Liveness-Enforcing Supervision for Resource Allocation Systems with Uncontrollable Behavior and Forbidden States

Jonghun Park and Spyros A. Reveliotis*

Abstract— **Although liveness enforcing supervision (LES) of sequential resource allocation systems (RAS) is currently a well-established problem in the Discrete Event System literature, all prior work on it has addressed the underlying LES synthesis problem under the assumption that the system behavior is totally controllable. The work presented in this paper seeks to develop correct and scaleable LES for RAS that present uncontrollability with respect to (i) the timing of some requested resource allocations, i.e., these allocations will take place as long as the requested resources are available, and/or (ii) the routing of certain job instances that, after some processing stages, might request special treatment or rework. In addition, the last part of the paper addresses the accommodation in the original LES synthesis problem of externally imposed logical constraints, that constitute "forbidden state" specifications and possess a linear characterization with respect to the system resource allocation state. All problems are addressed in the context of Conjunctive/Disjunctive (CD)-RAS, that constitutes one of the broadest RAS classes investigated in the literature, allowing for arbitrarily structured resource allocations associated with the various process stages, and process routing flexibility.**

Keywords— **Discrete Event Systems, Supervisory Control, Resource Allocation Systems, Liveness, Uncontrollability, Forbidden States**

I. Introduction

Liveness enforcing supervision of sequential resource allocation systems (RAS) is currently a well-established problem in the Discrete Event Systems (DES) literature. Briefly, the issue is to develop real-time control policies / supervisors that will constrain the system behavior in a – ideally, the *maximal* – strongly connected component of its underlying state space, which further contains the system initial empty state. Since, however, the computation of the maximally permissive liveness enforcing supervisor (LES) can be shown to be NP-complete even for the simplest sequential RAS structures [1], most of the prior research on this problem has been focused on developing sub-optimal provably correct solutions to it, that are, furthermore, computationally tractable. Indicative examples of this research can be found in [2], [3], [4], [5], [6].

All the aforementioned work presumes that the system is totally *controllable*, i.e., (i) that all the resource allocation events taking place in it can be disabled/postponed at will, and (ii) that jobs presenting routing flexibility can be forced, if necessary, to a certain direction / routing option. The work presented in this paper first seeks to relax the controllability assumption, by developing LES that are applicable to RAS in which some of the aforementioned decisions might be *uncontrollable* by the system supervisor. Uncontrollability with respect to the exact timing of the resource allocation might arise, for instance, by the lack of the necessary hardware that would enforce total controllability, or from the existence of time-critical operations that should not be externally delayed by the system supervisor, once all the required resources are available. On the other hand, uncontrollability with respect to the sequential logic / routing of the various job instances can arise from the need for special treatment and/or rework, given a certain process outcome, that underlies many contemporary technological applications. The results and methodology developed for addressing the aforementioned aspects of uncontrollability in the RAS behavior allow also the accommodation in the LES synthesis problem of *"forbidden state"* constraints that admit a *linear* characterization with respect to the system resource allocation. These constraints would express additional logical requirements that are to be imposed on the system operation due to technological and/or management policy considerations, and they are briefly addressed in the last part of the paper.

The entire problem is considered in the context of the so-called *Conjunctive/Disjunctive (CD)*-RAS, i.e., a RAS class allowing for arbitrarily structured resource allocations associated with the various process stages, and process routing flexibility. From a modeling standpoint, CD-RAS is one of the most powerful RAS classes investigated in the literature, and it has been extensively studied in [7], [6], [8]. From a methodological standpoint, the developed results are based on an extension of the work developed in [7], [8] for synthesizing LES that are appropriate for the class of CD-RAS, and its combination with the Petri net (PN)-based Supervisory Control theory of [9], [10]. Hence, the rest of the paper is organized as follows: Section 2 summarizes the necessary background for developing the key results of the paper. Specifically, the first part of this section overviews the theory of [9], [10] regarding the synthesis of a PN supervisor that enforces a set of linear "forbidden state" constraints on any given PN plant, in a way that respects the plant uncontrollability. The second part of Section 2 introduces the class of CD-RAS, that constitutes the starting point for this work, and its PN-based modeling, and subsequently it overviews the logic of the G-RUN LES [7], [8], which is a correct and scaleable LES for CD-RAS. Section 3 first models the behavior of CD-RAS presenting uncontrollable resource allocations with respect to the exact timing of the corresponding events, and introduces the problem of liveness enforcing supervision in this new RAS context. Subsequently, it demonstrates that the approach of [9], [10] might fail to maintain the controlled system liveness, when applied to modify a G-RUN LES that does not observe this type of RAS uncontrollability, and it proceeds to the (direct) modification of the original G-RUN logic so that the resulting policy version – to be called the *U(ncontrollable)-G-RUN* – provides correct and scaleable LES for CD-RAS with uncontrollable resource allocations. The first part of Section 4 undertakes the modeling of CD-RAS with uncontrolled job routings and reworks. The resulting RAS class constitutes a modification of the CD-RAS to be called the *Extended CD-RAS (ECD-RAS)*. The second part of Section 4 establishes that U-G-RUN provides also a correct LES for ECD-RAS, through a pertinent selection of the policy-defining parameters. Section 5 addresses the additional issue of accommodating linear "forbidden state" constraints in the LES synthesis problem, and integrates all the presented results in a LES synthesis algorithm for (E)CD-RAS, potentially with forbidden states. Finally, Section 6 concludes the paper and suggests directions for future research. In the following discussion, it is assumed that the reader is familiar with the basic Petri net (PN) structural and behavioral concepts; an excellent introduction on Petri net structural analysis can be found in [11].

II. Preliminaries

A. Petri Net-based Supervisory Control theory

The key points of the PN-based supervisory control theory presented in [9], [10] are as follows:[1]

* corresponding author

[1] As it is noticed in ([9], Section 3.1), the idea of enforcing behavioral constraints expressed as linear inequalities on the net (reachable) markings through place invariant-based controllers, and the investigation of the conditions under

I. Given a totally controllable marked PN $\mathcal{N} = (P, T, W, M_0)$, a control specification on the net (reachable) markings, M, expressed by the inequality

$$\mathbf{l}^T M \leq g \tag{1}$$

can be implemented on the net behavior by super-imposing on the net structure a *control* place p_c, connected to the rest of the network according to the flow matrix:

$$\theta_{\mathbf{c}} = -\mathbf{l}^T \mathbf{\Theta} \tag{2}$$

where $\mathbf{\Theta}$ denotes the flow matrix of the original network $\mathcal{N}$. The initial marking of place p_c must be set to:

$$M_{c0} = g - \mathbf{l}^T M_0 \tag{3}$$

and therefore, the problem is feasible only when

$$g - \mathbf{l}^T M_0 \geq 0 \tag{4}$$

The aforementioned controller imposes the specification of Equation 1 on the behavior of the controlled system, $\mathcal{N}' = (P \cup \{p_c\}, T, W', M_0')$, by establishing the following *place invariant* on its reachability space:

$$\mathbf{l}^T M + M_c = g \tag{5}$$

This controller is maximally permissive, since it acts to constrain the behavior of the original net $\mathcal{N}$ only when the firing of a transition $t \in T$ would result in $M_c < 0$, which, according to Equation 5, would violate the original specification constraint of Equation 1. In the case that the control specification is a *system* of inequalities of the type presented in Equation 1, the considered methodology essentially treats them as a logical conjunction, and therefore, it processes them separately, introducing a distinct control place p_{c_i} for each inequality.

II. For PN's containing uncontrollable transitions $T_u \subset T$, the considered methodology will lead to an *acceptable* controller, only if the resulting control net does not attempt to disable any of the uncontrollable transitions. In the light of Equation 2, a sufficient condition for controller acceptability is:

$$\mathbf{l}^T \mathbf{\Theta}_u \leq 0 \tag{6}$$

where $\mathbf{\Theta}_u$ denotes the part of the net flow matrix $\mathbf{\Theta}$ corresponding to the uncontrollable transitions $t \in T_u$.

III. If the place-invariant controller enforcing the specification of Equation 1 on a network $\mathcal{N}$ with uncontrollable transitions $T_u \subset T$ turns out to be unacceptable, then, the original specification must be transformed to another one, $\mathbf{l'}^T M \leq g'$, s.t. (i) its corresponding place invariant controller is acceptable, and (ii) it subsumes the original specification, i.e.,

$$\mathbf{l'}^T M \leq g' \Longrightarrow \mathbf{l}^T M \leq g \tag{7}$$

A set of specification transforms that are guaranteed to satisfy the condition of Equation 7 is as follows:

$$\begin{aligned} \mathbf{l}' &= \mathbf{r}_1 + r_2 \mathbf{l} \\ g' &= r_2(g+1) - 1 \end{aligned} \tag{8}$$

where $\mathbf{r}_1$ is a $|P|$-dimensional vector satisfying $\mathbf{r}_1^T M \geq 0$ for every (reachable) marking M, and r_2 is a (strictly) positive scalar.

which the resulting methodology leads to a control structure that is implementable in the presence of uncontrollable transitions, was also investigated in [12]. However, that work offered no transformation procedures to convert a possibly unacceptable constraint to an acceptable one.

To ensure that the transformed specification obtained through Equation 8 leads to an acceptable place-invariant controller, it must hold:

$$(\mathbf{r}_1 + r_2 \mathbf{l})^T \mathbf{\Theta}_u \leq 0 \tag{9}$$

whereas, the feasibility (non-negativity) of the initial marking for the control place requires that:

$$\mathbf{r}_1^T M_0 + r_2(\mathbf{l}^T M_0 - g - 1) \leq -1 \tag{10}$$

As a result, an effective way to compute the elements $\mathbf{r}_1$ and r_2 that will lead to an acceptable transformation of a non-acceptable specification $\mathbf{l}^T M \geq g$ on a given PN $\mathcal{N} = (P, T, W, M_0)$, is through the solution of the following integer program (IP):

$$\min_{(\mathbf{r_1}, r_2)} z(\mathbf{r_1}, r_2) = \mathbf{r_1}^T M_0 + r_2(\mathbf{l}^T M_0 - g - 1) \tag{11}$$

s.t.

$$\mathbf{r}_1^T \mathbf{\Theta}_u + r_2 \mathbf{l}^T \mathbf{\Theta}_u \leq 0 \tag{12}$$

$$\mathbf{r_1},\ r_2\ \in\ Z^+ \tag{13}$$

After solving the above IP, if the minimum of the objective function, $z^* = z(\mathbf{r}_1^*, r_2^*)$, is greater than -1, it is inferred that the transformation of the original specification according to Equation 8 is not possible, since the condition of Equation 10 cannot be satisfied. Otherwise, the optimal values $(\mathbf{r}_1^*, r_2^*)$ define, through Equation 8, an acceptable transformation, which can be imposed on the plant net $\mathcal{N}$ through a control place obtained according to Equations 2 and 3.

B. CD-RAS modeling and the G-RUN LES

The CD-RAS and its PN-based Model The *Conjunctive/Disjunctive (CD)*-RAS is formally defined by a set of *resource types* $\mathcal{R} = \{R_i, i = 1, \ldots, m\}$, and a set of *job types* $\mathcal{J} = \{J_j, j = 1, \ldots, n\}$. Every resource type R_i is further characterized by its *capacity* $C_i \in Z^+$, where Z^+ is the set of positive integers. Processing requirements of job type J_j are defined by a set of *stages*, partially ordered through a set of precedence constraints. Each job stage p is associated with a *conjunctive* resource allocation requirement, expressed by an m-dimensional vector $(a_{ip})^{i=1,\ldots,m}$, where $a_{ip} \in \{0\} \cup Z^+, i = 1, \ldots, m$, indicates how many units of resource R_i are required to support the stage execution.

To model the resource allocation dynamics taking place in CD-RAS by a Petri net, first, we represent the process flow of each job type J_j by a particular net structure known as *Simple Sequential Process (S^2P)* [13]. This net structure is formally defined by an ordinary strongly connected state machine $\mathcal{N}_j = (P_{S_j} \cup \{p_{0_j}\}, T_j, W_j)$ such that (i) $P_{S_j} \neq \emptyset$, $p_{0_j} \notin P_{S_j}$, (ii) every circuit of $\mathcal{N}_j$ contains $\{p_{0_j}\}$, and (iii) $\forall p \in P_{S_j}$ s.t. $p_{0_j} \in p^{\bullet\bullet}$, $p^{\bullet\bullet} \cap P_{S_j} = \emptyset$. In the S^2P net, each place $p \in P_{S_j}$ is called a *process place*, and it corresponds to a job stage of J_j. Place p_{0_j} represents the *idle place*, since its marking represents the jobs of type J_j waiting to be loaded to the RAS. The PN modeling of the CD-RAS is completed by interconnecting the S^2P nets through a set of *resource places*, P_R, which model the availability of the various resource types. The resulting PN class is referred to as *System of Simple Sequential Processes with General Resource Requirements*, and it will be denoted by S^3PGR^2. Formally, it is defined as follows:

Definition 1: A *well-marked* S^3PGR^2 net is a marked PN $\mathcal{N} = (P, T, W, M_0)$ such that
1. $P = P_S \cup P_0 \cup P_R$, where $P_S = \bigcup_{i=1}^n P_{S_i}$ s.t. $P_{S_i} \cap P_{S_j} = \emptyset, \forall i \neq j$, $P_0 = \bigcup_{i=1}^n \{p_{0_i}\}$ s.t. $P_0 \cap P_S = \emptyset$, and $P_R = \{r_1, \ldots, r_m\}$ s.t. $(P_S \cup P_0) \cap P_R = \emptyset$.

2. $T = \bigcup_{i=1}^{n} T_i$.
3. $W = W_S \cup W_R$, where $W_S : ((P_S \cup P_0) \times T) \cup (T \times (P_S \cup P_0)) \rightarrow \{0, 1\}$ s.t. $\forall j \neq i, ((P_{S_j} \cup P_{0_j}) \times T_i) \cup (T_i \times (P_{S_j} \cup P_{0_j})) \rightarrow \{0\}$, and $W_R : (P_R \times T) \cup (T \times P_R) \rightarrow \{0\} \cup Z^+$.
4. $\forall i,\ i = 1, \ldots, n$, the subnet $\mathcal{N}_i$ generated by $P_{S_i} \cup \{p_{0_i}\} \cup T_i$ is an S^2P net.
5. $\forall r \in P_R$, $\exists$ a unique minimal p-semiflow y_r s.t. $\|y_r\| \cap P_R = \{r\}$, $\|y_r\| \cap P_0 = \emptyset$, $\|y_r\| \cap P_S \neq \emptyset$, and $y_r(r) = 1$. Furthermore, $P_S = \bigcup_{r \in P_R} (\|y_r\| - P_R)$.
6. $\mathcal{N}$ is pure and strongly connected.
7. $\forall p \in P_S,\ M_0(p) = 0$; $\forall r \in P_R,\ M_0(r) \geq max_{p \in \|y_r\|} y_r(p)$; and $\forall p_{0_i} \in P_0,\ M_0(p_{0_i}) \geq 1$.

We remark that the S^3PGR^2 net structure, originally proposed in [14], models the same class of RAS behavior as the S^4PR net proposed in [15].

Algebraic LES The G-RUN (*Generalized Resource Upstream Neighborhood*) LES for CD-RAS, considered in this work, falls into the broader class of *algebraic* LES, that have been proposed in the literature as a mathematically elegant and computationally powerful solution to the problem of deadlock avoidance in sequential RAS, able to provide a viable trade-off between computational tractability and operational efficiency [2], [3], [5], [16], [6]. In the context of S^3PGR^2 nets, an algebraic LES is represented by the following system of linear inequalities:

$$\mathbf{A} \cdot M_S \leq \mathbf{f} \tag{14}$$

In Equation (14), $\mathbf{A} = [\bar{\alpha}_{ip}]_{p \in P_S}^{i=1,\ldots,N}$ is an $N \times |P_S|$ matrix such that $\bar{\alpha}_{ip} \geq 0, i = 1, \ldots, N,\ \forall p \in P_S$, and N is polynomially related to the number of the system resource types, m; M_S is a vector representation of the system resource allocation state, provided by the projection of the net marking M on the subspace defined by the process place subset P_S; $\mathbf{f} = (f_i)_{i=1,\ldots,N}$ is an N-dimensional vector of positive integers. As a control law, Equation 14 implies that the RAS state represented by vector M_S is *admissible* iff Equation 14 is satisfied.

Notice that the type of constraint imposed by Equation 14 on the S^3PGR^2 net modeling the uncontrolled RAS behavior, belongs to the broader class of control specifications expressed by Equation 1. Hence, in the case of totally controllable S^3PGR^2 nets, the discussion of Section 2.1, (c.f., item I) implies that the control logic of an algebraic LES can be superimposed on the original system through a set of control places, $P_W = \{w_1, w_2, \ldots, w_N\}$, that (i) are connected to the original system according to Equation 2, (ii) are initially marked according to Equation 3, and (iii) implement the place invariants expressed by Equation 5. Moreover, it is easy to see that in the operation of the controlled net, these places act as additional fictitious resources that are required for the execution of the various processing stages. This last remark implies that the class of S^3PGR^2 nets is *closed* under the control of algebraic LES, and therefore, the behavior of CD-RAS under an algebraic LES is amenable to the same liveness analysis techniques that apply to the liveness analysis of the uncontrolled system.

G-RUN G-RUN is a class of algebraic LES obtained for any given S^3PGR^2 net $\mathcal{N} = (P_0 \cup P_S \cup P_R, T, W, M_0)$, by setting $N = m(\equiv |\mathcal{R}|)$, $f_i = C_i$, $\forall i = 1, \ldots, m$, and synthesizing matrix $\mathbf{A}$ according to the following logic: (i) Select an (arbitrary) partial ordering $o_i = o(R_i)$, $i = 1, \ldots, m$, and $\forall p \in P_S$, let $\rho_p^{min} = \min\{o_i | a_{ip} > 0, i = 1, \ldots, m\}$. (ii) Define the *i(mmediate)-neighborhood* N_p of any place $p \in P_S$ by $N_p \subseteq p^{\bullet\bullet} \cap P_S$, and associate with every place $p \in P_S$ such that $p^{\bullet\bullet} \cap P_0 = \emptyset$, an i-neighborhood through an arbitrarily selected function $g() : \{p \in P_S \mid p^{\bullet\bullet} \cap P_0 = \emptyset\} \rightarrow \{N_p \neq \emptyset \mid p \in P_S \wedge p^{\bullet\bullet} \cap P_0 = \emptyset\}$. The set $\Psi \equiv \{(p, q) \mid p \in P_S \wedge p^{\bullet\bullet} \cap P_0 = \emptyset \wedge q \in g(p)\}$ will

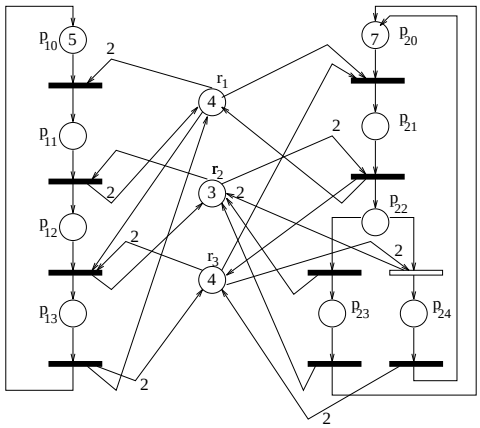

Fig. 1. Example 1: The S^3PGR^2 net

be called the community of $\mathcal{N}$ induced by function $g()$, while the set of all possible communities of $\mathcal{N}$ will be denoted by $\mathcal{C}_\mathcal{N}$. (iii) Matrix $\mathbf{A}$ corresponds to a G-RUN LES induced by ordering $o()$ and community Ψ if and only if (iff) its elements satisfy the following set of constraints:

$$\bar{\alpha}_{ip} \geq \bar{\alpha}_{iq} \qquad \forall (p, q) \in \Psi,\ \forall R_i \in \mathcal{R} \text{ s.t. } \exists R_j \in \mathcal{R} \text{ with } o_i \geq o_j \ \wedge\ \bar{\alpha}_{jp} > 0, \tag{15}$$

$$C_i \geq \bar{\alpha}_{ip} \geq a_{ip} \qquad \forall p \in P_S,\ \forall R_i \in \mathcal{R} \tag{16}$$

$$\bar{\alpha}_{ip} \in \{0\} \cup Z^+ \qquad \forall p \in P_S,\ \forall R_i \in \mathcal{R} \tag{17}$$

The correctness of the G-RUN LES can be established by an argument similar to that establishing the correctness of the original RUN LES in [16], [6], and a formal proof can be found in [7]. Furthermore, an extensive discussion on efficient implementations of G-RUN for any given S^3PGR^2 net is presented in [8]. In particular, for a given resource ordering $o()$ and community Ψ, the matrix $\mathbf{A}$ leading to an efficient implementation of G-RUN LES, can be obtained by solving the following Linear Program (LP) [8]:

$$\min G(\mathbf{A}; \mathcal{N}, o, \Psi) = \sum_{i=1,\ldots,m} \sum_{p \in P_S} \bar{\alpha}_{ip} \tag{18}$$

s.t.

$$\bar{\alpha}_{ip} \geq \bar{\alpha}_{iq} \qquad \forall (p, q) \in \Psi, \forall R_i \in \mathcal{R} \text{ s.t. } o_i \geq \rho_p^{min} \tag{19}$$

$$\bar{\alpha}_{ip} = 0 \qquad \forall p \in P_S, \forall R_i \in \mathcal{R} \text{ s.t. } o_i < \rho_p^{min} \tag{20}$$

$$C_i \geq \bar{\alpha}_{ip} \geq a_{ip} \qquad \forall p \in P_S, \forall R_i \in \mathcal{R} \tag{21}$$

$$\bar{\alpha}_{ip} \geq 0 \qquad \forall p \in P_S, \forall R_i \in \mathcal{R} \tag{22}$$

The next example demonstrates the implementation of G-RUN LES on an S^3PGR^2 net.

Example 1 Consider the S^3PGR^2 net depicted in Figure 1, that corresponds to a CD-RAS with 3 resources R_1, R_2 and R_3, with respective capacities 4, 3 and 4. In its current configuration, the system supports two job types, with the following process stage sequences: $J_1 =< [2,0,0]^T, [0,1,0]^T, [1,0,2]^T >$ and $J_2 =< [1,0,1]^T, [0,2,0]^T, \{[0,1,0]^T, [0,0,2]^T\} >$. Application of the LP of Equations 18 – 22 to this system, with resource ordering $o : (o_1 = 2,\ o_2 = 1,\ o_3 = 3)$ and community $\Psi = \{(p_{11}, p_{12}), (p_{12}, p_{13}), (p_{21}, p_{22}), (p_{22}, p_{23})\}$, leads to the G-RUN implementation expressed by the following constraints:

$$\begin{bmatrix} 2 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 2 & 1 & 0 \\ 2 & 2 & 2 & 1 & 0 & 0 & 2 \end{bmatrix} \cdot M_S \quad \leq \quad \begin{bmatrix} 4 \\ 3 \\ 4 \end{bmatrix} \tag{23}$$

In Equation 23, the rows of matrix $\mathbf{A}$ correspond to the resource sequence $< R_1, R_2, R_3 >$, and its columns correspond to the place sequence $< p_{11}, p_{12}, p_{13}, p_{21}, p_{22}, p_{23}, p_{24} >$.

III. Liveness Enforcing Supervision for CD-RAS with uncontrollable resource allocations

U-S^3PGR^2 nets and acceptable LES In order to formally analyze the problem of liveness enforcing supervision in CD-RAS with uncontrollable resource allocations, we extend the formal characterization of the S^3PGR^2 net, provided in Definition 1, by partitioning the transition set T to two subsets, T_c and T_u, denoting respectively the system controllable and uncontrollable transitions. The modified net structure will be called U-S^3PGR^2 net, and an LES for a given U-S^3PGR^2 net will be characterized as *acceptable* iff it does not disable any transition $t \in T_u$ (in more technical terms, the corresponding control subnet satisfies Equation 6 of Section 2.1).

The synthesis of acceptable LES for the class of U-S^3PGR^2 nets can be formally studied in the framework of *non-blocking* supervisory control (SC) [17], [18]. Hence, based on some broader results of that framework, it is known that the maximally permissive – i.e., *optimal* – LES for the class of U-S^3PGR^2 nets is uniquely defined. Furthermore, the structural boundedness of the U-S^3PGR^2 net implies that the optimal LES is effectively computable. Under the reasonable assumption that the job initiations/releases are controllable – i.e., $P_0^\bullet \subseteq T_c$ – the LES set for any given U-S^3PGR^2 net is non-empty, since the supervisor allowing only one job in the system and controlling no internal transitions, is an acceptable supervisor for this class of nets. As a result, the optimal LES will exist non-trivially. However, it is also true that the algorithmic techniques available in Ramadge & Wonham's SC framework [19] for the computation of the optimal LES for a U-S^3PGR^2 net, require the complete enumeration of the net reachability space, which is a task of exponential complexity. In fact, the complexity of computing the optimal LES for this class of systems can be shown to be an NP-Hard [20] problem, in the general case, and therefore, more computationally efficient techniques and policies, potentially suboptimal, must be developed for effective liveness enforcing supervision in real-life application contexts. The rest of this section first demonstrates that, in the case of nets with uncontrollable transitions, the straightforward transformation of an unacceptable G-RUN supervisor according to the methodology presented in Section 2.1 (c.f. item III), might fail to preserve the liveness of the controlled system. Subsequently, it introduces a modified version of G-RUN LES, to be called the U-G-RUN LES, and establishes that it is an acceptable LES for U-S^3PGR^2 nets.

Example 2 Consider that in the S^3PGR^2 net of Figure 1, $T_u = {}^\bullet p_{24}$.[2] Then, under the place ordering $< p_{10}, p_{11}, p_{12}, p_{13}, p_{20}, p_{21}, p_{22}, p_{23}, p_{24}, r_1, r_2, r_3 >$, the submatrix $\mathbf{\Theta}_u$ of Section 2.1 is equal to $[0,0,0,0,0,0,-1,0,1,0,2,-2]^T$, and the reader can verify that the first two rows of Equation 23, when padded appropriately with 0's corresponding to places $p \in P_0 \cup P_R$, satisfy the acceptability condition of Equation 6, while the third row violates it. Therefore, we apply the constraint transformation step (c.f. Item III) of Section 2.1 to this violating constraint. The reader can verify that the application of the logic of item III in Section 2.1 with $\mathbf{r}_1 = [0,0,0,0,0,0,2,0,0,0,0,0]^T$ and $r_2 = 1$ satisfies all the requirements of that step, and leads to the acceptable transformed constraint:

$$[2\ 2\ 2\ 1\ \mathbf{2}\ 0\ 2] \cdot M_S \leq 4 \tag{24}$$

[2] which in Figure 1 is depicted by drawing this transition as an empty box

which, according to the theory of [9], [10], must substitute the original third constraint in Equation 23. But then, it is easy to see that in the controlled net, the marking with $M(p_{21}) = 4$ and for all other $p \in P_S$, $M(p) = 0$, is reachable from M_0, and all jobs in place p_{21} are deadlocked, since, under the modified control logic, each of them requires an additional unit of the fictitious resource w_3 for its further advancement.

U-G-RUN Given a well-marked U-S^3PGR^2 net $\mathcal{N} = (P_0 \cup P_S \cup P_R, T_c \cup T_u, W, M_0)$, the algebraic LES $(\mathbf{A} = [\bar{\alpha}_{ip}]_{p \in P_S}^{i=1,\ldots,N}, \mathbf{f} = (f_i)_{i=1,\ldots,N})$, is a U-G-RUN LES iff $N = m$; $f_i = C_i$, $i = 1, \ldots, m$; and $\exists\ o() : \mathcal{R} \to \{1, \ldots, m\}$ and some community $\Psi \in \mathcal{C}_\mathcal{N}$, s.t.

$$\bar{\alpha}_{ip} \geq \bar{\alpha}_{iq} \quad \forall (p,q) \in \Psi, \forall R_i \in \mathcal{R}, \text{ s.t. } p^\bullet \cap {}^\bullet q \in T_c \wedge o_i \geq \rho_p^{min} \tag{25}$$

$$\bar{\alpha}_{ip} = 0 \quad \forall (p,q) \in \Psi, \forall R_i \in \mathcal{R}, \text{ s.t. } p^\bullet \cap {}^\bullet q \in T_c \wedge o_i < \rho_p^{min} \tag{26}$$

$$\bar{\alpha}_{ip'} \geq \bar{\alpha}_{iq'} \quad \forall (p',q'), \forall R_i \in \mathcal{R}, \text{ s.t. } p', q' \in P_S \wedge p'^\bullet \cap {}^\bullet q' \in T_u \tag{27}$$

$$C_i \geq \bar{\alpha}_{ip} \geq a_{ip} \quad \forall p \in P_S, \forall R_i \in \mathcal{R} \tag{28}$$

$$\bar{\alpha}_{ip} \in \{0\} \cup Z^+ \quad \forall p \in P_S, \forall R_i \in \mathcal{R} \tag{29}$$

Lemma 1: Consider a U-S^3PGR^2 net $\mathcal{N} = (P_0 \cup P_S \cup P_R, T_c \cup T_u, W, M_0)$, and a U-G-RUN realization on it. Then, the considered supervisor (i) belongs to the (broader) family of G-RUN supervisors defined in Section 2.2; (ii) is *acceptable* w.r.t. T_u.

A formal proof of this result can be found in [7]. When combined with the correctness of the original G-RUN LES, also established in [7], it leads to the following theorem regarding the correctness of U-G-RUN.

Theorem 1: The controlled net $\mathcal{N}' = (P_0 \cup P_S \cup P_R \cup P_W, T_c \cup T_u, W, M_0)$ corresponding to a U-G-RUN implementation on a given well-marked U-S^3PGR^2 net $\mathcal{N} = (P_0 \cup P_S \cup P_R, T_c \cup T_u, W, M_0)$, is a live U-S^3PGR^2 net.

Example 3 Substituting the set of constraints defining U-G-RUN in the LP of Equations 18 – 22, and solving it for the U-S^3PGR^2 net of Figure 1, using the the same resource ordering and community as in Example 1, leads to the following correct and acceptable algebraic LES for this system:

$$\begin{bmatrix} 2 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 2 & 1 & 0 \\ 2 & 2 & 2 & \mathbf{2} & \mathbf{2} & 0 & 2 \end{bmatrix} \cdot M_S \quad \leq \quad \begin{bmatrix} 4 \\ 3 \\ 4 \end{bmatrix} \tag{30}$$

Notice that the LES of Equation 30 still increases the value of $\bar{\alpha}_{3p_{22}}$ from 0 to 2, in order to accommodate the uncontrollability of ${}^\bullet p_{24}$, but it also adjusts appropriately the value of $\bar{\alpha}_{3p_{21}}$ from 1 to 2, so that the system remains live.

IV. The Extended CD-RAS: Accommodating uncontrollable job routings and reworks

ECD-RAS and the S^3PGR^3 net Uncontrollability with respect to job routings, including the potential need for job rework at certain processing stages, relates to the structuring of the sequential logic defining the various process flows supported by the system, and it is fundamentally different from the uncontrollability with respect to the event timings, that was addressed in Section 3. More specifically, this type of uncontrollability characterizes the forced selection of a certain routing option due to the inherent process dynamics, and introduces new mechanisms giving rise to non-live behavior that *cannot* be interpreted through the concept of empty / deadly marked siphon [6], which has been the main cause of non-liveness in

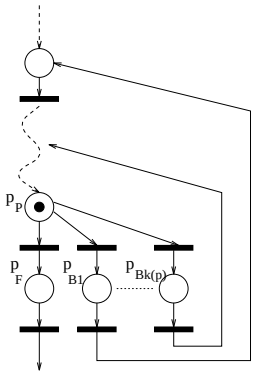

Fig. 2. Modeling uncontrollable job routings and reworks

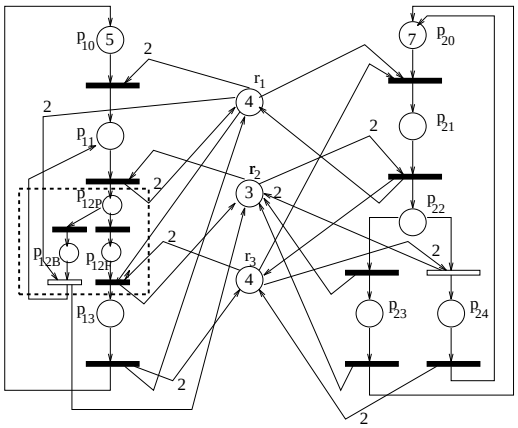

Fig. 3. Example 4: The S^3PGR^3 net

totally controlled networks. An additional, but minor complication, arises from the presence of internal cycles in the PN subnets modeling the system processes; these cycles are necessary to model repetitive processing due to rework. Yet, this section establishes that, in spite of the aforementioned modeling and analysis complications, for most practical applications, the modeling of uncontrollable behavior w.r.t. process routings and the associated rework requirements can be performed through a special PN structure, which when introduced in the original S^3PGR^2 net, allows the liveness enforcing supervision of the resulting system through an appropriately parameterized (U-)G-RUN LES.

The key idea underlying the proposed modeling approach is the separation of the internal process dynamics determining the (uncontrollable) routing of a certain job instance, and its potential need for rework, from the dynamics of the resource allocation concerning the job advancement among their various processing stages. In the PN modeling framework, this separation is implemented by modeling each process stage involving uncontrolled routing(s) of its processed outcomes, through the PN subnet depicted in Figure 2. In the depicted subnet, tokens in place p_P correspond to job instances still in execution of the considered processing stage; tokens in place p_F correspond to job instances that have completed successfully the considered stage and request transfer to the next one; finally, tokens in places $p_{B_l},\ l = 1, \ldots, k(p)$, correspond to job instances that failed the considered processing stage in a certain manner, and therefore, they request re-routing, possibly to one of the prior processing stages. Notice that under this extended modeling, the set of process places, P_S, is *partitioned* into three subsets P_P, P_F and P_B, and any "rework" loop involves only a *single* place in P_B. Furthermore, it should be obvious from the above discussion that for the subnet of Figure 2,

$$ {}^\bullet(p_P{}^\bullet \cap {}^\bullet p_F) \cap P_R = \emptyset \tag{31} $$

i.e., the advancement of a job instance from place p_P to p_F does not involve the allocation of any additional resources. Even though, under the aforementioned separation principle, a similar claim could be made for the transitions modeling the job advancement from place p_P to any of the places $p_{B_l},\ l = 1, \ldots, k(p)$, we still allow a non-zero resource allocation involved with these transitions, in order to model situations where a failing job instance might need some salvage and/or preparatory treatment before the actual rework stage. The resulting RAS class is characterized as *E(xtended)CD-RAS*, and the corresponding PN modeling sub-class, resulting from the S^3PGR^2 net through the aforementioned extensions/modifications, will be called the S^3PGR^3 net (*System of Simple Sequential Processes with General Resource Requirements and Reworks*). For the sake of brevity, we omit a formal characterization of S^3PGR^3 nets, and refer the reader to [7]. The next example elucidates the S^3PGR^3 net structure, by introducing the possibility for a rework requirement in the S^3PGR^2 net depicted in Figure 1.

Example 4 Suppose that process stage p_{12} in the S^3PGR^2 net of Figure 1 has a non-prefect yield, and the defective outcomes must be reworked, starting from stage p_{11}. This effect is modeled by substituting the process place p_{12} in Figure 1 with a subnet of the structure depicted in Figure 2, and with $k(p) = 1$. Furthermore, the resource allocation vectors associated with the places in this subnet are all equal to $[0, 1, 0]^T$, i.e., the original resource allocation vector corresponding to stage p_{12}. The resulting S^3PGR^3 net is depicted in Figure 3.

G-RUN LES for S^3PGR^3 nets The key observation underlying the application of the G-RUN logic for the liveness enforcing supervision of ECD-RAS and their modeling class of S^3PGR^3 nets, is that, in the S^3PGR^2 net context, the set of successor stages, $q \in p^{\bullet\bullet}$, that are *guaranteed* to be accessible from any given stage $p \in P_S$ with $p^{\bullet\bullet} \cap P_0 = \emptyset$, is determined by the employed policy community, Ψ, which constitutes one of the policy parameters.[3] Since, in the case of S^3PGR^3 nets, the selection of the successor stages for tokens in $p \in P_S$ with $p^{\bullet\bullet} \cap P_B \neq \emptyset$ is beyond the jurisdiction of the system controller, it must be ensured that the corresponding forced transitions are guaranteed by the policy realization. In the light of the previous remark, this is achieved simply by requiring that

$$ \begin{aligned} &\forall p \in P_S,\ \forall q \in p^{\bullet\bullet} \cap P_B,\quad (p, q) \in \Psi \\ &\forall p \in P_S,\ \exists q \notin p^{\bullet\bullet} \cap P_B,\quad (p, q) \in \Psi \end{aligned} \tag{32} $$

The above discussion is summarized in the next theorem.

Theorem 2: The controlled net $\mathcal{N} = (P_0 \cup P_S \cup P_R \cup P_W, T, W, M_0)$ corresponding to a G-RUN implementation on a well marked S^3PGR^3 net, which further observes the requirement of Equation 32, is live.

A formal proof for Theorem 2 is provided in [7]. Furthermore, in [7] it is also shown that the theory of Section 3, regarding the synthesis of LES for *U*-S^3PGR^2 nets, extends immediately to S^3PGR^3 nets with uncontrollable resource allocations – to be referred to as *U*-S^3PGR^3 nets – with the only modification being the addition of Constraint 32 in the original definition of the U-G-RUN LES. The next example employs this extended

[3] This remark is based on the detailed study of the role of the policy community in the proofs of its correctness, c.f. [7], [14], [6].

theory in order to compute a U-G-RUN LES for the $U\text{-}S^3PGR^3$ net of Figure 3.

Example 5 Consider that in the S^3PGR^3 net of Figure 3, $T_u = {p_{12B}}^\bullet \cup {}^\bullet p_{24}$. Then, implementation of U-G-RUN on this net, with resource ordering $o : (o_1 = 2,\ o_2 = 1,\ o_3 = 3)$ and community $\Psi = \{(p_{11}, p_{12}), (p_{12P}, p_{12F}), (p_{12P}, p_{12B}), (p_{12F}, p_{13}), (p_{12B}, p_{11}), (p_{21}, p_{22}), (p_{22}, p_{23})\}$, leads to the following set of constraints:

$$\begin{bmatrix} 2 & \mathbf{2} & 1 & \mathbf{2} & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 2 & 1 & 0 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 0 & 2 \end{bmatrix} \cdot M_S \quad \leq \quad \begin{bmatrix} 4 \\ 3 \\ 4 \end{bmatrix} \quad (33)$$

In Equation 33, the rows of matrix $\mathbf{A}$ correspond to the resource sequence $< R_1, R_2, R_3 >$, and its columns correspond to the place sequence $< p_{11}, p_{12P}, p_{12F}, p_{12B}, p_{13}, p_{21}, p_{22}, p_{23}, p_{24} >$. Juxtaposing the supervisor of Equation 33 with that of Equation 30, it can be seen that the primary effect of the introduced rework loop is to increase the effective requirement of stage p_{12} with respect to resource R_1 by one unit. This increase hedges against the possibility for re-execution of stage p_{11}, and the extra resource unit is released, once a positive process outcome is established.

V. Accommodating linear "forbidden state" constraints

The availability of a systematic procedure for the synthesis of an acceptable LES for any given $U\text{-}S^3PGR^2$ or $U\text{-}S^3PGR^3$ net, $\mathcal{N}$, allows also the integration of the liveness enforcing supervision problem with any additional logical constraints of the form

$$\mathbf{L} \cdot M_S \leq \mathbf{g} \quad (34)$$

where M_S is defined as in Equation 14. This capability results from the fact that an acceptable supervisor for enforcing Equation 34 to the original $U\text{-}S^3PGR^2$ (resp., $U\text{-}S^3PGR^3$) net is provided by the methodology of [9], [10], discussed in Section 2.1, while the resulting net, $\mathcal{N}'$, which includes the control places enforcing Equation 34, remains in the class of $U\text{-}S^3PGR^2$ (resp., $U\text{-}S^3PGR^3$) nets. Therefore, the liveness of $\mathcal{N}'$ can be established through a U-G-RUN LES, developed through the methodology of Section 3 (resp., 4). The detailed algorithm implementing this idea is as follows:

LES synthesis for (E)CD-RAS with additional forbidden states

1. Use the methodology of [9], [10] in order to (i) test the acceptability of each constraint $(\mathbf{L}_i, g_i),\ i = 1, \ldots, l\ (\equiv \dim(\mathbf{g}))$, and (ii) transform unacceptable constraints to acceptable. Let the resulting acceptable marking specification be denoted by $\mathbf{L}' \cdot M \leq \mathbf{g}'$.
2. Impose constraint $\mathbf{L}' \cdot M \leq \mathbf{g}'$ to the uncontrolled net $\mathcal{N}$ as a set of "logical resource" places, by applying Step I in Section 2.1. Let the resulting controlled net be denoted by $\mathcal{N}'$.
3. Obtain a G-RUN LES for net $\mathcal{N}'$, by solving the linear program defined by Constraints 25 - 28 and objective function 18, using a community Ψ that observes Equation 32.

Since G-RUN-LES are generally suboptimal, the superimposition of such a supervisor on net $\mathcal{N}'$ should be performed only in the case that net $\mathcal{N}'$ is not live. However, while a methodology for testing the liveness of $(U\text{-})S^3PGR^2$ nets can be found in [6], testing the liveness of a $(U\text{-})S^3PGR^3$ net is an open research issue.

VI. Conclusions

Starting from the observation that all existing results on liveness enforcing supervision for sequential RAS have addressed the problem under the assumption of total controllability of the system event set, the work presented in this paper modeled CD-RAS with uncontrollable behavior, and modified the defining logic of the G-RUN LES, which provides correct and scaleable policies appropriate for the class of CD-RAS, in a way that the resulting supervisors will also respect the potential system uncontrollability. The last part of the paper discussed the integration of the U-G-RUN logic with the PN-based supervisory control methodology developed in [9], [10], in order to systematically address the synthesis of LES for (E)CD-RAS, under the imposition of additional logical constraints on the RAS behavior, that constitute "forbidden state" requirements, linearly expressed in the system resource allocation state vector. Future work will seek to extend these results to broader RAS classes, e.g., in RAS with synchronizing transitions modeling assembly/disassembly operations.

References

[1] M. A. Lawley and S. A. Reveliotis, "Deadlock avoidance for sequential resource allocation systems: hard and easy cases," *Intl. Jrnl of FMS*, vol. 13, pp. 385–404, 2001.

[2] Z. A. Banaszak and B. H. Krogh, "Deadlock avoidance in flexible manufacturing systems with concurrently competing process flows," *IEEE Transactions on Robotics & Automation*, vol. 6, no. 6, pp. 724–734, 1990.

[3] S. A. Reveliotis and P. M. Ferreira, "Deadlock avoidance policies for automated manufacturing cells," *IEEE Transactions on Robotics & Automation*, vol. 12, no. 6, pp. 845–857, 1996.

[4] M. P. Fanti, B. Maione, S. Mascolo, and B. Turchiano, "Event-based feedback control for deadlock avoidance in flexible production systems," *IEEE Transactions on Robotics & Automation*, vol. 13, pp. 347–363, 1997.

[5] M. Lawley, S. Reveliotis, and P. Ferreira, "The application and evaluation of banker's algorithm for deadlock-free buffer space allocation in flexible manufacturing systems," *International Journal of Flexible Manufacturing Systems*, vol. 10, no. 1, pp. 73–100, 1998.

[6] J. Park and S. A. Reveliotis, "Deadlock avoidance in sequential resource allocation systems with multiple resource acquisitions and flexible routings," *IEEE Trans. on Automatic Control*, vol. 46, pp. 1572–1583, 2001.

[7] J. Park, *Structural Analysis and Control of Resource Allocation Systems using Petri nets*, Ph.D. thesis, Georgia Institute of Technology, Atlanta, GA, 2000.

[8] J. Park and S. Reveliotis, "Algebraic deadlock avoidance policies for conjunctive/disjunctive resource allocation systems," in *Proc. of ICRA'01*. IEEE, 2001.

[9] J. O. Moody and P. J. Antsaklis, *Supervisory Control of Discrete Event Systems Using Petri Nets*, Kluwer Academic Publishers, 1998.

[10] J. O. Moody and P. J. Antsaklis, "Petri net supervisors for des with uncontrollable and unobservable transitions," *IEEE Trasactions on Automatic Control*, vol. 45, no. 3, pp. 462–476, 2000.

[11] J. Desel and J. Esparza, *Free Choice Petri Nets*, Cambridge Univerrsity Press, 1995.

[12] A. Giua, F. DiCesare, and M. Silva, "Generalized mutual exclusion constraints on nets with uncontrollable transitions," in *Proceedings of the 1992 IEEE Intl. Conference on Systems, Man and Cybernetics*. IEEE, 1992, pp. 974–979.

[13] J. Ezpeleta, J. M. Colom, and J. Martinez, "A petri net based deadlock prevention policy for flexible manufacturing systems," *IEEE Transactions on Robotics & Automation*, vol. 11, pp. 173–184, 1995.

[14] J. Park and S. A. Reveliotis, "A polynomial-complexity deadlock avoidance policy for sequential resource allocation systems with multiple resource acquisitions and flexible routings," in *Proceedings of the IEEE International Conference on Decision & Control*. IEEE, 2000.

[15] F. Tricas, F. García-Vallés, J. M. Colom, and J. Ezpeleta, "An iterative method for deadlock prevention in fms," in *Proceedings of the 5th Workshop on Discrete Event Systems*, 2000.

[16] J. Park and S. Reveliotis, "Algebraic synthesis of efficient deadlock avoidance policies for sequential resource allocation systems," *IEEE Trans. on R&A*, vol. 16, pp. 190–195, 2000.

[17] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*, Kluwer Academic Publishers, 1999.

[18] Y. Li and W. M. Wonham, "Deadlock issues in supervisory control of disrete event systems," in *Proc. Conf. Inf. Sci. Syst.*, 1988, pp. 57–63.

[19] P. J. G. Ramadge and W. M. Wonham, "The control of discrete event systems," *Proceedings of the IEEE*, vol. 77, pp. 81–98, 1989.

[20] M. R. Garey and D. S. Johnson, *Computers and Intractability : A Guide to the Theory of NP-Completeness*, W. H. Freeman, New York, 1979.