Algebraic Synthesis of Efficient Deadlock Avoidance Policies for Sequential Resource Allocation Systems

Jonghun Park and Spyros A. Reveliotis
School of Industrial & Systems Engineering
Georgia Institute of Technology
Atlanta, GA 30332

Abstract

Deadlock avoidance in sequential resource allocation systems is a well-defined problem in Discrete Event System literature, as it underlies the operation of many contemporary technological systems. In the past, the problem has been studied by means of a number of formal frameworks, including the finite state automata (FSA) and Petri nets (PN). In this paper, it is shown that a significant class of deadlock avoidance policies (DAP), known as algebraic PK-DAP's, originally developed in the FSA paradigm, can be analyzed using recent results from PN structural analysis. Furthermore, the approach to DAP analysis and design taken in this paper has led to the effective generalization of the currently available algebraic PK-DAP's, and to their enrichment with new and more flexible policy implementations.

1 Introduction

Deadlock avoidance is an essential control requirement for automated operation of modern technological systems that involve resource sharing. In these environments, the competition of concurrently executing processes for a finite set of non-preemptible resources gives rise to conflicting situations – characterized as *deadlocks* – where there exists a subset of processes permanently blocked, due to the fact that each process in the set requires, for its further continuation, some resource(s) currently held by some other process in the set. Hence, the underlying (sequential) resource allocation must be carefully managed in order to ensure deadlock-free operation. Furthermore, given the increasing emphasis on the operational flexibility of these systems, the objective is to avoid deadlocks in the *least restrictive* manner, a requirement that establishes a notion of *optimality* in the considered problem context. However, in the general case, the computation of the optimal deadlock avoidance policy (DAP) is proven to be an $\mathcal{NP}$-hard problem [9]. Therefore, past research results have mainly focused on (i) the identification of *special sequential resource allocation system (RAS) structure* that admits polynomial-time optimal deadlock avoidance policy [13, 6, 10], or (ii) the design of suboptimal but *polynomially computable*[1] policies, which, when imposed on the system operation, guarantee deadlock-free behavior [1, 6, 7, 8, 9]. The resulting policies are characterized as *Polynomial Kernel* (PK)-DAP's, and the PK-DAP's for which the policy-defining condition can be expressed as a set of algebraic constraints on the underlying RAS state, are further referred to as *algebraic*. For example, the policies developed in [1, 9, 8] and some of the policies in [6] fall into the class of algebraic PK-DAP's, while the policy presented in [7] constitutes an example of an efficient but non-algebraic PK-DAP.

Petri net (PN) structural analysis has been the other major line of research on the deadlock avoidance problem in sequential RAS. In this framework, deadlock-free operation is formally expressed by the PN properties of *liveness* and *reversibility* [4]. Some representative work along this line of research can be found in, e.g. [14, 5]. Recently, it was established in [3] that (i) for a PN subclass, known as *Augmented Marked Graphs* (AMG), the absence of liveness and reversibility properties can be directly related to the potential development of a certain structure, known as *empty siphon*, and, (ii) for structurally bounded AMG's, the absence of potentially empty siphons can be tested by solving a *mixed integer program* (MIP) [12]. Motivated by these results, in this paper, we establish that the AMG class can effectively model the *controlled* behavior of a significant subclass of sequential RAS under algebraic PK-DAP control, and therefore, the results in [3] provide a novel framework for analyzing the correctness of any tentative algebraic PK-DAP's proposed for these systems. Furthermore, we show that this framework leads to the effective generalization of the currently known algebraic PK-DAP's, and present a method to enhance the flexibility supported by these policies. More specifically, the paper is organized as follows : Section 2 provides the necessary background on the aforestated AMG structural analysis through the MIP formulation as well as the theory of algebraic PK-DAP's for sequential RAS. Section 3 establishes that the AMG-based structural analysis provides a new formal framework for the correctness analysis and flexibility enhancement of existing PK-DAP's for a significant class of sequential RAS. The proposed method is demonstrated by an example. Finally Section 4 concludes the paper and suggests further research directions.

[1] We clarify that this complexity requirement concerns primarily the policy *execution time* rather than the policy development, since this is the control stage that must be supported in *real-time*.

2 Preliminaries

2.1 Sequential RAS and Algebraic PK-DAP's

A sequential RAS is defined by a set of resource types $\mathcal{R} = \{R_i, i = 1, \ldots, m\}$, and a set of process types $\mathcal{J} = \{JT_j, j = 1, \ldots, n\}$. Every resource R_i is further characterized by its *capacity* C_i, i.e., a finite positive integer indicating how many units of resource type R_i the RAS possesses. Process type JT_j is defined by a sequence of m-dimensional vectors $< JT_{jk}, k = 1, \ldots, l(j) >$, with component $JT_{jk}[v]$, $v = 1, \ldots, m$, indicating how many units of resource R_v are required for the execution of the k-th step. Furthermore, we shall impose the restriction that vectors JT_{jk} are elementary unit vectors, i.e., each stage requires only one unit from a single resource type for its operation. This subclass of sequential RAS has been known as *Single-Unit* (SU)-RAS [10]. The RAS state is determined by the current processing stages of all active process instances. Formally, it is denoted as a vector $\mathbf{q}(t)$, of dimension $D = \sum_{j=1}^{n} l(j)$ – i.e., equal to the total number of distinct route stages supported by the system configuration – and with component $\mathbf{q}_i(t)$, $i = 1, \ldots, D$, being equal to the number of processes executing step k of process type JT_u at time t, where u is the largest integer such that $i > \sum_{j=1}^{u-1} l(j)$ and $k = i - \sum_{j=1}^{u-1} l(j)$. Since each resource possesses finite buffering capacity, the set of distinct RAS states is finite, and it is denoted by $Q = \{\mathbf{q}^i, i = 0, \ldots, Z\}$, where $\mathbf{q}^0$ represents the initial empty state. As shown in [9], the state space of the RAS can be formally modeled by an FSA, and therefore, in order to avoid deadlock in the least restrictive way, the system operation should be confined to the maximal communicating class that contains $\mathbf{q}^0$. Let us denote this class by Q_s. Then every state $\mathbf{q} \in Q_s$ is characterized as *safe* while states $\mathbf{q}$ belonging to the complement of Q_s, denoted by Q_u, are characterized as *unsafe*. Hence the objective is to prevent the RAS from entering the unsafe region of its state space, Q_u. However, in general, the property $\mathcal{P}$ corresponding to the definition of the forbidden set of unsafe states Q_u is broader than the definition of deadlock states, and the evaluation $\mathcal{P}(\mathbf{q})$ to decide whether $\mathbf{q} \in Q_u$ is $\mathcal{NP}$-complete [9]. As a result, [9] proposes the notion of PK-DAP's which confine the system operation to a polynomially identifiable subset of its safe space. This idea requires the identification of a property $\mathcal{H}(\mathbf{q}), \mathbf{q} \in Q$, such that : (i) $\mathcal{H}(\mathbf{q}) \Rightarrow \neg\mathcal{P}(\mathbf{q}), \forall \mathbf{q} \in Q$, and (ii) $\mathcal{H}(\cdot)$ is polynomially computable on the given system state. Then, by allowing only transitions to states satisfying $\mathcal{H}(\cdot)$, through *one-step look-ahead*, it can be ensured that the visited states will also satisfy the condition $\neg\mathcal{P}$, and therefore the system will never enter the forbidden region Q_u. In addition, the subspace $Q(\mathcal{H})$ admitted by the resulting policy should be a single communicating class containing $\mathbf{q}^0$ in order to be *induced* deadlock-free. Among the PK-DAP's developed in the past, of particular interest to this research is a class of algebraic PK-DAP's in which $\mathcal{H}(\cdot)$ is represented as a *polynomially-sized* set of linear inequalities in the state vector :

$$\mathbf{A}_{\mathcal{P}} \cdot \mathbf{q} \leq \mathbf{f}_{\mathcal{P}} \tag{1}$$

$\mathbf{A}_{\mathcal{P}}$ is an incidence matrix. Each row of $\mathbf{A}_{\mathcal{P}}$ can be associated with the subset of process stages $JT^{(i)} = \{JT_{jk} \mid \mathbf{A}_{[i,(j,k)]} = 1\}$. Furthermore, the product $\mathbf{A}_{[i,\cdot]} \cdot \mathbf{q}$ counts the number of processes in state $\mathbf{q}$ which execute stages in $JT^{(i)}$. Therefore, an algebraic policy $\mathcal{P}$ can be expressed by the condition that a RAS state $\mathbf{q}$ is admissible *iff* for every process stage subset $JT^{(i)}$, the number of processes in state $\mathbf{q}$ executing stages in $JT^{(i)}$ does not exceed the policy-defined bound $\mathbf{f}_{\mathcal{P}}[i]$. Apparently, $\mathbf{f}_{\mathcal{P}}[i] \geq 1, \forall i$. As a concrete example, consider the *Resource Upstream Neighborhood* (RUN) policy, presented in [9]. RUN requires that a state is admissible *iff* the number of jobs in the upstream neighborhood of each resource R_i, denoted as $NH(R_i)$, does not exceed its buffering capacity C_i $(= \mathbf{f}_{\mathcal{P}}[i])$. $NH(R_i)$ is defined as follows :

Definition 1 $NH(R_i) = \{JT_{jk} \mid R(JT_{jk}) = R_i\} \cup \{JT_{j,k-q} \mid JT_{j,k-q+1} \in NH(R_i) \wedge O(R(JT_{j,k-q})) \leq O(R_i)\}$, $j = 1, \ldots, n$, $k = 1, \ldots, l(j)$, $q = 1, \ldots, k-1$, *where* $R(JT_{jk})$ *denotes the resource supporting the execution of stage* JT_{jk}, *and* $O(\cdot) : \mathcal{R} \to \{1, \ldots, m\}$ *is any partial ordering defined on* $\mathcal{R}$.

Example 1 Consider a simple SU-RAS, consisting of three resource types, R_1, R_2, and R_3, with capacities $C_1 = C_3 = 1$, and $C_2 = 2$. This RAS supports two process types JT_1 and JT_2, with the corresponding process routes $R_1 \to R_2 \to R_3$ and $R_3 \to R_2 \to R_1$. Hence, the system state is represented by a six-dimensional vector $\mathbf{q} = (q_{11}, q_{12}, q_{13}, q_{21}, q_{22}, q_{23})^T$, where component q_{ij} indicates the number of job instances executing stage JT_{ij}, $i = 1, 2$, $j = 1, 2, 3$. The implementation of RUN DAP on this system, based on the capacity-induced resource ordering: $O(R_1) = 1$, $O(R_2) = 2$, and $O(R_3) = 1$, imposes the following set of constraints on the system state (cf., Definition 1 and the aforestated RUN control logic):

$$\begin{bmatrix} 1 & & & & & 1 \\ 1 & 1 & & 1 & 1 & \\ & & 1 & 1 & & \end{bmatrix} \cdot \mathbf{q} \leq \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \tag{2}$$

Hence, state $\mathbf{q} = (0, 0, 0, 1, 0, 1)^T$ is a safe state admitted by the RUN implementation of 2, whereas state $\mathbf{q} = (1, 2, 0, 0, 0, 0)^T$ is policy non-admissible (even though it is indeed a safe state). $\diamond$

2.2 AMG Structural Analysis

This section introduces the Petri net (PN) class of augmented marked graphs (AMG), and reviews the key results of AMG structural analysis, developed in [3], that are employed in the DAP analysis framework of Section 3. The subsequent discussion assumes that the reader is familiar with the basic PN behavioral and structural concepts. A brief review of the elements of ordinary PN theory employed in this work can be found in the Appendix.

Definition 2 *[3] An* augmented marked graph (AMG) *is an ordinary PN* $G = < P \cup R, T, F, M_0 >$ *composed of two sets of places,* P *and* R*, a set of transitions* T*, and satisfying the following conditions :*
(i) The net G' *obtained by removing places in* R *is a marked graph.*
(ii) To each place $r \in R$ *are associated* $k_r > 1$ *pairs of transitions denoted by* $D^r = \{(a_{r1}, b_{r1}) \ldots (a_{rk_r}, b_{rk_r})\}$*, where* $a_{ri}, b_{ri} \in T, \forall i \in N_r = \{1 \ldots k_r\}$*. Furthermore,* $r^\bullet = \{a_{r1} \ldots a_{rk_r}\}$*,* $^\bullet r = \{b_{r1} \ldots b_{rk_r}\}$*,* $a_{ri} \neq a_{rj}$ *and* $b_{ri} \neq b_{rj}$*,* $\forall i \neq j \in N_r$*, and there exists a simple path* O_{ri} *in* G' *connecting* a_{ri} *to* b_{ri} *if* $a_{ri} \neq b_{ri}$*.*
(iii) Each elementary circuit in G' *is marked by* M_0*.*
(iv) Every place in R *is marked by* M_0 *and no path* O_{ri} *is marked at* M_0*.*

We also summarize some of the AMG properties established in [3] in the following proposition.

Proposition 1 *[3] Given an AMG* $G = < P \cup R, T, F, M_0 >$*, the following statements hold :*
(i) Each resource place $r \in R$*, together with places in paths* O_{ri}*, forms a p-invariant, i.e.,*

$$\forall M \in R(M_0): \quad M(r) + \sum_{i \in N_r} \sum_{p \in O_{ri}} M(p) = M_0(r) + \sum_{i \in N_r} \sum_{p \in O_{ri}} M_0(p)$$

(ii) G *is reversible if it is live.*
(iii) G *is live iff it does not contain any potential deadlock.*
(iv) Let S *denote a (minimal) siphon of* G*. Also, define* $F(S) \equiv min\{M(S) \mid M = M_0 + \Theta Y, M \geq 0, Y \geq 0\}$*, where* $M(S) \equiv \sum_{q \in S} M(q)$*, and* Θ *denotes the net incidence matrix. Then,* G *is live if each minimal siphon,* S*, containing at least one resource place, either contains a marked trap or has* $F(S) > 0$*.*

The requirement $F(S) > 0$ in the above proposition, is essentially a sufficient condition ensuring that siphon S is not a potential deadlock. However, its power as an analysis tool for AMG liveness and reversibility is limited by the fact that it requires the explicit enumeration of all minimal siphons of the net structure, a task of non-polynomial complexity with respect to the number of net places. Fortunately, for the class of structurally bounded nets, [3] provides an additional sufficient condition of potential deadlock-freedom that takes the convenient form of a MIP formulation of polynomial size with respect to the net elements. This result is stated in the following proposition :

Proposition 2 *[3] For a structurally bounded net* $N = (P, T, F, M_0)$*, consider the following MIP formulation :*

$$\mathcal{C}(N) = min \sum_{p \in P} v_p \tag{3}$$

s.t.

$$z_t \geq \sum_{p \in {}^\bullet t} v_p - |{}^\bullet t| + 1, \ \forall t \in T \tag{4}$$

$$v_p \geq z_t, \ \forall (t,p) \in F \tag{5}$$

$$v_p \geq \frac{M(p)}{SB(p)}, \ \forall p \in P \tag{6}$$

$$M = M_0 + \Theta \bar{y} \tag{7}$$

$$v_p, z_t \in \{0,1\}, \ \forall p \in P, \ \forall t \in T \tag{8}$$

$$M \geq 0, \ \bar{y} \geq 0 \tag{9}$$

Then, N will have no potential deadlocks if the optimal solution of the above MIP formulation satisfies $\mathcal{C}^*(N) = |P|$.

In the above formulation, v_p is an indicator variable associated with net place p, taking the value of 1 iff place p cannot be part of a reachable empty siphon. Similarly, z_t is an indicator variable associated with transition t, taking the value of 1 iff transition t cannot have an input place belonging to an empty siphon. $SB(p)$ denotes an (a priori established) structural bound for the markings of place p. M and $\bar{y}$ constitute a feasible solution to (the linear relaxation of) the net state equation. From Propositions 1 and 2, it is clear that the following corollary holds.

Corollary 1 *A sufficient condition for a structurally bounded AMG $G = (P \cup R, T, F, M_0)$ to be live and reversible, is that the MIP formulation of the Proposition 2 has an optimal objective value $\mathcal{C}^*(G) = |P| + |R|$.*

3 AMG-based Correctness Analysis of Algebraic PK-DAP's

In this section, first we formally establish that the operation of SU-RAS, controlled under algebraic PK-DAP's, can be effectively modeled as an AMG. Then, this finding leads (i) to a novel correctness analysis framework for algebraic PK-DAP's proposed for SU-RAS, and (ii) to more flexible implementations of the available algebraic PK-DAP's, which are not recognized (to be correct) by the original policy definitions. The enhanced design capabilities of the new framework are demonstrated by application to the logic of RUN DAP [9].

3.1 AMG-based Modeling of SU-RAS Controlled by Algebraic PK-DAP's

For the purpose of the subsequent analysis, we adopt the S^3PR model of [5], appropriately restricted to eliminate the feature of dynamic flexible routings; the resulting PN class is called S^2LSPR : *Systems of Simple Linear Sequential Processes with Resources.* First, we introduce the PN structure of *Simple Linear Sequential Process (SLSP)*, where $SLSP_j =< P_j, T_j, F_j, M_0^{(j)} >$ models the sequential logic of the SU-RAS process $JT_j, j = 1, \ldots, n$. The initial marking $M_0^{(j)}$ of $SLSP_j$ constitutes the projection of the initial marking, M_0, for the complete net model, over the place set P_j, and therefore, to simplify notation, the corresponding superscripts are omitted in the following discussion. Furthermore, the linear structure of process type JT_j, with $l(j)$ process stages, implies that $N_j =< P_j, T_j, F_j >$ forms a simple path $< t_{j0}, p_{j1}, t_{j1}, p_{j2}, \ldots, t_{j,l(j)-1}, p_{j,l(j)}, t_{j,l(j)} >$. In this structure, the firing of transition t_{j0} (resp. $t_{j,l(j)}$) corresponds to the loading (resp. unloading) of a job instance of type JT_j into (resp. from) the system. Similarly, the firing of transition $t_{jk}, k = 1, \ldots, l(j) - 1$ models the advancement of a job instance from stage JT_{jk} to stage $JT_{j,k+1}$. Hence, in the proposed model, the marking of place p_{jk}, $M(p_{jk})$, denotes the number of job instances executing stage JT_{jk}. Clearly, $M_0(p_{jk}) = 0, \forall j, k = 1, \ldots, l(j)$. Finally, we introduce an additional place p_{j0}, with $p_{j0}^\bullet = \{t_{j0}\}$ and ${}^\bullet p_{j0} = \{t_{j,l(j)}\}$; this place is characterized as the process *idle place* [5], and its introduction converts each N_j to a *circuit.* Note that the initial marking of p_{j0}, $M_0(p_{j0})$, essentially controls the maximum number of job instances of type JT_j that can be simultaneously loaded in the system. A natural upper bound to this number is provided by $\sum_{i \in I} C_i$, where $I = \{i \mid \exists k \text{ s.t. } R(JT_{jk}) = R_i\}$. Having defined the PN structure modeling the sequential logic underlying the process types of the SU-RAS class, we proceed with the modeling of the resource allocation taking place in these systems, and the competition of the concurrently executing processes for the limited system resources. Specifically, the limited resource availabilities are modeled by the introduction of a place, r_i, for each resource type $R_i, i = 1, \ldots, m$, with initial marking, $M_0(r_i) = C_i, i = 1, \ldots, m$. Let $R = \{r_i, i = 1, \ldots, m\}$. Allocation (resp. release) of resource $R_i, i = 1, \ldots, m$, to (resp. by) the executing processes is modeled by setting $r_i^\bullet$ (resp. ${}^\bullet r_i$) $= \{t_{j,k-1}(\text{resp. } t_{jk}) \mid \forall j, k \text{ s.t. } R(JT_{jk}) = R_i\}$. The single-unit nature of the considered resource allocation implies that : (i) $\forall j, k = 0, \ldots, l(j) - 1$, $|{}^\bullet t_{jk} \cap R| = 1$, (ii) $\forall j, {}^\bullet t_{j,l(j)} \cap R = \emptyset$, (iii) $\forall j, k = 1, \ldots, l(j)$, $|t_{jk}^\bullet \cap R| = 1$, (iv) $\forall j, t_{j0}^\bullet \cap R = \emptyset$. The resulting PN is characterized as S^2LSPR : *System of Simple Linear Sequential Processes with Resources.* The policy constraints imposed by an algebraic PK-DAP, defined according to the scheme of Equation 1, are modeled in the proposed PN representation by the introduction of additional control places, $w_i \in W$. Place w_i models the i-th constraint in Equation 1. Specifically, $M_0(w_i) = \mathbf{f}_{\mathcal{P}}[\text{i}]\ (\geq 1)$, $w_i^\bullet = \{t_{jk} \mid \forall j, k \text{ s.t. } JT_{jk} \notin JT^{(i)} \wedge JT_{j,k+1} \in JT^{(i)}\}$, and ${}^\bullet w_i = \{t_{jk} \mid \forall j, k \text{ s.t. } JT_{jk} \in JT^{(i)} \wedge JT_{j,k+1} \notin JT^{(i)}\}$. The resulting PN structure is

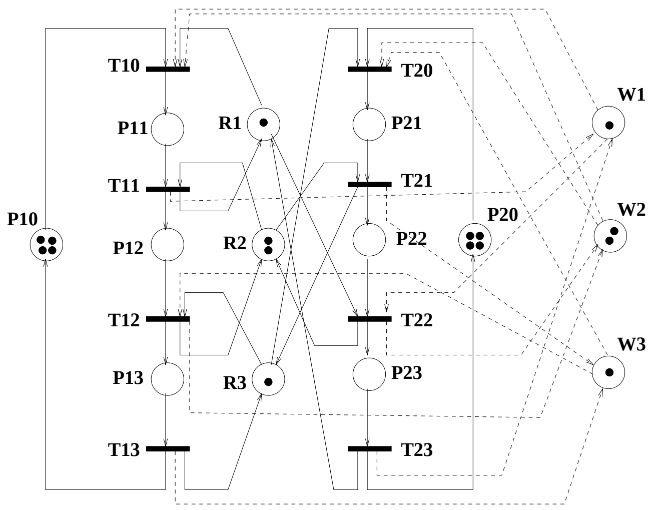

Figure 1: CS^2LSPR for Example 1

characterized as CS^2LSPR : *Controlled System of Simple Linear Sequential Processes with Resources.*

Figure 1 shows the CS^2LSPR constructed for Example 1. Notice that in the PN structure of CS^2LSPR, the control places $w_i \in W$, implementing algebraic PK-DAP's for SU-RAS, act as additional resources allocated and released during the execution of the process stages. Under this interpretation, it is easy to see that the class of CS^2LSPR is an AMG subclass. We state this as a proposition.

Proposition 3 *The PN class of CS^2LSPR, modeling the* controlled *behavior of SU-RAS under algebraic PK-DAP's, is a subclass of AMGs.*

3.2 AMG-based Correctness Analysis of Algebraic PK-DAP's

Proposition 3, when combined with the results of Proposition 1, suggests a novel framework for evaluating the correctness of any tentative algebraic PK-DAP proposed for SU-RAS. This framework is formalized in the following theorem:

Theorem 1 *Let $G =< P \cup R \cup W, T, F, M_0 >$ be a PN belonging to the class of CS^2LSPR. Then, G is live and reversible, if every siphon S containing* two or more *places in $R \cup W$, has $\sum_{q \in S} M(q) > 0$, $\forall M \geq 0$ s.t. $\exists Y \geq 0$ with $M = M_0 + \Theta Y$.*

Proof: First, consider the set $\mathcal{S}_1 \equiv \{S : \ S \subseteq P \ \wedge \ {}^\bullet S \subseteq S^\bullet\}$. Then, the definition of the PN class CS^2LSPR in Section 3.1 implies that every element $S \in \mathcal{S}_1$ is essentially defined by a subset of $SLSP_j$ subnets. Furthermore, it is easy to see that each $SLSP_j$ net, $j = 1, \ldots, n$, defines a p-invariant:

$$\forall j \in \{1, \ldots, n\} : \ \sum_{k=0}^{l(j)} M(p_{jk}) = M_0(p_{j0}) \tag{10}$$

Since, by construction, each $SLSP_j$ net has $M_0(p_{j0}) > 0$, it follows that all siphons $S \in \mathcal{S}_1$ do not constitute potential deadlocks.

Next, consider the set $\mathcal{S}_2 \equiv \{S \subseteq P \cup R \cup W \;:\; |S \cap (R \cup W)| = 1 \;\wedge\; {}^\bullet S \subseteq S^\bullet\}$. Let $S \cap (R \cup W) = \{q\}$. Then, the definition of CS^2LSPR in Section 3.1 implies that $S = \{q\} \cup (\bigcup_{i \in N_q} O_{qi})$. This remark, together with Propositions 3 and 1(i), further imply that for all $S \in \mathcal{S}_2$,

$$\forall M \in R(M_0): \quad M(q) + \sum_{i \in N_q} \sum_{p \in O_{qi}} M(p) = M_0(q) \tag{11}$$

where q is defined as above. Given that, by construction, $M_0(q) > 0$, $\forall q \in R \cup W$, it follows that siphons $S \in \mathcal{S}_2$ do not constitute potential deadlocks.

Finally, the truth of Theorem 1 is established by combining the above remarks regarding the elements of $\mathcal{S}_1$ and $\mathcal{S}_2$, with Propositions 3 and 1(ii–iv). $\diamond$

From a methodological perspective, an effective way to apply the criterion of Theorem 1 while establishing the correctness of a given algebraic PK-DAP for SU-RAS, is by showing that, otherwise, the existence of a siphon S with two or more places in $R \cup W$, and with $\sum_{q \in S} M(q) = 0$ for some $M = M_0 + \Theta Y$, $M \geq 0$, $Y \geq 0$, will violate at least one of the p-invariants expressed by Equation 11. Next, we demonstrate the applicability of this technique, by (re-)establishing the correctness of RUN DAP, initially proven in [9].

Re-establishing the correctness of RUN DAP by means of Theorem 1 According to the result of Theorem 1, it is sufficient to prove the following theorem:

Theorem 2 *Let $G =< P \cup R \cup W, T, F, M_0 >$ denote the CS^2LSPR net modeling the controlled behavior of an SU-RAS structurally controlled by RUN DAP. Then, for every siphon S containing two or more places in $R \cup W$, $\sum_{q \in S} M(q) > 0$, $\forall M$ s.t. $\exists Y$ with $M = M_0 + \Theta Y$, $M \geq 0$, $Y \geq 0$.*

Proof: We prove this result by contradiction. Suppose that there exist nonnegative vectors M and Y, s.t. $M = M_0 + \Theta Y$, and a (maximal) siphon S of G, containing at least two places in $R \cup W$, with $\sum_{q \in S} M(q) = 0$. Consider a place $q \in S \cap (R \cup W)$. Then, the corresponding p-invariant expressed by Equation 11, together with the CS^2LSPR structure defined in Section 3.1, imply that $\exists p \in \bigcup_{i \in N_q} O_{qi}$ with $M(p) > 0$ and $p^\bullet$ being a dead transition. This further implies the existence of another place $q' \in {}^\bullet(p^\bullet) \cap (R \cup W) \cap S$. Repeating the above argument on place q', and considering the finiteness of the set $R \cup W$, we conclude that there exists an ordered set $< q_1, q_2, \ldots, q_k > \subseteq S \cap (R \cup W)$, and a corresponding set $< p_1, p_2, \ldots, p_k > \subseteq P$, s.t. place p_i, $i = 1, \ldots, k (\geq 2)$, is a marked process place, belonging to the p-invariant of Equation 11 associated with with "resource" place q_i; furthermore, transition $p_i^\bullet$ cannot fire because of the emptiness of place q_{i+1}, $i = 1, \ldots, k$, and transition $p_k^\bullet$ cannot fire because of the emptiness of place q_1.

Next consider a place $q_{i^*} \in < q_1, q_2, \ldots, q_k >$ with $O(q_{i^*}) = \min_{i=1,\ldots,k} O(q_i)$, where $O() : R \cup W \rightarrow \{1, \ldots m\}$ denotes the ordering used for the definition of the resource neighborhoods, during the implementation the RUN DAP, extended to set W by defining $O(w_i) = O(r_i),\ i = 1.\ldots, m$. Let $\{r_l\} = {}^\bullet(p_{i^*}^\bullet) \cap R$. Also, let w_l denote the control place imposing the RUN constraint for the neighborhood of resource r_l. The definition of q_{i^*} implies that

$$p_{i^*} \in \bigcup_{j \in N_{w_l}} O_{w_l j} \ \wedge \ p_{i^*} \notin \bigcup_{j \in N_{r_l}} O_{r_l j} \tag{12}$$

Also, Definition 1 implies that

$$\bigcup_{j \in N_{r_l}} O_{r_l j} \subseteq \bigcup_{j \in N_{w_l}} O_{w_l j} \tag{13}$$

and

$$q_{i^*+1} \equiv r_l \tag{14}$$

Furthermore, $M(q_{i^*+1}) = 0$, combined with Equations 11 and 14, imply that

$$\sum_{j \in N_{r_l}} \sum_{q \in O_{r_l j}} M(q) = C_l \tag{15}$$

But then, Equations 12, 13 and 15, combined with the fact that $M(p_{i^*}) > 0$, imply that

$$\sum_{j \in N_{w_l}} \sum_{q \in O_{w_l j}} M(q) > C_l \tag{16}$$

which violates the p-invariant defined by Equation 11 for control place w_l. $\diamond$

3.3 Enhancing the Flexibility of Algebraic PK-DAP's

Next, we consider how the results established in Corollary 1 and Proposition 3 facilitate the flexibility enhancement of existing algebraic PK-DAP's for SU-RAS. The new policy implementations are obtained by (i) relaxing the right-hand-side (rhs) of Equation 1, and (ii) evaluating the criterion of Corollary 1 on the AMG resulting by the relaxed policy definition. The new policy is correct if the liveness and reversibility criterion of Corollary 1 is satisfied. More specifically, given an implementation of an algebraic PK-DAP, $\mathcal{P}$, on a certain SU-RAS configuration, $\mathcal{S}$, the policy generalization method developed in this paper is essentially a search over the space of *meaningful* rhs vectors, $\mathbf{f}_\mathcal{P}$, for *maximal* elements that lead to correct policy implementations. The space of meaningful rhs vectors is defined as follows.

Definition 3 *Let $\mathbf{A}_\mathcal{P} \cdot \mathbf{q} \leq \mathbf{f}_\mathcal{P}$ represent the constraints imposed by an algebraic PK-DAP, $\mathcal{P}$, on the resource allocation state of a given SU-RAS configuration, $\mathcal{S}$. Then, the set $\mathcal{F}(\mathcal{P}; \mathcal{S})$, of the meaningful rhs vectors, is defined as*

$$\mathcal{F}(\mathcal{P}; \mathcal{S}) = \{\mathbf{f} \in Z^{dim(\mathbf{f}_\mathcal{P})} | \mathbf{1} \leq \mathbf{f} \leq \bar{\mathbf{f}}\} \tag{17}$$

where $\forall i$, $\bar{\mathbf{f}}[i] = \sum_{\{l \ |\exists JT_{jk} \in JT^i \ with \ R(JT_{jk})=R_l\}} C_l$, *and the relation* $\leq$ *is defined such that, given two vectors* $\mathbf{f}$ *and* $\mathbf{f}'$, $\forall i$, $\mathbf{f} \leq \mathbf{f}' \iff \mathbf{f}[i] \leq \mathbf{f}'[i]$.

The set $\mathcal{F}(\mathcal{P};\mathcal{S})$, armed with the relation $\leq$, constitutes a finite *partially ordered set*. Furthermore, every maximal totally ordered subset $\hat{\mathcal{F}} \subseteq \mathcal{F}(\mathcal{P};\mathcal{S})$, constitutes a *chain* between the extreme values of $\mathbf{1}$ and $\bar{\mathbf{f}}$. Let $\mathcal{F}_c(\mathcal{P};\mathcal{S})$ denote the subset of $\mathcal{F}(\mathcal{P};\mathcal{S})$ consisting of all meaningful vectors that lead to implementations of policy $\mathcal{P}$ satisfying the criterion of Corollary 1. Note that $\mathcal{F}_c(\mathcal{P};\mathcal{S}) \neq \emptyset$ for a correct algebraic PK-DAP $\mathcal{P}$. A simple algorithm for identifying the *maximal* elements of $\mathcal{F}_c(\mathcal{P};\mathcal{S})$ can be developed as follows : Start from the maximal element of $\mathcal{F}(\mathcal{P};\mathcal{S})$, $\bar{\mathbf{f}}$, and generate the arborescence of the elements of $\mathcal{F}(\mathcal{P};\mathcal{S})$ defined by the $\leq$ ordering; terminate the search along each path when an element $\mathbf{f} \in \mathcal{F}_c(\mathcal{P};\mathcal{S})$ is identified. A formal statement of this algorithm is given in Figure 2. Notice that the search for the maximal elements of $\mathcal{F}_c(\mathcal{P};\mathcal{S})$ must traverse the chains of $\mathcal{F}(\mathcal{P};\mathcal{S})$ in a decreasing fashion, since, in general, given any pair of meaningful vectors $\mathbf{f}$ and $\mathbf{f}'$ s.t. $\mathbf{f}' \leq \mathbf{f}$, $\mathbf{f} \in \mathcal{F}_c(\mathcal{P};\mathcal{S}) \not\Longrightarrow \mathbf{f}' \in \mathcal{F}_c(\mathcal{P};\mathcal{S})$. Finally, we make the following comment regarding the complexity of the proposed algorithm: Since the algorithm essentially performs a search over the chains of poset $\mathcal{F}(\mathcal{P};\mathcal{S})$ for maximal elements belonging to $\mathcal{F}_c(\mathcal{P};\mathcal{S})$, in the worst case, it will have to evaluate $O([\max_i\{\bar{\mathbf{f}}[i]\}]^{dim(\mathbf{f}_{\mathcal{P}})})$ search nodes, with the evaluation of each search node involving the solution of the MIP formulation defined by Equations 3–9. Hence, it can be concluded that it is a rather computationally tedious proposition.[2] Notice, however, that the execution of this algorithm will take place off-line, as an optimization step during the policy development phase. Hence, the computational complexity underlying this phase of the policy development is not as critical an issue as the phase concerning the real-time policy implementation.[3] Furthermore, in very large-scale applications, where the cost of exhaustive search might be prohibitive, the basic policy enhancement logic underlying the proposed scheme can be used not for identifying maximal elements in $\mathcal{F}_c(\mathcal{P};\mathcal{S})$, but simply some element(s) $\mathbf{f} \in \mathcal{F}_c(\mathcal{P};\mathcal{S})$ that leads to a policy implementation which is more relaxed/flexible compared to the original policy definition. In such a case, the search can be based on randomized sampling of the space $\mathcal{F}'(\mathcal{P};\mathcal{S}) \equiv \{\mathbf{f} \in \mathcal{F}(\mathcal{P};\mathcal{S}) : \ \mathbf{f} \geq \mathbf{f}_0\}$, where $\mathbf{f}_0$ denotes the rhs vector employed by the original definition of $\mathcal{P}$. Lastly, we believe that more computationally streamlined versions of the proposed approach can be developed, e.g., through parallelization of the search task underlying the algorithm of Figure 2; however, the detailed development of such an enhanced computational scheme is beyond the scope of this technical note.

[2] More formally, the algorithm involves the solution of an exponential number of NP-complete problems.

[3] We remind the reader that the computational efficiency of algebraic PK-DAP's regarding their real-time execution is established by the requirement that the number of constraints in Equation 1 is polynomial w.r.t. the RAS size, where the latter is defined by the size of the PN N modeling the uncontrolled system behavior.

```
# Data Structures:
# W: The list with the Hasse diagram nodes (vectors) remaining to be processed,
#      organized as FIFO
# M: The set of identified maximal elements of F_c(P;S)

# Initialization:
W =< f̄ >;
M = ∅;

# Main Loop:
while (W ≠ null) do
begin
   f := Pop(W);
   if (C(f) = |P ∪ R ∪ W|) then
      M := M ∪ {f};
   else
      for (i := 1 to dim(f_P)) do
      begin
         f' := f − e_i;      # e_i denotes the i-th unit vector
         if (f' ≥ 1 ∧ ∄f'' ∈ M ∪ W : f'' ≤ f') then
            Push (W, f');
      endfor
   endif
endwhile

return(M);
```

Figure 2: An algorithm for evaluating the maximal elements of $\mathcal{F}_c(\mathcal{P};\mathcal{S})$

Next, we show that the above algorithm is an effective tool for developing new correct algebraic PK-DAP's for SU-RAS. This claim is validated by the following example.

Example 2 Let us consider the SU-RAS in Example 1 again, but with a different resource ordering: $O(R_1) = O(R_2) = O(R_3) = 1$. According to Definition 1, the new RUN implementation is defined by

$$\begin{bmatrix} 1 & & & 1 & 1 & 1 \\ 1 & 1 & & 1 & 1 & \\ 1 & 1 & 1 & 1 & & \end{bmatrix} \cdot \mathbf{q} \leq \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \tag{18}$$

Let $\mathbf{f}_0 \equiv (1,2,1)^T$, i.e. the rhs vector for the original RUN implementation of Equation 18. The policy itself will be denoted by $\mathcal{P}(\mathbf{f}_0)$. According to Definition 3, the maximal meaningful rhs vector for the $\mathbf{A}_{\mathcal{P}}$ given in Equation 18, is $\bar{\mathbf{f}} = (4,4,4)^T$. Application of the search algorithm over $\mathcal{F}(\mathcal{P};\mathcal{S})$ led to the following set of maximal vectors in $\mathcal{F}_c(\mathcal{P};\mathcal{S}) : \{(2,4,2)^T\}$. Let $\mathbf{f}^* \equiv (2,4,2)^T$, and $\mathcal{P}(\mathbf{f}^*)$ denote the algebraic PK-DAP defined by Equation 18, with $\mathbf{f}_0$ substituted by $\mathbf{f}^*$. Some important observations regarding $\mathcal{P}(\mathbf{f}^*)$ are as follows: (i) $\mathcal{P}(\mathbf{f}^*)$ is more flexible/permissive than $\mathbf{f}_0$, since it admits all RAS states admitted by $\mathbf{f}_0$, and also admits state $(1,1,0,0,1,0)^T$ which is not admitted by $\mathbf{f}_0$. (ii) Furthermore, state $(1,1,0,0,1,0)^T$ cannot be admitted by any implementation of RUN based on Definition 1, no matter which particular resource ordering is used. This can be argued as follows: (a) any RUN implementation based on a resource ordering with $O(R_1) \leq O(R_2)$, will have the constraint corresponding to the neighborhood of resource R_2 violated (the neighborhood job content is $3 > 2$); (b) similarly, any RUN implementation based on a resource ordering with $O(R_1) \geq O(R_2)$, will have the constraint corresponding to the neighborhood of resource R_1 violated (the neighborhood job content is $2 > 1$). Hence, $\mathcal{P}(\mathbf{f}^*)$ does not belong to the original RUN family, defined by the set of all possible orderings of the system resources that can be employed during the policy implementation. (iii) Finally, an argument similar to that of Observation (ii), regarding the admissibility of state $(1,0,0,1,0,0)^T$, establishes that $\mathcal{P}(\mathbf{f}^*)$ is also different from any implementation of RO [8] and BKDAA [1] DAP's.[4] In fact, this state is even discarded by the more general Banker's algorithm [7]. $\diamond$

Based on the previous example, it is clear that the proposed AMG-based framework has the capability (i) to enhance the flexibility of any existing algebraic PK-DAP's, and (ii) to enrich the space of effectively computable algebraic PK-DAP's with new policy instantiations. In fact, the proposed framework can be applied to any (tentative) algebraic PK-DAP, $\mathcal{P}$, obtained by employing some arbitrary structure for matrix $\mathbf{A}_{\mathcal{P}}$. In this case, however, there is no guarantee that $\mathcal{F}_c(\mathcal{P};\mathcal{S}) \neq \emptyset$.

[4] Currently, these are the other two major effectively computable classes of algebraic PK-DAP's.

4 Conclusions

In this paper, it was shown that the class of algebraic PK-DAP's for SU-RAS, originally developed in the FSA framework [9], can also be analyzed by use of recent results from PN structural analysis [3]. In addition, the approach to DAP analysis and design developed in this paper, has led to the effective generalization of the currently known families of DAP's, and to their enrichment with new and more flexible policy implementations. Future work will seek the extension of the above results to sequential RAS with more general structure. As a starting point, we observe that results similar to that of Proposition 1, relating the net liveness to the (non-)existence of potentially empty siphons, have recently appeared for additional PN classes appropriate for RAS modeling in [2, 11].[5] In fact, it can be easily shown that the AMG class itself can model the controlled behavior of a more general RAS class – characterized as *conjunctive* [10] – where each process stage can require more than one resource type, with the additional constraint, however, that only up to a single unit can be requested from each resource. This remark further implies that the behavior of this conjunctive RAS class can also be analyzed by means of the framework presented in this paper; i.e., algebraic PK-DAP's in the form of Equation 1 can be developed for this broader RAS subclass, and the result of Corollary 1 still provides an analytical framework for evaluating the policy correctness. Another challenging extension of the proposed research is to use the analysis framework developed in this paper in order to characterize the properties of the $\mathbf{A}_{\mathcal{P}}$ structure that supports correct policy implementations. Such an insight would allow the synthesis of broader classes of correct algebraic DAP's for sequential RAS, and the effective incorporation of additional external constraints in the policy development process.

A Appendix: An Introduction to Ordinary Petri nets (OPN)

An ordinary PN is defined by a quadruple $N = (P, T, F, M_0)$, where P is the set of *places*, T is the set of *transitions*, $F \subseteq (P \times T) \cup (T \times P)$ is the set of directed arcs connecting places with transitions, and $M_0 : P \to Z^+$ is the net *initial marking*, assigning to each place $p \in P$, $M_0(p)$ *tokens*. The set of input (resp., output) transitions of a place p is denoted by ${}^\bullet p$ (resp., $p^\bullet$). Similarly, the set of input (resp., output) places of a transition t is denoted by ${}^\bullet t$ (resp., $t^\bullet$). This notation is also generalized to any set of places or transitions, X, e.g. ${}^\bullet X = \bigcup_{x \in X} {}^\bullet x$. The ordered set $X =< x_1 \ldots x_n > \subseteq \; P \cup T$ is a *path*, iff $x_{i+1} \in x_i^\bullet, i = 1, \ldots, n-1$. Furthermore, a path X is characterized as *simple*, iff $x_i \neq x_j$, $\forall i \neq j$, and it is characterized as a *circuit* iff $x_1 \equiv x_n$. Given a PN marking M, a transition t is *enabled* or *firable* if $\forall p \in {}^\bullet t$, $M(p) \geq 1$. Firing an enabled transition t results in a new marking M', which is obtained by removing

[5] We would like to thank the paper reviewer that brought these papers to our attention.

one token from each place in ${}^\bullet t$, and placing one token in each place in $t^\bullet$. Furthermore, the marking M' reached from marking M by firing the transition sequence σ, can be computed through the following *state equation* :

$$M' = M + \Theta\bar{\sigma} \tag{19}$$

In Equation 19, $\Theta = [\theta_{ij}]$, $i = 1 \ldots |P|$, $j = 1 \ldots |T|$, is the net *incidence matrix*, with $\theta_{ij} = 1$ if $t_j \in {}^\bullet p_i \setminus p_i^\bullet$, $\theta_{ij} = -1$ if $t_j \in p_i^\bullet \setminus {}^\bullet p_i$, and $\theta = 0$ otherwise; $\bar{\sigma}$ is the *firing count vector*, of dimension $|T|$, and with component $\bar{\sigma}_i$ indicating how many times transition t_i was fired in sequence σ. The set of markings reachable from M_0 through any firable sequence of transitions is denoted by $R(M_0)$.

A nonnegative integer vector x, of dimensionality $|P|$, is called a *p-invariant* of the net, if $x^T\Theta = 0$. Combined with Equation 19, this definition of the p-invariant vector x implies that $x^T M = x^T M_0$, $\forall M \in R(M_0)$. A PN is said to be *structurally bounded* iff all markings $M \in R(M_0)$ remain bounded for any M_0. It is said to be *live* iff every transition in it can become firable from any marking $M \in R(M_0)$. It is said to be *reversible* iff M_0 is reachable from every marking $M \in R(M_0)$. Three important notions in the analysis of liveness and reversibility of ordinary PN are those of the PN siphon, trap, and potential deadlock. Formally, a *siphon* is defined as a set of places, S such that ${}^\bullet S \subseteq S^\bullet$. Similarly, a *trap* is a set of places, S, such that $S^\bullet \subseteq {}^\bullet S$. Finally, a *potential deadlock* is a siphon, S, that can eventually become empty. A siphon (resp., trap), S, is characterized as *minimal*, if it does not contain any other siphon (resp. trap). To understand the significance of the above three concepts in PN analysis, note that, due to the defining set inclusions and the aforestated rule for transition firing, an empty (resp. marked) siphon (resp. trap) will remain empty (resp. marked) in all subsequent markings. As a result, for an empty siphon S, transitions $t \in S^\bullet$ are *dead*, i.e., they will not be enabled in any subsequent marking. On the other hand, a siphon S that contains a trap T will never be empty. It can also be shown that for any PN marking M such that no transition is enabled, the set of empty places defines a siphon. Finally, if $\forall p \in P$, $|p^\bullet| = |{}^\bullet p| = 1$, the PN is called a *marked graph*.

References

[1] Z. A. Banaszak and B. H. Krogh. Deadlock avoidance in flexible manufacturing systems with concurrently competing process flows. *IEEE Transactions on Robotics & Automation*, 6(6):724–734, 1990.

[2] K. Barkaoui, A. Chaoui, and B. Zouari. Supervisory control of discrete event systems based on structure theory of petri nets. In *IEEE International Conference on Systems, Man, & Cybernetics*, pages 3750–3755. IEEE, 1997.

[3] F. Chu and X-L. Xie. Deadlock analysis of petri nets using siphons and mathematical programming. *IEEE Transactions on Robotics & Automation*, 13(6):793–804, 1997.

[4] J. Desel and W. Reisig. Place/transition petri nets. In W. Reisig and G. Rozenberg, editors, *Lectures on Petri Nets I: Basic Models*, volume 1491 of *Lecture Notes in Computer Science*, pages 122–173. Springer-Verlag, Berlin, 1998.

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

[6] 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*, 13:347–363, 1997.

[7] 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*, 10:73–100, 1998.

[8] M. Lawley, S. Reveliotis, and P. Ferreira. A correct and scalable deadlock avoidance policy for flexible manufacturing systems. *IEEE Transactions on Robotics & Automation*, 14(5):796–809, 1998.

[9] S. A. Reveliotis and P. M. Ferreira. Deadlock avoidance policies for automated manufacturing cells. *IEEE Transactions on Robotics & Automation*, 12(6):845–857, 1996.

[10] S. A. Reveliotis, M. A. Lawley, and P. M. Ferreira. Polynomial complexity deadlock avoidance policies for sequential resource allocation systems. *IEEE Transactions on Automatic Control*, 42(10):1344–1357, 1997.

[11] F. Tricas, F. Garcia-Vallez, J. M. Colom, and J. Ezpeleta. A structural approach to the problem of deadlock prevention in processes with resources. In *Proc. of the 4th Intl. Workshop on DES*, pages 273–278. IEE, AEI, 1998.

[12] L. A. Wolsey. *Integer Programming.* John Wiley & Sons, 1998.

[13] K. Y. Xing, B. S. Hu, and H. X. Chen. Deadlock avoidance policy for petri net modeling of flexible manufacturing systems with shared resources. *IEEE Transactions on Automatic Control*, 41:289–295, 1996.

[14] M. Zhou and F. Dicesare. *Petri Net Synthesis for Discrete Event Control of Manufacturing Systems.* Kluwer Academic Publishers, 1993.