Redundant Tasks in Multitasking Control of Discrete Event Systems

Klaus Schmidt, José E. R. Cury

Abstract

This paper is an extended version of [1]. It addresses the control of multitasking DES that allow for dealing with liveness properties in the case where multiple classes of tasks have to be independently completed by the system. Colored marking generators (CMG) have been previously introduced as a model to consider multitasking control. The computational cost of the supervisor synthesis for multitasking DES grows with the number of classes of tasks. In this paper we investigate conditions under which removing tasks of the DES model does not affect the result of supervisory control in the sense that their completion is guaranteed as a consequence of the completion of the other tasks in the DES model. Conditions are derived under which tasks of a class or a set of classes can be removed from the model, and the results are extended to the case of abstracted models in a hierarchical and decentralized control architecture. Those conditions, which can be verified in polynomial time, are stated as properties of strongly connected components of the automata models in different levels of the control hierarchy. The results of the paper are illustrated by a manufacturing system example, showing the potential gains of the approach.

I. Introduction

The supervisory control theory (SCT) is an expressive framework for the synthesis of controllers for discrete-event systems (DES) [2]. In the SCT, automata models are used as representations for the plant and closed-loop desired behaviors, while marked states are used to represent completion of tasks in the system. In this framework a supervisor is synthesized in a way that it constrains the behavior of the plant in order to respect the closed-loop specification and such that it ensures nonblocking, i.e., it always allows the controlled system to reach a marked state. In fact, in the SCT, nonblocking can be interpreted

K. Schmidt is with the Lehrstuhl für Regelungstechnik, Universität Erlangen-Nürnberg, Germany, klaus.schmidt@rt.eei.uni-erlangen.de

J.E.R. Cury is with the Department of Automation and Systems, Federal University of Santa Catarina, Florianópolis SC 88040-900 Brazil, {cury}@das.ufsc.br

as a liveness specification that ensures that the supervisor will never prevent the completion of a task in the system. In [3] an approach is introduced to allow for dealing with the case where multiple classes of tasks are identified and (strongly) nonblocking corresponds to the ability of the system to independently complete tasks of all different classes. DES problems comprising multiple classes of tasks often arise in applications, like in manufacturing and communication systems for example [4], [5], [3], [6], [7]. Colored marking generators (CMG) are introduced in [3] for the synthesis of a minimally restrictive supervisor that respects the admissible behavior and ensures the liveness of multiple tasks. Modular control in this framework is addressed in [5].

In [4], multitasking control is extended with hierarchical and decentralized control ideas [8], [9], [10] by combining the computational efficiency of hierarchical abstractions with the ability to specify multiple liveness objectives. To this end, a colored (multitasking) version of both the natural projection and the observer property [11] is employed in the hierarchical abstraction process such that the resulting hierarchical control architecture is hierarchically consistent [12] and (strongly) nonblocking.

The computational cost of the supervisor synthesis for multitasking DES grows with the number of classes of tasks in the system. This is essentially due to the co-accessibility test involved in the synthesis procedure which must be performed with relation to each of the classes of tasks (colors in the automata models). In some particular cases it may be observed that completion of tasks in a particular class is always guaranteed as a consequence of completion of tasks of other classes. Such *redundant* tasks, that may be introduced either by the modeling process of the DES or as a consequence of the abstraction process in hierarchical control, could be removed from the model to reduce the computational cost. In this paper we derive conditions under which tasks of a class can be identified as redundant tasks, and extend the results to the case where hierarchical and decentralized architectures are to be used. Those conditions, which can be verified in polynomial time, are stated as properties on strongly connected components of the automata models for the plant behaviors in different levels of the control hierarchy.

The results of the paper are illustrated by an example of hierarchical decentralized control of a manufacturing system where effective reductions in the number of classes of tasks of the abstracted models in different levels of the hierarchical architecture are obtained. Also, the example shows that, by applying the approach introduced in the paper, the number of states of the resulting abstracted high-level models is potentially reduced since the colored observer property need not be considered for the removed tasks.

The paper is organized as follows. Section II introduces the basic concepts of multitasking supervisory control. Main results are presented in Section III together with the description of the algorithmic procedure

to verify the stated conditions. Section IV provides an extension of the derived results to hierarchical and decentralized control. The detailed example in Section V illustrates the approach, and some conclusions are given in Section VI.

II. Multitasking Discrete Event Systems

A. Basic Notation

For a multitasking discrete-event system (MTDES), a color (label) can be associated to each class of task. Tasks belong to the same class when they are related to liveness objectives that have the same meaning in the control problem. Let Σ be the set of all events that can occur in the system and C be the set of all colors. Let Σ^* be the set of all finite strings of elements in Σ, including the empty string ϵ. A language L is a subset of Σ^*. $\overline{L}$ represents the prefix closure of L. Each color $c \in C$ is assigned to a language $L_c \in Pwr(\Sigma^*)$ (power set of Σ^*) that represents the set of all sequences of events in Σ that can complete a task of the respective class. Thus, the *colored behavior* of a MTDES can be modeled by the set $\Lambda_C \in Pwr(Pwr(\Sigma^*) \times C)$ given by $\Lambda_C := \{(L_c, c) | c \in C\}$.

For a colored behavior Λ_C, the language marked by $c \in C$ is defined by $L_c(\Lambda_C) := L$ such that $(L, c) \in \Lambda_C$. The language marked by $B \subseteq C$ is defined by $L_B(\Lambda_C) := \bigcup_{b \in B} L_b(\Lambda_C)$. For $M_{B_1} \in Pwr(Pwr(\Sigma_1^*) \times B_1)$ and $N_{B_2} \in Pwr(Pwr(\Sigma_2^*) \times B_2)$, $M_{B_1} \subseteq N_{B_2}$ if $B_1 \subseteq B_2$ and $\forall b \in B_1$, $L_b(M_{B_1}) \subseteq L_b(N_{B_2})$.

The *synchronous composition* of M_{B_1} and N_{B_2} is

$$\begin{aligned} M_{B_1} || N_{B_2} := & \{(L_b(M_{B_1}) || L_b(N_{B_2}), b), \forall b \in B_1 \cap B_2\} \\ \cup & \{(L_b(M_{B_1}) || L_{B_2}(N_{B_2}), b), \forall b \in B_1 - B_2\} \\ \cup & \{(L_{B_1}(M_{B_1}) || L_b(N_{B_2}), b), \forall b \in B_2 - B_1\}. \end{aligned}$$

An MTDES can be modeled by a Moore automaton, whose outputs, represented by subsets of colors, define the classes of tasks that are completed after the corresponding strings. Such a *colored marking generator* (CMG), is formally defined by a 6-tuple $G = (Q, \Sigma, C, \delta, \chi, q_0)$, where Q is a set of states; Σ is a set of events; C is a set of colors; $\delta : Q \times \Sigma \to Q$ is a transition function; $\chi : Q \to Pwr(C)$ is a marking function; q_0 is the initial state.

For a CMG G, the *eligible event function* $\Gamma : Q \to Pwr(\Sigma)$ associates each state $q \in Q$ to a subset of Σ with all events that can occur in q. In order to extend δ to a partial function on $Q \times \Sigma^*$, recursively let $\delta(q, \varepsilon) = q$ and $\delta(q, s\sigma) = \delta(\delta(q, s), \sigma)$, whenever both $q' = \delta(q, s)$ and $\delta(q', \sigma)$ are defined. The *generated language* $L(G) := \{s \in \Sigma^* | \delta(q_0, s) \text{ is defined}\}$ of G, is the set of all finite event strings that can be reached from the initial state q_0.

The language *marked* by $c \in C$, is given by $L_c(G) := \{s \in L(G) | c \in \chi(\delta(q_0, s))\}$. For the color set B, $\emptyset \subset B \subseteq C$, the language marked by B is defined by $L_B(G) := \{s \in L(G) | B \cap \chi(\delta(q_0, s)) \neq \emptyset\}$. The colored behavior of a CMG G is given by $\Lambda_C(G) := \{(L_c(G), c) | c \in C\}$.

A formal definition of the *synchronous composition* $G_1||G_2$ of two CMGs G_1 and G_2 is given in [3]. Note that $L(G_1||G_2) = L(G_1)||L(G_2)$ and $\Lambda_C(G_1||G_2) = \Lambda_C(G_1)||\Lambda_C(G_2)$.

Given a nonempty subset of colors B, a CMG G is *strongly nonblocking* w.r.t. B, if $\forall b \in B$, $L(G) = \overline{L_b(G)}$, that is, if any generated string can be completed (not necessarily in the same way) to a task of all the classes represented by colors of B. A colored behavior $\Lambda_C \in Pwr(Pwr(\Sigma^*) \times C)$ is strongly nonblocking w.r.t. $B \subseteq C$ when $\forall b \in B$, $\overline{L_b(\Lambda_C)} = \overline{L_C(\Lambda_C)}$.

B. Multitasking Supervisory Control

Let a MTDES be modeled by a colored marking generator $G = (Q, \Sigma, C, \delta, \chi, q_0)$, with eligible event function Γ, whose alphabet is partitioned into controllable events Σ_c and uncontrollable events Σ_u. We assume w.l.o.g. that a *colored specification* $A_D \subseteq Pwr(\Sigma^*) \times D$ is constructed from a safety specification $K = \overline{K} \subseteq L(G)$ and liveness conditions defined by the set of classes of tasks C and a set of new classes E s.t. $E \cap C = \emptyset$ and $D = C \dot{\cup} E$ as follows.

$$A_D = \{(L_c, c) | \quad c \in D \text{ s.t. } L_c = K \cap L_c(G) \text{ for } c \in C \text{ and } L_c \subseteq K \text{ for } c \in E\}. \tag{1}$$

A *coloring supervisor* $S : L(G) \to Pwr(\Sigma) \times Pwr(E)$ is a mapping that associates to each sequence of events of the plant a set of enabled events and a set of colors (of E) marking the sequence as a completed task of the classes represented by these colors.

For $S(s) = (\gamma, \mu)$, let $\mathcal{R}(S(s)) = \gamma$ and $\mathcal{I}(S(s)) = \mu$. The events that can occur in S/G after the occurrence of a string $s \in L(G)$ are given by $\mathcal{R}(S(s)) \cap \Gamma(\delta(q_0, s))$. A string $s \in L(S/G)$ is marked by a color $c \in C$ if $s \in L_c(G)$ or by a color $e \in E$ if $e \in \mathcal{I}(S(s))$. A coloring supervisor S is *admissible* if $\forall s \in L(G)$, $\Sigma_u \cap \Gamma(\delta(q_0, s)) \subseteq \mathcal{R}(S(s))$.

A supervisor S is strongly nonblocking w.r.t D if $\forall d \in D$, $\overline{L_d(S/G)} = L(S/G)$.

Theorem 1 ([3]): Necessary and sufficient conditions for the existence of an admissible coloring supervisor S strongly nonblocking w.r.t. D such that $\Lambda_D(S/G) = A_D$ and $L(S/G) = \overline{L_D(A_D)}$ are:

1) controllability: $\overline{L_D(A_D)}\Sigma_u \cap L(G) \subseteq \overline{L_D(A_D)}$;
2) D-closure: $\forall d \in (D \cap C)$, $L_d(A_D) = \overline{L_d(A_D)} \cap L_d(G)$;
3) strong nonblocking of A_D w.r.t. D.

In [3], it is also proved that the supremal controllable and strongly nonblocking colored behavior contained in A_D, named $\mathrm{SupCSNB}(A_D, G, D)$, exists and can be computed with complexity polynomial in the number of states of the model.

III. Removing Redundant Colors

The algorithmic computation of $\mathrm{SupCSNB}(A_D, G, D)$ as defined above relies on an iterative computation of nonblocking subbehaviors of $\Lambda_C(G)||A_D$ for all colors $d \in D$. Hence, each additional color contributes to the computational cost of the supervisor synthesis. The goal of this section is to identify and remove colors that are not relevant for the supervisor synthesis in order to reduce this computational cost. The idea is first illustrated by an example in Section III-A, and then formalized in Section III-B.

A. Motivation and Problem Formulation

We consider two neighboring components of the production cell in Fig. 1; the conveyor belt C1 and the machine M. The task of C1 is to transport parts to the machine M, which processes each part before it can depart. C1 is modeled by the CMG $G^{(1)}_{\mathrm{C1}}$ in Fig. 2, where C1 stops if `c1stp` occurs, and the events `c1-0`, `c0-1` and `c1-2`, `c2-1` describe the exchange of parts with the neighboring conveyor belts C0 and C2, respectively. The machine (CMG $G^{(1)}_{\mathrm{M}}$) can start processing (`ms`) and finishes processing with the uncontrollable event `mf`. In addition, one color is introduced for each component. It is desired that C1 can always become empty (C1e) and that the machine cannot be prevented from processing (Mp). Note that transitions with controllable events are labeled with a tick and that the set of colors is displayed next to the respective state in all plant models. It is specified in $M^{(1)}_{\mathrm{C1-M}}$ that every part entering C1 has to stop at M and can only leave C1 after processing is finished. Fig. 2 displays the CMG $R^{(1)}_{\mathrm{C1-M}}$ that represents the closed-loop behavior $SupCSNB(G, A_D, D)$ with the plant $G := G^{(1)}_{\mathrm{C1}}||G^{(1)}_{\mathrm{M}}$, the specification behavior $A_D := L(M^{(1)}_{\mathrm{C1-M}})||\Lambda_C(G)$ according to (1) and the color set $D = C = \{\mathrm{C1e}, \mathrm{Mp}\}$.

A closer inspection of $R^{(1)}_{\mathrm{C1-M}}$ reveals that, although the colors C1e and Mp were introduced independently in their respective component models, there is a direct dependency after the supervisor synthesis. Suppose an additional SNB supervisor $\tilde{S}$ shall be designed for a new specification $\tilde{A}_D$ and the plant $R^{(1)}_{\mathrm{C1-M}}$ with the color set $D = \{\mathrm{C1e}, \mathrm{Mp}\}$. It is now sufficient to synthesize a supervisor that is nonblocking w.r.t. the color set $\tilde{D} = \{\mathrm{Mp}\}$ since this already implies SNB w.r.t. D due to the plant structure. In particular, $\tilde{S}$ makes sure that a state with color Mp is always reachable in $\tilde{S}/R^{(1)}_{\mathrm{C1-M}}$. Observing that $\tilde{S}$ can never disable the uncontrollable event `mf`, this implies that it is also always possible that an unmarked state is reached in $\tilde{S}/R^{(1)}_{\mathrm{C1-M}}$. Then, it holds that on each path back to a state with

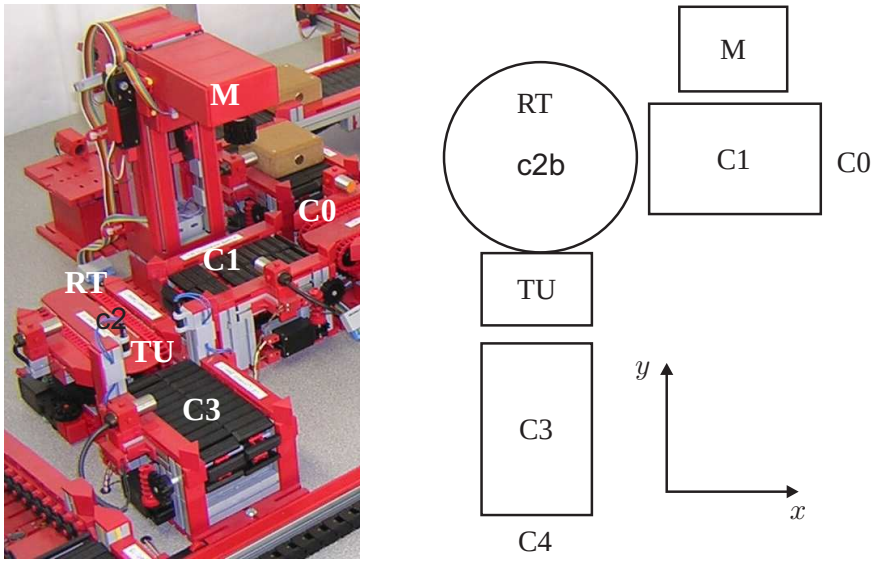

Fig. 1. Production cell: (a) Picture; (b) Schematic overview.

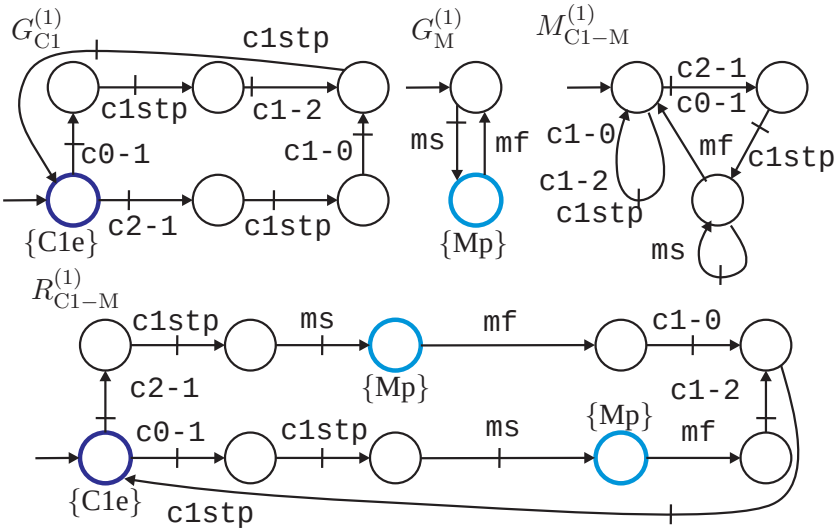

Fig. 2. Conveyor belt C1 and machine M.

color Mp, a state with the color C1e is passed. It can hence be concluded that the color C1e is not relevant for any further supervisor synthesis and can be removed from $R^{(1)}_{\mathrm{C1-M}}$.

Based on this motivating example, the goal of this section is to identify colors in plant models that are not relevant for the supervisor synthesis. In order to reduce the computational effort for the supervisor

computation, we then propose to remove such colors. Formally, we want to solve Problem 1.

Problem 1: Let $G = (Q, \Sigma, C, \delta, \chi, q_0)$ be a CMG, let Σ_u be a set of uncontrollable events and assume that $c \in C$ is a color. We want to determine verifiable conditions such that for all specifications A_D according to (1)

$$\Lambda_D(\tilde{S}/G) = \Lambda_D(SupCSNB(G, A_D, D)),$$

where $\tilde{S} : L(G) \to 2^\Sigma \times 2^{D-C}$ is a coloring supervisors for the reduced specification $A_{\tilde{D}} = \{(L_d(A_d), d) | d \in \tilde{D}\}$ over the color set $\tilde{D} = D - \{c\}$, i.e.,

$$\Lambda_{\tilde{D}}(\tilde{S}/G) = SupCSNB(G, A_{\tilde{D}}, \tilde{D}).$$

In that case, c need not be considered in the supervisor synthesis, i.e., c can be removed from the color set C of G and the specification $A_{\tilde{D}}$ can be used instead of A_D.

B. Condition for Color Removal

In order to formulate the main theorem of this section, we adapt the definition of a *strongly connected component* (SCC) in [13] to CMGs.

Definition 1 (SCC): Let $G = (Q, \Sigma, C, \delta, \chi, q_0)$ be a CMG. A subgraph of G with the states $\mathcal{G} \subseteq Q$ is called a strongly connected component (SCC) of G if for all state pairs $q, q' \in \mathcal{G}$, there is $u, u' \in \Sigma^*$ s.t. $\delta(q, u) = q'$ and $\delta(q', u') = q$ and for all $\mathcal{G}' \supset \mathcal{G}$, $\mathcal{G}'$ is not a SCC of G. □

Theorem 2 states sufficient conditions to solve Problem 1.

Theorem 2 (Main Theorem): Write $\tilde{C} = C - \{c\}$. Problem 1 is solved if the following condition is satisfied. There is no SCC with the states $\mathcal{G} \subseteq Q$ in G s.t.

(i) $\bigcup_{q \in \mathcal{G}} \chi(q) = \tilde{C}$

(ii) $\nexists \sigma \in \Sigma_\mathrm{u}$, $q \in \mathcal{G}$ s.t. $\delta(q, \sigma) \notin \mathcal{G}$.

The condition in Theorem 2 exploits structural information about the plant G. It is shown in Proposition 1 that it implies that whenever a supervisor is SNB for the reduced color set $\tilde{C}$, then it is also SNB for C.

Proposition 1: Let $G = (Q, \Sigma, C, \delta, \chi, q_0)$ be a CMG that is strongly nonblocking (SNB) w.r.t. C, Σ_u the set of uncontrollable events and $c \in C$ a color. Also write $\tilde{C} = C - \{c\}$. Then, there exists a supervisor $S : L(G) \to Pwr(\Sigma) \times Pwr(E)$ s.t. S/G is SNB w.r.t. $\tilde{C}$ but S/G is not SNB w.r.t. C if and only if the condition in Theorem 2 is violated. □

We first formulate the following lemma that will help to prove the necessary part of the proposition:

Lemma 1: Let $G = (Q, \Sigma, C, \delta, \chi, q_0)$ be a CMG that is strongly nonblocking (SNB) w.r.t. C and $c \in C$ a color. Also write $\tilde{C} = C - \{c\}$. Assume that $S : L(G) \to Pwr(\Sigma) \times Pwr(E)$ is a supervisor s.t. S/G is SNB w.r.t. $\tilde{C}$ but S/G is not SNB w.r.t. C and write $S/G = (X, \Sigma, C, \nu, \xi, x_0)$. Then, there is a SCC with the states $\mathcal{S} \subseteq X$ in S/G s.t.

(i) $\bigcup_{x \in \mathcal{S}} \xi(x) = \tilde{C}$

(ii) $\nexists x \in \mathcal{S}$, $\sigma \in \Sigma$ s.t. $\nu(x, \sigma) \notin \mathcal{S}$. □

Proof of Lemma 1: Since S is SNB w.r.t. $\tilde{C}$ and not SNB w.r.t. C, there must be a SCC with the states $\mathcal{S}_1$ s.t. $\bigcup_{x \in \mathcal{S}_1} \xi(x) = \tilde{C}$ and $\forall x \in \mathcal{S}_1$, $\nexists u \in \Sigma^*$ s.t. $c \in \xi(\nu(x, u))$. For $\mathcal{S}_1$, there are two possible cases.

If $\nexists x \in \mathcal{S}_1$, $\sigma \in \Sigma$ s.t. $\nu(x, \sigma) \notin \mathcal{S}_1$, the lemma is already proved with $\mathcal{S} = \mathcal{S}_1$. Otherwise, let $x \in \mathcal{S}_1$ and $\sigma_1 \in \Sigma$ s.t. $\nu(x, \sigma_1) \notin \mathcal{S}_1$. Since S is SNB w.r.t. $\tilde{C}$, there must be a $u_1 \in \Sigma^*$ that leads to a SCC $\mathcal{S}_2$ with $\bigcup_{x \in \mathcal{S}_2} \xi(x) = \tilde{C}$, i.e., $\nu(x, \sigma_1 u_1) \in \mathcal{S}_2$.

The same argument can now be applied to iteratively find SCCs $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \ldots$. However, since X is finite and all SCCs are disjoint, there must be a SCC $\mathcal{S}_m$, where $\nexists x \in \mathcal{S}_m$, $\sigma \in \Sigma$ s.t. $\nu(x, \sigma) \notin \mathcal{S}_m$. With $\mathcal{S} = \mathcal{S}_m$, the lemma is proved. ■

Now Proposition 1 can be proved.

Proof of Proposition 1: We first show the necessary part "⇒": According to Lemma 1, there is a SCC with the states $\mathcal{S} \subseteq X$ s.t. $\bigcup_{x \in \mathcal{S}} \xi(x) = \tilde{C}$ and $\nexists x \in \mathcal{S}$, $\sigma \in \Sigma$ s.t. $\nu(x, \sigma) \notin \mathcal{S}$.

Denote the states of the corresponding SCC in G as $\mathcal{G}$, i.e., $\mathcal{G} = \{q \in Q | q = \delta(q_0, s) \wedge s \in L(S/G)$ s.t. $\nu(x_0, s) \in \mathcal{S}\}$. As $L_{\tilde{c}}(S/G) = L(S/G) \cap L_{\tilde{c}}(G)$ and $\bigcup_{x \in \mathcal{S}} \xi(x) = \tilde{C}$, it must also hold that $\bigcup_{q \in \mathcal{G}} \chi(q) = \tilde{C}$.

It remains to show that $\nexists \sigma \in \Sigma_{\mathrm{u}}$, $q \in \mathcal{G}$ s.t. $\delta(q, \sigma) \notin \mathcal{G}$. Assume such σ exists for $q \in \mathcal{G}$. Also let $s \in L(S/G)$ s.t. $\nu(x_0, s) \in \mathcal{S}$ and $\delta(q_0, s) = q$. Considering that $\delta(q, \sigma) \notin \mathcal{G}$, it must be the case that $\nu(x_0, s\sigma)$ is not defined since $\nexists x \in \mathcal{S}$, $\sigma \in \Sigma$ s.t. $\nu(x, \sigma) \notin \mathcal{S}$ according to Lemma 1. Hence, $s\sigma \in L(G)$ and $s\sigma \notin L(S/G)$ imply that $\sigma \notin S(s)$. This is a contradiction to the assumption that $\sigma \in \Sigma_{\mathrm{u}}$. Thus, $\nexists \sigma \in \Sigma_{\mathrm{u}}$, $q \in \mathcal{G}$ s.t. $\delta(q, \sigma) \notin \mathcal{G}$ which concludes the proof of the necessary part.

Now we show the sufficient part "⇐": We simply construct a supervisor $S : L(G) \to Pwr(\Sigma) \times Pwr(E)$ that is SNB w.r.t. $\tilde{C}$ and not SNB w.r.t. C, For each string $s \in L(G)$, we define

$$S(s) := \begin{cases} \Sigma_{\mathrm{u}} \cup \{\sigma \in \Sigma | \delta(q_0, s\sigma) \in \mathcal{G}\} & \text{if } \delta(q_0, s) \in \mathcal{G} \\ \Sigma & \text{otherwise} \end{cases}$$

Note that $\Sigma_{\mathrm{u}} \subseteq S(s)$ for all $s \in L(G)$.

We first show that S/G is SNB w.r.t. $\tilde{C}$. Let $\tilde{c} \in \tilde{C}$ and $s \in L(S/G)$. First assume that $\delta(q_0, s) \in \mathcal{G}$. Then, $\exists u \in \Sigma^*$ s.t. $\tilde{c} \in \chi(\delta(q_0, su))$ and for all $u' \leq u$ it holds that $\delta(q_0, su') \in \mathcal{G}$. Applying the

supervisor S as defined above, it follows that also $su \in L(S/G)$ and hence $su \in L_{\tilde{c}}(S/G)$. Now assume that $\delta(q_0, s) \notin \mathcal{G}$. Since G is SNB w.r.t. C, there is $v \in \Sigma^*$ s.t. $\tilde{c} \in \chi(\delta(q_0, sv))$. If it holds that for all $v' \leq v$, $\delta(q_0, sv') \notin \mathcal{G}$, then $sv \in L(S/G)$ according to the supervisor definition and hence $sv \in L_{\tilde{c}}(S/G)$. Otherwise let $u' \leq v$ s.t. $\delta(q_0, su') \in \mathcal{G}$. But then, as shown above, there is a $u \in \Sigma^*$ s.t. $su'u \in L_{\tilde{c}}(S/G)$.

Finally, we prove that S is not SNB w.r.t. C. Let $s \in L(G)$ s.t. $\delta(q_0, s) \in \mathcal{G}$. Then, $s \in L(S/G)$. We want to show that $\nexists u \in \Sigma^*$ s.t. $su \in L_c(S/G)$ by contradiction. Assume that $u \in \Sigma^*$ s.t. $su \in L_c(S/G)$. Then, $su \in L(G)$ and it must hold that $\delta(q_0, su) \notin \mathcal{G}$. Hence, there is $u' \in \Sigma^*$ and $\sigma \in \Sigma$ s.t. $u'\sigma \leq u$, $\delta(q_0, su') \in \mathcal{G}$ and $\delta(q_0, su'\sigma) \notin \mathcal{G}$. But then, the definition of S implies that $\sigma \notin S(s)$, which contradicts the assumption that $su \in L_c(S/G)$. ■

Finally, Theorem 2 can be proved.

Proof of Theorem 2: For convenience, we write $B_D := \Lambda_D(SupCSNB(G, A_D, D))$ and $B_{\tilde{D}} = \{(L_d(B_D), d) | d \in \tilde{D}\}$. It has to be shown that $B_D = \Lambda_D(\tilde{S}/G)$. $B_D \subseteq \Lambda_D(\tilde{S}/G)$ directly follows since $B_{\tilde{D}} \subseteq A_{\tilde{D}}$, B_D is SNB w.r.t. $\tilde{D}$ and $L_D(B_D)$ is controllable w.r.t. G, i.e. $B_{\tilde{D}}$ is a controllable subbehavior of $A_{\tilde{D}}$ that is SNB w.r.t $\tilde{D}$.

It remains to show that also $\Lambda_D(\tilde{S}/G) \subseteq B_D$. Assume the contrary, i.e., $\Lambda_D(\tilde{S}/G) \supset B_D$. Then, there is $s \in L_D(\tilde{S}/G)$ s.t. $s \notin L_D(B_D)$. Since $\tilde{S}/G$ is SNB w.r.t. $\tilde{D}$, it holds for all $d \in \tilde{D}$ that $s \in \overline{L_d(\tilde{S}/G)}$. With (1), $s \in \overline{L_d(A_D)}$ for all $d \in \tilde{D}$, and hence $s \in K$.

Considering that $s \notin L_D(B_D)$, whereas $L(\tilde{S}/G)$ is controllable w.r.t. G and at the same time $s \in \overline{L_c(G)}$ since $\tilde{S}/G$ is nonblocking w.r.t. c according to Proposition 1, it must hold that $s \notin \overline{L_c(A_D)}$. However, since $s \in \overline{L_c(\tilde{S}/G)}$, there is $u \in \Sigma^*$ s.t. $su \in K$ and $su \in L_c(G)$. But then, $su \in L_c(A_D)$ because of (1), which contradicts the assumption that $s \notin \overline{L_c(A_D)}$. ■

The condition in Theorem 2 applies to the motivating example in Section III-A. It holds that both SCCs of $R^{(1)}_{\mathrm{C1-M}}$ with the color Mp (the two states colored with Mp) have an uncontrollable transition with `mf` leaving the respective SCC.

C. Algorithmic Verification

The following algorithm allows to check the condition in Theorem 2 by finding an SCC as specified in Theorem 2 if such SCC exists.

Algorithm 1 (Check Color Removal): The algorithm checks if a color $c \in C$ can be removed from a CMG G.

Given: CMG $G = (Q, \Sigma, C, \delta, \chi, q_0)$, color c.

Output: **true** if c can be removed, **false** otherwise

1. $\tilde{G} = (\tilde{Q}, \tilde{\Sigma}, \tilde{C}, \tilde{\delta}, \tilde{\chi}, \tilde{q}_0) = G$
2. delete all states with color c from $\tilde{G}$ and remove c from $\tilde{C}$:
$$\forall q \in \tilde{Q} : c \in \tilde{\chi}(q) \Rightarrow \tilde{Q} := \tilde{Q} - \{q\}; \quad \tilde{C} := \tilde{C} - \{c\}$$
3. find all SCCs in $\tilde{G}$ that contain states with all colors in $\tilde{C}$. Denote these SCCs as $\mathcal{G}_1, \ldots, \mathcal{G}_m$.
4. remove all states from $\tilde{G}$ that do not belong to $\bigcup_{i=1}^{m} \mathcal{G}_i$:
$$\forall q \in \tilde{Q} : q \notin \bigcup_{i=1}^{m} \mathcal{G}_i \Rightarrow \tilde{Q} := \tilde{Q} - \{q\}$$
5. delete all states in $\mathcal{G}_i$, $i = 1, \ldots, m$ that have uncontrollable transitions in the original automaton G that lead outside $\mathcal{G}_i$, i.e., $\forall i \in \{1, \ldots, m\}$:
$$\forall q \in \mathcal{G}_i, \forall \sigma \in \Sigma_{\mathrm{u}} : \delta(q, \sigma) \notin \mathcal{G}_i \Rightarrow \tilde{Q} := \tilde{Q} - \{q\}$$
6. **if** states were deleted in step 5. **and** $\tilde{Q}$ is not empty
   go to step 3.
7. **if** $\tilde{Q}$ is empty
   return true
   else
   return false □

The algorithm iteratively removes states from the plant CMG G if they violate item (i) (step 3. and 4.) or if they violate item (ii) (step 5.) in Theorem 2. The algorithm terminates in a maximum number of $|Q|$ steps, where $|Q|$ is the number of states of G. Furthermore, the computation of the SCCs in step 3. can be performed by Tarjan's algorithm in [13] with a complexity of $O(\max\{|Q|, |\delta|\})$, where $|\delta|$ denotes the number of transitions of G. Together, Algorithm 1 exhibits a computational complexity of $O(|Q| \cdot \max\{|Q|, |\delta|\})$.

We apply Algorithm 1 for $G = R^{(1)}_{\mathrm{C1-M}}$ in Section III-A and $c = \mathrm{C1e}$. In step 2., the initial state is removed. Two SCCs that consist of the states with the color MP remain after the steps 3. and 4. Since the uncontrollable event `mf` leads outside both SCCs, $\tilde{Q}$ is empty after step 5. Hence, the algorithm returns **true**, which is consistent with the previous discussion in Section III-A and III-B.

It is readily observed that an iterative application of the above procedure enables the removal of an arbitrary number of colors as long as Algorithm 1 returns **true**.

Remark 1: It has to be noted that the set of colors that can be removed for a given CMG G is not unique. Defining and deriving an optimal set of colors to be removed is not in the scope of this paper.

IV. Multitasking Hierarchical and Decentralized Control

In the previous section, it is pointed out that the removal of redundant colors leads to computational savings in the supervisor synthesis for MTDES. In this section, we combine the idea of removing colors with the hierarchical and decentralized control approach for MTDES as elaborated in [4]. The application example in Section V then illustrates that the combined approach can result in additional computational savings due to smaller plant models.

A. Control Approach

It is assumed that the original (low-level) plant is given as a set $G_i = (Q_i, \Sigma_i, C_i, \delta_i, \chi_i, q_{0,i})$, $i = 1, \ldots, n$ of CMGs, and the overall plant is $G = ||_{i=1}^{n} G_i$ with the color set $C := \bigcup_{i=1}^{n} C_i$. The hierarchical abstraction of G is based on the *colored natural projection*.

Definition 2 (Colored Natural Projection): Let $\Lambda_C \in Pwr(Pwr(\Sigma^*) \times C)$ be a colored behavior, and assume $\Sigma_0 \subseteq \Sigma$ with the natural projection $p_0 : \Sigma^* \to \Sigma_0^*$. The colored natural projection $m_0 : Pwr(Pwr(\Sigma^*) \times C) \to Pwr(Pwr(\Sigma_0^*) \times C)$ is defined such that

$$L_c(m_0(\Lambda_C)) = p_0(L_c(\Lambda_C)), \text{ for all } c \in C.$$

The *high-level plant* G_0 is then computed using abstractions of the plant components G_i, $i = 1, \ldots n$ on a superset of their *shared events* $\Sigma_{i,\cap} := \bigcup_{k=1, k\neq i}^{n} (\Sigma_i \cap \Sigma_k)$.

Definition 3 (High-level Plant): Let G and G_i, $i = 1, \ldots, n$, p_0 and m_0 be defined as above. Assume that high-level alphabets $\Sigma_{i,0} \subseteq \Sigma_i$ are given such that $\Sigma_{i,\cap} \subseteq \Sigma_{i,0}$ and introduce the natural projections $p_{\Sigma_i \to \Sigma_{i,0}}$ and the colored natural projections $m_{\Sigma_i \to \Sigma_{i,0}}$. Then, the high-level plant G_0 is defined by

$$G_0 = ||_{i=1}^{n} G_{i,0},$$

where $L(G_{i,0}) := p_{\Sigma_i \to \Sigma_{i,0}}(L(G_i))$ and $\Lambda_C(G_{i,0}) := m_{\Sigma_i \to \Sigma_{i,0}}(\Lambda_C(G_{i,0}))$. Furthermore, it is shown in [4] that $L(G_0) = p_0(L(G))$ and $\Lambda_C(G_0) = m_0(\Lambda_C(G))$. □

The abstraction process is illustrated on the right-hand side of Fig. 3. Given a coloring behavior $A_{D,0} \in Pwr(Pwr(\Sigma_0^*) \times D)$ as a high-level specification, the coloring *high-level supervisor* $S_0 : L(G_0) \to Pwr(\Sigma_0) \times Pwr(E)$ with $E = D - C$ is computed such that S_0 realizes $SupCSNB(A_{D,0}, G_0, D)$. The control action of the corresponding *low-level supervisor* $S : L(G) \to Pwr(\Sigma) \times Pwr(E)$ is then defined for each $s \in L(G)$ as

$$S(s) := \big(S_0(p_0(s)) \cup (\Sigma - \Sigma_0), \mathcal{I}(S_0(p_0(s)))\big), \tag{2}$$

such that $L(S/G) = L(S_0/G_0)||L(G)$ and $\Lambda_C(S/G) = \Lambda_C(S_0/G_0)||\Lambda_C(G)$.

Hence, the control action after a string $s \in L(G)$ observed by each subsystem is

$$\big(\mathcal{R}(S(s)) \cap \Sigma_i, \mathcal{I}(S(s)) \cap (C_i \cup E)\big).$$

The supervisor implementation is depicted on the left-hand side of Fig. 3.

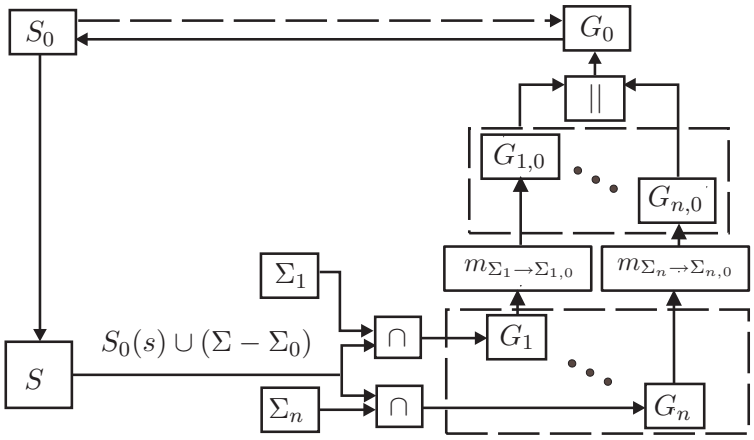

Fig. 3. Hierarchical and decentralized control architecture

In order to guarantee that the low-level closed loop S/G is SNB, the *colored observer* condition is introduced in [4].

Definition 4 (Colored Observer): Let $L \subseteq \Sigma^*$ be a language and let $\Lambda_C \in Pwr(Pwr(\Sigma^*) \times C)$ be a coloring behavior with $L_C(\Lambda_C) \subseteq L$. Also let $\Sigma_0 \subseteq \Sigma$ and p_0, m_0 be defined as above. m_0 is a Λ_C-observer (w.r.t. L) iff for each $c \in C$, p_0 is an $L_c(\Lambda_C)$-observer (w.r.t. L), i.e, for each $s \in \overline{L}$, $t \in \Sigma_0^*$, and $c \in C$

$$p_0(s)t \in L_c(m_0(\Lambda_C)) \Rightarrow \exists u \in \Sigma^* \text{ s.t. } su \in L_c(\Lambda_C) \text{ and } p_0(su) = p_0(s)t$$

Requiring that $m_{\Sigma_i \to \Sigma_{i,0}}$ is a $\Lambda_C(G_i)$-observer for $i = 1, \ldots, n$ is sufficient for strongly nonblocking control.

Theorem 3 ([4]): Assume that G_i, $G_{i,0}$, and $m_{\Sigma_i \to \Sigma_{i,0}}$, $i = 1, \ldots, n$ are defined as above. Also let S_0 be a strongly nonblocking coloring high-level supervisor with a low-level supervisor S as in (2). If $m_{\Sigma_i \to \Sigma_{i,0}}$ is a $\Lambda_C(G_i)$-observer (w.r.t. $L(G_i)$) for all $i = 1, \ldots, n$, then the overall closed loop is SNB, i.e., for all $c \in C$

$$\overline{L_c(S/G)} = L(S/G).$$

B. Removal of Redundant Colors

We now combine hierarchical control in the framework presented in the previous section with the idea of removing redundant colors. To this end, we first recall the *mutual controllability* from [14].

Definition 5 (Mutual Controllability): The CMGs G_i and G_j are denoted mutually controllable if

$$L(G_i)(\Sigma_{j,\mathrm{u}} \cap \Sigma_i) \cap p_{\Sigma_i \cup \Sigma_j \to \Sigma_i}(p^{-1}_{\Sigma_i \cup \Sigma_j \to \Sigma_j}(L(G_j)) \subseteq L(G_i)$$

$$L(G_j)(\Sigma_{i,\mathrm{u}} \cap \Sigma_j) \cap p_{\Sigma_j \cup \Sigma_i \to \Sigma_j}(p^{-1}_{\Sigma_j \cup \Sigma_i \to \Sigma_i}(L(G_i)) \subseteq L(G_j)$$

Mutual controllability ensures that after any execution of a composed system, the occurrence of a shared uncontrollable event is either feasible in every subsystem which shares it, or it is not feasible in any subsystem.

The following theorem relates the redundancy of a color c in the color set of G to the redundancy of the color in the components where c appears.

Theorem 4: Let $G = ||_{i=1}^{n}$ be a plant with the components G_i, $i = 1, \ldots, n$, and let G_0 be a high-level plant according to Definition 3. Assume that G_i, G_j are mutually controllable for all $i \neq j$. Also denote $C_{i,\cap} := \bigcup_{l=1, l\neq i}^{n}(C_i \cap C_l)$ as the set of shared colors with other components for $i = 1, \ldots, n$, and assume that $c \in C_k - C_{k,\cap}$ for some $k \in \{1, \ldots, n\}$, $\tilde{C} := C - \{c\}$, and S_0 is a supervisor such that S_0/G_0 is SNB for $\tilde{C}$. If G_k fulfills the condition in Theorem 2 for $C_k - \{c\}$, then S evaluated with (2) is SNB w.r.t. C. □

We first establish Proposition 2.

Proposition 2: Let G and G_i, $i = 1, \ldots, n$ be defined as above and assume that G_i, G_j are mutually controllable for all $i \neq j$. Then, for any $k \in \{1, \ldots, n\}$ and any $c \in C_k - C_{k,\cap}$, it holds that the condition in Theorem 2 is fulfilled for G and $\tilde{C} = C - \{c\}$ if it is fulfilled for G_k and $\tilde{C}_k = C_k - \{c\}$. □

Proof of Proposition 2: Assume that $c \in C_k - C_{k,\cap}$, all G_i, G_j are mutually controllable and the condition in Theorem 2 is fulfilled for G_k and $\tilde{C}_k$. To proceed by contradiction, let there be a SCC in G with the states $\mathcal{G} \subseteq Q$ such that $\tilde{C} \subseteq \bigcup_{q\in\mathcal{G}} \chi(q)$ and there is no $\sigma \in \Sigma_{\mathrm{u}}$, $q \in \mathcal{G}$ s.t. $\delta(q, \sigma) \notin \mathcal{G}$. Define $\mathcal{G}_k := \{q \in Q_k | q = \delta_k(q_{0,k}, p_{\Sigma\to\Sigma_k}(s)) \wedge \delta(q_0, s) \in \mathcal{G}\}$ the set of states in G_k that correspond to states in $\mathcal{G}$. Then $\mathcal{G}_k$ represents a SCC in G_k, and since $c \in C_k - C_{k,\cap}$, we have that $\bigcup_{q\in\mathcal{G}_k} \chi_k(q) = \tilde{C}_k$.

Considering that the condition in Theorem 2 is fulfilled for G_k and $\tilde{C}_k$, it must hold that there is a $s \in L(G)$, $\sigma \in \Sigma_{k,\mathrm{u}}$ s.t. $s\sigma \notin L(G)$ but $p_{\Sigma\to\Sigma_k}(s)\sigma \in L(G_k)$ while $\delta_k(q_{0,k}, p_{\Sigma\to\Sigma_k}(s)\sigma) \notin \mathcal{G}_k$. Hence, $\sigma \in \Sigma_{k,\cap}$, and for some $j \neq k$ with $\sigma \in \Sigma_{j,\cap}$, it must be true that $p_{\Sigma\to\Sigma_j}(s) \in L(G_j)$ but $p_{\Sigma\to\Sigma_j}(s)\sigma \notin L(G_j)$. However this contradicts the mutual controllability of G_k and G_j. ■

Based on this result, we can prove the above theorem.

Proof of Theorem 4: With Theorem 3, we know that S/G is SNB w.r.t. $\tilde{C}$. But then, Proposition 2 and Proposition 1 imply that S/G is also SNB w.r.t. C. ■

Employing the result in Theorem 4, we propose the following procedure for the combination of hierarchical abstraction and color removal.

1. Remove all redundant colors $c \notin C_{i,\cap}$ from each G_i and denote the remaining colors by $\tilde{C}_i \subseteq C_i$
2. Determine colored observers $m_{\Sigma_i \to \Sigma_{i,0}}$, $i = 1, \ldots, n$ and compute G_0
3. Synthesize the supervisor S according to (2) for a given high-level specification $A_{D,0}$.

Remark 2: Note that S is not necessarily maximally permissive as discussed in [4]. An extension to maximally permissive control as suggested in [15] is not in the scope of this paper.

V. Application Example

In this section, we apply hierarchical multitasking control to the production cell (PC) in Fig. 1, and illustrate how removing redundant colors can decrease the computational effort for the supervisor synthesis.

A. General Setup

In addition to the components C1 and M described in Section III-A, the PC consists of the conveyor belts C2 and C3, the rotary table RT, and a test unit TU. CMG models for all components have been determined based on physical plant events (sensors and actuators). However, the description in this paper starts with plant models on the hierarchical level (1) in order to provide a compact representation. The state counts of the closed-loop CMGs $R_i^{(0)}$, $i \in \mathcal{C} := \{\mathrm{C1}, \mathrm{C2}, \mathrm{C3}, \mathrm{M}, \mathrm{RT}, \mathrm{TU}\}$ on the lowest level (0) are displayed in Fig. 5 (next to the respective CMG).

B. Models on Level 1

The conveyor belt C1 and the machine M are described in Section III-A. We employ the following characterization for the remaining plant components, where the same convention for event names is used (see also Fig. 4).

Conveyor belt C2 ($G_{\mathrm{C2}}^{(1)}$): C2 allows to transport parts from C1 to C3 or from C3 to C2 and back to C3 or to C1. The color C2e requires C2 to always become empty again.

Rotary table RT ($G_{\mathrm{RT}}^{(1)}$): RT initially points in the x-direction. It can turn to the y-direction (`RTy`) and back to the x-direction (`RTx`). RT must always be able to stop (`RTstp`) in one of its two positions (color RTs).

Conveyor belt C3 ($G^{(1)}_{\mathrm{C3}}$): C3 accepts parts from C2 and C4, and then delivers them either to C2 or C4. The color C3e indicates that C3 should always become empty again.

Test unit TU ($G^{(1)}_{\mathrm{TU}}$): TU is located between C2 and C3. It checks parts that travel from C2 to C3 or vice versa, and decides if they are acceptable (`acc`) or have to be rejected (`rej`). TU keeps track of parts until they leave towards C1 (`c2-1`) or C4 (`c3-4`). By coloring, we ensure that parts can always be either accepted (A) or rejected (R).

It has been verified that all plant components are mutually controllable. Furthermore, it has to be noted that no colors are shared among the components.

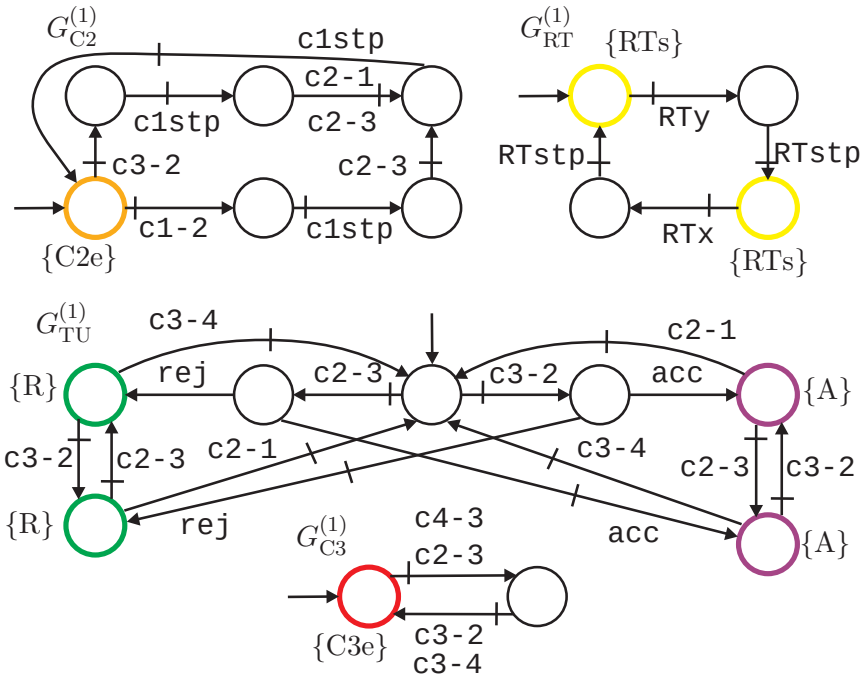

Fig. 4. Level 1 models of the production cell.

C. Hierarchical Supervisor Synthesis

We now perform hierarchical supervisor synthesis according to Section IV-B. The hierarchical architecture is presented in Fig. 5, where gray and white boxes denote closed-loop CMGs and abstracted plant models, respectively.

The computation of $R^{(1)}_{\mathrm{C1-M}}$ from $G^{(1)}_{\mathrm{C1}}$ and $G^{(1)}_{\mathrm{M}}$ is performed in Section III-A, where one color (C1e) was removed.

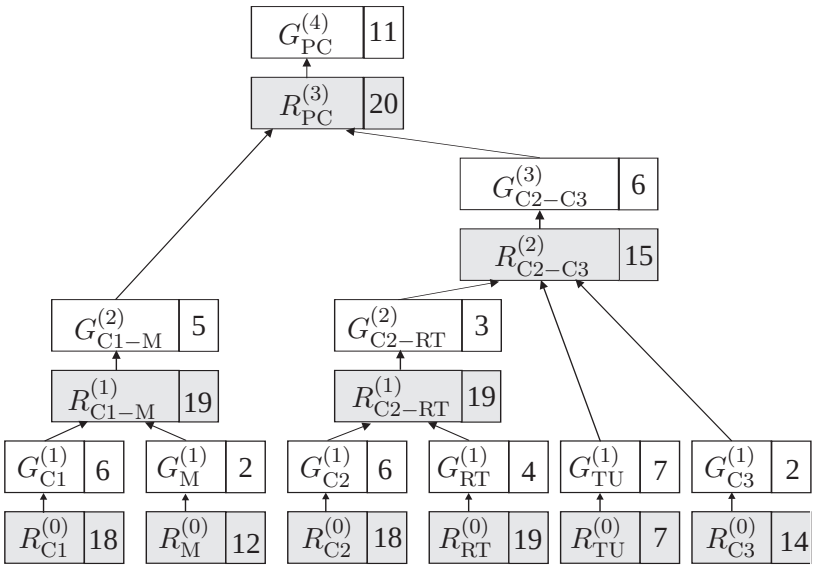

Fig. 5. Hierarchical architecture for the production cell

C2 and RT: A supervisor is designed for the conveyor belt C2 that is mounted on RT. The specifications $M^{(1)}_{\mathrm{C2-RT},1}$ and $M^{(1)}_{\mathrm{C2-RT},2}$ in Fig. 6 require that only one of the components is allowed to move, and RT has to turn according to delivery performed by C1, respectively. The resulting supervisor $R^{(1)}_{\mathrm{C2-RT}}$ has 19 states. Its abstraction $G^{(2)}_{\mathrm{C2-RT}}$ is shown in Fig. 7. No color can be removed in this step.

C2, RT, C3 and TU: The specification $M^{(2)}_{\mathrm{C2-C3}}$ addresses the combined behavior of C2, TU and C3 on level 2 of the hierarchy. It states that accepted parts have to leave PC via C4, while rejected parts have to pass M and leave towards C0. The supervisor $R^{(2)}_{\mathrm{C2-C3}}$ has 15 states and contains the redundant colors RTs and C2e. The abstraction $G^{(3)}_{\mathrm{C2-C3}}$ is shown in Fig. 7.

Production Cell: Finally, the overall PC is composed on level 3 of the hierarchy. Here, we do not consider an additional safety specification such that $R^{(3)}_{\mathrm{PC}}$ is only designed to be SNB. Again, one color (Mp) can be removed such that the abstraction $G^{(4)}_{\mathrm{PC}}$ in Fig. 7 only contains two of the originally seven colors. It can for example be used as a model of the PC in a larger manufacturing system.

In each of the presented steps, it is verified that the conditions in Theorem 4 are fulfilled. Hence, the overall closed-loop system represented by

$$S/G = (\|_{i\in\mathcal{C}} R^{(0)}_i)||R^{(1)}_{\mathrm{C1-M}}||R^{(1)}_{\mathrm{C2-RT}}||R^{(2)}_{\mathrm{C2-C3}}||R^{(3)}_{\mathrm{PC}} \tag{3}$$

is SNB and fulfills the given specifications.

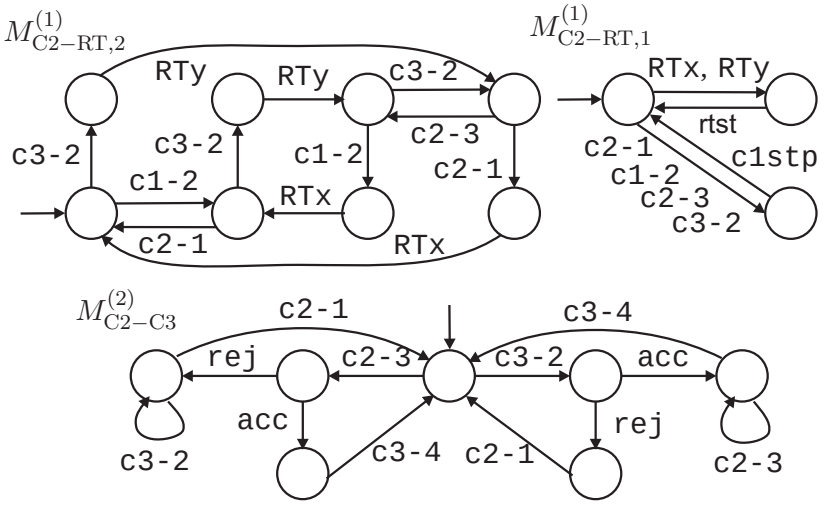

Fig. 6. Safety specifications for the production cell.

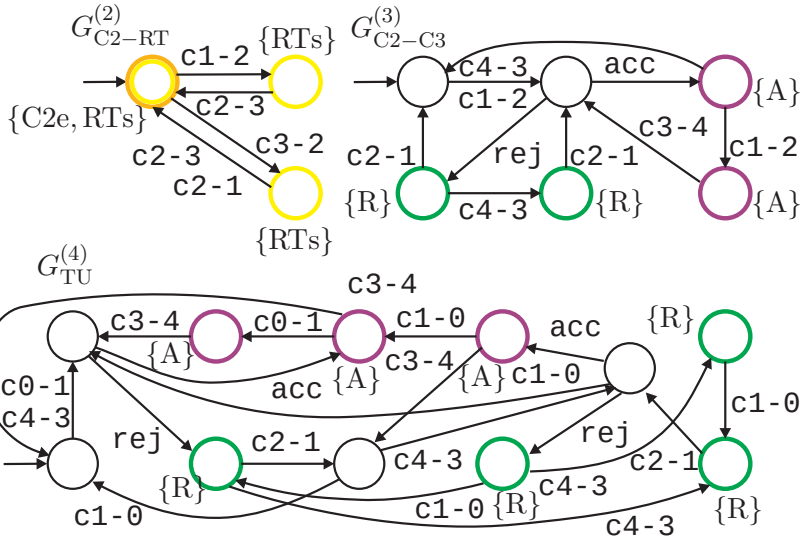

Fig. 7. Hierarchical supervisors for the production cell.

D. Performance Comparison

In comparison, a completely monolithic supervisor synthesis and a hierarchical synthesis without removing colors was carried out. All computations were performed using the "multitasking" plugin of the `libFAUDES` software library for discrete event systems [16].

In the case of monolithic control, the overall plant G comprises $1\,133\,484$ states, the composed

specification has $9\,298$ states, and a monolithic supervisor with $17\,355$ states has to be implemented. In contrast, the hierarchical synthesis in Section V-C comprises a sum of 161 states, since S/G in (3) need not be composed. Furthermore, the largest automaton in the hierarchical synthesis has 28 states.

If hierarchical control without removing colors is used, not only computations for all 7 colors have to be carried out, but it is also observed that the resulting high-level models $G^{(3)}_{\mathrm{C2-C3}}$ (11 states) and $G^{(4)}_{\mathrm{PC}}$ (31 states) are larger compared to the respective models in the previous section. This is due to the fact that the colored observer condition in Definition 4 has to be fulfilled for more colors if no colors are removed during the synthesis process.

VI. Conclusions

The results in this paper show how identifying and removing redundant tasks in multitasking control of DES may lead to considerable savings in the computational effort of synthesizing supervisors for this class of systems. The illustration of the established conditions in the example of a manufacturing cell puts in evidence the gains we can have in hierarchical and decentralized control architectures, not only by the removal of colors in the CMG models of different levels in the system hierarchy, but also by the reduction in the size of the abstracted models as a consequence of eliminating tasks. Further research currently being carried out on this subject includes applying the results in a larger example and deriving algorithmic computations of maximal sets of redundant classes of tasks.

References

[1] K. Schmidt and J. Cury, "Redundant tasks in multitasking control of discrete event systems," in *Workshop on Dependable Control of Discrete Event Systems*, 2009.

[2] P. J. Ramadge and W. M. Wonham, "Supervisory control of a class of discrete event processes," *SIAM Journal of Control and Optimization*, vol. 25, pp. 206–230, 1987.

[3] M. H. de Queiroz, J. E. R. Cury, and W. M. Wonham, "Multitasking supervisory control of discrete-event systems," *Journal on Discrete Event Dynamic Systems: Theory and Applications*, vol. 15, pp. 375–395, 2005.

[4] K. Schmidt, M. de Queiroz, and J. Cury, "Hierarchical and decentralized multitasking control of discrete event systems," *Decision and Control, 2007 46th IEEE Conference on*, pp. 5936–5941, Dec. 2007.

[5] M. H. de Queiroz and J. E. R. Cury, "Modular multi-tasking supervisory control of composite discrete event systems," in *IFAC World Congress*, 2005.

[6] M. Fabian and R. Kumar, "Mutually nonblocking supervisory control of des," *Automatica*, vol. 36, pp. 1863–1869, 2000.

[7] J. G. R. Thistle and P. Malhame, "Multiple marked languages and noninterference in modular supervisory control," in *35th Annual Allerton Conference*, 1997.

[8] R. Hill and D. Tilbury, "Modular supervisory control of discrete-event systems with abstraction and incremental hierarchical construction," *Workshop on Discrete Event Systems (WODES), Ann Arbor, USA*, 2006.

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

[10] K. Schmidt, T. Moor, and S. Perk, "Nonblocking hierarchical control of decentralized discrete event systems," *Automatic Control, IEEE Transactions on*, vol. 53, no. 10, pp. 2252–2265, Nov. 2008.

[11] K. C. Wong and W. M. Wonham, "Hierarchical control of discrete-event systems," *Discrete Event Dynamic Systems: Theory and Applications*, vol. 6, no. 3, pp. 219–314, 1996.

[12] H. Zhong and W. Wonham, "On the consistency of hierarchical supervision in discrete-event systems," *Automatic Control, IEEE Transactions on*, vol. 35, no. 10, pp. 1125–1134, Oct 1990.

[13] J. E. Hopcroft and J. D. Ullman, *The design and analysis of computer algorithms*. Addison-Wesley, 1975.

[14] S.-H. Lee and K. Wong, "Structural decentralised control of concurrent DES," *European Journal of Control*, vol. 35, pp. 1125–1134, October 2002.

[15] K. Schmidt and C. Breindl, "On maximal permissiveness of hierarchical and modular supervisory control approaches for discrete event systems," *Discrete Event Systems, International Workshop on*, pp. 462–467, May 2008.

[16] libFAUDES, "Friedrich-Alexander University Discrete Event Systems library," 2008. [Online]. Available: http://www.rt.eei.uni-erlangen.de/FGdes/faudes/index.php