{I}}(s) \cap \Sigma_u \subseteq Elig_{\mathcal{S}_H}(s)$

The theorem below states that verifying LD level-wise controllable is sufficient to verify that the flat supervisor is controllable for the flat plant.

Theorem 3.2 ([Led09])**.** *If* n^{th} *degree* $(n \geq 1)$ *interface system composed of DES* $\mathbf{G}^p_H, \mathbf{S}_H, \mathbf{G}^p_{L_1}, \mathbf{S}_{L_1}, \mathbf{G}_{I_1}, \ldots, \mathbf{G}^p_{L_n}, \mathbf{S}_{L_n}, \mathbf{G}_{I_n}$ *is LD level-wise controllable with respect to the alphabet partition given by (1), then*
$(\forall s \in L(\mathbf{Plant}) \cap L(\mathbf{Sup})) Elig_{L(\mathbf{Plant})}(s) \cap \Sigma_u \subseteq Elig_{L(\mathbf{Sup})}(s)$

Since checking that the LD level-wise controllable condition only requires at most a single component at a time, we note that we can evaluate each level independently.

4 Hierarchical Interface-Based Decentralized Supervisory Control

In Section 3, we described a system composed of plant DES $\mathbf{G}^p_H$, $\mathbf{G}^p_{L_1}, \ldots, \mathbf{G}^p_{L_n}$, supervisor DES $\mathbf{S}_H$, $\mathbf{S}_{L_1}, \ldots, \mathbf{S}_{L_n}$, and interface DES $\mathbf{G}_{I_1}, \ldots, \mathbf{G}_{I_n}$. Although the level-wise controllability condition [Led02, Led09] does effectively limit the high-level supervisor to events in Σ_{IH}, and the j^{th} low-level supervisor to events in Σ_{IL_j}, it requires the HISC structure and does not allow further restrictions outside of this structure. In order to allow decentralized supervisors within components, we need to extend the HISC structure.

We now introduce the *Hierarchical Interface-based Decentralized Supervisory Control (HIDSC)* architecture. HIDSC is an extension of HISC from centralized control to a decentralized architecture by allowing decentralized supervisors within a subsystem, but without additional HISC restrictions.

In the HIDSC framework, all the HISC supervisors are replaced by corresponding specification DES. In decentralized control, these specification DES represent the control behaviours we wish to implement as decentralized controllers, not specifications for synthesizing a centralized maximally permissive supervisor.

For HIDSC, we will replace supervisor $\mathbf{S}_H$ by specification DES $\mathbf{F}_H$ (defined over Σ_{IH}), and we will replace supervisor $\mathbf{S}_{L_j}$ by specification DES $\mathbf{F}_{L_j}$ (defined over Σ_{IL_j}). Typically, $\mathbf{F}_H$ will express system-wide constraints about how the components interact and what tasks the low levels should perform. $\mathbf{F}_{L_j}$ expresses how the j^{th} low level will perform the tasks (requests) given to it by the high level. For each component, there is a different index set of decentralized controllers.

We are now ready to define the structure of an HIDSC system.

Definition 4.1. *The* n^{th} degree decentralized specification interface system *with respect to the alphabet partition given by (1) is composed of plant DES* $\mathbf{G}_H^p$, $\mathbf{G}_{L_1}^p, \ldots, \mathbf{G}_{L_n}^p$*, specification DES* $\mathbf{F}_H$, $\mathbf{F}_{L_1}, \ldots, \mathbf{F}_{L_n}$*, interface DES* $\mathbf{G}_{I_1}, \ldots, \mathbf{G}_{I_n}$*, and high-level and low-level decentralized controllers. The system has the following structure.*

High level:

- *The high-level decentralized controllers have an index set* $D_H := \{N_{H,1}, \ldots, N_{H,n_0}\}$.
- *The event set for* $\mathbf{G}_H^p$, $\mathbf{F}_H$ *and the corresponding decentralized controllers is* Σ_{IH}.
- *For* $i \in D_H$, $\Sigma_{H,c,i} \subseteq \Sigma_c \cap \Sigma_{IH}$ *and* $\Sigma_{H,o,i} \subseteq \Sigma_o \cap \Sigma_{IH}$ *are the corresponding controllable and observable event subsets for the high-level decentralized controllers.*

Low level:

- *For* $j \in \{1, \ldots, n\}$*, the* j^{th} *low-level component has an index set* $D_{L_j} := \{N_{L_j,1}, \ldots, N_{L_j,n_j}\}$ *for its own decentralized controllers.*
- *The event set of each low-level component* $\mathbf{G}_{L_j}^p$, $\mathbf{F}_{L_j}$ *and the corresponding decentralized controllers is* Σ_{IL_j}.
- *For* $i \in D_{L_j}$, $\Sigma_{L_j,c,i} \subseteq \Sigma_c \cap \Sigma_{IL_j}$ *and* $\Sigma_{L_j,o,i} \subseteq \Sigma_o \cap \Sigma_{IL_j}$ *are the corresponding controllable and observable event subsets for the low-level decentralized controllers.*

Multi-level:

- *The index set for all decentralized controllers in the system is* $D := D_H \dot{\cup} \dot{\bigcup}_{j=1}^{n} D_{L_j} = \{1, ..., N\}$.
- $\cup_{i=1}^{N} \Sigma_{c,i} = \Sigma_c$ *and* $\cup_{i=1}^{N} \Sigma_{o,i} = \Sigma_o$

For the rest of this section, we will refer to such a system as an n^{th} degree decentralized specification interface system Ψ, or simply Ψ. Note that in Ψ, we do not specify the index of decentralized controllers by $\{1, ..., n_0\}$, $\{1, ..., n_j\}$, etc., because once combined they would overlap. We create the system index set using disjoint union.

The flat system $\mathbf{G}$ is the synchronization of all the plant, specification, and interface components in the system, i.e., $\mathbf{G} = \mathbf{G}_H^p \,||\, \mathbf{G}_{L_1}^p \,||\, \ldots \,||\, \mathbf{G}_{L_n}^p \,||\, \mathbf{F}_H \,||\, \mathbf{F}_{L_1} \,||\, \ldots \,||\, \mathbf{F}_{L_n} \,||\, \mathbf{G}_{I_1} \,||\, \ldots \,||\, \mathbf{G}_{I_n}$. We use the term flat system to mean the overall system ignoring the HIDSC structure.

It is important to note that for an HIDSC system, we would first design level-wise supervisors for the original HISC system while ignoring any decentralized restrictions. We would then use the HISC structure to verify that the system is nonblocking and controllable. We would next use these level-wise supervisors (which include the system's interface DES) as "specifications" for the design of the per-component decentralized supervisors specified by the HIDSC system. The final system would not contain any of these specification DES, just the resulting decentralized controllers that would provide us with equivalent closed-loop behaviour (see Corollary 4.1 in Section 4.2).

4.1 HIDSC Co-observability Definition and Theorem

The main focus of this section is to verify co-observability in an HIDSC system Ψ without explicitly constructing the flat system. We will only perform a per-component co-observability verification, but guarantee that the whole system is co-observable.

To aid in defining our per-component co-observability definition and HIDSC co-observability theorem, we specify some decentralized notations for Ψ.

We use $D_{H,c}(\sigma) := \{i \in D_H \,|\sigma \in \Sigma_{H,c,i}\}$ to denote the set of decentralized controllers in the high level that can control the event σ. We use $D_{H,o}(\sigma) := \{i \in D_H \,|\sigma \in \Sigma_{H,o,i}\}$ to denote the set of decentralized controllers in the high level that can observe the event σ. Correspondingly, $P_{H,i} : \Sigma^* \to \Sigma_{H,o,i}^*$ is the natural projection describing the partial view of controller $i \in D_H$.

For $j \in \{1, \ldots, n\}$, $D_{L_j,c}(\sigma) := \left\{i \in D_{L_j} \,|\sigma \in \Sigma_{L_j,c,i}\right\}$ is the set of decentralized controllers in the j^{th} low-level component that can control the event σ. We use $D_{L_j,o}(\sigma) := \left\{i \in D_{L_j} \,|\sigma \in \Sigma_{L_j,o,i}\right\}$ to represent the set of decentralized controllers in the j^{th} low-level component that can observe the event σ. Correspondingly, $P_{L_j,i} : \Sigma^* \to \Sigma_{L_j,o,i}^*$ is the natural projection describing the partial view of controller $i \in D_{L_j}$.

We use $D_c(\sigma) := \{i \in D \,|\sigma \in \Sigma_{c,i}\}$ to denote the set of decentralized controllers in the system that can control the event σ.

Further, we introduce a few languages used for the HIDSC co-observability definition and theorem.

$$\begin{aligned} \mathcal{F}_H &:= P_{IH}^{-1}(L(\mathbf{F}_H)), \qquad \mathcal{F}_{L_j} := P_{IL_j}^{-1}(L(\mathbf{F}_{L_j})) \\ \mathcal{F} &:= \mathcal{F}_H \cap \mathcal{F}_{L_1} \cap \ldots \cap \mathcal{F}_{L_n}, \, \mathcal{P} := \mathcal{H}^p \cap \mathcal{L}_1^p \cap \ldots \cap \mathcal{L}_n^p \end{aligned}$$

Language $\mathcal{F}_H$ represents the behaviour of the specification automata in the high-level subsystem, while $\mathcal{F}_{L_j}$ represents the behaviour of the specification automata for the j^{th} low-level subsystem. Language $\mathcal{F}$ represents the global specification for the flat system, and $\mathcal{P}$ represents the behaviour of the flat plant.

We now present the per-component level-wise co-observability definition for HIDSC system Ψ. We note that each individual condition needs at most a single subsystem for its verification, thus we do not need to construct the entire system model. This can save significant computation and can help to alleviate the state-space explosion problem.

Definition 4.2. *Let Ψ be an HIDSC n^{th} degree decentralized specification interface system. Then Ψ is* level-wise co-observable *if for all $j \in \{1, \ldots, n\}$ the following conditions hold:*
I) $(\forall t \in \mathcal{F}_H \cap \mathcal{H}^p \cap \mathcal{I})(\forall \sigma \in \Sigma_c)\ t\sigma \in (\mathcal{H}^p \cap \mathcal{I}) \backslash \mathcal{F}_H \Rightarrow$
$(\exists i \in D_{H,c}(\sigma))\ P_{H,i}^{-1}[P_{H,i}(t)]\sigma \cap \mathcal{F}_H \cap \mathcal{H}^p \cap \mathcal{I} = \emptyset,$
II) $(\forall t \in \mathcal{F}_{L_j} \cap \mathcal{I}_j \cap \mathcal{L}_j^p)(\forall \sigma \in \Sigma_c)\ t\sigma \in \mathcal{L}_j^p \backslash (\mathcal{F}_{L_j} \cap \mathcal{I}_j) \Rightarrow$
$(\exists i \in D_{L_j,c}(\sigma)) P_{L_j,i}^{-1}[P_{L_j,i}(t)]\sigma \cap \mathcal{F}_{L_j} \cap \mathcal{I}_j \cap \mathcal{L}_j^p = \emptyset.$

Definition 4.2 states that HIDSC system Ψ is *level-wise co-observable* if the high-level component is co-observable and each low-level component is co-observable.

We note that the interfaces are treated as specifications at the low level and treated as plants at the high level. This is done this way because interfaces represent the behaviour provided by its low level and the information needed to verify that it is co-observable is typically present at the low level but not the high level. To avoid having to repeat this information at the high level, we use the results of [LLMR14] that allow us to treat supervisors as if they are plants once we verify they are co-observable. By treating interfaces as plants at the high level, we allow the high-level supervisor to be more permissive in general as there will typically be fewer strings that can cause the co-observability verification to fail.

We now restate the co-observability definition in terms of our HIDSC system. We note that from their definition, we know that languages $\mathcal{F}$, $\mathcal{I}$, and $\mathcal{P}$ are prefix-closed. We also note that according to Definition 4.2, each $i \in D$ represents some $i_1 \in D_H$ or some $i_2 \in D_{L_j}$, $j \in \{1, ..., n\}$.

Definition 4.3. *Let Ψ be an HIDSC n^{th} degree decentralized specification interface system. Let $D = \{1, ..., N\} = D_H \dot{\cup} \dot{\bigcup}_{j=1}^{n} D_{L_j}$ be the index set for Ψ. Let $\Sigma_{c,i} \subseteq \Sigma$ and $\Sigma_{o,i} \subseteq \Sigma$ be sets of controllable and observable events, respectively, for $i \in D$, where $D_c(\sigma) = \{i \in D \mid \sigma \in \Sigma_{c,i}\}$. Let $P_i : \Sigma^* \rightarrow \Sigma_{o,i}^*$, $i \in D$, be natural projections. Then Ψ is* globally co-observable *if*
$(\forall t \in \mathcal{F} \cap \mathcal{I} \cap \mathcal{P})\ (\forall \sigma \in \Sigma_c)\ t\sigma \in \mathcal{P} \backslash (\mathcal{F} \cap \mathcal{I}) \Rightarrow$
$(\exists i \in D_c(\sigma)) P_i^{-1}[P_i(t)]\sigma \cap \mathcal{F} \cap \mathcal{I} \cap \mathcal{P} = \emptyset.$

We note that Definition 4.3 is the property we want to verify but we will do so by using our per-component co-observability definition.

The theorem below states that the level-wise co-observability property is sufficient to guarantee that the flat system is co-observable. This means that co-observability for the system can be verified while only constructing a single component at a time.

Theorem 4.1. *Let Ψ be an HIDSC n^{th} degree decentralized specification interface system. If Ψ is level-wise co-observable then Ψ is globally co-observable.*

Proof. See Appendix. □

We now examine the complexity of our approach. In monolithic verification, the n low-level subsystems are composed directly with the high-level system without using the interface structure. The size of the state space for the monolithic method is the size of the product state space of $\mathbf{G}_H \mid\mid \mathbf{G}_{L_1} \mid\mid \ldots \mid\mid \mathbf{G}_{L_n}$. If the size of the state space of $\mathbf{G}_H$ is bounded by N_H, and the size of the state space for each $\mathbf{G}_{L_j}$ is bounded by N_L, then the size of the state space of the monolithic method is bounded by $N_H N_L^n$.

Our method verifies each component separately, therefore the size of the state space is bounded by the size of the component combined with its interface. For $j = 1, ..., n$, if we assume that the size of the state space of $\mathbf{G}_{I_j}$ is bounded by N_I, then each low-level subsystem $\mathbf{G}_{L_j}||\mathbf{G}_{I_j}$ is bounded by $N_L N_I$. The high-level subsystem $\mathbf{G}_H||\mathbf{G}_{I_1}||\ldots||\mathbf{G}_{I_n}$ is bounded by $N_H N_I^n$. Therefore, our method is bounded by the larger of $N_H N_I^n$ and $N_L N_I$. Typically in an HIDSC design, the size of the high level is the limiting factor. This means that as long as $N_I \ll N_L$, we should achieve significant computational savings.

4.2 MNDSC Supervisor Existence Theorem

We now present the marking nonblocking decentralized supervisory control (MNDSC) existence theorem, which shows that there exists an MNDSC to achieve the specification if and only if K is controllable and co-observable.

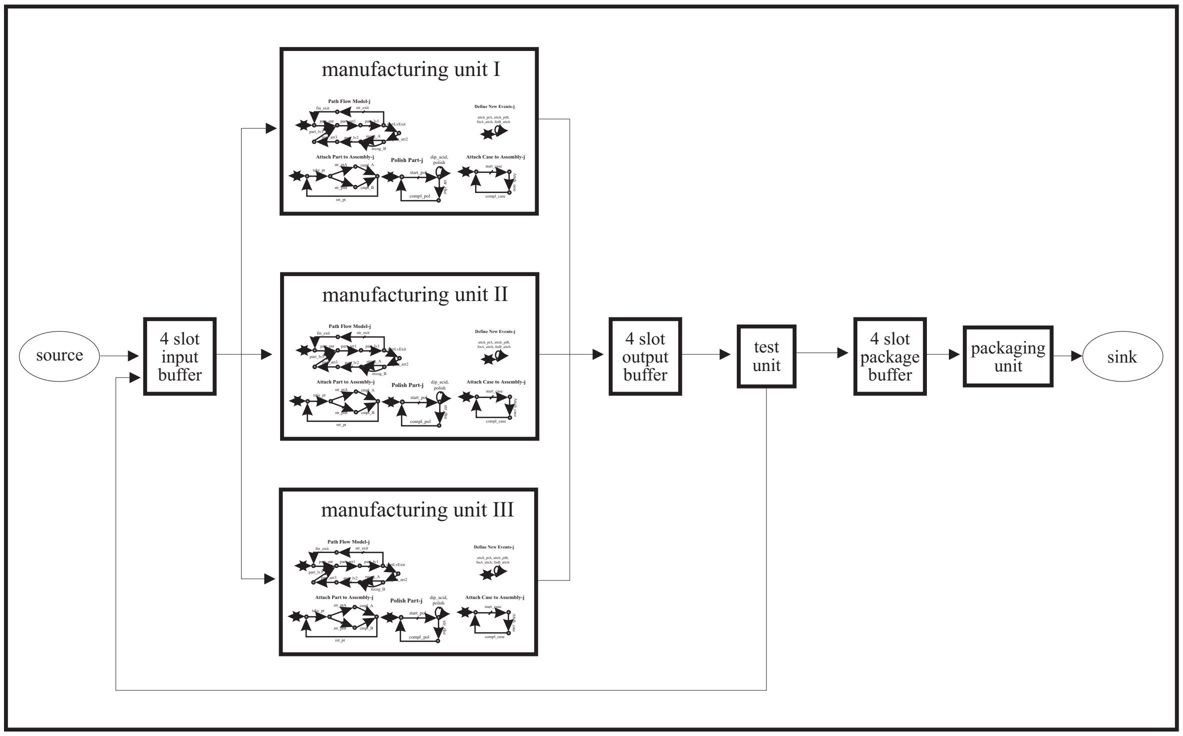


Figure 5: Block Diagram of Parallel Plant System.

Note that in Theorem 4.2 below we do not require that K be $L_m(\mathbf{G})$-closed which is assumed by traditional decentralized control [CL08]. This will allow us to apply the result to our HIDSC system as we have an HISC nonblocking result but not an HISC $L_m(\mathbf{G})$-closed result.

Theorem 4.2. *Let* $\mathbf{Plant} := (Q,\ \Sigma,\ \delta,\ q_0,\ Q_m)$*,* $K \subseteq L_m(\mathbf{Plant})$*, and* $K \neq \emptyset$*. There exists an MNDSC* $\mathcal{S}_{Con}$ *for* $(K, \mathbf{Plant})$ *such that* $L_m(\mathcal{S}_{Con}/\mathbf{Plant}) = K$ *if and only if* K *is controllable and co-observable with respect to* $L(\mathbf{Plant})$*.*

Proof. See Appendix. □

We will now relate Theorem 4.2 to our HIDSC system and nonblocking. In essence, we are requiring Ψ to have equivalent MNDSC behaviour with $\mathcal{S}_{Con}/\mathbf{Plant}$, which ensures our HIDSC system implementation will be nonblocking.

Corollary 4.1. *Let Ψ be an HIDSC n^{th} degree decentralized specification interface system. Let $\mathbf{Plant} := \mathbf{G}^p_H||\mathbf{G}^p_{L_1}|| \ldots ||\mathbf{G}^p_{L_n}$, and $\mathbf{Spec} := \mathbf{F}_H||\mathbf{F}_{L_1}||\ldots||\mathbf{F}_{L_n}||\mathbf{G}_{I_1}||\ldots||\mathbf{G}_{I_n}$. Let $L_m(\mathbf{Spec}) \cap L_m(\mathbf{Plant}) \neq \emptyset$. There exists an MNDSC $\mathcal{S}_{Con}$ for ($L_m(\mathbf{Spec}) \cap L_m(\mathbf{Plant})$, $\mathbf{Plant}$) such that $L_m(\mathcal{S}_{Con}/\ \mathbf{Plant}) = L_m(\mathbf{Spec}) \cap L_m(\mathbf{Plant})$, and $L(\mathcal{S}_{Con}/\mathbf{Plant}) = L(\mathbf{Spec}) \cap L(\mathbf{Plant})$, if and only if $L_m(\mathbf{Spec}) \cap L_m(\mathbf{Plant})$ is controllable and co-observable with respect to $L(\mathbf{Plant})$, and $\overline{L_m(\mathbf{Spec}) \cap L_m(\mathbf{Plant})} = L(\mathbf{Spec}) \cap L(\mathbf{Plant})$.*

Proof. See Appendix. □

For HIDSC system Ψ, Corollary 4.1 tells us that the marked behaviour of our MNDSC and flat plant is equal to $L_m(\mathbf{Spec}) \cap L_m(\mathbf{Plant})$ and their closed behaviour is equal to $L(\mathbf{Spec}) \cap L(\mathbf{Plant})$. To apply Corollary 4.1, we need to first show that Ψ is co-observable, nonblocking, and controllable. For scalability, we want to verify all these global properties using only per-component properties.

Theorem 4.1 states that level-wise co-observability gives us global co-observability. Theorems 3.1 and 3.2 state that the HISC LD level-wise nonblocking, LD interface consistent, and LD level-wise controllability properties together imply that our flat system is nonblocking and controllable. We can thus verify all needed global properties using per-component checks. As we never need to construct the full system model, this offers potentially great computational savings.

5 Manufacturing Example

To demonstrate the HIDSC method, we adapt a small manufacturing system from [Led02] that was originally modeled as an HISC system. The system, shown in Figure 5, is composed of three manufacturing units running in parallel, a testing unit, material feedback, a packaging unit, and three buffers to insure the proper flow of material.

Figure 6 shows which DES belong to the high-level subsystem ($\mathbf{G}_H$), the high-level plant ($\mathcal{G}_H$), the high-level specification automata ($\mathcal{S}_H$), the j^{th} low-level subsystem ($\mathbf{G}_{L_j}$), the j^{th} low-level plant ($\mathcal{G}_{L_j}$), the j^{th} low-level specification automata ($\mathcal{S}_{L_j}$), and the j^{th} interface DES ($\mathbf{G}_{I_j}$), $j = \mathrm{I}, \mathrm{II}, \mathrm{III}$. We note that the three low-level subsystems shown in Figure 6 are identical up to relabeling. Figure 7 shows the low-level subsystems in more detail.

In the diagrams, controllable events are those with a slash on the transition arrow, marked states are states with an unlabeled incoming arrow, and initial states are states with an unlabeled outgoing arrow.

5.1 Manufacturing System as an HIDSC System

Originally this example was modeled as an HISC system. We will now adapt it as an HIDSC system. Typically, we would only do this if the system had an inherent distributed nature forcing us to implement supervisors with partial observations and partial controllability beyond the compartmentalized limitations imposed by the HISC structure.

We define the alphabet partition $\Sigma := \dot\cup_{j\in\{\mathrm{I},\mathrm{II},\mathrm{III}\}}(\Sigma_{L_j} \dot\cup\ \Sigma_{R_j} \dot\cup \Sigma_{A_j})\ \dot\cup\ \Sigma_H$ below:

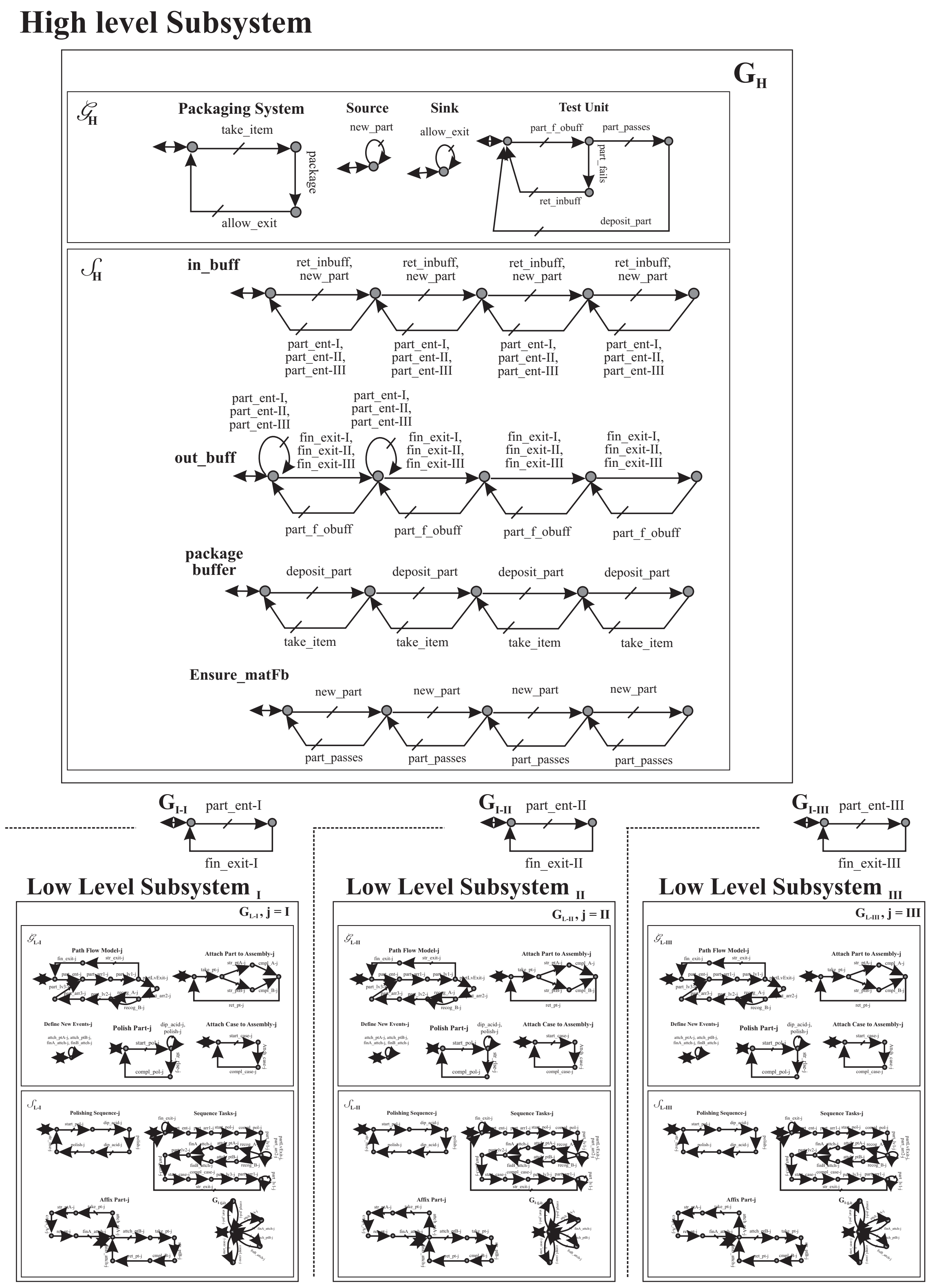


Figure 6: Complete Parallel System.

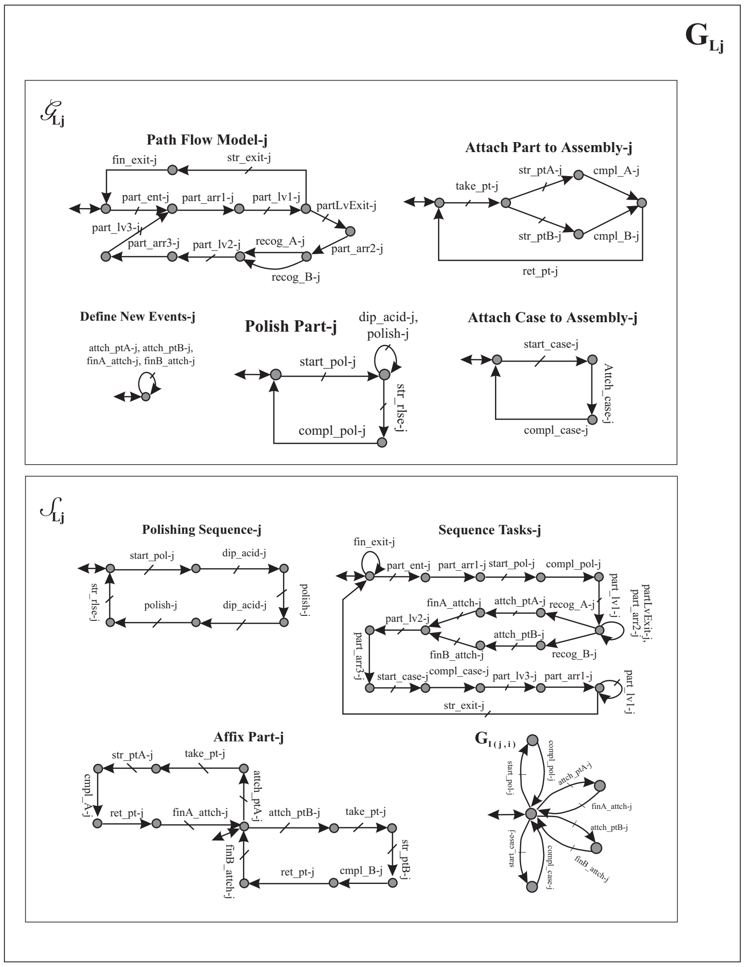


Figure 7: Low-Level Subsystem j.

$$
\begin{aligned}
\Sigma_H &= \{take\_item, package, allow\_exit, new\_part, part\_fails,\\
&\quad part\_f\_obuff, part\_passes, ret\_inbuff, deposit\_part\}\\
\Sigma_{R_j} &= \{part\_ent\text{-}j\}\\
\Sigma_{A_j} &= \{fin\_exit\text{-}j\}\\
\Sigma_{L_j} &= \{start\_pol\text{-}j, attch\_ptA\text{-}j, attch\_ptB\text{-}j, start\_case\text{-}j\\
&\quad comp\_pol\text{-}j, finA\_attch\text{-}j, finB\_attch\text{-}j, compl\_case\text{-}j,\\
&\quad part\_arr1\text{-}j, part\_lv1\text{-}j, partLvExit\text{-}j, str\_exit\text{-}j,\\
&\quad part\_arr2\text{-}j, recog\_A\text{-}j, recog\_B\text{-}j, part\_lv2\text{-}j,\\
&\quad part\_arr3\text{-}j, part\_lv3\text{-}j, take\_pt\text{-}j, str\_ptA\text{-}j, str\_ptB\text{-}j,\\
&\quad compl\_A\text{-}j, compl\_B\text{-}j, ret\_pt\text{-}j, dip\_acid\text{-}j, polish\text{-}j,\\
&\quad str\_rlse\text{-}j, Attch\_case\text{-}j\}
\end{aligned}
$$

Our first step is to replace the existing supervisors with specification automata; thus let

$\mathbf{F}_H = \mathcal{S}_H$ and $\mathbf{F}_{L_j} = \mathcal{S}_{L_j}$, $j = \mathrm{I}, \mathrm{II}, \mathrm{III}$.

We next design decentralized controllers ($H1$, $H2$, L_{I_1}, L_{I_2}, L_{II_1}, L_{II_2}, L_{III_1}, L_{III_2}) to define our HIDSC problem.

For the high-level subsystem, the observable and controllable alphabet for controller $H1$ is specified as:

$$\begin{aligned}\Sigma_{H,o,1} &= \Sigma_I \cup \{new\_part, ret\_inbuff, part\_f\_obuff\} \\ \Sigma_{H,c,1} &= (\Sigma_I \cap \Sigma_c) \cup \{part\_f\_obuff\}\end{aligned}$$

The observable and controllable alphabet for controller $H2$ is specified as:

$$\begin{aligned}\Sigma_{H,o,2} &= (\Sigma_I \cap \Sigma_{uc}) \cup \{take\_item, package, allow\_exit, \\ &\quad new\_part, part\_passes, part\_fails, ret\_inbuff, \\ &\quad deposit\_part\} \\ \Sigma_{H,c,2} &= \{take\_item, allow\_exit, new\_part, part\_f\_obuff, \\ &\quad part\_passes, ret\_inbuff, deposit\_part\}\end{aligned}$$

For the j^{th} low-level subsystem ($j = \mathrm{I}, \mathrm{II}, \mathrm{III}$), the observable alphabet for controllers L_{j_1} and L_{j_2} is specified as:

$$\begin{aligned}\Sigma_{L,o,j_1} &= \{part\_ent\text{-}j, fin\_exit\text{-}j, start\_pol\text{-}j, part\_arr1\text{-}j\} \\ \Sigma_{L,o,j_2} &= \Sigma_{L_j}\end{aligned}$$

The controllable alphabet for controllers L_{j_1} and L_{j_2} is specified as:

$$\begin{aligned}\Sigma_{L,c,j_1} &= \{part\_ent\text{-}j, start\_pol\text{-}j\} \\ \Sigma_{L,c,j_2} &= (\Sigma_{L_j} \cap \Sigma_c) \setminus \{start\_pol\text{-}j\}\end{aligned}$$

The index sets of decentralized controllers for each component are: D_H=$\{H1, H2\}$, D_{L_I}=$\{L_{I_1}, L_{I_2}\}$, $D_{L_{II}}$=$\{L_{II_1}, L_{II_2}\}$, and $D_{L_{III}}$=$\{L_{III_1}, L_{III_2}\}$.

We now define the *flat plant,* and the *flat specification automata* as follows:

$$\begin{aligned}\mathbf{Plant} &:= \mathcal{G}_H || \mathcal{G}_{L_\mathrm{I}} || \mathcal{G}_{L_\mathrm{II}} || \mathcal{G}_{L_\mathrm{III}} \\ \mathbf{Spec} &:= \mathbf{F}_H || \mathbf{F}_{L_I} || \mathbf{F}_{L_{II}} || \mathbf{F}_{L_{III}} || \mathbf{G}_{I_\mathrm{I}} || \mathbf{G}_{I_\mathrm{II}} || \mathbf{G}_{I_\mathrm{III}}\end{aligned}$$

5.2 Co-observability Verification for System

We now need to verify whether $L_m(\mathbf{Spec})$ is co-observable w.r.t. $L(\mathbf{Plant})$. We can then conclude, in combination with checking controllability and nonblocking, by Corollary 4.1 that there exists an MNDSC and that its resulting closed-loop behaviour is the same as that of the flat system of our HIDSC system. By Theorem 4.1, we know that to check co-observability of the HIDSC system, it is sufficient to verify level-wise co-observability.

The following steps for level-wise co-observability verification are:

Step 1. Verify whether the first low-level subsystem satisfies its portion of the level-wise co-observable definition, i.e., whether $L(\mathbf{F}_{L_\mathrm{I}} || \mathbf{G}_{I_\mathrm{I}})$ is co-observable w.r.t. $L(\mathbf{G}_{L_\mathrm{I}})$, $\Sigma_{L,c,i}$, $\Sigma_{L,o,i}$ for $i \in D_{L_I}$.

Step 2. Step 1 is sufficient to verify all three low levels as they are identical up to relabeling.

Step 3. Verify whether the high-level subsystem satisfies its portion of the level-wise co-observable definition, i.e., verifying whether $L(\mathbf{F}_H)$ is co-observable w.r.t. $L(\mathcal{G}_H||\mathbf{G}_{I_{\mathrm{I}}}||\mathbf{G}_{I_{\mathrm{II}}}||\mathbf{G}_{I_{\mathrm{III}}})$, $\Sigma_{H,c,i}$, $\Sigma_{H,o,i}$, for $i \in D_H$.

Using our software research tool, we verified that the first low-level component satisfies its portion of the level-wise co-observable definition. The monolithic verification ran for 5 hours without completing, so we stopped it. The run time of our incremental verification algorithm [Liu15, LLMR14] was 4.76 seconds. The low-level model contained 550 states.

We next verified that the high-level component satisfies its portion of the level-wise co-observable definition. The run time of our incremental verification algorithm was 424.78 seconds. The high-level model contained 3,120 states.

After completing steps 1-3, we conclude that the decentralized system is level-wise co-observable, thus globally co-observable by Theorem 4.1. The total verification run time was 429.54 seconds for a system whose complete system model has 2.78×10^{10} states.

We applied our incremental verification algorithm to the entire system model (i.e., to the flat system), but our software failed to complete after 5 hours.

Using our software tool DESpot [DES14], we verified that the system is LD level-wise controllable, LD level-wise nonblocking, and LD interface consistent. We can thus conclude by Theorem 3.1 and Theorem 3.2 that our flat system is nonblocking and controllable. We conclude by Corollary 4.1 that there exists a marking nonblocking decentralized supervisory control $\mathcal{S}_{Con}$ for **Plant**, and that **Spec**$||$**Plant** has equivalent MNDSC behaviour with $\mathcal{S}_{Con}/$**Plant**. This means that since **Spec**$||$**Plant** is nonblocking, $\mathcal{S}_{Con}/$**Plant** is also nonblocking.

5.3 Complexity Analysis for the Decentralized System

Applying DESpot to the small manufacturing system example, we found that the state size of the entire system was 2.78×10^{10}. However, the high-level state size was 3120 and the low-level state size was 550. As an HIDSC check only requires constructing a single component at a time, this is a potential savings of about seven orders of magnitude.

The computational complexity to verify co-observability using the monolithic approach in [RW95] is $O(|\Sigma||Y|^{2(N+2)})$, where Σ is the event set, Y is the state space, and N is the number of decentralized controllers. Substituting in for the small manufacturing system example, we found that verifying co-observability using the above method gives a computation bounded by $|42||2.78 \times 10^{10}|^{2(8+2)} = 3.19 \times 10^{210}$. Using our method, the computation is bounded by $|15||3120|^{2(2+2)} = 1.35 \times 10^{29}$. The potential computational saving is a 180 order of magnitude reduction.

6 Conclusions and Future Work

In decentralized control, agents have only a partial view and partial control of the system and must cooperate to achieve the control objective. In order to synthesize a decentralized control solution, a specification must satisfy the co-observability property. Existing co-observability verification methods require the possibly intractable construction of the complete system.

To address this issue, we adapted the existing HISC approach to support decentralized control. We introduced the HIDSC framework that included a per-component definition of co-observability. This allows co-observability to be evaluated using only a single component at a time. As a result, the entire system model never needs to be constructed which can provide significant savings. Finally, we provided and proved the necessary and sufficient conditions for supervisory control existence in the HIDSC framework.

We applied our approach to a small manufacturing example. It contained a high level with 3120 states, three low levels with 550 states each, and a flat model with 2.78×10^{10} states. We verified the per-component co-observability property in 429.54 seconds. We tried to verify the HIDSC system as a flat model but our software failed to complete after 5 hours.

For future work, we suggest extending HIDSC from the current two level approach to a multi-level method to allow HIDSC handle even larger systems. We also suggest introducing communication to allow certain events to be observable [BL00, RC11, WVS96] when a given component fails to be co-observable.

7 Proofs

Theorem 4.1:

Proof. Assume Ψ is level-wise co-observable. We will now show that Ψ is globally co-observable.

Sufficient to show that:

$(\forall t \in \mathcal{F} \cap \mathcal{I} \cap \mathcal{P})\ (\forall \sigma \in \Sigma_c)\ t\sigma \in \mathcal{P} \backslash (\mathcal{F} \cap \mathcal{I}) \Rightarrow (\exists i \in D_c(\sigma)) P_i^{-1}[P_i(t)]\sigma \cap \mathcal{F} \cap \mathcal{I} \cap \mathcal{P} = \emptyset$

Let $t \in \mathcal{F} \cap \mathcal{I} \cap \mathcal{P}$ and $\sigma \in \Sigma_c$. Assume $t\sigma \in \mathcal{P} \backslash (\mathcal{F} \cap \mathcal{I})$.

As $\mathcal{F} = \mathcal{F}_H \cap \mathcal{F}_{L_1} \cap \ldots \cap \mathcal{F}_{L_n}$, $\mathcal{P} = \mathcal{H}^p \cap \mathcal{L}_1^p \cap \ldots \cap \mathcal{L}_n^p$, and $\mathcal{I} = \mathcal{I}_1 \cap \ldots \cap \mathcal{I}_n$, we can conclude that: $t \in \mathcal{F}_H \cap \mathcal{H}^p \cap \mathcal{I}$, and $(\forall j \in \{1, \ldots, n\})\ t \in \mathcal{F}_{L_j} \cap \mathcal{I}_j \cap \mathcal{L}_j^p$ (**1**)

As $t\sigma \in \mathcal{P} \backslash (\mathcal{F} \cap \mathcal{I})$, we have: $t\sigma \in \mathcal{P}$ and $t\sigma \notin \mathcal{F} \cap \mathcal{I}$.

$\Rightarrow t\sigma \notin \mathcal{F}_H \cap \mathcal{F}_{L_1} \cap \ldots \cap \mathcal{F}_{L_n} \cap \mathcal{I}_1 \cap \ldots \cap \mathcal{I}_n$, by definition of $\mathcal{F}$ and $\mathcal{I}$

$\Rightarrow t\sigma \in \mathcal{P}$ and $t\sigma \notin \mathcal{F}_H$, or $t\sigma \in \mathcal{P}$ and $(\exists j \in \{1, \ldots, n\})\ t\sigma \notin \mathcal{F}_{L_j} \cap \mathcal{I}_j$

Case 1) $(\exists j \in \{1, \ldots, n\})\ t\sigma \notin \mathcal{F}_{L_j} \cap \mathcal{I}_j$

Let $j \in \{1, \ldots, n\}$ such that $t\sigma \notin \mathcal{F}_{L_j} \cap \mathcal{I}_j$. We also have $t \in \mathcal{F}_{L_j} \cap \mathcal{I}_j \cap \mathcal{L}_j^p$, by (**1**).

$\Rightarrow t \in \mathcal{F}_{L_j} \cap \mathcal{I}_j \cap \mathcal{L}_j^p$, $t\sigma \in \mathcal{L}_j^p$ and $t\sigma \notin \mathcal{F}_{L_j} \cap \mathcal{I}_j$ as $t\sigma \in \mathcal{P}$

As Ψ is level-wise co-observable, we have: $(\exists i \in D_{L_j,c}(\sigma))\ P_{L_j,i}^{-1}[P_{L_j,i}(t)]\sigma \cap \mathcal{F}_{L_j} \cap \mathcal{I}_j \cap \mathcal{L}_j^p = \emptyset$

$\Rightarrow (\exists i \in D_c(\sigma))\ P_i^{-1}[P_i(t)]\sigma \cap \mathcal{F}_{L_j} \cap \mathcal{I}_j \cap \mathcal{L}_j^p = \emptyset$, as $D_{L_j,c}(\sigma) \subseteq D_c(\sigma)$ and thus $P_{L_j,i} = P_i$

$\Rightarrow (\exists i \in D_c(\sigma))\ P_i^{-1}[P_i(t)]\sigma \cap \mathcal{F} \cap \mathcal{I} \cap \mathcal{P} = \emptyset$, as $\mathcal{F} \cap \mathcal{I} \cap \mathcal{P} \subseteq \mathcal{F}_{L_j} \cap \mathcal{I}_j \cap \mathcal{L}_j^p$

Case 2) $(\forall j \in \{1, \ldots, n\})\ t\sigma \in \mathcal{F}_{L_j} \cap \mathcal{I}_j$

From earlier we have: $t\sigma \in \mathcal{P}$ and $t\sigma \notin \mathcal{F}_H \cap \mathcal{F}_{L_1} \cap \ldots \cap \mathcal{F}_{L_n} \cap \mathcal{I}_1 \cap \ldots \cap \mathcal{I}_n$

As $(\forall j \in \{1, \ldots, n\})\ t\sigma \in \mathcal{F}_{L_j} \cap \mathcal{I}_j$, we have $t\sigma \in \mathcal{F}_{L_1} \cap \ldots \cap \mathcal{F}_{L_n} \cap \mathcal{I}_1 \cap \ldots \cap \mathcal{I}_n$.

$\Rightarrow t\sigma \notin \mathcal{F}_H$ and $t\sigma \in \mathcal{I}$

$\Rightarrow t\sigma \notin \mathcal{F}_H$ and $t\sigma \in \mathcal{H}^p \cap \mathcal{I}$, as $\mathcal{P} \subseteq \mathcal{H}^p$

We also have $t \in \mathcal{F}_H \cap \mathcal{H}^p \cap \mathcal{I}$ by (**1**).

As Ψ is level-wise co-observable, we have: $(\exists i \in D_{H,c}(\sigma))\ P_{H,i}^{-1}[P_{H,i}(t)]\sigma \cap \mathcal{F}_H \cap \mathcal{I} \cap \mathcal{H}^p = \emptyset$

$\Rightarrow (\exists i \in D_c(\sigma))\ P_i^{-1}[P_i(t)]\sigma \cap \mathcal{F}_H \cap \mathcal{I} \cap \mathcal{H}^p = \emptyset$, as $D_{H,c}(\sigma) \subseteq D_c(\sigma)$ and thus $P_{H,i} = P_i$

$\Rightarrow (\exists i \in D_c(\sigma))\ P_i^{-1}[P_i(t)]\sigma \cap \mathcal{F} \cap \mathcal{I} \cap \mathcal{P} = \emptyset$, as $\mathcal{F} \cap \mathcal{I} \cap \mathcal{P} \subseteq \mathcal{F}_H \cap \mathcal{I} \cap \mathcal{H}^p$

By Cases (1) and (2), we have: $(\exists i \in D_c(\sigma))\ P_i^{-1}[P_i(t)]\sigma \cap \mathcal{F} \cap \mathcal{I} \cap \mathcal{P} = \emptyset$

As $t \in \mathcal{F} \cap \mathcal{I} \cap \mathcal{P}$ and $\sigma \in \Sigma_c$ are chosen arbitrarily, we conclude that Ψ is globally co-observable.

$\square$

Theorem 4.2:

Proof. Let $K \subseteq L_m(\mathbf{Plant})$, $K \neq \emptyset$.

If part) Assume K is controllable and co-observable with respect to $L(\mathbf{Plant})$.

We will show this implies that there exists a marking nonblocking decentralized supervisory control $\mathcal{S}_{Con}$ for (K, $\mathbf{Plant}$) such that $L_m(\mathcal{S}_{Con}/\mathbf{Plant}) = K$.

We must first construct a suitable decentralized supervisory control $\mathcal{S}_{Con}$ for $\mathbf{Plant}$.

For each $i \in D$ and $t \in L(\mathbf{Plant})$, we define the local decentralized supervisory control as follows: $\mathcal{S}_{P_i}(t) := (\Sigma \backslash \Sigma_{c,i}) \cup \{\sigma \in \Sigma_{c,i} \mid P_i^{-1}[P_i(t)]\sigma \cap \overline{K} \cap L(\mathbf{Plant}) \neq \emptyset\}$.

The global decentralized supervisory control policy $\mathcal{S}_{Con}$ is defined as follows: $\mathcal{S}_{Con}(t) := \cap_{i=1}^{N} \mathcal{S}_{P_i}(t)$.

The language $L(\mathcal{S}_{Con}/\mathbf{Plant})$ is defined in Definition 2.8. Clearly, $\mathcal{S}_{Con}$ is a decentralized supervisory control as defined in Definition 2.9.

We will now show that $L_m(\mathcal{S}_{Con}/\mathbf{Plant}) = K$ (Step 1.2) and that $\mathcal{S}_{Con}$ is nonblocking (Step 1.3). To do this, our first step is to show that $L(\mathcal{S}_{Con}/\mathbf{Plant}) = \overline{K}$ (Step 1.1).

Step 1.1) Show that $L(\mathcal{S}_{Con}/\mathbf{Plant}) = \overline{K}$.

We will now show that (A) $L(\mathcal{S}_{Con}/\mathbf{Plant}) \subseteq \overline{K}$ and (B) $\overline{K} \subseteq L(\mathcal{S}_{Con}/\mathbf{Plant})$.

Part A) Show that $L(\mathcal{S}_{Con}/\mathbf{Plant}) \subseteq \overline{K}$.

Let $t \in L(\mathcal{S}_{Con}/\mathbf{Plant})$. We will now prove by induction on the length of string t that $t \in \overline{K}$.

Base case: $t = \epsilon$

We know that $\epsilon \in L(\mathcal{S}_{Con}/\mathbf{Plant})$ by definition. Further, $\epsilon \in \overline{K}$ since $K \neq \emptyset$ by assumption. We thus have $t \in \overline{K}$.

Inductive step: For $\sigma \in \Sigma$, we assume $t\sigma \in L(\mathcal{S}_{Con}/\mathbf{Plant})$ and $t \in \overline{K}$. We will now show this implies $t\sigma \in \overline{K}$.

We have $t \in L(\mathcal{S}_{Con}/\mathbf{Plant})$, $(\forall i \in D)\, \sigma \in \mathcal{S}_{P_i}(t)$, and $t\sigma \in L(\mathbf{Plant})$, by definition of $L(\mathcal{S}_{Con}/\mathbf{Plant})$ and $\mathcal{S}_{Con}$.

We have two cases: (A.1) $\sigma \in \Sigma_{uc}$ or (A.2) $\sigma \in \Sigma_c$.

Case A.1) $\sigma \in \Sigma_{uc}$

From above, we have: $t \in \overline{K}$, $\sigma \in \Sigma_{uc}$, and $t\sigma \in L(\mathbf{Plant})$.

As K is controllable, we have: $\overline{K}\Sigma_{uc} \cap L(\mathbf{Plant}) \subseteq \overline{K}$.

$\Rightarrow t\sigma \in \overline{K}$

Case A.2) $\sigma \in \Sigma_c$

From above, we have: $t \in \overline{K}$, $\sigma \in \Sigma_c$, $(\forall i \in D)\, \sigma \in \mathcal{S}_{P_i}(t)$, and $t\sigma \in L(\mathbf{Plant})$.

We will show $t\sigma \in \overline{K}$ using proof by contradiction. Assume $t\sigma \notin \overline{K}$.

$\Rightarrow t\sigma \in L(\mathbf{Plant}) \backslash \overline{K}$

As K is co-observable with respect to $L(\mathbf{Plant})$, we have: $(\exists i \in D_c(\sigma))\; P_i^{-1}[P_i(t)]\sigma \cap \overline{K} \cap L(\mathbf{Plant}) = \emptyset$.

$\Rightarrow (\exists i \in D_c(\sigma))\, \sigma \notin \mathcal{S}_{P_i}(t)$

$\Rightarrow (\exists i \in D)\, \sigma \notin \mathcal{S}_{P_i}(t)$

$\Rightarrow \sigma \notin \mathcal{S}_{Con}(t)$, by definition of $\mathcal{S}_{Con}$

$\Rightarrow t\sigma \notin L(\mathcal{S}_{Con}/\mathbf{Plant})$

This is a contradiction. We thus conclude that $t\sigma \in \overline{K}$.

By Cases (A.1) and (A.2), we have $t\sigma \in \overline{K}$.

This completes the inductive step. We thus conclude by induction that $L(\mathcal{S}_{Con}/\mathbf{Plant}) \subseteq \overline{K}$.

Part B) Show that $\overline{K} \subseteq L(\mathcal{S}_{Con}/\mathbf{Plant})$.

Let $t \in \overline{K}$. We will prove by induction on the length of string t that $t \in L(\mathcal{S}_{Con}/\mathbf{Plant})$.

Base case: $t = \epsilon$

We know that $\epsilon \in \overline{K}$ since $K \neq \emptyset$ by assumption. Further, we have $\epsilon \in L(\mathcal{S}_{Con}/\mathbf{Plant})$ by definition. We thus have $t \in L(\mathcal{S}_{Con}/\mathbf{Plant})$.

Inductive step: For $\sigma \in \Sigma$, we assume $t\sigma \in \overline{K}$ and $t \in L(\mathcal{S}_{Con}/\mathbf{Plant})$. We will now show this implies $t\sigma \in L(\mathcal{S}_{Con}/\mathbf{Plant})$.

We next note that we have $t\sigma \in L(\mathbf{Plant})$ as $t\sigma \in \overline{K}$ and by the assumption that $K \subseteq L_m(\mathbf{Plant}) \subseteq L(\mathbf{Plant})$.

We have two cases: (B.1) $\sigma \in \Sigma_{uc}$ or (B.2) $\sigma \in \Sigma_c$.

Case B.1) $\sigma \in \Sigma_{uc}$

$\Rightarrow \sigma \in \mathcal{S}_{Con}(t)$ as uncontrollable events are enabled by default for $\mathcal{S}_{Con}$

From above we have: $t \in L(\mathcal{S}_{Con}/\mathbf{Plant})$, $\sigma \in \mathcal{S}_{Con}(t)$ and $t\sigma \in L(\mathbf{Plant})$.

$\Rightarrow t\sigma \in L(\mathcal{S}_{Con}/\mathbf{Plant})$ by definition of $L(\mathcal{S}_{Con}/\mathbf{Plant})$

Case B.2) $\sigma \in \Sigma_c$

From above we have: $t \in L(\mathcal{S}_{Con}/\mathbf{Plant})$, $t \in \overline{K}$, $t\sigma \in \overline{K}$, $\sigma \in \Sigma_c$, and $t\sigma \in L(\mathbf{Plant})$

From the definition of $L(\mathcal{S}_{Con}/\mathbf{Plant})$ and $\mathcal{S}_{Con}$, to show that $t\sigma \in L(\mathcal{S}_{Con}/\mathbf{Plant})$, it is sufficient to show that $(\forall i \in D)\, \sigma \in \mathcal{S}_{P_i}(t)$.

Let $i \in D$. If $\sigma \notin \Sigma_{c,i}$, we immediately have: $\sigma \in \mathcal{S}_{P_i}(t)$ as $\sigma \in \Sigma_c \backslash \Sigma_{c,i}$.

We now consider $\sigma \in \Sigma_{c,i}$. It is sufficient to show that: $P_i^{-1}[P_i(t)]\sigma \cap \overline{K} \cap L(\mathbf{Plant}) \neq \emptyset$.

We first note that: $t \in P_i^{-1}[P_i(t)] := \{s \in \Sigma^* | P_i(s) \in \{P_i(t)\}\}$

$\Rightarrow t\sigma \in P_i^{-1}[P_i(t)]\sigma$

As we have $t\sigma \in \overline{K}$ and $t\sigma \in L(\mathbf{Plant})$ from above, we have: $t\sigma \in P_i^{-1}[P_i(t)]\sigma \cap \overline{K} \cap L(\mathbf{Plant})$.

$\Rightarrow P_i^{-1}[P_i(t)]\sigma \cap \overline{K} \cap L(\mathbf{Plant}) \neq \emptyset$

We thus conclude $t\sigma \in L(\mathcal{S}_{Con}/\mathbf{Plant})$.

By Cases (B.1) and (B.2), we have $t\sigma \in L(\mathcal{S}_{Con}/\mathbf{Plant})$.

This completes the inductive step. We thus conclude by induction that $\overline{K} \subseteq L(\mathcal{S}_{Con}/\mathbf{Plant})$.

By Parts (A) and (B), we have $L(\mathcal{S}_{Con}/\mathbf{Plant}) = \overline{K}$.

Step 1.2) Show that $L_m(\mathcal{S}_{Con}/\mathbf{Plant}) = K$.

By the definition of marking nonblocking decentralized supervisory control, we have: $L_m(\mathcal{S}_{Con}/\mathbf{Plant}) = L(\mathcal{S}_{Con}/\mathbf{Plant}) \cap K$.

Substituting $L(\mathcal{S}_{Con}/\mathbf{Plant}) = \overline{K}$ (by Step (1.1)), we have: $L_m(\mathcal{S}_{Con}/\mathbf{Plant}) = \overline{K} \cap K = K$.

Step 1.3) Show that $\mathcal{S}_{Con}$ is nonblocking.

It is sufficient to show that $\overline{L_m(\mathcal{S}_{Con}/\mathbf{Plant})} = L(\mathcal{S}_{Con}/\mathbf{Plant})$.

The result is automatic as $L(\mathcal{S}_{Con}/\mathbf{Plant}) = \overline{K}$ (by Step (1.1)) and $L_m(\mathcal{S}_{Con}/\mathbf{Plant}) = K$ (by Step (1.2)).

By Steps (1.1), (1.2) and (1.3), we conclude that there exists a marking nonblocking decentralized supervisory control $\mathcal{S}_{Con}$ for $(K, \mathbf{Plant})$ such that $L_m(\mathcal{S}_{Con}/\mathbf{Plant}) = K$.

Only if part) Assume there exists a marking nonblocking decentralized supervisory control $\mathcal{S}_{Con}$ for $(K, \mathbf{Plant})$ such that $L_m(\mathcal{S}_{Con}/\mathbf{Plant}) = K$.

We will now show this implies that K is controllable (Step 2.1) and co-observable with respect to $L(\mathbf{Plant})$ (Step 2.2).

We first note that as $\mathcal{S}_{Con}$ is nonblocking, $\overline{K} = \overline{L_m(\mathcal{S}_{Con}/\mathbf{Plant})} = L(\mathcal{S}_{Con}/\mathbf{Plant})$.

Step 2.1) Show that K is controllable with respect to $L(\mathbf{Plant})$. Sufficient to show that $\overline{K}\Sigma_{uc}\cap L(\mathbf{Plant})\subseteq\overline{K}$.

Let $t \in \overline{K}$, $\sigma \in \Sigma_{uc}$ and $t\sigma \in L(\mathbf{Plant})$.

$\Rightarrow t \in L(\mathcal{S}_{Con}/\mathbf{Plant})$ and $\sigma \in \mathcal{S}_{Con}(t)$, as $L(\mathcal{S}_{Con}/\mathbf{Plant}) = \overline{K}$ and by the definition of $\mathcal{S}_{Con}$

$\Rightarrow t\sigma \in L(\mathcal{S}_{Con}/\mathbf{Plant})$, by definition of $L(\mathcal{S}_{Con}/\mathbf{Plant})$

$\Rightarrow t\sigma \in \overline{K}$, as $L(\mathcal{S}_{Con}/\mathbf{Plant}) = \overline{K}$

$\Rightarrow \overline{K}\Sigma_u\cap L(\mathbf{Plant})\subseteq\overline{K}$

Step 2.2) Show that K is co-observable with respect to $L(\mathbf{Plant})$.

Sufficient to show that: $(\forall t \in \overline{K} \cap L(\mathbf{Plant}))$ $(\forall \sigma \in \Sigma_c)$ $t\sigma \in L(\mathbf{Plant})\backslash\overline{K} \Rightarrow (\exists i \in D_c(\sigma))$ $P_i^{-1}[P_i(t)]\sigma \cap \overline{K} \cap L(\mathbf{Plant}) = \emptyset$.

Let $t \in \overline{K} \cap L(\mathbf{Plant})$, $\sigma \in \Sigma_c$ and $t\sigma \in L(\mathbf{Plant})\backslash\overline{K}$.

$\Rightarrow t\sigma \in L(\mathbf{Plant})$ and $t\sigma \notin \overline{K}$

$\Rightarrow t\sigma \notin L(\mathcal{S}_{Con}/\mathbf{Plant})$ as $L(\mathcal{S}_{Con}/\mathbf{Plant}) = \overline{K}$

$\Rightarrow (\exists i \in D)\sigma \notin \mathcal{S}_{P_i}(t))$, by the definition of $L(\mathcal{S}_{Con}/\mathbf{Plant})$ and $\mathcal{S}_{Con}$

$\Rightarrow (\exists i \in D)$ $(\sigma \in \Sigma_{c,i}) \wedge (P_i^{-1}[P_i(t)]\sigma \cap \overline{K} \cap L(\mathbf{Plant}) = \emptyset)$, by the definition of $\mathcal{S}_{P_i}$

$\Rightarrow (\exists i \in D_c(\sigma))$ $P_i^{-1}[P_i(t)]\sigma \cap \overline{K} \cap L(\mathbf{Plant}) = \emptyset$, by the definition of $D_c(\sigma)$.

By Steps (2.1) and (2.2), we conclude that K is controllable and co-observable with respect to $L(\mathbf{Plant})$.

By If and Only if part, we conclude that there exists a marking nonblocking decentralized supervisory control $\mathcal{S}_{Con}$ for $(K, \mathbf{Plant})$ such that $L_m(\mathcal{S}_{Con}/\mathbf{Plant}) = K$ if and only if K is controllable and co-observable with respect to $L(\mathbf{Plant})$.

□

Corollary 4.1:

Proof.

If part) Assume $L_m(\mathbf{Spec}) \cap L_m(\mathbf{Plant})$ is controllable and co-observable with respect to $L(\mathbf{Plant})$, and $\overline{L_m(\mathbf{Spec}) \cap L_m(\mathbf{Plant})} = L(\mathbf{Spec}) \cap L(\mathbf{Plant})$.

Take $K = L_m(\mathbf{Spec}) \cap L_m(\mathbf{Plant})$ and we have by Theorem 4.2 there exists an MNDSC $\mathcal{S}_{Con}$ for $(L_m(\mathbf{Spec}) \cap L_m(\mathbf{Plant}), \mathbf{Plant})$ such that $L_m(\mathcal{S}_{Con}/\mathbf{Plant}) = L_m(\mathbf{Spec}) \cap L_m(\mathbf{Plant})$.

As $\mathcal{S}_{Con}$ is nonblocking by Theorem 4.2, we have:

$\overline{L_m(\mathbf{Spec}) \cap L_m(\mathbf{Plant})} = \overline{L_m(\mathcal{S}_{Con}/\mathbf{Plant})} = L(\mathcal{S}_{Con}/\mathbf{Plant})$.

As $\overline{L_m(\mathbf{Spec}) \cap L_m(\mathbf{Plant})} = L(\mathbf{Spec}) \cap L(\mathbf{Plant})$ by assumption, we have:

$L(\mathcal{S}_{Con}/\mathbf{Plant}) = L(\mathbf{Spec}) \cap L(\mathbf{Plant})$.

Only if part) Assume there exists an MNDSC $\mathcal{S}_{Con}$ for $(L_m(\mathbf{Spec})\cap L_m(\mathbf{Plant}), \mathbf{Plant})$ such that $L_m(\mathcal{S}_{Con}/\mathbf{Plant}) = L_m(\mathbf{Spec}) \cap L_m(\mathbf{Plant})$ and $L(\mathcal{S}_{Con}/\mathbf{Plant}) = L(\mathbf{Spec})\cap L(\mathbf{Plant})$.

Take $K = L_m(\mathbf{Spec}) \cap L_m(\mathbf{Plant})$ and we have by Theorem 4.2 that $L_m(\mathbf{Spec}) \cap L_m(\mathbf{Plant})$ is controllable and co-observable with respect to $L(\mathbf{Plant})$.

As $\mathcal{S}_{Con}$ is nonblocking, we have: $\overline{L_m(\mathbf{Spec}) \cap L_m(\mathbf{Plant})}$
$= \overline{L_m(\mathcal{S}_{Con}/\mathbf{Plant})} = L(\mathcal{S}_{Con}/\mathbf{Plant}) = L(\mathbf{Spec}) \cap L(\mathbf{Plant})$.

$\square$

References

[BL00] George Barrett and Stephane Lafortune. Decentralized supervisory control with communicating controllers. *IEEE Transactions on Automatic Control*, 45(9):1620–1638, 2000.

[BMM04] Bertil A Brandin, Robi Malik, and Petra Malik. Incremental verification and synthesis of discrete-event systems guided by counter examples. *IEEE Transactions on Control Systems Technology*, 12(3):387–401, 2004.

[CL08] Christos G Cassandras and Stephane Lafortune. *Introduction to discrete event systems, 2nd edition*. Springer, 2008.

[DES14] DESpot. The official website for the DESpot project. [Online] Available: http://www.cas.mcmaster.ca/~leduc/DESpot.html, 2014.

[HJDQ$^+$10] R.C. Hill, Cury J.E.R., MH De Queiroz, DM Tilbury, and S Lafortune. Multi-level hierarchical interface-based supervisory control. *Automatica*, 46(7):1152–1164, 2010.

[LBLW05] Ryan J. Leduc, Bertil A Brandin, Mark Lawford, and WM Wonham. Hierarchical interface-based supervisory control-part I: serial case. *IEEE Transactions on Automatic Control*, 50(9):1322–1335, 2005.

[Led02] R. J. Leduc. *Hierarchical Interface-based Supervisory Control*. PhD thesis, Department of Electrical and Computer Engineering, University of Toronto, Toronto, Ont., 2002.

[Led09] Ryan J. Leduc. Hierarchical interface-based supervisory control with data events. *International Journal of Control*, 82(5):783–800, 2009.

[Liu15] Huailiang Liu. *Hierarchical Interface-Based Decentralized Supervisory Control*. PhD thesis, Department of Computing and Software, McMaster University, Hamilton, Ont., 2015.

[LLD06] Ryan J. Leduc, Mark Lawford, and Pengcheng Dai. Hierarchical interface-based supervisory control of a flexible manufacturing system. *IEEE Transactions on Control Systems Technology*, 14(4):654–668, 2006.

[LLMR14] Huailiang Liu, Ryan J. Leduc, Robi Malik, and S. L. Ricker. Incremental verification of co-observability in discrete-event systems. In *Proc. of 2014 American Control Conference*, pages 5446–5452, Portland, Oregon, USA, June 2014.

[LLR15] Huailiang Liu, Ryan J. Leduc, and S. L. Ricker. Hierarchical interface-based decentralized supervisory control. In *Proceedings of 54th IEEE Conference on Decision and Control*, pages 1693–1700, Osaka, Japan, December 2015.

[LLW05] Ryan J. Leduc, Mark Lawford, and W Murray Wonham. Hierarchical interface-based supervisory control-part II: parallel case. *IEEE Transactions on Automatic Control*, 50(9):1336–1348, 2005.

[LW88] F. Lin and W. M. Wonham. On observability of discrete-event systems. *Infomation Sciience*, 44:173–198, 1988.

[RC11] Laurie Ricker and Benoit Caillaud. Mind the gap: Expanding communication options in decentralized discrete-event control. *Automatica*, 47(11):2364–2372, 2011.

[RW87] P. Ramadge and W. Murray Wonham. Supervisory control of a class of discrete-event processes. *SIAM J. Control Optim*, 25(1):206–230, 1987.

[RW92] Karen Rudie and W Murray Wonham. Think globally, act locally: Decentralized supervisory control. *IEEE Transactions on Automatic Control*, 37(11):1692–1708, 1992.

[RW95] Karen Rudie and Jan C Willems. The computational complexity of decentralized discrete-event control problems. *IEEE Transactions on Automatic Control*, 40(7):1313–1319, 1995.

[RYL03] Kurt Rohloff, Tae-Sic Yoo, and Stéphane Lafortune. Deciding co-observability is PSPACE-complete. *IEEE Transactions on Automatic Control*, 48(11):1995–1999, 2003.

[SB11] Klaus Schmidt and Christian Breindl. Maximally permissive hierarchical control of decentralized discrete event systems. *IEEE Transactions on Automatic Control*, 56(4):723–737, 2011.

[SM06] Klaus Schmidt and Thomas Moor. Marked-string accepting observers for the hierarchical and decentralized control of discrete event systems. In *Proceedings of 8th International Workshop on Discrete Event Systems*, pages 413–418, Ann Arbor, Michigan, USA, July 2006.

[SMP08] Klaus Schmidt, Thomas Moor, and Sebastian Perk. Nonblocking hierarchical control of decentralized discrete event systems. *IEEE Transactions on Automatic Control*, 53(10):2252–2265, 2008.

[TL09] JG Thistle and HM Lamouchi. Effective control synthesis for partially observed discrete-event systems. *SIAM Journal on Control and Optimization*, 48(3):1858–1887, 2009.

[Tri04] Stavros Tripakis. Undecidable problems of decentralized observation and control on regular languages. *Information Processing Letters*, 90(1):21–28, 2004.

[Tsi89] John N Tsitsiklis. On the control of discrete-event dynamical systems. *Mathematics of Control, Signals and Systems*, 2(2):95–107, 1989.

[Won14] W.M. Wonham. Supervisory control of discrete-event systems. Department of Electrical and Computer Engineering, University of Toronto, July 2014. [Online] Available: http://www.control.toronto.edu/DES/.

[WR87] W. M. Wonham and P. Ramadge. On the supremal controllable sublanguage of a given language. *SIAM J. Control Optim*, 25(3):637–659, May 1987.

[WVS96] KC Wong and JH Van Schuppen. Decentralized supervisory control of discrete-event systems with communication. *Report-Department of Operations Research, Statistics, and System Theory*, (6):1–10, 1996.

[YL02] Tae-Sic Yoo and Stephane Lafortune. NP-completeness of sensor selection problems arising in partially observed discrete-event systems. *IEEE Transactions on Automatic Control*, 47(9):1495–1499, 2002.