Performance Optimization for a Class of Generalized Stochastic Petri Nets

Ran Li and Spyros Reveliotis
School of Industrial & Systems Engineering
Georgia Institute of Technology
{rli63,spyros}@isye.gatech.edu

Abstract

Motivated by the current scheduling needs of complex resource allocation systems, this paper introduces a novel methodology for performance optimization in DES applications that evolve over very large and complex state spaces and the main objective is expressed as the long-run maximization of some reward rate. The proposed methodology leverages the Generalized Stochastic Petri Net (GSPN) modeling framework in order to effect the seamless integration of the logical and performance-oriented control of the aforementioned applications, define a pertinent policy space, and (re-)cast the performance optimization problem into a mathematical programming formulation that is eventually solved through sensitivity analysis of Markov reward processes and stochastic approximation algorithms. An important attribute of the proposed methodology is that it facilitates an explicit control of the existing trade-off between the computational tractability of the employed formulation and the performance of the derived policies. Furthermore, by posing the eventually defined problem as a constrained nonlinear programming formulation, the presented methodology inherits all the analytical tools and insights that are offered by that vast area of optimization theory. In the current manuscript, all these possibilities are demonstrated through the application of the proposed approach to the throughput maximization of a capacitated re-entrant line abstracting the operation of an automated manufacturing cell.

1 Introduction

Generalized stochastic Petri nets (GSPNs) are one of the best known and most extensively used models in the class of *timed* PN models. The defining element of these nets is that they differentiate their transitions into two classes, respectively known as "timed" and "untimed". Timed transitions present non-zero firing times drawn from exponential distributions, while untimed transitions have zero firing times. In [2], the seminal paper that introduced the GSPN model, but also in most of the remaining works in the GSPN-related literature [1], untimed transitions have been promoted as a "mechanism" for controlling the complexity of the time-based analysis of the corresponding PN models, while preserving all the behavioral traits of the underlying system that are captured by these transitions. In that modeling paradigm, untimed transitions essentially correspond to dynamics of the underlying system behavior that are much faster than the dynamics that are modeled by the timed transitions, and therefore, the relevant delays can be safely ignored in the stochastic process that eventually models the timed behavior of the underlying system.

In this paper, we consider a different role for the untimed transitions in the proposed GSPN models. More specifically, in our models, untimed transitions constitute "decisions" that are to be determined by an external control function in an effort to govern the system dynamics, while timed transitions correspond to the execution of the operations that are commanded by the aforementioned decisions. Similar to the standard GSPN modeling framework, conflicting decisions at any given marking are resolved by the specification of a *"random switch"*, i.e., a probability distribution that regulates the selection of these decisions. But while in the past applications of the GSPN model the necessary random switches are specified by the human modeler so that they reflect properly the (intended) operation of the underlying system, in this work, the probabilities defining these random switches are "decision variables" to be determined so that certain performance objective(s) are optimized, by the techniques to be developed herein.

From a more practical standpoint, the developments presented in this work have been motivated by an emerging need to develop a modeling and computational framework that is able to address, in an integrative manner, logical and performance-oriented control / scheduling problems in Discrete Event Systems (DES) theory, while coping effectively with the enormous computational challenges that underlie both of these problem classes. Indeed, due to their compactness and rich analytical properties, PNs are currently recognized as one of the most effective modeling frameworks for supporting logical – or qualitative – analysis and control of various DES classes. At the same time, the extension of the basic PN model to a GSPN through the introduction of the aforementioned timing attributes for its transitions, enables the modeling and analysis of the resultant timed dynamics through the well-developed theory of (semi-)Markov processes [38]. The developments to be pursued in this paper will further relate these dynamics to the theory of Markov Reward and Markov Decision Processes (MDPs) [32, 9].

On the other hand, while the classical MDP modeling framework can provide a natural representation for the DES performance optimization problems that are considered in this work, its practical applicability is severely constrained by the very large size of the involved state spaces. The presented approach deals with these computational challenges by (i) restricting the target policy space to a subset of policies that admit a more parsimonious representation compared to the policies that are pursued by the basic MDP formulation, and (ii) recasting the original MDP formulation into a Mathematical Programming (MP) formulation that is effectively solvable, within the scope of the target policies, through results borrowed from the sensitivity analysis of Markov reward processes [9] and Stochastic Approximation (SA) [21]. As it will be revealed in the following, the adopted GSPN modeling framework provides fundamental mechanisms and insights for defining, both, the target policy space and the solution algorithms for the resulting MP formulations. In fact, according to these insights, the proposed methodology can be re-interpreted as an approximation scheme to the original MDP formulation that employs a state-space aggregation scheme defined by a "static" specification of the involved random switches. The realization of this last effect further enables the potential refinement of the employed policy space through (partial) disaggregation.

The theoretical developments that are described in the previous paragraphs and their potential, are highlighted in the second part of this paper through a demonstrating example borrowed from the area of the real-time management of complex resource allocation systems (RAS) [37]. The autonomous operation of these systems defines a DES subclass that must be carefully controlled for liveness and reversibility of the underlying resource allocation function, and the corresponding supervisory control problem has received extensive attention in the DES litera-

ture. In more specific terms, the provided example concerns the throughput maximization of an autonomous production cell that has the structure of a re-entrant line. However, contrary to the more classical re-entrant lines studied in the past literature [19], the considered re-entrant line has finite buffering capacity at its various workstations, and therefore, it is characterized as "capacitated". Furthermore, the blocking effects that arise from the presence of the finite buffers, invalidate the existing scheduling results for re-entrant lines with infinite buffer sizes, and render pretty challenging the analysis and resolution of the resulting scheduling problem [36]. In order to facilitate the relevant exposition, the system structure that is considered in the aforementioned developments is intentionally small; yet, both, the considered system and the pursued analysis are sufficiently rich to demonstrate effectively all the salient elements of the proposed methodology.

Given the above positioning of the presented work in terms of its main perspectives and the intended contribution, the rest of the paper is organized as follows: The next section reviews certain parts of the current literature that relate to the developments pursued in this work, and that will help define and position more clearly these developments and their intended contribution. Section 3 provides fundamental background for the main technical developments of the paper by overviewing the GSPN modeling framework and the steady-state performance evaluation of the GSPN models through their reduction to pertinent semi-Markov and Markov processes. Section 4 defines the MDP problem considered in this work, by further qualifying the structure of the considered GSPN models and policies. Section 5 discusses the solution of the MDP problem formulated in Section 4 through sensitivity analysis of the underlying stochastic processes and the integration of these results in SA algorithms. Some technical details that are needed for the implementation of these algorithms are provided in an appendix. Section 6 demonstrates the effectiveness of the derived results and their practical power and value by applying them on the prototypical RAS scheduling problem concerning the throughput maximization of the capacitated re-entrant line that was mentioned in the previous paragraph. Finally, Section 7 concludes the paper and proposes important directions for future work. Closing this introductory discussion, we also notice, for completeness, that an abridged version of the results presented in this paper has appeared in [23].

2 Literature Review

As remarked in the introductory section, a large set of problems concerning the performance control of Discrete Event Systems can be addressed through the modeling framework of MDPs and (stochastic) Dynamic Programming (DP) [32, 6]. Critical for this capability is the method of stages, which approximates any event timing distribution, to any desired degree of accuracy, by a phase-type distribution [10]. Phase-type distributions can be effectively represented by the time that is necessary to traverse an appropriately structured Markovian net, and they are easily implementable in the GSPN modeling framework through the substitution of the timed transition that models the considered event by the corresponding Markovian subnet. Furthermore, it is well known that MDP and DP formulations are further classified by the length of the considered horizon, and the structure of the optimized criterion. In this work, we are primarily focusing on infinite-horizon, average-reward problems (although the presented ideas can be adapted to address discounted-reward MDP formulations, as well).

For infinite-horizon, average-reward (AR-)MDPs, the tractability of the considered models in terms of the computation of the optimal policy and the relevant sensitivity analysis, strongly

depends on the communication structure of the Markov chains that are defined by the various candidate policies. In particular, some of the strongest computational and analytical results for these MDP formulations concern the case where all the candidate policies share a common behavioral space. In DES theory, this property can be effectively enforced through the design of pertinent supervisory control (SC) policies [33, 10] that establish nonblocking behavior, confining the system in a single communicating class of the underlying state space; once the system is confined in such a communicating subspace, the commonality of this subspace across all the considered policies can be achieved with a slight randomization of the policy decisions at each visited state. It is also advantageous to have the admissible behavioral space as large as possible, since such a maximized behavioral latitude provides a richer policy space for the corresponding MDP formulation. SC policies that maximize the admissible behavioral space are characterized as "maximally permissive" and there is a well-developed theory regarding the characterization and effective computation of such policies [42, 10]. Furthermore, there is an extensive specialization of this general theory to the particular problem of establishing maximally permissive liveness-enforcing supervision (LES) for the subclass of DES modeling complex RAS [37, 24, 35]. A particular important attribute of the LES-synthesis theory for complex RAS is that it is able to provide LES that are maximally permissive and realizable with a very low on-line computational cost. Furthermore, for most RAS instantiations, these LES admit either an immediate PN-based representation or they can be effectively approximated by some compact PN subnet [13]. We shall return to this capability in Section 6, where we discuss the application of the presented methodology to the throughput maximization of the capacitated re-entrant lines.

Under the behavioral restrictions that were described in the previous paragraph, the AR-MPDs considered in this work can be solved, in principle, by any classical MDP approach, i.e., value-iteration, policy-iteration or linear programming [32, 6]. However, all these methods require an explicit enumeration of the underlying state space, and for most DES applications, this state space has an exponential size w.r.t. any parsimonious description of the controlled plant. This effect is known as the "curse of dimensionality" [5] that plagues most practical applications of DP, and in the case of DES applications, it stems from the fact that the traced behavior is the composition of the behaviors that are generated by a number of interacting components evolving over their own subspaces. In fact, the aforementioned "curse of dimensionality" challenges even the mere description of an optimal policy for the considered MDP formulations, since such a description requires the specification of an optimal action for each state. Hence, in the recent years, there has been an extensive effort to develop methodology that can provide approximating policies to the optimal policies of the target MDP formulations, which admit a more parsimonious representation and yet remain efficient w.r.t. the stated performance objectives. All these methods are collectively known as Approximate DP (ADP) [6, 31].

Some prominent directions in ADP theory are as follows [6]: A first line of research tries to identify and aggregate states that should share the same action under an optimized control policy. Hence, the target policies are eventually defined w.r.t. the state aggregates, and the corresponding approximation schemes are said to be based on state aggregation. But from a more practical standpoint, the specification of meaningful aggregation schemes for various MDP formulations is a theme that has received limited attention in the existing literature, and the relevant results depend extensively upon the underlying application context. An alternative approach seeks to develop a good approximation of the underlying value function by means of a parameterized representational structure that is known as the approximating "architecture". This architecture is determined in an effort to establish a pertinent trade-off between the sought

representational accuracy of the target value function and the resultant representational and computational complexity [7]. Once the approximating architecture has been determined, the MDP problem boils down to the problem of estimating the corresponding parameters. However, the synthesis of a pertinent architecture for a given MDP formulation is currently a rather *ad hoc* process, while the corresponding estimation theory is primarily developed for linear architectures that represent the state values as weighted sums of a certain set of numerical attributes extracted from the broader information that is provided by the system state; for this last case, the literature offers variations of the value and the policy iteration methods, as well as LP formulations, that are defined on the underlying parameter space, and, under certain assumptions, they have strong convergence guarantees [6]. On the other hand, it is also true that the greedy policies that are defined by the obtained value functions can have a quite erratic performance, when compared to the performance of the actual optimal policy. Recognizing the existing limitations and challenges of the ADP methods that are based on value function approximation, a third ADP approach seeks to define explicit parameterized representations of the target policies themselves, and eventually optimize over this new parameter space. Instrumental for this approach is the ability to develop "sensitivity" results that relate the observed performance to the parameterization of the applied policies. The availability of such a capability, e.g., in the form of performance gradients or sub-gradients, can reduce the overall optimization problem to the solution of an MP formulation. Of particular interest are the variations of this approach where the necessary (sub-)gradients are computed through simulation or through the actual observation of the considered system under operation by the evaluated policy; the corresponding optimization area is broadly known as stochastic approximation (SA) [21]. Furthermore, the DES theory avails of an entire class of algorithms collectively known as "Infinitesimal Perturbation Analysis (IPA)", that are sample-path-based and concern the parameter optimization of certain scheduling policies for queueing stations and networks [16, 8]. More recently, this theory has also been extended to the sensitivity analysis and optimization of Markov reward processes [9]. An alternative parameter optimization method for DES is the "score function" method of [39], while of particular affinity to the developments that are presented in this paper are the simulation-based policy-optimization methods developed in [26, 18, 27].

In the light of the above discussion, the methodology that is pursued in the rest of this paper can be described as a simulation-based optimization approach that optimizes over a (continuous-)parameter space defining the target set of policies. The method uses the GSPN modeling framework in order to motivate and describe the target policy space, but also in order to structure the SA algorithm that is used for the solution of the resulting MP formulation. In addition, this methodology is based on perspectives and results concerning the sensitivity analysis and optimization of Markov reward processes taken from [9]. As it will be revealed in the remaining sections, the resulting approach is computationally tractable, and it enables a systematic optimization over a policy set that subsumes the typical heuristics currently used in the target application areas. Furthermore, the policy structure that results from the proposed policy parameterization can be re-interpreted as a state aggregation scheme, and this realization allows the further enrichment of the underlying policy space through controlled (partial) disaggregation. At the same time, the juxtaposition of the paper developments with the results presented in [26, 18, 27] indicates that the methodology appearing in this last set of works could be adjustable to the considered MDP formulation, allowing, thus, for the development of single-sample-path optimization algorithms for the considered problem. The rest of the paper articulates and concretizes the above claims, while the complete exploration and realization of the potential that is implied by some of these statements is also part of our current investiga-

tions.

Closing this literature survey on the past works pertaining to the results that are presented in this paper, we also notice, for completeness, that the work presented in [4] employs the GSPN modeling framework for representing decision-making processes that are conceptually similar to those considered in this paper. However, the modeling assumptions and details in the two works and the pursued approaches for the simplification and solution of the derived analytical formulations are fundamentally different. Furthermore, the work of [3] presents a very interesting extension of the GSPN modeling framework that is used for the characterization and resolution of a series of probabilistic properties of the underlying plant model. The methodology supporting the assessment of these properties is based on a combination of results borrowed from MDP theory with a structural analysis of the underlying transition model.

3 Generalized Stochastic Petri Nets

In this section we provide an overview of the basic GSPN theory. In the subsequent discussion we assume that the reader is familiar with the basic PN modeling framework, and we focus primarily on the time-related aspects of the GSPN model and its analysis; a nice exposition of the basic PN modeling framework and its supporting theory can be found in [29].

Following [2], we define a *Generalized Stochastic Petri Net (GSPN)* as a Petri net system $\mathcal{N} = (P, T, W, m_0)$ where the transition set T is partitioned in two subsets T_t and T_u denoting respectively the sets of *timed* and *untimed* transitions.[1] Furthermore, there is a mapping $R: T_t \to \mathbb{R}^+$ with $r_t \equiv R(t),\ t \in T_t$, denoting the (instantaneous hazard) rate of the exponential distribution that determines the *firing times* for transition t. These firing times are interpreted as a delay between the enabling of transition t and the actual firing of the transition.[2] On the other hand, transitions $t \in T_u$ have zero firing times, i.e., these transitions can fire as soon as they are enabled.

Let $\mathcal{R}(\mathcal{N})$ denote the reachability space of net $\mathcal{N}$. The firing dynamics of the GSPNs $\mathcal{N}$ considered in [2] are further qualified by the following assumptions: (i) $|\mathcal{R}(\mathcal{N})| < \infty$, and (ii) for any marking $m \in \mathcal{R}(\mathcal{N})$, the set of enabled transitions $\mathcal{E}(m)$ is non-empty (in other words, the considered GSPNs do not contain any total deadlocks). For any marking $m \in \mathcal{R}(\mathcal{N})$, let $\mathcal{E}_t(m)$ and $\mathcal{E}_u(m)$ denote respectively the subsets of the timed and untimed transitions enabled in m, and suppose first that $\mathcal{E}_u(m) = \emptyset$ and $\mathcal{E}_t(m) = \{t_1, t_2, \ldots, t_n\}$. From the basic properties of the exponential distribution [10], marking m has an expected sojourn time of $s(m) = 1/\sum_{t \in \mathcal{E}_t(m)} r_t > 0$, and therefore, it is called a *tangible* marking. Furthermore, the transitions in $\mathcal{E}_t(m)$ define an exponential race in m, and the probability that transition $t_i,\ i = 1, \ldots, n$, will fire first is equal to $p_{t_i}(m) = r_{t_i}/\sum_{t \in \mathcal{E}_t(m)} r_t$. On the other hand, in a marking $m \in \mathcal{R}(\mathcal{N})$ for which $\mathcal{E}_u(m) \neq \emptyset$, any transition $t \in \mathcal{E}_u(m)$ is expected to fire before a transition $t \in \mathcal{E}_t(m)$. In particular, if $\mathcal{E}_u(m)$ is a singleton, then, the single transition t in it will fire at m with certainty. If $\mathcal{E}_u(m)$ contains more than one transitions, then, there is a need for some

[1] Also, we should notice, for completeness, that according to standard PN notation, the other three elements in the net-defining tuple are the place set P, the flow relation W that defines the connectivity of the net, and the net initial marking m_0.

[2] As remarked in Section 2, the restriction of the GSPN timed dynamics to Markovian behavior does not restrict substantially its modeling power, since (i) more generally distributed firing times can be approximated to any desired degree of accuracy by phase-type distributions, and (ii) the latter can be modeled by appropriately structured GSPN subnets [10].

arbitrating mechanism that will define the prospects of every transition t in $\mathcal{E}_u(m)$ to be the firing transition in m. In the standard GSPN theory this arbitrating mechanism is provided by a probability distribution defined on $\mathcal{E}_u(m)$ that is called the *random switch* associated with m. Furthermore, it should be clear from the above discussion that a marking $m \in \mathcal{R}(\mathcal{N})$ with $\mathcal{E}_u(m) \neq \emptyset$ has zero sojourn time, and therefore, such a marking is characterized as *vanishing* in the relevant literature. In the following, we shall denote the sets of tangible and vanishing markings by $\mathcal{R}_t(\mathcal{N})$ and $\mathcal{R}_v(\mathcal{N})$, respectively. Clearly, $\mathcal{R}_t(\mathcal{N})$ and $\mathcal{R}_v(\mathcal{N})$ define a partition of $\mathcal{R}(\mathcal{N})$. Finally, the random switch associated with a marking $m \in \mathcal{R}_v(\mathcal{N})$, with $\mathcal{E}_u(m) = \{t_1, t_2, \ldots, t_n\}$, will be denoted by $\Xi(m) = [\xi_1, \xi_2, \ldots, \xi_n]^T$.

The firing dynamics of the GSPN model that were described in the previous paragraph define a stochastic process on $\mathcal{R}(\mathcal{N})$ that has the particular structure of a semi-Markov process [2]; in the following, we shall denote this semi-Markov process by $\mathcal{SM}(\mathcal{N})$. Sojourn times and branching probabilities for states (or, equivalently, markings) $m \in \mathcal{R}_t(\mathcal{N})$ are determined by the corresponding exponential races described in the previous paragraph and the net flow relation W_t, i.e., the restriction of W on the timed transitions of the net. On the other hand, sojourn times for markings $m \in \mathcal{R}_v(\mathcal{N})$ are equal to zero and the corresponding branching probabilities are determined by the random switches $\Xi(m)$ assigned to those markings, together with the net flow relation W_u. When net $\mathcal{N}$ is also reversible under the control of the specified random switches $\Xi(m),\ m \in \mathcal{R}_v(\mathcal{N})$, process $\mathcal{SM}(\mathcal{N})$ can be shown to be ergodic [10], and thus, analyzed for its long-term (or *"steady-state"*) behavior through standard techniques borrowed from the theory of ergodic stochastic processes. For ergodic GSPNs it is worth-noticing that, since markings $m \in \mathcal{R}_v(\mathcal{N})$ have zero sojourn time in $\mathcal{SM}(\mathcal{N})$, they will also have zero steady-state probabilities. More generally, the timed dynamics of any given GSPN $\mathcal{N}$ can be effectively analyzed by the sub-process $\mathcal{M}(\mathcal{N})$ that projects the dynamics of $\mathcal{SM}(\mathcal{N})$ on the subspace of tangible markings. $\mathcal{M}(\mathcal{N})$ is a CTMC and its infinitesimal generator $Q(\mathcal{N})$ can be computed from the structural characterization of $\mathcal{SM}(\mathcal{N})$ [2]. For an ergodic $\mathcal{SM}(\mathcal{N})$, the availability of $Q(\mathcal{N})$ enables also the characterization of the steady-state probability distribution $\pi \equiv [\pi(m),\ m \in \mathcal{R}_t(\mathcal{N})]$ of $\mathcal{M}(\mathcal{N})$, and the long-term behavior of the underlying system.

4 The considered optimization problem

The original optimization problem It should be clear from the discussion of Section 3 that, in the current literature, the GSPN model has functioned as a performance evaluation tool for the underlying system. In particular, for ergodic such nets, the analysis of the induced processes $\mathcal{SM}(\mathcal{N})$ and $\mathcal{M}(\mathcal{N})$ can provide a succinct characterization of the steady-state dynamics. Additional long-term performance considerations can be investigated in this limiting regime by introducing an *"(immediate) reward"* or *"cost"* function on the net markings and/or its transitions, and considering the expectation of these functions w.r.t. the steady-state probability distribution π. In this way, one can characterize and compute "throughput" and "utilization rates" w.r.t. various transitions of the net as well as "holding costs" associated with the marking of its various places [12].

This work intends to use the GSPN modeling framework in a more "prescriptive" manner, namely for the framing and the investigation of the following fundamental problem: *Given a (RAS-modeling) GSPN net defined by (i) the net topology and (ii) the firing rates of its timed transitions, and (iii) an immediate reward function defined on the net markings and/or transitions that characterizes a performance measure of interest, find a set of random switches*

for the net vanishing markings that maximizes the long-term performance of the net w.r.t. the aforementioned performance measure.

As explained in the introductory section, we perceive the random switches associated with the various vanishing markings as "decisions" that are effected upon the net dynamics through the selection and firing of an enabled untimed transition from their support transition sets according to the corresponding probability distributions. On the other hand, timed transitions model the execution of the various processes / operations that are activated by these decisions. Hence, the net dynamics evolve through an alternation of tangible markings that correspond to the concurrent execution of a number of processes in the net, and a set of vanishing markings that constitute decision points in response to the completion of some of the activated processes. In fact, such a process completion can activate a cascading sequence of decisions, that are modeled by a corresponding sequence of vanishing markings, before the net settles to another processing phase modeled by the next tangible marking.

The GSPN performance optimization problem described in the previous paragraphs, when combined with the Markovian nature of the stochastic processes that model the timed behavior of the GSPN nets, can be casted in the framework of MDP theory [32]. In particular, the specification of a set of random switches that will regulate the transition firing in the vanishing markings of the considered GSPNs essentially defines a stationary policy that shapes the net dynamics by defining the stochastic processes $\mathcal{SM}(\mathcal{N})$ and $\mathcal{M}(\mathcal{N})$ that were discussed in the previous section. In the following, we shall denote by ϕ the stationary policy that results from any particular selection of random switches, and we shall also use Φ to denote the space of all the considered stationary policies. We further stipulate that the considered policies are able to ensure ergodic behavior of the underlying net. From a practical standpoint, this requirement can be satisfied by (i) superimposing on the original net an appropriate supervisory control policy (SCP) [10] that will enforce the reversibility of the net behavior,[3] and (ii) lower-bounding the probability values of the random switches that will coordinate the restricted net behavior by a small value $\delta > 0$.[4] Then, letting $f \equiv [f(m),\ m \in \mathcal{R}_t(\mathcal{N})]$ denote a reward rate function defined on the markings $m \in \mathcal{R}_t(\mathcal{N})$, $\pi(\phi)$ denote the steady-state probability distribution for the CTMC $\mathcal{M}(\mathcal{N})$ that results from the application of policy ϕ, and $\eta(\phi)$ denote the resulting (long-term) average reward, our optimization problem can be expressed by the following mathematical programming (MP) formulation:

$$\max_{\phi \equiv \{\Xi(m), m \in \mathcal{R}_v(\mathcal{N})\}} \eta(\phi) = \pi^T(\phi) \cdot f \tag{1}$$

s.t.

$$\pi^T(\phi) \cdot [Q(\phi)\ \mathbf{1}] = [\mathbf{0}^T 1] \tag{2}$$

$$\forall m \in \mathcal{R}_v(\mathcal{N}),\ \ \Xi^T(m) \cdot \mathbf{1} = 1.0 \tag{3}$$

$$\forall m \in \mathcal{R}_v(\mathcal{N}),\ \forall t \in \mathcal{E}_u(m),\ \ \delta \leq \xi(t) \tag{4}$$

The entities $\mathbf{1}$ and $\mathbf{0}$ that appear in the above formulation denote appropriately dimensioned column vectors with all their elements respectively equal to one and zero. Equation 1 of this formulation expresses the objective of maximizing the long-term average reward that is collected

[3] For notational economy, in the following we shall use $\mathcal{N}$ to refer to the controlled net, as well.

[4] Besides its mathematical necessity for the well-posedness of the proposed optimization problem, the randomization in the decision-making process introduced by δ will also be a useful "exploration" mechanism in the solution of the proposed formulation through the sample-path-based techniques that are pursued in this work.

according to the reward function f. Constraints 3 and 4 define the structure of the random switches, that are the primary decision variables in the considered optimization problem. On the other hand, Constraint 2 defines the steady-state probability distribution $\pi(\phi)$ for the CTMC that is induced by any pricing of the random switches $\Xi(m),\ m \in \mathcal{R}_v(\mathcal{N})$, according to Constraints 3 and 4; it is important to notice that in the considered formulation, this probability distribution is an auxiliary decision variable that enables the statement of the objective of the formulation.

Complexity considerations and the revised optimization problem From a computational standpoint, the formulation of Equations 1–4 is challenged by the fact that its size, in terms of the employed decision variables and constraints, is commensurate to the size of $\mathcal{R}_v(\mathcal{N})$, which grows exponentially w.r.t. the size of underlying the GSPN model; the latter is defined by $|P| \times |T|$, which is the size of the largest item in the data structure that defines $\mathcal{N}$. This limitation is essentially the same with the computational challenges that limit the practical solution of the considered optimization problem through the classical MDP theory [30]. In particular, the employment of a separate random switch for every marking in $\mathcal{R}_v(\mathcal{N})$ implies that even the mere enumeration of a given policy ϕ from the considered class is a task of non-polynomial complexity w.r.t. the size of the underlying RAS. We shall characterize this fact by saying that the aforementioned policies ϕ possess *exponential space complexity.*

To deal with this increased, and frequently computationally prohibitive complexity, in this work we shall restrict attention to a class of policies that require a reduced space complexity for their characterization. The considered policies are suggested naturally by the structure of the underlying GSPN model, and in many cases, they also define policy classes of practical significance and value for the original optimization problem under consideration. From a mathematical modeling standpoint, these policies will be defined through the introduction of additional constraints in the MP formulation of Equations 1–4 that will establish some "coupling" among the variables of the random switches $\Xi(m)$ that appear in the formulation of Equations 1–4, and will enable the elimination of some of these variables. A simple way to implement this idea is by introducing the following additional constraint to the optimization problem of Equations 1–4:

$$\forall m, m' \in \mathcal{R}_v(\mathcal{N}) \text{ with } \mathcal{E}_u(m) = \mathcal{E}_u(m'),\ \ \Xi(m) = \Xi(m') \tag{5}$$

Under Equation 5, the random switches $\Xi(m)$ that are employed by the resultant formulation, are essentially defined only by their support sets $\mathcal{E}_u(m)$ and not by the marking m itself. Such random switches are characterized as *static* in the GSPN literature.[5] Also, we shall refer to the optimization problem that is defined by Equations 1–5 as the *"revised optimization problem"*. The reader should notice that the set of static random switches is of cardinality $O(2^{|T_u|})$, i.e., in principle, the problem space complexity remains an exponential function of the size of the underlying GSPN $\mathcal{N}$. However, it is generally true that $2^{|T_u|} << |\mathcal{R}_v(\mathcal{N})|$. Furthermore, in most practical cases, the subsets of T_u that define support sets for some random switch $\Xi(m)$ of $\mathcal{N}$ will be significantly less than $2^{|T_u|}$, since these subsets are further constrained by the topology of $\mathcal{N}$ and the dynamics that are induced by this topology. Hence, it is expected that the revised optimization problem will be much more manageable in terms of its space and time complexity than the original optimization problem of Equations 1–4.[6]

[5] On the other hand, random switches that depend also on the marking m are characterized as *dynamic*.

[6] The last remarks are further exemplified through the case study that is presented in Section 6.

We should also point out that, in spite of the simplicity of the logic that underlies Equation 5, the resultant class of policies is pretty rich and of practical relevance in many practical applications. In particular, the policy space that is defined by the set of static random switches enables the modeling of static-priority policies defined on the basis of various "class" concepts. Such policies have been attractive in many application contexts due to their operational simplicity, and in various cases they have been proven to be optimal (c.f., for instance, [19, 20]).

Furthermore, it is easy to see that in the revised problem formulation, Constraint 5 essentially defines an aggregation scheme on the underlying subspace $\mathcal{R}_v(\mathcal{N})$, by imposing the requirement that decisions in vanishing markings m with the same set of enabled untimed transitions $\mathcal{E}_v(m)$ should be governed by the same decision rule (i.e., by the same random switch $\Xi(m)$). One can envision the further enhancement of this formulation through a refining process that partitions (some of) the aggregated state sets into smaller subsets. When this refinement is taken to its extreme, one retrieves the original formulation of Equations 1–4. In the general case, such a refinement can be effected through trial-and-error-based procedures and search-based mechanisms similar to those used in the area of combinatorial optimization [15]. In certain cases, it may also be based on special problem characteristics that are derived from the structure of the underlying system. The concise and explicit representation of the structure of the underlying system by the employed GSPN model enables a succinct articulation of the applied refinement logic, and it facilitates the identification of behavioral traits and attributes that may be exploitable by the refinement process.

Uniformization and a further reduction of the revised optimization problem In the next section we shall present a solution of the revised optimization problem based on some results for sensitivity analysis of Markov reward processes. To apply these results, it is convenient to discretize the Markovian dynamics that underlie the revised problem formulation, by uniformizing [10] these dynamics with an appropriate sampling rate r_u, e.g., $r_u = \sum_{t \in T_t} r_t$. In the following, we shall denote the resulting discrete-time Markov chain (DTMC) by $\mathcal{U}(\mathcal{N})$ and the corresponding one-step-transition probability matrix by $\hat{P}(\mathcal{N})$. Also, let us denote by $\bar{\xi}$ the set of variables that remain in the final formulation of the considered problem, after removing all the variables that are rendered superfluous by the constraint of Equation 5 (or, more generally, by the additional constraints that define the target policies). Then, recognizing also the auxiliary role of variables $\pi(\phi)$, the final problem formulation can be reduced, at least in principle, to

$$\max_{\bar{\xi}} \ \eta(\bar{\xi}) \tag{6}$$

s.t.

$$\forall \xi \in \bar{\xi}, \quad \delta \leq \xi \tag{7}$$

$$\forall \Xi, \quad \delta \leq 1.0 - \sum_{\xi \in \Xi \cap \bar{\xi}} \xi \tag{8}$$

In Equation 8, Ξ denotes those random switches that are recognized as distinct entities by the definition of the underlying policy space. We also notice that in the rest of this work we shall use interchangeably the vector $\bar{\xi}$ and the policy ϕ that is induced by this vector.

5 Solving the considered optimization problem

The performance sensitivity formula The optimization problem of Equations 6–8 can be addressed by rather standard techniques from the theory of Mathematical Programming (MP)

[25], provided that the partial derivatives $\frac{\partial \eta(\bar{\xi})}{\partial \xi}$ are well defined over the feasibility space defined by Constraints 7 and 8. In [9] it is shown that for discrete-time Markov reward processes of the type that underlie the MP formulation of Equations 6–8, the derivative of the performance index η w.r.t. any parameter ξ that is involved in the definition of the corresponding one-step-transition probability matrix $\hat{P}$, can be computed as follows:

$$\frac{\partial \eta(\bar{\xi})}{\partial \xi} = \pi(\bar{\xi})^T \cdot \frac{\partial}{\partial \xi} \hat{P}(\bar{\xi}) \cdot g(\bar{\xi}) \tag{9}$$

The vector $g(\bar{\xi})$ that appears in the above equation is the *relative value function* (or the *potential function*) for the uniformized Markov reward process $\mathcal{U}(\mathcal{N}, f; \phi)$. In the considered class of Markov reward processes, vector $g(\bar{\xi})$ is defined by the corresponding Bellman equation up to an additive constant [32]. However, since $\hat{P}(\bar{\xi})$ is a stochastic matrix, the matrix $\frac{\partial}{\partial \xi}\hat{P}(\bar{\xi})$ has zero row sums, and therefore, the computation of $\frac{\partial \eta(\bar{\xi})}{\partial \xi}$ through Equation 9 is invariant to the various selections of $g(\bar{\xi})$. Furthermore, in the next paragraphs we shall show that, in the considered GSPN models, the elements of the matrix $\hat{P}(\bar{\xi})$ are polynomials of the decision variables ξ. Therefore, the (matrix) partial derivative $\frac{\partial}{\partial \xi}\hat{P}(\bar{\xi})$, that appears in Equation 9, will always exist. Hence, as long as the variable vector $\bar{\xi}$ is selected in the space defined by Equations 7 and 8, and the resultant matrix $\hat{P}(\xi)$ corresponds indeed to the one-step-transition probability matrix of an irreducible, ergodic Markov chain, the derivative $\frac{\partial \eta(\bar{\xi})}{\partial \xi}$ will be well defined.

Sample-path-based estimation of the performance derivatives From a computational standpoint, the practical value of Equation 9 is limited by the fact that each of the three factors that appear in its right-hand-side is an entity of size commensurate to the size of the underlying state space; as explained in Section 4, the state space for the considered processes will explode even for rather small GSPN configurations. But the results of [9] provide also sample-path-based estimators for $\frac{\partial \eta(\bar{\xi})}{\partial \xi}$ that are derived from Equation 9 and the finite, irreducible nature of the underlying Markov process. Such an estimator can be obtained, for instance, from the following result [9]:

$$\frac{\partial \eta(\bar{\xi})}{\partial \xi} = \frac{E\left[\sum_{k=u_\nu}^{u_{\nu+1}-1} [f(m_k) - \eta(\bar{\xi})] \cdot \hat{r}_k(\xi)\right]}{E[u_{\nu+1} - u_\nu]} \tag{10}$$

In Equation 10, $\mathbf{m} = \langle m_0, m_1, m_2, \ldots \rangle$ denotes a sample path of the considered process, and the sequence $u_\nu,\ \nu = 0, 1, 2, \ldots$, collects the time points of the path visits to some selected state m^* that constitutes a regenerative point for the underlying stochastic process. The quantity $\hat{r}_k(\xi)$ that appears in Equation 10 is defined by

$$\hat{r}_k(\xi) \equiv \sum_{j=u_{\nu(k)}}^{k} \frac{\frac{\partial}{\partial \xi}\hat{p}(m_{j-1}, m_j; \bar{\xi})}{\hat{p}(m_{j-1}, m_j; \bar{\xi})} \tag{11}$$

where $u_{\nu(k)}$ denotes the time step of the last visit, prior to time step k, to the aforementioned regenerative point m^*. Equation 11 implies that the detailed specification of a $\frac{\partial \eta(\bar{\xi})}{\partial \xi}$ estimator on the basis of Equation 10 necessitates the characterization of the elements of matrix $\hat{P}(\bar{\xi})$ as functions of $\bar{\xi}$, and of the partial derivatives of these elements w.r.t. each decision variable ξ. Next we provide these additional characterizations that will enable, in the considered application context, the estimation of $\frac{\partial \eta(\bar{\xi})}{\partial \xi}$ on the basis of Equations 10–11.

Computing the elements of matrix $\hat{P}(\bar{\xi})$ and their partial derivatives As explained in Section 4, the interpretation of the random switches Ξ as "decisions" that drive the system behavior, further implies that, in the underlying semi-Markov process $\mathcal{SM}(\mathcal{N})$, the transition from a tangible state m to another tangible state m' will be interfered, in general, by a sequence of vanishing markings $\hat{M} = \{\hat{m}_1, \hat{m}_2, \ldots, \hat{m}_k\}$. Sequence $\hat{M}$ will contain no cyclical behavior; i.e., $\hat{m}_i \neq \hat{m}_j$ for $i \neq j$, since the repetition of vanishing markings in sequence $\hat{M}$ can be perceived as a manifestation of "confusion" in the underlying decision-making process. However, it is conceivable that two or more of the vanishing markings in $\hat{M}$ might possess the same static random switch Ξ; i.e., it is possible that $\mathcal{E}_v(\hat{m}_i) = \mathcal{E}_v(\hat{m}_j)$ for some $i \neq j$, in which case, Equation 5 further implies that $\Xi(\hat{m}_i) = \Xi(\hat{m}_j)$. Next, consider the transition sequence $\hat{T} = \{\hat{t}_0, \hat{t}_1, \ldots, \hat{t}_k\}$ where $m \xrightarrow{\hat{t}_0} \hat{m}_1$, $\hat{m}_i \xrightarrow{\hat{t}_i} \hat{m}_{i+1}$ for $i = 1, \ldots, k-1$, and $\hat{m}_k \xrightarrow{\hat{t}_k} m'$. $\hat{T}$ can be considered as another representation of the dynamics that lead process $\mathcal{SM}(\mathcal{N})$ from m to m' through the sequence of vanishing markings $\hat{M}$. Furthermore, this representation facilitates a straightforward computation of the realization probability of the corresponding dynamics. More specifically, let $\hat{p}(\hat{t}_i; \bar{\xi})$, $i = 0, 1, \ldots, k$, denote the probability for firing transition $\hat{t}_i$ in the corresponding marking (i.e., marking m for $i = 0$, or marking $\hat{m}_i$ for $i = 1, \ldots, k$). Then, taking also into consideration (i) the transitional dynamics that are introduced by the uniformizing operation that was introduced in Section 4, and (ii) the specification of the variable vector $\bar{\xi}$ employed by the formulation of Equations 6–8, we have that:

$$\hat{p}(\hat{t}_i; \bar{\xi}) = \begin{cases} \frac{r_{\hat{t}_i}}{r_u}, & \text{for } i = 0 \\ \xi(\hat{t}_i) \in \Xi(\hat{m}_i), & \text{for } i = 1, \ldots, k \ \wedge \ \xi(\hat{t}_i) \in \bar{\xi} \\ 1.0 - \sum_{\xi \in \Xi(\hat{m}_i) \cap \bar{\xi}} \xi, & \text{for } i = 1, \ldots, k \ \wedge \ \xi(\hat{t}_i) \notin \bar{\xi} \end{cases} \tag{12}$$

Also, the probability for the entire transition sequence $\hat{T}$ is given by:

$$\hat{p}(\hat{T}; \bar{\xi}) = \prod_{i=0}^{k} \hat{p}(\hat{t}_i; \bar{\xi}) \tag{13}$$

From Equations 12 and 13 it is easy to see that $\hat{p}(\hat{T}; \bar{\xi})$ constitutes a polynomial function in $\bar{\xi}$. This function characterizes the probability of transitioning from a tangible marking m to another tangible marking m' through a single intermediate sequence of vanishing markings $\hat{M}$. It is possible, however, that the transition from m to m' can be materialized in $\mathcal{SM}(\mathcal{N})$ through more than one sequence $\hat{M}$. Let all these sequences be denoted by $\hat{M}^1, \ldots, \hat{M}^l$, and furthermore, let $\hat{T}^1, \ldots, \hat{T}^l$ denote the corresponding transition sequences defined according to the above discussion. Then, it should be clear that the total probability of transitioning from a tangible marking m to another marking m' in the uniformized process $\mathcal{U}(\mathcal{N}; \bar{\xi})$ is given by:

$$\hat{p}(m, m'; \bar{\xi}) = \sum_{j=1}^{l} \hat{p}(\hat{T}^j; \bar{\xi}) = \sum_{j=1}^{l} \prod_{i=0}^{k(j)} \hat{p}(\hat{t}_i^j; \bar{\xi}) \tag{14}$$

where each factor $\hat{p}(\hat{t}_i^j; \bar{\xi})$ is defined according to Equation 12. Hence, $\hat{p}(m, m'; \bar{\xi})$ remains a polynomial function in $\bar{\xi}$.

Equation 14 suggests a straightforward algorithm for computing $\hat{p}(m, m'; \bar{\xi})$ while utilizing the information that is encoded in the underlying GSPN $\mathcal{N}$. In particular, for any given transition pair (m, m') in $\mathcal{U}(\mathcal{N}; \bar{\xi})$, we first construct the *acyclic* digraph $\mathcal{G}_v(\mathcal{N}; m)$ which unfolds the

transitional dynamics of GSPN $\mathcal{N}$ when the latter is initialized at marking m and evolves first through the execution of the transitions that are enabled in m and subsequently through the execution of transition sequences that consist of untimed transitions only. Hence, graph $\mathcal{G}_v(\mathcal{N}; m)$ contains (i) marking m as its "source" node, (ii) the set of markings $\tilde{M}$ that can be reached from m through the firing of any transition $t \in \mathcal{E}(m)$, and (iii) all the markings that can be reached from the markings in $\tilde{M}$ by firing untimed transitions only. Furthermore, the "leaf" nodes of $\mathcal{G}_v(\mathcal{N}; m)$ will be a set $\bar{M}$ of tangible markings with $m' \in \bar{M}$.[7] The availability of $\mathcal{G}_v(\mathcal{N}; m)$ subsequently enables the enumeration of all the transition sequences T^j that connect markings m and m' according to the previous discussion, and the computation of $\hat{p}(m, m'; \bar{\xi})$ according to the formula of Equation 14.

The deployment of the digraph $\mathcal{G}_v(\mathcal{N}; m)$, for any tangible marking $m \in \mathcal{R}_t(\mathcal{N})$, is a "local" computation when perceived in the context of the entire reachability space $\mathcal{R}(\mathcal{N})$, and it is generally expected that each such digraph $\mathcal{G}_v(\mathcal{N}; m)$ will be a pretty small graph. This is also suggested from the conceptual interpretation of $\mathcal{G}_v(\mathcal{N}; m)$ as the digraph that encodes the decision sequences that can be effected in response to the events (i.e., the transition firings) that take place in marking m. Hence, the computation of the digraphs $\mathcal{G}_v(\mathcal{N}; m)$, and the extraction of the necessary transition sequences T^j from them, are supposed to be performed on the fly, upon the observation of the corresponding transition (m, m') in the processed sample path $\mathbf{m}$.

Furthermore, the availability of the transition sequences T^j for any pair $m, m' \in \mathcal{R}_t(\mathcal{N})$ enables also the computation of $\frac{\partial}{\partial \xi}\hat{p}(m, m'; \bar{\xi})$ for any $\xi \in \bar{\xi}$. To define an algorithm for this computation, we first notice that it suffices to provide an algorithm for the computation of $\frac{\partial}{\partial \xi}\hat{p}(\hat{T}^j; \bar{\xi})$, for any sequence T^j; once this algorithm is available, Equation 14 implies that $\frac{\partial}{\partial \xi}\hat{p}(m, m'; \bar{\xi})$ can be computed by applying it for every T^j and summing up the obtained results. But an algorithm for the computation of $\frac{\partial}{\partial \xi}\hat{p}(\hat{T}^j; \bar{\xi})$ can be obtained from Equation 13, which implies that

$$\frac{\partial}{\partial \xi}\hat{p}(\hat{T}^j; \bar{\xi}) = \sum_{q=1}^{k(j)} \{[I_{\{\xi(\hat{t}_q^j)=\xi\}} - I_{\{(\xi(\hat{t}_q^j)\notin\bar{\xi})\wedge(\xi\in\Xi(\hat{m}_q)\cap\bar{\xi})\}}] \prod_{i=0;i\neq q}^{k(j)} \hat{p}(\hat{t}_i^j; \bar{\xi})\} \tag{15}$$

In Equation 15, the quantity $I_{\{\cdot\}}$ denotes the indicator variable for the condition that is expressed in the brackets. Hence, in more practical terms, $\frac{\partial}{\partial \xi}\hat{p}(\hat{T}^j; \bar{\xi})$ can be computed as follows: First, $\frac{\partial}{\partial \xi}\hat{p}(\hat{T}^j; \bar{\xi})$ is initialized to zero. Subsequently, the subsequence $\{\hat{t}_q^j : q = 1, \ldots, k(j)\}$ is parsed from left to right, and every time that a transition $\hat{t}_q^j$ is encountered such that the probability $\xi(\hat{t}_q^j)$ is expressed by variable ξ, then the corresponding product $\prod_{i=0;i\neq q}^{k(j)} \hat{p}(\hat{t}_i^j; \bar{\xi})$ is added to the partial sum expressing $\frac{\partial}{\partial \xi}\hat{p}(\hat{T}^j; \bar{\xi})$. On the other hand, if the encountered transition $\hat{t}_q^j$ has a corresponding probability $\xi(\hat{t}_q^j) \notin \bar{\xi}$ but the corresponding random switch $\Xi(\hat{m}_q)$ contains ξ, then the product $\prod_{i=0;i\neq q}^{k(j)} \hat{p}(\hat{t}_i^j; \bar{\xi})$ is subtracted from the partial sum expressing $\frac{\partial}{\partial \xi}\hat{p}(\hat{T}^j; \bar{\xi})$. In all other cases, the computed sum remains unchanged.

The above discussion outlines a complete algorithm for the systematic evaluation of the formula of Equation 11 in the considered problem context, for any chosen sample path $\mathbf{m}$. Additional efficiencies can be built in this algorithm through a more streamlined coding of the involved computation, or, in certain cases, by taking advantage of additional structure that might exist

[7] The reader should also notice that the "unfolding" nature of the construction of $\mathcal{G}_v(\mathcal{N}; m)$ implies that it is possible that $m \in \bar{M}$. In fact, this will be the case for the "fictitious" transitions in $\mathcal{U}(\mathcal{N})$ that are introduced by the uniformization.

in the considered GSPN $\mathcal{N}$ and its reachable state space $\mathcal{R}(\mathcal{N})$; since these efficiencies will be context-specific in general, we leave the relevant details to the reader.

Solving the MP formulation of Equations 6–8 through a stochastic approximation algorithm In this part, we provide an SA algorithm that can be used for the solution of the formulation of Equations 6–8 while employing the estimates for the partial derivatives $\frac{\partial \eta(\bar{\xi})}{\partial \xi}$ that were presented in the previous paragraphs. This algorithm can be perceived as a customization to the considered problem of some more generic results developed in ([21], Section 5.2). In order to facilitate the subsequent discussion, and establish clarity and consistency for the adopted notation, we epitomize these more general results from [21] in a supporting theorem – Theorem 2 – that is provided in the Appendix.

To employ the result of Theorem 2 for the solution of the formulation (6–8), we shall correspond the vector θ to the variable vector $\bar{\xi}$, the function $g(\cdot)$ to the objective function $\eta(\cdot)$ of the considered formulation, and the set H to the feasible region that is defined by Constraints 7–8. Under this correspondence, Y_n should denote an observation of the gradient $\nabla_{\bar{\xi}}\eta(\bar{\xi}_n)$, obtained through the sample-path based approach that was outlined in the previous paragraphs of this section. However, from the broader spirit of gradient-based optimization algorithms, it can be seen that the primary information provided by Y_n is a sense for "direction" for the update performed in the n-th step of the contemplated algorithm. Hence, in the following, we propose to focus on the numerator of Equation 10 that provides the components of the gradient $\nabla_{\bar{\xi}}\eta(\bar{\xi}_n)$, and define Y_n as a vector that collects the sample averages of the corresponding sums $\sum_{k=u_\nu}^{u_{\nu+1}-1}[f(m_k)-\eta(\bar{\xi}_n)]\cdot\hat{r}_k(\xi)$. More specifically, at any given point $\bar{\xi}_n$, the proposed algorithm will first run N_1 replications of the Markov reward process $\mathcal{M}(\mathcal{N};f)$ in order to obtain an estimate of $\eta(\bar{\xi}_n)$, $\hat{\eta}(\bar{\xi}_n)$, and subsequently it will employ this estimate in order to obtain a number of samples for each of the sums $\sum_{k=u_\nu}^{u_{\nu+1}-1}[f(m_k)-\hat{\eta}(\bar{\xi}_n)]\cdot\hat{r}_k(\xi)$ that correspond to the components of Y_n; Y_n is finally obtained by averaging the samples that were collected for each of its components.

The computation of Y_n outlined in the previous paragraph further implies that

$$E_n[Y_n] \equiv E[Y_n|\bar{\xi}_0, Y_i, i<n] = E[u_{\nu+1}-u_\nu]\cdot\nabla_{\bar{\xi}}\eta(\bar{\xi}_n)+\beta_n \tag{16}$$

The first term in the right-hand-side of the above equation is suggested by Equation 10. The second term is a bias that is introduced by the employment of the estimate $\hat{\eta}(\bar{\xi}_n)$ in the computation of Y_n. In fact, following the developments of [26] (c.f. Proposition 2), it can be shown that

$$\beta_n = E\left[\sum_{k=u_\nu+1}^{u_{\nu+1}-1}(u_{\nu+1}-k)\frac{\nabla_{\bar{\xi}}\hat{p}(m_{k-1},m_k;\bar{\xi}_n)}{\hat{p}(m_{k-1},m_k;\bar{\xi}_n)}\right]\Big(\eta(\bar{\xi}_n)-\hat{\eta}(\bar{\xi}_n)\Big) \tag{17}$$

Using Equation 17, it can also be shown that $\beta_n \to 0$ w.p. 1 as long as the number of replications N_1 that are used in the computation of $\hat{\eta}(\bar{\xi}_n)$ keeps increasing with n.[8] Furthermore, it can be checked that the stochastic process $\{Y_n\}$ will satisfy Assumption II in the statement of Theorem 2. More specifically, the satisfaction of Assumption II by $\{Y_n\}$ results from the positive recurrence of the DTMC $\mathcal{U}(\mathcal{N};\bar{\xi}_n)$ and the fact that the terms of the sum $\sum_{k=u_\nu}^{u_{\nu+1}-1}|f(m_k)-\hat{\eta}(\bar{\xi}_n)|\cdot|\hat{r}_k(\xi)|$ can be bounded uniformly in n. In particular, the factors $|f(m_k)-\hat{\eta}(\bar{\xi}_n)|$ can be bounded uniformly in n by the difference of the extreme values of $f(\cdot)$. On the other hand,

[8] However, this increase can take place in an arbitrarily slow rate.

Algorithm 1 Project $\bar{\xi}$ onto the feasible region of formulation (6–8)

Input: the variable vector $\bar{\xi}$ and the set of the random switches $\bar{\Xi}$ currently identified by Algorithm 2, the randomizing parameter δ.

Output: The projected value of $\bar{\xi}$.

for $\Xi \in \bar{\Xi}$ **do**
  $sum \leftarrow 0$.
  for $\xi \in \bar{\xi} \cap \Xi$ **do**
    if $\xi < \delta$ **then**
      $\xi \leftarrow \delta$.
    end if
    $sum \leftarrow sum + \xi$.
  end for
  if $1 - sum < \delta$ **then**
    $y \leftarrow$ vector of elements in $\bar{\xi} \cap \Xi$ in descending order.
    Define mapping $j = Index(i)$ such that y_i stands for $\bar{\xi}_j$.
    for $i = 1 \rightarrow |\bar{\xi} \cap \Xi|$ **do**
      $m \leftarrow \frac{1}{i} \sum_{k=1}^{i} y_k$.
      $residual \leftarrow (1 - |\Xi|\delta)/i$.
      if $m - y_i \leq residual$ and $m - y_{i+1} \geq residual$ **then**
        break.
      end if
    end for
    for $k = 1 \rightarrow i$ **do**
      $\bar{\xi}_{Index(i)} \leftarrow y_i - m + residual + \delta$.
    end for
    for $k = i + 1 \rightarrow |\bar{\xi} \cap \Xi|$ **do**
      $\bar{\xi}_{Index(i)} \leftarrow \delta$.
    end for
  end if
end for
Return $\bar{\xi}$.

the existence of uniform bounds in n for the factors $|\hat{r}_k(\xi)|$ is a consequence of the imposition of the lower bound $\delta > 0$ for the elements of the vector $\bar{\xi}$.

In the proposed algorithm, the step sequence $\{\epsilon_n\}$ will follow the pattern

$$\epsilon_n = \frac{1+a}{n+b}, \quad n \geq 0 \tag{18}$$

where a and b are nonnegative reals. It can be easily checked that the sequence $\{\epsilon_n\}$ satisfies the stipulations of Assumption III in Theorem 2.

On the other hand, the projection operator Π_H w.r.t. the feasible region that is defined by Constraints 7–8 is implemented by the algorithm that is depicted in Algorithm 1. The derivation of this algorithm is provided in [22], and it is based on the standard MP-based characterization of the notion of projection and fairly standard arguments and techniques coming from the area of nonlinear programming [25].

A complete description of the SA algorithm that was outlined in the previous paragraphs

Algorithm 2 The proposed stochastic approximation algorithm

Input: $GSPN\mathcal{N} = (P, T_t, T_u, W, m_0, R, f), \delta, \epsilon_1, a, b, n_{end}, trial, N_1, rep_{inc}, t_{end}, N_2$.
Output: $\bar{\xi},\ \bar{\Xi},\ \hat{\eta}(\bar{\xi})$.

$r_u \leftarrow$ uniformizing rate for $\mathcal{U}(\mathcal{N})$; $\bar{\xi} \leftarrow \emptyset$; $\bar{\Xi} \leftarrow \emptyset$.
for $n = 1 \rightarrow n_{end}$ **do**
  Starting from the initial marking m_0, simulate the GSPN $\mathcal{N}$ for $trial$ transitions to get the most visited tangible marking m^*. Also extend the variable vector $\bar{\xi}$ with any newly encountered random switches Ξ, initialize the corresponding variables ξ to represent a uniform distribution, and add Ξ to the set $\bar{\Xi}$.
  $reward_{total} \leftarrow 0$; $t_{total} \leftarrow 0$.
  for $rep = 1 \rightarrow N_1 + \lfloor \frac{n}{rep_{inc}} \rfloor$ **do**
    $t \leftarrow 0$; $m \leftarrow m^*$; re-initialize the random number generator.
    while $t \leq t_{end}/r_u$ or $m \neq m^*$ **do**
      if $m \in \mathcal{R}_v(\mathcal{N})$ **then**
        Extend $\bar{\xi}$ and $\bar{\Xi}$ if a new random switch is found.
        Transition $\mathcal{N}$ from m according to the corresponding prob. distr. given in $\bar{\xi}$.
      else
        Transition $\mathcal{N}$ from m according to the exp. race defined by $\mathcal{E}_t(m)$. Let $t(m)$ be the corresponding sojourn time at m.
        $reward_{total} \leftarrow reward_{total} + f(m) \cdot t(m)$; $t \leftarrow t + t(m)$.
      end if
    end while
    $t_{total} \leftarrow t_{total} + t$.
  end for
  $\hat{\eta} \leftarrow reward_{total}/t_{total}$. /* This is the throughput estimate */
  $Y_{total} \leftarrow 0$; $cycles_{total} \leftarrow 0$.
  for $rep = 1 \rightarrow N_2$ **do**
    $t \leftarrow 0$; $cycles \leftarrow 0$; $m \leftarrow m^*$; $\hat{r} \leftarrow 0$; re-initialize the random number generator.
    while $t \leq t_{end}$ or $m \neq m^*$ **do**
      $t \leftarrow t + 1$.
      Evolve GSPN $\mathcal{N}$ to its next tangible marking m', and compute the corresponding transition probability $\hat{p}(m, m'; \bar{\xi}_n)$ and its gradient w.r.t. to $\bar{\xi}$ according to Equations 14 and 15. Also, extend $\bar{\xi}$, $\bar{\Xi}$ and Y_{total} if new random switches have been found.
      if $m' = m^*$ **then**
        $cycles \leftarrow cycles + 1$; $\hat{r} \leftarrow 0$.
      else
        Update $\hat{r}$ according to Equation 11; $Y_{total} \leftarrow Y_{total} + (f(m') - \hat{\eta}) \cdot \hat{r}$.
      end if
    end while
    $cycles_{total} \leftarrow cycles_{total} + cycles$.
  end for
  $Y \leftarrow Y_{total}/cycles_{total}$ /* This is an estimate of the numerator of Equation 10 */
  $\bar{\xi} \leftarrow \bar{\xi} + \epsilon_1 \cdot \frac{1+a}{n+b} \cdot Y$.
  Project $\bar{\xi}$ onto feasible region H using Algorithm 1.
end for
Return $(\bar{\xi}, \bar{\Xi}, \hat{\eta})$.

for the MP formulation (6–8), is provided in Algorithm 2. The next theorem establishes that Algorithm 2 will converge to some stationary point(s) of formulation (6–8).

Theorem 1 *The sequence* $\{\bar{\xi}_n\}$ *that is generated from the recursion*

$$\bar{\xi}_{n+1} = \Pi_H[\bar{\xi}_n + \frac{1+a}{n+b} \cdot Y_n], \quad n \geq 0 \tag{19}$$

of Algorithm 2 will converge w.p. 1 to a subset S_i *of the set* S_H *of the stationary points of the MP formulation (6–8), where the latter is defined by*

$$S_H = \{\bar{\xi} : \nabla_{\bar{\xi}}\eta(\bar{\xi}) + z = 0; z \in -C(\bar{\xi})\} \tag{20}$$

and the sets $C(\bar{\xi})$ *are the convex cones that are defined by the outer normals of the active constraints at* $\bar{\xi}$*. Furtthermore, function* $\eta(\cdot)$ *is constant over the set* S_i*.*

Proof: As established in the discussion that precedes Theorem 1, the recursion of Equation 19 satisfies the Assumptions II–IV that underlie the statement of Theorem 2 in the Appendix.

The additional requirement of Theorem 2 that the function $\eta(\cdot)$ is constant on each subset S_i of the set S_H that contains the stationary points of the feasible region of the MP formulation (6–8) can be established by showing that function $\eta(\cdot)$, and also the functions $\psi_i(\cdot)$ that define the feasible region, are twice continuously differentiable [21]. The satisfaction of this condition by the functions $\psi_i(\cdot)$ can be checked immediately from Constraints 7–8. On the other hand, the satisfaction of the aforementioned condition for function $\eta(\cdot)$ can be argued from the facts that (i) the elements of the transition probability matrix $\hat{P}(\bar{\xi})$ are polynomial functions of $\bar{\xi}$, and (ii) the steady-state probability distribution $\pi(\bar{\xi})$ is well-defined over the considered feasible region (c.f. Equation 1).

A minor complication arises from the redefinition of the stochastic process $\{Y_n\}$ as discussed in the previous paragraphs. However, the scaling of the Y_n values that is introduced by this redefinition leaves intact the set of stationary points, S_H, defined by Equation 20, and it can be easily checked that all the arguments that underlie the proof of Theorem 2 carry over to the modified recursion of Equation 19. □

We conclude the discussion of this section by noticing that, while Theorem 1 guarantees the convergence of the proposed algorithm to some stationary point(s) of the considered MP formulation (with stationarity understood in the spirit of Equation 20), it does not guarantee that the obtained solution(s) will be globally optimal over the considered feasible region, or even that these solutions will be maximizers of the objective function, as requested by formulation (6–8). Convergence to a maximizer instead of some other type of stationary point is typically attained by the fact that the noise inherent in the considered algorithm will tend to destabilize it when it gets close to stationary points of the wrong type. On the other hand, the algorithm prospects to converge to some global maxima are typically enhanced by re-running it from different initial points $\bar{\xi}_0$. In the next section we demonstrate the various properties and the practical potential of the proposed algorithm by applying it to the throughput maximization of a capacitated re-entrant line.

6 An example application

In this section we concretize the modeling paradigm and the optimization theory that were introduced in the previous sections through a particular example. As discussed in the introductory section, this example is drawn from the area of the scheduling of complex resource allocation systems, and, for expository purposes, it is kept quite small; but it still exhibits all the salient features of the developed methodology.

The particular system to be considered is the capacitated re-entrant line (CRL) depicted in Figure 1. This line abstracts the operation of a small automated manufacturing cell consisting of two workstations, W_1 and W_2, and supporting the production of a single job type. Jobs requesting processing at this cell are initially routed to workstation W_1 to execute their first processing stage. After being processed by the server of that workstation, the jobs will advance to workstation W_2 to execute their second processing stage. Finally, after being processed at the second workstation, the jobs will return to the first workstation for the execution of a third processing stage, and subsequently they will be unloaded from the system. The job transfer among the workstations and the I/O port of the cell are facilitated by a robotic manipulator. However, in the context of the considered example, we assume that transport times are negligible. Furthermore, the allocation of the robotic manipulator itself to the advancing jobs does not generate any interesting behavioral dynamics as long as the allocation of the buffering capacity of the two workstations is properly handled (c.f. the relevant discussion further below). Therefore, the material handling operations are not modeled explicitly in the considered example.

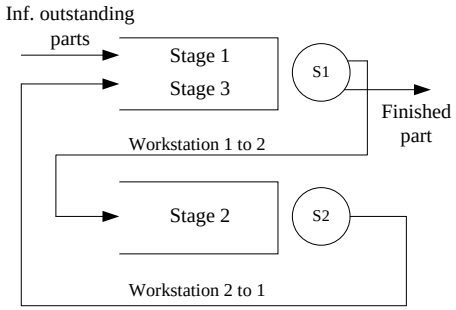

Figure 1: The capacitated re-entrant line used in the example of Section 6.

On the other hand, the considered example assumes that each workstation $W_i, i = 1, 2$, possesses a single server S_i, and that it has only two buffer slots for accommodating the jobs routed to it.[9] It is further assumed that, at each workstation, the processing of a job takes place in situ; i.e., with the server moving to the corresponding buffer slot allocated to the job. Hence, the total buffering capacities for the two workstations are $C_1 = C_2 = 2$. It should further be noticed that the buffer and the server of workstation W_1 are shared by the jobs instances executing processing stages 1 and 3. Finally, the processing times for the three processing stages annotated in Figure 1 are assumed exponentially distributed with means $\tau_j = 1/\mu_j > 0,\ j = 1, 2, 3$.[10]

We want to control the workflow dynamics of the aforementioned CRL in order to maximize its throughput. To address this objective, first we notice that the smooth operation of the considered CRL can be challenged by the finiteness of the buffering capacity of its two workstations in combination with the re-entrant nature of its material flow. In particular, if the buffer of workstation W_1 is allocated to two jobs in their first processing stage and the buffer-

[9] Thus, the characterization "*capacitated* re-entrant line".

[10] As already mentioned, more general distributions can be effectively approximated by phase-type distributions [10]. Hence, the restriction of this example to exponentially distributed processing times does not compromise its generality. We also notice that the cluster tools that are used in the contemporary semiconductor manufacturing industry are miniaturized versions of the robotic cells that are considered in this example [40, 41].

ing capacity of workstation W_2 is also fully allocated, then it is impossible for the underlying robotic manipulator to advance any of these jobs to their next stage, and the operation of the system will be permanently stalled, or *deadlocked*. As discussed in Section 2, the prevention of deadlock arising in complex RAS can be a challenging task, and it has been extensively studied in the DES literature. In the operational context of the considered cell, maximally permissive liveness-enforcing supervision can be effectively established through the following requirement: the combined number of jobs executing their first and their second processing stages must remain less than or equal to three.

A GSPN modeling the operational dynamics of the CRL of Figure 1, under the aforementioned SCP, is depicted in Figure 2. In the depicted net, the straight vertical path starting with transition $T_{1a=1l}$ and finishing with transition $T_{3p=3d}$ models the processing sequence of the single process type for this CRL. More specifically, tokens in the places labelled P_{ki}, $k = 2, 3$, model jobs that wait for the execution of their k-th processing stage while holding a buffer slot at the corresponding workstation. On the other hand, since the considered example assumes zero transport times, it is pertinent to assume that jobs will be loaded in workstation W_1 only when they have secured the station server, and therefore, there is no place P_{1i} in the considered GSPN model. Tokens in places P_{kp}, $k = 1, 2, 3$, model jobs executing their k-th processing stage, and tokens in places P_{ko}, $k = 1, 2$, model jobs that have completed the execution of their k-th processing stage and they are awaiting for their transport to the next station. Again, the assumption of zero transport times implies that jobs having completed their last processing stage can immediately exit the system, and therefore, there is no place P_{3o} in the considered model.

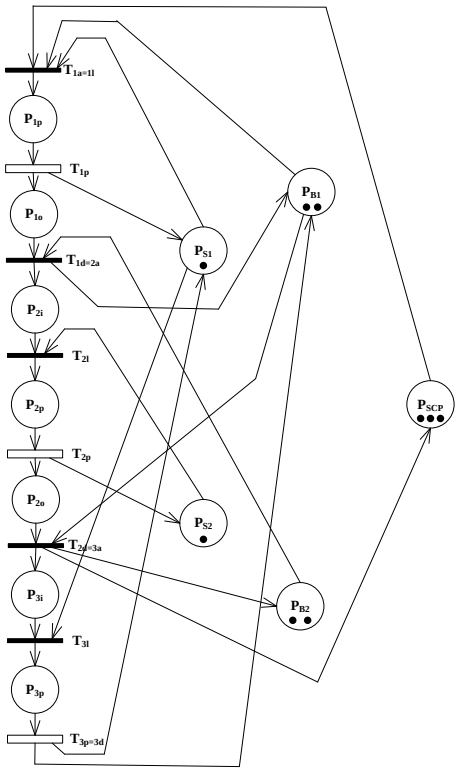

Figure 2: The GSPN modeling the CRL of Figure 1 under the control of the maximally permissive SCP.

Places P_{S1}, P_{S2}, P_{B1}, and P_{B2} model respectively the availability of the servers of the two workstations W_1 and W_2, and the buffering capacity of the same stations. The depicted connectivity of these places to the transitions of the aforementioned path that defines the process type, expresses the resource allocation that is associated with each processing stage $j, j = 1, 2, 3$, and defines the meaning of the events that correspond to these transitions. Hence, for instance, transition $T_{1a=1l}$ denotes the action of loading a new job from outside into workstation W_1 and the allocation of a buffer slot and of the server capacity of that workstation to this new job. On the other hand, transition T_{1p} denotes the completion of the processing at workstation W_1 of a job in stage 1 and the release of the station server. It is also important to notice that among the transitions that define the considered process, the timed ones are those that correspond to the completion of a processing stage. These transitions are depicted by white bars in Figure 2, and the exponential distributions that control their firing are those specified in the opening part of this section. The remaining transitions, depicted by black bars, essentially model the underlying resource allocation decisions and they are untimed transitions.

Place P_{SCP} with its initial marking of three tokens enforces the inequality $m(P_{1p}) + m(P_{1o}) + m(P_{2i}) + m(P_{2p}) + m(P_{2o}) \leq 3$, that expresses the aforementioned liveness-enforcing supervisor for the considered cell. This place enforces the aforementioned inequality by acting as a fictitious resource with a capacity of three units.[11]

The above discussion on the GSPN-based modeling of the operations of the considered CRL, and also of the logic of the corresponding maximally permissive LES, substantiates our earlier claims about the ability of this modeling framework to provide an integrative representation of the behavioral dynamics of the underlying system and of the supervisory control logic that might be necessary in order to conform these dynamics to certain behavioral specifications. On the other hand, Figure 3 depicts the structure of the state transition diagram (STD) of the semi-Markov process that corresponds to the GSPN of Figure 2, revealing, thus, the ability of the GSPN-based modeling framework to provide also an effective representation of the timed dynamics and the scheduling decisions involved in the underlying resource allocation process. Double-circled states in the depicted STD correspond to tangible markings and single-circled to vanishing ones. A complete characterization of these markings is provided in Table 1.[12]

In the STD depicted in Figure 3, chains of consecutive vanishing markings correspond to the advancement of an entire set of jobs that had been blocked by the applied SCP. Furthermore, recognizing the fact that, in the GSPN of Figure 2, the objective of maximizing the throughput of the CRL of Figure 1 is equivalent to the maximization of the average firing rate of transition $T_{3p=3d}$, and that vanishing markings do not contribute to the collection of any reward, the original STD of Figure 3 can be simplified by removing its segments that are depicted in dashed lines. All these segments correspond to transitions among the net tangible markings that can also be materialized through the remaining paths that are depicted in solid line. From a computational standpoint, the identification of this last effect and the corresponding simplifications can be performed through some local structural analysis of the digraphs $\mathcal{G}_v(\mathcal{N}; m)$ that model the untimed dynamics that emanate from any given tangible marking m, and therefore, they can be easily supported by the sample-path-based computation that was discussed in the previous section. We also notice that, in the general case, additional simplifications of the underlying STD can be effected even by the elimination of certain resource allocation options from some vanishing markings that can be shown to be suboptimal through some structural analysis of similar localized nature. It is evident from the presented example that the considered simplifications can reduce substantially the complexity of the underlying decision-making process, while retaining a very structured perspective for the involved system dynamics and of the impact of the attempted simplifications upon these dynamics.

In the simplified STD of Figure 3 there are only seven undefined random switches, corresponding to the vanishing markings that are depicted in color (i.e., the markings numbered 12, 18, 21, 26, 33, 47 and 57). Furthermore, under the aggregation criterion of Equation 5, vanishing

[11] More formally, from the standpoint of the PN-based supervisory control theory, P_{SCP} is a "monitor" place that enforces the aforementioned inequality by introducing an appropriate p-semiflow in the net dynamics [14]. The theory of imposing constraints on the PN behavior that are expressed as linear inequalities in the net marking, is pretty well developed [28, 17], and in the particular context of the liveness-enforcing supervision of complex RAS, it can support the implementation of the maximally permissive LES or of some pretty tight approximations of that policy [37, 34].

[12] We should also notice, for completeness, that the optimal scheduling policy for capacitated re-entrant lines might involve deliberate idleness. The possibility of modeling such deliberate idleness in the GSPN framework has been demonstrated in [11]. In this work, we have opted to ignore this behavioral element in order to maintain the presentational tractability of the considered example. Furthermore, it can be verified that for the considered example, the optimal scheduling policy does not involve any deliberate idleness.

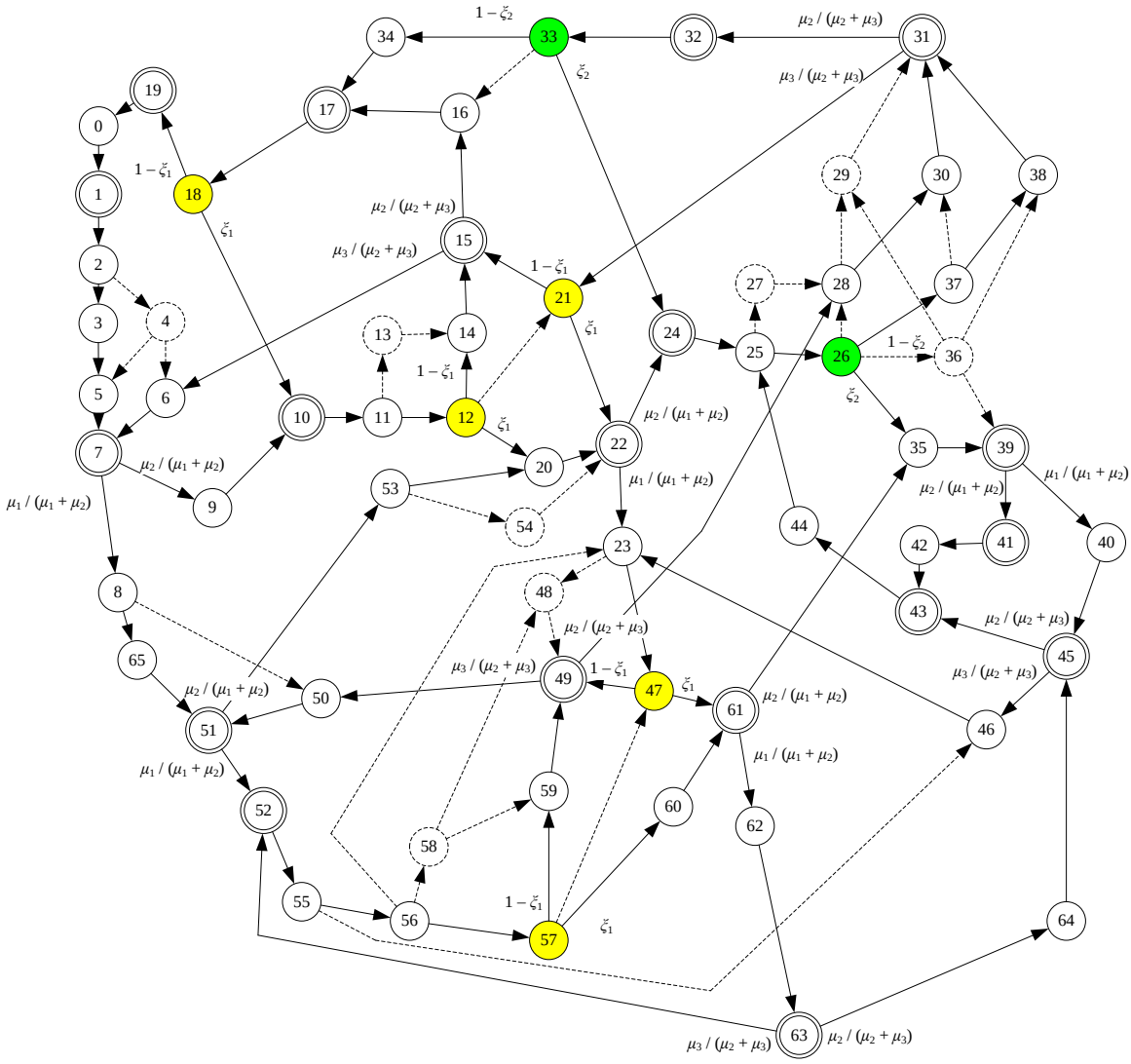

Figure 3: The semi-Markov process for the GSPN of Figure 2.

markings 12, 18, 21, 47 and 57, colored in yellow, are regulated by a common random switch Ξ_1, that is defined by the set of enabled untimed transitions $\mathcal{E}_u^1 = \{T_{1a=1l}, T_{3l}\}$; and vanishing markings 26 and 33, colored in green, are regulated by another common random switch Ξ_2, that is defined by the set of enabled untimed transitions $\mathcal{E}_u^2 = \{T_{1a=1l}, T_{2d=3a}\}$. In the spirit of the revised optimization problem introduced in Section 4, the two undefined random switches Ξ_1 and Ξ_2 can be respectively modeled by two decision variables, ξ_1 and ξ_2, that express the firing probability for transition $T_{1a=1l}$ in each of these two random switches. From a conceptual standpoint, random switch Ξ_1 models the dilemma between loading and processing a new job in workstation W_1 while another job in stage 3 is also waiting to be processed in workstation W_1. On the other hand, random switch Ξ_2 models the dilemma concerning the allocation of a single free buffer-slot in workstation W_1 to a newly loaded job versus the allocation of this slot to a job that has completed its second processing stage in workstation W_2.

The pricing of the decision variables ξ_1 and ξ_2 in a way that maximizes the CRL throughput

Table 1: The marking description for the STD of Figure 3.

s_k	$P_{1p}P_{1o}$	$P_{2i}P_{2p}P_{2o}$	$P_{3i}P_{3p}$	$P_{S_1}P_{S_2}$	$P_{B_1}P_{B_2}P_{SCP}$
0	0 0	0 0 0	0 0	1 1	2 2 3
1	1 0	0 0 0	0 0	0 1	1 2 2
2	0 1	0 0 0	0 0	1 1	1 2 2
3	1 1	0 0 0	0 0	0 1	0 2 1
4	0 0	1 0 0	0 0	1 1	2 1 2
5	1 0	1 0 0	0 0	0 1	1 1 1
6	0 0	0 1 0	0 0	1 0	2 1 2
7	1 0	0 1 0	0 0	0 0	1 1 1
8	0 1	0 1 0	0 0	1 0	1 1 1
9	1 0	0 0 1	0 0	0 1	1 1 1
10	1 0	0 0 0	1 0	0 1	0 2 2
11	0 1	0 0 0	1 0	1 1	0 2 2
12	0 0	1 0 0	1 0	1 1	1 1 2
13	0 1	0 0 0	0 1	0 1	0 2 2
14	0 0	1 0 0	0 1	0 1	1 1 2
15	0 0	0 1 0	0 1	0 0	1 1 2
16	0 0	0 0 1	0 1	0 1	1 1 2
17	0 0	0 0 0	1 1	0 1	0 2 3
18	0 0	0 0 0	1 0	1 1	1 2 3
19	0 0	0 0 0	0 1	0 1	1 2 3
20	1 0	1 0 0	1 0	0 1	0 1 1
21	0 0	0 1 0	1 0	1 0	1 1 2
22	1 0	0 1 0	1 0	0 0	0 1 1
23	0 1	0 1 0	1 0	1 0	0 1 1
24	1 0	0 0 1	1 0	0 1	0 1 1
25	0 1	0 0 1	1 0	1 1	0 1 1
26	0 0	1 0 1	1 0	1 1	1 0 1
27	0 1	0 0 1	0 1	0 1	0 1 1
28	0 0	1 0 1	0 1	0 1	1 0 1
29	0 0	0 1 1	0 1	0 0	1 0 1
30	0 0	1 0 0	1 1	0 1	0 1 2
31	0 0	0 1 0	1 1	0 0	0 1 2
32	0 0	0 0 1	1 1	0 1	0 1 2
33	0 0	0 0 1	1 0	1 1	1 1 2

Table 1: The marking description for the STD of Figure 3 (cont).

s_k	$P_{1p}P_{1o}$	$P_{2i}P_{2p}P_{2o}$	$P_{3i}P_{3p}$	$P_{S_1}P_{S_2}$	$P_{B_1}P_{B_2}P_{SCP}$
34	0 0	0 0 0	2 0	1 1	0 2 3
35	1 0	1 0 1	1 0	0 1	0 0 0
36	0 0	0 1 1	1 0	1 0	1 0 1
37	0 0	1 0 0	2 0	1 1	0 1 2
38	0 0	0 1 0	2 0	1 0	0 1 2
39	1 0	0 1 1	1 0	0 0	0 0 0
40	0 1	0 1 1	1 0	1 0	0 0 0
41	1 0	0 0 2	1 0	0 1	0 0 0
42	0 1	0 0 2	1 0	1 1	0 0 0
43	0 1	0 0 2	0 1	0 1	0 0 0
44	0 1	0 0 2	0 0	1 1	1 0 0
45	0 1	0 1 1	0 1	0 0	0 0 0
46	0 1	0 1 1	0 0	1 0	1 0 0
47	0 0	1 1 0	1 0	1 0	1 0 1
48	0 1	0 1 0	0 1	0 0	0 1 1
49	0 0	1 1 0	0 1	0 0	1 0 1
50	0 0	1 1 0	0 0	1 0	2 0 1
51	1 0	1 1 0	0 0	0 0	1 0 0
52	0 1	1 1 0	0 0	1 0	1 0 0
53	1 0	1 0 1	0 0	0 1	1 0 0
54	1 0	0 1 1	0 0	0 0	1 0 0
55	0 1	1 0 1	0 0	1 1	1 0 0
56	0 1	1 0 0	1 0	1 1	0 1 1
57	0 0	2 0 0	1 0	1 1	1 0 1
58	0 1	1 0 0	0 1	0 1	0 1 1
59	0 0	2 0 0	0 1	0 1	1 0 1
60	1 0	2 0 0	1 0	0 1	0 0 0
61	1 0	1 1 0	1 0	0 0	0 0 0
62	0 1	1 1 0	1 0	1 0	0 0 0
63	0 1	1 1 0	0 1	0 0	0 0 0
64	0 1	1 0 1	0 1	0 1	0 0 0
65	1 1	0 1 0	0 0	0 0	0 1 0

can be addressed through the opimization theory that was developed in the earlier parts of this manuscript. Figure 4 plots in cyan and green the exact CRL throughput that is obtained for all possible choices of the vector $\bar{\xi} = (\xi_1, \xi_2)$, when the timing parameters of the considered CRL are set to $\tau_1 = \tau_2 = \tau_3 = 1.0$. For the considered example, the maximum throughput is equal to 0.480 and it is attained at $\bar{\xi} = (1, 1)$. The green area in the provided plot annotates the region where the attained throughput is within a distance of less than 0.001 from the optimal value of 0.480; i.e., the objective value of the solutions in the green area is greater than 0.479. This area is perceived as particularly difficult for the dynamics of the applied stochastic approximation algorithm, since it is pretty flat, and thus, it is hard for the considered algorithm to find good improvement directions due to the non-negligible error in the throughput and the gradient estimations.

In the implementation of the MP formulation (6–8) for this example, the randomizing parameter δ was set to 0.005, and the initial values of the stochastic approximation algorithm for ξ_1 and ξ_2 were set to 0.5. The step sizes ϵ_n were computed from the recipe

$$\epsilon_n = \epsilon_1 \frac{1+o}{n+o}, \quad n = 1, 2, \ldots$$

where the parameter ϵ_1 determines the initial step size, and the parameter o controls the reduction rate.

The path depicted in thick brown line in Figure 4 is the trajectory taken by the considered algorithm when using the exact gradient value at each step, so that the employed direction is always correct. Also, in this implementation, the employed step sizes are kept pretty small, so that the generated path is very smooth. This path defines a "reference point" for understanding the algorithm behavior and its performance under more realistic implementations that rely on sample-path-based estimates of the throughput and its gradient w.r.t. the variable vector $\bar{\xi}$.

The algorithm behavior under some of these more realistic implementations is visualized by the lines that are depicted in blue, red and black in Figure 4. Each of these lines visualizes the execution of the considered stochastic approximation algorithm for 1,000 steps, and they use the same seed for the pseudo-random number generator that drives the underlying simulations. Furthermore, in each case, we set the initial number of replications used for the estimation of the throughput $\eta(\bar{\xi_n})$ to $N_1 = 10$, and we increased this number by 1 every one hundred steps (i.e., $rep_{inc} = 100$). The number of replications N_2 used for the estimation of the gradient $\nabla_{\bar{\xi}}(\bar{\xi_n})$ was set to $N_2 = 3$. All the aforementioned replications were ran for $t_{end} = 100,000$ steps.[13] The blue path was obtained by setting $\epsilon_1 = 1; o = 1$ in the formula that generates the step sequence $\{\epsilon_n\}$, which stood for a conservative exploration, As it can be seen in Figure 4, this path was the slowest but most accurate. More specifically, we can see from Figure 4 that this path reached the difficult green area at a very late stage, and it could not get to the optimal solution point at the right-upper corner of that region in 1,000 steps. The red path was obtained by setting $\epsilon_1 = 3; o = 10$, which stood for moderate step sizes. We can see from Figure 4, and its zoom-in that is provided in Figure 5, that this path reached the optimal point and subsequently it kept oscillating around it. The black path was obtained by setting $\epsilon_1 = 5; o = 50$, which stood for an aggressive exploration. This path was the fastest to reach the optimal solution, but it also presents the oscillation with the greater amplitude, as shown in Figure 5.

[13] We opted for a fairly large N_1, especially when compared to N_2, since (i) the analysis of Section 5 has revealed the criticality of the size of N_1 for controlling the bias in the observations Y_n w.r.t. the actual direction of the gradient $\nabla_{\bar{\xi}}(\bar{\xi_n})$, and furthermore, (ii) the computational cost of the replications that provide the throughput estimates are significantly lower than the computational cost of the replications that provide the Y_n-values.

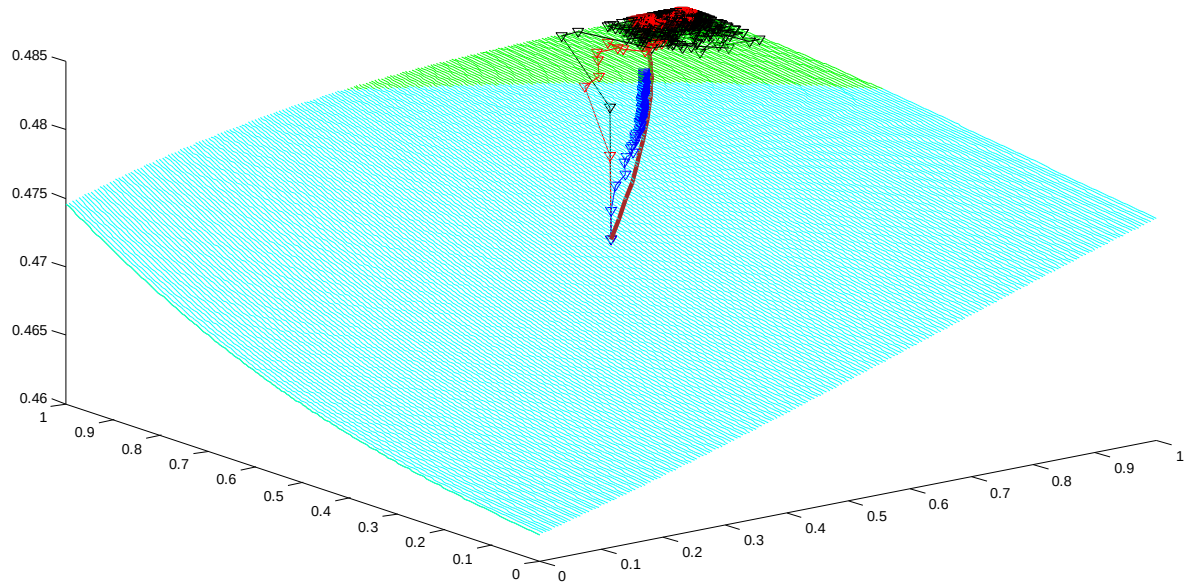

Figure 4: A visualization of the behavior of Algorithm 2 when applied to the performance optimization of the GSPN of Figure 2.

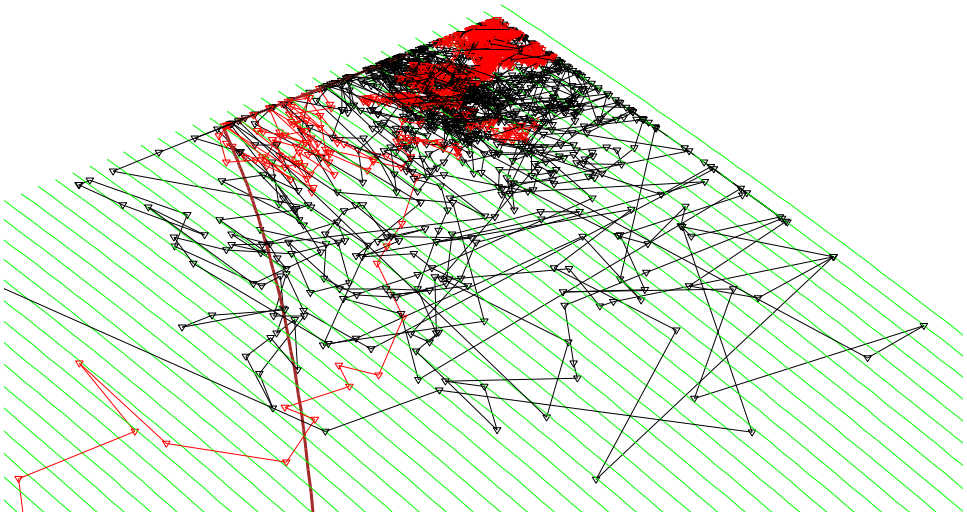

Figure 5: A zoom-in of Figure 4 at the area near the optimal point; this figure reflects the fact that $\delta = 0.005$ in the considered formulation.

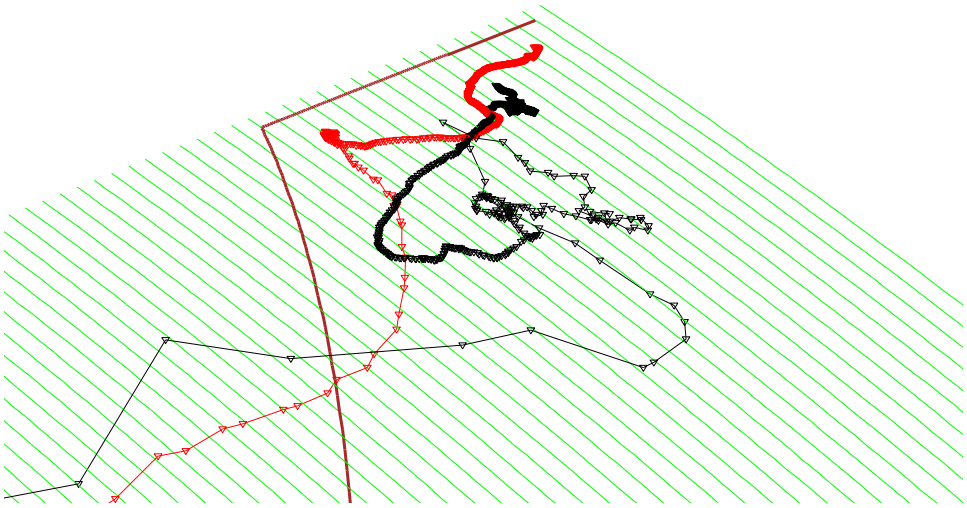

Figure 6: The behavior of the red and the black paths of Figure 4 at the area near the optimal point under Polyak averaging; as in the case of Figure 5, this figure reflects the fact that $\delta = 0.005$ in the considered formulation.

The behavior of Algorithm 2 that is demonstrated in Figures 4 and 5, and further detailed in the previous paragraph, is consistent with the more general understanding of the dynamics that are demonstrated by the considered stochastic approximation algorithms. It also provides a measure for the efficacy of the proposed algorithm in the considered application context, and it suggests some guidelines for selecting the algorithm parameters in order to control the dynamics that govern its empirical convergence. In general, smaller and more cautious steps are expected to effect a stabler convergence towards the target limit-points of the algorithm but at the cost of a slow convergence rate. Larger step sizes can reach the neighborhoods of the target points faster, but through a more "noisy" transient phase and with extensive oscillatory behavior once the target neighborhood is reached. The noise and the oscillatory behavior in the latter case can be further controlled through the aplication of additional averaging schemes on the originally generated paths that are known as "Polyak averaging"; we refer to ([21], Section 1.3.3) for some introduction of these averaging schemes. Figure 6 demonstrates the application of Polyak averaging on the considered example.

Finally, we notice, for completeness, that in the case of the considered example, the optimal policy that was derived for the revised optimization problem, defined by the two static random switches Ξ_1 and Ξ_2 described in the previous paragraphs, is also the optimal policy for the problem formulation that employs a separate random switch for each vanishing marking of the STD of Figure 3; i.e., in this case, the aggregation that is introduced by the static nature of the considered random switches does not compromise the quality of the derived policy.

7 Conclusion

Motivated by the current scheduling needs of complex RAS, this paper introduced a novel methodology for performance optimization in DES applications that evolve over very large and complex state spaces and the main objective is the long-run maximization of some reward rate. By leveraging the modeling powers of the GSPN modeling framework, the presented

methodology enables the seamless integration of the logical and performance-oriented control of the aforementioned applications, while at the same time, it uses fundamental concepts from this framework in order to define a pertinent policy space and to (re-)cast the performance optimization problem into an MP formulation. This formulation is eventually solved through results pertaining to the sensitivity analysis of Markov reward processes and SA theory.

An important attribute of the proposed methodology is that it facilitates an explicit control of the trade-off that exists between the computational tractability of the employed formulation and the performance of the derived policies. Furthermore, by posing the eventually defined problem as a constrained nonlinear programming formulation, the presented methodology inherits all the analytical tools and insights that are offered by this vast area of optimization theory. In the current manuscript, all these possibilities were demonstrated through the application of the proposed approach to the throughput maximization of a capacitated re-entrant line abstracting the operation of an automated manufacturing cell.

Our future work will seek to further develop the presented methodology and substantiate its potential. Hence, on the more theoretical side, we need to better understand the structure of the objective function $\eta(\bar{\xi})$ that is implied by the policy-defining constraints and the state space aggregation induced by them. This will also help us understand the nature and the density of the local optima that might appear in this function.

Another interesting problem is the potential replacement of the constraints of Equation 5 with constraints that express other policy classes of interest, or with constraints that define a refinement of the state-space aggregation induced by Equation 5, along the lines that were discussed in Section 4. This last task must be further supported by the development of a search process that will seek an optimized refinement of the policy-defining logic while retaining the computational tractability of the resulting MP formulation. As suggested in Section 4, this search process can be based on ideas and techniques borrowed from combinatorial optimization theory.

The substitution of the stochastic approximation algorithm that is employed in the current work by another similar algorithm that would be based on a single sample-path, like those presented in [26, 18], is another issue that needs a systematic investigation. Besides the apparent computational gains, the development of such an algorithm would also render the proposed methodology more amenable to on-line / real-time implementation.

When considered in the context of the complex RAS scheduling problems that motivated this research in the first place, the presented methodology must be further extended in order to accommodate "fairness" constraints, i.e., constraints that regulate the ratios of completed instances among the various process types. These constraints are essentially additional logical requirements that must be enforced upon the behavior of the underlying RAS, and they must be addressed by extending the existing RAS SC theory.

Finally, from a more practical standpoint, the results that will be obtained from all the aforementioned developments must be applied on larger problem instances in various application areas that are amenable to the presented methodology, and benchmarked against the performance that is attained by the current "best practices" in those areas.

A Appendix: The generic SA algorithm that is employed in this work

Consider the recursion

$$\theta_{n+1} = \Pi_H[\theta_n + \epsilon_n Y_n], \quad n \geq 0 \tag{21}$$

that is further specified by the following assumptions:

I. $\{Y_n\}$ is a stochastic process such that $E_n[Y_n] \equiv E[Y_n|\theta_0, Y_i, i < n] = \nabla_\theta g(\theta_n) + \beta_n$ where $g(\cdot)$ is a continuously differentiable real-valued function, and $\beta_n \to 0$ w.p. 1.

II. $\sup_n E[|Y_n|^2] < \infty$.

III. $\epsilon_n \geq 0;\ \epsilon_n \to 0;\ \sum_{n=0}^{\infty} \epsilon_n = \infty;\ \sum_{n=0}^{\infty} \epsilon_n^2 < \infty$.

IV. $H \equiv \{\theta : \psi_i(\theta) \leq 0, i = 1, \ldots, c\}$, where $\psi_i(\cdot),\ i = 1, \ldots, c$, are continuously differentiable real-valued functions. It is further assumed that $\forall i,\ \nabla_\theta \psi_i(\theta) \neq 0$ if $\psi_i(\theta) = 0$. Furthermore, the set H is assumed to be connected, compact and non-empty. Finally, the operator $\Pi_H[\cdot]$ projects any given θ upon the set H, i.e., it returns a point $\theta' \in H$ such that $\theta' \in \arg\min\{|\theta - \theta''|^2 :\ \theta'' \in H\}$ and the considered norm is the Euclidean norm.

It is clear from the above description that the sequence $\{\theta_n\}$ generated by the recursion of Equation 21 is constrained to evolve in H. For any point $\theta \in H$, let $A(\theta) = \{i : \psi_i(\theta) = 0\}$. The set $A(\theta)$ is the set of the active constraints at θ. Furthermore, consider the set $\Omega(\theta) = \{\nabla_\theta \psi_i(\theta) : i \in A(\theta)\}$, i.e., the set containing the outward normals of the active constraints at θ, and let $C(\theta)$ denote the convex cone that is generated by the elements of $\Omega(\theta)$. For the particular case where $A(\theta) = \emptyset$, $C(\theta)$ will contain only the zero element. On the other hand, for $A(\theta) \neq \emptyset$, and under the further assumption that the elements of $\Omega(\theta)$ are linearly independent, the recursion of Equation 21 can be rewritten as follows:

$$\theta_{n+1} = \theta_n + \epsilon_n Y_n + \epsilon_n Z_n, \quad n \geq 0 \tag{22}$$

where Z_n is a vector belonging in $-C(\theta_{n+1})$. Finally, let

$$S_H = \{\theta : \nabla_\theta g(\theta) + z = 0; z \in -C(\theta)\} \tag{23}$$

We shall refer to S_H as the (set containing the) "stationary" points of H. S_H can be divided into disjoint compact and connected subsets $S_j,\ j = 0, \ldots$ The next theorem characterizes the limiting behavior of the recursion of Equation 21 under the further structure that is defined in the above discussion.

Theorem 2 *[21] Consider the recursion of Equation 21 under Assumptions I-IV, and further suppose that the underlying function $g(\cdot)$ is constant on each subset S_i of the set S_H. Then, the sequence $\{\theta_n\}$ that is generated by this recursion converges to a unique subset S_i w.p. 1.*

Acknowledgement

This work was partially supported by NSF grant CMMI-0928231.

References

[1] M. Ajmone Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis. *Modeling with Generalized Stochastic Petri Nets*. John Wiley & Sons, 1994.

[2] M. Ajmone Marsan, G. Conte, and G. Balbo. A class of generalized stochastic Petri nets for the performance evaluation of multiprocessor systems. *ACM Trans. Comput. Sys.*, 2:93–122, 1984.

[3] L. de Alfaro. *Formal Verification of Probabilistic Systems*. PhD thesis, Stanford University, Palo Alto, VA, 1997.

[4] M. Beccuti, G. Franceschinis, and S. Haddad. Markov Decision Petri Net and Markov Decision Well-Formed Net formalisms. In *ICATPN 2007 – LNCS 4546*, pages 43–62, 2007.

[5] R. Bellman. *Applied Dynamic Programming*. Princeton University, Princeton, N. J., 1957.

[6] D. P. Bertsekas. *Dynamic Programming and Optimal Control (3rd ed.)*. Athena Scientific, Belmont, MA, 2005.

[7] D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, Belmont, MA, 1996.

[8] X.-R. Cao. *Realization Probabilities: The dynamics of queueing systems*. Springer-Verlag, NY,NY, 1994.

[9] X.-R. Cao. *Stochastic Learning and Optimization*. Springer, NY,NY, 2007.

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

[11] 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.

[12] G. Ciardo, J. K. Muppala, and K. S. Trivedi. On the solution of GSPN reward models. *Performance Evaluation*, 12:237–253, 1991.

[13] R. Cordone, A. Nazeem, L. Piroddi, and S. Reveliotis. Designing optimal deadlock avoidance policies for sequential resource allocation systems through classification theory: existence results and customized algorithms. *IEEE Trans. Autom. Control*, 58:2772–2787, 2013.

[14] A. Giua, F. DiCesare, and M. Silva. Generalized mutual exclusion constraints on nets with uncontrollable transitions. In *Proceedings of the 1992 IEEE Intl. Conference on Systems, Man and Cybernetics*, pages 974–979. IEEE, 1992.

[15] A. Gosavi. *Simulation-based Optimization: Parametric Optimization Techniques and Reinforcement Learning*. Kluwer Academic Publishers, Norwell, MA, 2003.

[16] Y. C. Ho and X.-R. Cao. *Perturbation Analysis of Discrete Event Systems*. Kluwer Academic Publishers, Boston, MA, 1991.

[17] M. V. Iordache and P. J. Antsaklis. *Supervisory Control of Concurrent Systems: A Petri net structural approach*. Birkhäuser, Boston, MA, 2006.

[18] V. R. Konda and J. N. Tsitsiklis. On actor-critic algorithms. *SIAM Jrnl on Control & Optimization*, 42:1143–1166, 2003.

[19] P. R. Kumar. Scheduling manufacturing systems of re-entrant lines. In D. D. Yao, editor, *Stochastic Modeling and Analysis of Manufacturing Systems*, pages 325–360. Springer-Verlag, 1994.

[20] P. R. Kumar. Scheduling semiconductor manufacturing plants. *IEEE Control Systems Magazine*, 14–6:33–40, 1994.

[21] H. J. Kushner and G. G. Yin. *Stochastic Approximation and Recursive Algorithms and Applications*. Springer, NY, NY, 2003.

[22] R. Li. *Optimized Scheduling of Complex Resource Allocation Systems*. PhD thesis, ISyE, Georgia Tech, Atlanta, GA (in preparation).

[23] R. Li and S. Reveliotis. Performance optimization for a class of Generalized Stochastic Petri Nets. In *Proc. of CDC 2013*. IEEE, 2013.

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

[25] D. G. Luenberger. *Linear and Nonlinear Programming (2nd ed.)*. Addison Wesley, Reading, MA, 1984.

[26] P. Marbach and J. N. Tsitsiklis. Simulation-based optimization of Markov reward processes. *IEEE Trans. on Automatic Control*, 46:191–209, 2001.

[27] R. Moazzez Estanjini, K. Li, and I. Ch. Paschalidis. A least squares temporal difference actor-critic algorithm with applications to warehouse management. *Naval Research Logistics*, 59:197–211, 2012.

[28] J. O. Moody and P. J. Antsaklis. *Supervisory Control of Discrete Event Systems using Petri nets*. Kluwer Academic Pub., Boston, MA, 1998.

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

[30] C. H. Papadimitriou and J. N. Tsitsiklis. The complexity of optimal queueing network control. *Math. of OR*, 24:293–305, 1999.

[31] W. B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. Wiley, N.Y., N.Y, 2007.

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

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

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

[35] S. Reveliotis and A. Nazeem. Deadlock avoidance policies for automated manufacturing systems using finite state automata. In J. Campos, C. Seatzu, and X. Xie, editors, *Formal Methods in Manufacturing*. CRC Press / Taylor & Francis, 2014.

[36] S. A. Reveliotis. The destabilizing effect of blocking due to finite buffering capacity in multi-class queueing networks. *IEEE Trans. on Autom. Control*, 45:585–588, 2000.

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

[38] S. M. Ross. *Stochastic Processes*. Wiley, N.Y., 1983.

[39] R. Y. Rubinstein and A. Shapiro. *Discrete Event Systems: Sensitivity Analysis and Stochastic Optimization by the Score Function Method*. Wiley and Sons, NY, 1993.

[40] P. Singer. The driving forces in cluster tool development. *Semiconductor International*, July '95:113–118, 1995.

[41] M. Weiss. Semiconductor factory automation. *Solid State Technology*, pages 89–96, 1996.

[42] W. M. Wonham. Supervisory control of discrete event systems. Technical Report ECE 1636F / 1637S 2006-07, Electrical & Computer Eng., University of Toronto, 2006.