Controlled Stochastic Petri Nets

Hermann de Meer and Oliver-Rainer Düsterhöft
Department of Computer Science – University of Hamburg
Vogt-Kölln-Str. 30, 22527 Hamburg, Germany
{demeer, 1duester}@informatik.uni-hamburg.de

Abstract - A new framework for the extension of stochastic Petri nets (SPNs) is introduced in this paper. SPNs are extended by elements providing means for a *dynamic optimization* of performability measures. A new type of transition is defined offering a feature for specification of controlled switching, called *reconfiguration*, from one marking of a SPN to another marking. Optional reconfiguration transitions are evaluated in order to optimize a specified reward or cost function. The result of an analysis is provided in the output of a numerical computation, in the form of a graphical presentation of an optimal, marking dependent control strategy and the resulting performability measure when applying the optimal strategy. The extended SPNs are called *COSTPNs (COntrolled STochastic Petri Nets)*. COSTPNs are mapped on EMRMs (Extended Markov Reward Models) for a numerical analysis. Computational analysis is possible with algorithms adopted from Markov decision theory, including transient and stationary optimization. The scope of this paper is to introduce the new control structure for SPNs and to present an algorithm for the mapping of COSTPNs on EMRMs.

Keywords: Stochastic Petri nets, performability, dynamic optimization, extended Markov reward models, Markov decision theory.

1 Introduction

In the early eighties Petri nets were extended with the notion of time, for example by Molloy [10]. Marsan et al. promoted SPN for the modeling and performability analysis [9]. SPNs were extended to stochastic reward nets (SRNs) by Trivedi, Ciardo and Muppala [13]. Many more approaches and tools relating to SPNs do exist, most of them being covered in the overview paper by Haverkort and Niemegeers [6].

Often, it has been advocated, for instance by Kramer and Magee, that large scale and distributed systems should be provided with techniques allowing dynamic reconfiguration in the presence of environmental changes that affect their running conditions [7]. Adaptation seems to be a particularly promising approach for the management of communication systems with multimedia applications, as these impose challenging computational and timing requirements on resources [5]. But strategic knowledge is required in order to execute adaptation and reconfiguration actions in a most effective manner. It has been argued that quantitative modeling support could provide useful guidelines for required control decisions in this respect [2].

The extension of SPNs with features directly providing decision support for such adaptation and reconfiguration tasks seems to be a promising approach, due to the wide acceptance of SPNs for performability modeling. The proposed new control structure provides means for a specification of optional reconfigurations on the relatively high level of SPNs. As the result of an application of a numerical evaluation procedure, control strategies are computed that allow the optimization of a specified performability measure. The resulting strategies can be directly applied to control adaptation and reconfiguration.

For a numerical analysis, COSTPN have to be mapped on a model representation that allows the application of some optimization and evaluation algorithms. We propose to use EMRMs, which were originally introduced in previous work [2, 3], for this purpose. Algorithms from Markov decision theory were adopted to provide techniques for transient and stationary optimization of performability measures. For the mapping of COSTPNs on EMRMs, it is necessary to modify the standard algorithm applied in the generation of extended reachability graphs (ERGs) in ordinary SPN analysis. Besides this algorithm, it is finally shown how EMRMs can be constructed from the modified ERGs.

The features of the modeling tool, which provides a usage-friendly environment for optimization and computational experimentations, are beyond the scope of this paper [4].

In Sec. 2, we briefly repeat the concept of EMRM. The new control structure of COSTPN is introduced in Sec. 3. The full mapping algorithm of COSTPN on EMRM is discussed in detail in Sec. 4. Sec. 5 is used to demonstrate the applicability of COSTPNs on an optimization problem by means of example. Sec. 6 concludes the paper.

2 EMRM

Performability modeling makes extensive use of Markov reward models (MRMs). Let $Z = \{Z(t), t \geq 0\}$ denote a continuous time Markov chain with finite state space Ω. To each state $s \in \Omega$ a real-valued *reward rate* $r(s)$, $r : \Omega \to \mathbb{R}$, is assigned, such that if the Markov chain is in state $Z(t) \in \Omega$ at time t, then the *instantaneous reward rate* of the Markov chain at time t is defined as $X(t) = r_{Z(t)}$. In the time horizon $[0, ...t)$ the *total reward* $Y(t) = \int_0^t X(\tau)d\tau$ is accumulated. Note that $X(t)$ and $Y(t)$ depend on $Z(t)$ and on an initial state $\in \Omega$. The probability distribution function $\Psi(y, t) = P(Y(t) \leq y)$ is called the *performability*. For ergodic models the instantaneous reward rate and the time averaged total reward converge in the limit to the same overall reward rate $E[X] = \lim_{t\to\infty} E[X(t)] = \lim_{t\to\infty} \frac{1}{t}E[Y(t)]$. The introduction of reward functions provides a framework for a formal def. of a "yield measure" or a "loss measure" being imposed on the model under investigation.

EMRMs provide a framework for the combined evaluation and optimization of reconfigurable systems by introducing some new features for MRMs [2, 3]. EMRMs are the result of a marriage between Markov decision processes and performability techniques. A *reconfiguration* arc, which can originate from any Markov state of a model, specifies an optional, instantaneous state transition that can be controlled for an optimization. The resulting strategy is commonly time-dependent. The so called *branching* states provide another feature of EMRMs. No time is spent in such states, but a pulse reward may be associated with them. The introduction of branching states has motivation similar to the introduction of immediate transitions to stochastic Petri nets [9], so that branching states also are called *vanishing* states.

Reconfiguration arcs denote options to reconfigure from one state to another. At every point of time a different decision is possible for each reconfiguration arc. A strategy $\mathbf{S}(t)$ comprises a tuple of decisions for all options in the model at a particular point of time t, $0 \leq t \leq T$. Strategies can be time dependent, $\mathbf{S}(t)$, or time independent, $\mathbf{S} = \mathbf{S}(t)$. A strategy $\hat{\mathbf{S}}(t)$ is considered *optimal* if the performability measure under strategy $\hat{\mathbf{S}}(t)$ is greater equal than the performability measure under any other strategy $\mathbf{S}(t)$. With $X^{\mathbf{S}}$, $Y_i^{\mathbf{S}(t)}(t)$, $Y_i^{\mathbf{S}}(\infty)$ denoting the overall reward rate, the conditional accumulated reward and the accumulated reward until absorption gained *under strategy* $\mathbf{S}$ *or* $\mathbf{S}(t)$ respectively, a strategy $\hat{\mathbf{S}}$ or $\hat{\mathbf{S}}(t)$ is optimal, iff

- $E[Y_i^{\hat{\mathbf{S}}(t)}(t)] \geq E[Y_i^{\mathbf{S}(t)}(t)] \; \forall \mathbf{S}(t) \; \forall i \in \Omega$
  for transient optimization.
- $E[X^{\hat{\mathbf{S}}}] \geq E[X^{\mathbf{S}}] \; \forall \mathbf{S}$
  for stationary optimization (ergodic).
- $E[Y_i^{\hat{\mathbf{S}}}(\infty)] \geq E[Y_i^{\mathbf{S}}(\infty)] \; \forall \mathbf{S} \; \forall i \in \Omega$
  for stationary optimization (nonergodic).

For the optimization approach we refer back to the performability framework and provide two types of methods for the computation of optimal strategies and performance functions:

Transient optimization, where the expected accumulated reward $E[Y_i(t)]$ is used as an optimization criterion. The algorithm for an analysis within a finite period of time $[0, T)$ has been introduced in earlier work [2, 3] and is derived from Euler's method for the numerical solution of ordinary differential equations [12]. It extends an approach of Lee and Shin [11], who had introduced and proved the correctness of transient optimization for *acyclic* CTMCs (continuous time Markov chains), to *general-type* CTMCs.

Stationary optimization, which is performed for an infinite time horizon $[0, \infty)$. As optimization criteria, we distinguish between *time averaged mean total reward in steady-state*, $E[X] = E[X_i] = \lim_{t\to\infty} \frac{1}{t}E[Y_i(t)]$ for all i, where $E[X]$ is independent of initial state i for a selected strategy, and the *conditional accumulated reward until absorption*, $E[Y_i(\infty)] = \lim_{t\to\infty} E[Y_i(t)]$, which is computed for non-ergodic models containing absorbing states. In the latter case, $E[Y_i(\infty)]$ can be dependent on an initial state i. The optimization itself is performed by deployment of variants of *value iteration* or *strategy iteration* type methods [2], relying on numerical algorithms such as Gaussian elimination, Gauss-Seidel iteration, successive over-relaxation (SOR), or the Power-Method. All these methods are implemented and can be deliberately chosen for a computation or they are by default automatically selected, thereby adapting to the actual model structure.

3 COSTPN

For a dynamic optimization of performability measures, a new feature is introduced to SPN. It comprises a control structure that allows one to specify a controlled switching between markings of a SPN. Such a controlled switching is interpreted as a *reconfiguration* in the modeled system. A reconfiguration is modeled by the firing of a new type of transition, called a *reconfiguring* transition. The introduction of reconfiguring transitions leads to a new modeling tool, called COSTPN, and provides a way to combine the classical performability modeling of SPNs with the option to dynamically optimize measures.

In the following, we discuss the enabling and the firing rule of reconfiguring transitions for COSTPN first. Then, the formal def. of COSTPN is presented. The formal def. reflects the properties of the enabling rule of reconfiguring transitions. The *enabling rule* of reconfiguring transitions is applied in the *model generation phase*, in which an EMRM is constructed from the COSTPN. In the *model evaluation* phase, in which the constructed EMRM is computationally analyzed, the *firing rule* of reconfiguring transitions is applied in order to optimize a given performability measure.

3.1 Enabling and firing rule of reconfiguring transitions

The markings of a SPN are partitioned in two types, *vanishing markings* and *tangible markings*. In vanishing markings of a SPN, immediate transitions are enabled and can fire, no time is spent in these markings. Immediate transitions have always higher priority than timed transitions. The other markings of a SPN are called tangible markings. In tangible markings, timed transitions can be enabled and can fire. A tangible marking is called absorbing, if no timed transition is enabled in it. We adopt these definitions to COSTPN.

The newly introduced control mechanism can only be applied in tangible markings, but reconfiguration itself is assumed to be instantaneously performed. These properties are reflected by the *enabling rule* of reconfiguring transitions. Reconfiguring transitions are only enabled in tangible markings. The conditions for the enabling of reconfiguring transitions are the same as the conditions for the enabling of timed transitions. In particular, immediate transitions have higher priority than reconfiguring transitions.

The *firing rule* of reconfiguring transitions is based on the aim to optimize a performability measure, which is defined through a reward structure. Reconfiguring transitions are enabled together with timed transitions and the conflict between enabled reconfiguring transitions and enabled timed transitions is solved in order to optimize a performability measure. Whenever the modeled system resides in a tangible marking, in which a reconfiguring transition is enabled, the following options are given. One option is to instantaneously reconfigure to the marking which is reached through the firing of the enabled reconfiguring transition; no timed transition can fire in the current marking in this case. Another option is to stay in the current marking and *not* to fire the enabled reconfiguring transition, so that the enabled timed transitions can fire in the current marking in their usual manner. The decision, which option to select, i.e., the optimal one, is based on the comparison of optimization criteria as described in Sec. 2. The optimization criterion is computed for all options and the one with the highest expected reward is selected.

More than one reconfiguring transition can be enabled in a tangible marking. In this case, the number of options corresponds to the number of enabled reconfiguring transitions plus one. Every firing of an enabled reconfiguring transition corresponds to one option and not to fire an enabled reconfiguring transition is another option. The decision, which of these options to select, is based on the comparison of optimization criteria.

3.2 Definition of COSTPN

Before defining COSTPNs in detail, some abbreviations are introduced. RS denotes the *reachability set* of a COSTPN, where a reachability set is the set of all markings which can be reached through the firing of transitions from an initial marking M_0. RG denotes the *reachability graph* of a COSTPN. RG is a directed graph (N,A), where the set of nodes N corresponds to the set of markings in the reachability set RS and the set of arcs A is defined by the reachability relation between the markings in RS. RS and RG are generated according to the enabling pattern immanent in a COSTPN. ERG denotes the *extended reachability graph* of the COSTPN. The ERG is transformed from the RG in order to generate an EMRM. In this transformation the stochastic elements, like rates and probabilities, of the COSTPN are used to transform the transitions of the RG into transitions of a Markov chain.

The formal def. of COSTPNs can now be presented:
Let **COSTPN** $= (PN, T_1, T_2, T_3, W, Pr, Rew)$, where
$PN = (P, T, I, O, H, M_0)$ is the underlying Petri net.
P : Set of places, $T = T_1 \cup T_2 \cup T_3$: Set of transitions,
$I \subset P \times T \times \mathbb{N}$: Set of input arcs with multiplicity,
$O \subset T \times P \times \mathbb{N}$: Set of output arcs with multiplicity,
$H \subset P \times T \times \mathbb{N}$: Set of inhibitor arcs with multiplicity,
M_0 : initial marking.
$T_1 \subseteq T$ with $T_1 \cap T_2 = \emptyset$ and $T_1 \cap T_3 = \emptyset$. T_1 is the set of timed transitions.
$T_2 \subseteq T$ with $T_2 \cap T_1 = \emptyset$ and $T_2 \cap T_3 = \emptyset$. T_2 is the set of immediate transitions.
$T_3 \subseteq T$ with $T_3 \cap T_1 = \emptyset$ and $T_3 \cap T_2 = \emptyset$. T_3 is the set of reconfiguring transitions.
$W : (T_1 \cup T_2) \to \mathbb{R}$. $W(t_i)$ is the (marking-dependent) firing rate, if t_i is timed, and the (marking-dependent) firing weight, if t_i is immediate.
$Pr : T \to \mathbb{N}$. Pr is the priority function that maps transitions onto natural numbers. The priority level of timed and reconfiguring transitions is always set to 0 and the priority level of immediate transitions is always greater than 0.
$Rew : RS \longrightarrow \mathbb{R}$ defines the reward structure of a COSTPN, that assigns a reward rate or a pulse reward to any marking of a COSTPN.

By means of the defined priority function Pr the formal def. of COSTPN reflects the properties of the enabling rule of reconfiguring transitions. The priority level of reconfiguring transitions is always lower than the priority level of immediate transitions. The reconfiguring transitions have the same priority as the timed transitions. The enabling conditions for reconfiguring transitions are the same as timed transitions.

The firing rule of reconfiguring transitions is applied in the model evaluation phase of a COSTPN in order to optimize a given performability measure.

3.3 Example of a COSTPN

A simple COSTPN example helps to clarify the basic definitions. Models of on-off sources have been widely applied for the evaluation of continuous media streams in distributed systems (in particular for voice communication). We adopt such a setting and include some provisions for performance optimization.

Often, multimedia streams are modeled in such a way that periods of silence, when no data is provided, alternate with active phases, when data is generated and sent over a communication network. To avoid network and end-systems congestion effects, traffic control must be applied as an effective measure.

It is assumed that on-periods and off-periods are described by independent, exponentially distributed random variables with parameters λ and δ, respectively. Each multimedia source in an on-period with average duration $1/\lambda$ sends out data packets with rate ρ. The congestion state of the underlying network is reflected by means of parameter γ, indicating that the mean time between overload events is a function of $1/\gamma$. In a smaller setting, like a local area network, it would likely cause further harm to add on more intensive data load on the network. Therefore, it is suggested to control locally the addition of further multimedia sources.

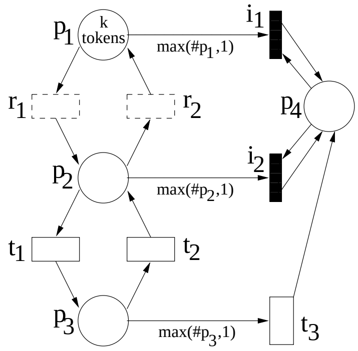
It is reasonable to assume a service provider would be interested in applying a strategy which maximizes the number of delivered data units in presence of an unreliable network due to congestion effects. According

parameter	meaning
$1/\delta$	mean off-period duration
$1/\lambda$	mean on-period duration
ρ	data transmission rate
$1/f(n,\gamma)$	mean time between overload with n active sources and network state γ

Table 1: Parameters of the COSTPN.

to the nature of multimedia applications which require to obey strict timing requirements, it is assumed that the occurrence of an overload situation would result in an unrecoverable interruption of service.

The described scenario is reflected by the COSTPN depicted in Fig. 1. Letting $n \leq k$ denote the number of multimedia sources in on-period, the mean time between overload events is described by a function of n and γ as



transition	priority	firing rate/weight	type
t_1	0	$\#p_2 * \delta$	timed
t_2	0	$\#p_3 * \lambda$	timed
t_3	0	$f(\#p_3, \gamma)$	timed
r_1, r_2	0	–	reconfiguring
i_1	1	1	immediate
i_2	2	1	immediate

place	meaning
p_1	multimedia sources excluded from transmission
p_2	multimedia sources in off-period
p_3	multimedia sources in on-period
p_4	condition place : failure in the system

Figure 1: COSTPN for the modeled system.

$1/f(n,\gamma)$. The states of multimedia sources are represented by tokens in the places p_1, p_2 and p_3. Appearance of a token in p_4 indicates a failure of the network. Each token in place p_2 models a multimedia source being in off-period and each token in place p_3 models a multimedia source in on-period. So the value of n is the number of tokens in place p_3 ($\#p_3$). Transitions t_1 and t_2 model the intermittent behavior between off- and on-periods. The firing of transition t_3 represents a possible failure of the network due to congestion effects. All tokens are removed from p_3 if such an event occurs. If a token appears in the absorbing place p_4, all remaining tokens are also removed from p_1 and p_2, via the firing of immediate transitions i_1 and i_2, such that all applications are immediately stopped.

The reconfiguring transitions r_1 and r_2 are used to control admission to the system. As long as the maximum number k of sources is not exceeded, sources can be deliberately admitted to or removed from the system while being in off-period. A token in place p_1 represents a source excluded from transmission. There is a trade-off to be considered between the number of active sources, as a measure of return to a service provider, and the risk of service disruption due to overload. The risk of failure increases as a function of the number of sources imposing load on the network while being in on-period.

Controlling admission of active sources in such way that the total number of transmitted data units is maximized, is the goal of our example study in Sec. 5.

4 Construction of an EMRM

In the following we briefly repeat the well-known algorithm to generate a MRM from a SPN [13]. We point out the differences of that algorithm to the one proposed for generating EMRMs from COSTPNs. To complete this sec., the algorithm is presented in detail.

4.1 Sketch of the SPN → MRM algorithm

To generate a MRM from a SPN the following algorithm can be used:

1. Generation of the *RG* of the SPN according to the enabling rule.
2. Transformation of the *RG* into an *ERG* by replacing the transition names labeling the arcs by the firing rate (timed transition) or by the ratio of the firing weight (immediate transition) to the firing weights of all enabled immediate transitions.
3. Association of the defined reward structure of the SPN to the states of the *ERG*. Pulse rewards assigned to vanishing states are only allowed if steady-state analysis is applied.
4. Elimination of vanishing states, where no absorbing loops of vanishing states may exist. The existence of such a loop implies a stochastic discontinuity.
5. Transformation of the transition matrix U into an infinitesimal generator matrix Q of the MRM.

$$Q_{i,j} = \begin{cases} U_{i,j} & if\ i \neq j \\ -\sum_{\forall k, k\neq j} U_{i,k} & if\ i = j \end{cases}$$

4.2 Extensions to the basic algorithm

The main differences between the algorithm SPN → MRM and COSTPN → EMRM are as follows:

1. COSTPN has *different enabling rules.*
2. During the transformation of a *RG* into an *ERG* the arcs labeled with reconfiguring transitions must be transformed into reconfiguration arcs.
3. Pulse rewards are generally allowed at vanishing states, even if transient analysis is performed.
4. For EMRM not all vanishing states need to be eliminated. Only vanishing loops must be eliminated. Vanishing loops are not allowed in EMRMs in the current implementation [8]. The pulse rewards assigned to vanishing states must be transformed before the loops can be eliminated.
5. The transformation of the transition matrix U of the *ERG* into an infinitesimal generator matrix Q has not to be done, because EMRMs may deliberately contain vanishing states with pulse rewards attached to them.

The elimination of the vanishing loops is implemented by the following algorithm:

1. Identify all vanishing loops in the *ERG*. This can be accomplished by using standard graph search algorithm for identifying strongly connected structures [1].
2. Identify all subgraphs containing a vanishing loop. Such a subgraph contains the vanishing states of the vanishing loop, entrance states z_{en}, from which the loop is entered, and exit states z_{ex}, to which the loop is left. A subgraph is described by the following transition matrix:

$$\mathbf{U}_{subgraph\ with\ loop} = \begin{pmatrix} \mathbf{C}\,\mathbf{D} \\ \mathbf{E}\,\mathbf{F} \end{pmatrix}$$

(Submatrix **C** contains the probabilities of transitions between vanishing states within the loop. Submatrix **D** contains the probabilities of transitions from each vanishing state of the loop to each exit state. Submatrix **E** contains rates and probabilities of transitions from each entrance state to each vanishing state in the loop. Matrix **E** also contains the reconfiguring transitions from Markov entrance states to each vanishing state in the loop. Since transitions between entrance and exit states need not be considered in the loop elimination procedure, submatrix **F** contains only zero elements.)

3. **FOR** every subgraph, $\mathbf{U}_{subgraph\ with\ loop}$, containing a vanishing loop **DO**:
   - (a) **FOR** any reconfiguring transition through which the loop can be reached **DO**:
     Replace the reconfiguring transition so that the loop cannot be reached through it. A new vanishing state is inserted so that the reconfiguring transition leads to the new vanishing state, from which the loop is reached with probability 1.
   - (b) Determine the set of all pairs of states $\{(z_{en}, z_{ex})\}$, where the loop is entered from entrance state z_{en} and left to exit state z_{ex}.
   - (c) **FOR** every such pair (z_{en}, z_{ex}) **DO**:
     Compute the expected accumulated reward $R_{en,ex}$ for the passage from z_{en} to z_{ex}, only passing through states of the loop, under the condition that the loop is entered from z_{en} and left to z_{ex}:

$$R_{en,ex} = \frac{R_{en\to ex}}{1 - q_{loop}} + \frac{R_{loop} * q_{loop}}{(1 - q_{loop})^2}.$$

     - $R_{en\to ex}$: pulse reward of the direct paths from z_{ex} to z_{en}. No state is visited more than once on these paths.

- q_{loop}: probability of passing through the loop ($q_{loop} < 1$, no absorbing loops are admissible in a valid ERG).
- R_{loop}: pulse reward gained by passing through the loop exactly once.

Note that $R_{en,ex} = 0 \, \forall \, (z_{en}, z_{ex})$, if no pulse reward is attached to any vanishing state of the loop.
(For a derivation of $R_{en,ex}$ consider the following infinite sum, which converges:

$$\begin{aligned} R_{en,ex} &= R_{en \to ex} + \\ & (R_{en \to ex} + R_{loop}) * q_{loop} + \\ & (R_{en \to ex} + 2 * R_{loop}) * q_{loop}^2 + \ldots \end{aligned}$$

The first term is the reward accumulated on direct paths from z_{en} to z_{ex}. The second term is the reward accumulated on paths from z_{en} to z_{ex} passing through the loop exactly once. The third term is the reward accumulated on paths from z_{en} to z_{ex} passing through the loop twice, and so on.)

(d) Eliminate the loop from $\mathbf{U}_{subgraph\,with\,loop}$ by applying the standard algorithm [13]:

$$\mathbf{U}_{subgraph\,without\,loop} = \mathbf{E} * (\mathbf{I} - \mathbf{C})^{-1} * \mathbf{D}.$$

($\mathbf{I}$ is the identity matrix with the dimension of $\mathbf{C}$.)

(e) For every pair of states (z_{en}, z_{ex}), for which $R_{en,ex} \neq 0$, a new state z_v is generated, which is associated with computed reward $R_{en,ex}$. z_v is inserted between the pair of state (z_{en}, z_{ex}). The transition from z_{en} to z_{ex} is replaced by a transition from z_{en} to z_v. The rate/probability of this new transition is the same as of the transition from z_{en} to z_{ex}, which results from the elimination of the loop in step (d). From z_v a transition leads to z_{ex} with probability 1.

(f) Replace $\mathbf{U}_{subgraph\,with\,loop}$ in the ERG by the subgraph computed in the steps (a) till (e).

4.3 Summary of the extended algorithm

The COSTPN $\rightarrow$ EMRM generation algorithm can be summarized as follows:

1. Create the RG according to the enabling rule of COSTPN as described in Sec. 3.

2. Transform the RG into an ERG by replacing the transition names/arcs as follows:

   - Replace the name of a timed transition by its firing rate.
   - Replace the name of an immediate transition by the ratio of its firing weight to the firing weights of all enabled immediate transitions.
   - Replace the arc of a reconfiguring transition by a reconfiguration arc.

3. Associate the reward structure of the COSTPN to the states of the ERG.

4. Eliminate all vanishing loops by applying the algorithm described above. Absorbing vanishing loops are not allowed in the ERG.

The resulting EMRMs can be directly analyzed by the implemented numerical algorithms [2].

5 An example study

5.1 The COSTPN model

We refer back to the example introduced in Sec. 3.3. To keep the example simple, the maximum number of multimedia sources is limited to 3. A non-linear dependence of the congestion probability on the number of active sources is assumed: $f(n, \gamma) = (\#p_3)^2 * \gamma$. The performability measure we want to use as the optimization criterion in the example study is the number of delivered data units in a given time interval. The reward function Rew specifies this performability measure by assigning the reward rate $\#p_3 * \rho$ to each tangible marking M and the pulse reward 0 to each vanishing marking M of the COSTPN.

$$Rew(M) = \begin{cases} \#p_3 * \rho & \text{if } M \text{ is tangible} \\ 0 & \text{if } M \text{ is vanishing} \end{cases}$$

The reward rate reflects that a multimedia source in on-period (a corresponding token is in place p_3) generates data units with rate ρ and sends these over a communication network.

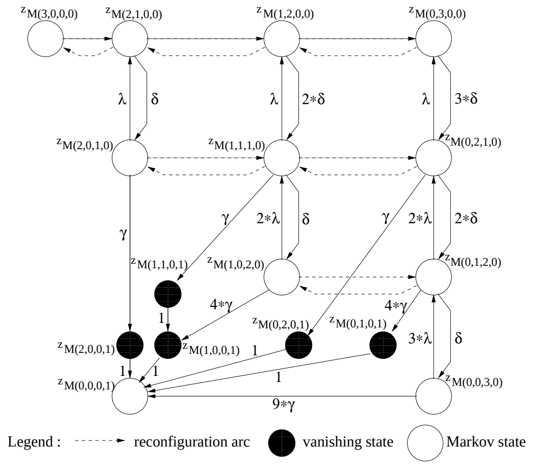
5.2 The EMRM model

As a prerequisite for optimization, an EMRM must be generated from the COSTPN.

First the RG of the COSTPN is generated from the COSTPN. The RG contains tangible and vanishing markings. The marking names $M(\#p_1, \#p_2, \#p_3, \#p_4)$ code the corresponding markings of the COSTPN, where $\#p_i$ is the number of tokens in place p_i.

Finally, the ERG in Fig. 2 is derived from the RG. The nodes, which are derived from vanishing markings, are called *vanishing states*, and the nodes, which are derived from tangible markings, are called *Markov states*. The states of the ERG are denoted by z_M in order to indicate the correspondence of the ERG states to the markings M of the COSTPN. The vanishing states are presented by dots and Markov states are presented by circles. According to the given algorithm, the reward rates are attached to the Markov states and the pulse rewards are attached to the vanishing states. The reward rates and pulse rewards are given in Fig. 2. The dashed arcs in the ERG represent the reconfiguration arcs. The arcs originating from Markov states are labeled with the firing rates of corresponding timed transitions. The arcs originating from vanishing states are labeled with a transition probability. In our example, no vanishing loop is included in the ERG. Therefore, step 4 of the generation algorithm need not to be executed.

The resulting ERG corresponds to an EMRM and can be directly analyzed by the implemented numerical algorithms.



state (z_M)	$Rew(M)$
$z_{M(2,0,1,0)}$	ρ
$z_{M(1,1,1,0)}$	ρ
$z_{M(0,2,1,0)}$	ρ

state (z_M)	$Rew(M)$
$z_{M(1,0,2,0)}$	$2*\rho$
$z_{M(0,1,2,0)}$	$2*\rho$
$z_{M(0,0,3,0)}$	$3*\rho$
others	0

Figure 2: ERG of the COSTPN in Fig. 1.

5.3 The results

A transient optimization study is presented in this sec. The investigated time horizon covers the interval $[0, T)$, $T = 1000$, and the studies are executed with the parameters $\delta = 0.25$, $\lambda = 0.4$, $\rho = 0.4$ and $\gamma = [0.00001, 0.03]$. The sending rate ρ of each active multimedia source is set equal to λ. As a result of this simplification, the number of transmitted packets equals the total number of completed on-phases.

The resulting transient control strategies are graphically represented in Fig. 3. The whole control space is separated into regions, where certain optimal strategies do apply. The boundaries between the regions correspond to curves which mark the instants of optimal strategy switches. The strategies are depicted as a function of the network congestion parameter γ and of time. Time is represented in a reverse pattern, namely as the

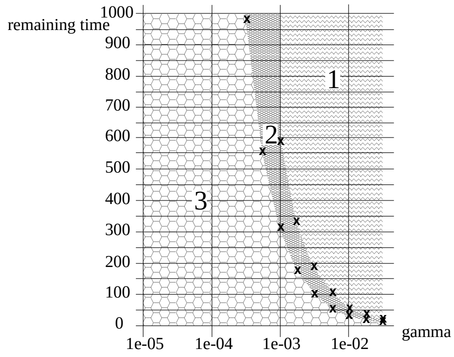


Figure 3: Transient optimization strategies.

remaining time $t' = T - t$, where T is the time horizon and t is the elapsed time. The reason for the decision space being divided into three disjunctive regions is that each one represents a class of related decisions. This results from the fact that in principle there are three types of optimal configurations: 1, 2, or 3 sources actively transmitting data packets. In each marking of the COSTPN, in which a reconfiguring transition is enabled, an optimal decision is provided in relation to the three identified regions.

To give an example, the curve between the regions 2 and 3 in Fig. 3, indicates the boundary where the optimum number of on-sources switches from 2 to 3 with respect to t' and γ. Marking $M(1, 0, 2, 0)$, for example, denotes the system state with two sources in on-period ($\#p_3 = 2$) and one source excluded from transmission ($\#p_1 = 1$). If the current situation in relation to t' and γ is classified to be in the region *above* the curve, *no* additional source should be added with respect to the marking. In terms of our COSTPN, the reconfiguring transition r_1 *must not fire* under these conditions. If the current situation is classified to be in the region *below* the curve, the alternative decision applies: The originally excluded source should also be allowed to transmit data over the network in order to maximize the total reward. In terms of our COSTPN, the reconfiguring transition r_1 *does fire* under these conditions of congestion parameter γ and time t'. Relating the results to some numbers, for a given level of congestion $\gamma = 10^{-3}$, say, a third source should be kept switched off as long as the remaining time is approximately more than 310 units of time. If less time remains for the current application, the third source should be admitted to service.

In summary, the overall interpretation of the computed strategy graph is that for sufficiently small values of γ, $\gamma < 3 * 10^{-4}$, the modeled system reconfigures to states in which as many multimedia sources as possible should be on, irrespective of time. In the interval $[3 * 10^{-4} \leq \gamma < 10^{-3})$ the strategy becomes highly time-dependent: In the long run only two sources should be accepted, while for shorter periods of remaining time three sources should be admitted. For congestion conditions $\gamma \geq 10^{-3}$ three different optimal strategies apply, depending on the current instant of time: In the long run one multimedia source would be optimal, intermediately two, and finally three. The explanation for this behavior is obvious. The greater the value of γ the greater is the risk of an overflow event. Due to the finite time horizon the risk of occurrence of an overflow event is the smaller the closer the end of an application is.

In what follows, the effect of transient versus stationary optimization is compared. We choose the stationary

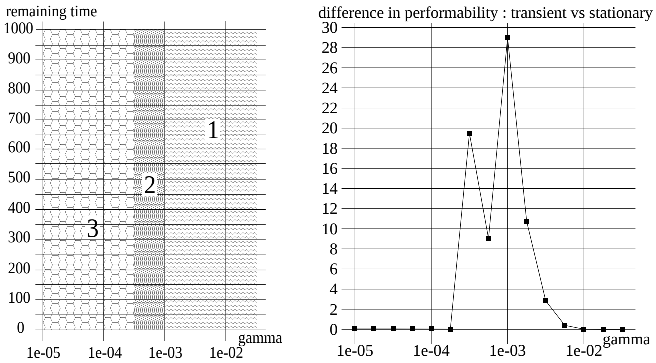


Figure 4: Stationary strategies (left) and difference in performability: transient vs. stationary (right).

strategies as shown in Fig. 4. A strategy is called stationary, if it is constantly applied, irrespective of time. To relate the impact of the strategies to each other, the chosen stationary strategies are applied in the same finite horizon, $[0, 1000)$, as the transient ones. As a result, the difference in the gained overall reward can be compared in the right part of Fig. 4. For the parameter range where the strategies are highly time-dependent, in particular in the interval $(1.5*10^{-4} < \gamma < 10^{-2})$, the transient strategies substantially outperform the stationary strategies. For $\gamma \leq 1.5*10^{-4}$ the strategies are identical and therefore no difference results in the overall performance. A similar argument applies when $\gamma \geq 10^{-2}$.

6 Conclusion

We have introduced controlled stochastic Petri nets that can be applied for simultaneous evaluation and optimization of performability models. A new type of transition was defined, providing the means for a specification of optional reconfigurations on the level of stochastic Petri nets. Different algorithms are adopted under the unifying high-level paradigm of reconfigurability, as the basis for a computation of adaptation and optimization strategies. Both transient and stationary strategies can be computed and used as a rich basis for control decisions. The algorithm for a mapping of COSTPNs on EMRMs, on which the optimization algorithms can be directly applied, was discussed in detail. The applicability of COSTPNs to an optimization problem was demonstrated by means of an example.

Further studies are necessary to fully incorporate COSTPNs into our tool environment, which provides an easily usable interface for generation, execution, and evaluation of a series of optimization experiments [4].

References

[1] Baase S.: *Computer algorithms, introduction to design and analysis*; Addison-Wesley, 2nd Edition, chapter 4 (1988).

[2] De Meer H.: "Transiente Leistungsbewertung und Optimierung rekonfigurierbarer fehlertoleranter Rechensysteme"; *Arbeitsberichte des IMMD der Universität Erlangen-Nürnberg* 25 (10) (October 1992).

[3] De Meer H., K. S. Trivedi and M. Dal Cin: "Guarded Repair of Dependable Systems"; *Theoretical Computer Science* 129, Special Issue on Dependable Parallel Computing, pp. 179–210 (July 1994).

[4] De Meer H. and H. Ševčíková: "PENELOPE - dependability evaluation and the optimization of performability"; *9th Intern. Conf. on Computer Performance Evaluation*, St. Malo, France. Lectures Notes on Computer Science (LNCS 1245), pp. 19-31 (June 1997).

[5] Fischer S., A. Hafig, G. von Bochmann and H. de Meer: "Cooperative QoS Management for Multimedia Applications"; *4th IEEE Int. Conf. on Multimedia Computing and Systems,* IEEE Computer Society Press, pp. 303-310 (June 1997).

[6] Haverkort B.R., I.G. Niemegeers: "Performability Modeling Tools and Techniques"; *Performance Evaluation 25 (1)*, pp. 17-40 (March 96).

[7] Kramer J. and J. Magee: "Dynamic configuration of distributed systems"; *IEEE Transactions on Software Engineering 11 (4)*, pp 424-436 (April 1985).

[8] Mauser H. : "Implementierung eines Optimierungsverfahrens für rekonfigurierbare Systeme"; Master thesis, University of Erlangen-Nürnberg (1990).

[9] Marsan M.A., G. Balbo, G. Conte, S. Donatelli and G. Franceschinis : *Modeling with generalized stochastic Petri nets*; John Wiley & Sons, Series in parallel computing (1995).

[10] Molloy M.K. : "On the integration of delay and throughput measures in distributed processing models"; Ph.D. thesis, University of California (1981).

[11] Lee Y.H., K.G. Shin : "Optimal Reconfiguration Strategy for a Degradable Multimodule Computing System"; *Journal of the ACM 34 (2)*, pp. 326-348 (1987).

[12] Stewart W.J. : *Introduction to the Numerical Solution of Markov Chains*; Princeton University Press (1994).

[13] Trivedi K.S., G. Ciardo and J.K. Muppala : "On the solution of GSPN reward models"; *Performance Evaluation 12*, pp. 237-253 (1991).