2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007

Generalized Algebraic Deadlock Avoidance Policies for Sequential Resource Allocation Systems

Spyros Reveliotis, Elzbieta Roszkowska and Jin Young Choi

Abstract— **Currently, one of the most actively researched approaches regarding the design of deadlock avoidance policies for sequential resource allocation systems is based on concepts and techniques provided by the, so called, *theory of regions*, that addresses the broader problem of synthesizing PN models with pre-specified behaviors. However, one limitation of the theory of regions and its aforementioned derivatives is that they cannot be applied when the target behavior has a non-convex representation in the underlying state space. In this paper, we show how this problem can be circumvented by appropriately generalizing the employed class of the candidate policies.**

I. Introduction

The problem of deadlock avoidance in sequential resource allocation systems (RAS) is a well established and extensively studied problem in the relevant literature. Generally speaking, the problem concerns the coordinated allocation of a finite set of reusable resources to a set of concurrently executing processes so that circular waiting situations – i.e., situations where a subset of processes wait upon each other for the release of the necessary resources for their advancement – are avoided and each process can proceed to its successful completion. Past work has formally characterized the problem by means of a number of modelling frameworks provided by qualitative Discrete Event Systems (DES) theory [1] – e.g., finite state automata, Petri nets (PN), and various other graph theoretic models – and it has also provided a number of methodologies for the synthesis of the necessary deadlock avoidance policies (DAP's) for various sub-classes of these systems; we refer the reader to [2], [3] for a systematic and comprehensive exposition of all the currently available results. The main focus of this work is a particular DAP class that in the past has been characterized as *algebraic*, since the relevant policies seek to ensure the deadlock-freedom of the underlying RAS by confining its operation in a subspace that satisfies a properly chosen set of linear inequalities. Some typical examples of such policies are the RUN and RO DAP's, introduced respectively in [4] and [5]. Furthermore, references [2], [3] provide some generic methodology for synthesizing appropriate algebraic DAP's for a very broad RAS class, while more recently, the works of [6], [7] have offered some interesting additional insights regarding the functionality of these policies.

S. Reveliotis is with the School of Industrial & Systems Engineering, Georgia Institute of Technology, USA spyros@isye.gatech.edu

E. Roszkowska is with the Institute of Computer Engineering, Control and Robotics, Wroclaw University of Technology, Poland ekr@pwr.wroc.pl

J. Y. Choi is with the Digital Communications Infra Division, Samsung Networks, Inc., Korea jin_young.choi@samsung.com

When it comes to the synthesis of algebraic DAP's, one of the currently most active approaches seeks first (i) to deploy the reachable state space of the underlying RAS, and subsequently (ii) to exploit the information provided in the obtained reachability graph in order to derive the linear inequalities that will successfully establish deadlock-free operation. The original works of [8], [9] that introduced this method, motivated and justified it on the basis of some more general results pertaining to the synthesis of PN models with pre-specified behaviors, collectively known as the *theory of regions* [10]. Furthermore, the aforementioned works constrained the synthesis problem to that of obtaining the *maximally permissive* or *optimal* DAP, i.e., the DAP that admits the maximal possible subspace that guarantees deadlock-free behavior.[1] However, there are two potential problems that can arise during the deployment of the aforementioned approach, when confined to the computation of the maximally permissive DAP: (i) The first problem relates to the computational complexity of the resulting policy, since it is possible that the maximally permissive DAP is characterized by a number of linear inequalities that is a super-polynomial function of the size of the underlying RAS. (ii) The second problem is of a more existential nature, since it is also possible that the target behavioral space characterizing the optimal DAP is not convex, and therefore, it cannot be effectively characterized by a system of linear inequalities. Motivated by these remarks, the work of [11] proposed a variation of the original methodology that, instead of always seeking to compute the maximally permissive DAP, it computes, through an appropriate Mixed Integer Programming (MIP) formulation [12], the most efficient DAP that can be expressed by a user-specified number of linear inequalities. Clearly, this new approach can guarantee the polynomial complexity of the derived policy by (pre-)selecting a number of linear inequalities that is polynomially related to the underlying RAS size. Furthermore, the methodology can overcome the second problem stated above by returning a sub-optimal DAP with a convex admissible state space, in the case that the subspace defined by the optimal DAP is not convex. The work of [11] also discusses how to accommodate additional design considerations, like uncontrollable process advancement and resource allocation, and how to deal with the computational complications arising from the potentially large-scale nature of the underlying

[1] All the technical concepts and results that are necessary for the thorough understanding of the presented work are systematically introduced in the subsequent parts of this manuscript.

1-4244-0602-1/07/$20.00 ©2007 IEEE.

reachability space. Finally, from a more analytical standpoint, the constraints of the MIP formulation presented in [11] provide a complete, formal characterization of the entire class of algebraic DAP's that are appropriate for any given RAS and do not exceed a certain dimensionality bound.

Yet, one particular issue that remains unresolved by the aforecited works, is how to extend the concept of the algebraic DAP and how to appropriately modify the outlined methodology in order to be able to obtain the maximally permissive DAP even in the case that the relevant state space is non-convex. This particular problem is undertaken in this work. More specifically, the results provided in this paper (i) first re-cast the MIP formulation developed in [11] so that it pertains more directly to the concepts underlying the DAP synthesis problem – i.e., stripping it from the "overhead" concepts and elements introduced by the previously adopted PN formalism[2] – and subsequently (ii) they exploit the insights obtained from this new formulation, in order to address the policy extension problem, by introducing the class of *generalized* algebraic DAP's. The last part of the paper also discusses how to modify the derived MIP formulation so that it applies to the synthesis of generalized algebraic DAP's. We start the development of this material by formally introducing in the next section the notion of sequential RAS and the corresponding deadlock avoidance problem. In order to provide a more concrete exposition of our results, we confine our discussion in the class of disjunctive/conjunctive (D/C-)RAS; however, it must be pointed out that the presented methodology pertains to any other RAS covered by the representational framework introduced in [2]. Finally, we notice that while the imposed space limitations do not allow the inclusion of demonstrative examples in this manuscript, such examples can be readily found in [11] and the other provided references (e.g., [8], [9]).

II. Disjunctive / Conjunctive Resource Allocation Systems and their Deadlock Avoidance Problem

D/C-RAS For the purposes of this work, a *Disjunctive / Conjunctive Resource Allocation System (D/C-RAS)* is formally defined by a 4-tuple $\Phi =< \mathcal{R}, C, \mathcal{P}, \mathcal{A} >$ where: (i) $\mathcal{R} = \{R_1, \ldots, R_m\}$ is the set of the system *resource types*. (ii) $C : \mathcal{R} \rightarrow Z^+$ – the set of strictly positive integers – is the system *capacity* function, characterizing the number of identical units from each resource type available in the system. Resources are considered to be *reusable*, i.e., each allocation cycle does not affect their functional status or subsequent availability, and therefore, $C(R_i) \equiv C_i$ constitutes a system *invariant* for each i. (iii) $\mathcal{P} = \{\Pi_1, \ldots, \Pi_n\}$ denotes the set of the system *process types* supported by the considered system configuration. Each process type Π_j is a composite element itself, in particular, $\Pi_j =< \mathcal{S}_j, \mathcal{G}_j >$, where: (a) $\mathcal{S}_j = \{\Xi_{j1}, \ldots, \Xi_{j,l(j)}\}$ denotes the set of *processing stages* involved in the definition of process type Π_j, and (b) $\mathcal{G}_j$ is an *acyclic digraph* with its node set, V_j, being bijectively related to the set $\mathcal{S}_j$. Let $V_j^{\nearrow}$ (resp., $V_j^{\searrow}$) denote the set of *source* (resp., *sink*) nodes of $\mathcal{G}_j$. Then, any *path* from some node $v_s \in V_j^{\nearrow}$ to some node $v_f \in V_j^{\searrow}$ defines a *process plan* for process type Π_j. (iv) $\mathcal{A} : \bigcup_{j=1}^{n} \mathcal{S}_j \rightarrow \prod_{i=1}^{m}\{0, \ldots, C_i\}$ is the *resource allocation function* associating every processing stage Ξ_{jk} with a *resource allocation request* $\mathcal{A}(j,k) \equiv A_{jk}$. More specifically, each A_{jk} is an m-dimensional vector, with its i-th component indicating the number of resource units of resource type R_i necessary to support the execution of stage Ξ_{jk}. Obviously, in a well-defined RAS, $A_{jk}(i) \leq C_i, \ \forall j,k,i$. Furthermore, the resource set A_{jk}, required for the execution of a particular processing stage Ξ_{jk}, is allocated exclusively and non-preemptively to each process instance, and it is released by it only upon the allocation of the resources required for the execution of the subsequent stage. Finally, $|\Phi| \equiv |\mathcal{R}| + |\bigcup_{j=1}^{n} \mathcal{S}_j| + \sum_{i=1}^{m} C_i$ will be referred to as the *size* of Φ.

A logical characterization of the RAS behavior The behavior generated by the D/C-RAS can be formally modeled as a *Finite State Automaton (FSA)* [1]. The state of this automation is defined as follows:

Definition 1: The D/C-RAS *state*, $s(t)$, at time t, is a vector of dimensionality D, equal to the total number of distinct processing stages, such that each of its components $s(t;q)$ corresponds to a processing stage Ξ_{jk} and indicates the number of process instances executing stage Ξ_{jk} at time t. $\diamond$

To simplify the notation, the following discussion omits the dependence of state s on time t. The information contained in the RAS state is sufficient for the determination of the distribution of the resource units to the various processing stages, as well as of the *slack* (or *idle*) resource capacity in the system; in particular, the *slack* capacity, $\delta_i(s)$, of resource R_i at sate s, can be computed as $\delta_i(s) \equiv C_i - \sum_{q=1}^{D} s(q(j,k)) \cdot A_{jk}(i)$. The set S of *feasible* resource allocation states for the considered RAS is defined by $S \equiv \{s \in (Z_0^+)^D : \delta_i(s) \geq 0, \ \forall i = 1, \ldots, m\}$. The finiteness of the resource capacities implies that $card(S) \equiv |S| < \infty$. However, in general, $|S|$ will be a *super-polynomial* function of the RAS size.

The set of *events*, E, that can change the system state, comprises: (i) the events $e_{jk}^{l}, \ j = 1, \ldots, n, \ k \in \{1, \ldots, l(j) : \ \Xi_{jk} \in V_j^{\nearrow}\}$, corresponding to the *loading* of a new instance of process type Π_j into the system, that is to follow a process plan starting with stage Ξ_{jk}, (ii) the events $e_{jkh}^{a}, \ j = 1, \ldots, n, \ k,h = 1, \ldots, l(j), \ k \neq h$, corresponding to the *advancement* of a process instance executing stage Ξ_{jk} to a successor stage Ξ_{jh}, and (iii) the events e_j^u, corresponding to the *unloading* of a finished process instance of type Π_j. Without loss of generality, it is assumed that, during a single state transition, only one of these events can take place. The resulting transition, however, is *feasible* only if the additionally requested set of resources

[2]Of course, as mentioned above, this formalism was instrumental for the original conception and the formal justification of the approach.

can be obtained from the system slack capacity.

A natural definition for the *initial* state is $s_0 \equiv \mathbf{0}$ i.e., the state in which the system is idle and empty of any process instances. Since the main logical concern addressed herein is the establishment of non-blocking behavior, the set of *marked* states is defined as $S_m \equiv \{s_0\}$. Hence, the *marked language* $\mathcal{L}_m$ of this automaton corresponds to *"complete runs"*.

The above FSA-based model of the RAS behavior can be expressed graphically by the *State Transition Diagram (STD)*, i.e., a digraph $\mathcal{G}$ with *nodes* corresponding to the FSA states, and *edges* corresponding to the feasible state transitions. Of particular interest is the STD subgraph induced by the nodes s that are reachable from node s_0; this subgraph is denoted by S_r and it is characterized as the *reachable* subspace of the considered RAS.

Deadlock and Deadlock Avoidance in D/C-RAS A major concern in the logical control of RAS is the establishment of *live* – or *deadlock-free* or *non-blocking* – behavior. *Deadlocks* are defined as RAS states where there is a set of process instances, such that each of its processes, in order to advance, requests the allocation of resources currently held by some other process(es) in the considered set. Their development results from (i) the fact that processes will hold upon their allocated resources in a non-preemptive manner and (ii) the arbitrary structure of the process routes that can give rise to cyclical patterns of resource requests among the various executing processes.

In the FSA-based modelling of the RAS operation, deadlocks are represented by the formation of *strongly connected components* in the system reachable space, S_r, which, however, are not *co-accessible*, i.e., the empty state, s_0, is not reachable from them through any sequence of feasible transitions. Hence, a *correct Deadlock Avoidance Policy (DAP)*, Δ, tries to restrict the system operation to a *strongly connected component of S_r which contains the empty state s_0*. The RAS subspace that is reachable under – or *admissible* by – some DAP Δ will be denoted by $S_r(\Delta)$. Given a D/C-RAS configuration, an applied DAP is characterized as *optimal*, if the corresponding admissible subspace is the *maximal* strongly connected component of S_r which contains the empty state s_0. The set of states admitted by the optimal DAP, Δ^*, is characterized as (the set of) *reachable safe* states, and it is denoted by S_{rs}. The complement of S_{rs} with respect to S_r is denoted by S_{ru}, and it constitutes the system *reachable unsafe* (sub-)space.

In the D/C-RAS operational context, the optimal DAP, Δ^*, is well-defined, and it is effectively computable through an *one-step lookahead* scheme that admits a tentative resource allocation if and only if (*iff*) the resulting state is safe. However, the corresponding state safety problem is NP-complete [13]. In the light of this result, the research community has sought the development of sub-optimal DAP's that are implementable in polynomial complexity with respect to the underlying RAS size, and yet, efficient, i.e., they manage to admit a large part of S_{rs}. This idea has been formalized by the concept of *Polynomial Kernel (PK-)* DAP [2]. From an implementational standpoint, a typical approach to the design of PK-DAP's is the identification of a property $\mathcal{H}(s),\ s \in S$, such that (i) the complexity of testing $\mathcal{H}()$ on the RAS states is polynomial with respect to the RAS size, and (ii) the subspace $\{s \in S_r : \mathcal{H}(s) = \texttt{TRUE}\}$ is strongly connected.[3] In this setting, *algebraic* PK-DAP's can be defined as the particular class of PK-DAP's where the property $\mathcal{H}(s)$ constitutes a system of linear inequalities on the RAS state s that is polynomially sized with respect to the RAS size $|\Phi|$. In the next section, first we characterize the set of correct algebraic DAP's for any given D/C-RAS Φ that can be expressed as a system of K linear inequalities, where K is an externally specified parameter, and subsequently we employ this characterization towards the development of a mathematical programming (MP) formulation that will return the most efficient DAP in the aforementioned class, when assuming that efficiency is characterized by a "weight" function defined on the reachable space of Φ, S_r.

III. Design of Algebraic DAP's through the Deployment of the RAS Reachability Space

In the subsequent discussion, the considered class of algebraic DAP's will be represented with the tuple (A, b), where A is a $K \times D$ real-valued matrix and b is a K-dimensional real-valued vector. Furthermore, the algebraic policy

$$A \cdot s \leq b \tag{1}$$

obtained for some particular pricing of the matrix A and vector b, will be denoted by $\Delta(A, b)$. It must be noticed that this definition of the algebraic DAP constitutes already a generalization of the way that this concept was employed in the earlier works, since it allows for real-valued entries of the policy-defining elements A and b.[4] Also, it is easy to see that under this extended, real-valued representation, all the elements of A and b can be appropriately scaled so that they belong in the interval $[-1, 1]$, while maintaining the discriminatory power of the original constraint set; i.e., all the key problem variables in this section can be naturally bounded in the interval $[-1, 1]$, and this assumption will be applied in the subsequent developments:

$$\forall i = 1, \ldots, K,\ \forall j = 1, \ldots, D,$$
$$-1 \leq A(i, j) \leq 1 \tag{2}$$
$$\forall i = 1, \ldots, K,$$
$$-1 \leq b(i) \leq 1 \tag{3}$$

We remind the reader that according to the characterizations provided in the previous section, an algebraic DAP $\Delta(A, b)$ will be correct *iff every state $s \in S_r \backslash \{s_0\}$ that is*

[3] We notice, for completeness, that an additional condition that is expected to be satisfied by the property $\mathcal{H}()$ defining a PK-DAP, is that $\mathcal{H}(s_0) = \texttt{TRUE}$; c.f. [2]. However, this requirement can be relaxed since the state s_0 can be easily recognized and admitted through an additional step that tests explicitly for "$s = s_0$" during the policy implementation. This approach is actually presumed in the rest of this work.

[4] The integrality of the elements of matrix A and vector b in the algebraic DAP's appearing in all the past developments, was the result of the (ad-hoc) reasoning underlying the specification of these DAP's and/or the PN-based modelling framework employed for their analysis.

reachable under the policy, is also co-reachable *under it, i.e., there is an admissible transition sequence from state s back to the initial state s_0*. This correctness specification can be expressed analytically through a constraint set that confines the pricing of variables $A(i,j),\ i=1,\ldots,K,\ j=1,\ldots,D$ and $b(i),\ i=1,\ldots,K$, and it is developed as follows:

First we introduce the binary variables $x_l^i,\ l=1,\ldots,|S_r|,\ i=1,\ldots,K$, that will be priced to one if state s_l satisfies the inequality $A(i,\cdot)\cdot s_l \le b(i)$, and to zero, otherwise. This pricing can be achieved by the following constraint set:

$$\forall l=1,\ldots,|S_r|,\ \forall i=1,\ldots,K,$$
$$b(i)-A(i,\cdot)\cdot s_l \le \epsilon\cdot(x_l^i-1)+U\cdot x_i^l \quad (4)$$
$$b(i)-A(i,\cdot)\cdot s_l \ge \epsilon\cdot x_l^i+U\cdot(x_l^i-1) \quad (5)$$
$$x_l^i\in\{0,1\} \quad (6)$$

The introduction of the parameter $\epsilon>0$ in Eqs 4 and 5 seeks to prevent any potential ambiguity in the developed formulation, by forcing the value of the difference $|A(i,\cdot)\cdot s_l-b(i)|$ away from zero. Its value should be chosen small enough so that it does not affect the outcome of the formulation, and it can be determined by trial and error. On the other hand, the parameter U appearing in Eqs 4 and 5 denotes an upper bound for the quantity $|A(i,\cdot)\cdot s_l-b(i)|,\ l=1,\ldots,|S_r|,\ i=1,\ldots,K$, and it can be readily obtained when taking into consideration the bounds established by Eqs 2 and 3, and the bounds established for the components of the state vector s by the finiteness of the resource capacities. It is easy to see, then, that a positive value for the difference $b(i)-A(i,\cdot)\cdot s_l$ will force x_l^i to one, while a negative value for this difference will force x_l^i to zero.

Given the variables x_l^i, the admissibility of state s_l by policy $\Delta(A,b)$ can be expressed by the real-valued variable x_l, which is priced as follows:

$$\forall l=1,\ldots,|S_r|,\forall i=1,\ldots,K,$$
$$x_l\le x_l^i \quad (7)$$
$$\forall l=1,\ldots,|S_r|,$$
$$x_l\ge\sum_{i=1}^{K}x_l^i-K+1 \quad (8)$$
$$0\le x_l\le 1 \quad (9)$$

Constraint 7 forces x_l to zero, if any of the policy-defining inequalities are violated by state s_l. In the opposite case, the combination of Constraints 8 and 9 forces x_l to one.

In order to formally express the aforestated policy correctness requirement, we also need to characterize the reachability and co-reachability of any state $s_l\in S_r\backslash\{s_0\}$, under $\Delta(A,b)$. For this task, we introduce the additional sets of real-valued variables z_l^q and $y_l^q,\ l=0,\ldots,|S_r|,\ q=0,\ldots\bar{Q}$, such that the pricing of the variable z_l^q (resp., y_l^q) to one indicates that there is a policy-admissible path from state s_0 to s_l (resp., from state s_l to s_0) and the minimal length of any such path is equal to q steps; in any other case, z_l^q (resp., y_l^q) should be priced to zero. The parameter $\bar{Q}$ appearing in the above definition denotes an upper bound to the maximal length of any loop-free path emanating from or resulting to state s_0, and it can be easily obtained from the deployed reachability graph. The availability of the variables z_l^q and y_l^q enables the straightforward expression of the policy correctness requirement through the following constraint set:

$$\forall l=1,\ldots,|S_r|,\ \sum_{q=1}^{\bar{Q}}z_l^q\le\sum_{q=1}^{\bar{Q}}y_l^q \quad (10)$$

It remains, however, to introduce additional constraint sets that will enforce the desired pricing for z_l^q and y_l^q. The set enforcing the correct pricing of variables z_q^l is as follows:

$$\forall l=0,\ldots,|S_r|,\ z_l^0=I_{\{s_l=s_0\}} \quad (11)$$
$$\forall l=1,\ldots,|S_r|,\forall q=1,\ldots\bar{Q},\ z_l^q\ge 0 \quad (12)$$
$$\forall l=1,\ldots,|S_r|,\forall q=1,\ldots\bar{Q},\ z_l^q\le\sum_{m\in IP(l)}z_m^{q-1} \quad (13)$$
$$\forall l=1,\ldots,|S_r|,\forall q=1,\ldots\bar{Q},\ z_l^q\le x_l \quad (14)$$
$$\forall l=1,\ldots,|S_r|,\forall q=1,\ldots\bar{Q},\forall m\in IP(l),$$
$$z_l^q\ge z_m^{q-1}-(1-x_l)-\sum_{\zeta=0}^{q-1}z_l^\zeta \quad (15)$$

Eq. 11 is a "boundary condition" that prices the variables z_l^0; the parameter $I_{\{s_l=s_0\}}$ is the corresponding indicator variable. Eq. 12 states the nonnegative real nature of the variables z_l^q, while the pricing of these variables in a way that is consistent with their definition is enforced by Eqs 13–15. More specifically, the quantity $IP(l)$ appearing in these two equations denotes the set of immediate predecessor states of state s_l, i.e., those states s_m from which there is an immediate transition to state s_l. Hence, Eq. 13 expresses the fact that for state $s_l\in S_r\backslash\{s_0\}$ to be accessible under policy $\Delta(A,b)$ in q steps, there must be an immediate predecessor state s_m that is accessible under policy Δ in $q-1$ steps. Similarly, Eq. 14 expresses the fact that state $s_l\in S_r\backslash\{s_0\}$ will be accessible under $\Delta(A,b)$ only if it satisfies the policy-defining constraints. Finally, Eq. 15 forces z_l^q to 1, if there is an immediate predecessor state s_m that is accessible from state s_0 through a minimal policy-admissible path of $q-1$ steps; however, this enforcement takes place only if (i) state s_l itself is policy-admissible and (ii) there is no other shorter policy-admissible path to it.

The correct pricing of the variables y_l^q can be enforced by a constraint set analogous to that of Eqs 11–15, when noticing that co-reachability becomes equivalent to reachability, once the arcs of the underlying reachability graph have been reversed. Defining the set $IS(l)$ as the set of immediate successor states for state $s_l\in S_r\backslash\{s_0\}$, we obtain:

$$\forall l=0,\ldots,|S_r|,\ y_l^0=I_{\{s_l=s_0\}} \quad (16)$$
$$\forall l=1,\ldots,|S_r|,\forall q=1,\ldots\bar{Q},\ y_l^q\ge 0 \quad (17)$$
$$\forall l=1,\ldots,|S_r|,\forall q=1,\ldots\bar{Q},\ y_l^q\le\sum_{m\in IS(l)}y_m^{q-1} \quad (18)$$
$$\forall l=1,\ldots,|S_r|,\forall q=1,\ldots\bar{Q},\ y_l^q\le x_l \quad (19)$$

$$\forall l = 1, \ldots, |S_r|, \forall q = 1, \ldots \bar{Q}, \forall m \in IS(l),$$
$$y_l^q \geq y_m^{q-1} - (1 - x_l) - \sum_{\zeta=0}^{q-1} y_l^\zeta \quad (20)$$

Hence, for any given D/C-RAS Φ, the constraint set of Eqs 2–20 characterizes the entire set of algebraic DAP's, $\Delta(A, b)$, that can be expressed by no more than K linear inequalities. A notion of optimality can be introduced over this set, by employing an objective function

$$\max \sum_{l=1}^{|S_r|} w_l \cdot \sum_{q=1}^{\bar{Q}} z_l^q \quad (21)$$

where w_l is some "weight" function defined on the set of states $s_l \in S_r \backslash \{s_0\}$. In particular, setting $w_l \equiv 1$, $\forall l$, enables the computation of the maximally permissive DAP, as defined in Section II. Finally, the following theorem is an immediate consequence of the above discussion and it parallels the developments presented in [11]:

Theorem 1: For any given D/C-RAS Φ and a value K characterizing the (maximum) number of the linear inequalities to be employed by the designed algebraic DAP, the formulation of Eqs 2–21 will return the best possible[5] algebraic DAP $\Delta(A, b)$, according to the performance criterion established by the function w_l, $l = 1, \ldots, |S_r|$. $\diamond$

The formulation of Eqs 2–21 will be always feasible, since it contains the *trivial* policy that confines the RAS to its initial state s_0. Of course, such a result should be interpreted as lack of an effective DAP in the considered policy space. If we want such a negative result to be communicated as *infeasibility* by the proposed formulation, we can add the constraint

$$\sum_{l=1}^{|S_r|} z_l^1 \geq 1 \quad (22)$$

Furthermore, in most practical cases, one would like to enforce the existence of at least one policy-admissible process plan for each process type Π_j, $j = 1, \ldots, n$. In such a case, Constraint 22 should be replaced with the following stronger requirement:

$$\forall j = 1, \ldots, n, \quad \sum_{l \in LD(j)} \sum_{q=1}^{\bar{Q}} z_l^q \geq 1 \quad (23)$$

where $LD(j)$ denotes the set of states involving a loading event for process type Π_j.

IV. Generalized Algebraic DAP's

As it was pointed out in the introductory section, if the reachable safe space S_{rs} is a non-convex area according to the state representation introduced by Definition 1, the methodology developed in the previous section will fail to return the optimal DAP, Δ^*, no matter how large we set the value of the design parameter K. This can be immediately deduced by the fact that the polytope defined by Eq. 1 is always a convex region [12]. To circumvent this problem, we need to identify mechanisms that are able to effectively recognize non-convex patterns. It turns out that such a mechanism can be provided by the concept of the *"Committee Machine" (CM)*, a pattern classifier that has been studied by the machine learning community since the early sixties [14].[6] In this section, first we introduce the CM concept and establish its capability to recognize non-convex patterns in the (state) spaces of interest in this work, and subsequently we show how to modify the methodology introduced in Section III, so that it applies to the design of DAP's that are based on the CM concept. For reasons that will become clear in the following, we shall refer to this new class of DAP's as *generalized algebraic DAP's*.

Committee Machines Given an n-dimensional pattern space Ω, a *(generalized) committee machine (CM)* is defined by a quadruple $\mathcal{CM} =< A, b, \pi, \theta >$, where: (i) A is a real-valued matrix of some dimensionality $m \times n$; (ii) b is a real-valued m-dimensional vector; (iii) π is a real-valued m-dimensional vector; and (iv) θ is a real-valued scalar. The machine accepts a pattern $\omega \in \Omega$ *iff*

$$\sum_{i=1}^{m} \pi(i) \cdot I_{\{A(i,\cdot)\cdot\omega \leq b(i)\}} \leq \theta \quad (24)$$

In Eq. 24, $I_{\{A(i,\cdot)\cdot\omega \leq b(i)\}}$ denotes the indicator variable that is priced to one if the inequality $A(i,\cdot) \cdot \omega \leq b(i)$ is satisfied, and to zero, otherwise. The following theorem is an immediate consequence of the results presented in ([14], Chpt. 6):

Theorem 2: Given two distinct, finite subsets of a finite-dimensional pattern space Ω, there is always a generalized committee machine to dichotomize them. $\diamond$

Generalized algebraic DAP's It should be clear to the reader that the reachable safe and unsafe subspaces of any given D/C-RAS Φ, satisfy the conditions of Theorem 2. Therefore, there will always exist a committee machine, $\mathcal{CM} =< A, b, \pi, \theta >$, able to effectively recognize the safe sub-space S_{rs}. We shall refer to the DAP established by this CM as a *generalized algebraic DAP*, and we shall denote it by $\Delta(A, b, \pi, \theta)$. The following corollary formalizes the above remarks:

Corollary 1: Given a D/C-RAS Φ, there will always exist a generalized algebraic DAP $\Delta(A, b, \pi, \theta)$ such that $S_r(\Delta(A, b, \pi, \theta)) = S_{rs}$, i.e., the class of generalized algebraic DAP's will always contain the maximally permissive DAP, Δ^*. $\diamond$

Next we modify the design methodology developed in Section III so that it applies to the design of generalized algebraic DAP's.

A MIP formulation for the design of generalized algebraic DAP's We start the discussion of this paragraph by noticing that the scaling effect implied by Eqs 2, 3 pertains also to the defining elements of the committee machine.

[5]We avoid to use the word "optimal" in order to prevent confusion with the optimal DAP Δ^*.

[6]In fact, the concept utilized in this work corresponds more directly to that of the *"two-layered machine"* in [14]. Yet, we opted for the term *"(generalized) committee machine"* since it is more expressive of the dynamics underlying the logic of the resulting policies.

Hence, Eqs 2, 3 will constitute part of the new formulation, and furthermore, they will be complemented by the following set:

$$\forall i = 1, \ldots, K, \quad -1 \leq \pi(i) \leq 1 \tag{25}$$

$$-1 \leq \theta \leq 1 \tag{26}$$

Similarly, Constraints 4–6, that characterize the satisfaction of the linear inequalities $A(i,\cdot) \cdot s_l \leq b(i)$, $l = 1, \ldots, |S_r|$, $i = 1, \ldots, K$, by state s_l, as well as Constraints 10–20, that characterize the policy correctness and the pricing of the indicator variables z_l^q and y_l^q, will not be affected by change of the policy logic, and the same remark applies to the characterization of the problem objective function by Eq. 21. Next, we modify Constraints 7–9 of the previous formulation, so that they express the new admissibility condition of the committee machine.

Given the variables x_l^i, that are priced according to Constraints 4–6, the admissibility of a state s_l, $l = 1, \ldots, |S_r|$, by the generalized algebraic DAP $\Delta(A, b, \pi, \theta)$ can be expressed by a set of binary variables, x_l, that are priced as follows:

$$\forall l = 1, \ldots, |S_r|,$$

$$\theta - \sum_{i=1}^{K} \pi(i) \cdot x_l^i \leq \varepsilon \cdot (x_l - 1) + \Lambda \cdot x_l \tag{27}$$

$$\theta - \sum_{i=1}^{K} \pi(i) \cdot x_l^i \geq \varepsilon \cdot x_l + \Lambda \cdot (x_l - 1) \tag{28}$$

$$x_l \in \{0, 1\} \tag{29}$$

Eqs 27–29 are similar in spirit to Eqs 4–6; in particular, the parameters ε and Λ, appearing in Eqs 27 and 28, play a role similar to that of the parameters ϵ and U in Eqs 4 and 5. However, one problem with the constraint set of Eqs 27–29 is that it is nonlinear, since it involves the products $\pi(i) \cdot x_l^i$. This issue can be remedied by replacing the products $\pi(i) \cdot x_l^i$ in Eqs 27 and 28 with the dummy variables τ_l^i that are priced according to the following constraints:

$$\forall l = 1, \ldots, |S_r|, \ \forall i = 1, \ldots, K,$$

$$-x_l^i \leq \tau_l^i \leq x_l^i \tag{30}$$

$$(x_l^i - 1) \leq \tau_l^i - \pi(i) \leq (1 - x_l^i) \tag{31}$$

Clearly, when $x_l^i = 0$, τ_l^i is forced to zero by Constraint 30, while Constraint 31 becomes identical to Constraint 25. On the other hand, when $x_l^i = 1$, Constraint 31 forces $\tau_l^i = \pi(i)$, and then, Constraint 30 becomes equivalent to Constraint 25. Hence, the variables τ_l^i are properly priced in all cases and no additional side-effects are caused by their introduction.

The next theorem constitutes the counterpart of Theorem 1, for the case of generalized algebraic DAP's, and its correctness is an immediate consequence of the previous discussion.

Theorem 3: For any given D/C-RAS Φ and a value K characterizing the (maximum) number of the linear inequalities to be employed by the designed generalized algebraic DAP, the MIP formulation of Eqs 2–6, 10–21 and 25–31, where the products $\pi(i) \cdot x_l^i$ in Eqs 27 and 28 have been replaced by the variables τ_l^i, will return the best possible generalized algebraic DAP $\Delta(A, b, \pi, \theta)$, according to the performance criterion established by the function w_l, $l = 1, \ldots, |S_r|$. $\diamond$

V. Conclusions

The key contribution of this paper was the extension of the MIP-based methodology for the design of correct algebraic DAP's for sequential RAS, that was originally developed in [11], so that the optimal DAP, Δ^*, is always within the scope of the potential solutions. We also notice, for completeness, that the presented method can be easily adjusted, in a spirit similar to that of [11], in order to account for uncontrollable behavior and/or to cope with the large-scale nature of the underlying state spaces; the relevant details can be traced in [11] and they are left to the reader.

References

[1] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*. Boston, MA: Klumwer Academic Pub., 1999.

[2] S. A. Reveliotis, *Real-time Management of Resource Allocation Systems: A Discrete Event Systems Approach*. NY, NY: Springer, 2005.

[3] M. Zhou and M. P. Fanti (editors), *Deadlock Resolution in Computer-Integrated Systems*. Singapore: Marcel Dekker, Inc., 2004.

[4] M. Lawley, S. Reveliotis, and P. Ferreira, "FMS Structural Control and The Neighborhood Policy, Part 1: Correctness and Scalability," *IIE Trans.*, vol. 29, pp. 877–887, 1997.

[5] ——, "A correct and scalable deadlock avoidance policy for flexible manufacturing systems," *IEEE Trans. on Robotics & Automation*, vol. 14, pp. 796–809, 1998.

[6] Z. W. Li and M. C. Zhou, "Elementary siphons of Petri nets and their application to deadlock prevention in flexible manufacturing systems," *IEEE Trans. on SMC – Part A*, vol. 34, pp. 38–51, 2004.

[7] S. A. Reveliotis, "Implicit siphon control and its role in the liveness enforcing supervision of sequential resource allocation systems," *IEEE Trans. on SMC: Part A*, vol. (to appear).

[8] A. Ghaffari, N. Rezg, and X. Xie, "Design of a live and maximally permissive Petri net controller using the theory of regions," *IEEE Trans. on Robotics & Automation*, vol. 19, pp. 137–141, 2003.

[9] M. Uzam, "An optimal deadlock prevention policy for flexible manufacturing systems using Petri net models with resources and the theory of regions," *Intl. Jrnl of Advanced Manufacturing Technology*, vol. 19, pp. 192–208, 2002.

[10] E. Badouel and P. Darondeau, "Theory of regions," in *LNCS 1491 – Advances in Petri Nets: Basic Models*, W. Reisig and G. Rozenberg, Eds. Springer-Verlag, 1998, pp. 529–586.

[11] S. A. Reveliotis and J. Y. Choi, "Designing reversibility-enforcing supervisors of polynomial complexity for bounded Petri nets through the theory of regions," in *Proceedings of ATPN 2006*, 2006, pp. 322–341.

[12] W. L. Winston, *Introduction To Mathematical Programming: Applications and Algorithms, 2nd ed.* Belmont, CA: Duxbury Press, 1995.

[13] T. Araki, Y. Sugiyama, and T. Kasami, "Complexity of the deadlock avoidance problem," in *2nd IBM Symp. on Mathematical Foundations of Computer Science*. IBM, 1977, pp. 229–257.

[14] N. J. Nilsson, *The Mathematical Foundations of Learning Machines*. San Mateo, CA: Morgan Kaufmann, 1990.