Designing parsimonious scheduling policies for complex resource allocation systems through concurrency theory*

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

Abstract

In a recent work we have proposed a theoretical framework for developing optimized scheduling policies for complex resource allocation systems (RAS). This framework relies heavily on the expression of the RAS dynamics in the modeling framework of the Generalized Stochastic Petri Nets (GSPNs), and the employment of this GSPN-based representation towards the establishment of a systematic trade-off between the representational economy of the target scheduling policies and their operational efficiency. In this paper, we enhance the representational economy of the target policies in the aforementioned framework by taking advantage of some notions of "(non-)conflict" in the transitional dynamics of the underlying RAS-modeling GSPNs. A series of numerical experiments demonstrate that the representational gains resulting from the presented methodology can be very substantial.

1 Introduction

The problem of allocating a finite set of reusable resources to a set of concurrently executing processes, in a way that each process is able to secure the requested resources and complete its execution in a smooth and expedient manner, is a fundamental problem that arises in many contemporary applications. In our past research program, this problem has been investigated through the formal abstraction of the "(sequential) resource allocation system (RAS)", and the corresponding dynamics have been modeled and analyzed through a set of modeling frameworks and analytical tools that are provided by the broader area of Discrete Event Systems (DES) [16].

More specifically, in [16], a DES-based controller for the aforementioned resource allocation problem is structured as a two-tiered function, with the first tier seeking the restriction of the underlying RAS behavior in order to prevent the formation of RAS deadlock and ensure the ability of all the activated processes to reach their completion, and the second tier trying to further "bias" the behavior that is determined by the first-tier control function in order to address certain performance considerations. In more DES-theoretic terms, the first-tier control function outlined above is essentially a supervisory controller that seeks to elicit from the feasible behavior of the considered RAS an admissible non-blocking behavior, while the second-tier

*This work was partially supported by NSF grants CMMI-0928231 and ECCS-1405156.

control function is essentially a scheduler that seeks to attend to typical performance objectives like throughput maximization, delay minimization for the constituent processes, etc.

Furthermore, the perusal of the relevant literature reveals that, in the past two decades, the problem of nonblocking supervision for the considered RAS classes has received extensive attention, and currently there is a rich set of results able to provide effective and, in many cases, maximally permissive nonblocking supervision for RAS instances of pretty large size and complex behavior. On the other hand, the currently existing results for the corresponding scheduling problem are quite limited. More specifically, this scheduling problem has been formally characterized, at considerable generality, by means of the classical theory of Markov Decision Processes (MDPs) [3, 16], but this characterization is of limited practical value since the resulting formulations suffer from the notorious "curse of dimensionality". In fact, the explosive size of the underlying state spaces challenges not only the computation of an optimized scheduling policy, but even the efficient representation of any given such policy, since, in the standard MDP framework, each policy is defined by an action selection probability distribution for each state of the underlying state space.

Hence, motivated by these practical limitations, in a recent work [12] we have proposed the restriction of the RAS scheduling problem in a particular policy space that admits a more parsimonious representation of the constituent policies. This policy space is defined through the formal representation of the timed RAS dynamics in the modeling framework of Generalized Stochastic Petri Nets (GSPNs) [2]. GSPNs enable an explicit representation of the structure of the underlying resource allocation function, and also, the integration in this representation of the necessary supervisory control policies. At the same time, the additional, time-oriented attributes of the net transitions in the GSPN model, and the related notion of the "random switch", enable the further modeling of the timed dynamics of the considered RAS and of the scheduling problems that are defined with respect to (w.r.t.) these dynamics.

More specifically, in the GSPN-based RAS modeling framework developed in [12], the timed transitions are used to model the processing times that are experienced at the various processing stages of the underlying RAS, while the untimed transitions model the various decisions regarding the initiating, advancing and terminating events for the various running processes and the corresponding resource allocation function.[1] Hence, in this modeling framework, the RAS scheduling problem is posed naturally as the regulation of the firing of the untimed transitions, especially in those markings that enable more than one such transition. In the standard GSPN modeling framework this regulation takes the form of externally specified probability distributions that arbitrate the firing of the contesting untimed transitions, and are known as the corresponding "random switches". In the modeling framework of [12] these random switches must be specified in a way that optimizes some performance index defined w.r.t. the timed dynamics of the considered RAS.

In [12], the aforementioned optimization problem is formulated as a nonlinear program (NLP) and a solution methodology is proposed based on stochastic approximation (SA) methods [11]. However, in its basic positioning, this approach is challenged by the explosive number of vanishing markings in the GSPN state space, that further results in an explosive number of random switches and decision variables for the aforementioned NLP. In order to control this explosion, in

[1]The key assumption behind this modeling practice is that the determination and communication of these resource allocation decisions requires a negligible time compared to the times that are involved in the execution of the various processing steps of the underlying system. Also, we notice that all the technical terms employed in this introductory discussion are fully defined in the more technical part of the paper.

[12] we proposed to restrict the considered optimization problem to a policy space where markings with the same set of enabled untimed transitions share the same regulating distributions, i.e., the same "random switches". The resulting random-switch scheme is known as "static" in the GSPN literature, and the coupling that is introduced by the shared random switches among the various net markings can lead to a substantial reduction of the number of the decision variables that appear in the final NLP formulation. From a modeling standpoint, the RAS-scheduling NLP formulations that are based on the employment of static random switches are still very relevant to the current industrial practice, since they enable the representation of static-priority policies, a class of policies that has raised extensive theoretical and practical interest. On the other hand, the set of static random switches is exponentially large w.r.t. the number of the untimed transitions appearing in the underlying GSPN, and therefore, it is still possible that, for larger RAS instances, the NLP formulations of [12] and the corresponding SA algorithms will have to cope with a very large number of decision variables.

The closing remark in the previous paragraph defines a need for even more parsimonious policy spaces for the considered RAS scheduling problem, in terms of, both, the representational complexity of the target policies and the size of corresponding NLP formulations (where this size is considered primarily w.r.t. the decision variables involved). At the same time, we would like to attain these representational and computational gains without compromising the quality of the currently employed policy spaces in terms of the performance potential that is carried by their constituent policies. The work presented herein details a methodology for attaining this task. This methodology is based on the realization that the regulation exerted by the employed random switches should focus primarily on transitions that are in an actual "conflicting" relationship; non-conflicting transitions can fire in any order without impairing each other, and therefore, there is no need for an explicit arbitration of these firings. On the other hand, the notion of "(non-)conflict" to be employed in the developments that are presented in this work must reflect and support the aforestated objective of retaining the performance potential of the originally employed policy spaces. This need stipulates an analysis that develops at the interface of the timed and untimed behavior of the underlying GSPN, and differentiates the notion of "(non-)conflict" that is presented herein from past investigations of this concept in the PN literature that were based primarily on the untimed aspects of the underlying PN.[2] In view of these remarks, our results can be summarized as follows: (i) We introduce a series of formal relations among the considered scheduling policies and the corresponding policy spaces that enable a rigorous and succinct statement of the policy-space redefinition task that is addressed in this work. (ii) We develop a set of algorithms for supporting this space redefinition in a way that is compatible with the computational framework that has been developed in [12] for the solution of the corresponding NLP formulation. (iii) We also develop an additional set of results, of a more structural nature, that provide a partial characterization of the notion of "(non-)conflict" employed in this work in the form of sufficient conditions, and simplify further the computation of the sought policy spaces. (iv) Finally, we provide a set of computational results that demonstrate the computational efficiency of the presented methods, and their potential to lead to a very substantial reduction of the representational complexity of the considered policy spaces

[2] A work in the past DES literature that has a similar flavor to the results that are presented herein, in that it tries to analyze the effects of some structural and behavioral properties of certain DES classes on their timed dynamics, is that of [7]. Also, from a conceptual standpoint, the work presented in this paper has some affinity with the developments that are presented in [18], which seeks to establish representational economies for the supremal supervisor of DES that are modeled by finite state automata, through the identification and elimination of redundant information encoded in the original representation of this supervisor that is computed by the standard theory of [14].

and of the number of variables that are employed by the corresponding NLP formulations.

The rest of this manuscript is organized as follows: Section 2 surveys the basic GSPN model and the GSPN-based formulation of the considered RAS scheduling problem that was developed in [12]. Section 3 provides the formal characterization of the policy-space redefinition task that is pursued in this work, and the main algorithmic developments that are proposed in support of this task. Section 4 presents the sufficient conditions that complement the results of Section 3. Section 5 provides the computational results that demonstrate the efficacy and the computational tractability of the presented method, and the substantial gains that result from this method w.r.t. the representational complexity of the considered scheduling problem (in terms of the decision variables involved). The material of this section also provides a concrete demonstration of the simplification problem that is addressed in this work and its defining elements. Finally, Section 6 concludes the paper and proposes some directions for future work. Closing this introductory section, we notice, for completeness, that an abridged version of the results presented herein have been submitted to CDC 2015; that write-up contains an exposition of the main ideas that are pursued in this work, without providing, however, the complete technical analysis and the proofs that support the corresponding developments.

2 The GSPN modeling framework and the scheduling problem considered in [12]

This section provides a more technical overview of the developments of [12] that are necessary for the detailed positioning of the problem that is addressed in this work. In the following discussion of this section we assume that the reader is familiar with the basic PN theory, and even with the main concepts and methodology that underlie the GSPN modeling framework, and we stress those aspects of these two frameworks that are most relevant to the results of [12]. Some comprehensive and excellent introductions to the basic PN theory and to the GSPN modeling framework are provided, respectively, in [5] and [2].

2.1 The considered GSPN model

Following the developments of [12], we represent a GSPN $\mathcal{N}$ by a septuple $\mathcal{N} = (P, T_t, T_u, W_t, W_u, m_0, R)$, where: P is the set of places; T_t and T_u are respectively the sets of timed and untimed transitions; W_t and W_u are the flow relations that are respectively defined in $(P \times T_t) \cup (T_t \times P)$ and $(P \times T_u) \cup (T_u \times P)$ and express the connectivity of the elements of T_t and T_u to the elements of P; m_0 is the initial marking of $\mathcal{N}$; and, finally, the mapping $R : T_t \mapsto \mathbb{R}_+$ gives $r_t \equiv R(t), t \in T_t$, the rate of the exponential distribution that determines the firing times for transition t. The considered nets are non-ordinary, and therefore, each pair in the flow relations W_t and W_u is associated with a weight, $w(p,t)$ or $w(t,p)$, that denotes the corresponding token flow.

For a given marking $m \in \mathbb{Z}_+^{|P|}$ and a place $p \in P$, $m(p)$ denotes the number of tokens placed by marking m in place p. Furthermore, the basic transitional dynamics of net $\mathcal{N}$ are expressed by the transition function $tr(\cdot,\cdot) : \mathbb{Z}_+^{|P|} \times T \mapsto \mathbb{Z}_+^{|P|}$ that is defined as follows: $m' = tr(m,t)$ if $m'(p) := m(p) - w(p,t) + w(t,p)$ and $m(p) - w(p,t) \geq 0$ for all $p \in P$; otherwise, $tr(m,t)$ is undefined. We shall use the notation $tr(m,t)!$ to indicate that function $tr(\cdot,\cdot)$ is well-defined for the pair (m,t), and we shall say that $t(\cdot,\cdot)$ is "enabled" or "fireable" at m. Also, we shall

extend the domain of $tr(\cdot,\cdot)$ from $\mathbb{Z}_+^{|P|} \times T$ to $\mathbb{Z}_+^{|P|} \times T^*$ in the standard manner:[3]

$$tr(m, t_1 \dots t_n) = \begin{cases} m & n = 0 \\ tr(m, t_1) & n = 1 \\ tr(tr(m, t_1 \dots t_{n-1}), t_n) & n \geq 2 \end{cases} \quad (1)$$

In line with typical convention in the PN literature, we shall use the notation $\bullet x$ and $x\bullet$ to denote, respectively, the input and the output nodes for a node x (place or transition) of the PN $\mathcal{N}$, and we shall also employ the natural extension of this notation to entire sets of nodes of the same type. At times, we shall also use the notation $m_1 \xrightarrow{t_1 \dots t_n} m_2$ instead of $m_2 = tr(m_1, t_1 \dots t_n)$. Finally, for any given transition sequence $\sigma \in T^*$, we use the notation $|\sigma|$ to denote the "scoring" – or the "Parikh" – vector of this sequence; i.e., $|\sigma|$ is a vector of dimensionality equal to $|T|$, with its i-th component reporting the number of the appearances of the transition t_i in σ.

We also adopt the following notation from [12]: $\mathcal{R}(\mathcal{N}) \subseteq \mathbb{Z}_+^{|P|}$ is the set of all markings of the GSPN $\mathcal{N}$ that are reachable from its initial marking m_0 under the (standard) transition firing scheme that is defined by $tr(\cdot,\cdot)$. For a marking $m \in \mathcal{R}(\mathcal{N})$: $\mathcal{E}_u(m)$ is the set of the enabled untimed transitions at m; $\mathcal{E}_t(m)$ is the set of the enabled timed transitions at m; and $\mathcal{E}(m) = \mathcal{E}_u(m) \cup \mathcal{E}_t(m)$. If $\mathcal{E}_u(m) = \emptyset$, then m is characterized as a "tangible" marking; otherwise, m is a "vanishing" marking. The sets of all reachable tangible and vanishing markings of the GSPN $\mathcal{N}$ are denoted, respectively, by $\mathcal{R}_\mathcal{T}(\mathcal{N})$ and $\mathcal{R}_\mathcal{V}(\mathcal{N})$, and they form a partition of $\mathcal{R}(\mathcal{N})$. Finally, it is further assumed that the considered GSPNs $\mathcal{N}$ are reversible; i.e., for every marking $m \in \mathcal{R}(\mathcal{N})$ there exists a transition sequence $t_1 \dots t_n$ such that $m \xrightarrow{t_1 \dots t_n} m_0$.[4]

Given a marking $m \in \mathcal{R}(\mathcal{N})$, in the following we shall also use the notation $\mathcal{R}^u_\mathcal{V}(m)$ and $\mathcal{R}^u_\mathcal{T}(m)$ to denote, respectively, the sets of all vanishing and tangible markings that are reachable from marking $m \in \mathcal{R}(\mathcal{N})$ by firing some transition sequence in T_u^*. Hence, if $m \in \mathcal{R}_\mathcal{V}(\mathcal{N})$, then $m \in \mathcal{R}^u_\mathcal{V}(m)$. On the other hand, if $m \in \mathcal{R}_\mathcal{T}(\mathcal{N})$, then $\mathcal{R}^u_\mathcal{V}(m) = \emptyset$ and $\mathcal{R}^u_\mathcal{T}(m) = \{m\}$. In the sequel, $\mathcal{R}^u_\mathcal{V}(m)$ will be characterized as the "untimed vanishing reach" from marking m, and $\mathcal{R}^u_\mathcal{T}(m)$ will be the "untimed tangible reach" from m. The reader should also notice that the net reversibility assumption that was introduced in the previous paragraph further implies that, for all markings $m \in \mathcal{R}_\mathcal{V}(\mathcal{N})$, $\mathcal{R}^u_\mathcal{T}(m) \neq \emptyset$; i.e., the considered GSPN $\mathcal{N}$ cannot be trapped in a subspace of $\mathcal{R}(\mathcal{N})$ that consists entirely of vanishing markings. In fact, since the modeling paradigm of [12] interprets the untimed transitions as (resource allocation) decisions, it is further assumed in [12] that the untimed vanishing reach $\mathcal{R}^u_\mathcal{V}(m)$ of any marking $m \in \mathcal{R}_\mathcal{V}(\mathcal{N})$ corresponds to a subspace that is an acyclic digraph. Finally, for every marking $m \in \mathcal{R}_\mathcal{T}(\mathcal{N})$, we also define the non-empty set $\mathcal{R}^1_\mathcal{T}(m) \equiv \bigcup_{m' : \exists t \in \mathcal{E}(m) \text{ s.t. } m \xrightarrow{t} m'} \mathcal{R}^u_\mathcal{T}(m')$; in plain terms, $\mathcal{R}^1_\mathcal{T}(m)$ denotes the set of all the tangible markings that are reachable from the tangible marking m through the firing of some transition in $\mathcal{E}(m)$ and an additional (possibly empty) sequence of untimed transitions.

It is well known from the basic GSPN theory that the reachability space $\mathcal{R}(\mathcal{N})$ of any given GSPN $\mathcal{N}$, when endowed with the transitional dynamics that characterize the timed and the

[3] We remind the reader that T^* denotes the Kleene closure of the set T, i.e., all the finite-length strings consisting of elements from this set, including the empty string ϵ.

[4] In the case of RAS-modeling GSPNs, reversibility is ensured by (i) the assumption that the corresponding RAS starts its operation empty, and (ii) the role of the applied liveness-enforcing supervisor (LES), which ensures that every initiated process instance can run successfully to its completion.

untimed transitions of $\mathcal{N}$, constitutes a semi-Markov process [2], to be denoted by $\mathcal{SM}(\mathcal{N})$. The transitional dynamics of $\mathcal{SM}(\mathcal{N})$ at any tangible marking $m \in \mathcal{R}_{\mathcal{T}}(\mathcal{N})$ are determined by the exponential race that is defined by $\mathcal{E}_t(m)$, i.e., the enabled transitions at m. On the other hand, the transitional dynamics of $\mathcal{SM}(\mathcal{N})$ at any vanishing marking $m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$ are defined by (i) the set $\mathcal{E}_u(m)$, i.e., the set of the enabled untimed transitions at m, and (ii) a probability distribution $\Xi(m)$ that is supported by $\mathcal{E}_u(m)$ and regulates the firing of the transitions in this set; in the GSPN terminology, $\Xi(m)$ is known as the "random switch" (r.s.) that is associated with vanishing marking m. For the considered GSPN, the specification of a well defined set of random switches $\Xi(m),\ \forall m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$, induces, for every marking $m \in \mathcal{R}_{\mathcal{T}}(\mathcal{N})$, a one-step transition probability distribution $p(\cdot|m)$ with corresponding support the set $\mathcal{R}^1_{\mathcal{T}}(m)$ that was defined in the previous paragraph. In this way, the timed dynamics that are expressed by the semi-Markov process $\mathcal{SM}(\mathcal{N})$ can be reduced to a continuous-time Markov chain (CTMC), $\mathcal{M}(\mathcal{N})$, that is defined on $\mathcal{R}_{\mathcal{T}}(\mathcal{N})$. Finally, if the CTMC $\mathcal{M}(\mathcal{N})$ satisfies the standard ergodicity assumptions of the relevant Markov chain theory (e.g., c.f. [3]), it will also possess an equilibrium distribution $\pi \equiv [\pi(m),\ m \in \mathcal{R}_{\mathcal{T}}(\mathcal{N})]$ that defines the long-term behavior – or, the "steady-state" dynamics – of the underlying GSPN $\mathcal{N}$.

2.2 The GSPN-based scheduling problem addressed in [12]

An additional attribute of the RAS-modeling GSPN $\mathcal{N}$ considered in [12] is the finiteness of its reachability set $\mathcal{R}(\mathcal{N})$.[5] This last property, when combined with the presumed reversibility of the considered nets, further implies that the corresponding CTMC $\mathcal{M}(\mathcal{N})$ will be ergodic for any selection of random switches $\{\Xi(m) : m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})\}$ that retains the net reversibility. A practical way to establish this last requirement is by maintaining some randomization in each random switch $\Xi(m)$ that guarantees a minimum positive selection probability δ for each element of the set $\mathcal{E}_u(m)$. Given a net $\mathcal{N}$ from the considered GSPN class, in the sequel we shall use the notation $\Pi(\delta)$ to denote the set containing all the possible selections of random switches $\{\Xi(m) : m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})\}$ that satisfy this randomization requirement. Hence, $\Pi(\delta)$ defines a set of "control policies" for net $\mathcal{N}$ that are characterized by well-defined steady-state dynamics.

However, the practical applicability of the policies in $\Pi(\delta)$ is limited by the fact that they require the specification of a separate random switch $\Xi(m)$ for every marking $m \in \mathcal{R}(\mathcal{N})$; in most practical applications, $|\mathcal{R}(\mathcal{N})|$ will grow very fast w.r.t. any parsimonious representation of the septuple that defines the net $\mathcal{N}$ itself. Hence, [12] also proposes the restriction of the considered class of control policies to the particular subset of $\Pi(\delta)$ that satisfies the following additional constraint:

$$\forall m, m' \in \mathcal{R}_{\mathcal{V}}(\mathcal{N}) \text{ with } \mathcal{E}_u(m) = \mathcal{E}_u(m'),\ \ \Xi(m) = \Xi(m') \tag{2}$$

As explained in the introductory section, any selection of random switches that observes Eq. (2) is characterized as "static", since it defines the transitional dynamics at any marking $m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$ on the basis of the information that is provided in the more structural entity of the corresponding set $\mathcal{E}_u(m)$, and not the information that is provided in the marking m itself. In the sequel, the set of control policies in $\Pi(\delta)$ that satisfies the additional constraint of Eq. (2) will be denoted by $\Pi^S(\delta)$.

[5] The finiteness of $\mathcal{R}(\mathcal{N})$, for any RAS-modeling GSPN $\mathcal{N}$, results from the finiteness of the various resource types of this RAS and the fact that each processing stage requires at least one resource unit for its execution.

We complete our overview of the developments in [12], by adding that these developments also presume the existence of a function $f(\cdot)$ that is defined on $\mathcal{R}_{\mathcal{T}}(\mathcal{N})$ and specifies an "(immediate) reward rate" that is collected by net $\mathcal{N}$ when visiting the corresponding tangible marking. The availability of function $f(\cdot)$ enables the definition of an optimization problem that concerns the selection of a control policy from the set $\Pi^S(\delta)$ that maximizes the collected long-term average reward. The primary decision variables for this optimization problem are the selection probabilities, ξ, appearing in the various random switches that are employed by the policies in $\Pi^S(\delta)$. Letting $\bar{\xi}$ denote a vector that collects all these decision variables, and $\eta(\bar{\xi})$ denote the resulting value for the aforementioned objective function, the considered optimization problem can be concisely expressed as follows:

$$\max_{\bar{\xi}} \ \eta(\bar{\xi}) \tag{3}$$

s.t.

$$\forall \xi \in \bar{\xi}, \quad \delta \leq \xi \tag{4}$$

$$\forall \Xi, \quad \delta \leq 1.0 - \sum_{\xi \in \Xi \cap \bar{\xi}} \xi \tag{5}$$

In the above formulation, the constraints of Eq. (4) express the requirement that, in the considered class of policies, $\Pi^S(\delta)$, every enabled untimed transition has a minimal selection probability of δ. On the other hand, the constraints of Eq. (5) are necessitated by the fact that for every employed random switch Ξ, the selection probability for one of the elements of the corresponding set $\mathcal{E}_u$ is modeled only implicitly by the difference in the right-hand-side of these constraints; hence, the constraints of Eq. (5) express the same requirement that is expressed by the constraints of Eq. (4), but for those selection probabilities that are modeled implicitly. The last part of [12] outlines a methodology for the solution of the mathematical programming (MP) formulation of Eqs (3)–(5) through some relevant theory from the sensitivity analysis of Markov reward processes [4] and stochastic approximation [11]; we refer the reader to [12] for the relevant details.

3 A refined policy space for the considered scheduling problem

3.1 Preamble

It is clear from the discussion of Section 2 that, in any MP formulation for the considered scheduling problems along the lines of Eqs (3)–(5), the number of the employed decision variables ξ is determined by (i) the sets of the enabled untimed transitions, $\mathcal{E}_u(m)$, at each marking $m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$, and (ii) the sets of the distinct random switches that are defined w.r.t. each of these transition sets. In this section we propose a refinement of the sets $\mathcal{E}_u(m)$ that can lead to a substantial reduction of, both, the set of the random switches and the corresponding set of the decision variables that will appear in the final MP formulation; controlling the size of these two sets is very important for enhancing the effectiveness and the efficiency of the SA methods that are used in the solution of this formulation.

The departing point for the subsequent developments of this section is the observation that the performance of any given policy $\mathcal{P}$ from the policy spaces $\Pi(\delta)$ and $\Pi^S(\delta)$ that were introduced in Section 2, is essentially determined by the structure of the corresponding CTMC $\mathcal{M}(\mathcal{N};\mathcal{P})$, i.e., (i) the set of the reachable tangible markings under policy $\mathcal{P}$, $\mathcal{R}_{\mathcal{T}}(\mathcal{N};\mathcal{P})$, and (ii) the

one-step transition probability distributions $p(\cdot|m;\mathcal{P})$, $m \in \mathcal{R}_{\mathcal{T}}(\mathcal{N};\mathcal{P})$, that are induced by the random switches employed by $\mathcal{P}$. Hence, any two policies $\mathcal{P}_1$ and $\mathcal{P}_2$ from these policy spaces will present the same performance as long as they evolve the underlying GSPN $\mathcal{N}$ over the same set of tangible markings and with the same set of one-step transition probability distributions. This last remark motivates the following definition:

Definition 1 *For the considered scheduling problem:*

*i. Two policies $\mathcal{P}_1$ and $\mathcal{P}_2$ from the policy space $\Pi(\delta)$ will be characterized as "performance-equivalent" if and only if (*iff*) they define the same CTMC $\mathcal{M}(\mathcal{N})$ for the underlying GSPN $\mathcal{N}$.*

ii. For any two subspaces $\Pi_1(\delta)$ and $\Pi_2(\delta)$ of $\Pi(\delta)$, we shall say that policy space $\Pi_1(\delta)$ "carries the performance potential" of policy space $\Pi_2(\delta)$ iff *there exists an optimal policy $\mathcal{P}$ in $\Pi_2(\delta)$ that has a performance-equivalent policy $\mathcal{P}'$ in $\Pi_1(\delta)$.*

In view of the above discussion on the challenges that are posed to the solution of the MP formulation of Eqs (3)–(5) by vectors $\bar{\xi}$ of high dimensionality, we would like to identify a policy space $\tilde{\Pi}(\delta)$ that carries the performance potential of the original policy space $\Pi(\delta)$ and results in the smallest possible vector $\bar{\xi}$ for the corresponding MP formulation of Eqs (3)–(5). However, the effective resolution of this last problem would necessitate a holistic view of the underlying reachability space $\mathcal{R}(\mathcal{N})$, and, thus, it is challenged by complexity considerations similar to those that were discussed in the previous sections. Hence, in the rest of this work, we take a more pragmatic stance to this problem, seeking to develop a heuristic approach that will redefine the sets of the activated untimed transitions at the various vanishing markings of the considered GSPN models, based on some logic that will depend on some more "local" attributes of these markings.

More specifically, in the rest of this section we provide a systematic methodology that will enable the identification of a set collection $\{\tilde{\mathcal{E}}_u(m) \subseteq \mathcal{E}_u(m),\ m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})\}$, such that the policy space $\tilde{\Pi}(\delta)$ that is induced by these sets (i) will effect a significant reduction to the numbers of the random switches and of the decision variables ξ that are employed by the corresponding MP formulation of Eqs (3)–(5), and, at the same time, (ii) will carry the performance potential of the original policy space $\Pi(\delta)$. Furthermore, as it will be shown in the following, the methodology that is developed in this work can be combined with the notion of the "static" random switch that is defined by Eq. 2, by applying this equation on the revised transition sets $\tilde{\mathcal{E}}_u(m)$; the resulting policy space will be denoted by $\tilde{\Pi}^S(\delta)$, and the corresponding MP formulation of Eqs (3)–(5) will involve a number of decision variables ξ that is substantially smaller than the number of the decision variables that are employed by the same formulation when applied on the original policy space $\Pi^S(\delta)$. Finally, the subsequent developments will also establish that the aforementioned reductions and simplifications essentially result from the ability of the presented methodology to identify and control the real conflicts that arise in the underlying RAS, while avoiding the arbitration of any elements of concurrency that do not correspond to conflicting behavior.

3.2 Some defining properties for the sets $\tilde{\mathcal{E}}_u(m)$

According to the discussion of the previous section, the sought set collection $\{\tilde{\mathcal{E}}_u(m) \subseteq \mathcal{E}_u(m),\ m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})\}$ must induce a policy space $\tilde{\Pi}(\delta)$ that carries the performance potential of the

policy space $\Pi(\delta)$, and also controls effectively the complexity of the corresponding scheduling problem. In order to address the first of these two requirements, we stipulate the following condition for the sought sets $\tilde{\mathcal{E}}_u(m)$.

Condition 1 *The untimed tangible reach $\mathcal{R}^u_{\mathcal{T}}(m')$ for any vanishing marking $m' \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$, under the revised sets $\tilde{\mathcal{E}}_u(m),\ m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$, must remain the same with the untimed tangible reach of m' under the original sets $\mathcal{E}_u(m),\ m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$.*

It is easy to see that Condition 1 retains the communication structure among all the tangible markings $m \in \mathcal{R}_{\mathcal{T}}(\mathcal{N})$. Furthermore, the acyclic nature of the subspace that is induced by the untimed vanishing reach $\mathcal{R}^u_{\mathcal{V}}(m)$ of any vanishing marking $m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$, implies that we can effect any desired probability distribution that will regulate the transition from marking m to its untimed tangible reach $\mathcal{R}^u_{\mathcal{T}}(m)$, by pricing appropriately the random switches that appear in the untimed vanishing reach of m.[6] This last remark, when applied to the vanishing markings m' that result from some tangible marking $m \in \mathcal{R}_{\mathcal{T}}(\mathcal{N})$ through the firing of a transition $t \in \mathcal{E}(m)$, further implies that the random switches that are induced by any set collection $\{\tilde{\mathcal{E}}_u(m) \subseteq \mathcal{E}_u(m),\ m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})\}$ that satisfies Condition 1 will materialize any one-step transition probability distribution $p(\cdot|m)$, for the considered tangible marking m, that is also realizable in the original policy space $\Pi(\delta)$. Hence, Condition 1 is sufficient for ensuring the realization, under the refined sets $\tilde{\mathcal{E}}_u(m)$, of any CTMC $\mathcal{M}(\mathcal{N})$ that is realizable in the original policy space $\Pi(\delta)$. But then, the above discussion leads to the following conclusion:

Proposition 1 *The policy space $\tilde{\Pi}(\delta)$, induced by any collection $\{\tilde{\mathcal{E}}_u(m),\ m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})\}$ that satisfies Condition 1, carries the performance potential of the original policy space $\Pi(\delta)$.*[7]

When it comes to the second objective that must be observed by the refined sets $\tilde{\mathcal{E}}_u(m) \subseteq \mathcal{E}_u(m)$, $m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$, i.e., the control of the representational and the computational complexity of the resulting scheduling problem as manifested by the number of the decision variables ξ in the corresponding MP formulation of Eqs (3)–(5), we already remarked in Section 3.1 that any complete solution procedure would require the expansion of the entire reachability graph $\mathcal{R}(\mathcal{N})$. Since this is an intractable proposition, we propose to use as a general guideline for the selection among the set collections $\{\tilde{\mathcal{E}}_u(m) \subseteq \mathcal{E}_u(m),\ m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})\}$ that satisfy Condition 1, the notion of "minimality" that is established in the following condition.

Condition 2 *For any vanishing marking $m' \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$ and any transition $t \in \tilde{\mathcal{E}}_u(m')$, the substitution of the set $\tilde{\mathcal{E}}_u(m')$ by the set $\tilde{\mathcal{E}}_u(m') \setminus \{t\}$ in the set collection $\{\tilde{\mathcal{E}}_u(m), m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})\}$ will result in the violation of Condition 1.*

[6] The validity of this claim is easily established by some results in (static) network flow theory establishing the ability to distribute a unit of flow entering an acyclic connected digraph from a single source node to its terminal nodes in any possible manner, as long as the total amount of flow that is received by these nodes is equal to one.

[7] On the other hand, it is also important to notice that Condition 1 cannot guarantee that the policy space $\tilde{\Pi}^S(\delta)$ carries the performance potential of its counterpart policy space $\Pi^S(\delta)$. This limitation is due to the coupling that is introduced by Eq. 2 in the pricing of the random switches that correspond to distinct vanishing markings. Nevertheless, Condition 1 still guarantees that every CTMC $\mathcal{M}(\mathcal{N})$ that is realizable in the original policy space $\Pi(\delta)$, is also realizable by the policies in the policy space $\tilde{\Pi}^S(\delta)$ through a controlled (partial) relaxation of the coupling that is defined by Eq. 2.

The reader should notice that the notion of minimality that is introduced by Condition 2 does not determine uniquely the target sets $\tilde{\mathcal{E}}_u(m),\ m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$. Furthermore, different collections $\{\tilde{\mathcal{E}}_u(m),\ m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})\}$, satisfying Conditions 1 and 2 will result in different dimensions for the variable vector $\bar{\xi}$ that is employed by the corresponding MP formulation of Eqs (3)–(5). Hence, in general, there is a need for additional selection logic that will resolve this ambiguity, and possibly enable the further optimization of the final selection of these sets in various application contexts. We shall revisit this issue in the next section, where we detail a particular algorithm for the determination of the target sets $\tilde{\mathcal{E}}_u(m),\ m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$.

3.3 Computing the refined sets $\tilde{\mathcal{E}}_u(m)$

In this section, first we provide a more explicit characterization of the sets $\tilde{\mathcal{E}}_u(m),\ m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$, that satisfy Condition 1, and subsequently we propose some further logic that can facilitate the selection of the final collection $\{\tilde{\mathcal{E}}_u(m),\ m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})\}$ among all those that satisfy both Conditions 1 and 2. These two developments will also suggest a very straightforward algorithm for the effective computation of the collection $\{\tilde{\mathcal{E}}_u(m),\ m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})\}$ that is targeted by the aforementioned logic. We start these developments with the following proposition.

Proposition 2 *Consider a marking $\hat{m} \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$ and a set $\hat{T} \subseteq \mathcal{E}_u(\hat{m})$. Then, the substitution of the set $\tilde{\mathcal{E}}_u(\hat{m})$ by the set $\hat{T}$ in any collection $\{\tilde{\mathcal{E}}_u(m),\ m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})\}$ that satisfies Condition 1, will retain the satisfaction of this condition for the considered marking $\hat{m}$* iff *the set $\hat{T}$ satisfies the following condition:*

$$\bigcup_{m':\exists t \in \hat{T}, \hat{m} \xrightarrow{t} m'} \mathcal{R}^u_{\mathcal{T}}(m') = \mathcal{R}^u_{\mathcal{T}}(\hat{m}) \tag{6}$$

Proof: We remind the reader that Condition 1 requires that the selected transition sets will establish, for every vanishing marking $m'' \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$, an untimed tangible reach equal to $\mathcal{R}^u_{\mathcal{T}}(m'')$ (i.e., the corresponding untimed tangible reach which results from the original collection $\{\mathcal{E}_u(m),\ \forall m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})\}$ that is defined by the natural dynamics of net $\mathcal{N}$). Also, for the needs of this proof, we introduce the following additional notation: (a) For any marking $m'' \in \mathcal{R}(\mathcal{N})$, $\mathcal{R}^u_{\mathcal{T}}(m''; \tilde{\mathcal{E}})$ will denote the untimed tangible reach of m'' defined by the collection $\{\tilde{\mathcal{E}}_u(m),\ m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})\}$ that is considered by Proposition 2. (b) On the other hand, for any marking $m'' \in \mathcal{R}(\mathcal{N})$, $\mathcal{R}^u_{\mathcal{T}}(m''; \hat{T})$ will denote the untimed tangible reach of m'' that is defined by the collection $\{\tilde{\mathcal{E}}_u(m),\ m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})\}$ under the modification described in Proposition 2 (i.e., with the transition set $\tilde{\mathcal{E}}_u(\hat{m})$ replaced by the set $\hat{T}$ defined in the proposition). Finally, we also remind the reader that, in the considered GSPN class, the markings m'' in $\mathcal{R}^u_{\mathcal{V}}(\hat{m})$ (i.e., the untimed vanishing reach of the considered vanishing marking $\hat{m}$) induce, by assumption, an acyclic subspace of $\mathcal{R}(\mathcal{N})$. But then, the substitution of the set $\tilde{\mathcal{E}}_u(\hat{m})$ by the set $\hat{T}$, as suggested by Proposition 2, does not impact the untimed tangible reaches of the markings m' that result from $\hat{m}$ by firing any transition in $\hat{T}$; i.e.,

$$\forall m' \text{ s.t. } \exists t \in \hat{T} : \hat{m} \xrightarrow{t} m', \quad \mathcal{R}^u_{\mathcal{T}}(m'; \hat{T}) = \mathcal{R}^u_{\mathcal{T}}(m'; \tilde{\mathcal{E}}) \tag{7}$$

Since the considered set collection $\{\tilde{\mathcal{E}}_u(m),\ m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})\}$ satisfies Condition 1, it also holds that

$$\forall m' \text{ s.t. } \exists t \in \hat{T} : \hat{m} \xrightarrow{t} m', \quad \mathcal{R}^u_{\mathcal{T}}(m'; \tilde{\mathcal{E}}) = \mathcal{R}^u_{\mathcal{T}}(m') \tag{8}$$

Hence, the left-hand-side of Eq. (6) essentially expresses $\mathcal{R}^u_{\mathcal{T}}(\hat{m};\hat{T})$ – the untimed tangible reach of the considered marking $\hat{m}$ for the collection of the untimed transition sets that is defined in Proposition 2 – and the validity of the claimed result becomes obvious. $\square$

Of particular interest in the subsequent developments is the following corollary of Proposition 2, that can be established from this proposition through an inductive argument on the acyclic graphs that correspond to the untimed vanishing reaches of the vanishing markings $m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$.

Corollary 1 *A collection* $\{\tilde{\mathcal{E}}_u(m) \subseteq \mathcal{E}_u(m),\ m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})\}$ *satisfies Condition 1* iff *every set* $\tilde{\mathcal{E}}_u(m),\ m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$*, satisfies the condition of Eq. (6).*

Corollary 1 enables a localized computation, at any marking $m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$, of all the sets $\tilde{\mathcal{E}}_u(m)$ that can be part of any collection that will satisfy Condition 1. This decomposing effect can be further exploited towards the identification of such collections $\{\tilde{\mathcal{E}}_u(m) \subseteq \mathcal{E}_u(m),\ m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})\}$ that will also satisfy Condition 2. In particular, the above developments imply that any collection $\{\tilde{\mathcal{E}}_u(m) \subseteq \mathcal{E}_u(m),\ m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})\}$ consisting of *minimal-cardinality* subsets of the corresponding parent sets $\mathcal{E}_u(m)$ that satisfy the condition of Eq. (6), will also satisfy Condition 2 (since, if this condition was violated by some particular set $\tilde{\mathcal{E}}_u(m')$, then this set would not be among the minimal-cardinality subsets of $\mathcal{E}_u(m')$ that satisfies the condition of Eq. (6)). Hence, we have the following proposition:

Proposition 3 *A collection* $\{\tilde{\mathcal{E}}_u(m) \subseteq \mathcal{E}_u(m),\ m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})\}$ *consisting of* minimal-cardinality *subsets of the corresponding parent sets* $\mathcal{E}_u(m)$ *that satisfy the condition of Eq. (6), will satisfy both Conditions 1 and 2.*

At this point, the following remarks are in order: (a) By minimizing the cardinality of the selected sets $\tilde{\mathcal{E}}_u(m),\ m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$, the selection logic that is implied by Proposition 3 is generally conducive to the primary objective of minimizing the number of the decision variables ξ. (b) On the other hand, the resulting selection scheme still needs further arbitration / disambiguation in the case of vanishing markings $m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$ with more than one minimal-cardinality subsets of the corresponding set $\mathcal{E}_u(m)$ that satisfy the condition of Eq. (6). Ideally, this conflict should be arbitrated in a way that leads to the minimal-dimensionality vector $\bar{\xi}$ for the corresponding MP formulation of Eqs (3)–(5). Since, however, such an arbitration scheme is not practically feasible in the context of a real-time resolution of the considered ambiguity, as is the case with the computational framework of [12], in this work we propose to resolve this ambiguity by taking, at every vanishing marking $m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$, the minimal-cardinality set $\tilde{\mathcal{E}}_u(m) \subseteq \mathcal{E}_u(m)$ that satisfies the condition of Eq. (6) and it is also lexicographically minimal w.r.t. the ordering that is defined by the natural (or, perhaps, some other) indexing of the transition set T_u. Then, at every visited vanishing marking $m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$, the corresponding set $\tilde{\mathcal{E}}_u(m)$ can be effectively obtained through the execution of the computation that is depicted in Algorithm 1.

Algorithm 1 first computes the set $\mathcal{E}_u(m)$, and, subsequently, for each transition t in this set, it computes (a) the marking $m'(t)$ that results from the firing of transition t in m, and (b) the untimed tangible reach of $m'(t)$ under the original collection $\{\mathcal{E}_u(m''),\ m'' \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})\}$ of [12]. Also, the algorithm computes the untimed tangible reach of marking m as the union of the untimed tangible reaches of the markings $m'(t),\ t \in \mathcal{E}_u(m)$. In the last phase of its computation, Algorithm 1 enumerates the subsets of the set $\mathcal{E}_u(m)$, sequencing them first in

Algorithm 1 Computing the target set $\tilde{\mathcal{E}}_u(m)$ at a visited vanishing marking m

Input: $GSPN \mathcal{N} = (P, T_t, T_u, W)$; vanishing marking m
Output: the transition set $\tilde{\mathcal{E}}_u(m)$

```
T̂ ← E_u(m)
for each t ∈ T̂ do
  m'(t) ← tr(m, t);
  UTR(t) ← R^u_T(m'(t))
end for
UTR0 ← ⋃_{t∈T̂} UTR(t)
for i := 1, ... |T̂| do
  for j := 1, ..., C^i_{|T̂|} do
    T̃ ← the j-th subset of T̂ of cardinality i, where these subsets are enumerated lexicographically based on some ordering of the elements of T̂
    if ⋃_{t∈T̃} UTR(t) = UTR0 then
      return T̃
    end if
  end for
end for
```

increasing cardinality, and second (i.e., the subsets with equal cardinality) lexicographically according to some ordering that is imposed on the elements of $\mathcal{E}_u(m)$;[8] for every transition set $\tilde{T}$ that is generated by this enumeration, Algorithm 1 checks whether this set satisfies the condition of Eq. (6) in Proposition 2, in which case it terminates returning the set $\tilde{T}$ as the sought set $\tilde{\mathcal{E}}_u(m)$.

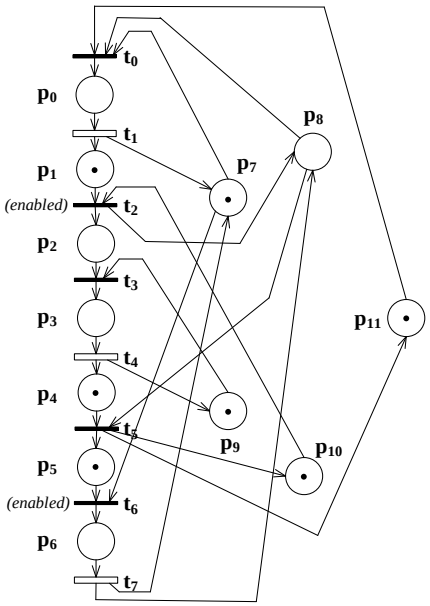
The computational complexity of Algorithm 1 is determined primarily by (a) the computation of the untimed tangible reach for every marking $m'(t)$, and (b) the enumeration of the subsets of $\mathcal{E}_u(m)$. Both of these elements imply a super-polynomial worst-case complexity w.r.t. the "size" of the underlying GSPN $\mathcal{N}$.[9] Since, however, (i) the computation that provides the aforementioned untimed tangible reaches is performed on acyclic subgraphs of a more local nature w.r.t. the entire reachability space $\mathcal{R}(\mathcal{N})$, and (ii) the sets $\mathcal{E}_u(m)$ are usually small subsets of the transition set T, it is expected that the empirical computational complexity of Algorithm 1 will be quite benign. These remarks are corroborated by the corresponding results of the numerical experiments that are reported in Section 5.2.

Finally, the next example demonstrates the execution of Algorithm 1 and the above claims regarding the low empirical complexity of the algorithm. Furthermore, the example reveals some additional attributes of the collections $\{\tilde{\mathcal{E}}_u(m) \subseteq \mathcal{E}_u(m),\ m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})\}$ that are generated by the considered algorithm.

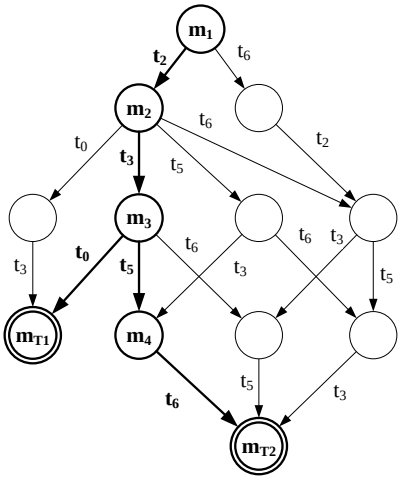
Example Consider the marked GSPN $\mathcal{N}$ depicted in Figure 1a. In the depicted net, the black transitions are untimed, while the white ones are timed. Since the depicted marking enables the untimed transitions t_2 and t_6, it is vanishing; for further reference, we shall label this marking as m_1. The digraph of Figure 1b represents the untimed vanishing and tangible

[8] In lack of any bias that might result from special structure or further input, this can be the ordering that is defined by the natural indexing of the net transitions in T.

[9] We remind the reader that this last quantity is defined as the size of any parsimonious encoding / representation of the septuple that defines GSPN $\mathcal{N}$.



(a) Example: the considered GSPN marking m_1.



(b) The digraph characterizing the untimed vanishing and tangible reaches of marking m_1, $\mathcal{R}^u_{\mathcal{V}}(m_1)$ and $\mathcal{R}^u_{\mathcal{T}}(m_1)$, under (i) the original random switches and (ii) the refinement of these random switches by Algorithm 1.

Figure 1: The GSPN vanishing marking, m_1, and the digraph that characterizes its untimed tangible reach, for the example of Section 3.3.

reaches of marking m_1. More specifically, in Figure 1b, the nodes depicted as single-bordered correspond to vanishing markings in the untimed reach of marking m_1, and those depicted as double-bordered correspond to tangible markings. Finally, the nodes and the edges that are depicted with thicker lines in the digraph of Figure 1b, represent the untimed vanishing and tangible reaches of the considered vanishing marking m_1 under the random switches that are returned by Algorithm 1.

It is interesting to notice in the digraph of Figure 1b the extent of the reduction that is effected by Algorithm 1 in terms of (a) the reachable vanishing markings and (b) the dimensionality of the random switches that are constructed for the remaining vanishing markings. Both of these reductions simplify extensively the arbitration that must be effected on this part of the underlying state space by the corresponding scheduling problem. Furthermore, the small size of the random switches that are eventually employed at the reachable vanishing markings visited by the algorithm, also corroborates our earlier claims that, in general, the enumeration of the subsets of the sets $\mathcal{E}_u(m)$ that is performed by Algorithm 1 at each visited vanishing marking $m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$, will be terminated in its early stages.

Finally, next we take a closer look at the way that Algorithm 1 treats the existing conflicts among the transitions of the sets $\mathcal{E}_u(m)$ in its selection of the corresponding sets $\tilde{\mathcal{E}}_u(m)$. Hence, consider first the execution of Algorithm 1 at the original vanishing marking m_1. At this marking, the algorithm picked transition t_2 over t_6. The reader can check that these two transitions are not in conflict, since $\bullet t_2 = \{p_1, p_{10}\}$, and $\bullet t_6 = \{p_5, p_7\}$. However, selecting among non-conflicting transitions is still quite subtle. More specifically, the firing of the transition t_2 at marking m_1 will release tokens to its output places p_2 and p_8. And adding one token in p_8 will

enable the untimed transition t_0, which is in conflict with t_6, since t_0 and t_6 have the common input place p_7. Hence, giving arbitrary precedence to transition t_6 over transition t_2 at marking m_1 would have a restrictive effect on the potential behavior that can be generated by the eventual firing of t_2. On the other hand, firing t_6 at m_1 will only release one token to the place p_6, which is not an input place of any untimed transitions. Thus, firing t_6 cannot add any new elements to the set $\mathcal{E}_u(m)$, and there are no hidden conflicts by dropping t_6 from the refined random switch of m_1. Similarly, the algorithm selection at marking m_2 can be interpreted by the fact that, for the effected partition $(\{t_3\},\ \{t_0, t_5, t_6\})$ of the original set $\mathcal{E}_u(m_2)$, it holds that $\bullet\{t_3\} = \{p_2, p_9\}$, and $\bullet\{t_0, t_5, t_6\} = \{p_4, p_5, p_7, p_8, p_{11}\}$; hence, there is no conflict between these two sets. Furthermore, from the information that is provided in Figure 1b, it can be easily checked that no new untimed transitions will be enabled before reaching some tangible marking from m_2. Thus, in this case, we can eliminate either $\{t_3\}$ or $\{t_0, t_5, t_6\}$, according to the logic that interpreted the output of Algorithm 1 on m_1. Eventually, Algorithm 1 gives precedence to the set $\{t_3\}$ since it is the set with the smallest cardinality. Finally, at marking m_3, Algorithm 1 returns the transition set $\{t_0, t_5\}$, and transition t_6 is eliminated from the original set of enabled untimed transitions. When considering the original set $\mathcal{E}_u(m_3) = \{t_0, t_5, t_6\}$, we can see that t_0 is in conflict with t_5 and t_6, but t_5 and t_6 are not in conflict. Since in this case there are no further hidden conflicts that could have been revealed only after the firing of the non-conflicting transitions t_5 and t_6, Algorithm 1 includes only one of the non-conflicting transitions in the returned random switch, and this is transition t_5 due to the applied lexicographic ordering. The next section provides a formal characterization to the apparent connection between the selection logic of Algorithm 1 and the nature of the conflicting relations between the selected transition sets $\tilde{\mathcal{E}}_u(m)$ and their complements $\mathcal{E}_u(m) \setminus \tilde{\mathcal{E}}_u(m)$, that was observed in this example.

4 Some sufficient conditions for the condition of Proposition 2

In this section we provide two results that define sufficient conditions for the key condition of Eq. 6 in Proposition 2. Hence, when applicable, both of these conditions provide alternative means to the logic that is pursued in Algorithm 1 for validating the condition of Eq. 6 for any given pair of a vanishing marking $m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$ and a subset $\hat{T}$ of the corresponding set $\mathcal{E}_u(m)$. In particular, the second of these two conditions involves only structural attributes of the elements of the sets $\hat{T}$ and $\mathcal{E}_u(m) \setminus \hat{T}$, and therefore, its testing can be much more efficient than any test of the condition of Eq. 6 that is based on the enumeration of the corresponding untimed tangible reach $\mathcal{R}^u_{\mathcal{T}}(m)$. Finally, the first of the two presented conditions highlights the existing connection between Condition 1 of Section 3.2 and the notion of "(non-)conflict" that was indicated in the discussion of the example of Section 3.3.

Proposition 4 *Consider a vanishing marking m of a GSPN $\mathcal{N}$, and a partition (T_1, T_2) of $\mathcal{E}_u(m)$. Furthermore, let $\mathcal{E}_u((T_u \setminus T_1)^*) \subseteq T_u \setminus \mathcal{E}_u(m)$ denote the set containing those untimed transitions of $\mathcal{N}$ that are not enabled in m but get enabled after firing some transition sequences in $(T_u \setminus T_1)^*$. Also, set $\hat{T}_2 \equiv T_2 \cup \mathcal{E}_u((T_u \setminus T_1)^*)$. Finally, suppose that for each transition sequence $\sigma \in \hat{T}_2^*$, there exists a transition sequence $\hat{\sigma} t_1 \in \hat{T}_2^* T_1$ such that the transition sequences $\sigma\hat{\sigma}t_1$ and $t_1\sigma\hat{\sigma}$ are feasible at m. Then, the set T_1 satisfies the condition of Eq.* (6) *in Proposition 2.*

Proof: We need to show that $\mathcal{R}^u_{\mathcal{T}}(m) = \bigcup_{m' \in \mathcal{M}} \mathcal{R}^u_{\mathcal{T}}(m')$ where $\mathcal{M} = \{m' \in \mathcal{R}(\mathcal{N}) : \exists t \in T_1 \text{ s.t. } m' = tr(m, t)\}$. The inclusion $\mathcal{R}^u_{\mathcal{T}}(m) \supseteq \bigcup_{m' \in \mathcal{M}} \mathcal{R}^u_{\mathcal{T}}(m')$ is obvious since $T_1 \subseteq \mathcal{E}_u(m)$. So, next, we focus on the inclusion $\mathcal{R}^u_{\mathcal{T}}(m) \subseteq \bigcup_{m' \in \mathcal{M}} \mathcal{R}^u_{\mathcal{T}}(m')$.

Consider any tangible marking $m_{\mathcal{T}} \in \mathcal{R}^u_{\mathcal{T}}(m)$. Then, there exists a feasible transition sequence $\sigma^n = t_1 t_2 \dots t_n$ leading from marking m to $m_{\mathcal{T}}$. To show that $m_{\mathcal{T}} \in \bigcup_{m' \in \mathcal{M}} \mathcal{R}^u_{\mathcal{T}}(m')$, we distinguish the following two cases:

Case 1: $t_1 \in T_1$. Then, $m_{\mathcal{T}} \in \mathcal{R}^u_{\mathcal{T}}(tr(m, t_1)) \subseteq \bigcup_{m' \in \mathcal{M}} \mathcal{R}^u_{\mathcal{T}}(m')$.

Case 2: $t_1 \in T_2$. Then, by the last assumption of Proposition 4, there exists $t_i \in T_1$ in the sequence σ^n such that the transition sequence $\sigma^{i-1} = t_1 \dots t_{i-1}$ is composed by transitions in $\hat{T}_2$ only. According to the same assumption, we can exchange the firing order of t_i and σ^{i-1} and get the transition sequence $t_i t_1 \dots t_{i-1}$ which is feasible at m.

But then, the sequence $t_i t_1 \dots t_{i-1} t_{i+1} \dots t_n$ is a feasible sequence leading from marking m to $m_{\mathcal{T}}$, and $m_{\mathcal{T}} \in \mathcal{R}^u_{\mathcal{T}}(tr(m, t_i)) \subseteq \bigcup_{m' \in \mathcal{M}} \mathcal{R}^u_{\mathcal{T}}(m')$. $\square$

Example Returning to the example of Section 3.3, first we notice that by setting $T_1 = \{t_2\}$ at the vanishing marking m_1 of Figure 1b, we satisfy the condition of Proposition 4: Indeed, the only transition sequence in $\hat{T}_2^*$ (i.e., not containing an element of T_1) that emanates from marking m_1, is the sequence $\sigma = t_6$, and this sequence is followed by the firing of the transition $t_2 \in T_1$. Furthermore, the transition sequence $t_2 t_6$ is also fireable from m_1, as stipulated by Proposition 4. But then, these remarks further imply the satisfaction of the condition of Eq. (6) in Proposition 2 by the set $T_1 = \{t_2\}$ and render this set a viable candidate for a refined random switch for the considered marking m_1 while foregoing the complete enumeration of the corresponding untimed reaches $\mathcal{R}^u_{\mathcal{V}}(m_1)$ and $\mathcal{R}^u_{\mathcal{T}}(m_1)$.

On the other hand, setting $T_1 = \{t_6\}$ results in the presence of the transition sequences $t_2 t_3 t_0$ and $t_2 t_0 t_3$ in the corresponding set $\hat{T}_2^*$. Both of these transition sequences lead from marking m_1 to the tangible marking m_{T1} in the STD of Figure 1b, and obviously they are not extensible by any sequence in $\hat{T}_2^* T_1$ (since the only transitions fireable in marking m_{T1} are timed transitions). Hence, in this case, the criterion of Proposition 4 correctly fails to recognize the corresponding set T_1 as a viable candidate for a refined random switch of m_1. $\square$

The requirements that are posed in the next result are stronger than the condition set of Proposition 4, but as remarked in the opening discussion of this section, their primary value lies in the fact that they are much easier to verify than the condition of Eq. (6) and the condition that is stipulated in Proposition 4. More specifically, while the conditions of Eq. (6) and Proposition 4 are tested on the untimed reach of any given marking $m \in \mathcal{R}_{\mathcal{V}}(\mathcal{N})$, and therefore, their assessment has an exponential worst-case complexity w.r.t. the size of the underlying GSPN $\mathcal{N}$, the conditions that are stipulated by the next proposition are polynomially verifiable; this computational efficiency comes from the structural nature of these conditions.

Proposition 5 *Consider a vanishing marking m of a GSPN $\mathcal{N}$, a partition (T_1, T_2) of $\mathcal{E}_u(m)$, and further suppose that:*

i. $\exists \hat{t} \in T_1$ *such that* $\forall t \in T_2$, $\bullet \hat{t} \cap \bullet t = \emptyset$;

ii. $\forall t \in T_2$ *and* $\forall p \in P$, $p \in t\bullet \implies p\bullet \cap T_u = \emptyset$.

Then, the set T_1 satisfies the condition of Eq. (6) *in Proposition 2.*

Proof: We shall prove the result of Proposition 5 by showing that the two conditions stated in this proposition are sufficient to satisfy the requirements of Proposition 4.

We start by noticing that condition (ii) guarantees that the firing of any transition sequence $\sigma \in T_2^*$ does not add any tokens to the input places of any untimed transition. Therefore, under this condition, $\hat{T}_2 = T_2$, where the set $\hat{T}_2$ is defined in the statement of Proposition 4, and we only need to show that $\forall \sigma \in T_2^*,\ \exists \hat{\sigma} t_1 \in T_2^* T_1$ such that both sequences $\sigma\hat{\sigma}t_1$ and $t_1\sigma\hat{\sigma}$ are feasible at m.

Consider any transition sequence $\sigma = t_1 t_2 \dots t_n \in T_2^*$ feasible at m. From condition (i), $\bullet\hat{t} \cap \bullet t_i = \emptyset,\ \forall i \in \{1, 2, \dots, n\}$. Therefore, firing σ does not decrease the number of tokens in any input place of $\hat{t}$, and $\hat{t}$ is enabled at the marking m' that results from the firing of σ. In other words, $\sigma\hat{t}$ is feasible at m. The feasibility of $\hat{t}\sigma$ can be proved in a similar manner. Therefore, the condition of Proposition 4 is met by setting $\hat{\sigma} = \epsilon$ and $t_1 = \hat{t}$. $\square$

In [13] we employ the condition of Proposition 5 in order to define a very efficient and pertinent policy space $\tilde{\Pi}(\delta)$ for a particular GSPN class that models the resource allocation taking place in capacitated re-entrant lines (CRLs), i.e., re-entrant lines with finite buffering capacity at their workstations [15]. The CRL concept is also employed in the next section in order to demonstrate the efficacy of the random-switch refinement process that is developed in this work.

5 Some computational results

The re-entrant line is a workflow-modeling abstraction that has generated a lot of interest in the area of stochastic scheduling, since it constitutes one of the simplest workflow structures that gives rise to nontrivial scheduling problems [9]. Furthermore, these lines have been utilized as a modeling abstraction for the operations taking place in the fabrication of the various chips by modern semiconductor manufacturing [10]. In its basic definition, a re-entrant line consists of a number of single-server workstations that support the execution of a single process type. Instances of this process type follow a fixed path among the system workstations, that defines the corresponding process plan. However, this process plan involves some revisits to the line stations, and therefore, process instances that are waiting for processing at the various workstations are differentiated by their processing stage. This differentiation renders nontrivial the allocation of the free(d) station server to the waiting processes, and defines some interesting scheduling problems.

However, the past studies of the re-entrant line scheduling problem have addressed this problem under the assumption of infinite buffering capacity at every station. At the same time, the work of [15] has shown that the blocking effects that arise from the presence of finite buffers in those lines negate the aforementioned past results and renders the CRL scheduling problem an open issue in the current scheduling literature. In this section, we employ the CRL scheduling context in order to demonstrate the efficacy of the random-switch refinement process that is developed in this work.[10] More specifically, in the first part of this section we present another example that will provide a more concrete demonstration of (i) the problem addressed in this work, and (ii) the representational gains regarding the target policy space of this problem that are attained by Algorithm 1 of Section 3.3. In the second part of this section, we also report the results of a more extensive computational experiment that further demonstrate and assess the aforementioned gains.

[10] A detailed treatment of the CRL scheduling problem through the methodological framework that is pursued by the considered research program can be found in [13].

5.1 Example: Applying the presented methodology to a CRL scheduling problem

The considered CRL In this example, we consider the CRL depicted in Figure 2. This re-entrant line consists of two workstations, WS_1 and WS_2, and an I/O port that interfaces it with the rest of its operational environment. Each workstation has a single server and two units of buffering capacity. Jobs visiting each of the two workstations are accommodated to one of the available buffer slots, and receive processing by the workstation server in situ (i.e., while staying in their allocated buffer slots). A robotic manipulator supports the necessary material handling functions, and integrates the entire facility to a fully automated cell. The process route that is executed by the job instances that visit this cell is also annotated in the figure; since workstation WS_1 is visited twice by any running job, the considered layout constitutes a re-entrant line. In the sequel, we shall also use the notation $\mathcal{W}(j),\ j = 1, 2, 3$, to denote the workstation that supports the j-th processing stage of the depicted process route; e.g., $\mathcal{W}(1) = WS_1$. Finally, we assume that the processing times for each of the three processing stages are exponentially distributed with corresponding rates $\mu_j,\ j = 1, 2, 3$.[11]

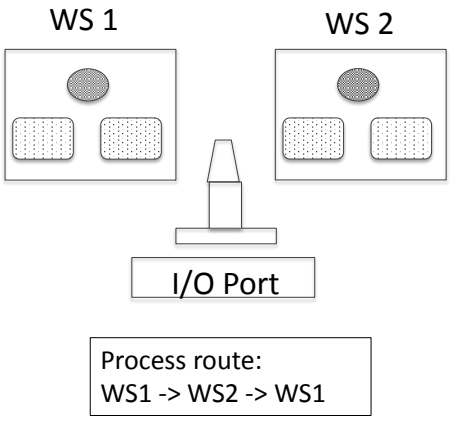


Figure 2: An example CRL.

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 workstations in combination with the re-entrant nature of its material flow. For instance, if the buffer of workstation WS_1 is allocated to two jobs in their first processing stage and the buffering capacity of workstation WS_2 is also fully allocated, then it is impossible for the cell robot to advance any of these jobs to their next stage, and the operation of the system will be permanently stalled, or *deadlocked*. This deadlocking problem can be prevented by the application of an appropriate liveness-enforcing supervisor (LES). The reader is referred to [16, 17] for a comprehensive exposition of the current theory on the synthesis of effective and efficient LES for sequential RAS, including the RAS corresponding to the considered CRL. Many of these LES take the convenient form of a set of linear inequalities that control the number of process instances situated at certain subsets of the line workstations, and as discussed further below, they admit a very efficient and elegant PN-based representation. For the simple example depicted in Figure 2, it can be easily checked that deadlocks can be avoided as long as the combined number of jobs situated at workstations WS_1 and WS_2 for the respective execution of their first and second processing stages are kept less than or equal to three. Furthermore, this restriction implements the "maximally permissive" LES for this particular CRL, in the sense that the enforced constraint blocks only transitions to CRL states that contain a deadlock or states from which deadlock is unavoidable.

[11] This assumption is in line with the standard timed semantics of the GSPN model, but more general distributions for the processing times involved can be approximated to any desired degree of accuracy through the methods of stages [5].

Table 1: The semantics encoded by the GSPN of Figure 1a.

Node	Description
p_0 (resp., p_3, p_6)	A job receiving processing at workstation $\mathcal{W}(1)$ (resp., $\mathcal{W}(2)$, $\mathcal{W}(3)$)
p_1 (resp., p_4)	A job having completed processing at workstation $\mathcal{W}(1)$ (resp., $\mathcal{W}(2)$)
p_2 (resp., p_5)	A job waiting for processing initiation at workstation $\mathcal{W}(2)$ (resp., $\mathcal{W}(3)$)
p_7 (resp., p_9)	Server availability at workstation WS_1 (resp., WS_2)
p_8 (resp., p_{10})	Buffer slot availability at workstation WS_1 (resp., WS_2)
p_{11}	A "monitor" place enforcing the marking inequality that defines the maximally permissive LES, according to the theory of [6, 8]
t_0	LES-admissible loading and initiation of processing of a new job at workstation $\mathcal{W}(1)$ by allocating to this job a buffer slot and the station server; appropriate updating of the LES-enforcing monitor
t_1, (reps., t_4)	Completion of a job processing at workstation $\mathcal{W}(1)$ (resp., $\mathcal{W}(2)$), and release of the corresponding server
t_2, (reps., t_5)	LES-admissible advancement of a completed job at workstation $\mathcal{W}(1)$ (resp., $\mathcal{W}(2)$) to the next workstation by allocating and de-allocating the relevant buffer slots; appropriate updating of the LES-enforcing monitor
t_3 (resp., t_6)	Initiation of a job processing at workstation $\mathcal{W}(2)$ (resp., $\mathcal{W}(3)$), by allocating the corresponding server
t_7	Completion of a job processing at workstation $\mathcal{W}(3)$ and unloading of the job from the system; release of the corresponding server and buffer slot

GSPN-based modeling of the considered CRL The marked GSPN modeling the operational dynamics of the CRL of Figure 2 is that depicted in Figure 1a, with its initial marking m_0 set to $[0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 1, 2, 3]$. The semantics encoded by this GSPN are detailed in Table 1. We should also note that the provided GSPN model does not represent explicitly the allocation of the robotic manipulator since (i) in order to simplify the exposition of the considered example, we further assume that transport times are negligible, and (ii) it can be easily checked that if the allocation of the workstation buffers is kept deadlock-free, then the robotic manipulator has only a supportive role to the overall cell functionality and it cannot be a cause of any further problematic behavior. Finally, the reader can check that the aforementioned initial marking m_0 corresponds to the "empty" state of the considered CRL, i.e., the state in which all the workstations of the line are idle and empty of any jobs.

In the GSPN of Figure 1a, untimed transitions are represented by black bars and timed transitions are represented by white bars. Furthermore, as revealed by the juxtaposition of Figure 1a and Table 1, timed transitions encode the completion of the processing activity at the line workstations and the corresponding delays that are incurred by this activity; hence, the firing rates that are associated with the timed transitions t_1, t_4 and t_7 are the corresponding rates μ_j, $j = 1, 2, 3$, of the exponential distributions that characterize the processing times for the three processing stages.[12] On the other hand, in the depicted GSPN, the set of the untimed transitions, $T_u = \{t_0, t_2, t_3, t_5, t_6\}$, models the job advancement through the various stages of the underlying process plan, and the corresponding resource allocation function under the control of the maximally permissive LES that was described in the previous paragraphs.

[12] In the case of processing stages with more general distributions for their processing times, the method of stages will substitute the corresponding timed transition with a Markovian subnet that will model the approximating "phase type" distribution.

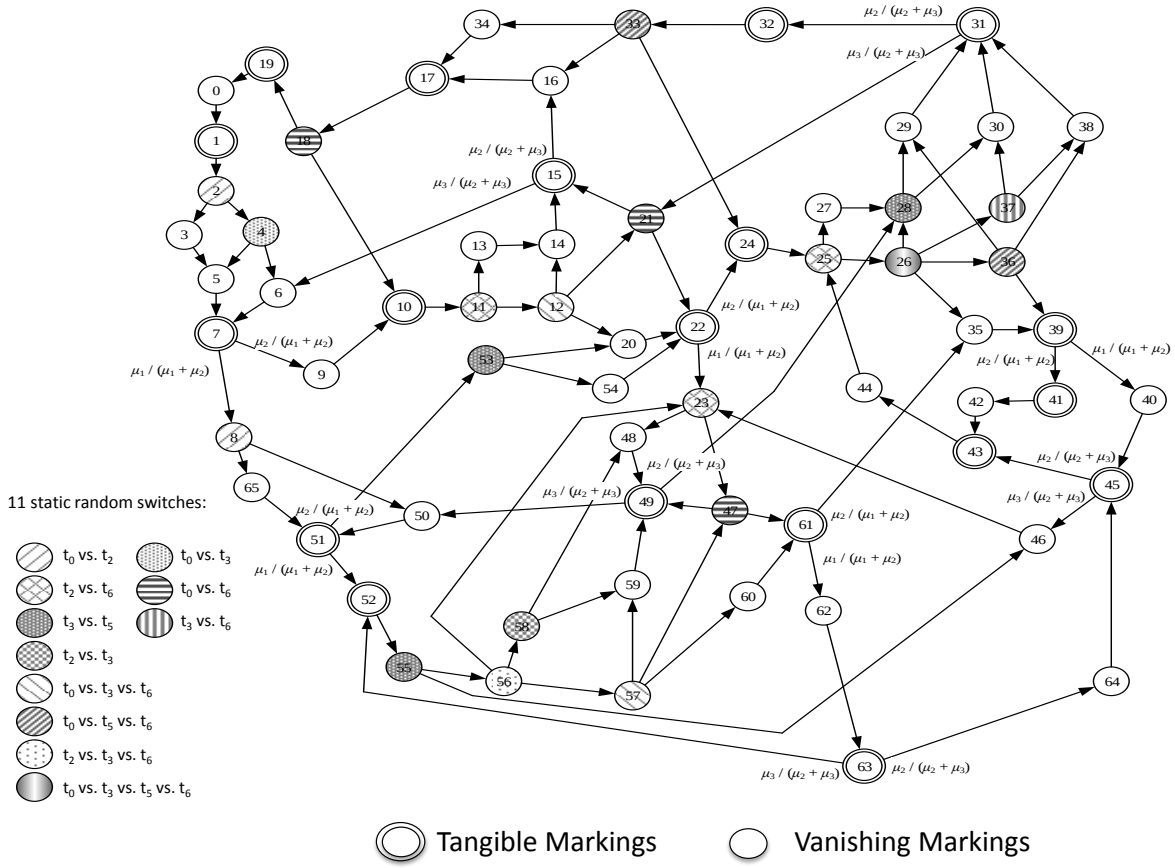


Figure 3: The semi-Markov process modeling the timed dynamics of the example CRL of Figure 2 and the corresponding scheduling problem.

The semi-Markov-based modeling of the timed dynamics of the considered CRL and the corresponding scheduling problem The timed dynamics of the GSPN of Figure 1a are represented by the semi-Markov process of Figure 3. This figure depicts the branching probabilities at the various tangible markings of the process, as determined by the corresponding exponential races. It is further assumed that at the tangible markings activating the job-unloading transition t_7, reward is collected with an instantaneous rate of μ_3; in this way, the steady-state average reward of the considered semi-Markov process will model the (long-term) throughput of the line. Furthermore, the branching probabilities at the various vanishing markings must be determined so that this long-term throughput is maximized. Finally, Figure 3 also depicts the coupling among the random switches of this semi-Markov process that is introduced by the notion of "static" random switches that is defined by Eq. (2). It can be checked that this coupling reduces the number of the decision variables that must be employed in the formulation of the aforementioned scheduling problem by means of the Eqs (3)–(5) from 27 to 16, a considerable reduction even for this small example.

Figure 4 presents the semi-Markov process and the static random switches that are defined for the CRL of Figure 2 through the refinement process of Algorithm 1. The juxtaposition of the untimed part of the semi-Markov process depicted in Figure 4 with the untimed part of the semi-Markov process of Figure 3 reveals the very drastic simplification that has been brought about by this refinement process. More specifically, the number of the employed random switches

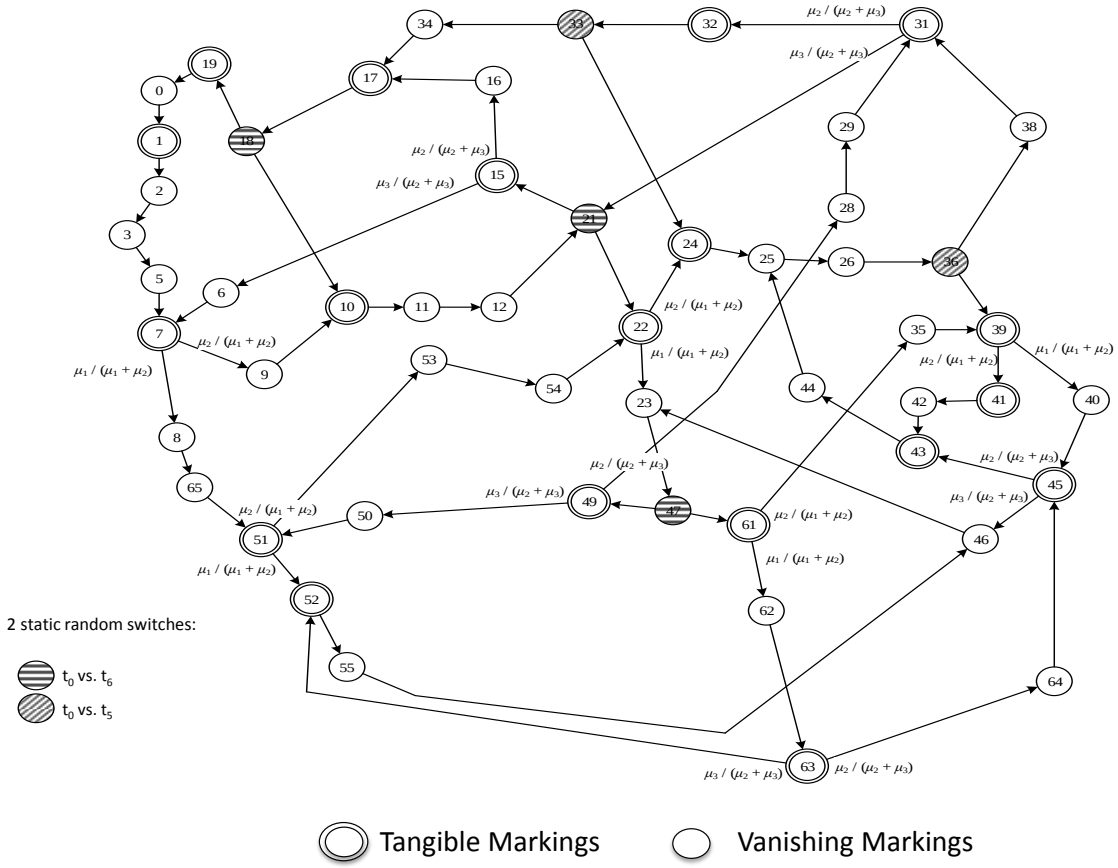


Figure 4: The semi-Markov process modeling the timed dynamics of the considered CRL and the corresponding scheduling problem after the refinement of the original set of random switches through Algorithm 1.

has been reduced from 11 to 2, and the corresponding MP formulation of Eqs (3)–(5) involves only two variables! In addition, the actual decision points in any path interconnecting any two tangible markings (i.e., the vanishing markings with more than one activated untimed transitions in those paths) are now no more than one, and these markings reflect the actual conflicts in the underlying CRL.

Finally, closing the discussion on the example presented in this section, we want to point out that, as clearly demonstrated by this example, the simplification that is exerted by the refinement process introduced in this paper is different from the simplification processes that have been pursued in the context of the more classical GSPN theory. In the latter cases[13], the key objective is to extract a simplified stochastic process (typically a CTMC) from the original semi-Markov process; but in those cases, the original semi-Markov process is fully-defined, by fixing the random switches associated with the various vanishing markings to specific (externally provided) values. From a methodological standpoint, the extraction of the target CTMC from the original semi-Markov process is based on results relating to the computation of the transient behavior and the absorption probabilities in Discrete-Time Markov chains (DTMCs). On the other hand, the simplification that is pursued in this paper is on the *structure* of the

[13]c.f., for instance, the corresponding appendix in [1], that introduced the GSPN model

original semi-Markov process with the intention of generating another semi-Markov process. The new semi-Markov process must induce a policy space $\tilde{\Pi}(\delta)$, through the set of its random switches, that carries the performance potential of the policy space $\Pi(\delta)$ that is induced by the original semi-Markov process.[14] Another differentiating attribute between the two simplification processes discussed above is that, while the extraction of the CTMC from the original semi-Markov process is perceived as an "off-line" operation simplifying the stochastic process that models the timed dynamics of the underlying GSPN and the analysis of these dynamics, the refinement process implemented by Algorithm 1 is an "on-line" process that must be embedded in the simulation optimization algorithm that will solve the NLP formulation of Eqs (3)–(5); this, in turn, generates different computational constraints and implementational requirements for each process.

5.2 A more extensive numerical experiment

In this section we report the results of a more extensive computational experiment that applied the Algorithm 1 of Section 3.3 on the 20 CRL configurations listed in Table 2.[15] The obtained results are reported in Table 3. More specifically, the first part of Table 3 reports the results of the refinement that was effected by Algorithm 1 in combination with the notion of the *static* random switches that was introduced in [12]. As explained in the earlier parts of this document, this refinement retains the one-step connectivity of the tangible markings in the CTMCs that are induced by the underlying GSPN model, and the performance potential of the optimized scheduling policies that are obtained by the corresponding NLP formulations. However, the numbers of the visited vanishing markings, and the extent of branching that takes place at these markings, are significantly reduced by the exerted refinement. Table 3 also provides the number of tangible markings for each configuration, which can function as a proxy measure for the representational and computational complexity of the original scheduling problem that is defined over the policy space $\Pi(\delta)$.[16]

The results provided in Table 3 demonstrate very vividly the dramatic reductions in the number of static random switches and in the corresponding number of the decision variables ξ that is effected by the application of Algorithm 1. These reductions subsequently have a very strong impact on the solvability of the corresponding MP formulations of Eqs (3)–(5) by means of the SA methods that are pursued in [12, 13].

The second part of Table 3 also reports, for each of the considered configurations, the average and the maximum times that were spent by Algorithm 1 for the computation of the untimed tangible reach $\mathcal{R}^u_{\mathcal{T}}(m)$ and the selection of the refined transition set $\tilde{\mathcal{E}}_u(m)$ at each visited vanishing marking m. The experiment was performed on a Window 7 computer with an AMD

[14] We remind the reader that the technical meaning of this last statement is formalized in this paper through Definition 1 and its supporting discussion, and that, as far as we can tell, the content of this definition is a novel contribution of this work, especially when considered in the context of the GSPN modeling framework and its supporting theory.

[15] Configuration 1 in Table 2 is the CRL configuration discussed in Section 5.1.

[16] As revealed by the example digraph of Fig. 1b, the acyclic subspaces of $\mathcal{R}(\mathcal{N})$ that interconnect each tangible marking m to its corresponding set of tangible markings $\mathcal{R}^1_{\mathcal{T}}(m)$, may involve a number of different paths connecting m with each element of $\mathcal{R}^1_{\mathcal{T}}(m)$, and each of these paths may involve a cascade of "decisions" modeled by the random switches of the corresponding vanishing markings. Hence, the actual number of the decision variables ξ for the scheduling problems that are formulated on the original policy space $\Pi(\delta)$ typically will be higher than the number of tangible markings that is provided in the second column of Table 3, sometimes by an order of magnitude.

Table 2: The CRL configurations employed in the numerical experiment that is reported in Section 5.2.

Config.	work-stations	# of job stages (JS) and job routes	workstation buffering capacities
1	2	3JS ($WS_1 \to WS_2 \to WS_1$)	$(B_1, B_2) = (2, 2)$
2	2	3JS ($WS_1 \to WS_2 \to WS_1$)	$(B_1, B_2) = (1, 2)$
3			$(B_1, B_2) = (3, 2)$
4			$(B_1, B_2) = (4, 4)$
5			$(B_1, B_2) = (10, 10)$
6	3	4JS ($WS_1 \to WS_2 \to WS_3 \to WS_1$)	$(B_1, B_2, B_3) = (1, 2, 2)$
7			$(B_1, B_2, B_3) = (3, 2, 2)$
8			$(B_1, B_2, B_3) = (4, 3, 2)$
9			$(B_1, B_2, B_3) = (5, 5, 6)$
10	4	7JS ($WS_1 \to WS_2 \to WS_4 \to WS_1 \to WS_2 \to WS_3 \to WS_1$)	$(B_1, B_2, B_3, B_4) = (3, 2, 1, 2)$
11	3	5JS ($WS_1 \to WS_2 \to WS_3 \to WS_1 \to WS_2$)	$(B_1, B_2, B_3) = (3, 4, 3)$
12	3	5JS ($WS_1 \to WS_2 \to WS_3 \to WS_2 \to WS_3$)	$(B_1, B_2, B_3) = (3, 3, 3)$
13	3	5JS ($WS_1 \to WS_2 \to WS_1 \to WS_3 \to WS_2$)	$(B_1, B_2, B_3) = (3, 4, 1)$
14			$(B_1, B_2, B_3) = (2, 2, 2)$
15	3	6JS ($WS_1 \to WS_2 \to WS_3 \to WS_1 \to WS_2 \to WS_3$)	$(B_1, B_2, B_3) = (2, 3, 2)$
16			$(B_1, B_2, B_3) = (2, 2, 2)$
17	4	7JS ($WS_1 \to WS_2 \to WS_4 \to WS_1 \to WS_2 \to WS_3 \to WS_1$)	$(B_1, B_2, B_3, B_4) = (3, 3, 3, 3)$
18	5	7JS ($WS_1 \to WS_2 \to WS_1 \to WS_3 \to WS_4 \to WS_5 \to WS_4$)	$(B_1, B_2, B_3, B_4, B_5) = (2, 2, 2, 3, 3)$
19	4	8JS ($WS_1 \to WS_2 \to WS_3 \to WS_2 \to WS_3 \to WS_4 \to WS_3 \to WS_4$)	$(B_1, B_2, B_3, B_4) = (3, 3, 3, 3)$
20	5	8JS ($WS_1 \to WS_2 \to WS_3 \to WS_2 \to WS_3 \to WS_4 \to WS_5 \to WS_3$)	$(B_1, B_2, B_3, B_4, B_5) = (3, 3, 3, 3, 3)$

Table 3: Comparison of the numbers of static random switches (R.S.) and decision variables (D.V.) for the CRL configurations of Table 2 that result from (i) no refinement, and (ii) refinement using Algorithm 1. The table also provides some statistics for the calculation of the untimed tangible reach (UTR) and the selection of the refined random switch by Algorithm 1 at each visited vanishing marking.

Conf.	Tang. Mark.	No Refining		Algor. 1		UTR Calculation			R.S. Selection		
		Num. of R.S.	Num. of D.V.	Num. of R.S.	Num. of D.V.	Number of Calls	Mean Time (μs)	Max Time (ms)	Number of Calls	Mean Time (μs)	Max Time (ms)
1	19	11	16	2	2	30	6.50	0.03	47	0.46	0.00
2	7	4	4	1	1	10	5.41	0.01	18	0.06	0.00
3	33	12	18	2	2	54	6.55	0.04	85	0.50	0.00
4	87	12	18	2	2	154	7.05	0.07	265	0.66	0.00
5	579	12	18	2	2	1,102	9.01	0.25	139	0.88	0.01
6	42	20	27	1	1	82	6.44	0.04	531	0.14	0.00
7	148	35	61	2	2	332	7.08	0.13	1,175	0.35	0.00
8	301	35	61	2	2	716	7.30	0.16	2,343	0.36	0.00
9	1,593	38	68	2	2	4,212	7.79	0.39	7,644	0.44	0.02
10	4,245	397	1,104	13	15	11,288	10.49	1.48	17,902	0.73	0.03
11	2,511	113	251	4	4	6,234	11.39	0.70	11,397	0.82	0.03
12	1,162	89	181	4	4	2,991	9.04	0.45	4,879	0.70	0.03
13	1,045	81	165	5	5	2,403	9.38	0.52	4,158	0.77	0.03
14	261	73	136	5	5	587	7.28	0.16	937	0.49	0.01
15	1,518	188	414	6	6	3,767	9.11	0.61	6,270	0.58	0.03
16	694	164	342	6	6	1,635	8.49	0.44	2,583	0.55	0.02
17	41,097	579	1,827	15	17	126,946	12.79	2.83	232,901	1.08	0.05
18	20,389	191	497	4	4	75,276	9.60	1.95	121,013	0.61	0.03
19	98,133	771	2,456	19	22	335,341	12.62	7.44	556,492	1.01	0.14
20	198,231	739	2,342	14	17	775,744	13.41	5.19	1,366,010	1.13	0.08

Opteron tm Processor (model: 6376, frequency: 2.30 GHz) and a RAM of 3.00 GB. It can be checked in the provided data that, for all configurations, the quoted averages are in the order of microseconds for both stages of the performed computation. And while there is an increase of these averages for larger configurations, this increase is not explosive (as is the case with the size of the corresponding state spaces); as explained in earlier parts of this document, this last result is due to (i) the "local" nature of the performed computation when considered in the context of the STD of the underlying semi-Markov process, and (ii) the typically small number of the untimed transitions activated at m that are actually in a conflicting relation according to the characterizations of this notion that were provided in Sections 3.3 and 4.

6 Conclusions

This paper has developed a methodology that complements the results of [12] regarding the GSPN-based performance modeling and control of complex resource allocation systems, by enabling the specification of some refined policy spaces that include policies with a much more compact representation compared to the primary policy space that was employed in [12]. More specifically, these refined policy spaces employ a smaller number of random switches, with smaller supports, than the set of random switches that defines the policy space of [12]. On the other hand, the new policy spaces retain the basic structure of the CTMC that defines the performance of the considered GSPN models, and therefore, under a "dynamic" interpretation of the employed random switches, they carry the performance potential of the original MDP formulation of the addressed scheduling problem. Even more importantly, the random switches that are constructed in this work reveal the essential conflicts in the scheduling problem addressed, and enable an effective and fine control of the trade-off between the computational tractability of the problem formulation and the operational efficiency of the derived solutions; this last effect can be attained by the further development of mechanisms that will provide a pertinent "middle ground" between the (fully) static and the (fully) dynamic interpretation of the refined random switches that are developed in this work.

From a methodological standpoint, the presented developments lie at the intersection of the qualitative and quantitative analyses of Discrete Event Systems (DES) [5], a theme that has received only limited attention in the relevant literature.

Finally, as already mentioned, the work presented in [13] has further demonstrated and concretized the practical value of the presented developments by adapting the obtained results to the particular problem of throughput maximization in capacitated re-entrant lines.

Our future work will complement the developments presented in this paper in the following two directions: (i) First, we shall seek the enhancement of the SA algorithm presented in [12] in order to render it more robust to the gradient-estimation errors that arise in the computations of that algorithm. This endeavor will be based on the integration into the SA algorithm of [12] of some results and techniques borrowed from the area of statistical inference. (ii) The second task to be pursued in our future work concerns the further management of the existing trade-off between (a) the representational and computational tractability and (b) the performance potential of the sought scheduling policies; in more practical terms, this task will seek the development of pertinent partial decoupling techniques for the static random switches that are provided by the presented methodology.

References

[1] M. Ajmone Marsan, G. Balbo, and G. Conte. *Performance Models of Multiprocessor Systems*. The MIT Press, Cambridge, MA, 1986.

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

[3] D. P. Bertsekas. *Dynamic Programming and Optimal Control, Vol. II (4th ed.)*. Athena Scientific, Belmont, MA, 2012.

[4] X. Cao. *Stochastic Learning and Optimization*. Springer, NY,NY, 2007.

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

[6] 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.

[7] P. Glasserman and D. Yao. *Monotone Structure in Discrete-Event Systems*. John Wiley & Sons, Inc., NY,NY, 1994.

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

[9] 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.

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

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

[12] R. Li and S. Reveliotis. Performance optimization for a class of generalized stochastic Petri nets. *Discrete Event Dynamic Systems: Theory and Applications*, 25:387, 2015.

[13] R. Li and S. A. Reveliotis. Optimized scheduling of capacitated re-entrant lines. Technical Report (under development), Georgia Institute of Technology, 2015.

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

[15] 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.

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

[17] S. A. Reveliotis. A theory of sequential resource allocation systems for automation. Technical Report (submitted for publication), School of Industrial & Systems Eng., Georgia Tech, 2014.

[18] R. Su and W. M. Wonham. Supervisor reduction for discrete-event systems. *Discrete Event Dynamic Systems: Theory and Applications*, 14:31–53, 2004.