Optimal deadlock avoidance for complex resource allocation systems through classification theory ⋆

Ahmed Nazeem * **Spyros A. Reveliotis** * **Yin Wang** **
Stéphane Lafortune ***

* *School of Industrial & Systems Engineering, Georgia Institute of Technology (e-mail: {anazeem@,spyros@isye.}gatech.edu).*
** *Hewlett-Packard Labs (e-mail: yin.wang@hp.com).*
*** *Department of Electrical Engineering and Computer Science University of Michigan (e-mail: stephane@eecs.umich.edu).*

Abstract: Most of the past research on the problem of deadlock avoidance for sequential complex resource allocation systems (RAS) has acknowledged the fact that the maximally permissive deadlock avoidance policy (DAP) possesses super-polynomial complexity for most RAS classes, and it has resorted to solutions that trade off maximal permissiveness for computational tractability. In this work, we seek the effective implementation of the maximally permissive DAP for a broad spectrum of RAS, by distinguishing between the *off-line* and the *on-line* computation that is required for the specification of this policy, and developing a representation of the derived result that will require *minimal* on-line computation. The particular representation that we adopt is that of a compact *classifier* that will effect the underlying dichotomy of the reachable state space into safe and unsafe subspaces. Through a series of reductions of the posed classification problem, we are also able to attain extensive reductions in the computational complexity of the off-line task of the construction of the sought classifier. A series of computational experiments demonstrate the efficacy of the proposed approach and establish its ability to provide tractable implementations of the maximally permissive DAP for problem instances significantly beyond the capacity of any other approach currently available in the literature.

Keywords: Deadlock, Supervisory Control, Discrete Event Systems, Classifiers, Optimality

1. INTRODUCTION

Deadlock avoidance for sequential resource allocation systems (RAS) is an ubiquitous problem that arises in most technological applications involving the concurrent execution of a set of processes that compete for the staged acquisition and release of some underlying set of system resources. In particular, the applied control logic must avoid the development of circular waiting patterns where a subset of these processes are waiting upon each other for the release of the resources that are needed for their further advancement, a situation characterized as "deadlock" in the relevant terminology. From a methodological standpoint, the problem can be characterized in the classical Ramadge & Wonham Supervisory Control (R&W SC) framework (Ramadge and Wonham (1989); Cassandras and Lafortune (2008)) in a straightforward manner, by (i) expressing the underlying resource allocation dynamics through a Finite State Automaton (FSA) and (ii) requesting the confinement of the RAS behavior to the subspace of this FSA that is defined by its maximal strongly connected component that contains the system state $\mathbf{s}_0$ where the RAS is idle and empty of any jobs. [1] In fact, such a characterization of the problem and its solution establishes also a notion of optimality for the considered problem, since the resulting policy prevents effectively the formation of deadlock while retaining the maximum possible behavioral latitude for the underlying RAS.

However, the direct implementation of the solution approach outlined in the previous paragraph, is seriously impeded by the fact that it necessitates the "real-time" – or the "on-line" – assessment of the co-accessibility of any given RAS state to the empty state $\mathbf{s}_0$, a property that is otherwise known as the state *"safety"*, in the relevant terminology; for most RAS classes, the assessment of state safety is an NP-complete problem (see Araki et al. (1977); Gold (1978); Lawley and Reveliotis (2001)). Hence, the research community has tried to circumvent the limitations imposed by this negative result by either (a) compromising for sub-optimal – i.e., non-maximally permissive – solutions that are based on polynomially assessed properties of the relevant RAS states, or (b) adopting alternative, more compact representations of the considered RAS dynamics and hoping that the compactness of these alternative representations, combined with further structural properties and insights revealed by them, will also lead, at least in most practical cases, to fairly compact characterizations of the target policy and to more efficient approaches for its derivation. A modeling framework that seems to hold particular promise along this second line of research, and therefore, has been explored more persistently in the past, is that of Petri nets (PN) (Murata (1989)). In particular, the attribution of the non-liveness of the RAS-modeling PNs to the formation of some structural objects known as "empty – or, more generally, deadly marked – siphons", theoretically enables the determination of the maximally permissive DAP of any given RAS through an incremental computation that identifies the aforementioned problematic siphons in the net behavior and controls them through the net augmentation with an additional set of places, known as "monitor" places. However, this PN-based structural approach to the computation of the maximally permissive DAP is not fully developed yet, and its practical applicability is not fully explored and understood. An alternative prominent

⋆ The first two authors were partially supported by NSF grants CMMI-0619978 and CMMI-0928231. The research of the fourth author was supported in part by NSF grants CCF-0819882 and CNS-0930081, and by an award from HP Labs' Innovation Research Program.

[1] All technical concepts are defined more systematically in subsequent parts of this manuscript.

approach that has been pursued within the context of the PN modeling framework is that of the "theory of regions" (Badouel and Darondeau (1998)) and its derivatives. The key idea behind the theory of regions, as implemented in the considered problem context, is to first compute the maximally permissive deadlock avoidance policy (DAP) using the standard R&W SC representations and methods mentioned at the beginning of this document, and subsequently encode this policy to a PN model; this last PN is once again obtained as an augmentation of the PN modeling the original RAS with appropriate "control" or "monitor" places. The reader can find an extensive coverage of all these past developments in Zhou and Fanti (2004); Reveliotis (2005, 2007); Li et al. (2008), and the references cited therein.

Motivated by the above remarks, in this work we pursue an alternative approach to the design of the maximally permissive DAP for any given RAS. Our work presents some conceptual similarity to the approach of the theory of regions outlined in the previous paragraph, in that we also organize the overall computation of the optimal DAP into two stages, with the first stage obtaining this policy in the R&W SC framework, and the second stage trying to express the obtained result in a more compact form. However, instead of explicitly relying this compression to concepts and results coming from the PN modeling framework, we perceive the optimal DAP as a "dichotomy" of the RAS state space and we essentially seek a compact *"classifier"* that will effect this dichotomy. In this way we are able to tap upon concepts, insights and results that are coming directly from the relevant classification theory. Indeed, as it is revealed in the rest of this document, the methods pursued in this work open new ways for thinking about the considered problem that effectively complement all the previously used approaches. This new line of reasoning subsequently results into new fundamental insights and connects the overall analysis to very classical, and yet very powerful, representation frameworks and techniques. From a more practical standpoint, the methodology pursued in this paper has allowed the effective and efficient implementation of the maximally permissive DAP for very large-scale RAS, with sizes and underlying state spaces way beyond of those addressed in the current literature. This capability is effectively established upon the ability of our new methodology (i) to define explicitly the classifier design problem as a minimization problem over a certain parameter space, and (ii) to effect further reductions of this problem that facilitate its computational tractability, even for very large-scale RAS. Furthermore, an additional set of results that have been obtained from this problem definition, concerns the development of heuristics that can effectively balance the structural optimality of the sought classifier and the computational complexity that is involved in its development.

All the above ideas are pursued in this work in the context of a particular RAS class that is known as the *"Gadara"* RAS (Wang et al. (2009)). A salient feature of the Gadara RAS, that defines and facilitates all the developments presented in the rest of this document, is that the state dichotomy that is effected by the maximally permissive DAP, can be expressed as a set of linear inequalities (Wang et al. (2009)). From a more practical standpoint, the Gadara RAS was defined in the context of an ongoing project of ours, that is also called "Gadara", and is performed in collaboration with HP labs. The goal of the Gadara project is to develop a software tool that will automatically take a multi-threaded deadlock-prone program as input and instrument it with appropriate control logic so that the resultant code is deadlock-free. Hence, the entire problem addressed by Gadara reduces to the design and the deployment of a DAP that will manage the allocation of the "locks" shared by the concurrently executing processes (i.e., threads) in the context of the considered program (see Wang et al. (2008b,a); Kelly et al. (2009)). Furthermore, in the Gadara context, the deployed DAP must ensure the safe and live execution of the underlying processes in a way that (a) imposes the minimal necessary restriction in the execution of these processes, and (b) causes the minimum possible computational "overhead" for the underlying operating system. Requirement (a) implies that the maximal permissiveness of the applied control logic is of paramount importance in the considered application context. Requirement (b) introduces the additional notion of *"structural minimality"* for the derived supervisors that is pursued in this work.

In light of the above positioning of the paper objectives and contributions, the rest of the manuscript is organized as follows: Section 2 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. Section 3 presents the classification problem to be considered in this work, and 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 4 formulates the synthesis of a linear classifier for the reduced classification problem as a mixed integer program. Section 5 reports a series of computational experiments that demonstrate the efficacy of the proposed approach and the compact nature of the obtained classifiers. Finally, Section 6 concludes the paper by summarizing its main contributions and suggesting possible extensions of the presented results. Due to the space limitations imposed for this manuscript, most of our technical results are provided without formal proofs. These proofs can be found in Nazeem et al. (2010), that constitutes a more extensive and more complete development of the relevant results.

2. 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. As mentioned in the introductory section, this RAS class was motivated by the resource allocation addressed by the Gadara project, and therefore, it will be referred to as the Gadara RAS. From a more technical standpoint, the Gadara RAS is a specialization of the generic RAS structure introduced in Reveliotis (2005). An instance Φ from this class is defined by a triplet $< \mathcal{R}, \mathcal{P}, \mathcal{A} >$ where: (i) $\mathcal{R} = \{R_1, \ldots, R_m\}$ is the set of the system *resources*, each available with single multiplicity. (ii) $\mathcal{P} = \{J_1, \ldots, J_n\}$ 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. (iii) $\mathcal{A} : \bigcup_{j=1}^{n} \mathcal{S}_j \rightarrow \prod_{i=1}^{m} \{0,1\}$ 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 applied 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 $(A_{j,k+1} - A_{jk})^+$; and it is only upon this advancement that the process will release the resource units $|(A_{j,k+1} - A_{jk})^-|$, that are not needed anymore.

For complexity considerations, we also define the quantity $|\Phi| \equiv |\mathcal{R}| + |\bigcup_{j=1}^{n} \mathcal{S}_j|$ as the *"size"* of RAS Φ. Furthermore, 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. Finally, 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.

Modeling the dynamics of the Gadara RAS as a Finite State Automaton The dynamics of the Gadara RAS $\Phi =< \mathcal{R}, \mathcal{P}, \mathcal{A} >$, introduced in the previous paragraph, can be formally described by a *Deterministic Finite State Automaton (DFSA)* (Cassandras and Lafortune (2008)), $G(\Phi) = (S, E, f, \mathbf{s}_0, S_M)$, that is defined as follows: [2]

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

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

where, according to the adopted notation, $\mathcal{A}(\Xi_q)[i]$ denotes the allocation request for resource R_i that is posed by stage Ξ_q, and the unit element appearing in the right-hand-side of the equation expresses the unit resource capacities that define the class of Gadara RAS.

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) $\bar{E} = \{e_{rp} : \exists j \in 1,\ldots,n \text{ s.t. } \Xi_p$ 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.

[2] An alternative modeling of the resource allocation dynamics of the Gadara RAS through the Petri net modeling framework, is presented in Wang et al. (2009). We also notice, for completeness, that with respect to the RAS taxonomy presented in Reveliotis (2005), the Gadara RAS can be perceived as a Conjunctive-Disjunctive-RAS where (a) all resources have unit capacity and (b) processes can present cycling among their various stages.

3. The *state transition function* $f : S \times E \to S$ 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:

$$\mathbf{s}'[q] = \begin{cases} \mathbf{s}[q] - 1 & \text{if } \; q = r \\ \mathbf{s}[q] + 1 & \text{if } \; q = p \\ \mathbf{s}[q] & \text{otherwise} \end{cases}$$

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 $\mathbf{0}$, which corresponds to the situation when the system is empty of any process instances.

5. The *set of marked states* S_M is the singleton $\{\mathbf{s}_0\}$, and it expresses the requirement for complete process runs.

Let $\hat{f}$ denote the natural extension of the state transition function f to $S \times E^*$. 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:

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

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

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

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 LES The desired – or "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:

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

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_s. 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 establish that the subspaces S_{rs} and $S_{r\bar{s}}$ present some additional structure that will become useful in the design of the target classifier. To facilitate the formal statement and development of the relevant results we need the following definition:

Definition 1. Let $\mathbf{x}, \mathbf{x}'$ denote two vectors in the vector space $(\mathbb{Z}_0^+)^{\xi}$. Then, the (partial) ordering relationship "$\leq$" imposed on $(\mathbb{Z}_0^+)^{\xi}$ is defined by:

$$\mathbf{x} \leq \mathbf{x}' \iff (\forall i = 1,\ldots\xi, \;\; \mathbf{x}[i] \leq \mathbf{x}'[i]) \qquad (5)$$

Furthermore, the fact that $\mathbf{x} \leq \mathbf{x}'$ (resp. $\mathbf{x} \geq \mathbf{x}'$) and there is at least a pair of components $\mathbf{x}[i], \mathbf{x}'[i]$ for which the corresponding inequality is strict, will be denoted by $\mathbf{x} < \mathbf{x}'$ (resp. $\mathbf{x} > \mathbf{x}'$).

It should be clear from the discussion provided in the previous paragraphs that the ability of the activated processes in a given RAS state $\mathbf{s} \in S$ to proceed to completion, depends on the existence of a sequence $< \mathbf{s}^{(0)} \equiv \mathbf{s}, e^{(1)}, \mathbf{s}^{(1)}, e^{(2)}, \mathbf{s}^{(2)}, \ldots, \mathbf{s}^{(n-1)}, e^{(n)}, \mathbf{s}^{(n)} \equiv \mathbf{s}_0 >$, such that at every state $\mathbf{s}^{(i)},\ i = 0, 1 \ldots, n-1$, the free (or "slack") resource capacities at that state enable the job advancement corresponding to event $e^{(i+1)}$. Furthermore, if such a terminating sequence exists for a given state $\mathbf{s}$, then the event feasibility condition defined by Equation 1 implies that this sequence will also provide a terminating sequence for every other state $\mathbf{s}' \leq \mathbf{s}$. On the other hand, if state $\mathbf{s}$ possesses no terminating sequences, then it can be safely inferred that no such terminating sequences will exist for any other state $\mathbf{s} \leq \mathbf{s}'$ (since, otherwise, there should also exist a terminating sequence for $\mathbf{s}$, according to the previous remark). The next proposition provides a formal statement to the above observations; these results are well known in the literature, and therefore, their formal proof is omitted.

Proposition 1. Let $\mathbf{s}$, $\mathbf{s}'$ denote two states of a given Gadara RAS Φ. Then,

(1) $\mathbf{s} \in S_s \ \wedge\ \mathbf{s}' \leq \mathbf{s} \implies \mathbf{s}' \in S_s$
(2) $\mathbf{s} \in S_{\bar{s}} \ \wedge\ \mathbf{s} \leq \mathbf{s}' \implies \mathbf{s}' \in S_{\bar{s}}$

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

Definition 2. Given a Gadara RAS $\Phi = (\mathcal{R}, \mathcal{P}, \mathcal{A})$,

(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 $\mathbf{s}' > \mathbf{s}$;
(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 $\mathbf{s}' < \mathbf{s}$.

Also, 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 $\bar{S}_{r\bar{s}}$.

The binary nature of the state space of the Gadara RAS and its implications For any Gadara RAS $\Phi =< \mathcal{R}, \mathcal{P}, \mathcal{A} >$, Equation 1 when combined with the non-zero nature of the resource allocation requests $\mathcal{A}_{jk}$ further imply that the RAS state vector $\mathbf{s}$ is of a binary nature; i.e., the state space S of the DFSA $G(\Phi)$, as well as any other subspace of S, consist of a number of extreme points on the ξ-dimensional hypercube $\mathcal{C}$ defined by $\mathcal{C} \equiv \{(x_1, x_2, \ldots, x_\xi) :\ 0 \leq x_i \leq 1,\ \forall i = 1, \ldots, \xi\}$.

In Nazeem et al. (2010) we show that every extreme point $\mathbf{x}$ of $\mathcal{C}$ can be separated from the rest by a single linear inequality

$$\mathbf{a}^T \cdot \mathbf{x} \leq b \tag{6}$$

where

$$\mathbf{a}[i] := \begin{cases} 1, \text{ if } \mathbf{x}[i] = 1 \\ -1, \text{ if } \mathbf{x}[i] = 0 \end{cases} \quad ; \quad b := \sum_{i=1}^{\xi} \mathbf{x}[i] - 1 \tag{7}$$

A further implication of the above result is stated in the following proposition:

Proposition 2. Consider two sets X and $\hat{X}$ that consist of binary vectors from some ξ-dimensional space, and further assume that $\hat{X} \subset X$. Then, there exists a system of linear inequalities $\{\mathbf{a}_i^T \cdot \mathbf{x} \leq b_i,\ i = 1, \ldots, \nu\}$ that is satisfied by every $\mathbf{x} \in \hat{X}$ and it is violated by every $\mathbf{x} \in X \backslash \hat{X}$, and therefore, it can function as a linear classifier for $\hat{X}$ and $X \backslash \hat{X}$.

A formal proof of Proposition 2 can be found in Nazeem et al. (2010). Here we focus on its more practical significance in the context of the presented developments, i.e., the assertion that in the class of Gadara RAS, the safe and the unsafe subspaces will always be separated by a set of linear inequalities. The next two sections formalize the relevant classification structure through the concept of a "linear classifier" and they investigate the problem of the effective construction of such a classifier for any given Gadara RAS.

3. THE CLASSIFIER DESIGN PROBLEM IN THE CONTEXT OF THE GADARA RAS AND ITS SIMPLIFICATION

The classification problem considered in this work This section considers the problem of synthesizing an effective and compact classifier that will separate the reachable safe subspace S_{rs} from the reachable unsafe subspace $S_{r\bar{s}}$, for any given RAS Φ that belongs to the class of Gadara RAS. Proposition 2 of the previous section guarantees the existence of a set of linear inequalities that will perform the requested separation. Our objective is to find the minimum number of linear inequalities that achieves the separation. The next definition provides a formal characterization of the concepts that are necessary for the exact positioning of our problem:

Definition 3. Consider two vector sets G and H from an ξ-dimensional vector space V.

(1) We shall say that sets G and H are *linearly separated* by a set of k linear inequalities $\{(\mathbf{A}(i,.), b_i) : i \in \{1, \cdots, k\}\}$ iff

$$(\forall \mathbf{g} \in G :\ \forall i \in \{1, \cdots, k\},\ \mathbf{A}(i,.) \cdot \mathbf{g} \leq b_i) \quad \wedge$$
$$(\forall \mathbf{h} \in H : \exists i_s \in \{1, \cdots, k\},\ \mathbf{A}(i_s,.) \cdot \mathbf{h} > b_{i_s}) \tag{8}$$

(2) A linear classifier – or separator – for vector sets G and H is *minimal*, iff it uses the minimum possible number of linear inequalities that can separate these two sets.

Hence, the problem addressed in this section can be succinctly stated as follows:

Definition 4. **– The considered classification problem** Given a Gadara RAS Φ, construct a minimal linear separator for the vector sets corresponding to the sub-spaces S_{rs} and $S_{r\bar{s}}$, i.e., the reachable safe and the reachable unsafe states of the considered RAS Φ.

A major difficulty for the systematic construction of the aforementioned classifier for any practical instantiation of the Gadara RAS, is the huge cardinality of the sets S_{rs} and $S_{r\bar{s}}$. In the following, we show that the properties of the state space, S, of the Gadara RAS established in Section 2, can enable the reduction of the aforestated linear classification problem to another linear classification problem that is defined on a pair of vector sets with a smaller cardinality and dimensionality than their original counterparts. This reduction does not compromise either the effectiveness or the minimality of the derived classifier; the relevant technical developments are detailed in Nazeem et al. (2010) and they are summarized next in a series of "technical results".

Technical Result 1. **– 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 2, 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. Therefore, the sought classifier needs to separate effectively only the set of reachable safe states S_{rs} and the subset of the reachable unsafe states $S_{r\bar{s}}^b$ that are reachable from some state $\mathbf{s}' \in S_{rs}$ in a single transition.

Technical Result 2. **– Thinning the sets S_{rs} and $S_{r\bar{s}}^b$ by respectively focusing on their maximal and minimal elements** We remind the reader that $\bar{S}_{rs}$ denotes the maximal elements of the

set S_{rs} according to the ordering relationship of Definition 1. Also, let $\bar{S}^b_{r\bar{s}}$ denote the minimal elements of the set $S^b_{r\bar{s}}$ according to the same relationship. Then, any linear separator $(\mathbf{A},\mathbf{b})$ for the sets $\bar{S}_{rs}$ and $\bar{S}^b_{r\bar{s}}$ with non-negative coefficients a_{ij} and b_i, is also an effective separator for the entire sets S_{rs} and $S^b_{r\bar{s}}$. Furthermore, the set of minimal linear separators for the classification problem of Definition 4 will always contain a separator $(\mathbf{A},\mathbf{b})$ with non-negative coefficients a_{ij} and b_i, [3] and therefore, the restriction of the classification problem of Definition 4 to separators with non-negative coefficients compromises neither the feasibility of the problem nor the optimality of the derived solution.

Technical Result 3. **– Converting the separation problem of $\bar{S}_{rs}$ and $\bar{S}^b_{r\bar{s}}$ to an equivalent separation problem of reduced dimensionality** Let V denote the ξ-dimensional vector space supporting the vector sets $\bar{S}_{rs}$ and $\bar{S}^b_{r\bar{s}}$, I denote the set of coordinates of space V, and I_0 denote the set of coordinates that are identically zero in the vectors of $\bar{S}^b_{r\bar{s}}$. Also, let P denote the orthogonal projection that removes from the elements of $\bar{S}_{rs}$ and $\bar{S}^b_{r\bar{s}}$ those coordinates that correspond to the aforementioned set I_0. Set $I_P \equiv I \backslash I_0$, and let V_P denote the $|I_P|$-dimensional sub-space supporting the projection P. Also, let $\Gamma : \mathbb{N} \rightarrow \mathbb{N}$ be a bijection that maps the elements of the coordinate set I_P to the coordinates 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, there exists a set of k hyperplanes with non-negative coefficients, Q, that separates the projected sets $P(\bar{S}_{rs})$ and $P(\bar{S}^b_{r\bar{s}})$ in subspace V_P, iff there exists a set of k hyperplanes, H, that separates the sets $\bar{S}_{rs}$ and $\bar{S}^b_{r\bar{s}}$ in the original space V. Furthermore, the separator H can be obtained from separator Q as follows:

$$b_{h_i} := b_{q_i} \;\; \wedge$$
$$\forall j \in I_0 : \; \mathbf{a}_{h_i}[j] := 0 \;\; \wedge \;\; \forall j \in I_P : \; \mathbf{a}_{h_i}[j] := \mathbf{a}_{q_i}[\Gamma(j)] \quad (9)$$

Technical Results 1–3 imply that one can construct the sought separator H for the state sets S_{rs} and $S_{r\bar{s}}$ by first extracting the smaller vector sets (in terms of, both, cardinality and dimensionality) $P(\bar{S}_{rs})$ and $P(\bar{S}^b_{r\bar{s}})$, design a linear separator Q with non-negative coefficients for the last two sets, and finally obtain H through Equation 9. In fact, the projected set $P(\bar{S}_{rs})$ can be further "thinned" to set $\overline{P(\bar{S}_{rs})}$ by recognizing that for the set $\bar{S}_{rs}$ the aforestated projection P is non-bijective, and therefore, it can create additional dominance among the elements of $P(\bar{S}_{rs})$ according to the ordering relationship of Definition 1; set $\overline{P(\bar{S}_{rs})}$ is obtained from set $P(\bar{S}_{rs})$ by identifying and removing the dominated elements. The entire workflow described above is depicted in the flowchart of Figure 1. The detailed algorithms supporting the execution of the various pre-processing stages in this flowchart are straightforward and therefore omitted from this discussion; interested readers can trace them in Nazeem and Reveliotis (2010a). In the next section we discuss the support of the last stage in the flowchart of Figure 1, i.e., the construction of the minimal linear separator Q for the vector sets $\overline{P(\bar{S}_{rs})}$ and $P(\bar{S}^b_{r\bar{s}})$.

[3] This result is a consequence of the monotonicity of state safety that is established in Proposition 1; its detailed development can be found in Nazeem et al. (2010).

4. SYNTHESIZING CLASSIFIER Q THROUGH MATHEMATICAL PROGRAMMING

The MIP formulation In this section we provide a Mixed Integer Programming (MIP) formulation for the construction of the separator Q, that was specified in Section 3. We remind the reader that the primary inputs to this final stage of the proposed design process, are:

- the elements $\mathbf{x}_i$ of the projected safe state set $\overline{P(\bar{S}_{rs})}$, and
- the elements $\mathbf{y}_i$ of the projected unsafe state set $P(\bar{S}^b_{r\bar{s}})$.

In the subsequent discussion, we shall set $m_s \equiv |\overline{P(\bar{S}_{rs})}|$, $m_u \equiv |P(\bar{S}^b_{r\bar{s}})|$, and $n \equiv |I_P|$, i.e., n denotes the dimensionality of the subspace V_P supporting the vectors $\mathbf{x}_i$ and $\mathbf{y}_i$. Some additional inputs that parameterize this last stage of our design process and provide additional controls to it, are as follows:

- The parameter w which provides an upper bound for the "size" of – i.e., the number of inequalities employed by – the sought separator. Such an upper bound is readily obtained from Proposition 2 of Section 2 and the Technical Results 1–3 of Section 3 as $w = m_u$.
- A strictly positive parameter ε that controls the minimum distance of the points $\mathbf{y}_i$ from the separating hyperplanes and should be priced sufficiently close to zero. This parameter can be perceived as a "degree of separation" that is enforced between the two sets $\overline{P(\bar{S}_{rs})}$ and $P(\bar{S}^b_{r\bar{s}})$. 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 $\{\varepsilon_i\}$ such that $\varepsilon_i \rightarrow 0^+$, and consider the stability of the size of the obtained supervisors.
- A strictly positive parameter M that is useful for the modelling 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:

- z_l, $l = 1,\ldots,w$, is a binary variable that is priced to one if the l-th inequality is used for separation and to zero otherwise.
- $(\mathbf{A}[l,.],\mathbf{b}[l])$, $l = 1,\ldots,w$, are the coefficients to be employed by the l-th separating linear inequality.
- δ_{il}, $i = 1,\ldots,m_u$, $l = 1,\ldots,w$, is a binary variable that must be priced to one if the state $\mathbf{y}_i$ and the hyperplane $(\mathbf{A}[l,.],\mathbf{b}[l])$ satisfy the inequality $\mathbf{A}[l,.]\cdot\mathbf{y}_i \leq \mathbf{b}[l]+\varepsilon$.

Finally, the formulation itself takes the following form:

$$\min \sum_{l=1}^{w} z_l \quad (10)$$

$$\forall i \in \{1,\cdots,m_s\},\ \forall l \in \{1,\cdots,w\} : \; \mathbf{A}[l,.]\cdot\mathbf{x}_i - \mathbf{b}[l] \;\leq\; 0 \quad (11)$$

$$\forall i \in \{1,\cdots,m_u\},\ \forall l \in \{1,\cdots,w\} : \quad \mathbf{A}[l,.]\cdot\mathbf{y}_i - \mathbf{b}[l] + M\cdot\delta_{il} \geq\; \varepsilon \quad (12)$$

$$\forall i \in \{1,\cdots,m_u\} : \; \sum_{l=1}^{w} \delta_{il} \;\leq\; w-1 \quad (13)$$

$$\forall l \in \{1,\cdots,w\},\ \forall j \in \{1,\cdots,n\} : \; 0 \;\leq\; \mathbf{A}[l,j] \;\leq\; z_l \quad (14)$$

$$\forall i \in \{1,\cdots,m_u\},\ \forall l \in \{1,\cdots,w\} : \; \delta_{il} \in \{0,1\} \quad (15)$$

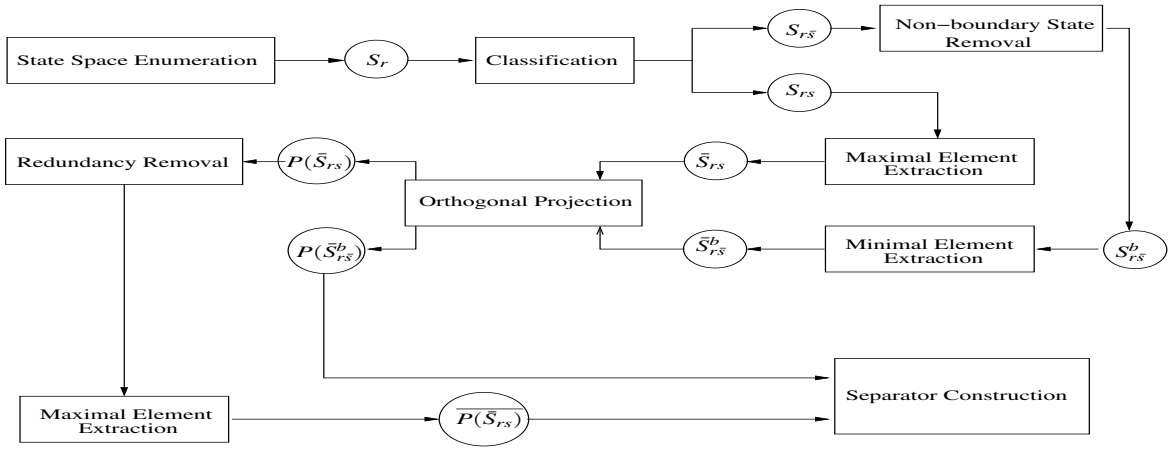

Fig. 1. The proposed workflow for the construction of a minimal linear separator for the safe and unsafe subspaces of a Gadara RAS.

$$\forall l \in \{1,\cdots,w\}:\ z_l \in \{0,1\} \tag{16}$$

The validity of the above MIP formulation as a construction tool for the sought separator Q can be established as follows: First, the reader should notice that Equation 10 defines the objective of the formulation as the minimization of the number of hyperplanes that will be actively used for the pursued separation. Also, Equations 15 and 16 respectively enforce the binary nature of the variables δ_{il} and z_l. On the other hand, the constraints of Equation 14 enforce (i) the non-negativity of the matrix $\mathbf{A}$ in the returned solution, and also (ii) the requirement that the coefficients of inactive inequalities should be set to zero. An additional implication of the constraints expressed by the right inequality in Equation 14, is the restriction of all the elements of matrix $\mathbf{A}$ to values no greater than one. This effect does not compromise the generality of the obtained solution, since the values of the intercepts $\mathbf{b}[l],\ l = 1,\ldots,w$, are not restricted by this constraint.[4] On the other hand, we shall see in the following discussion that the restriction of the elements of matrix $\mathbf{A}$ in the interval $[0,1]$ enables the effective resolution of some other aspects of the formulation.

The separation logic is primarily expressed by the constraints of Equations 11–13. More specifically, the constraints of Equation 11 force every point corresponding to a safe vector $\mathbf{x}_i$, $i = 1,\ldots,m_s$, to lie below each separating hyperplane. When combined with the non-negativity of vectors $\mathbf{x}_i$ and of matrix $\mathbf{A}$, this constraint also enforces the non-negativity of the intercepts $\mathbf{b}[l]$. On the other hand, the constraints of Equations 12 and 13 require that every point corresponding to an unsafe vector $\mathbf{y}_i$, $i = 1,\ldots,m_u$, lies above of at least one of the separating hyperplanes. To understand the detailed mechanism that enforces this requirement, first notice that for any vector $\mathbf{y}_i$ and inequality $(\mathbf{A}[l,.],\mathbf{b}[l])$ such that $\mathbf{A}[l,.]\cdot\mathbf{y}_i \geq \mathbf{b}[l]+\varepsilon$, Equation 12 is satisfied irrespective of the value of the binary variable δ_{il} (provided that $M \geq 0$). In the opposite case, the corresponding variable δ_{il} must be set to one, in order to attain the satisfaction of Equation 12 (provided that the non-negative parameter M is sufficiently large). But at the same time, Equation 13 requires that, for every point $\mathbf{y}_i$, at least one of the corresponding variables δ_{il}, $l = 1,\ldots,w$, must be equal to zero. Therefore, in any feasible solution of the proposed formulation, every point $\mathbf{y}_i$ must be above at least one of the hyperplanes $(\mathbf{A}[l,.],\mathbf{b}[l])$.

The above discussion also reveals the condition for the proper pricing of the parameter M:

$$M \ \geq\ \sup\{[\mathbf{A}[l,.]\cdot\mathbf{y}_i - \mathbf{b}[l] - \varepsilon]^-\} \tag{17}$$

where $[a]^- \equiv |\min\{0,a\}|$ and the supremum is taken over all pairs of vectors $\mathbf{y}_i$ and inequalities $(\mathbf{A}[l,.],\ \mathbf{b}[l])$ in any viable solution. An upper bound for the quantity on the right-hand-side of Equation 17 can be obtained by setting $\mathbf{A}[l,.]\cdot\mathbf{y}_i = 0$, and considering an upper bound for $\sup\{b[l]+\varepsilon\}$, where the latter is taken over all the inequalities that can constitute a viable component of the sought separator Q. To obtain this last upper bound, we recall that the minimality of Q implies that, for every constituent inequality $(\mathbf{A}[l,.],\ \mathbf{b}[l])$, there exists at least one point $\mathbf{y}_j$ such that $\mathbf{A}[l,.]\cdot\mathbf{y}_i \geq \mathbf{b}[l]+\varepsilon$. But then, Equations 14 and 16, together with the binary nature of vectors $\mathbf{y}_i$, further imply that $\sup\{b[l]+\varepsilon\} \leq n$, where n denotes the dimensionality of the supporting hyperspace V_P. Hence, M can be set equal to n during the solution of the considered formulation.

The next theorem provides a formal statement of the validity of the MIP formulation of Equations 10–16 as a classifier design tool for the classification problem considered in this work, and it is an immediate implication of all the above discussion provided in this section and of the developments of Section 3.

Theorem 3. The application of the formulation of Equations 10–16 to the sets $\overline{P(\bar{S}_{rs})}$ and $P(\bar{S}^b_{r\bar{s}})$ corresponding to any given Gadara RAS Φ, returns a minimal linear classifier for these two sets, and through Equation 9, a minimal linear classifier for the original sets S_{rs} and $S^b_{r\bar{s}}$.

Complexity considerations It should be clear from the above discussion that the MIP formulation of Equations 10–16 involves $(m_u+1)\cdot w$ binary variables, $(|I_P|+1)\cdot w$ real variables and $w\cdot(m_s+m_u+|I_P|)+m_u$ 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 $P(\bar{S}^b_{r\bar{s}})$ and the dimensionality of their supporting sub-space V_P. In the next section we provide an extensive set of computational results that demonstrate the tractability of the MIP formulation of Equations 10–16 for RAS instantiations from the Gadara RAS class with size comparable to those encountered in many practical applications. These results make us believe that the MIP formulation of Equations 10–16 will be an effective computational tool for the synthesis of minimal linear separators Q for a very broad range of the Gadara RAS instantiations to be encountered in "real-life" applications. Finally, we notice, that the material of Nazeem et al. (2010) also presents a heuristic that can provide an effective trade-off between the structural minimality of the

[4] In other words, every derived inequality $(\mathbf{A}[l,.],\mathbf{b}[l])$ can have its coefficients normalized by the maximum element of the row vector $\mathbf{A}[l,.]$ and this normalization will not affect its informational content.

[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.

obtained separator, as measured by the number of the involved inequalities, and the computational effort for its development. This heuristic is essentially a greedy approach that constructs the sought separator Q one hyperplane at a time, and it can be shown to return a solution of size no larger than $\ln|P(\bar{S}^b_{r\bar{s}})|$ times the size of the optimal separator.

5. COMPUTATIONAL RESULTS

In this section we report the results from a series of computational experiments, in which we applied the DAP design methodology described in the earlier parts of this manuscript upon a number of randomly generated instantiations of Gadara RAS. The particular RAS dynamics adopted in the presented experiments were chosen so that they imitate closely the lock allocation and deallocation in real computer programs. We remind the reader that according to the class definition, all resources in a Gadara RAS possess unit capacity. On the other hand, each of the generated instances was further specified by:

I. The number of resources in the system; the range of this parameter was between 1 and 14.

II. The number of process types in the system; the range of this parameter was between 2 and 14. Furthermore, in the considered experiments all process types were assumed to have a simple linear structure, with the corresponding graphs $\mathcal{G}_j$ being simple paths (i.e., paths without any loops) for all j.

III. The number of transitions in each process, with each transition corresponding to a single resource acquisition or a single resource release. The range of this parameter was between 2 and 14, but additional logic was applied to ensure a meaningful resource allocation sequence; hence, the eventual number of transitions appearing in every generated process differed by the originally specified number. In particular, upon its initiation, a process was allocated randomly one of the system resources. At every subsequent transition, the process was either releasing one of its allocated resources, or it was acquiring one of the remaining resources. The association of any given transition with a resource release or a resource acquisition was equiprobable, except for the case where the process found itself possessing no resources; in that case the next transition was an acquisition with probability one. Similarly, the selection of the resource to be released by the process during a release transition, or the resource to be added to its current acquisitions during an acquisition transition, was determined equiprobably among the corresponding resource sets. Once the pre-specified number of transitions was determined as described above, the necessary number of release transitions was appended so that the process eventually returned all the acquired resources. Furthermore, in order to remain consistent with the definition of the Garada RAS provided in Section 2, all process stages resulting from the above construction that might correspond to zero resource allocation were identified and pruned from the process-defining sequence.

The employed RAS generator was encoded in Java, and it was compiled and linked by Java 1.6.0. Each generated RAS instance, Φ, was subjected to the DAP design process described in Figure 1. The detailed algorithms employed for the execution of the various pre-processing steps indicated in Figure 1, can be found in Nazeem and Reveliotis (2010a). The construction of the linear classifier itself was performed according to the methodology described in Section 4, where we imposed a hard limit of 30 minutes (or 1800 secs) for the solution of the MIP formulation of Equations 10–16. For instances that were not completely solved within this time budget, the solver terminated prematurely and reported the best feasible solution identified up to that point. All our computational 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. The algorithms involved in the preprocessing stages of the proposed methodology were encoded in C++, compiled and linked by the GNU g++ compiler under Unix, while the MIP formulation of Equations 10–16 was solved through ILOG CPLEX 11.1 with ILOG Concert technology using C++.

Table 1 reports a representative sample of the results that we obtained in our experiments. This sample has been selected from data resulting by the application of the proposed methodology on more than 500 instances of the Gadara RAS generated according to the logic described in the earlier parts of this section. The reported cases are ordered in decreasing magnitude of the corresponding $|S_{rs}|$, the number of reachable and safe states (second column in the presented table). The first column in Table 1 reports the dimensionality of the original state space corresponding to each listed configuration. Columns 3 – 9 report the cardinalities of the state subsets extracted through the various processing stages depicted in Figure 1, and also, the dimensionality reduction that was obtained through the projection P, discussed in Section 3. Columns 10, entitled by k_{MIP}, reports the number of linear inequalities in the solution returned by the MIP formulation of Section 4. Furthermore, the qualification [O] and [F] of the values reported in Column 10 indicates whether the obtained solution was optimal or just a feasible one (this would happen if the solution algorithm was terminated prematurely, upon the exhaustion of the 30 minute budget). Finally, Columns 11 and 12, respectively entitled by $t_{thinning}$ and t_{MIP}, report the amount of computing time (in seconds) that was required to (i) execute the pre-processing steps indicated in Figure 1, and (ii) solve the MIP formulation of Section 4. It should be clear from the perusal of the data provided in Table 1 that (i) the set-"thinning" process of Figure 1 results to very significant (in fact, dramatic) reductions of the data that must be explicitly considered by the MIP formulation of Section 4, and (ii) these reductions subsequently enable the solution of this MIP formulation for very large configurations from the considered RAS class. In fact, as we commented in the introductory section, to the best of our knowledge, this is the first line of work that has been able to effectively develop practical, compact implementations of the maximally permissive DAP for RAS of the scale reported in Table 1.

6. CONCLUSIONS

This work has developed a novel methodology for the effective deployment of the maximally permissive DAP in the context of the complex resource allocation that takes place in many contemporary applications. Our results are enabled by (i) a careful distinction between the off-line and the on-line parts of the computation that is required for the effective characterization and deployment of the target policy, and (ii) the treatment of the on-line implementation of the target policy as a classifier that effects the separation of the underlying RAS state space to its safe and unsafe regions. Extensive numerical experimentation reported in the last part of the manuscript confirmed the tractability of the approach and its ability to provide compact implementations of the maximally permissive DAP for RAS with a very large size and very large state spaces.

Future work will seek the extension and the detailed implementation of the basic approach outlined in the previous paragraph, to RAS classes beyond that of the Gadara RAS, that was the primary focus of this work. Some initial results in this direction can be found in Nazeem and Reveliotis (2010b).

REFERENCES

Araki, T., Sugiyama, Y., and Kasami, T. (1977). Complexity of the deadlock avoidance problem. In *2nd IBM Symp. on*

Table 1. A sample of our experimental results

ξ	$\lvert S_{rs}\rvert$	$\lvert S_{r\bar{s}}\rvert$	$\lvert S^b_{r\bar{s}}\rvert$	$\lvert\bar{S}_{rs}\rvert$	$\lvert\bar{S}^b_{r\bar{s}}\rvert = \lvert P(\bar{S}^b_{r\bar{s}})\rvert$	$\lvert I_P\rvert$	$\lvert P(\bar{S}_{rs})\rvert$	$\lvert\overline{P(\bar{S}_{rs})}\rvert$	k_{MIP}	$t_{thinning}$	t_{MIP}
61	8,696,502	71,677	71,677	162,052	5	7	35	9	1 [O]	80	0
75	5,699,463	268,807	267,853	389,332	43	23	3,613	315	4 [O]	57	26
107	5,696,776	1,165,958	1,021,301	1,544,165	155	35	3,286	533	12 [F]	100	1,800
70	5,501,728	1,321,928	1,137,856	152,570	21	21	1,340	245	4 [O]	70	10
74	4,432,641	283,561	278,490	660,951	28	21	3,068	367	4 [O]	42	4
60	3,994,272	348,576	348,576	105,606	12	10	44	22	2 [O]	39	0
97	3,718,540	706,177	622,035	938,461	122	31	2,672	418	8 [F]	56	1,800
79	3,144,658	249,690	246,301	754,307	75	22	1,366	330	9 [F]	30	1,800
69	3,102,964	56,752	56,752	201,944	38	20	2,386	235	4 [O]	25	12
87	2,841,494	834,672	750,951	386,960	40	28	1,802	175	4 [O]	43	45
112	2,521,030	556,743	518,684	929,498	168	38	2,633	643	10 [F]	42	1,800
64	2,501,508	501,060	433,632	54,496	18	17	307	71	3 [O]	27	1
74	1,953,671	110,937	103,311	109,964	57	17	272	81	2 [O]	18	0
73	1,906,704	152,387	150,349	423,799	50	21	1,139	266	6 [F]	17	1,800
106	1,696,349	382,291	352,622	587,314	152	36	2,295	553	8 [F]	25	1,800
63	1,567,434	17,579	17,579	84,109	29	17	1,194	163	3 [O]	11	1
96	1,240,726	188,189	181,689	413,175	113	33	2,005	467	8 [F]	13	1,800
67	1,197,240	121,442	97,434	30,481	13	15	86	30	2 [O]	11	0
53	963,900	9,618	9,618	29,354	5	7	35	9	1 [O]	6	0
71	911,283	209,248	199,507	98,772	13	18	380	67	2 [O]	8	0
59	849,136	197,624	165,200	10,886	18	17	307	71	3 [O]	8	1

Mathematical Foundations of Computer Science, 229–257. IBM.

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

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

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

Kelly, T., Wang, Y., Lafortune, S., and Mahlke, S. (2009). Eliminating concurrency bugs with control engineering. *IEEE Computer*, 42(12), 52–60.

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

Li, Z., Zhou, M., and Wu, N. (2008). 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.

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

Nazeem, A. and Reveliotis, S. (2010a). Designing maximally permissive deadlock avoidance policies for complex resource allocation systems through classification theory. Technical report, Georgia Inst. of Technology.

Nazeem, A. and Reveliotis, S. (2010b). A practical approach to the design of maximally permissive liveness-enforcing supervisors for complex resource allocation systems. In *Proceedings of the 6th IEEE Conf. on Automation Science and Engineering*. IEEE.

Nazeem, A., Reveliotis, S., Wang, Y., and Lafortune, S. (2010). Designing maximally permissive deadlock avoidance policies for sequential resource allocation systems through classification theory: the linear case. Technical report, Georgia Inst. of Technology.

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

Reveliotis, S. (2007). Algebraic deadlock avoidance policies for sequential resource allocation systems. In M. Lahmar (ed.), *Facility Logistics: Approaches and Solutions to Next Generation Challenges*, 235–289. Auerbach Publications.

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

Wang, Y., Kelly, T., Kudlur, M., Lafortune, S., and Mahlke, S. (2008a). Gadara: Dynamic deadlock avoidance for multithreaded programs. In *OSDI '08 – Proceedings of the 8th Usenix Symposium on Operating Systems Design and Implementation*, 281–294. USENIX Association.

Wang, Y., Kelly, T., Kudlur, M., Mahlke, S., and Lafortune, S. (2008b). The application of supervisory control to deadlock avoidance in concurrent software. In *WODES '08 – Proceedings of the 9th International Workshop on Discrete Event Systems*, 287 – 292. IEEE.

Wang, Y., Liao, H., Reveliotis, S., Kelly, T., Mahlke, S., and Lafortune, S. (2009). Gadara nets: Modeling and analyzing lock allocation for deadlock avoidance in multithreaded software. In *Proceedings of the 48th IEEE Conference on Decision and Control*, 4971 – 4976.

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