The Thinning Problem*

Spyros A. Reveliotis and Jin Young Choi
School of Industrial & Systems Engineering
Georgia Institute of Technology
{spyros,choijy68}@isye.gatech.edu

Abstract

The main problem addressed in this work is how to confine the set of sequential processes of a disjunctive / conjunctive resource allocation system (D/C-RAS) to a subset of their feasible behaviors while optimizing some performance criterion. We provide a canonical characterization of this problem in the form of a Mixed Integer Programming (MIP) formulation, for the case that the optimized performance criterion is the maximization of the system throughput. However, an important additional development of this effort is a complete analytical characterization of the performance control problem of D/C-RAS. This characterization is based on an effective integration of recent results on the liveness-enforcing supervision of D/C-RAS with the theory of Markov Decision Processes, and it concretizes the interaction between the D/C-RAS logical and performance control. Throughout the paper development, a small example highlights the involved concepts and formulations.

Keywords: Resource Allocation Systems, Deadlock Avoidance, Markov Decsion Processes, Performance Modelling and Analysis, Congestion Control

1 Introduction

A general statement of the main problem addressed in this work is as follows: Given a set of sequential processes, each of which can be executed through a number of *process plans* – i.e., sequences of processing stages with each

*An abridged version of this paper was presented at the 7^{th} Workshop on Discrete Event Systems (WODES'04).

stage requesting the exclusive allocation of some resource set – and a *"budget"* on the *"active"* process plans for each process, select the process plans to be activated with respect to each process, so that some performance index is optimized. As we shall see in the following, the entire set of process plans available for each process can be effectively represented by a digraph with its nodes defined by the set of processing stages, and its edges indicating all the possible transitions among these stages. In the context of this representational framework, the problem can be posed as the selection of a number of edges from each digraph, so that the restriction of the system operation on the process plans expressed by the selected edges optimizes the aforementioned performance index. Since, this edge selection process is equivalent to the removal of the remaining edges from the corresponding process graphs, we perceive the entire problem as a *"thinning"* operation of the original process graphs.

A practical motivation for this problem has been our earlier work on the deadlock avoidance of resource allocation systems (RAS) with complex processes [3, 13]. As it was established in [3], as long as (i) the process-defining logic is finitely terminating, and (ii) the system has sufficient resources to support the execution of a single instance from any given process type, the problem of synthesizing a liveness-enforcing supervisor (LES) for these complex RAS can be reduced to the LES synthesis problem for another simpler RAS. Specifically, in this new RAS, the sequential logic for any process is modelled by a strongly connected state machine that encodes all the possible execution sequences for an instance of this process that runs in isolation through the considered RAS. Hence, the resulting RAS belongs to the class of D(isjunctive)/C(onjunctive)-RAS [15], for which efficient LES are readily available in [6, 11]. However, a computational limitation for this approach comes from the fact that the new representation of the process-defining logic will not be polynomially related to the original process representation.[1] Thus, the thinning problem arises in this context as a natural proposition for obtaining LES of polynomial complexity with respect to the size of the original RAS representation.

In the following, we focus on the particular criterion of *maximizing the RAS throughput*, and we provide a canonical characterization of the corresponding thinning problem in the form of a *Mixed Integer Programming (MIP)* formulation. An important additional development of this effort is

[1] This results from the fact that the new process representation is the reachability graph modelling the behavior of any single process instance executing in isolation in the considered RAS, trimmed with respect to a state that models the initial and final state of the process evolution; c.f. [3].

the derivation of a complete formulation of the D/C-RAS performance control problem. This formulation is based on an effective integration of some recent results on the D/C-RAS liveness-enforcing supervision [6, 11] with the theory of Markov Decision Processes (MDP) [12].

Collectively, the above results provide new analytical insights regarding the interaction of logical and performance-oriented control, and they complement some prior work of ours presented in [4, 14]. However, when viewed from a more practical computational standpoint, the applicability of the derived formulations is limited by the fact that the numbers of the involved variables and constraints are a super-polynomial function of the size of the underlying system configuration. Therefore, there is a remaining need for effective approximations to the optimal solutions of the presented formulations that can be obtained in a computationally efficient manner. Some preliminary results towards developing such effective approximations for the D/C-RAS performance control problem can be found in [5]; on the other hand, the development of computationally efficient methods for the thinning problem itself is part of our current investigations.

The rest of the paper is organized as follows: Section 2 introduces the considered class of D/C-RAS, and it surveys the currently available results on the development of liveness-enforcing supervisors for these systems. Section 3 proceeds to the modelling of the timed dynamics of the logically controlled D/C-RAS, and it provides a linear programming (LP) formulation for their performance control problem, based on concepts and techniques borrowed from the field of Continuous-Time Markov Decision Processes (CT-MDP) [12]. Section 4 builds upon the development of the earlier sections in order to (i) provide a formal characterization of the thinning problem, and (ii) derive a Mixed Integer Programming (MIP) formulation for it. Finally, Section 5 concludes the paper and discusses directions for future work.

2 Disjunctive / Conjunctive Resource Allocation Systems and their Liveness-Enforcing Supervision

This section introduces the considered class of D/C-RAS, and surveys the currently available results on the development of liveness-enforcing supervisors for these systems.

D/C-RAS The class of *Disjunctive / Conjunctive Resource Allocation Systems (D/C-RAS)*, considered in this work, is formally characterized as follows:

Definition 1 *A* disjunctive / conjunctive resource allocation system (D/C-RAS) *is defined as an 5-tuple* $\Phi = < \mathcal{R}, C, \mathcal{P}, \mathcal{A}, \mathcal{T} >$ *where:*

1. $\mathcal{R} = \{R_1, \ldots, R_m\}$ *is the set of the system* resource types*.*

2. $C : \mathcal{R} \rightarrow Z^+$ *– the set of strictly positive integers*[2] *– is the system* capacity *function, characterizing the number of identical units from each resource type available in the system. Resources are considered to be* reusable*, i.e., each allocation cycle does not affect their functional status or subsequent availability, and therefore,* $C(R_i) \equiv C_i$ *constitutes a system* invariant *for each* i*.*

3. $\mathcal{P} = \{\Pi_1, \ldots, \Pi_n\}$ *denotes the set of the system* process types *supported by the considered system configuration. Each process type* Π_j *is a composite element itself, in particular,* $\Pi_j =< \mathcal{S}_j, \mathcal{G}_j >$*, where:*

   (a) $\mathcal{S}_j = \{\Xi_{j1}, \ldots, \Xi_{j,l(j)}\}$ *denotes the set of* processing stages *involved in the definition of process type* Π_j*, and*

   (b) $\mathcal{G}_j$ *is an* acyclic digraph *with its node set,* V_j*, being bijectively related to the set* $\mathcal{S}_j$*. Let* $V_j^{\nearrow}$ *(resp.,* $V_j^{\searrow}$*) denote the set of* source *(resp.,* sink*) nodes of* $\mathcal{G}_j$*. Then, any* path *from some node* $v_s \in V_j^{\nearrow}$ *to some node* $v_f \in V_j^{\searrow}$ *defines a* process plan *for process type* Π_j*.*

4. $\mathcal{A} : \bigcup_{j=1}^{n} \mathcal{S}_j \rightarrow \prod_{i=1}^{m} \{0, \ldots, C_i\}$ *is the* resource allocation function *associating every processing stage* Ξ_{jk} *with a* resource allocation request $\mathcal{A}(j,k) \equiv A_{jk}$*. More specifically, each* A_{jk} *is an* m*-dimensional vector, with its* i*-th component indicating the number of resource units of resource type* R_i *necessary to support the execution of stage* Ξ_{jk}*. Obviously, in a well-defined RAS,* $A_{jk}(i) \leq C_i, \ \forall j, k, i$*. Furthermore, the resource set* A_{jk}*, required for the execution of a particular processing stage* Ξ_{jk}*, is allocated exclusively and non-preemptively to each process instance, and it is released by it only upon the allocation of the resources required for the execution of the subsequent stage.*

5. $\mathcal{T} : \bigcup_{j=1}^{n} \mathcal{S}_j \rightarrow \mathcal{D}$ *is the* timing function*, corresponding to each processing stage* Ξ_{jk} *a distribution* D_{jk} *that characterizes the statistics of the processing time* t_{jk}*, experienced during the execution of stage* Ξ_{jk}*.*

[2] Also, in this work, Z_0^+ will denote the set of nonnegative integers, Z will denote the set of all integers, and $\Re$ will denote the set of reals.

Finally, $|\Phi| \equiv |\mathcal{R}| + |\bigcup_{j=1}^{n} \mathcal{S}_j| + \sum_{i=1}^{m} C_i$ *will be referred to as the* size *of* Φ. $\diamondsuit$

For reasons that will become obvious in the subsequent development, we augment each of the aforementioned graphs $\mathcal{G}_j$, with an additional node v_{j0}, and a new set of edges $\{(v_{j0}, v_{jk}) : \ v_{jk} \in V_j^{\nearrow}\} \cup \{(v_{jk}, v_{j0}) : \ v_{jk} \in V_j^{\searrow}\}$. The resulting graph will be denoted by $\bar{\mathcal{G}}_j$. Notice that $\bar{\mathcal{G}}_j$ is strongly connected, with each cycle in it containing node v_{j0}. Node v_{j0} models the RAS environment, i.e., that state occupied by the process before entering or upon leaving the system.

A logical characterization of the RAS behavior A formal logical characterization of the behavior generated by the D/C-RAS of Definition 1, can be provided by means of the modelling framework of *Finite State Automaton (FSA)* [8]. In particular, the following definition of the D/C-RAS *state* facilitates the logical analysis of its behavior.

Definition 2 *Consider a D/C-RAS* $\Phi = < \mathcal{R}, C, \mathcal{P}, \mathcal{A}, \mathcal{T} >$. *For the purposes of logical analysis, its* state $s(t)$ *at time* t *is defined as a vector of dimensionality* $D = \sum_{j=1}^{n} l(j)$ *– i.e., equal to the total number of distinct processing stages in the system – and with components* $s(q;t),\ q = 1, \ldots, D$, *being in one-to-one correspondence with the RAS processing stages,* $\Xi_{jk},\ j = 1, \ldots, n,\ k = 1, \ldots, l(j)$. *Furthermore, component* $s(q(j,k);t)$, *corresponding to processing stage* Ξ_{jk}, *indicates the number of process instances executing stage* Ξ_{jk} *at time* t. $\diamondsuit$

A natural way to define the correspondence between the state components and the RAS processing stages is by setting $q(j,k) = k + \sum_{r=1}^{j-1} l(r)$; this will be the mapping assumed in the following. Also, to simplify the notation, in the following discussion we omit the dependence of state s on time t. The information contained in the RAS state is sufficient for the determination of the distribution of the resource units to the various processing stages, as well as of the *slack* (or *idle*) resource capacity in the system. In particular, we define the *slack* capacity, $\delta_i(s)$, of resource R_i at state s, by

$$\delta_i(s) \equiv C_i - \sum_{q=1}^{D} s(q(j,k)) \cdot A_{jk}(i) \tag{1}$$

Then, the set S of *feasible* resource allocation states for the considered RAS is defined by

$$S \equiv \{s \in (Z_0^+)^D : \delta_i(s) \geq 0,\ \forall i = 1, \ldots, m\} \tag{2}$$

The finiteness of the resource capacities implies that $card(S) \equiv |S| < \infty$. However, in general, $|S|$ will be a *super-polynomial* function of the RAS size.[3] The set of *events*, E, that can change the system state, comprises: (i) the events $e^l_{jk},\ j = 1, \ldots, n,\ k \in \{1, \ldots, l(j) :\ \Xi_{jk} \in V_j^{\nearrow}\}$, corresponding to the *loading* of a new instance of process type Π_j into the system, that is to follow a process plan starting with stage Ξ_{jk}, (ii) the events $e^a_{jkh},\ j = 1, \ldots, n,\ k, h = 1, \ldots, l(j),\ k \neq h$, corresponding to the *advancement* of a process instance executing stage Ξ_{jk} to a successor stage Ξ_{jh}, and (iii) the events $e^u_{jk},\ j = 1, \ldots, n,\ k \in \{1, \ldots, l(j) :\ \Xi_{jk} \in V_j^{\searrow}\}$, corresponding to the *unloading* of a finished process instance of type Π_j, whose last processing stage was stage $\Xi_{jk} \in V_j^{\searrow}$. Without loss of generality, it is assumed that, during a single state transition, only one of these events can take place. The resulting transition, however, is *feasible* only if the additionally requested set of resources can be obtained from the system slack capacity; i.e., for each state s, the set of *feasible* events, $\Gamma(s)$, contains only those events that abide to the resource allocation dynamics stated in item 4 of Definition 1. Based on the above, the *state transition function* f and the *feasible event function* Γ for the RAS-modelling FSA, are formally defined as follows:

$\forall s \in S, \forall e \in E, \quad f(s, e) \equiv$

$$\begin{cases} s + \mathbf{1}_{q(j,k)} & \text{if } e \equiv e^l_{jk} \text{ and } s + \mathbf{1}_{q(j,k)} \in S \\ s - \mathbf{1}_{q(j,k)} + \mathbf{1}_{q(j,h)} & \text{if } e \equiv e^a_{jkh} \text{ and} \\ & \qquad s - \mathbf{1}_{q(j,k)} + \mathbf{1}_{q(j,h)} \in S \\ s - \mathbf{1}_{q(j,k)} & \text{if } e \equiv e^u_{jk} \text{ and } s - \mathbf{1}_{q(j,k)} \in S \\ \text{undefined} & \text{otherwise} \end{cases} \tag{3}$$

$\forall s,$

$$\Gamma(s) \equiv \{e \in E : f(s, e)!\} \tag{4}$$

In Equation 3, $\mathbf{1}_{q(j,k)}$ denotes the D-dimensional *unit* vector, with its unit element in the component corresponding to stage Ξ_{jk}. A natural definition for the *initial* state s_0 is

$$s_0 \equiv \mathbf{0} \tag{5}$$

i.e., the state in which the system is idle and empty of any process instances. Since, in the following, our main logical concern is the establishment of non-blocking behavior, we define S_m, the set of *marked* states of the RAS-modelling automaton, as

$$S_m \equiv \{s_0\} \tag{6}$$

[3] in the particular case where each process stage requires a single unit from a single resource type, $|S| = \prod_{i=1}^{m} \frac{(C_i + |\mathcal{S}(R_i)|)!}{C_i! |\mathcal{S}(R_i)|!}$, where C_i denotes the capacity of resource R_i and $\mathcal{S}(R_i)$ denotes the set of processing stages requesting resource R_i for their execution.

Hence, the *marked language* $\mathcal{L}_m$ of this automaton corresponds to *"complete (processing) runs"*.

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

Example: As a concrete example for the above characterization of the D/C-RAS structure and its logical behavior, consider the small system of Figure 1. This RAS consists of three resources R_1, R_2 and R_3, each possessing unit capacity, i.e., $C_1 = C_2 = C_3 = 1$. In its current configuration, the system supports a single process type, Π_1, with the sequential logic characterized by graph $\mathcal{G}_1$; specifically, there are two possible process plans for this process type: $\Xi_{11} \rightarrow \Xi_{13}$ and $\Xi_{12} \rightarrow \Xi_{13}$. Figure 1 indicates also the resource allocation requests A_{jk} associated with each processing stage Ξ_{jk}: stage Ξ_{11} requires one unit of resource R_1 to support its execution, stage Ξ_{12} requires one unit of resource R_2, and stage Ξ_{13} requires one unit of resource R_3. Figure 2 depicts the STD structure for the reachable subspace for this RAS, while the detailed characterization of the RAS states corresponding to the various STD nodes is provided in Table 1.

Liveness and Liveness-Enforcing Supervision for D/C-RAS A major concern in the logical control of RAS is the establishment of *live* – or *deadlock-free* or *non-blocking* – behavior. *Deadlock* is technically defined as a RAS state containing a set of process instances with each process in this set requesting for its advancement the allocation of resources currently held by some of its other processes [15]. The development of deadlock results from the fact that (i) the RAS process routes are structured in an arbitrary manner that can give rise to cyclical patterns of resource requests among the various executing processes, while (ii) processes will hold upon their allocated resources in a non-preemptive manner. The occurrence of a deadlock stalls the progress of all the processes involved in it, and necessitates the application of appropriate exception handling procedures for its resolution; these procedures are disruptive and (frequently) costly for the operation of the underlying application.

In the FSA-based modelling of the RAS operation, deadlocks are represented by the formation of *strongly connected components* in the system reachable space, S_r, which, however, are not *co-accessible*, i.e., the empty state, s_0, is not reachable from them through any sequence of feasible tran-

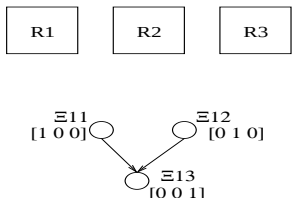

Figure 1: An example D/C-RAS

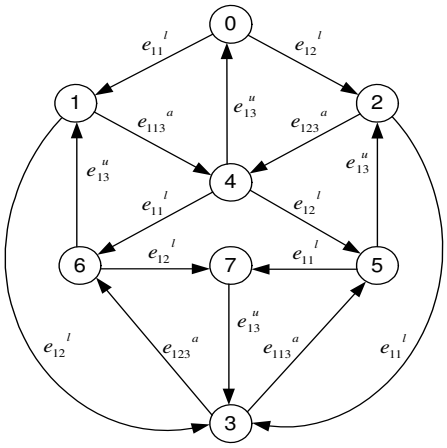

Figure 2: The State Transition Diagram (STD) for the D/C-RAS of Figure 1

Table 1: State information for the STD of Figure 1

s_k	$s(q(1,1))$	$s(q(1,2))$	$s(q(1,3))$
0	0	0	0
1	1	0	0
2	0	1	0
3	1	1	0
4	0	0	1
5	0	1	1
6	1	0	1
7	1	1	1

sitions. Hence, a *correct Liveness-Enforcing Supervisor (LES)*, Δ, tries to restrict the system operation to a *strongly connected component of S_r which contains the empty state s_0*. The RAS subspace that is reachable from the initial state, s_0, under supervision by some LES Δ, will be denoted by $S_r(\Delta)$. A correct LES, Δ^*, for a given D/C-RAS configuration, Φ, is characterized as *optimal*, if the corresponding admissible subspace $S_r(\Delta^*)$ is the *maximal* strongly connected component of S_r that contains the empty state s_0. The set of states admitted by the optimal LES, Δ^*, will be characterized as (the set of) *reachable safe* states, and it will be denoted by S_{rs}. The complement of S_{rs} with respect to S_r will be denoted by S_{ru}, and it constitutes the system *reachable unsafe* region; formally, $S_{ru} = S_r \backslash S_{rs}$.

Clearly, in the D/C-RAS operational context, the optimal LES, Δ^*, is well-defined, and furthermore, it is effectively computable through an *one-step lookahead* scheme that admits a tentative resource allocation if and only if (*iff*) the resulting state is safe. However, it has been shown that the problem of state safety is NP-complete for the class of D/C-RAS [1, 9]. In the light of this result, the research community has sought the development of sub-optimal LES that are implementable in polynomial complexity with respect to the underlying RAS size, and yet, efficient, i.e., they manage to admit a large part of S_{rs}. This idea has been formalized by the concept of *Polynomial Kernel (PK-)* LES's, in the relevant literature [16]. From an implementational standpoint, the design of PK-LES requires the identification of a property $\mathcal{H}(s)$, $s \in S$, such that: (i) $\mathcal{H}(s_0) = \texttt{TRUE}$, (ii) $\mathcal{H}(s) \Rightarrow \texttt{safe}(s), \forall s \in S_r$, and (iii) the complexity of testing $\mathcal{H}()$ on the RAS states is *polynomial with respect to the RAS size*. Then, by allowing only transitions to states satisfying $\mathcal{H}$, through one-step look-ahead, it can be ensured that the visited states will be safe. An additional requirement, however, is that (iv) the resulting LES is *correct*, i.e., the LES-admissible subspace is *strongly connected*, since, otherwise, the system could reach *LES-induced* deadlocks, i.e., RAS states where any further progress is stalled by the LES itself. Two conditions $\mathcal{H}()$ leading to correct PK-LES for D/C-RAS are provided in [6] and [11].

Example: Notice that the STD of Figure 2 is itself a strongly connected graph. Hence, the uncontrolled – or *feasible* – behavior of the D/C-RAS depicted in Figure 1, is already deadlock-free. This effect results from the fact that the process routing logic expressed by graph $\mathcal{G}_1$ will not give rise to any *circular-waiting* patterns in the contest of the various process instances for the system resources.

3 Performance modelling and control of logically controlled RAS

The assumption of exponentially distributed processing times In order to study the performance of the D/C-RAS, we need to augment the FSA model developed in Section 2, with the timing aspects of the system operation. We remind the reader that the main timing information provided by the D/C-RAS object of Definition 1, concerns the distributions D_{jk}, characterizing the statistics of the processing times experienced during the execution of stages Ξ_{jk}. For reasons that will become obvious in the subsequent analysis, we assume that each D_{jk} is an *exponential* distribution, with parameter λ_{jk}. Although this assumption can be considered as a restriction of the model and its applicability, the derived results can still be applied in an approximate sense to D/C-RAS involving more general distributions D_{jk}, by substituting the relevant non-exponential stages by some appropriately structured acyclic process sub-nets modelling a *phase-type* distribution that approximates D_{jk} to some satisfactory level; we refer the reader to [10] for the details of this approximation.

The refined RAS state space Under the assumption of exponentially distributed processing times, the temporal aspects of the RAS behavior can be captured through the following refinement of the basic FSA model introduced in Section 2:

1. Each state component $s(q(j,k))$ is expanded to a 3-dimensional vector

$$[s^p(q(j,k)), s^f(q(j,k)), s^b(q(j,k)]$$

   Components $s^p(q(j,k))$ and $s^f(q(j,k))$ indicate the numbers of process instances of stage Ξ_{jk} that are, respectively, in *processing* or *finished*. Component $s^b(q(j,k))$ is a *binary*-valued component, with the value of 1 indicating that the advancement of the finished processes in stage Ξ_{jk} is currently *blocked*. Furthermore, an additional component $s^b(q(j,0))$ is added for each process type Π_j. This component is also *binary*-valued, and its value of 1 indicates that the loading of any further instances of process type Π_j is currently *blocked*.

2. At any point in time, there should be at least one process instance under processing in the system, i.e., the system operation should be *globally non-idling*. The feasibility of this requirement is guaranteed by (i) the correctness of the applied LES, and (ii) the further assumption of an *infinite* number of instances from each process type waiting for

loading in the considered D/C-RAS; this last assumption is consistent with the posed objective of throughput maximization.

3. At any RAS state, the completion of an active process instance immediately resets all the state components of $s^b()$ type to 0, i.e., it unblocks all the process loading and advancing events.

4. A state containing unblocked loading and advancing events is followed by a sequence of *immediate* transitions each of which either (i) advances an unblocked finished process instance to its next stage, or (ii) loads a new process instance of some unblocked process type, or (iii) sets an $s^b()$ state component to its blocked state. Process instances j_j having completed a *"sink"* processing stage Ξ_{jk} in their corresponding graph $\mathcal{G}_j$ cannot be blocked, but they must be unloaded from the system.[4] The advancement of any other process instance to its next state, or the loading of a new process instance, must be admissible by the applied LES.

The underlying Semi-Markov structure It is clear from the description of the previous paragraph that states containing unblocked finished process instances, or unblocked loading events, present zero sojourn times, and therefore, they are characterized as *vanishing*. The remaining states contain only process instances that are in processing or blocked. These states will be characterized as *tangible*, since they have a finite sojourn time, that is determined by the *"exponential race"* of all the process instances that are in processing, for completion of their running processing stage. More specifically, the processing times experienced at tangible states are exponentially distributed with a rate equal to $\sum_{(j,k)}[\lambda_{jk} \cdot s^p(q(j,k))]$. Let $S_{rv}(\Delta)$ and $S_{rt}(\Delta)$ denote respectively the sets of vanishing and tangible states that are reachable under the RAS supervision by LES Δ. Then, the considered RAS performance control problem reduces to *the determination of a probability distribution for each vanishing state, that will regulate the execution of the corresponding LES-admissible immediate events, so that the long-run system throughput is maximized*. The resulting problem is a *semi-Markov decision process (semi-MDP)* [12].

Example: Figure 3 and Table 2 encode the semi-MDP for the example D/C-RAS of Figure 1, under the further assumption that the processing

[4]This requirement implies that state components $s^f(q(j,k))$ and $s^b(q(j,k))$ for "sink" processing stages Ξ_{jk} will always be equal to zero, and therefore, they can be dropped from the state vector.

Table 2: State information for the semi-Markov process characterizing the timed dynamics of the D/C-RAS of Figure 1

s_k	$s^b(q(1,0))$	$s^p(q(1,1))\ s^J(q(1,1))\ s^b(q(1,1))$	$s^p(q(1,2))\ s^J(q(1,2))\ s^b(q(1,2))$	$s^p(q(1,3))$
0	0	0 0 0	0 0 0	0
1	0	1 0 0	0 0 0	0
2	0	0 0 0	1 0 0	0
3	0	1 0 0	1 0 0	0
4	1	1 0 0	0 0 0	0
5	1	0 0 0	1 0 0	0
6	1	1 0 0	1 0 0	0
7	0	0 1 0	0 0 0	0
8	0	0 0 0	0 1 0	0
9	0	0 1 0	1 0 0	0
10	0	1 0 0	0 1 0	0
11	0	0 0 0	0 0 0	1
12	0	0 0 0	1 0 0	1
13	1	0 1 1	1 0 0	0
14	0	1 0 0	0 0 0	1
15	1	1 0 0	0 1 1	0
16	1	0 0 0	0 0 0	1
17	0	1 0 0	1 0 0	1
18	1	0 0 0	1 0 0	1
19	0	0 1 0	0 1 0	0
20	1	1 0 0	0 0 0	1
21	1	1 0 0	1 0 0	1
22	0	0 0 0	0 1 0	1
23	0	0 1 0	0 0 0	1
24	0	0 1 0	1 0 0	1
25	0	1 0 0	0 1 0	1
26	1	0 0 0	0 1 1	1
27	1	0 1 1	0 0 0	1
28	1	0 1 1	1 0 0	1
29	1	1 0 0	0 1 1	1
30	0	0 1 0	0 1 0	1
31	1	0 1 1	0 1 1	1

time corresponding to processing stage $\Xi_{1k},\ k = 1, 2, 3$, is exponentially distributed with rate λ_k. Double-lined nodes in Figure 3 indicate the process *tangible* states, while the expressions on the edges emanating from these nodes characterize the corresponding *branching* probabilities of the semi-Markov process. These probabilities are determined by the fact that the process transition out of each tangible state is regulated by the "exponential race" of all the active processes in that state to complete their currently executed processing stage. For the remaining vanishing states, the corresponding branching probabilities must be externally provided by the performance control policy, in a way that optimizes some pre-specified performance objective (in this study, the long-run system throughput).

The induced CT-MDP The above semi-MDP formulation of the considered RAS performance control problem can be further reduced to a *Continuous-Time MDP (CT-MDP)* problem, by focusing on the transitions of the underlying stochastic process among the subclass of vanishing states that are entered from some tangible state, upon the finishing of some RAS process instance. Let us denote this particular subclass of vanishing states by $S^0_{rv}(\Delta)$; i.e., $S^0_{rv}(\Delta)$ defines the *state space* of the induced process. Ac-

cording to item 3 in the description of the refined state space $S_r(\Delta)$, any state $s \in S^0_{rv}(\Delta)$ enables all the Δ-admissible process loading and advancing events. The set of all the *control actions available at state* $s \in S^0_{rv}(\Delta)$ is defined by the set $\Sigma(s)$ of all the Δ-admissible immediate transition sequences σ that can be executed starting from state s; indeed, enabling of the process loading and advancing events in each transition sequence σ is completely at the jurisdiction of the RAS controller. On the other hand, the induced process states $s' \in S^0_{rv}(\Delta)$, that can result from s by taking an action $\sigma \in \Sigma(s)$, are determined by the transitions that are enabled in the tangible state s'' that is obtained in the original semi-MDP process from the execution of sequence σ upon state s; i.e., for any given control action $\sigma \in \Sigma(s)$, the resulting states s' correspond to the finishing of some process instance at stage Ξ_{jk} with $(s'')^p(q(j,k)) > 0$. The corresponding *branching probabilities* are determined by

$$p(s'; s, \sigma) = \frac{\lambda_{jk} \cdot (s'')^p(q(j,k))}{\sum_{(j',k')}[\lambda_{j'k'} \cdot (s'')^p(q(j',k'))]} \tag{7}$$

The *sojourn time* associated with the transition resulting from the selection of control action σ at state s is exponentially distributed with *rate*

$$\Lambda(s, \sigma) = \sum_{(j,k)} [\lambda_{jk} \cdot (s'')^p(q(j,k))] \tag{8}$$

and *mean value*

$$\tau(s, \sigma) = \frac{1}{\sum_{(j,k)}[\lambda_{jk} \cdot (s'')^p(q(j,k))]} \tag{9}$$

The above characterization of the branching probabilities and the state sojourn times indicates clearly that the decision process induced by the transitions of the original semi-MDP process over its $S^0_{rv}(\Delta)$ sub-space, is CT-MDP.

Example: The induced CT-Markov process for the example D/C-RAS of Figure 1 is characterized by Tables 3 and 4, and Figure 4. While the depicted tangible markings are not, in a strict technical sense, part of the considered CT-Markov process, we have opted to include them in the graph of Figure 4 in order to visualize the connection between the state space of this process and the state space of the semi-Markov process depicted in Figure 3. Hence, for instance, there are three possible transitions out of the initial state s_0, in the considered CT-Markov process. The first transition corresponds to the loading of a new process instance and the initiation of

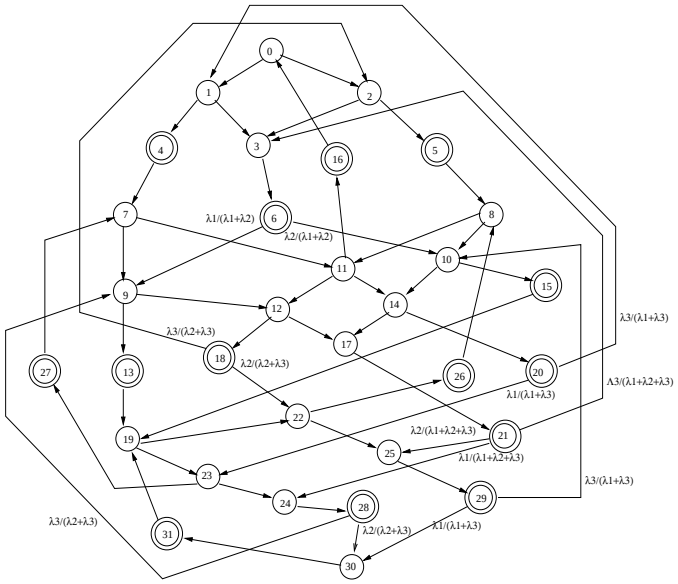

Figure 3: The embedded Markov Chain of the semi-Markov process characterizing the timed dynamics of the D/C-RAS of Figure 1

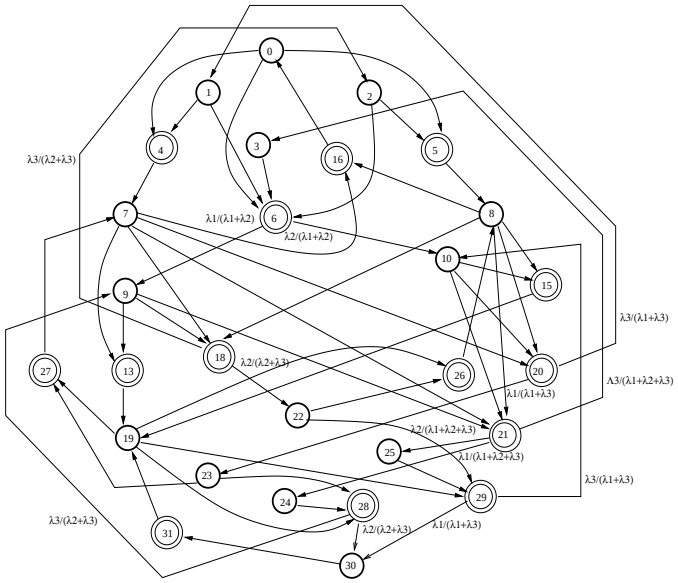

Figure 4: The induced CT-Markov process for the D/C-RAS of Figure 1

Table 3: Vanishing state information for the CT-Markov process of the D/C-RAS of Figure 1

s_k	$s^b(q(1,0))$	$< s^p s^f s^b > (q(1,1))$	$< s^p s^f s^b > (q(1,2))$	$s^p(q(1,3))$	Res. Tan. States
0	0	0 0 0	0 0 0	0	4, 5, 6
1	0	1 0 0	0 0 0	0	4, 6
2	0	0 0 0	1 0 0	0	5, 6
3	0	1 0 0	1 0 0	0	6
7	0	0 1 0	0 0 0	0	13, 18, 21, 16, 20
8	0	0 0 0	0 1 0	0	16, 18, 21, 15, 20
9	0	0 1 0	1 0 0	0	13, 18, 21
10	0	1 0 0	0 1 0	0	15, 20, 21
19	0	0 1 0	0 1 0	0	27, 28, 26, 29
22	0	0 0 0	0 1 0	1	26, 29
23	0	0 1 0	0 0 0	1	27, 28
24	0	0 1 0	1 0 0	1	28
25	0	1 0 0	0 1 0	1	29
30	0	0 1 0	0 1 0	1	31

Table 4: Tangible state information for the CT-Markov process of the D/C-RAS of Figure 1

s_k	$s^b(q(1,0))$	$< s^p s^f s^b > (q(1,1))$	$< s^p s^f s^b > (q(1,2))$	$s^p(q(1,3))$	Res. Van. States
4	1	1 0 0	0 0 0	0	7
5	1	0 0 0	1 0 0	0	8
6	1	1 0 0	1 0 0	0	9, 10
13	1	0 1 1	1 0 0	0	19
15	1	1 0 0	0 1 1	0	19
16	1	0 0 0	0 0 0	1	0
18	1	0 0 0	1 0 0	1	22, 2
20	1	1 0 0	0 0 0	1	23, 1
21	1	1 0 0	1 0 0	1	24, 25, 3
26	1	0 0 0	0 1 1	1	8
27	1	0 1 1	0 0 0	1	7
28	1	0 1 1	1 0 0	1	30, 9
29	1	1 0 0	0 1 1	1	30, 10
31	1	0 1 1	0 1 1	1	19

processing stage Ξ_{11}; the resulting state is s_7, while the expected sojourn time for this transition is $1/\lambda_1$. The second transition corresponds to the loading of a new process instance and the initiation of processing stage Ξ_{12}; the resulting state is s_8, while the expected sojourn time for this transition is $1/\lambda_2$. Finally, the third transition corresponds to the event sequence of loading two instances in the considered RAS, the first of them initiated to processing stage Ξ_{11} and the second initiated to processing stage Ξ_{12}. In this case, the resulting state could be either s_9 or s_{10}, depending on whether the process instance completing first its current stage is that executing stage Ξ_{11} or Ξ_{12}. The corresponding branching probabilities are $\lambda_1/(\lambda_1+\lambda_2)$ and $\lambda_2/(\lambda_1+\lambda_2)$, and the transition expected sojourn time is $1/(\lambda_1+\lambda_2)$. The rest of the graph can be interpreted in a similar fashion.

Characterizing the objective function of the CT-MDP To completely define this CT-MDP problem, it remains to specify the *(immediate) gain* $g(s,\sigma)$ associated with each state-option pair (s,σ), and the *objective* under consideration. The characterization of these two issues is determined

by our focus on the long-run system throughput as the performance criterion of interest. Hence, $g(s,\sigma)$ is defined as

$$g(s,\sigma) = \sum_{(j,k)} \frac{\lambda_{jk} \cdot (s'')^p(q(j,k))}{\sum_{(j',k')}[\lambda_{j'k'} \cdot (s'')^p(q(j',k'))]} \cdot I \quad \begin{array}{l}\{\Xi_{jk} \text{ is a} \\ \text{“sink” stage} \\ \text{of } \mathcal{G}_j\}\end{array} \tag{10}$$

i.e., it is equal to the probability that some process instance will be completed and unloaded during the transition resulting from the state-option pair (s,σ). In the light of this definition of the immediate gains, the long-run system throughput obtained by initializing the RAS in some reachable LES-admissible state s^0 and applying some *policy* Ψ for governing the selection of the options σ to be enabled at each state s, is characterized by the *average expected gain* $\bar{g}^{\Psi}(s^0)$, which is formally defined as follows:

$$\bar{g}^{\Psi}(s^0) = \liminf_{t\to\infty} \frac{1}{t} E^{\Psi}_{s^0}[\sum_{n=0}^{\infty} \int_0^t g(s_n,\sigma_n) \cdot \delta(t-t_n)] \tag{11}$$

In Equation 11, t_n denotes the time of the n-th transition in the CT-Markov chain obtained by setting the considered RAS at state s^0 in time $t_0 = 0$, and subsequently controlling it by policy Ψ. The expectation $E^{\Psi}_{s^0}[\,]$ is taken over all the corresponding sample paths of the aforementioned CT-Markov chain, and the function $\delta(t)$ is Kronecker's delta function. Our objective is to identify a policy Ψ^* such that

$$\bar{g}^{\Psi^*}(s) = \sup_{\Psi} \left\{ \bar{g}^{\Psi}(s), \quad \forall s \in S^0_{rv}(\Delta) \right\} \tag{12}$$

Example: To provide a more concrete demonstration of the gain structure implied by Equation 10, let us consider the gain to be employed in the CT-MDP problem of Figure 4, for the state-option pair that leads the underlying RAS from vanishing state s_7 to the tangible marking s_{20}. The associated scheduling decision corresponds to the event sequence of advancing a process instance that just completed the execution of processing stage Ξ_{11} to its next stage Ξ_{13}, followed by the immediate initiation of another processing instance at stage Ξ_{11}. The probability of experiencing a process unloading event during the current transition in the underlying CT-Markov chain, is equal to the probability that the process executing processing stage Ξ_{13}, in tangible state s_{20}, will complete first. Hence, the considered gain is equal to $\lambda_3/(\lambda_1+\lambda_3)$. On the other hand, all the transitions emanating from state s_0, considered in the example of the previous paragraph, do not involve any

unloading events, and therefore, the corresponding gains should be set equal to zero.

An LP formulation for the performance control problem The observation of some correct LES Δ during the construction of the state space of the aforementioned CT-MDP guarantees that there exists a deterministic policy that can take the RAS from any given state $s \in S^0_{rv}(\Delta)$ to any other state $s' \in S^0_{rv}(\Delta)$. As a result, the considered CT-MDP is classified as a *communicating* model and the optimal average expected gain $\bar{g}^{\Psi^*}(s)$ is constant for all initial states $s \in S^0_{rv}(\Delta)$ ([12], Section 8.3). In the following, this constant optimal gain will be denoted by $\bar{g}^*$. Its value and an optimal policy Ψ^* can be obtained by solving the following *linear programming (LP)* formulation in variables $x(s,\sigma),\ s \in S^0_{rv}(\Delta),\ \sigma \in \Sigma(s)$ (c.f. [12], Sections 8.8, 9.5, 11.4–5):

$$\max \sum_{s \in S^0_{rv}(\Delta)} \sum_{\sigma \in \Sigma(s)} g(s,\sigma) x(s,\sigma) \tag{13}$$

s.t.

$$\forall s \in S^0_{rv}(\Delta),$$
$$\sum_{\sigma \in \Sigma(s)} x(s,\sigma) - \sum_{s' \in S^0_{rv}(\Delta)} \sum_{\sigma \in \Sigma(s')} p(s;s',\sigma) x(s',\sigma) = 0 \tag{14}$$

$$\sum_{s \in S^0_{rv}(\Delta)} \sum_{\sigma \in \Sigma(s)} \tau(s,\sigma) x(s,\sigma) = 1 \tag{15}$$

$$x(s,\sigma) \geq 0, \quad \forall s \in S^0_{rv}(\Delta),\ \forall \sigma \in \Sigma(s) \tag{16}$$

The value of the optimal gain, $\bar{g}^*$, is equal to the optimal objective value for this formulation. The optimal values of variables $x(s,\sigma)$ – to denoted by $x^*(s,\sigma)$ – can be interpreted as the *rates* with which the underlying RAS finds itself in state s having selected option σ in that state, when operated under the optimal policy Ψ^*. Ψ^* itself is determined as follows: The optimal solution, $x^*(s,\sigma)$, of the above formulation, partitions the state space $S^0_{rv}(\Delta)$ into two classes, $S^{0r}_{rv}(\Delta)$ and $S^{0t}_{rv}(\Delta)$: Class $S^{0r}_{rv}(\Delta)$ contains the states $s \in S^0_{rv}(\Delta)$ with $\sum_{\sigma' \in \Sigma(s)} x^*(s,\sigma') > 0$, that constitute the *recurrent* states during the system operation under policy Ψ^*. Class $S^{0t}_{rv}(\Delta)$ contains the states $s \in S^0_{rv}(\Delta)$ with $\sum_{\sigma' \in \Sigma(s)} x^*(s,\sigma') = 0$, that are the system *transient* states. For the recurrent states $s \in S^{0r}_{rv}(\Delta)$, the optimal policy Ψ^* is defined by setting:

$$\psi^*(\sigma;s) = \frac{x^*(s,\sigma)\tau(s,\sigma)}{\sum_{\sigma' \in \Sigma(s)} x^*(s,\sigma')\tau(s,\sigma')}, \quad \forall \sigma \in \Sigma(s) \tag{17}$$

1. $\bar{S}^{0r}_{rv}(\Delta) := S^{0r}_{rv}(\Delta)$;
2. `While` $\mathcal{S}^0_{rv}(\Delta) \backslash \bar{S}^{0r}_{rv}(\Delta) \neq \emptyset$ `do`
   (a) Find a state $s \in \mathcal{S}^0_{rv}(\Delta) \backslash \bar{S}^{0r}_{rv}(\Delta)$ and an action $\sigma^* \in \Sigma(s)$ s.t. $\sum_{s' \in \bar{S}^{0r}_{rv}(\Delta)} p(s'; s, \sigma^*) > 0$;
   (b) $\forall \sigma \in \Sigma(s)$, `if` $\sigma = \sigma^*$ `then` $\psi^*(\sigma; s) := 1$ `else` $\psi^*(\sigma; s) := 0$;
   (c) $\bar{S}^{0r}_{rv}(\Delta) := \bar{S}^{0r}_{rv}(\Delta) \cup \{s\}$

   `endwhile`
3. `RETURN` $\psi^*(\cdot; s)$ for each $s \in S^{0t}_{rv}(\Delta)$.

Figure 5: Defining the optimal policy Ψ^* on transient states $s \in S^{0t}_{rv}(\Delta)$

For the remaining transient states, the optimal policy is defined by any action selection scheme that establishes a path from each transient state to the class of recurrent states; a detailed algorithm for identifying such a set of paths covering all transient states, is provided in Figure 5. Finally, it can be shown that, for any recurrent state $s \in S^{0r}_{rv}(\Delta)$, any basic feasible solution of the LP formulation of Equations 13-16 will have $x^*(x, u) > 0$ for only one control action $\sigma \in \Sigma(s)$ [12]; i.e., the optimal policy Ψ^*, returned by this approach, will be *deterministic*.

Incorporating additional throughput-ratio constraints In case of RAS with more than one process types, the definition of the process gains and the problem objective function provided by Equations 10–12, implies that the ultimate objective is the *unconditional* maximization of the cumulative throughput across all process types. In most practical situations, the throughput with respect to the various process types will be required to observe some *ratio constraints*, i.e.,

$$TH_j = \rho_j TH_1, \quad \forall j \geq 2 \tag{18}$$

The resulting problem is a *constrained* CT-MDP problem ([12], Section 8.9). Equation 18 suggests that, in this case, it is pertinent to seek the maximization of TH_1, while observing the ratio constraints. Since we need to differentiate the throughput attained with respect to each process type, the constrained version of the problem will employ a set of gains $g_j(s, \sigma)$, one

gain function for each process type Π_j, with

$$g_j(s,\sigma) = \sum_k \frac{\lambda_{jk} \cdot (s'')^p(q(j,k))}{\sum_{(j',k')}[\lambda_{j'k'} \cdot (s'')^p(q(j',k'))]} \cdot I \begin{Bmatrix} \{\Xi_{jk} \text{ is a} \\ \text{“sink” stage} \\ \text{of } \mathcal{G}_j\} \end{Bmatrix} \tag{19}$$

The long-term throughput $\bar{g}^*$ and an optimal policy Ψ^* for the constrained version of the RAS performance control problem can be obtained by an LP formulation similar to that of Equations 13–16. The new objective function for this formulation is

$$\max \sum_{s \in S^0_{rv}(\Delta)} \sum_{\sigma \in \Sigma(s)} g_1(s,\sigma) x(s,\sigma) \tag{20}$$

and the constraint set is augmented with the following constraint that expresses the throughput ratio requirement of Equation 18:

$$\sum_{s \in S^0_{rv}(\Delta)} \sum_{\sigma \in \Sigma(s)} g_j(s,\sigma) x(s,\sigma) - \rho_j \sum_{s \in S^0_{rv}(\Delta)} \sum_{\sigma \in \Sigma(s)} g_1(s,\sigma) x(s,\sigma) = 0, \quad \forall j \geq 2 \tag{21}$$

The optimal throughput $\bar{g}^*$ and a stationary optimal policy Ψ^* are determined as in the unconstrained case; however, there might *not* exist a deterministic optimal policy for this case.[5]

4 The thinning problem

Problem statement In the context of the RAS modelling framework presented in Sections 2 and 3, the thinning problem can be abstracted as the selection of a subset of edges, $\mathcal{E}_j^S$, from each process graph, $\mathcal{G}_j$, of a D/C-RAS, Φ, such that (i) $|\mathcal{E}_j^S| = \mu_j$, $\forall j$, (ii) and some performance index of the RAS Φ', that results by restricting each process type Π_j to its process plans encompassed by $\mathcal{E}_j^S$, is optimized. Parameters μ_j are externally specified, and they quantify the size constraints imposed on the selected process subnets. Next we provide a detailed, analytical characterization of the thinning problem for the case that the observed performance objective is the maximization of the RAS throughput, which was also the performance objective considered in the development of Section 3.

[5] In fact, the policy randomization is the mechanism enabling the satisfaction of the throughput ratio constraint expressed by Equation 18.

1. Construct the graph $\bar{\mathcal{G}}_j$, by adding to graph $\mathcal{G}_j$ a node v_{j0} together with the set of edges $\{(v_{j0}, v_{jk}) :\ v_{jk} \in V_j^{\nearrow}\} \cup \{(v_{jk}, v_{j0}) :\ v_{jk} \in V_j^{\searrow}\}$.

2. Construct the subgraph $\bar{\mathcal{G}}_j(\mathcal{E}_j^S)$ of $\bar{\mathcal{G}}_j$, with node set $\{v_{jk},\ k = 0, 1, \ldots, l(j)\}$, and edge set $\mathcal{E}_j^S \cup \{(v_{j0}, v_{jk}) :\ v_{jk} \in V_j^{\nearrow}\} \cup \{(v_{jk}, v_{j0}) : v_{jk} \in V_j^{\searrow}\}$.

3. Trim the graph $\bar{\mathcal{G}}_j(\mathcal{E}_j^S)$ with respect to node v_{j0}, to obtain the graph $\bar{\mathcal{G}}_j^{Tr}(\mathcal{E}_j^S)$.

4. Delete the node v_{j0} and its incident edges from $\bar{\mathcal{G}}_j^{Tr}(\mathcal{E}_j^S)$, to obtain the graph $\mathcal{G}_j(\mathcal{E}_j^S)$.

5. `RETURN` $\mathcal{G}_j(\mathcal{E}_j^S)$.

Figure 6: Characterizing all the process plans encompassed in an edge selection $\mathcal{E}_j^S \subseteq \mathcal{E}_j$ of the process graph $\mathcal{G}_j$ corresponding to a process type Π_j of a D/C-RAS Φ

Consider the graph $\mathcal{G}_j$ characterizing the execution logic of some process type Π_j of a D/C-RAS Φ, and a selection of edges from this graph, $\mathcal{E}_j^S \subseteq \mathcal{E}_j$. The subgraph $\mathcal{G}_j(\mathcal{E}_j^S)$ that encodes all the process plans for process type Π_j involving only transitions corresponding to edges in $\mathcal{E}_j^S$, can be computed by the algorithm of Figure 6. Then, given a D/C-RAS Φ and sets $\mathcal{E}_j^S \subseteq \mathcal{E}_j$ for each process type Π_j, let $\Phi(\mathcal{E}_1^S, \ldots, \mathcal{E}_n^S)$ denote the D/C-RAS obtained by substituting each process graph $\mathcal{G}_j$ by $\mathcal{G}_j(\mathcal{E}_j^S)$, in the definition of Φ. Of particular interest for the pursued formulation are all the D/C-RAS $\Phi(\mathcal{E}_1^S, \ldots, \mathcal{E}_n^S)$ for which (i) the edge sets $\mathcal{E}_j^S$ have $|\mathcal{E}_j^S| = \mu_j$, where $\mu_j \leq |\mathcal{E}_j|$ is externally defined, and (ii) $\mathcal{G}_j(\mathcal{E}_j^S) \neq NULL,\ \forall j$. In the following, the distinct edge selections defining these D/C-RAS will be denoted by $\{\mathcal{S}_1, \ldots, \mathcal{S}_N\}$, and the corresponding D/C-RAS by $\Phi(\mathcal{S}_u),\ u = 1, \ldots, N$. Clearly, N is finite; in particular, $N \leq \prod_{j=1}^{n} \begin{pmatrix} |\mathcal{E}_j| \\ \mu_j \end{pmatrix}$.

In the context of this formalism, the thinning problem can be posed as follows: Given a D/C-RAS Φ, a correct LES Δ, target throughput ratios ρ_j for $j = 2, \ldots, |\mathcal{P}|$,[6] and parameters $\mu_j \leq |\mathcal{E}_j|,\ j = 1, \ldots, |\mathcal{P}|$, find an edge

[6] In the case of D/C-RAS with a single process type, this data set is vacuous, and the unconstrained version of the throughput maximization problem must be employed.

selection $\mathcal{S}^*$ such that

$$\mathcal{S}^* = \arg\max_{\mathcal{S}} \left\{ TH_1^*(\Phi(\mathcal{S}); \Delta, \rho_2, \ldots, \rho_{|\mathcal{P}|}) \right\} \tag{22}$$

s.t.

$$|\mathcal{E}_j^S| = \mu_j \ \wedge \ \mathcal{G}_j(\mathcal{E}_j^S) \neq NULL, \ \forall \mathcal{E}_j^S \in \mathcal{S} \tag{23}$$

The maximal throughput, $TH_1^*(\Phi(\mathcal{S}); \Delta, \rho_2, \ldots, \rho_{|\mathcal{P}|})$, for D/C-RAS $\Phi($ $\mathcal{S})$, can be computed through the techniques presented in Section 3. More specifically, under the assumption that all the timing distributions, D_{jk}, of the considered RAS, Φ, are exponential, the maximal throughput $TH_1^*(\Phi(\mathcal{S})$; $\Delta, \rho_2, \ldots \rho_{|\mathcal{P}|})$ can be computed through the formulation of Equations 20, 14–16 and 21, by forcing to zero every variable $x^*(s, \sigma)$ that corresponds to a transition leading to a state s' which does not belong to $S_r(\Phi(\mathcal{S}); \Delta)$, the reachable Δ-admissible subspace for the D/C-RAS $\Phi(\mathcal{S})$.

Next we establish a dominance relationship that will allow the further restriction of the set of selections to be considered in the solution of the thinning problem defined by Equations 22 and 23. Given two selections $\mathcal{S}_1$ and $\mathcal{S}_2$ satisfying Constraint 23, let $\mathcal{S}_1 \preceq \mathcal{S}_2$ denote that graph $\mathcal{G}(\mathcal{E}_j^{S_1})$ is a subgraph of $\mathcal{G}(\mathcal{E}_j^{S_2})$, for all j. The relationship '$\mathcal{S}_1 \preceq \mathcal{S}_2$' defines a *partial order* on the set of selections satisfying Constraint 23, and we shall say that selection $\mathcal{S}_1$ is *dominated by* – or it is *smaller than* – selection $\mathcal{S}_2$. Clearly, $\mathcal{S}_1 \preceq \mathcal{S}_2$ implies that $S_r(\Phi(\mathcal{S}_1); \Delta) \subseteq S_r(\Phi(\mathcal{S}_2); \Delta) \subseteq S_r(\Phi; \Delta)$. Suppose that the evaluation of the optimal throughput, $TH_1^*(\Phi(\mathcal{S}_i); \Delta, \rho_2, \ldots, \rho_{|\mathcal{P}|})$, for RAS $\Phi(\mathcal{S}_i), \ i = 1, 2$, is performed through the formulation of Equations 20, 14–16 and 21. Then, the above set inclusions imply that, (i) every feasible solution for the LP formulation estimating $TH_1^*(\Phi(\mathcal{S}_1); \Delta, \rho_2, \ldots, \rho_{|\mathcal{P}|})$ is also a feasible solution for the LP formulation estimating $TH_1^*(\Phi(\mathcal{S}_2); \Delta, \rho_2, \ldots, \rho_{|\mathcal{P}|})$, while (ii) the objective function of the former formulation is subsumed in the objective function of the latter. The last two observations further imply that $TH_1^*(\Phi(\mathcal{S}_1); \Delta, \rho_2, \ldots, \rho_{|\mathcal{P}|}) \leq TH_1^*(\Phi(\mathcal{S}_2); \Delta, \rho_2, \ldots, \rho_{|\mathcal{P}|})$. Hence, it can be concluded that, among the selections that satisfy Constraint 23, only those that are *maximal* with respect to relationship ' $\preceq$' need to be considered in the solution of Equation 22.

Example: Suppose that (i) the processing times, t_{1k}, for the three stages $\Xi_{1k}, \ k = 1, 2, 3$, of the D/C-RAS of Figure 1 are exponentially distributed with corresponding rates λ_k, and (ii) the "thinning" requirement of $\mu_1 = 1$ is imposed on its operation. It is obvious from the structure of the graph $\mathcal{G}_1$, depicted in Figure 1, that this thinning requirement essentially implies the enactment of *only* one of the two possible process plans during the system operation. Technically, these two selections are defined by the edge sets

$\mathcal{E}_1^{\mathcal{S}_1} = \{(\Xi_{11}, \Xi_{13})\}$ and $\mathcal{E}_1^{\mathcal{S}_2} = \{(\Xi_{12}, \Xi_{13})\}$, both of which are maximal with respect to relation '$\preceq$'. The subspaces $S_r(\Phi(\mathcal{S}_i); \Delta^*),\ i = 1, 2$, can be easily obtained from the information provided in the state space S_r of Figure 2 and in Table 1. For this simple case, it is easy to see that the optimal selection, $\mathcal{S}^*$, for the considered thinning problem, is given by[7]

$$\mathcal{S}^* = \begin{cases} \mathcal{S}_1, & \text{for } \lambda_1 \geq \lambda_2 \\ \mathcal{S}_2, & \text{for } \lambda_1 \leq \lambda_2 \end{cases} \tag{24}$$

A MIP formulation of the thinning problem for Markovian RAS In the case of RAS with exponentially distributed stage processing times, Equation 15 implies that, in the computation of $TH_1^*(\Phi(\mathcal{S}); \Delta, \rho_2, \ldots, \rho_{|\mathcal{P}|})$ for some given edge selection $\mathcal{S}$, the requirement of forcing to zero every variable $x(s, \sigma)$ that corresponds to a transition leading to a state $s' \notin S_r(\Phi(\mathcal{S}); \Delta)$, can be expressed by the addition of the following constraint in the LP formulation of Equations 20, 14–16 and 21:

$$\forall s \in S_{rv}^0(\Delta),\ \forall \sigma \in \Sigma(s), \quad x(s, \sigma) \leq \{\min_{(s,\sigma):\ s \in S_{rv}^0(\Delta),\ \sigma \in \Sigma(s)} \tau(s, \sigma)\}^{-1} I_{\{s' \in S_r(\Phi(\mathcal{S}); \Delta)\}} \tag{25}$$

The pricing logic underlying Equation 25, that facilitates the performance evaluation of the D/C-RAS, $\Phi(\mathcal{S})$, resulting from a single edge selection $\mathcal{S}$, can be subsequently extended to a *Mixed Integer Programming (MIP)* [17] formulation for the entire thinning problem, as follows: Consider an enumeration $\{\mathcal{S}_1, \ldots, \mathcal{S}_{N'}\}$ of all the '$\preceq$'-*maximal* edge selections satisfying Constraint 23, and associate a *binary* variable I_u with each selection $\mathcal{S}_u$, such that $I_u = 1$ indicates that the finally selected edge sets $\mathcal{E}_j^S$ are those in selection $\mathcal{S}_u$. Obviously,

$$\sum_u I_u = 1 \tag{26}$$

In addition, in the spirit of Equation 25, we must have:

$$\forall s \in S_{rv}^0(\Delta),\ \forall \sigma \in \Sigma(s), \quad x(s, \sigma) \leq \{\min_{(s,\sigma):\ s \in S_{rv}^0(\Delta),\ \sigma \in \Sigma(s)} \tau(s, \sigma)\}^{-1} \sum_{u: s' \in S_r(\Phi(\mathcal{S}_u); \Delta)} I_u \tag{27}$$

[7] A systematic way to obtain this result is by: (i) developing closed-form expressions for the optimal throughput $TH^*(\Phi(\mathcal{S}_i); \Delta^*),\ i = 1, 2$, through the steady-state analysis of the induced CTMC's describing the restriction of the RAS operation on the corresponding subspaces; and (ii) studying the sign of the difference $TH^*(\Phi(\mathcal{S}_1); \Delta^*) - TH^*(\Phi(\mathcal{S}_2); \Delta^*)$ for the two cases of Equation 24. We leave the computational details to the reader.

i.e., the transition corresponding to control action $\sigma \in \Sigma(s)$ can be activated in the final solution, only if this solution engages an edge selection $\mathcal{S}_u$ such that the resulting state s' is in $S_r(\Phi(\mathcal{S}_u); \Delta)$.

Example: To provide a more concrete demonstration of the synthesis of the constraint set implied by Equation 27, consider the thinning problem studied in the earlier example. Letting I_1, I_2 be defined as above, the constraints generated by Equation 27 for the decision variables $x(s, \sigma)$ corresponding to the three transitions out of state s_0 in the underlying CT-Markov process in Figure 4, are as follows:

$$x(s_0 \rightarrow s_7) \leq (\lambda_1 + \lambda_2 + \lambda_3) I_1 \tag{28}$$

$$x(s_0 \rightarrow s_8) \leq (\lambda_1 + \lambda_2 + \lambda_3) I_2 \tag{29}$$

$$x(s_0 \rightarrow \{s_9, s_{10}\}) \leq 0 \tag{30}$$

$\diamondsuit$

The complete formulation of the thinning problem is obtained by combining Equations 26 and 27 with the formulation of Equations 20, 14–16 and 21:

$$\max \sum_{s \in S^0_{rv}(\Delta)} \sum_{\sigma \in \Sigma(s)} g_1(s, \sigma) x(s, \sigma) \tag{31}$$

s.t.

$$\forall s \in S^0_{rv}(\Delta), \quad \sum_{\sigma \in \Sigma(s)} x(s, \sigma) - \sum_{s' \in S^0_{rv}(\Delta)} \sum_{\sigma \in \Sigma(s')} p(s; s', \sigma) x(s', \sigma) = 0; \tag{32}$$

$$\sum_{s \in S^0_{rv}(\Delta)} \sum_{\sigma \in \Sigma(s)} \tau(s, \sigma) x(s, \sigma) = 1; \tag{33}$$

$$\sum_{s \in S^0_{rv}(\Delta)} \sum_{\sigma \in \Sigma(s)} g_j(s, \sigma) x(s, \sigma) - \rho_j \sum_{s \in S^0_{rv}(\Delta)} \sum_{\sigma \in \Sigma(s)} g_1(s, \sigma) x(s, \sigma) = 0, \forall j \geq 2; \tag{34}$$

$$\forall s \in S^0_{rv}(\Delta),\ \forall \sigma \in \Sigma(s), \quad x(s, \sigma) \leq \Big\{ \min_{(s,\sigma):\ s \in S^0_{rv}(\Delta),\ \sigma \in \Sigma(s)} \tau(s, \sigma) \Big\}^{-1} \sum_{u : s' \in S_r(\Phi(\mathcal{S}_u); \Delta)} I_u; \tag{35}$$

$$\sum_u I_u = 1; \tag{36}$$

$$x(s, \sigma) \geq 0, \quad \forall s \in S^0_{rv}(\Delta),\ \forall \sigma \in \Sigma(s); \tag{37}$$

$$I_u \in \{0, 1\},\ \forall u. \tag{38}$$

In principle, this formulation can be solved through available commercial solvers [17]. However, we notice that the size of both variable sets, $x(s, \sigma)$ and I_u, is a super-polynomial function of $|\Phi|$ and, in general, these sets will grow very fast. Therefore, for practical purposes, it is important to develop alternative approximating schemes to the above MIP formulation. Such approximating schemes can be based on the adaptation, for the considered problem, of typical search-based combinatorial optimization algorithms, like *Tabu search* [7] and *genetic algorithms* [2], and they are part of our current investigations.

5 Conclusions

This paper provided a detailed formal characterization of the "thinning" problem, that was originally defined in [3]. The derived formulation is a mixed integer program, and therefore, it can be solved, in principle, through well-developed computational algorithms. A severe limitation of the approach, though, is that the number of the involved variables and constraints grows super-polynomially with respect to the underlying RAS size. Therefore, future work will seek to exploit the derived formulation towards the development of efficient approximating solution algorithms for the thinning problem.

An additional important contribution of the presented results was the employment of the CT-MDP modelling framework for the systematic characterization of (i) the D/C-RAS performance control problem, and (ii) the impact of the decisions pertaining to the D/C-RAS logical control on the achievable performance by the underlying system.[8]

Finally we notice that the thinning problem studied in this work belongs to the class of *congestion control* problems, since the quest is for a system configuration that will result in enhanced performance by minimizing the negative effects of the process contest for a shared set of resources. Under this interpretation, many of the ideas and techniques developed herein will

[8] Regarding this last issue, the formulation of Equations 31–38 has further prototypical value. For instance, one can easily envision a generalization of the aforementioned formulation where the applied LES Δ is an additional problem variable to be selected among an enumeration of candidate LES's. The formulation of Equations 31–38 can address this extended version of the problem, by redefining the binary variables I_u so that each of them corresponds to a combination of a candidate LES with a maximal edge selection. Furthermore, by fixing the structure of the process graphs, $\mathcal{G}_j$, according to some particular edge selection $\mathcal{S}$, the same formulation will return the LES Δ^* that maximizes the throughput of the resulting D/C-RAS $\Phi(\mathcal{S})$.

apply to a larger set of application contexts; as a concrete example, we mention the problem of selecting a set of routes for travelling between any pair of nodes in an AGV or an urban railway network, that maximizes the system throughput by minimizing the blockage experienced by the various vehicles.

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] B. P. Buckles and F. E. Petry (eds.). *Genetic Algorithms*. IEEE Computer Society Press, Los Alamitos, CA, 1992.

[3] S. F. Chew, M. Lawley, and S. Reveliotis. Liveness enforcing supervision for resource allocation with complex workflows. In *Proceedings of MMAR'03*, pages 823–829. IEEE, 2003.

[4] J. Y. Choi and S. A. Reveliotis. A generalized stochastic petri net model for performance analysis and control of capacitated re-entrant lines. *IEEE Trans. on Robotics and Automation*, 19:474–480, 2003.

[5] J. Y. Choi and S. A. Reveliotis. Relative value function approximation for the capacitated re-entrant line scheduling problems: An experimental investigation. In *Proceedings of CDC'04*, pages –. IEEE, 2004.

[6] J. Ezpeleta, F. Tricas, F. Garcia-Valles, and J. M. Colom. A banker's solution for deadlock avoidance in fms with flexible routing and multiresource states. *IEEE Trans. on R&A*, 18:621–625, 2002.

[7] F. Glover. Tabu search: A tutorial. *INTERFACES*, 20:74–94, 1990.

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

[9] 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.

[10] H. T. Papadopoulos, C. Heavy, and J. Browne. *Queueing Theory in Manufacturing Systems Analysis and Design*. Chapman & Hall, New York, NY, 1993.

[11] 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.

[12] M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 1994.

[13] S. A. Reveliotis. On the siphon-based characterization of liveness in sequential resource allocation systems. In *Applications and Theory of Perti Nets 2003*, pages 241–255, 2003.

[14] S. A. Reveliotis and J. Y. Choi. On the optimality of randomized deadlock avoidance policies. *Jrnl of DEDS: Theory and Applications*, 13:303–320, 2003.

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

[16] S. A. Reveliotis, M. A. Lawley, and P. M. Ferreira. Structural control of large-scale flexibly automated manufacturing systems. In C. T. Leondes, editor, *The Design of Manufacturing Systems*, pages 4–1 – 4–34. CRC Press, 2001.

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