Designing compact and maximally permissive deadlock avoidance policies for complex resource allocation systems through classification theory: the non-linear case

Ahmed Nazeem and Spyros Reveliotis

Abstract— **In a recent work of ours [19], we have proposed the reformulation of the synthesis of the maximally permissive deadlock avoidance policy for certain classes of complex resource allocation systems (RAS) as the design of a linear compact classifier effecting the dichotomy of the underlying reachable state space into its safe and unsafe subspaces. In this work, we extend the results of [19] for the case that the sought dichotomy cannot be represented by a linear classifier. We propose new classification schemes for this more complex case and establish formally their completeness, i.e., their ability to provide an effective classifier for every instance of the considered RAS class. We also provide effective and computationally efficient procedures for the synthesis of the sought classifiers. Finally, the effectiveness and the efficacy of our approaches are demonstrated and assessed through a series of computational experiments.**

I. Introduction

The problem of deadlock avoidance for complex resource allocation systems is a well established problem in the controls literature [25], [29], [15]. Generally speaking, the problem concerns the coordinated allocation of a finite set of reusable resources to a set of concurrently executing processes, so that deadlock is avoided, and every activated process can proceed to its completion. In the operational context considered in this work, deadlock corresponds to a circular waiting pattern where a subset of active processes are waiting upon each other for the release of resources necessary for their further advancement; therefore, these processes are permanently stalled. A deadlock avoidance policy (DAP) is responsible for ensuring that the system avoids these problematic states and that every active process can proceed to its completion.

The study of deadlock and of the deadlock avoidance problem was initiated in the late 60's and early 70's, and it was motivated by the need to manage the resource allocation taking place in the context of the computing technologies that were emerging at that time [11], [10], [5], [12]. Some important contributions of that era were (i) the formalization of the concept of deadlock and of the resource allocation dynamics that lead to its formation by means of graph-theoretic concepts and structures, and (ii) the identification of off-line structural conditions and on-line resource allocation policies that guarantee the deadlock-free operation of the underlying system; designing the resource allocation processes so that they do not give rise to any circular waiting patterns is an example of the aforementioned structural conditions, while Banker's algorithm [6] is the best known DAP of that era. An additional but later development of that era (late 70's) was the systematic study of the computational complexity of the maximally permissive[1] DAP for any given RAS and the establishment of its NP-hardness for the majority of RAS behavior [1], [9]. The problem of deadlock avoidance was subsequently revived in the early 90's, primarily in the context of the resource allocation taking place in flexibly automated production systems and intelligent transportation systems [28], [7], [26], [27]. The defining characteristics of these new studies were (i) the better specificity and predictability of the underlying resource allocation processes with respect to their resource allocation requests, and (ii) the employment of the simultaneously emerging qualitative Discrete Event Systems (DES) theory [4] as a powerful and rigorous base for modeling, analyzing and eventually controlling the considered RAS dynamics. The combination of these two effects has led to (a) a more profound understanding of the process of deadlock formation and of the RAS structural attributes that facilitate this process, under standard DES-based representations like Finite State Automata (FSA) [13] and Petri nets [16], and also to (b) a multitude of methodologies that can provide effective DAPs for various RAS classes.

From a methodological standpoint, the deadlock avoidance problem can be characterized in the supervisory control framework of Ramadge and Wonham (R&W SC) [23], [4] by (i) expressing the underlying RAS as a finite state automaton, (ii) identifying the maximal strongly connected component containing the empty (initial) state, and (iii) requesting the confinement of the underlying RAS behavior to the subspace represented by the aforementioned strongly connected component of the FSA. The aforestated strong connectivity implies that there exists a path from each state in this component to the empty state, which in turn implies that it is possible for the system to advance from each state in this component to the empty state. In fact, such a char-

A. Nazeem and S. Reveliotis are with the School of Industrial & Systems Engineering, Georgia Institute of Technology, email: {anazeem@,spyros@isye.}gatech.edu. This work was partially supported by NSF grant CMMI-0928231.

[1]Maximal permissiveness and all other technical concepts appearing in this introductory discussion will be systematically defined in the subsequent sections.

acterization of the problem and its solution establishes also a notion of optimality for the considered problem. Hence, an optimal DAP is a policy that blocks access only to any state that does not belong to this maximal strongly connected component. However, the direct implementation of the aforementioned approach is challenged by the fact that the assessment of the co-accessibility of any given RAS state to the empty state, a property that is otherwise known as the state "*safety*", is an NP-complete problem for most RAS classes [14]. To circumvent this challenge, the research community addressed the problem in two main directions: The first direction devised algorithms that restrict the system behavior to a subset of safe states that represents a strongly connected component containing the empty state in the R&W SC framework introduced above. The sought component is not necessarily maximal. The main advantage of this approach is that the characterization of such a subset of states is of polynomial complexity with respect to the size of the underlying RAS (e.g., [3], [26]). The second direction adopted a more compact representation of the considered RAS dynamics hoping that the compactness, combined with further structural properties, will lead to a compact characterization of the target policy and to efficient approaches for its derivation. Along this line of research, Petri net (PN) is adopted as the formalism for modeling the underlying RAS. In particular, the non-liveness of the RAS-modeling PNs can be attributed to the formation of some structural properties known as "empty" – or, more generally, "deadly marked" – siphons [7], [22]. The concept of siphons has led to a multitude of efforts that seek to characterize the maximally permissive DAP by imposing the minimum possible amount of control that will prevent the formation of deadly marked siphons [15]. The siphon-based methods are computationally tractable but, in general, the synthesized DAP is suboptimal. An alternative approach that falls within the context of PN-based approaches employs the theory of regions [2], [8]. The key idea behind the theory of regions is to compute the maximally permissive DAP using R&W SC framework and subsequently to encode this policy in a PN model. However, this approach is computationally expensive. An additional major drawback of the PN-based methods is that the maximally permissive DAP might not admit a Petri net representation; a detailed counter-example that establishes this fact is provided in [24], while this effect is also demonstrated by all the RAS instances that are employed in the case study and the computational experiments reported in Section VI of this manuscript.

It can be inferred from the above discussion that the two most prominent challenges in deploying the maximally permissive DAP for any given RAS configuration are (i) the NP-hardness of the computation of the target policy, and (ii) the inability of the PN modeling framework to guarantee an effective representation of the maximally permissive DAP for any given RAS configuration. We shall refer to the second limitation by saying that the corresponding framework is "*incomplete*" with respect to the maximally permissive DAP. In this work we seek to address the two limitations stated above. The first problem is mitigated by discriminating between (a) the computational complexity that concerns the off-line computational effort necessary for acquiring the target policy, and (b) the computational complexity that concerns the on-line implementation of this policy. In general, the computational budget for the former might tolerate expensive computations while that for the latter will be quite stringent. On the other hand, the second problem – i.e., the incompleteness of the PN modeling framework with respect to the maximally permissive DAP – is addressed by selecting an alternative policy representation that guarantees the effective representation of the maximally permissive DAP for any given RAS configuration.

In fact, the computation of the maximally permissive DAP using R&W SC framework is straightforward [4]. However, the size of the FSA modeling the considered RAS grows exponentially with respect to some of its more natural and more compact representations. The severity of the problem is mitigated by performing the aforementioned computation in the off-line stage where the computational requirements are more affordable. On the other hand, the online real-time implementation of the synthesized policy can be perceived as a classifier that effects the dichotomy of the state space of the FSA modeling the RAS behavior into safe and unsafe states, and allows an event to fire if and only if the resultant state is safe. To minimize the computational overhead imposed by the policy on the underlying RAS, the eventually synthesized classifier should be adequately compact.

The ideas outlined in the previous paragraph have been partially investigated in our previous work presented in [19], [18]. Both of these lines of work seek to implement the maximally permissive DAP for certain RAS classes through a two-stage computation: First, an optimal DAP is obtained using R&W SC framework, and subsequently the policy is encoded into a compact classifier that effects the dichotomy of the state space. In [19], the sought classifier consists of a set of linear inequalities. A classifier with such a structure is called a linear classifier. The key property that enables the synthesis of a linear classifier in [19] is the restriction of the considered problem to a RAS class where every resource possesses unit capacity. On the other hand, in [18], the restriction of the resource capacities is relaxed, leading to the inability of the linear classifiers to guarantee the effective representation of the maximally permissive DAP.[2] Therefore, a non-parametric representation is adopted to represent the sought classifier. More specifically, decision diagrams are used to encode the maximally permissive DAP through the storage of a pertinently selected subset of the RAS unsafe states.

This manuscript can be perceived as an extension of

[2]Technically, this inability is established by the infeasibility of the mathematical programming formulations provided in [19] for the design of the relevant linear classifiers. We also point out that the absence of a linear classifier for the maximally permissive DAP implies the further inability to express this policy in the PN modeling framework; the reader is referred to [24] for further discussion on this issue.

those previous lines of work. The RAS class considered in this work has resources of arbitrary capacities. Inspired by the mathematical theory of artificial neural networks [21], we propose a parametric representation for the classifier structure. In particular, the proposed classifier consists of two successive layers of linear inequalities. A RAS state vector is evaluated against each linear inequality in the first layer, and the results are represented by a vector of binary indicators. Each indicator is associated with a linear inequality and indicates whether the inequality is satisfied or violated by the state vector. For the purposes of the proposed classification scheme, this vector of binary indicators can be perceived as the image of the original state vector under the transformation performed by the linear inequalities of the first layer. Applying the first-layer transformation, the problem is reproduced in the binary domain. Hence, we can leverage a result presented in [19], which states that it is always possible to separate two disjoint sets of binary vectors using a set of linear inequalities. For this reason, the first-layer inequalities are constructed such that the image of the set of safe states does not intersect with the image of the set of unsafe states. Then, the second layer of linear inequalities is constructed to separate the image vectors of the safe and the unsafe states. We call a classifier with such a structure a "two-layer classifier". We shall prove that the two-layer classifier is complete with respect to the classification problem addressed in this work.

Our ultimate intention is to effectively compute *compact* two-layer classifiers implementing the maximally permissive DAP for any given RAS from the considered class; i.e., classifiers that will represent effectively the state space dichotomy implied by the corresponding DAP while minimizing the computational effort that is required for the classification of any given RAS state. The construction of such compact classifiers is facilitated by the extension to this new problem context of the "thinning" techniques introduced in [19]; these techniques reduce dramatically the size of the data explicitly involved in the design of the sought classifiers, in terms of, both, the cardinalities of the classified vector sets and their dimensionality. But, to be able to use the aforementioned thinning techniques, we have to restrict our search to the class of two-layer classifiers whose linear-inequality coefficients are non-negative. This effect can possibly increase the size of the classifier, but at the same time, it increases the capability of our design methodology to compute the maximally permissive DAP for RAS configurations with larger state spaces. We formulate the corresponding design problem as a Mixed Integer Program (MIP). Additionally, we offer a set of heuristics that can be employed to handle RAS configurations with very large states spaces, for which the solution of the MIP formulation might be an intractable proposition. A computational study reported in the last part of this document reveals that the derived methods can provide practical implementations of the maximally permissive DAP for RAS configurations that have been considered intractable by the standards of the current literature.

In light of the above positioning of the paper objectives and contributions, the rest of the manuscript is organized as follows: Section II introduces the RAS class to be considered in this work, defines formally the corresponding deadlock avoidance problem, and establishes some of its properties that are crucial for the development of the main results of the paper. The main results themselves are presented in Sections III–V. More specifically, Section III first provides a formal definition of the classification problem to be considered in this work, and subsequently, it proceeds to its reduction to an equivalent classification problem with a much smaller input set in terms of the explicitly considered state vectors and their dimensionality. Section IV addresses the synthesis of a two-layer classifier for the reduced classification problem, by formulating and solving this problem as a mixed integer program. On the other hand, Section V offers a set of heuristics for the synthesis of the sought classifier, that can be used in the case that the MIP formulation of Section IV is deemed to be computationally too costly. Section VI reports a series of computational experiments that demonstrates the proposed methodologies and their efficacy. Finally, Section VII concludes the paper by summarizing its main contributions and suggesting possible extensions of the presented results.

II. The considered RAS class and the corresponding deadlock avoidance problem

The considered RAS class We begin the technical discussion of the paper developments, by providing a formal characterization of the RAS class to be considered in this work. An instance Φ from this class is defined as a 4-tuple $< \mathcal{R}, C, \mathcal{P}, \mathcal{A} >$ where: (i) $\mathcal{R} = \{R_1, \ldots, R_m\}$ is the set of the system *resources*. (ii) $C : \mathcal{R} \rightarrow {\mathbb{Z}_0}^+ -$, i.e., the set of strictly positive integers– is the system *capacity* function, with $C(R_i) \equiv C_i$ characterizing the number of identical units from resource type R_i that are available in the system. Resources are considered to be *reusable*, i.e., they are engaged by the various processes according to an allocation/de-allocation cycle, and each such cycle does not affect their functional status or subsequent availability. (iii) $\mathcal{P} = \{J_1, \ldots, J_n\}$ is the set of the system *process types* supported by the considered system configuration. Each process type J_j is a composite element itself; in particular, $J_j = < \mathcal{S}_j, \mathcal{G}_j >$, where: (a) $\mathcal{S}_j = \{\Xi_{j1}, \ldots, \Xi_{j,l(j)}\}$ is the set of *processing stages* involved in the definition of process type J_j, and (b) $\mathcal{G}_j$ is a connected digraph $(\mathcal{V}_j, \mathcal{E}_j)$ that defines the sequential logic of process type J_j, $j = 1, \ldots, n$. More specifically, the node set $\mathcal{V}_j$ of graph $\mathcal{G}$ is in one-to-one correspondence with the processing stage set, $\mathcal{S}_j$, and furthermore, there are two subsets $\mathcal{V}_j^{\nearrow}$ and $\mathcal{V}_j^{\searrow}$ of $\mathcal{V}_j$ respectively defining the sets of initiating and terminating processing stages for process type J_j. The connectivity of digraph $\mathcal{G}_j$ is such that every node $v \in \mathcal{V}_j$ is accessible from the node set $\mathcal{V}_j^{\nearrow}$ and co-accessible to the node set $\mathcal{V}_j^{\searrow}$. Finally, any directed path of $\mathcal{G}_j$ leading from a node of $\mathcal{V}_j^{\nearrow}$ to a node of $\mathcal{V}_j^{\searrow}$ constitutes a complete execution sequence – or a "route" – for process type J_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, which associates every processing stage Ξ_{jk} with a *resource allocation request* $\mathcal{A}(j,k) \equiv \mathcal{A}_{jk}$. More specifically, each $\mathcal{A}_{jk}$ 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}. Furthermore, it is assumed that (a) $\mathcal{A}_{jk} \neq \mathbf{0}$, i.e., every processing stage requires at least one resource unit for its execution, and (b) the resource allocation requests $\mathcal{A}_{jk}$, $j = 1, \ldots, n$, $k = 1, \ldots, l(j)$, are "conjunctive", i.e., a processing stage Ξ_{jk} can request an arbitrary nonempty subset of the system resources for its execution. Finally, according to the applying resource allocation protocol, a process instance executing processing stage Ξ_{jk} will be able to advance to a successor processing stage $\Xi_{j,k+1}$, only after it is allocated the resource differential $(\mathcal{A}_{j,k+1} - \mathcal{A}_{jk})^+$; and it is only upon this advancement that the process will release the resource units $|(\mathcal{A}_{j,k+1} - \mathcal{A}_{jk})^-|$, that are not needed anymore.

For notational convenience, in the following we shall set $\xi \equiv \sum_{j=1}^{n} |\mathcal{S}_j|$; i.e., ξ denotes the number of distinct processing stages supported by the considered RAS, across the entire set of its process types. Also, in some of the subsequent developments, the various processing stages Ξ_{jk}, $j = 1, \ldots, n$, $k = 1, \ldots, l(j)$, will be considered in the context of a total ordering imposed on the set $\bigcup_{j=1}^{n} \mathcal{S}_j$; in that case, the processing stages themselves and their corresponding attributes will be indexed by a single index q that runs over the set $\{1, \ldots, \xi\}$ and indicates the position of the processing stage in the considered total order.

Finally, we notice that the RAS class defined above is a variation – actually, a super-class – of the *conjunctive / disjunctive (C/D-)* RAS class defined in [25]; in particular, the RAS class considered in this work allows for internal loops in the process-defining logic, an attribute that was not permitted in the originally defined C/D-RAS. In fact, the perusal of the relevant definitions in [25] will reveal that the results presented in this paper are applicable or easily extensible to other, more complex classes of the RAS taxonomy presented in [25]. However, we have opted to restrict the discussion of the subsequent developments in the particular class defined above for concreteness and ease of exposition.

Modeling the dynamics of the RAS as a Finite State Automaton The dynamics of the RAS $\Phi = < \mathcal{R}, C, \mathcal{P}, \mathcal{A} >$, introduced in the previous paragraph, can be formally described by a *Deterministic Finite State Automaton (DFSA)* ([4]), $G(\Phi) = (S, E, f, \mathbf{s}_0, S_M)$, that is defined as follows:

1. The *state set* S consists of ξ-dimensional vectors $\mathbf{s}$. The components $\mathbf{s}[q]$, $q = 1, \ldots, \xi$, of $\mathbf{s}$ are in one-to-one correspondence with the RAS processing stages, and they indicate the number of process instances executing the corresponding stage in the RAS state modeled by $\mathbf{s}$. Hence, S consists of all the vectors $\mathbf{s} \in (\mathbb{Z}_0{}^+)^{\xi}$ that further satisfy

$$\forall i = 1, \ldots, m, \quad \sum_{q=1}^{\xi} \mathbf{s}[q] \cdot \mathcal{A}(\Xi_q)[i] \leq C_i \tag{1}$$

where $\mathcal{A}(\Xi_q)[i]$ denotes the allocation request for resource R_i that is posed by stage Ξ_q.

2. The *event set* E is the union of the disjoint event sets $E^{\nearrow}$, $\bar{E}$ and $E^{\searrow}$, where: (i) $E^{\nearrow} = \{e_{rp} : r = 0,\ \Xi_p \in \bigcup_{j=1}^{n} \mathcal{V}_j^{\nearrow}\}$, i.e., event e_{rp} represents the *loading* of a new process instance that starts from stage Ξ_p. (ii) $\bar{E} = \{e_{rp} : \exists j \in 1, \ldots, n$ s.t. Ξ_p is a successor of Ξ_r in digraph $\mathcal{G}_j\}$, i.e., e_{rp} represents the *advancement* of a process instance executing stage Ξ_r to a successor stage Ξ_p. (iii)$E^{\searrow} = \{e_{rp} : \Xi_r \in \bigcup_{j=1}^{n} \mathcal{V}_j^{\searrow},\ p = 0\}$, i.e, e_{rp} represents the *unloading* of a finished process instance after executing its last stage Ξ_r.

3. The *state transition function* $f : S \times E \rightarrow S$ is defined by $\mathbf{s}' = f(\mathbf{s}, e_{rp})$, where the components $\mathbf{s}'[q]$ of the resulting state $\mathbf{s}'$ are given by:

$$\mathbf{s}'[q] = \begin{cases} \mathbf{s}[q] - 1 & \text{if} \quad q = r \\ \mathbf{s}[q] + 1 & \text{if} \quad q = p \\ \mathbf{s}[q] & \text{o.w.} \end{cases}$$

Furthermore, $f(\mathbf{s}, e_{rp})$ is a *partial* function defined only if the resulting state $\mathbf{s}' \in S$.

4. The *initial state* $\mathbf{s}_0$ is set equal to $\mathbf{0}$.

5. The *set of marked states* S_M is the singleton $\{\mathbf{s}_0\}$.

Let $\hat{f}$ denote the natural extension of the state transition function f to $S \times E^*$; i.e., for any $\mathbf{s} \in S$ and the empty event string ϵ, $\hat{f}(\mathbf{s}, \epsilon) = \mathbf{s}$, while for any $\mathbf{s} \in S$, $\sigma \in E^*$ and $e \in E$, $\hat{f}(\mathbf{s}, \sigma e) = f(\hat{f}(\mathbf{s}, \sigma), e)$. Also, it is implicitly assumed that $\hat{f}(\mathbf{s}, \sigma e)$ is undefined if any of the one-step transitions that are involved in the right-hand-side recursion are undefined. Then, the behavior of RAS Φ is modeled by the *language* $L(G)$ *generated* by DFSA $G(\Phi)$, i.e., by all strings $\sigma \in E^*$ such that $\hat{f}(\mathbf{s}_0, \sigma)$ is defined. Furthermore, the *reachable subspace* of $G(\Phi)$ is the subset S_r of S defined as follows:

$$S_r \equiv \{\mathbf{s} \in S : \exists \sigma \in L(G) \text{ s.t. } \hat{f}(\mathbf{s}_0, \sigma) = \mathbf{s}\} \tag{2}$$

We also define the *safe subspace* of $G(\Phi)$, S_s, by:

$$S_s \equiv \{\mathbf{s} \in S : \exists \sigma \in E^* \text{ s.t. } \hat{f}(\mathbf{s}, \sigma) = \mathbf{s}_0\} \tag{3}$$

S_s contains those states of S that are co-accessible to the marked state $\mathbf{s}_0$. In the following, we shall denote the complements of S_r and S_s with respect to S by $S_{\bar{r}}$ and $S_{\bar{s}}$, and we shall refer to them as the *unreachable* and *unsafe* subspaces. Finally, S_{xy}, $x \in \{r, \bar{r}\}$, $y \in \{s, \bar{s}\}$, will denote the intersection of the corresponding sets S_x and S_y.

The target behavior of $G(\Phi)$ and the structure of the maximally permissive DAP The target behavior of RAS Φ is expressed by the *marked language* $L_m(G)$, which is defined by means of the marked state $\mathbf{s}_0$, as follows:

$$L_m(G) \ \equiv \ \{\sigma \in L(G) : \hat{f}(\mathbf{s}_0, \sigma) = \mathbf{s}_0\} \tag{4}$$

Equation 4, when combined with all the previous definitions, further implies that the set of states that are accessible under $L_m(G)$ is exactly equal to S_{rs}. Hence, starting from state $\mathbf{s}_0$, a *maximally permissive deadlock avoidance policy* must allow a system-enabled transition to a next state $\mathbf{s}$ if and only if (*iff*) $\mathbf{s}$ belongs to S_{rs}. This characterization of the maximally permissive DAP ensures its

uniqueness for any given RAS instantiation. It also implies that the policy can be effectively implemented through any mechanism that recognizes and rejects the unsafe states that are accessible through one-step transitions from S_{rs}.

Some monotonicities observed by the state safety and unsafety concepts Next we review some additional structure that was introduced in [19] and will play an important role in compressing the reachable state space.

Proposition 1: Consider the (partial) ordering relationship "$\leq$" imposed on the state space S of a given RAS Φ that is defined as follows:

$$\forall \mathbf{s}, \mathbf{s}' \in S, \quad \mathbf{s} \leq \mathbf{s}' \iff (\forall i = 1, \ldots \xi, \quad \mathbf{s}[i] \leq \mathbf{s}'[i]) \quad (5)$$

Then,

1. $\mathbf{s} \in S_s \ \wedge \ \mathbf{s}' \leq \mathbf{s} \implies \mathbf{s}' \in S_s$
2. $\mathbf{s} \in S_u \ \wedge \ \mathbf{s} \leq \mathbf{s}' \implies \mathbf{s}' \in S_u$

□

In light of Proposition 1, we define the concepts of *maximal safe state* and *minimal unsafe state*, that will play an important role in the subsequent developments.

Definition 1: Given a RAS $\Phi =< \mathcal{R}, C, \mathcal{P}, \mathcal{A} >$,

1. a reachable safe state $\mathbf{s} \in S_{rs}$ is *maximal iff* $\neg\exists$ a reachable safe state $\mathbf{s}' \in S_{rs}$ such that $\mathbf{s}' \geq \mathbf{s}$;
2. a reachable unsafe state $\mathbf{s} \in S_{r\bar{s}}$ is *minimal iff* $\neg\exists$ a reachable unsafe state $\mathbf{s}' \in S_{r\bar{s}}$ such that $\mathbf{s}' \leq \mathbf{s}$. □

In the sequel, the set of maximal reachable safe states will be denoted by $\bar{S}_{rs}$, and the set of minimal reachable unsafe states will be denoted by $\bar{S}_{r\bar{s}}$.

Linear separation Finally, we review the concept of linear separation, defined in [19].

Definition 2: Consider two vector sets G and H from a ξ-dimensional vector space V. We shall say that sets G and H are *linearly separated* by a set of k linear inequalities $\{(\mathbf{A}(i,.),\ \mathbf{b}[i]),\ i := \{1, \cdots, k\}\}$ iff

$$\forall \mathbf{g} \in G,\ \forall i \in \{1, \cdots, k\} :\ \mathbf{A}(i,.) \cdot \mathbf{g} \leq \mathbf{b}[i]$$
$$\wedge \quad \forall \mathbf{h} \in H,\ \exists \hat{i} \in \{1, \cdots, k\} :\ \mathbf{A}(\hat{i},.) \cdot \mathbf{h} > \mathbf{b}[\hat{i}] \quad (6)$$

III. The classifier design problem and its simplification

This section considers the problem of synthesizing an effective and compact classifier that separates the reachable safe subspace S_{rs} from the reachable unsafe subspace $S_{r\bar{s}}$ for any RAS Φ that belongs to the class of resource allocation systems considered in this work. We shall proceed by first defining the constructs employed by the sought classifier, and subsequently we shall formally introduce this classifier and prove its capability of achieving the sought separation. In order to alleviate the computational problems arising from the huge cardinality of the underlying state space, we utilize the thinning techniques developed in [19]. To this end, we shall eventually restrict our attention to the subclass of two-layer classifiers with non-negative coefficients, and we shall also prove the capability of this subclass of classifiers of achieving the sought separation. Finally, we prove the applicability of the thinning techniques of [19] to this restricted subclass of classifiers.

Definition 3: Given a point $\mathbf{x}$ and a linear inequality $< \mathbf{a}, b >$, we define:

$$\gamma(\mathbf{x}, \mathbf{a}, b) = \begin{cases} 0, & \text{if } \mathbf{a} \cdot \mathbf{x} \leq b \\ 1, & \text{if } \mathbf{a} \cdot \mathbf{x} > b \end{cases}$$

We also define:

$$\gamma(\mathbf{x}, \begin{bmatrix} \mathbf{a_1}^T \\ \cdots \\ \mathbf{a_k}^T \end{bmatrix}, \begin{bmatrix} b_1 \\ \cdots \\ b_k \end{bmatrix}) = (\gamma(\mathbf{x}, \mathbf{a_1}, b_1), \ldots, \gamma(\mathbf{x}, \mathbf{a_k}, b_k))^T$$

□

Hence, given a vector $\mathbf{x} \in \mathbb{N}^\xi$, a $k \times \xi$ real-valued matrix $\mathbf{A}$, and a vector $\mathbf{b} \in \Re^k$, $\gamma(\mathbf{x}, \mathbf{A}, \mathbf{b})$ can be seen as a transformation $\mathbb{N}^\xi \rightarrow \{0,1\}^k$. To simplify the notation, we shall refer to $\gamma(\mathbf{x}, \mathbf{A}, \mathbf{b})$ as $\gamma_{\mathbf{x}}$ when $\mathbf{A}$ and $\mathbf{b}$ are known. Next, we introduce the two-layer classifier.

Definition 4: Consider two vector sets G and H from a ξ-dimensional vector space V. A two-layer classifier TLC is defined as a 4-tuple $< \mathbf{A}, \mathbf{b}, \mathbf{C}, \mathbf{d} >$ where $\mathbf{A}$ is a $k_1 \times \xi$ real-valued matrix, $\mathbf{b}$ is a vector in $\Re^{k_1}$, $\mathbf{C}$ is a $k_2 \times k_1$ real-valued matrix, and $\mathbf{d}$ is a vector in $\Re^{k_2}$. We say that TLC separates G and H iff

$$\forall \mathbf{g} \in G, \forall i \in \{1, \ldots, k_2\} : \mathbf{C}(i,.) \cdot \gamma(\mathbf{g}, \mathbf{A}, \mathbf{b}) \leq \mathbf{d}[i] \ \wedge$$
$$\forall \mathbf{h} \in H,\ \exists \hat{i} \in \{1, \ldots, k_2\} : \mathbf{C}(\hat{i},.) \cdot \gamma(\mathbf{h}, \mathbf{A}, \mathbf{b}) > \mathbf{d}[\hat{i}] \quad (7)$$

The size of the classifier, $|TLC|$, is determined by the total number of operations required to classify a given vector. Thus, $|TLC| = (2 \cdot \xi + 1) \cdot k_1 + (2 \cdot k_1 + 1) \cdot k_2$. □

The first layer of the classifier of Definition 4 implements the transformation $\gamma(., \mathbf{A}, \mathbf{b})$, whereas the second layer acts as a linear separator for the sets $I(G)$ and $I(H)$, where $I(G) \equiv \{\mathbf{z} \in \{0,1\}^k \mid \exists \mathbf{g} \in G \text{ s.t. } \mathbf{z} = \gamma(\mathbf{g}, \mathbf{A}, b)\}$ and a similar definition applies for $I(H)$.

To illustrate the size of the classifier, we notice that: (i) the first layer of the classifier $(\mathbf{A}, \mathbf{b})$ has k_1 linear inequalities, and (ii) to evaluate each of these linear inequalities, ξ multiplication operations, ξ addition operations and one comparison operation are required. Therefore, $(2 \cdot \xi + 1) \cdot k_1$ operations are required by the first layer to evaluate a given vector. Similar analysis can be applied to the second layer to see that $(2 \cdot k_1 + 1) \cdot k_2$ operations are required by the second layer to classify a given vector. Therefore, we can conclude that $(2 \cdot \xi + 1) \cdot k_1 + (2 \cdot k_1 + 1) \cdot k_2$ operations are required by a two-layer classifier to classify a given vector.

Finally, we notice that the linear classifier $< \mathbf{A}, \mathbf{b} >$ is a special case of the two-layer classifier $< \mathbf{A}, \mathbf{b}, \mathbf{C}, \mathbf{d} >$ where the second layer $< \mathbf{C}, \mathbf{d} >$ is given by the linear inequality $\sum_{i=1}^{k_1} \gamma_i \leq 0$.

Given the above definitions, the problem addressed in this manuscript can be succinctly stated as follows:

Definition 5: – **The considered classification problem:** Given a RAS Φ, construct a minimum-sized two-layer classifier for the vector sets corresponding to the subspaces S_{rs} and $S_{r\bar{s}}$, i.e., the reachable safe and the reachable unsafe states of the considered RAS Φ. □

In order to prove the capability of the two-layer classifier to achieve the sought separation, first we present the following proposition which essentially collects a set of results that were established in [19]:

Proposition 2: For any two sets $G,\ H \subseteq \{0,1\}^m$ such that $G \cap H = \emptyset$, there exists a system of linear inequalities $\{\mathbf{A}(i,.) \cdot \mathbf{x} \leq \mathbf{b}(i),\ i := 1,\ldots,k\}$ that can function as a linear separator of these two sets.

Furthermore, if it also holds that

$$\forall \mathbf{x} \in G, \forall \mathbf{x}' \in \{0,1\}^m, \mathbf{x}' \leq \mathbf{x} \implies \mathbf{x}' \in G \tag{8}$$

then, the set of *minimal* linear separators for G and H – i.e., the set of linear separators that employ the smallest possible nubmber of linear inequalities – will contain a separator $< \mathbf{A}, \mathbf{b} >$ with *non-negative* coefficients.

Proof: As already mentioned, both of the results of Proposition 2 have been established in [19]. More specifically, the first result is established by Proposition 2 and Corollary 2 in [19], and it is an immediate consequence of the fact that every extreme point of the hypercube that is defined by the set of binary vectors of any given dimensionality m, can be separated from the remaining extreme points of that hypercube by a single hyperplane. The second result, i.e., the existence of a minimal linear separator with non-negative coefficients, is established by Proposition 4 of [19], where it is obtained through an elaborate application of Farkas' lemma. □

Also, the following functions will be useful in the subsequent developments.

Definition 6: Given the set of reachable states S_r of a RAS instance Φ with state dimensionality ξ, define

- $CAP_i \equiv \begin{cases} 0, & i = 0 \\ \sup\{\mathbf{x} \in S_r : \mathbf{x}[i]\}, & i := 1,\ldots,\xi \end{cases}$
- $l(m) \equiv \sum_{j=0}^{m} CAP_j,\ m := 0,\ldots,\xi.$
- $v(i) \equiv m,\ i := 1,\ldots,\sum_{j=1}^{\xi} CAP_j \wedge l(m-1) < i \leq l(m).$

□

In more natural terms, for every $i \in \{1,\ldots,\xi\}$, CAP_i represents the maximum number of process instances in processing stage i across all the reachable states $\mathbf{s} \in S_r$. CAP_0 is defined only for mathematical convenience. On the other hand, the functions $l(m),\ m \in \{0,\ldots,\xi\}$, and $v(i),\ i \in \{1,\ldots,\sum_{j=1}^{\xi} CAP_j\}$, essentially establish an indexing scheme that will be useful for the formal characterization of the sought classifier. More specifically, these two functions are employed in the statement and proof of the following lemma:

Lemma 1: Consider a RAS instance Φ from the RAS class defined in Section II, and the corresponding set of reachable states S_r. Also, let ξ denote the dimensionality of the state vectors in S_r, and define the real matrix $\mathbf{A}$ and the vector $\mathbf{b}$ as follows:

$$\forall i \in \{1,\ldots,\sum_{q=1}^{\xi} CAP_q\},\ \forall j \in \{1,\ldots,\xi\},$$
$$\mathbf{A}(i,j) = \begin{cases} 1, & \text{if } j = v(i) \\ 0, & \text{o.w.} \end{cases} \tag{9}$$
$$\forall i \in \{1,\ldots,\sum_{l=1}^{\xi} CAP_l\},$$
$$\mathbf{b}[i] = i - l(v(i)-1) - 1 \tag{10}$$

The quantities CAP_q, and the functions $l(\cdot)$, $v(\cdot)$ that appear in the above two equations are those established in Definition 6. Then,

$$\forall \mathbf{s}_1, \mathbf{s}_2 \in S_r,\quad \mathbf{s}_1 \neq \mathbf{s}_2 \Longrightarrow \gamma(\mathbf{s}_1, \mathbf{A}, \mathbf{b}) \neq \gamma(\mathbf{s}_2, \mathbf{A}, \mathbf{b}) \tag{11}$$

Proof: First we notice that Equations 9 and 10 are essentially a more compact and programmatic representation of the following structure:

$$\mathbf{A} = \begin{bmatrix} 10\cdots0 \\ 10\cdots0 \\ \cdots \\ 10\cdots0 \\ 01\cdots0 \\ \cdots \\ 01\cdots0 \\ \cdots \\ \cdots \\ 00\cdots1 \\ \cdots \\ 00\cdots1 \end{bmatrix} \quad \wedge \quad \mathbf{b} = \begin{bmatrix} 0 \\ 1 \\ \cdots \\ CAP_1 - 1 \\ 0 \\ \cdots \\ CAP_2 - 1 \\ \cdots \\ \cdots \\ 0 \\ \cdots \\ CAP_\xi - 1 \end{bmatrix} \tag{12}$$

Next, consider two states $\mathbf{s}_1, \mathbf{s}_2 \in S_r$, such that $\mathbf{s}_1 \neq \mathbf{s}_2$. Therefore, there exists $i \in \{1,\ldots,\xi\}$ such that $\mathbf{s}_1[i] \neq \mathbf{s}_2[i]$. Without loss of generality, assume that $\mathbf{s}_1[i] > \mathbf{s}_2[i]$ (otherwise, simply reverse the roles of $\mathbf{s}_1$ and $\mathbf{s}_2$ in the subsequent argument). Then, consider the inequality defined by the k-th row of matrix $\mathbf{A}$ and the corresponding component of vector $\mathbf{b}$, where $k = \sum_{q=0}^{i-1} CAP_q + \mathbf{s}_2[i] + 1$. From the structure implied by Equation 12, it follows that $\forall \mathbf{s} \in S_r,\ \mathbf{A}[k,\cdot] \cdot \mathbf{s} = \mathbf{s}[i]$ and $\mathbf{b}[k] = \mathbf{s}_2[i]$. Since it has been assumed that $\mathbf{s}_1[i] > \mathbf{s}_2[i]$, Definition 3 implies that $\gamma_{\mathbf{s}_1}[k] = 1$ and $\gamma_{\mathbf{s}_2}[k] = 0$, and our result has been established. □

The next example concretizes the constructs that were introduced in the previous lemma.

Example 1 Assume that for a given RAS Φ, $S_{rs} \equiv \{\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3, \mathbf{s}_4, \mathbf{s}_5\} \equiv \{(0,0),\ (1,0),\ (2,0),\ (0,1),\ (0,2)\}$ and $S_{r\bar{s}} \equiv \{\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \mathbf{u}_4\} \equiv \{(1,1),\ (2,1),\ (1,2),\ (2,2)\}$. The corresponding set of reachable states S_r is plotted in Figure 1. We can see that $CAP_1 = CAP_2 = 2$. Furthermore, the matrix $\mathbf{A}$ and the vector $\mathbf{b}$ that are defined by Equations 9 and 10 (or, equivalently, by Equation 12) for this particular case, induce the following set of inequalities on the RAS state: $\{x_1 \leq 0,\ x_1 \leq 1,\ x_2 \leq 0,\ x_2 \leq 1\}$. These

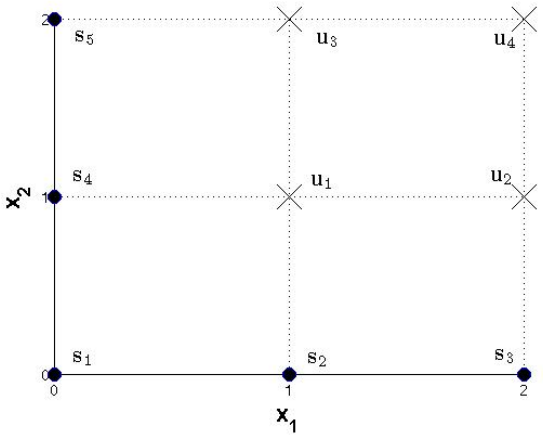

Fig. 1. A plot of the sets S_{rs} and $S_{r\bar{s}}$ given in Example 1

TABLE I
THE INDICATORS OF THE STATES OF EXAMPLE 1 W.R.T. THE LINEAR INEQUALITIES GIVEN AT EACH COLUMN

	$x_1 \leq 0$	$x_1 \leq 1$	$x_2 \leq 0$	$x_2 \leq 1$
$\mathbf{s}_1$	0	0	0	0
$\mathbf{s}_2$	1	0	0	0
$\mathbf{s}_3$	1	1	0	0
$\mathbf{s}_4$	0	0	1	0
$\mathbf{s}_5$	0	0	1	1
$\mathbf{u}_1$	1	0	1	0
$\mathbf{u}_2$	1	1	1	0
$\mathbf{u}_3$	1	0	1	1
$\mathbf{u}_4$	1	1	1	1

inequalities, when combined with the transformation introduced in Definition 3, result in the indicator vectors shown in Table I. As established by Lemma 1, all the indicator vectors are different from each other. □

The next proposition plays a central role in this work.

Proposition 3: Given a RAS instance Φ and the corresponding state sets S_{rs} and $S_{r\bar{s}}$, there exists a two-layer classifier that is capable of separating these two subspaces.

Proof: Consider the pair $(\mathbf{A}, \mathbf{b})$ defined in Lemma 1 and the image sets $I(S_{rs})$, $I(S_{r\bar{s}})$ of the sets S_{rs} and $S_{r\bar{s}}$ that are defined by the pair $(\mathbf{A}, \mathbf{b})$ and the binary transformation introduced in Definition 3. Since both sets S_{rs} and $S_{r\bar{s}}$ are subsets of the reachable space S_r, Lemma 1 implies that $I(S_{rs}) \cap I(S_{r\bar{s}}) = \emptyset$. But then, Proposition 2 implies that there exists a set of linear inequalities $< \mathbf{C}, \mathbf{d} >$ that linearly separates $I(S_{rs})$ and $I(S_{r\bar{s}})$, and the result is established. □

Example 1 (cont.) It can be easily checked inTable I that $I(S_{rs}) \equiv \{(0,0,0,0),\ (1,0,0,0),\ (1,1,0,0),\ (0,0,1,0),\ (0,0,1,1)\}$ and $I(S_{r\bar{s}}) \equiv \{(1,0,1,0),\ (1,1,1,0),\ (1,0,1,1),\ (1,1,1,1)\}$, and, clearly, $I(S_{rs}) \cap I(S_{r\bar{s}}) = \emptyset$. A linear inequality that separates $I(S_{rs})$ and $I(S_{r\bar{s}})$ is: $2 \cdot x_1 + x_3 + x_4 \leq 2$. □

In order to cope with the huge cardinality of the state spaces involved in the considered RAS class, we shall utilize the thinning techniques introduced in [19]. These techniques make it possible to build a two-layer classifier for S_{rs} and $S_{r\bar{s}}$ by focusing the classifier design on smaller vector sets in terms of, both, cardinality and dimensionality. However, the employment of these techniques requires the restriction of the coefficients of the two-layer classifier to non-negative values. Next we show that the class of the two-layer classifiers that results from this restriction remains complete with respect to the classification problem of Definition 5.[3]

We start with the following lemma that plays an important role in establishing the capability of the two-layer classifier with non-negative coefficients to achieve the sought separation. Furthermore, this lemma is crucial for the development of a set of heuristics that will be introduced in Section V.

Lemma 2: A two-layer classifier $TLC =< \mathbf{A}, \mathbf{b}, \mathbf{C}, \mathbf{d} >$ with $\mathbf{C} \geq \mathbf{0}$ and $\mathbf{d} \geq \mathbf{0}$ is capable of separating S_{rs} and $S_{r\bar{s}}$ if $\mathbf{A}$ and $\mathbf{b}$ are constructed such that $\nexists(\gamma_{\mathbf{s}} \in I(S_{rs}),\ \gamma_{\mathbf{u}} \in I(S_{r\bar{s}}))$ with $\gamma_{\mathbf{s}} \geq \gamma_{\mathbf{u}}$.

Proof: Suppose that $\mathbf{A}$ and $\mathbf{b}$ are constructed such that $\nexists(\gamma_{\mathbf{s}} \in I(S_{rs}),\ \gamma_{\mathbf{u}} \in I(S_{r\bar{s}}))$ s.t. $\gamma_{\mathbf{s}} \geq \gamma_{\mathbf{u}}$. Let $\hat{I}(S_{rs}) \equiv \{\hat{\gamma}_{\mathbf{s}} \in \{0,1\}^{k_1} \mid \exists \gamma_{\mathbf{s}'} \in I(S_{rs})$ s.t. $\mathbf{0} \leq \hat{\gamma}_{\mathbf{s}} \leq \gamma_{\mathbf{s}'}\}$. By assumption, $\hat{I}(S_{rs}) \cap I(S_{r\bar{s}}) = \emptyset$. Also by construction, $\hat{I}(S_{rs})$ contains all possible chains of integer vectors between the origin $\mathbf{0}$ and its maximal elements. Therefore, application of Proposition 2 on the sets $\hat{I}(S_{rs})$ and $I(S_{r\bar{s}})$, implies the existence of a linear separator $< \mathbf{C}, \mathbf{d} >$ with non-negative coefficients for the two sets. Since $I(S_{rs}) \subseteq \hat{I}(S_{rs})$, we can see that $< \mathbf{C}, \mathbf{d} >$ linearly separates $I(S_{rs})$ and $I(S_{r\bar{s}})$. □

Now we are ready to state and prove the main result of this section.

Proposition 4: Given the two sets S_{rs} and $S_{r\bar{s}}$ for any RAS Φ considered in this work, there exists a two-layer classifier with non-negative coefficients that is capable of separating the two subspaces.

Proof: Construct the first layer $< \mathbf{A}, \mathbf{b} >$ as in Equations 9, 10. It can be seen that

$$\forall \mathbf{s} \in S_r,\ \forall i \in \{1, \ldots \xi\},\ \mathbf{s}[i] = \sum_{j=l(v(i)-1)+1}^{l(v(i))} \gamma_{\mathbf{s}}[j] \qquad (13)$$

For the sake of contradiction, assume that $\exists(\gamma_{\mathbf{s}} \in I(S_{rs}),\ \gamma_{\mathbf{u}} \in I(S_{r\bar{s}}))$ s.t. $\gamma_{\mathbf{s}} \geq \gamma_{\mathbf{u}}$. Then, $\forall j_1, j_2 \in \{1, \ldots, k_1\}$ s. t. $j_1 \leq j_2$, $\sum_{j=j_1}^{j_2} \gamma_{\mathbf{s}}[j] \geq \sum_{j=j_1}^{j_2} \gamma_{\mathbf{u}}[j]$. Therefore, from Equation 13, $\forall i \in \{1, \ldots \xi\} : \mathbf{s}[i] \geq \mathbf{u}[i]$ where $\mathbf{s} \in S_{rs}$ and $\mathbf{u} \in S_{r\bar{s}}$; this result violates the monotonicity property of Proposition 1 in Section II. Therefore, $\nexists(\gamma_{\mathbf{s}} \in I(S_{rs}),\ \gamma_{\mathbf{u}} \in I(S_{r\bar{s}}))$ s.t. $\gamma_{\mathbf{s}} \geq \gamma_{\mathbf{u}}$. Applying Lemma 2 on $I(S_{rs})$ and $I(S_{r\bar{s}})$, we can see that there exists a linear separator $< \mathbf{C}, \mathbf{d} >$ with non-negative coefficients

[3]On the other hand, contrary to the case considered in [19], we have not been able to provide a guarantee that the restriction imposed on the classifier coefficients will not affect the size of the minimal classifier.

for the two sets. Since, by their definition, $\mathbf{A} \geq \mathbf{0}$ and $\mathbf{b} \geq \mathbf{0}$, the sought result is established. □

From here on, we restrict our attention to two-layer classifiers with non-negative coefficients. Next, we recast the thinning operations introduced in [19] so that they apply for the two-layer classifier, and we establish their validity in the context of this new classification problem. The presented results are similar, in spirit, to their counterparts in [19], although the relevant proofs differ in their technicalities. Therefore, in the rest of this section we focus primarily on the motivation and formal statement of these results, while the corresponding proofs are collected in an Appendix.

Thinning the set $S_{r\bar{s}}$ by focusing on its "boundary" to the reachable and safe subspace As observed in the characterization of the maximally permissive DAP in Section II, the effective implementation of this policy for any given RAS Φ is equivalent to the recognition and the blockage of transitions from the safe to the unsafe region of the underlying state space S. Thus, we can focus on the subset of unsafe states that are reachable from some safe state in a single transition. If the access to this subset is blocked, then access to the whole set of unsafe states $S_{r\bar{s}}$ is automatically blocked. In the following, we shall denote this subset of $S_{r\bar{s}}$ by $S^b_{r\bar{s}}$ and we shall refer to its elements as the *"boundary"* reachable unsafe states.

Thinning the sets S_{rs} and $S^b_{r\bar{s}}$ by respectively focusing on their maximal and minimal elements Under the adopted non-negativity restriction for the parameters of the sought classifier, it is possible to obtain a two-layered classifier for the entire sets S_{rs} and $S^b_{r\bar{s}}$, by focusing the classifier design process only on the maximal elements of the first set, $\bar{S}_{rs}$, and the minimal elements of the second set, $\bar{S}^b_{r\bar{s}}$. This result is formally stated in the following proposition.

Proposition 5: Any two-layer classifier $TLC = < \mathbf{A}, \mathbf{b}, \mathbf{C}, \mathbf{d} >$ with non-negative coefficients that separates the sets $\bar{S}_{rs}$ and $\bar{S}^b_{r\bar{s}}$, is also an effective separator for the entire sets S_{rs} and $S^b_{r\bar{s}}$.

Converting the separation problem of $\bar{S}_{rs}$ and $\bar{S}^b_{r\bar{s}}$ to an equivalent separation problem of reduced dimensionality In our numerical experimentation we have consistently encountered a situation where many components of the vectors included in the set $\bar{S}_{rs}$ are consistently greater than or equal to the corresponding components of the vectors included in the set $\bar{S}^b_{r\bar{s}}$. Next, we show that the removal of these components from further consideration, through the orthogonal projection of the vector sets $\bar{S}_{rs}$ and $\bar{S}^b_{r\bar{s}}$ to the subspace defined by the remaining components, retains all the information that is necessary for the development of a two-layer classifier that will separate effectively the original state subsets S_{rs} and $S_{r\bar{s}}$. To formalize the subsequent discussion, let V denote the ξ-dimensional vector space supporting the vector sets $\bar{S}_{rs}$ and $\bar{S}^b_{r\bar{s}}$, L denote the set of dimensions of space V, and L_0 denote the set of dimensions that have the property that $\forall \mathbf{s} \in \bar{S}_{rs},\ \forall \mathbf{u} \in \bar{S}^b_{r\bar{s}} :\ \mathbf{s}[i] \geq \mathbf{u}[i]$. This set of dimensions is removed by the proposed projection P. Let $L_P \equiv L \backslash L_0$, and V_P denote the $|L_P|$-dimensional subspace supporting the projection P. Also, let $\Gamma : \mathbb{N} \rightarrow \mathbb{N}$ be a bijection that maps the elements of the dimension set L_P to the dimensions of subspace V_P. Finally, let $P(\bar{S}_{rs})$ and $P(\bar{S}^b_{r\bar{s}})$ denote respectively the images of the sets $\bar{S}_{rs}$ and $\bar{S}^b_{r\bar{s}}$ under P. Then, we have the following proposition:

Proposition 6: Consider a RAS instance Φ and its corresponding sets $\bar{S}_{rs}$ and $\bar{S}^b_{r\bar{s}}$. Furthermore, suppose that the two-layer classifier with non-negative coefficients $TLC^Q =< \mathbf{A}^Q, \mathbf{b}^Q, \mathbf{C}^Q, \mathbf{d}^Q >$ separates the projected sets $P(\bar{S}_{rs})$ and $P(\bar{S}^b_{r\bar{s}})$ in subspace V_P. Then a two-layer classifier TLC^H for the original sets $\bar{S}_{rs}$ and $\bar{S}^b_{r\bar{s}}$ is induced from TLC^Q according to the following equation:

$$\begin{aligned} &\forall i,\ \forall j \in L_0 :\ \mathbf{A}^H(i,j) = 0 \\ &\wedge\ \ \forall i,\ \forall j \in L_P :\ \mathbf{A}^H(i,j) = \mathbf{A}^Q(i,\Gamma(j))\ \ \wedge \\ &\mathbf{b}^H = \mathbf{b}^Q\ \ \wedge\ \ \mathbf{C}^H = \mathbf{C}^Q\ \ \wedge\ \ \mathbf{d}^H = \mathbf{d}^Q \end{aligned} \tag{14}$$

□

As remarked at the beginning of this paragraph, the practical implication of Proposition 6 is that we can construct a two-layer classifier TLC^H for the sets $\bar{S}^b_{r\bar{s}}$ and $\bar{S}_{rs}$ by first developing a two-layer classifier TLC^Q for the projected sets $P(\bar{S}^b_{r\bar{s}})$ and $P(\bar{S}_{rs})$, and subsequently constructing TLC^H from TLC^Q through Equation 14. Furthermore, Equation 14 also implies that the construction of classifier TLC^H from classifier TLC^Q maintains the non-negativity of the coefficients, and therefore, it can be carried out on the thinned sets $\bar{S}^b_{r\bar{s}}$ and $\bar{S}_{rs}$ and their projections, without compromising the validity of the derived classifiers for the broader sets of insterest, $S^b_{r\bar{s}}$ and S_{rs}. It remains to establish that there will always exist a two-layer classifier with non-negative coefficients, TLC^Q, for the projected sets $P(\bar{S}^b_{r\bar{s}})$ and $P(\bar{S}_{rs})$; this is done by the following proposition.

Proposition 7: For any RAS instance Φ from the RAS class considered in this work, there will always exist a two-layer classifier with non-negative coefficients, TLC^Q, that separates the projected sets $P(\bar{S}^b_{r\bar{s}})$ and $P(\bar{S}_{rs})$ in space V_P. □

On the other hand, projection P is not bijective, and therefore, it may introduce some redundancy among the elements of $P(\bar{S}_{rs})$ and the elements of $P(\bar{S}^b_{r\bar{s}})$. Also the removal of the components in L_0 can introduce some dominance among the elements of both sets with respect to the ordering '$\leq$'. Hence, the redundant and the non-maximal (resp., non-minimal) elements should be removed from the set $P(\bar{S}_{rs})$ (resp., $P(\bar{S}^b_{r\bar{s}})$). The resulting sets will be denoted by $\overline{P(\bar{S}_{rs})}$ and $\overline{P(\bar{S}^b_{r\bar{s}})}$.[4] Finally, the image of these sets under the transformation of the first-layer linear inequalities will be denoted by $I(\overline{P(\bar{S}_{rs})})$ and $I(\overline{P(\bar{S}^b_{r\bar{s}})})$.

[4]The fact that this additional "thinning" of the sets $P(\bar{S}_{rs})$ and $P(\bar{S}^b_{r\bar{s}})$ does not compromise the effectiveness of the obtained separator TLC^Q with respect to the separation of the sets $P(\bar{S}_{rs})$ and $P(\bar{S}^b_{r\bar{s}})$, can be argued on the basis of the non-negativity of the coefficients of the target separator TLC^Q; c.f., Proposition 5.

IV. Synthesizing classifier $TLC^{\mathcal{Q}}$ through Mathematical Programming

In this section, we provide a Mixed Integer Programming (MIP) formulation [20] for the construction of the separator $TLC^{\mathcal{Q}}$, that was specified in Section III. In the subsequent discussion, let $m_s \equiv |\overline{P(\bar{S}_{rs})}|$, $m_u \equiv |\overline{P(\bar{S}^b_{r\bar{s}})}|$, $\Upsilon \equiv \overline{P(\bar{S}_{rs})} \cup \overline{P(\bar{S}^b_{r\bar{s}})}$, and $n \equiv |L_P|$. In addition to $\overline{P(\bar{S}_{rs})}$ and $\overline{P(\bar{S}^b_{r\bar{s}})}$, the following parameters are used to provide additional control to the MIP formulation:

- The parameter k_1 provides an upper bound for the number of inequalities employed by the first layer. Such an upper bound is readily obtained as $k_1 = \sum_{i=1}^{n} CAP_i$ from Proposition 4 where CAP_i is computed in the reduced subspace V_P. We shall refer to the term $\sum_{i=1}^{n} CAP_i$ as τ.
- The parameter k_2 provides an upper bound for the number of inequalities employed by the second layer. Such an upper bound is readily obtained as $k_2 = m_u$ from [19]-Section IV.
- A strictly positive parameter ϵ that controls the minimum distance of the points from the separating hyperplanes and should be priced sufficiently close to zero. This parameter can be perceived as a "degree of separation". In order to guarantee that the employed value of ϵ is not unnecessarily large to the extent that it compromises the minimality of the derived solution, one should re-solve the proposed formulation for a sequence $\{\epsilon_i\}$ such that $\epsilon_i \to 0^+$, and consider the stability of the size of the obtained supervisors.
- A strictly positive parameter M that is useful for the modeling of the separation logic in the proposed MIP formulation and must take sufficiently large values. This is the notorious "big-M" parameter that appears in many MIP formulations. A more detailed discussion on its role in the proposed formulation, as well as on its appropriate pricing, is provided in later parts of this section.

The variables employed by the proposed formulation are as follows:

- $\alpha[i]$, $i := 1, \ldots, k_1$, is a binary variable priced to one iff the i-th linear inequality of the first layer is used for separation.
- $\beta[i]$, $i := 1, \ldots, k_2$, is a binary variable priced to one iff the i-th linear inequality of the second layer is used for separation.
- $(\mathbf{A}(i,.), \mathbf{b}[i])$, $i := 1, \ldots, k_1$, are the coefficients to be employed by the i-th linear inequality of the first layer.
- $(\mathbf{C}(i,.), \mathbf{d}[i])$, $i := 1, \ldots, k_2$, are the coefficients to be employed by the i-th linear inequality of the second layer.
- $\gamma_{\mathbf{x}}[i]$, $\mathbf{x} \in \Upsilon$, $i := 1, \ldots, k_1$, is a binary variable priced to one iff $\mathbf{x}$ violates the linear inequality $\mathbf{A}(i,.) \cdot \mathbf{x} \leq \mathbf{b}[i]$.
- $\delta_{\mathbf{x}}[i]$, $\mathbf{x} \in \Upsilon$, $i := 1, \ldots, k_2$, is a binary variable priced to one iff γ_x violates the linear inequality $\mathbf{C}(i,.) \cdot \gamma_x \leq \mathbf{d}[i]$.
- $\mathbf{r}_{\mathbf{x}}(i,j)$ is a real variable that is equal to $C(i,j) \cdot \gamma_{\mathbf{x}}[j]$.
- k_1^* is an integer variable that represents the total number of active linear inequalities employed by the first layer.
- k_2^* is an integer variable that represents the total number of active linear inequalities employed by the second layer.
- $k_{1\cdot 2}^*$ is an integer variable that is equal to $k_1^* \cdot k_2^*$.
- $\mathbf{z}[i]$, $i := 0, \ldots, k_1$, is an integer variable used to resolve the non-linearity of the term $k_1^* \cdot k_2^*$, as it will be revealed in the sequel.

Finally, the formulation itself takes the following form:

$$\text{Min } (2 \cdot n + 1) \cdot k_1^* + k_2^* + 2 \cdot k_{1\cdot 2}^* \tag{15}$$

$$\begin{aligned} &\forall \mathbf{x} \in \{\Upsilon\},\ \forall i \in \{1, \cdots, k_1\}: \\ &\epsilon + M \cdot (\gamma_{\mathbf{x}}[i] - 1) \leq \mathbf{A}(i,.) \cdot \mathbf{x} - \mathbf{b}[i] \leq M \cdot \gamma_{\mathbf{x}}[i] \end{aligned} \tag{16}$$

$$\begin{aligned} &\forall \mathbf{x} \in \{\Upsilon\},\ \forall i \in \{1, \ldots, k_2\},\ \forall j \in \{1, \ldots, k_1\}: \\ &C(i,j) + \gamma_{\mathbf{x}}[j] - 1 \leq \mathbf{r}_{\mathbf{x}}(i,j) \leq C(i,j) \end{aligned} \tag{17}$$

$$\begin{aligned} &\forall \mathbf{x} \in \{\Upsilon\},\ \forall i \in \{1, \ldots, k_2\},\ \forall j \in \{1, \ldots, k_1\}: \\ &0 \leq \mathbf{r}_{\mathbf{x}}(i,j) \leq \gamma_{\mathbf{x}}[j] \end{aligned} \tag{18}$$

$$\begin{aligned} &\forall \mathbf{x} \in \{\Upsilon\},\ \forall i \in \{1, \cdots, k_2\}: \\ &\epsilon + M \cdot (\delta_{\mathbf{x}}[i] - 1) \leq \sum_{j=1}^{k_1} \mathbf{r}_{\mathbf{x}}(i,j) - \mathbf{d}[i] \leq M \cdot \delta_{\mathbf{x}}[i] \end{aligned} \tag{19}$$

$$\forall \mathbf{s} \in \overline{P(\bar{S}_{rs})}: \ \sum_{i=1}^{k_2} \delta_{\mathbf{s}}[i] = 0 \tag{20}$$

$$\forall \mathbf{u} \in \overline{P(\bar{S}^b_{r\bar{s}})}: \ \sum_{i=1}^{k_2} \delta_{\mathbf{u}}[i] \geq 1 \tag{21}$$

$$\forall i \in \{1, \ldots, k_1\},\ \forall j \in \{1, \ldots, n\}: \ 0 \leq \mathbf{A}(i,j) \leq \alpha[i] \tag{22}$$

$$\forall i \in \{1, \ldots, k_1\}: \ 0 \leq \mathbf{b}(i) \leq \alpha[i] \tag{23}$$

$$\forall i \in \{1, \ldots, k_2\},\ \forall j \in \{1, \ldots, k_1\}: \ 0 \leq \mathbf{C}(i,j) \leq \beta[i] \tag{24}$$

$$\forall i \in \{1, \ldots, k_2\}: \ 0 \leq \mathbf{d}(i) \leq \beta[i] \tag{25}$$

$$k_1^* = \sum_{i=1}^{k_1} \alpha[i] \quad \wedge \quad k_2^* = \sum_{i=1}^{k_2} \beta[i] \tag{26}$$

$$\mathbf{z}[0] = 0 \ \wedge \ \forall i \in \{1, \ldots, k_1\}: \ 0 \leq \ \mathbf{z}[i] - \mathbf{z}[i-1] \ \leq k_2^* \tag{27}$$

$$\begin{aligned} &\forall i \in \{1, \ldots, k_1\}: \\ &k_2^* + M \cdot (\alpha[i] - 1) \leq \ \mathbf{z}[i] - \mathbf{z}[i-1] \ \leq M \cdot \alpha[i] \\ &\wedge \ k_{1\cdot 2}^* = \mathbf{z}[k_1] \end{aligned} \tag{28}$$

$$\forall i:\ \alpha[i],\ \beta[i],\ \gamma_{\mathbf{x}}[i],\ \delta_{\mathbf{x}}[i] \in \{0,1\},\quad \mathbf{z}[i] \in {\mathbb{Z}_0}^{+} \tag{29}$$

The validity of the above MIP formulation as a construction tool for the sought classifier TLC^Q can be established as follows: First, the reader should notice that Equation 15 defines the objective of the formulation as the minimization of the size of the classifier. Also, Equation 29 enforces the binary nature of the variables $\alpha[i]$, $\beta[i]$, $\gamma_{\mathbf{x}}[i]$, and $\delta_{\mathbf{x}}[i]$ and the discrete nature of $\mathbf{z}[i]$. On the other hand, the constraints of Equations 22 through 25 enforce (i) the non-negativity of the matrices $\mathbf{A}$ and $\mathbf{C}$ and the vectors $\mathbf{b}$ and $\mathbf{d}$ in the returned solution, and also (ii) the requirement that the coefficients of the inactive inequalities should be set to zero. An additional implication of the constraints enforced through Equations 22– 25 is the restriction of all the elements of the matrices $\mathbf{A}$ and $\mathbf{C}$ and the elements of the vectors $\mathbf{b}$ and $\mathbf{d}$ to be no greater than one. This effect does not compromise the generality of the obtained solution(s) since these elements can always be normalized to have values no greater than one. The first-layer transformation is performed by Equation 16. In particular, we can see that by setting $\gamma_{\mathbf{x}}[i] = 0$, Equation 16 can be rewritten as $\epsilon - M \leq \mathbf{A}(i,.) \cdot \mathbf{x} - \mathbf{b}[i] \leq 0$, whereas by setting $\gamma_{\mathbf{x}}[i] = 1$, it can be rewritten as $\epsilon \leq \mathbf{A}(i,.) \cdot \mathbf{x} - \mathbf{b}[i] \leq M$. Therefore, provided that M is large enough and ϵ is small enough, if $\mathbf{A}(i,.) \cdot \mathbf{x} \leq \mathbf{b}(i)$, then $\gamma_{\mathbf{x}}[i]$ is forced to zero, whereas if $\mathbf{A}(i,.) \cdot \mathbf{x} > \mathbf{b}(i)$, then $\gamma_{\mathbf{x}}[i]$ is forced to one. In a similar way, Equation 19 sets the second layer indicators by forcing $\delta_{\mathbf{x}}[i]$ to zero if $\sum_{j=1}^{k_1} \mathbf{r}_{\mathbf{x}}(i,j) \leq \mathbf{d}(i)$ and to one if $\sum_{j=1}^{k_1} \mathbf{r}_{\mathbf{x}}(i,j) > \mathbf{d}(i)$.

The constraints of Equations 17 and 18 resolve the non-linearity of the term $C(i,j) \cdot \gamma_{\mathbf{x}}[j]$ by introducing the variable $\mathbf{r}_{\mathbf{x}}(i,j)$ to represent this non-linear term in the following way: Equation 18 forces $\mathbf{r}_{\mathbf{x}}(i,j)$ to zero if $\gamma_{\mathbf{x}}[j]$ equals zero. On the other hand, Equation 17 sets $\mathbf{r}_{\mathbf{x}}(i,j) = C(i,j)$ if $\gamma_{\mathbf{x}}[j]$ equals one.

The constraints of Equation 20 enforce the requirement that the image of each safe state under the first-layer transformation should satisfy all the linear inequalities of the second layer, whereas the constraints of Equation 21 enforce the requirement that the image of each unsafe state under the first-layer transformation should violate at least one linear inequality of the second layer.

Equation 26 introduces the variables k_1^* and k_2^* as the number of active linear inequalities in the first layer and the second layer, respectively.

Equations 27–28 are used to resolve the non-linearity of the term $k_1^* \cdot k_2^*$ by introducing the variable $k_{1.2}^*$ to represent this non-linear term. More specifically, $\mathbf{z}[i] - \mathbf{z}[i-1]$ equals zero if $\alpha[i] = 0$ and equals k_2^* if $\alpha[i] = 1$. Therefore, by setting $\mathbf{z}[0] = 0$, we can see that $\mathbf{z}[k_1] = k_1^* \cdot k_2^*$. In particular, if $\alpha[i] = 0$, Equation 28 can be rewritten as $k_2^* - M \leq \mathbf{z}[i] - \mathbf{z}[i-1] \leq 0$; taking Equation 27 into account, we can see that $\mathbf{z}[i] - \mathbf{z}[i-1] = 0$ in this case. On the other hand, if $\alpha[i] = 1$, Equation 28 can be rewritten as $k_2^* \leq \mathbf{z}[i] - \mathbf{z}[i-1] \leq M$; again taking Equation 27 into account, we can see that $\mathbf{z}[i] - \mathbf{z}[i-1] = k_2^*$ in this case.

The above discussion also reveals the condition for the proper pricing of the parameter M. More specifically, the analysis of the pricing of $\gamma_{\mathbf{x}}[j]$ implies that

$$M \ \geq\ \sup\{\mathbf{A}(i,.) \cdot \mathbf{x} - \mathbf{b}(i)\} \tag{30}$$

where the the supremum is taken over all pairs of vectors $\mathbf{x}$ and inequalities $(\mathbf{A}(i,.),\ \mathbf{b}(i))$ in any viable solution. An upper bound for the quantity on the right-hand-side of Equation 30 can be obtained by setting $\mathbf{b}(i) = 0$, and considering an upper bound for $\sup\{\mathbf{A}(i,.) \cdot \mathbf{x}\}$. The boundedness of the vectors $\mathbf{x}$ and the matrix $\mathbf{A}$ implies that $\sup\{\mathbf{A}(i,.) \cdot \mathbf{x}\} \leq \tau$

Furthermore, the pricing of $\gamma_{\mathbf{x}}[j]$ implies also that

$$M \ \geq\ \sup\{[\mathbf{A}(i,.) \cdot \mathbf{x} - \mathbf{b}(i) - \epsilon]^-\} \tag{31}$$

where $[a]^- \equiv |\min\{0, a\}|$ and the supremum is taken over all pairs of vectors $\mathbf{x}$ and inequalities $(\mathbf{A}(i,.),\ \mathbf{b}(i))$ in any viable solution. An upper bound for the quantity on the right-hand-side of Equation 31 can be obtained by setting $\mathbf{A}(i,.) \cdot \mathbf{x} = 0$, and considering an upper bound for $\sup\{b(i)\}$ which is bounded above by 1. Hence, the pricing of $\gamma_{\mathbf{x}}[j]$ implies that M can be set equal to τ.

Similar analysis can be applied to the pricing of $\delta_{\mathbf{x}}[i]$ to get that M can be set equal to k_1. Equation 28 implies that $M \geq k_2^*$. But since k_2^* is not known a priori and $k_2 \geq k_2^*$, M can be set equal to k_2.

Finally, since $k_1 \leq \tau$, M can be set equal to $max\{\tau, k_2\}$ during the solution of the proposed formulation.

The next theorem provides a formal statement of the validity of the MIP formulation of Equations 15–29 as a classifier design tool for the classification problem considered in this work.

Theorem 1: The application of the formulation of Equations 15–29 to the sets $\overline{P(\bar{S}_{rs})}$ and $\overline{P(\bar{S}^b_{r\bar{s}})}$ corresponding to a given RAS Φ, returns a minimum-sized two-layer classifier for these two sets, and through Equation 14, a minimum-sized two-layer classifier for the original sets S_{rs} and $S^b_{r\bar{s}}$.

Complexity It should be clear from the above discussion that the MIP formulation of Equations 15–29 involves $\mathcal{O}((m_u + m_s) \cdot (k_1 + k_2) + n \cdot k_1 + k_1 \cdot k_2)$ binary variables, $\mathcal{O}(n \cdot k_1 + (m_u + m_s) \cdot k_1 \cdot k_2)$ real variables, $\mathcal{O}(k_1)$ integer variables, and $\mathcal{O}(n \cdot k_1 + (m_u + m_s) \cdot k_1 \cdot k_2)$ technological constraints.[5] Hence, the size of this formulation, in terms of the variables and constraints involved, is polynomially related to the size of the classified sets $\overline{P(\bar{S}_{rs})}$ and $\overline{P(\bar{S}^b_{r\bar{s}})}$ and the dimensionality of their supporting sub-space V_P.

V. Two-stage synthesis of the classifier TLC^Q

In this section, we introduce an algorithm that constructs the sought classifier TLC^Q in two stages. In particular, the first stage involves the generation of the coefficients of the

[5]By technological constraints we mean all the formulation constraints except from those that impose the nonnegative and the binary nature of the various variables. These are the constraints that are explicitly considered in the computations performed by any solution algorithm for the MIP formulation.

inequalities of the first layer of the classifier, whereas the second stage involves the generation of the coefficients of the inequalities of the second layer of the classifier. This approach reduces the complexity of the classifier synthesis and circumvents the complications resulting from the non-linearity incurred by the classifier construction through the MIP formulation of Section IV. On the other hand, the size of the obtained classifier might be larger. Lemma 2 plays a central role in this approach. According to Lemma 2, the separation is attained by TLC^Q if $\mathbf{A}$ and $\mathbf{b}$ are constructed such that $\nexists(\gamma_{\mathbf{s}} \in I(\overline{P(\bar{S}_{rs})}),\ \gamma_{\mathbf{u}} \in I(\overline{P(\bar{S}^b_{r\bar{s}})})) :\ \gamma_{\mathbf{s}} \geq \gamma_{\mathbf{u}}$; i.e., if

$$\forall \mathbf{s} \in \overline{P(\bar{S}_{rs})},\ \forall \mathbf{u} \in \overline{P(\bar{S}^b_{r\bar{s}})},\ \exists \hat{i} \in \{1, \ldots, k_1\} : \\ \gamma_{\mathbf{u}}[\hat{i}] = 1\ \wedge\ \gamma_{\mathbf{s}}[\hat{i}] = 0 \tag{32}$$

Algorithm 1 provides the outline of the two-stage approach. The goal of the first stage is to synthesize a minimal-cardinality set of linear inequalities such that Equation 32 is satisfied for each pair of safe and unsafe states. The next step is to obtain the image of the safe and the unsafe states under the first-layer transformation. By construction, these image vectors are binary vectors that possess the following monotonicity property between safe and unsafe states: $\nexists(\gamma_{\mathbf{s}} \in I(\overline{P(\bar{S}_{rs})}),\ \gamma_{\mathbf{u}} \in I(\overline{P(\bar{S}^b_{r\bar{s}})})) : \gamma_{\mathbf{s}} \geq \gamma_{\mathbf{u}}$. Therefore, we can utilize the MIP formulation given by Equations 39-45 in [19] to separate the image vectors. The following theorem formalizes these remarks.

Theorem 2: The application of Algorithm 1 to the sets $\overline{P(\bar{S}_{rs})}$ and $\overline{P(\bar{S}^b_{r\bar{s}})}$ corresponding to a given RAS Φ, returns a two-layer classifier for these two sets, and through Equation 14, a two-layer classifier for the original sets S_{rs} and $S^b_{r\bar{s}}$.

Proof: The result follows immediately from Lemma 2. □

To alleviate the computational complexity involved with the synthesis of the linear inequalities for each layer, we also consider the possibility of synthesizing these sets of linear inequalities in an incremental manner. For the first stage, we start by having all the pairs of safe and unsafe states in the set W, i.e., $W = \overline{P(\bar{S}_{rs})} \times \overline{P(\bar{S}^b_{r\bar{s}})}$. Next, we synthesize one linear inequality at a time; in particular, at each iteration, a linear inequality is synthesized to satisfy Equation 32 for the maximum number of pairs of safe and unsafe states from the set W, and these pairs are subsequently removed from W. The procedure is repeated until the set W is empty. For the second stage, we start by having the set $I_U = I(\overline{P(\bar{S}^b_{r\bar{s}})})$. Next, we synthesize one linear inequality at at time; in particular, at each iteration, a linear inequality is synthesized to separate the maximum number of states of the set I_U from the states of the set $I(\overline{P(\bar{S}_{rs})})$, and the separated states are removed from I_U. The procedure is repeated until the set I_U is empty.

In the rest of this section, first we give an MIP formulation that synthesizes the first-layer inequalities. Next, we give the additional, more incremental procedure to synthesize the first-layer inequalities described above. For the synthesis of the second layer-linear inequalities, using an MIP formulation and the incremental procedure that was described in the previous paragraph, the reader is referred to [19] – Sections V and VI.

Algorithm 1 The outline of the two-stage construction algorithm of the classifier TLC^Q

Input: (i) $\overline{P(\bar{S}_{rs})}$; (ii) $\overline{P(\bar{S}^b_{r\bar{s}})}$.
Output: $TLC^Q =< \mathbf{A}, \mathbf{b}, \mathbf{C}, \mathbf{d} >$.
1: Synthesize $(\mathbf{A}, \mathbf{b})$ such that Equation 32 is satisfied;
2: Compute the sets $I(\overline{P(\bar{S}_{rs})})$ and $I(\overline{P(\bar{S}^b_{r\bar{s}})})$ using $(\mathbf{A}, \mathbf{b})$;
3: Synthesize $(\mathbf{C}, \mathbf{d})$ to linearly separate $I(\overline{P(\bar{S}_{rs})})$ and $I(\overline{P(\bar{S}^b_{r\bar{s}})})$;
4: Return $(\mathbf{A}, \mathbf{b}, \mathbf{C}, \mathbf{d})$;

A. *Synthesizing the inequalities of the first layer using mathematical programming*

In this part, we provide a MIP formulation for the construction of the first layer of the separator TLC^Q. The main input for this formulation is $\Upsilon \equiv \overline{P(\bar{S}_{rs})} \cup \overline{P(\bar{S}^b_{r\bar{s}})}$. Additionally, ϵ and M are analogous to their counterparts in Equations 15-29. Similarly, the variables $\alpha[i]$, $(\mathbf{A}(i,.), \mathbf{b}[i])$, and $\gamma_{\mathbf{x}}[i]$ are analogous to their counterparts in Equations 15-29. On the other hand, the binary variable $\theta_{\mathbf{su}}[i] = 1$ only if $\gamma_{\mathbf{s}}[i] = 0\ \wedge\ \gamma_{\mathbf{u}}[i] = 1$. The formulation itself is expressed by the following equations:

$$\text{Min} \sum_{i=1}^{k_1} \alpha[i] \tag{33}$$

$$\forall \mathbf{x} \in \{\Upsilon\},\ \forall i \in \{1, \cdots, k_1\} : \\ \epsilon + M \cdot (\gamma_{\mathbf{x}}[i] - 1) \leq \mathbf{A}(i,.) \cdot \mathbf{x} - \mathbf{b}[i] \leq M \cdot \gamma_{\mathbf{x}}[i] \tag{34}$$

$$\forall i \in \{1, \cdots, k_1\},\ \forall \mathbf{s} \in \overline{P(\bar{S}_{rs})},\ \forall \mathbf{u} \in \overline{P(\bar{S}^b_{r\bar{s}})} : \\ 2 \cdot \theta_{\mathbf{su}}[i] \leq \gamma_{\mathbf{u}}[i] - \gamma_{\mathbf{s}}[i] + 1 \tag{35}$$

$$\forall \mathbf{s} \in \overline{P(\bar{S}_{rs})},\ \forall \mathbf{u} \in \overline{P(\bar{S}^b_{r\bar{s}})} :\ \sum_{i=1}^{k_1} \theta_{\mathbf{su}}[i] \geq 1 \tag{36}$$

$$\forall i \in \{1, \ldots, k_1\},\ \forall j \in \{1, \ldots, n\} :\ 0 \leq \mathbf{A}(i,j) \leq \alpha[i] \tag{37}$$

$$\forall i \in \{1, \ldots, k_1\} :\ 0 \leq \mathbf{b}(i) \leq \alpha[i] \tag{38}$$

$$\alpha[i],\ \gamma_{\mathbf{x}}[i],\ \theta_{\mathbf{su}}[i] \in \{0, 1\} \tag{39}$$

The objective of the formulation is defined as the minimization of the number of active linear inequalities employed by the synthesized first layer. The mechanics of Equation 34 and Equations 37– 39 are very similar to the corresponding equations in the formulation given by Equations 15-29. On the other hand, Equation 35 implies that

$\theta_{\mathbf{su}}[i] = 1$ only if $\gamma_{\mathbf{u}}[i] = 1$ and $\gamma_{\mathbf{s}}[i] = 0$. In particular, $\gamma_{\mathbf{u}}[i] - \gamma_{\mathbf{s}}[i] + 1 \in \{0, 1, 2\}$. Thus $\theta_{\mathbf{su}}[i]$ can be set to one only if $\gamma_{\mathbf{u}}[i] - \gamma_{\mathbf{s}}[i] + 1 = 2 \Leftrightarrow \gamma_{\mathbf{u}}[i] - \gamma_{\mathbf{s}}[i] = 1 \Leftrightarrow \gamma_{\mathbf{u}}[i] = 1 \ \wedge \ \gamma_{\mathbf{s}}[i] = 0$. Finally, Equation 36 implements the requirement of Equation 32. For a proper pricing of M, an analysis similar to that presented in Section IV can be applied to get the conclusion that M can be set equal to τ.

Theorem 3: The application of the formulation of Equations 33–39 to the sets $\overline{P(\bar{S}_{rs})}$ and $\overline{P(\bar{S}^b_{r\bar{s}})}$ corresponding to a given RAS Φ, returns a minimum-cardinality set of linear inequalities $(\mathbf{A}, \mathbf{b})$ that satisfies Equation 32.

Complexity The MIP formulation of Equations 33–39 involves $\mathcal{O}(m_u \cdot m_s \cdot k_1)$ binary variables, $\mathcal{O}(n \cdot k_1)$ real variables, and $\mathcal{O}(m_u \cdot m_s \cdot k_1 + k_1 \cdot n)$ technological constraints.

B. Iterative procedures for the first-layer construction

In this subsection, we present an iterative algorithm that synthesizes one linear inequality at each iteration. The synthesized set of linear inequalities of the first layer should satisfy Equation 32. Therefore, we maintain the set W that contains all the pairs of safe and unsafe states that have not yet been separated according to Equation 32 by any of the generated linear inequalities. At each iteration, a linear inequality is generated to separate the maximum number of pairs in W, and the separated pairs are removed from W. The algorithm terminates when W is empty, that is, Equation 32 is satisfied by the generated linear inequalities. Let $S \equiv \{\mathbf{s} \mid (\mathbf{s}, \mathbf{u}) \in W\}$ and $U \equiv \{\mathbf{u} \mid (\mathbf{s}, \mathbf{u}) \in W\}$ be the sets of safe and unsafe states in W. After each iteration, the sets S and U might shrink down, an effect that might result in having a dimension j such that $\forall \mathbf{s} \in S, \ \forall \mathbf{u} \in U : \ \mathbf{s}[j] \geq \mathbf{u}[j]$. Thus, this dimension can be dropped as explained in Section III. To this end, we define J_0 as the set of dimensions removed from the state space by the projections applied in the course of this iterative procedure. Let $J_P \equiv L_P \setminus J_0$, and V_{JP} denote the $|J_P|$-dimensional subspace supporting the further projection. Finally, let $\Gamma : \mathbb{N} \rightarrow \mathbb{N}$ be a bijection that maps the elements of the dimension set J_P to the dimensions of subspace V_{JP}. The linear inequalities in the subspace V_P are constructed from the linear inequalities synthesized in the projected subspace V_{JP} as follows:

$$\forall j \in J_0 : \ \mathbf{a}[j] = 0 \ \ \wedge \ \ \forall j \in J_P : \mathbf{a}[j] = \mathbf{a}'[\Gamma(j)] \ \ \wedge \ \ \mathbf{b} = \mathbf{b}' \tag{40}$$

where $< \mathbf{A}', \mathbf{b}' >$ are the linear inequalities synthesized in the projected subspace V_{JP}. The complete iterative procedure is depicted in Algorithm 2.

Next, we present the MIP formulation utilized by Algorithm 2 to synthesize a linear inequality that separates the maximum number of pairs of safe and unsafe states from W according to Equation 32. The main inputs to the formulation are: (i) the set W of pairs of safe and unsafe states that have not been yet separated according to Equation 32, and (ii) the set Υ of safe and unsafe states that appear in W. Also, in the subsequent discussion, we set $w \equiv |W|$, $m_s \equiv |S|$, $m_u \equiv |U|$, and $n \equiv |J_P|$. Additionally,

Algorithm 2 The iterative construction of the first layer of a two-layer classifier

Input: (i) $\overline{P(\bar{S}_{rs})}$; (ii) $\overline{P(\bar{S}^b_{r\bar{s}})}$; (iii) ϵ and M.
Output: $(\mathbf{A}, \mathbf{b})$.

```
W = P(S̄_rs) × P(S̄^b_rs̄);   i = 0;
J_P = L_P;  J_0 = ∅;
while W ≠ ∅ do
  i = i + 1;
  for all { j | ∀(s, u) ∈ W : s[j] ≥ u[j] } do
    Remove j from J_P;  Add j to J_0;
  end for
  W ← Project the elements of W to subspace V_JP;
  S = {s | (s, u) ∈ W}; U = {u | (s, u) ∈ W};
  Υ = S ∪ U;
  Generate the linear inequality (A'(i, .), b'[i]) by solving
    the MIP formulation of Equations 41–45 with inputs
    W, Υ, ϵ, and M.
  Construct (A(i, .), b[i]) from (A'(i, .), b'[i]) through
    Equation 40.
  Remove the set {(s, u) ∈ W | θ_su = 1} from W.
end while
Return (A, b);
```

ϵ and M are similar to their counterparts in Equations 33-39. Finally, the variables $\mathbf{a}$, b, $\gamma_{\mathbf{x}}$, and $\theta_{\mathbf{su}}$ are analogous to their counterparts in Equations 33-39 with $k_1 = 1$. The formulation itself takes the following form:

$$\text{Max} \sum_{(\mathbf{s},\mathbf{u}) \in W} \theta_{\mathbf{su}} \tag{41}$$

$$\forall \mathbf{x} \in \{\Upsilon\} : \ \epsilon + M \cdot (\gamma_{\mathbf{x}} - 1) \leq \mathbf{a} \cdot \mathbf{x} - b \leq M \cdot \gamma_{\mathbf{x}} \tag{42}$$

$$\forall (\mathbf{s}, \mathbf{u}) \in W : \ 2 \cdot \theta_{\mathbf{su}} \leq \gamma_{\mathbf{u}} - \gamma_{\mathbf{s}} + 1 \tag{43}$$

$$\forall i \in \{1, \ldots, n\} : \ 0 \leq \mathbf{a}[i] \leq 1 \ \ \wedge \ \ 0 \leq b \leq 1 \tag{44}$$

$$\gamma_{\mathbf{x}}, \ \theta_{\mathbf{su}} \in \{0, 1\} \tag{45}$$

The objective of the formulation is to maximize the number of pairs of safe and unsafe states from the set W that are separated according to Equation 32. The mechanics of Equations 42–45 resemble their corresponding counterparts in the formulation given by Equations 33–39. Analyzing Equation 42, we can see that M can be set equal to τ.

Complexity The MIP formulation of Equations 41–45 involves $\mathcal{O}(m_u + m_s + w)$ binary variables, $\mathcal{O}(n)$ real variables, and $\mathcal{O}(m_s + m_u + w)$ technological constraints.

Synthesizing an "upper-bound" linear inequality By an "upper-bound" linear inequality, we mean a linear inequality whose coefficients except for exactly one, are set to zero. Thus, it can be described as $\mathbf{a}[i^*] \cdot \mathbf{x}[i^*] \leq b$. The proof of Proposition 4 demonstrates the existence

TABLE II

THE RAS CONSIDERED IN THE EXAMPLE OF SECTION VI

Resource Types: $\{R_1, \ldots, R_6\}$ Resource Capacities: $C_i = 3, \forall i \in \{1, \ldots, 6\}$
Process Type 1: $2R_4 \rightarrow R_2 \rightarrow R_1$ Process Type 2: $R_3 \rightarrow 3R_6$ Process Type 3: $2R_1 \rightarrow R_5 \rightarrow 3R_1 \rightarrow R_6 \rightarrow 3R_2$ Process Type 4: $2R_3 \rightarrow R_1 \rightarrow 3R_5$

of a two-layer classifier whose first-layer inequalities are upper-bound inequalities. A formulation that can provide these inequalities can be obtained through a straightforward modification of the formulation of Eqs 41–45. It should be clear that synthesizing an upper-bound linear inequality is computationally easier as it involves the selection of $\mathbf{a}[i^*]$ and b only rather than the selection of all the coefficients of the linear inequality $(\mathbf{a}, b)$. This effect will be demonstrated in the next section.

VI. COMPUTATIONAL RESULTS

In this section we report a set of experiments that demonstrates and assesses the applicability of the proposed methodologies. First, we apply the introduced methods to the RAS configuration defined in Table II where the considered RAS has six resource types, $R_1, \ldots, R_6$, each with capacity $C_i = 3$. It also has four process types, $\Pi_1, \ldots, \Pi_4$, where each process type consists of a set of consecutive processing stages with each stage engaging a single resource type at the amount indicated by the corresponding coefficient; for instance, the first processing stage of the first process type engages two units of resource R_4, the second stage of the same type engages one unit of resource R_2, etc. The depicted RAS has 13 processing stages and this number defines the dimensionality of the state vector. The application of the proposed DAP design methodology revealed that the considered RAS has a reachable state space of 19,980 states, with 13,092 of them being safe states and the remaining 6,888 being unsafe states. Applying the introduced thinning techniques led to a set $\overline{P(\bar{S}_{rs})}$ with 6 states, and a set $\overline{P(\bar{S}^b_{r\bar{s}})}$ with 3 states, whereas the dimensionality of the projected subspace V_P is 6. These sets are reported in Table III. Furthermore, Table III reports the two-layer classifiers obtained by applying (i) Equations 15–29, (ii) Algorithm 1 in conjunction with Equations 33–39, (iii) Algorithm 1 in conjunction with Algorithm 2, and (iv) Algorithm 1 in conjunction with Algorithm 2 and the restriction of the first-layer inequalities to the upper-bound type. The two-layer classifiers obtained by the first three methods have the same size which equals $(2{\cdot}|L_P|{+}1){\cdot}k_1{+}(2{\cdot}k_1{+}1){\cdot}k_2 = (2{\cdot}6{+}1){\cdot}3{+}(2{\cdot}3{+}1){\cdot}1 = 46$. On the other hand, the size of the classifier obtained by the last method equals $(2 \cdot |L_P| + 1) \cdot k_1 + (2 \cdot k_1 + 1) \cdot k_2 = (2 \cdot 6 + 1) \cdot 7 + (2 \cdot 7 + 1) \cdot 2 = 121$.

Table IV reports the results obtained from the application of the proposed methods for the deployment of the maximally permissive DAP on additional 39 RAS configurations. These configurations provide a representative sample of the results obtained in our experiments. More specifically, for each of these configurations, Table IV reports: (i) the total number of processing stages ξ, (ii) the cardinality of the reachable safe subspace S_{rs}, (iii) the cardinality of the reachable unsafe subspace $S_{r\bar{s}}$, (iv) the dimensionality of the projected subspace V_P, (v) the cardinality of the set of maximal projected safe states $\overline{P(\bar{S}_{rs})}$, (vi) the cardinality of the set of minimal projected boundary unsafe states $\overline{P(\bar{S}^b_{r\bar{s}})}$, (vii) the size of the two-layer classifier obtained by applying the formulation of Equations 15–29, (viii) the size of the two-layer classifier obtained by applying Algorithm 1 in conjunction with Equations 33–39, (ix) the size of the two-layer classifier obtained by applying Algorithm 1 in conjunction with Algorithm 2, and (x) the size of the two-layer classifier obtained by applying Algorithm 1 in conjunction with Algorithm 2 while restricting the first layer to upper-bound linear inequalities. (xi) Finally, the last five columns (11-15) report, in secs, the computational times that were required for the generation of the corresponding classifiers in columns 7-10. In particular, column 11 reports the time that is required to perform the preprocessing steps for the generation of the sets $\overline{P(\bar{S}_{rs})}$ and $\overline{P(\bar{S}^b_{r\bar{s}})}$, and it is common for all four approaches. The remaining four columns (12-15) report the compuational times that are required in order to obtain the target clas-

TABLE III

THE RESULTS OF THE APPLICATION OF THE PROPOSED METHODS ON THE RAS CONFIGURATION DEPICTED IN TABLE II

$\overline{P(\bar{S}_{rs})} = \{[3\ 3\ 0\ 0\ 0\ 3]^T,\ [1\ 3\ 0\ 0\ 1\ 3]^T,\ [0\ 3\ 1\ 2\ 0\ 0]^T,$ $[0\ 3\ 0\ 3\ 0\ 3]^T,\ [0\ 3\ 0\ 2\ 1\ 3]^T,\ [0\ 2\ 1\ 2\ 0\ 3]^T\}$ $\overline{P(\bar{S}^b_{r\bar{s}})} = \{[0\ 3\ 1\ 0\ 0\ 1]^T,\ [0\ 0\ 0\ 3\ 1\ 0]^T,\ [1\ 0\ 0\ 1\ 0\ 0]^T\}$
The two-layer classifier obtained by the formulation of Equations 15–29
First layer inequalities : $\{\ 0.03 \cdot x_2 + 0.91 \cdot x_3 + 0.01 \cdot x_6 \leq 1;$ $x_1 + 0.01 \cdot x_4 + 0.01 \cdot x_5 \leq 0.03;$ $x_4 \leq 0\}$ Second layer inequalities: $\{\ \gamma_1 + 0.01 \cdot \gamma_2 + 0.01 \cdot \gamma_3 \leq 0.01\ \}$
The two-layer classifier obtained by Algorithm 1 in conjunction with Equations 33–39
First layer inequalities : $\{\ 0.03 \cdot x_2 + 0.03 \cdot x_3 + 0.01 \cdot x_6 \leq 0.12;$ $x_3 + x_4 \leq 0;$ $0.99 \cdot x_1 + 0.33 \cdot x_4 + 0.01 \cdot x_5 \leq 1\}$ Second layer inequalities: $\{\ \gamma_1 + \gamma_2 + \gamma_3 \leq 1\ \}$
The two-layer classifier obtained by Algorithm 1 in conjunction with Algorithm 2
First layer inequalities : $\{\ 0.99 \cdot x_1 + 0.33 \cdot x_4 + 0.01 \cdot x_5 \leq 1;$ $0.25 \cdot x_2 + 0.25 \cdot x_3 + 0.0833 \cdot x_6 \leq 1;$ $x_1 \leq 0\}$ Second layer inequalities: $\{\ 0.01 \cdot \gamma_1 + \gamma_2 + 0.01 \cdot \gamma_3 \leq 0.01\ \}$
The two-layer classifier obtained by Algorithm 1 in conjunction with Algorithm 2 and the restriction of the first-layer inequalities to the upper-bound type
First layer inequalities : $\{\ x_1 \leq 0.99;\ x_2 \leq 2.99;\ x_3 \leq 0.99;$ $x_4 \leq 2.99;\ x_4 \leq 0.99;\ x_5 \leq 0.99;$ $x_6 \leq 0.99\}$ Second layer inequalities: $\{\ 0.02 \cdot \gamma_1 + 0.01 \cdot \gamma_3 + 0.01 \cdot \gamma_4 +$ $0.02 \cdot \gamma_5 + 0.01 \cdot \gamma_6 \leq 0.03;$ $0.01 \cdot \gamma_2 + 0.01 \cdot \gamma_3 +$ $0.01 \cdot \gamma_6 \leq 0.02\ \}$

TABLE IV
EXPERIMENTAL RESULTS

$\|\xi\|$	$\|S_{rs}\|$	$\|S_{r\bar{s}}\|$	$\|L_P\|$	$\|\overline{P(\bar{S}_{rs})}\|$	$\|\overline{P(\bar{S}^b_{r\bar{s}})}\|$	$\|TLC_1\|$	$\|TLC_2\|$	$\|TLC_3\|$	$\|TLC_4\|$	t_g	t_1	t_2	t_3	t_4
15	5984	2560	8	22	15	86 [F]	77	107	302	0	7200	7200	0	0
15	3360	1728	7	11	8	52	52	69	192	0	901	11	0	0
12	3274	656	7	12	7	52 [F]	35	35	154	0	7200	10	0	0
9	3376	368	5	13	7	40	66	40	152	0	434	82	0	0
12	1270	682	8	4	15	39	39	58	134	0	8	1	0	0
12	892	461	7	7	6	52	59	52	213	0	843	22	0	0
12	1149	66	6	12	6	31	31	31	121	0	9	2	0	0
12	887	223	8	13	11	58	58	96	233	0	327	38	0	0
12	636	408	6	9	7	46	46	46	138	0	500	4	0	0
9	879	75	6	17	7	61 [F]	46	61	212	0	7200	45	0	0
9	753	159	6	8	6	46	46	61	138	0	83	2	0	0
10	728	56	5	23	12	27	27	27	131	0	33	29	0	0
26	196021	11451	8	46	15	-	170	86	275	4	-	7200	183	0
15	2827	1178	9	33	11	-	127	106	328	0	-	7200	1	0
12	354	259	9	20	10	-	127	147	274	0	-	7200	4	0
15	2030	1870	10	36	14	-	177	152	410	0	-	7200	98	0
64	34695	1773193	55	1612	427	-	-	11012	9180	54	-	-	2439	870
39	757699	700781	21	1927	89	-	-	2144	1603	54	-	-	2131	57
56	21099	906478	48	703	225	-	-	6164	6679	40	-	-	2698	375
64	16170	445499	51	1272	320	-	-	6012	7636	12	-	-	2471	388
56	8464	187610	45	622	250	-	-	3797	6113	7	-	-	776	275
28	99548	82989	19	747	75	-	-	1839	1965	4	-	-	316	63
78	3799	166858	70	549	407	-	-	5044	12254	7	-	-	1401	267
24	94431	72238	17	657	71	-	-	1228	1664	3	-	-	970	71
60	9448	152984	49	530	225	-	-	3597	6793	5	-	-	1097	305
24	104550	49620	15	1158	67	-	-	447	1089	2	-	-	1452	193
24	115766	28510	11	84	12	-	-	176	380	2	-	-	328	26
52	1622861	2600349	37	14257	257	-	-	-	4764	261	-	-	-	2143
52	853187	1993667	39	16036	576	-	-	-	7183	141	-	-	-	3199
63	65992	2105590	56	2203	708	-	-	-	12576	79	-	-	-	3594
65	344779	1230069	51	22876	905	-	-	-	12397	88	-	-	-	700
88	53080	1410311	74	5460	1046	-	-	-	17139	87	-	-	-	2467
64	118470	1253425	54	3203	612	-	-	-	10873	42	-	-	-	2024
56	55832	800208	48	3402	687	-	-	-	9046	31	-	-	-	2606
78	31856	740548	65	3077	933	-	-	-	14197	31	-	-	-	3542
48	80343	691959	39	2657	241	-	-	-	5471	26	-	-	-	634
54	36934	443053	48	1463	474	-	-	-	7885	15	-	-	-	848
91	6546	444303	82	998	684	-	-	-	17215	22	-	-	-	615
56	11803	413231	47	1874	755	-	-	-	8015	13	-	-	-	917

sifier from the sets $\overline{P(\bar{S}_{rs})}$ and $\overline{P(\bar{S}^b_{r\bar{s}})}$.

A "-" entry in the presented data indicates that the relevant methodology failed to find a solution within an 120 min time-budget or that it led to memory overflow.[6] On the other hand, the qualification [F] in column $|TLC_1|$ indicates that the obtained solution is just a feasible solution; this would happen if the solution algorithm was terminated prematurely, upon the exhaustion of the 120 min budget.

The following observations can be drawn from the reported results:

- Similar to the case of [19], the various "set-thinning" steps introduced in Section III play a very significant role in reducing the data to be considered explicitly in the synthesis of the sought classifiers, and therefore, in establishing the feasibility of the proposed methods.
- Restricting Algorithm 2 to consider only upper-bound linear inequalities for the first-layer, enables the solution of larger problem instances, and it tends to require significantly less time for the generation of the target classifier than the other approaches. On the other hand, this restriction typically increases the size of the synthesized classifier.
- Algorithm 2 is capable of solving larger problem instances compared to the approach that is based on Equations 33–39.
- For small problem instances, Equations 15–29 can be solved to optimality; hence, the minimum-size classifier is synthesized.
- Comparing the approach based on Equations 15–29 to the approach based on Algorithm 1 in conjunction with 33–39, we can see that the tractability of the second is slightly better than that of the first. Both methods are not scalable. On the other hand, the size of the synthesized classifiers are of comparable order.

Closing the discussion on our computational experiments, we want also to notice that none of the RAS configurations employed in the presented experiments accepts

[6]Relaxing the time constraint did not help in finding a solution. For the sake of completeness, we also report that these experiments were performed on a 2.66 GHz quad-core Intel Xeon 5430 processor with 6 MB of cache memory and 32 GB RAM; however, each job ran on a single core.

a characterization of its maximally permissive DAP as a set of linear inequalities of the form presented in [19], and therefore, they are not amenable to the Petri net-based methodologies discussed in the introductory section. This is an attribute that differentiates the presented approach in a strong qualitative sense from the past approaches that have appeared in the relevant literature.

VII. Conclusion

This work has extended the results of [19], that sought the reformulation of the synthesis of the maximally permissive DAP for complex RAS as the design of a compact classifier effecting the dichotomy of the underlying reachable state space into its safe and unsafe subspaces, to the case where the sought dichotomy cannot be represented by a linear classifier. We proposed new classification schemes for this more complex case and established formally their completeness, i.e., their ability to provide an effective classifier for every instance of the considered RAS class. We also provided effective and computationally efficient procedures for the synthesis of the sought classifiers. Finally, the effectiveness and the efficacy of the presented approaches were demonstrated and assessed through a series of computational experiments. To the best of our knowledge, this is the first set of results to provide a complete analytical characterization of the maximally permissive DAP for the considered RAS class. Our computational results also establish that the presented methods can be applicable to RAS of structure and size comparable to those encountered in modern technological applications.

Concluding the presentation of our results, we also notice that an alternative way to express the dichotomy of the image sets $I(S_{rs})$ and $I(S_{r\bar{s}})$, that are derived by the proposed transformation $\gamma(\cdot, \mathbf{A}, \mathbf{b})$, is through a Boolean function $F(\gamma)$ that is defined on the components of γ. A preliminary study of this alternative representational scheme of the maximally permissive DAP and its potential to provide effective compact classifiers for the considered RAS class is reported in [17]. Our future work will seek a more systematic study of the comparative advantages between the DAP representation presented in this work and the alternative representation of [17]. In addition, we shall seek the implementation of the presented results in particular application domains, and their extension to other supervisory control problems of more general forbidden state nature, which might also involve additional effects like uncontrollable or unobservable transitions.

Appendix

A. A useful lemma

The following lemma is used below in the proof of Proposition 5.

Lemma 3: Consider the vectors $\mathbf{s}$, $\mathbf{s}'$, $\mathbf{u}$, and $\mathbf{u}'$, and the two-layer classifiers $TLC =< \mathbf{A}, \mathbf{b}, \mathbf{C}, \mathbf{d} >$ and $TLC' =< \mathbf{A}', \mathbf{b}', \mathbf{C}, \mathbf{d} >$, both with non-negative coefficients, and where each of the constructs $\mathbf{A}$, $\mathbf{A}'$, $\mathbf{b}$, and $\mathbf{b}'$ has k_1 rows. Also, assume that $\mathbf{s}$ and $\mathbf{u}$ are respectively classified by TLC as a safe and an unsafe state. Then:
1. If $\forall i \in \{1, \ldots, k_1\}: \ \mathbf{A}'(i,.) \cdot \mathbf{s}' - \mathbf{b}'[i] \leq \mathbf{A}(i,.) \cdot \mathbf{s} - \mathbf{b}[i]$, then $\mathbf{s}'$ is classified as a safe state by TLC'.
2. If $\forall i \in \{1, \ldots, k_1\}: \ \mathbf{A}'(i,.) \cdot \mathbf{u}' - \mathbf{b}'[i] \geq \mathbf{A}(i,.) \cdot \mathbf{u} - \mathbf{b}[i]$, then $\mathbf{u}'$ is classified as an unsafe state by TLC'.

Proof: First, assume that $\forall i \in \{1, \ldots, k_1\}: \ \mathbf{A}'(i,.) \cdot \mathbf{s}' - \mathbf{b}'[i] \leq \mathbf{A}(i,.) \cdot \mathbf{s} - \mathbf{b}[i]$. Thus, $\gamma_{\mathbf{s}}[i] = 0 \Rightarrow \gamma'_{\mathbf{s}'}[i] = 0$. Therefore, $\gamma'_{\mathbf{s}'} \leq \gamma_{\mathbf{s}}$. Since $\mathbf{s}$ is classified as a safe state by TLC, $\forall i \in \{1, \ldots, k_2\}: \ \mathbf{C}(i,.) \cdot \gamma_{\mathbf{s}} - \mathbf{d}[i] \leq 0$. Then the non-negativity of $\mathbf{C}$ implies that $\forall i \in \{1, \ldots, k_2\}: \ \mathbf{C}(i,.) \cdot \gamma'_{\mathbf{s}'} - \mathbf{d}[i] \leq \mathbf{C}(i,.) \cdot \gamma_{\mathbf{s}} - \mathbf{d}[i] \leq 0$. Therefore, $\mathbf{s}'$ is classified as a safe state by TLC'.

Next, assume that $\forall i \in \{1, \ldots, k_1\}: \ \mathbf{A}'(i,.) \cdot \mathbf{u}' - \mathbf{b}'[i] \geq \mathbf{A}(i,.) \cdot \mathbf{u} - \mathbf{b}[i]$. Thus, $\gamma_{\mathbf{u}}[i] = 1 \Rightarrow \gamma'_{\mathbf{u}'}[i] = 1$. Therefore, $\gamma'_{\mathbf{u}'} \geq \gamma_{\mathbf{u}}$. Since $\mathbf{u}$ is classified as an unsafe state by TLC, $\exists \hat{i} \in \{1, \ldots, k_2\}: \ \mathbf{C}(\hat{i},.) \cdot \gamma_{\mathbf{u}} - \mathbf{d}[\hat{i}] > 0$. Then the non-negativity of $\mathbf{C}$ implies that $\mathbf{C}(\hat{i},.) \cdot \gamma'_{\mathbf{u}'} - \mathbf{d}[\hat{i}] \geq \mathbf{C}(\hat{i},.) \cdot \gamma_{\mathbf{u}} - \mathbf{d}[\hat{i}] > 0$. Therefore, $\mathbf{u}'$ is classified as an unsafe state by TLC'.

B. Proof of Proposition 5

Let $\mathbf{s} \in S_{rs}$ be an arbitrary non-maximal safe state vector, and $\mathbf{s}^* \in \bar{S}_{rs}$ be a maximal safe state vector such that $\mathbf{s}^* > \mathbf{s}$.

Also, let $\mathbf{u} \in S^b_{r\bar{s}}$ be an arbitrary non-minimal boundary unsafe state vector, and $\mathbf{u}^* \in \bar{S}^b_{r\bar{s}}$ be a minimal boundary unsafe state vector such that $\mathbf{u}^* < \mathbf{u}$.

Assume that we have already identified a two-layer classifier $TLC =< \mathbf{A}, \mathbf{b}, \mathbf{C}, \mathbf{d} >$ s.t. $\mathbf{A} \geq 0$, $\mathbf{b} \geq 0$, $\mathbf{C} \geq 0$ and $\mathbf{d} \geq 0$, that separates $\bar{S}_{rs}$ and $\bar{S}^b_{r\bar{s}}$. Then, the non-negativity of all the coefficients of the linear inequalities, combined with the presumed relations of $\mathbf{s}$ to $\mathbf{s}^*$ and of $\mathbf{u}$ to $\mathbf{u}^*$, further imply that:
- $\forall i \in \{1, \ldots, k_1\}: \ \mathbf{A}(i,.) \cdot \mathbf{s} - \mathbf{b}[i] \leq \mathbf{A}(i,.) \cdot \mathbf{s}^* - \mathbf{b}[i]$. Hence, by Lemma 3, $\mathbf{s}$ is classified as a safe state.
- $\forall i \in \{1, \ldots, k_1\}: \ \mathbf{A}(i,.) \cdot \mathbf{u} - \mathbf{b}[i] \geq \mathbf{A}(i,.) \cdot \mathbf{u}^* - \mathbf{b}[i]$. Hence, by Lemma 3, $\mathbf{u}$ is classified as unsafe state.

Since states $\mathbf{s}$ and $\mathbf{u}$ were arbitrarily chosen, we can infer that the two-layer classifier $TLC =< \mathbf{A}, \mathbf{b}, \mathbf{C}, \mathbf{d} >$ is also an effective separator for the entire sets S_{rs} and $S^b_{r\bar{s}}$.

C. Proof of Proposition 6

To see the validity of Proposition 6, first notice that, under the stated assumptions,

$$\begin{aligned} \forall \mathbf{x} \in \bar{S}_{rs} \cup \bar{S}^b_{r\bar{s}}, \ \forall i: \quad \mathbf{A}^H(i,.) \cdot \mathbf{x} &= \\ \textstyle\sum_{j \in L} \mathbf{A}^H(i,j) \cdot \mathbf{x}[j] &= \\ \textstyle\sum_{j \in L_P} \mathbf{A}^H(i,j) \cdot \mathbf{x}[j] &= \\ \mathbf{A}^Q(i,.) \cdot \mathbf{x}^p & \end{aligned}$$

where $\mathbf{x}^p$ is the image of $\mathbf{x}$ in subspace V_P. Then, the result follows from the fact that $\mathbf{b}^H = \mathbf{b}^Q \ \wedge \ \mathbf{C}^H = \mathbf{C}^Q \ \wedge \ \mathbf{d}^H = \mathbf{d}^Q$ (c.f. Eq. 14).

D. Proof of Proposition 7

First we notice that that the projected sets $P(\bar{S}^b_{r\bar{s}})$ and $P(\bar{S}_{rs})$ retain the monotonicity property of Proposition 1; in particular, there are no vectors $\mathbf{x}_1 \in P(\bar{S}_{rs})$ and

$\mathbf{x}_2 \in P(\bar{S}^b_{r\bar{s}})$ such that $\mathbf{x}_2 \leq \mathbf{x}_1$. Indeed, the existence of such a pair of vectors, $\mathbf{x}_1, \mathbf{x}_2$, when combined with the condition that defines projection P, would further imply the existence of vectors $\mathbf{x'}_1 \in \bar{S}_{rs}$ and $\mathbf{x'}_2 \in \bar{S}^b_{r\bar{s}}$ such that $\mathbf{x'}_2 \leq \mathbf{x'}_1$, and Proposition 1 would be violated.

Having established that the projected sets $P(\bar{S}^b_{r\bar{s}})$ and $P(\bar{S}_{rs})$ retain the monotonicity property of Proposition 1, the existence of a two-layer classifier with non-negative coefficients that separates these two sets can be established with an argument similar to that of Lemma 2 and Proposition 4.

References

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

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

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

[4] C. G. Cassandras and S. Lafortune. *Introduction to Discrete Event Systems (2nd ed.)*. Springer, NY,NY, 2008.

[5] E. G. Coffman, M. J. Elphick, and A. Shoshani. System deadlocks. *Computing Surveys*, 3:67–78, 1971.

[6] E. W. Dijkstra. Cooperating sequential processes. Technical report, Technological University, Eindhoven, Netherlands, 1965.

[7] J. Ezpeleta, J. M. Colom, and J. Martinez. A Petri net based deadlock prevention policy for flexible manufacturing systems. *IEEE Trans. on R&A*, 11:173–184, 1995.

[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*, 19:137–141, 2003.

[9] E. M. Gold. Deadlock prediction: Easy and difficult cases. *SIAM Journal of Computing*, 7:320–336, 1978.

[10] A. N. Habermann. Prevention of system deadlocks. *Comm. ACM*, 12:373–377, 1969.

[11] J. W. Havender. Avoiding deadlock in multi-tasking systems. *IBM Systems Journal*, 2:74–84, 1968.

[12] R. D. Holt. Some deadlock properties of computer systems. *ACM Computing Surveys*, 4:179–196, 1972.

[13] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley, Reading, MA, 1979.

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

[15] Z. Li, M. Zhou, and N. Wu. A survey and comparison of Petri net-based deadlock prevention policies for flexible manufacturing systems. *IEEE Trans. Systems, Man and Cybernetics – Part C: Applications and Reviews*, 38:173–188, 2008.

[16] T. Murata. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77:541–580, 1989.

[17] A. Nazeem and S. Reveliotis. Designing maximally permissive deadlock avoidance policies for sequential resource allocation systems through classification theory. In *Proceedings of the 7th IEEE Conf. on Automation Science and Engineering*, pages –. IEEE, 2011.

[18] A. Nazeem and S. Reveliotis. A practical approach for maximally permissive liveness-enforcing supervision of complex resource allocation systems. *IEEE Transactions on Automation Science and Engineering*, 8:766–779, 2011.

[19] A. Nazeem, S. Reveliotis, Y. Wang, and S. Lafortune. Designing compact and maximally permissive deadlock avoidance policies for complex resource allocation systems through classification theory: the linear case. *IEEE Transactions on Automatic Control*, 56:1818–1833, 2011.

[20] G.L. Nemhauser and L.A. Wolsey. *Integer and combinatorial optimization.* Wiley, New York, 1988.

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

[22] 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*, 46:1572–1583, 2001.

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

[24] S. Reveliotis, E. Roszkowska, and J. Y. Choi. Generalized algebraic deadlock avoidance policies for sequential resource allocation systems. *IEEE Trans. on Automatic Control*, 52:2345–2350, 2007.

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

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

[27] E. Roszkowska and S. Reveliotis. On the liveness of guidepath-based, zoned-controlled, dynamically routed, closed traffic systems. *IEEE Trans. on Automatic Control*, 53:1689–1695, 2008.

[28] N. Viswanadham, Y. Narahari, and T. L. Johnson. Deadlock avoidance in flexible manufacturing systems using petri net models. *IEEE Trans. on Robotics and Automation*, 6:713–722, 1990.

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