PERGAMON

Computers & Operations Research 30 (2003) 1005–1035

computers & operations research

www.elsevier.com/locate/dsw

Computation of bounds for transient measures of large rewarded Markov models using regenerative randomization

Juan A. Carrasco*

Departament d'Enginyeria Electrònica, Universitat Politècnica de Catalunya, Diagonal 647, plta. 9, 08028 Barcelona, Spain

Received 1 February 2001; received in revised form 1 September 2001

Abstract

In this paper we generalize a method (called regenerative randomization) for the transient solution of continuous time Markov models. The generalized method allows to compute two transient measures (the expected transient reward rate and the expected averaged reward rate) for rewarded continuous time Markov models with a structure covering bounding models which are useful when a complete, exact model has unmanageable size. The method has the same good properties as the well-known (standard) randomization method: numerical stability, well-controlled computation error, and ability to specify the computation error in advance, and, for large enough models and long enough times, is significantly faster than the standard randomization method. The method requires the selection of a regenerative state and its performance depends on that selection. For a class of models, class C′, including typical failure/repair models with exponential failure and repair time distributions and repair in every state with failed components, a natural selection for the regenerative state exists, and results are available assessing approximately the performance of the method for that natural selection in terms of "visible" model characteristics. Those results can be used to anticipate when the method can be expected to be significantly faster than standard randomization for models in that class. The potentially superior efficiency of the regenerative randomization method compared to standard randomization for models not in class C′ is illustrated using a large performability model of a fault-tolerant multiprocessor system.

Scope and purpose

Rewarded continuous time Markov models are widely used for performance, dependability and performability analysis of computer and telecommunication systems. Realistic modeling of such systems usually yields Markov models whose size exceeds the available memory resources. An approach to deal with the "largeness" problem is the use of bounding Markov models. However, even those bounding models can have very

* Tel.: +34-3-4016652; fax: +34-3-4017785.
E-mail address: carrasco@eel.upc.es (J.A. Carrasco).

PII: S0305-0548(02)00053-9

large state spaces, making important the development of efficient numerical solution techniques. Often, one is interested in transient characteristics of the system which require the transient analysis of the model. This paper generalizes a method, called regenerative randomization, for the transient analysis of Markov models, so that it can be used for the computation of bounds for the expected transient reward rate and the expected averaged reward rate transient measures with general reward rate structures including arbitrary non-negative reward rates associated with the states of the model. Examples of such measures are the unavailability of a fault-tolerant system at time t and the expected interval unavailability of a fault-tolerant system at time t. The method has the same good properties as the well-known standard randomization (also called uniformization) method: numerical stability, well-controlled computation error, and ability to specify the computation error in advance, and, for large models and long mission times, can be significantly faster than that method. Furthermore, for a class of models, class C′, including typical failure/repair models with exponential failure and repair time distributions and repair in every state with failed components, theoretical results are available assessing the performance of the method in terms of "visible" model characteristics. Those results can be used to anticipate when the generalized regenerative randomization method can be expected to be competitive for class C′ models. The generalized regenerative randomization method allows a numerically stable, with well-controlled and specifiable-in-advance error, solution of some large rewarded continuous time Markov models in affordable CPU times.

Keywords: Continuous time Markov chains; Rewarded models; Transient analysis; Randomization

1. Introduction

Homogeneous continuous time Markov chains (CTMCs) are frequently used for performance, dependability and performability modeling. The transient analysis of these models is usually significantly more costly than the steady-state analysis, and very costly in absolute terms when the CTMC is large. This makes the development of efficient transient analysis techniques for CTMCs a research topic of great interest. Commonly used methods are ODE (ordinary differential equation) solvers and randomization. Good recent reviews of these methods with new results can be found in [1–3]. The randomization method (also called uniformization) is attractive because of its excellent numerical stability and the facts that the computation error is well-controlled and can be specified in advance.[1] It was first proposed by Grassman [5] and has been further developed by Gross and Miller [6]. The method is also offered by the well-known performance, dependability and performability modeling packages [7–10]. The randomization method is based on the following result [11, Theorem 4.19]. Let $X = \{X(t); t \geqslant 0\}$ be a CTMC with finite state space Ω; let $\lambda_{i,j},\ i,j \in \Omega,\ i \neq j$ be the transition rate of X from state i to state j, and let $\lambda_i = \sum_{j \in \Omega - \{i\}} \lambda_{i,j},\ i \in \Omega$ be the output rate of state i. Consider any $\Lambda \geqslant \max_{i \in \Omega} \lambda_i$ and define the homogeneous discrete time Markov chain (DTMC) $\hat{X} = \{\hat{X}_k; k = 0, 1, 2, \ldots\}$ with same state space and initial probability distribution as X and transition probabilities $P[\hat{X}_{k+1} = j \mid \hat{X}_k = i] = P_{i,j} = \lambda_{i,j}/\Lambda,\ i \neq j,\ P[\hat{X}_{k+1} = i \mid \hat{X}_k = i] = P_{i,i} = 1 - \lambda_i/\Lambda$.

[1] The computation error has two components: truncation error and round-off error; the truncation error can be made arbitrarily small, the round-off error will have a very small relative value due to the numerical stability of the method if double precision is used. Rigorous bounds for the round-off errors have been obtained in [4] under certain conditions concerning the values that transition rates can have and assuming a special method for computing Poisson probabilities.

Let $Q=\{Q(t);t\geqslant 0\}$ be a Poisson process with arrival rate Λ ($P[Q(t)=k]=\mathrm{e}^{-\Lambda t}(\Lambda t)^k/k!$) independent of $\hat{X}$. Then, $X=\{X(t);t\geqslant 0\}$ is probabilistically identical to $\{\hat{X}_{Q(t)};t\geqslant 0\}$ (we call this the *randomization result*). The DTMC $\hat{X}$ is called the randomized DTMC of X with rate Λ. The CTMC X is called the derandomized CTMC of $\hat{X}$ with rate Λ.

Assume that a reward rate structure, $r_i\geqslant 0$, $i\in\Omega$ is defined over the state space of X. The quantity r_i has the meaning of "rate" at which reward is earned while X is in state i. Then, two useful measures to consider are the expected transient reward rate $ETRR(t)=E[r_{X(t)}]$ and the expected averaged reward rate $EARR(t)=E[\int_0^t r_{X(\tau)}\,\mathrm{d}\tau/t]$. As an example of instances of the generic $ETRR(t)$ and $EARR(t)$ measures, consider a CTMC modeling a fault-tolerant system which can be up or down, and assume that a reward rate 0 is assigned to the states in which the system is up and a reward rate 1 is assigned to the states in which the system is down. Then, $ETRR(t)$ would be the unavailability of the system at time t and $EARR(t)$ would be the expected interval unavailability at time t (i.e., the expected value of the fraction of time that the system is down in the interval $[0,t]$).

Using the facts that $X=\{X(t);t\geqslant 0\}$ and $\{\hat{X}_{Q(t)};t\geqslant 0\}$ are probabilistically identical and that $\hat{X}$ and Q are independent, we can express $ETRR(t)$ in terms of the transient regime of $\hat{X}$ as

$$\begin{aligned} ETRR(t)&=\sum_{i\in\Omega} r_i P[X(t)=i]=\sum_{i\in\Omega} r_i \sum_{k=0}^{\infty} P[\hat{X}_k=i]P[Q(t)=k]\\ &=\sum_{k=0}^{\infty}\sum_{i\in\Omega} r_i P[\hat{X}_k=i]\mathrm{e}^{-\Lambda t}\frac{(\Lambda t)^k}{k!}=\sum_{k=0}^{\infty} d(k)\mathrm{e}^{-\Lambda t}\frac{(\Lambda t)^k}{k!}, \end{aligned} \tag{1}$$

with $d(k)=\sum_{i\in\Omega} r_i P[\hat{X}_k=i]$. Denoting by $\mathbf{q}(k)=(P[\hat{X}_k=i])_{i\in\Omega}$ the probability row vector of $\hat{X}$ at step k, $\mathbf{q}(k)$, $k>0$ can be obtained from $\mathbf{q}(0)$ using

$$\mathbf{q}(k+1)=\mathbf{q}(k)\mathbf{P}, \tag{2}$$

where $\mathbf{P}=(P_{i,j})_{i,j\in\Omega}$ is the transition probability matrix of $\hat{X}$.

In a practical implementation of the randomization method, an approximate value for $ETRR(t)$, $ETRR_N^a(t)$, is obtained by truncating the series (1) so that N steps have to be given to $\hat{X}$:

$$ETRR_N^a(t)=\sum_{k=0}^{N} d(k)\mathrm{e}^{-\Lambda t}\frac{(\Lambda t)^k}{k!},$$

and, taking into account that $d(k)\leqslant r_{\max}=\max_{i\in\Omega} r_i$, the error is upper bounded as

$$ETRR(t)-ETRR_N^a(t)\leqslant r_{\max}\sum_{k=N+1}^{\infty}\mathrm{e}^{-\Lambda t}\frac{(\Lambda t)^k}{k!}.$$

A usual accuracy requirement is to limit the error in $ETRR(t)$ to a value $\leqslant\varepsilon$. Then, N is chosen as

$$N=\min\left\{m\geqslant 0:\ r_{\max}\sum_{k=m+1}^{\infty}\mathrm{e}^{-\Lambda t}\frac{(\Lambda t)^k}{k!}\leqslant\varepsilon\right\}.$$

For $EARR(t)$, noting that $EARR(t) = \int_0^t ETRR(\tau)\,d\tau/t$ and using (1),

$$EARR(t) = \frac{1}{t}\sum_{k=0}^{\infty} d(k) \int_0^t e^{-\Lambda\tau}\frac{(\Lambda\tau)^k}{k!}\,d\tau.$$

Using $\int_0^t e^{-\Lambda\tau}(\Lambda\tau)^k/k!\,d\tau = (1/\Lambda)\sum_{l=k+1}^{\infty} e^{-\Lambda t}(\Lambda t)^l/l!$ (which follows from [12, Formula 14.512] and $\sum_{l=0}^{\infty} e^{-\Lambda t}(\Lambda t)^l/l! = 1$), we get

$$EARR(t) = \frac{1}{\Lambda t}\sum_{k=0}^{\infty} d(k) \sum_{l=k+1}^{\infty} e^{-\Lambda t}\,\frac{(\Lambda t)^l}{l!}.$$

In a practical implementation, an approximate value for $EARR(t)$, $EARR_N^a(t)$, can be obtained by truncating both series so that N steps have to be given to $\hat{X}$ as

$$EARR_N^a(t) = \frac{1}{\Lambda t}\sum_{k=0}^{N} d(k) \sum_{l=k+1}^{N+1} e^{-\Lambda t}\frac{(\Lambda t)^l}{l!} = \frac{1}{\Lambda t}\sum_{k=1}^{N+1} e^{-\Lambda t}\,\frac{(\Lambda t)^k}{k!}\sum_{l=0}^{k-1} d(l)$$

and, taking into account $d(k) \leqslant r_{\max}$, the error can be upper bounded as

$$\begin{aligned} EARR(t) - EARR_N^a(t) &= \frac{1}{\Lambda t}\sum_{k=0}^{\infty} d(k) \sum_{l=k+1}^{\infty} e^{-\Lambda t}\frac{(\Lambda t)^l}{l!} - \frac{1}{\Lambda t}\sum_{k=0}^{N} d(k) \sum_{l=k+1}^{N+1} e^{-\Lambda t}\frac{(\Lambda t)^l}{l!} \\ &= \frac{1}{\Lambda t}\sum_{l=N+2}^{\infty} e^{-\Lambda t}\,\frac{(\Lambda t)^l}{l!}\sum_{k=0}^{l-1} d(k) \leqslant \frac{r_{\max}}{\Lambda t}\sum_{l=N+2}^{\infty} l e^{-\Lambda t}\frac{(\Lambda t)^l}{l!} \\ &= r_{\max}\sum_{k=N+1}^{\infty} e^{-\Lambda t}\frac{(\Lambda t)^k}{k!}. \end{aligned}$$

Being ε the absolute error with which $EARR(t)$ has to be computed, N can be chosen as

$$N = \min\left\{ m \geqslant 0:\ r_{\max}\sum_{k=m+1}^{\infty} e^{-\Lambda t}\frac{(\Lambda t)^k}{k!} \leqslant \varepsilon \right\}.$$

Stable and efficient computation of the Poisson probabilities $e^{-\Lambda t}(\Lambda t)^k/k!$ avoiding overflows and intermediate underflows is a delicate issue and several alternatives have been proposed [13–16]. Our implementation of both standard randomization and the generalized regenerative randomization method use the method described in [15, pp. 1028–1029] (see also [17]), which has good numerical stability.

For large models, the computational cost of the randomization method is roughly due to the N vector–matrix multiplications (2). The truncation parameter N increases with Λt and, for that reason, Λ is usually taken equal to $\max_{i\in\Omega}\lambda_i$. Using the well-known result [18] that $Q(t)$ has, for $\Lambda t \to \infty$, an asymptotic normal distribution with mean and variance Λt, it is easy to realize that, for large Λt and $\varepsilon \ll 1$, the required N will be $\approx \Lambda t$. If one is interested in solving the model for values of t for which Λt is very large, the randomization method will be highly inefficient.

The randomization result can also be exploited to develop methods to compute more complex measures such as the distribution of the interval availability [19–21] and the performability [22–25]. The performance of those methods also degrades as Λt increases.

Several variants of the (standard) randomization method have been proposed to improve its efficiency. Miller has used selective randomization to solve reliability models with detailed representation of error handling activities [26]. The idea behind selective randomization [27] is to randomize the model only in a subset of the state space. Reibman and Trivedi [3] have proposed an approach based on the multistep concept. The idea is to compute $\mathbf{P}^M$ explicitly, where M is the length of the multistep, and use the recurrence $\mathbf{q}(k+M)=\mathbf{q}(k)\mathbf{P}^M$ to advance $\hat{X}$ faster for steps which have negligible contributions to the transient solution of X. Since, for large Λt, the number of $\mathbf{q}(k)$'s with significant contributions is of the order of $\sqrt{\Lambda t}$, the multistep concept allows a significant reduction of the required number of vector–matrix multiplications. However, when computing $\mathbf{P}^M$, significant fill-in can occur if $\mathbf{P}$ is sparse. Adaptive uniformization [28] is a recent method in which the randomization rate is adapted depending on the states in which the randomized DTMC can be at a given step. Numerical experiments have shown that adaptive uniformization can be significantly faster than standard randomization for short to medium mission times. In addition, it can be used to solve models with infinite state spaces and not uniformly bounded output rates. Recently, it has been proposed the combination of adaptive and standard uniformization to obtain a method which outperforms both adaptive uniformization and standard randomization for most models [16]. Another recent proposal to speed up the randomization method is steady-state detection [1]. Recently, a method based on steady-state detection which gives error bounds has been developed [29]. Steady-state detection is useful for models which reach their steady state before the largest time at which the measure has to be computed.

In this paper we generalize the regenerative randomization method described in [30]. We will consider CTMCs X with finite state space $\Omega=S\cup\{f_1,f_2,\ldots,f_A\}$, $|S|\geqslant 2$, $A\geqslant 0$, where f_i are absorbing states and, either (a) all states in S are transient, or (b) S has a single trapping component[2] and the chosen regenerative state r belongs to that component, with a reward rate structure $r_i\geqslant 0$, $i\in\Omega$, with different reward rates assigned to the A absorbing states, will assume that all states are reachable from some state with non-null initial probability, and will consider the problem of computing the measures $ETRR(t)$ and $EARR(t)$. Also, to simplify the discussion, we will assume that X has some transition rate from r to $S'=S-\{r\}$. That condition can, however, be easily circumvented in practice by adding, if X has no transition rate from r to S', a tiny transition rate $\lambda\leqslant 10^{-10}\varepsilon/(2r_{\max}t_{\max})$ from r to some state in S', where ε is the allowed error, $r_{\max}=\max_{i\in\Omega}r_i$ and $t_{\max}$ is the largest time at which the measure has to be computed, with a negligible impact on the measure $\leqslant 10^{-10}\varepsilon$, $t\leqslant t_{\max}$.[3] In [30], only the measure $ETRR(t)$ for the particular case $r_i=0$, $i\in S$ and $A\geqslant 1$ was considered. The assumed structure for X covers bounding CTMC models which are useful when an exact, complete CTMC has unmanageable size. Those bounding models have a

[2] Two states of a CTMC i, j are strongly connected if there are paths in the state transition diagram of the CTMC from i to j and from j to i; a state is strongly connected with itself; a component is a maximal subset of strongly connected states; a component is trapping if no state of the component has transition rates to states outside the component.

[3] Letting $\mathbf{p}(\lambda,t)$ the probability distribution column vector of X at time t as a function of λ, we have [30] $\|\mathbf{p}(\lambda,t)-\mathbf{p}(0,t)\|_1\leqslant 2\lambda t$, which implies an absolute error in both $ETRR(t)$ and $EARR(t)\leqslant 2r_{\max}\lambda t\leqslant 10^{-10}\varepsilon(t/t_{\max})\leqslant 10^{-10}\varepsilon$, $t\leqslant t_{\max}$.

state space $\Omega = B \cup \{f_1\}$, where B is a portion of the state space of the complete CTMC and f_1 is an absorbing state, have the same transition rates between states in B and same initial probability distribution in B as the complete CTMC model, transition rates from states $i \in B$ to f_1 equal to the transition rates of the complete CTMC model from states $i \in B$ to the subset including the states outside B, and initial probability in f_1 equal to the initial probability of the complete CTMC model in the subset including the states outside B. Then, assigning to the states in B of the bounding model the same reward rates as in the complete model, and assigning to f_1 a lower bound for the reward rate of any state of the complete model, the $ETRR(t)$ and $EARR(t)$ measures of the bounding model lower bound, respectively, the measures $ETRR(t)$ and $EARR(t)$ of the complete model. If a reward rate upper bounding the reward rate of any state of the complete model is assigned instead to f_1, then the measures $ETRR(t)$ and $EARR(t)$ of the bounding model upper bound, respectively, the measures $ETRR(t)$ and $EARR(t)$ of the complete model.

The basic idea in regenerative randomization is to obtain a truncated transformed model of potentially smaller size than the original model by characterizing with enough accuracy the behavior of the original model from S' up to hit of state r or a state f_i and from r until next hit of r or a state f_i, where $r \in S$ is the so-called "regenerative" state,[4] and solve the truncated transformed model by standard randomization. The performance of the method depends, of course, on the selection of the regenerative state. The method offers the same good properties as standard randomization: numerical stability, well-controlled computation error, and ability to specify the computation error in advance, and can be significantly faster than standard randomization. The rest of the paper is organized as follows. Section 2 develops and describes the generalized method. Section 3 establishes the so-called benign behavior of the method, discusses qualitatively the efficiency of the method compared with that of standard randomization, and for a class of models, class C′, including typical failure/repair models with exponential failure and repair time distributions and repair in every state with failed components, gives stronger theoretical results assessing the efficiency of regenerative randomization in terms of "visible" model characteristics. Section 4 illustrates the potentially superior performance of the regenerative randomization method compared to standard randomization for models not in class C′ using a large performability model of a fault-tolerant multiprocessor system. Section 5 presents the conclusions. Finally, the appendix includes some proofs. Throughout the paper we will make reference to results formally proved in [30] for the case $A \geqslant 1$. Those results trivially extend to the more general case $A \geqslant 0$ considered in this paper.

2. The method

In the remaining of the paper we will use the notation $\lambda_{i,B} = \sum_{j \in B} \lambda_{i,j}$, where $B \subset \Omega - \{i\}$, and $P_{i,B} = \sum_{j \in B} P_{i,j}$, where $B \subset \Omega$. Also, given a DTMC Y, we will denote by $Y_{m:n}c$ the predicate which is true when Y_k satisfies condition c for all k, $m \leqslant k \leqslant n$ (by convention $Y_{m:n}c$ will be true for $m > n$) and by $\#(Y_{m:n}c)$ the number of indices k, $m \leqslant k \leqslant n$, for which Y_k satisfies condition c. Let $\alpha_i = P[X(0) = i] = P[\hat{X}_0 = i]$. We will use the notation $\alpha_B = \sum_{i \in B} \alpha_i$. To build the truncated transformed model, two transient DTMCs, Z, Z', have to be stepped, in general.

[4] State r is called "regenerative" because the times at which X enters r are regeneration points of X.

The first one, $Z=\{Z_k;k=0,1,2,\ldots\}$, is defined from a version of $\hat{X}$, $\hat{X}'$, with initial probability distribution concentrated in state r, as

$$Z_0=r,$$

$$Z_k=\begin{cases} i\in S'\cup\{f_1,f_2,\ldots,f_A\} & \text{if } \hat{X}'_{1:k}\neq r\wedge\hat{X}'_k=i \\ a \text{ if } \#(\hat{X}'_{1:k}=r)>0 \end{cases}, \quad k>0. \tag{3}$$

The DTMC Z has state space $S\cup\{f_1,f_2,\ldots,f_A,a\}$, where f_i and a are absorbing states and, given the assumed structure for X, all states in S are transient, and its (possibly) non-null transition probabilities are

$$P[Z_{k+1}=j\,|\,Z_k=i]=P_{i,j},\ i\in S,\ j\in S'\cup\{f_1,f_2,\ldots,f_A\}, \tag{4}$$

$$P[Z_{k+1}=a\,|\,Z_k=i]=P_{i,r},\quad i\in S, \tag{5}$$

$$P[Z_{k+1}=f_i\,|\,Z_k=f_i]=P[Z_{k+1}=a\,|\,Z_k=a]=1,\quad 1\leqslant i\leqslant A. \tag{6}$$

The DTMC $Z'=\{Z'_k;k=0,1,2,\ldots\}$ is defined as

$$Z'_k=\begin{cases} i\in S'\cup\{f_1,f_2,\ldots,f_A\} & \text{if } \hat{X}_{0:k}\neq r\wedge\hat{X}_k=i \\ a \text{ if } \#(\hat{X}_{0:k}=r)>0. \end{cases} \tag{7}$$

The DTMC Z' has state space $S'\cup\{f_1,f_2,\ldots,f_A,a\}$, where f_i and a are absorbing states and, given the assumed structure for X, all states in S' are transient. The initial probability distribution of Z' is $P[Z'_0=i]=\alpha_i,\ i\in S'\cup\{f_1,f_2,\ldots,f_A\},\ P[Z'_0=a]=\alpha_r$, and its (possibly) non-null transition probabilities are:

$$P[Z'_{k+1}=j\,|\,Z'_k=i]=P_{i,j},\quad i\in S',\ j\in S'\cup\{f_1,f_2,\ldots,f_A\}, \tag{8}$$

$$P[Z'_{k+1}=a\,|\,Z'_k=i]=P_{i,r},\quad i\in S', \tag{9}$$

$$P[Z'_{k+1}=f_i|Z'_k=f_i]=P[Z'_{k+1}=a|Z'_k=a]=1,\quad 1\leqslant i\leqslant A. \tag{10}$$

Let $\pi_i(k)=P[Z_k=i]$, $\pi'_i(k)=P[Z'_k=i]$ and consider the row vectors $\boldsymbol{\pi}(k)=(\pi_i(k))_{i\in S}$ and $\boldsymbol{\pi}'(k)=(\pi'_i(k))_{i\in S'}$. Let $\mathbf{P}_Z$ be the transition probability matrix of Z restricted to S. Let $\mathbf{P}_{Z'}$ be the transition probability matrix of Z' restricted to S'. Vector $\boldsymbol{\pi}(0)$ has components $\pi_r(0)=1,\ \pi_i(0)=0,\ i\in S'$. From $\boldsymbol{\pi}(0)$, $\boldsymbol{\pi}(k)$, $k>0$ can be obtained using

$$\boldsymbol{\pi}(k+1)=\boldsymbol{\pi}(k)\mathbf{P}_Z.$$

Vector $\boldsymbol{\pi}'(0)$ has components $\pi'_i(0)=\alpha_i,\ i\in S'$. From $\boldsymbol{\pi}'(0)$, $\boldsymbol{\pi}'(k)$, $k>0$ can be obtained using

$$\boldsymbol{\pi}'(k+1)=\boldsymbol{\pi}'(k)\mathbf{P}_{Z'}.$$

In order to simplify the discussion, we will assume that the randomization rate Λ is taken slightly larger than $\max_{i\in S}\lambda_i$ (i.e. $\Lambda=(1+\theta)\max_{i\in S}\lambda_i$, where θ is a small quantity, say, 10^{-4}). Note that this implies $P_{i,i}>0,\ i\in S$. This, with $\lambda_{r,S'}>0$, guarantees (see [30]) that the quantities $a(k)$, $k=0,1,2,\ldots$ to be defined next are >0 and that, if $\alpha_{S'}>0$, the quantities $a'(k)$, $k=0,1,2,\ldots$ to be defined next

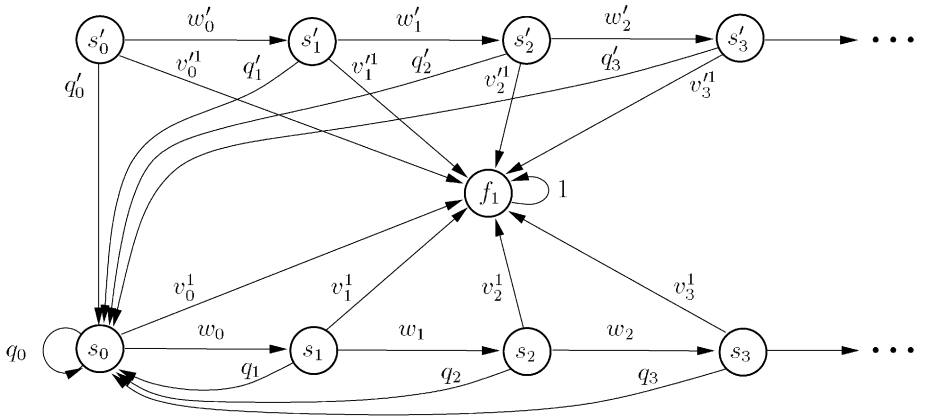

Fig. 1. State transition diagram of the DTMC $\hat{V}$ for the case $\alpha_{S'} > 0$, $A = 1$.

are > 0. The transformed model on which the regenerative randomization method is based is the derandomized CTMC $V = \{V(t); t \geqslant 0\}$ with rate Λ of the DTMC[5] $\hat{V} = \{\hat{V}_k; k = 0, 1, 2, \ldots\}$ defined as

$$\hat{V}_k = \begin{cases} s_l & \text{if } 0 \leqslant l \leqslant k \wedge \hat{X}_{k-l} = r \wedge \hat{X}_{k-l+1:k} \in S', \\ s'_k & \text{if } \hat{X}_{0:k} \in S', \\ f_i & \text{if } \hat{X}_k = f_i. \end{cases} \tag{11}$$

In words, $\hat{V}$ keeps track of when $\hat{X}$ made its last visit to r: $\hat{V}_k = s_l$ if, by step k, $\hat{X}$ has not left S, has made some visit to r, and the last visit to r was at step $k - l$; $\hat{V}_k = s'_k$ if, by step k, $\hat{X}$ has not left S'; and $\hat{V}_k = f_i$ if, by step k, $\hat{X}$ has been absorbed into state f_i. Note that $\hat{V}_k = s_0$ if and only if $\hat{X}_k = r$.

Assume $\alpha_{S'} > 0$. Let $a(l) = \sum_{i \in S} \pi_i(l)$ and $a'(l) = \sum_{i \in S'} \pi'_i(l)$. Let $v^j_l = \sum_{i \in S} \pi_i(l) P_{i,f_j}/a(l)$, $q_l = \sum_{i \in S} \pi_i(l) P_{i,r}/a(l)$, $w_l = \sum_{i \in S} \pi_i(l) P_{i,S'}/a(l)$, $v'^j_l = \sum_{i \in S'} \pi'_i(l) P_{i,f_j}/a'(l)$, $q'_l = \sum_{i \in S'} \pi'_i(l) P_{i,r}/a'(l)$, $w'_l = \sum_{i \in S'} \pi'_i(l) P_{i,S'}/a'(l)$. Then, it has been shown in [30] that $\hat{V}$ is a DTMC with initial probability distribution $P[\hat{V}_0 = s_0] = \alpha_r$, $P[\hat{V}_0 = s'_0] = \alpha_{S'}$, $P[\hat{V}_0 = f_i] = \alpha_{f_i}$, $P[\hat{V}_0 = i] = 0$, $i \notin \{s_0, s'_0, f_1, f_2, \ldots, f_A\}$, and (possibly) non-null transition probabilities $P[\hat{V}_{k+1} = f_j \mid \hat{V}_k = s_l] = v^j_l$, $P[\hat{V}_{k+1} = s_0 \mid \hat{V}_k = s_l] = q_l$, $P[\hat{V}_{k+1} = s_{l+1} | \hat{V}_k = s_l] = w_l$, $P[\hat{V}_{k+1} = f_j | \hat{V}_k = s'_l] = v'^j_l$, $P[\hat{V}_{k+1} = s_0 | \hat{V}_k = s'_l] = q'_l$, $P[\hat{V}_{k+1} = s'_{l+1} | \hat{V}_k = s'_l] = w'_l$, and $P[\hat{V}_{k+1} = f_i | \hat{V}_k = f_i] = 1$. The state transition diagram of $\hat{V}$ is illustrated in Fig. 1 for the case $A = 1$. In the case $\alpha_{S'} > 0$, V has state space $\{s_0, s_1, \ldots\} \cup \{s'_0, s'_1, \ldots\} \cup \{f_1, f_2, \ldots, f_A\}$, initial probability distribution $P[V(0) = s_0] = \alpha_r$, $P[V(0) = s'_0] = \alpha_{S'}$, $P[V(0) = f_i] = \alpha_{f_i}$, $P[V(0) = i] = 0$, $i \notin \{s_0, s'_0, f_1, f_2, \ldots, f_A\}$, and the state transition diagram illustrated in Fig. 2 for the case $A = 1$. In the case $\alpha_{S'} = 0$, the state transition diagrams of both $\hat{V}$ and V lose their upper part, corresponding to states s'_k, $k \geqslant 0$. Let I_c denote the indicator function returning the value 1 if condition c is satisfied and the value 0 otherwise. We have the following result:

[5] It is shown in [30] that the discrete-time stochastic process $\hat{V}$ is a DTMC.

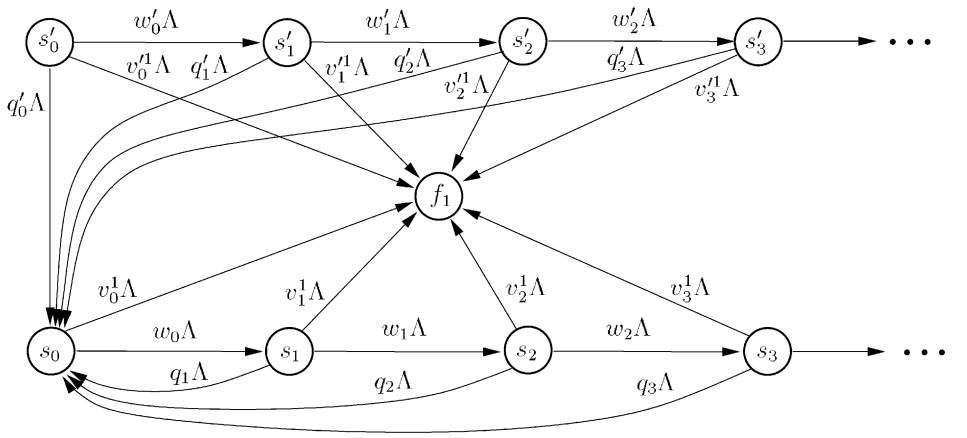

Fig. 2. State transition diagram of the CTMC V for the case $\alpha_{S'} > 0$, $A = 1$.

Proposition 1. *For* $i \in S$,

$$P[X(t) = i] = I_{\alpha_{S'}>0 \wedge i \neq r} \sum_{k=0}^{\infty} \frac{\pi'_i(k)}{a'(k)} P[V(t) = s'_k] + \sum_{k=0}^{\infty} \frac{\pi_i(k)}{a(k)} P[V(t) = s_k].$$

Proof. See the appendix. □

Let $b(k) = \sum_{i \in S} r_i \pi_i(k)/a(k)$ and, if $\alpha_{S'} > 0$, let $b'(k) = \sum_{i \in S'} r_i \pi'_i(k)/a'(k)$. Let $ETRR^V(t)$ and $EARR^V(t)$ be, respectively, the expected transient reward rate and expected averaged reward rate of V with reward rate structure $r'_{s_k} = b(k)$, $k = 0, 1, 2, \ldots$, if $\alpha_{S'} > 0$, $r'_{s'_k} = b'(k)$, $k = 0, 1, 2, \ldots$, and $r'_{f_i} = r_{f_i}$, $1 \leqslant i \leqslant A$. We have

Theorem 1. $ETRR^V(t) = ETRR(t)$, $EARR^V(t) = EARR(t)$.

Proof. Using Proposition 1 and $P[V(t) = f_i] = P[X(t) = f_i]$ (which easily follows from $P[\hat{V}_k = f_i] = P[\hat{X}_k = f_i]$, and the probabilistic identity between $X = \{X(t); t \geqslant 0\}$ and $\{\hat{X}_{Q(t)}; t \geqslant 0\}$ on one hand and $V = \{V(t); t \geqslant 0\}$ and $\{\hat{V}_{Q(t)}; t \geqslant 0\}$ on the other hand, being Q a Poisson process with arrival rate Λ independent of both $\hat{X}$ and $\hat{V}$),

$$\begin{aligned} ETRR(t) &= \sum_{i \in \Omega} r_i P[X(t) = i] = \sum_{i \in S} r_i P[X(t) = i] + \sum_{i=1}^{A} r_{f_i} P[X(t) = f_i] \\ &= \sum_{i \in S} r_i \left(I_{\alpha_{S'}>0 \wedge i \neq r} \sum_{k=0}^{\infty} \frac{\pi'_i(k)}{a'(k)} P[V(t) = s'_k] + \sum_{k=0}^{\infty} \frac{\pi_i(k)}{a(k)} P[V(t) = s_k] \right) \\ &\quad + \sum_{i=1}^{A} r_{f_i} P[V(t) = f_i] \end{aligned}$$

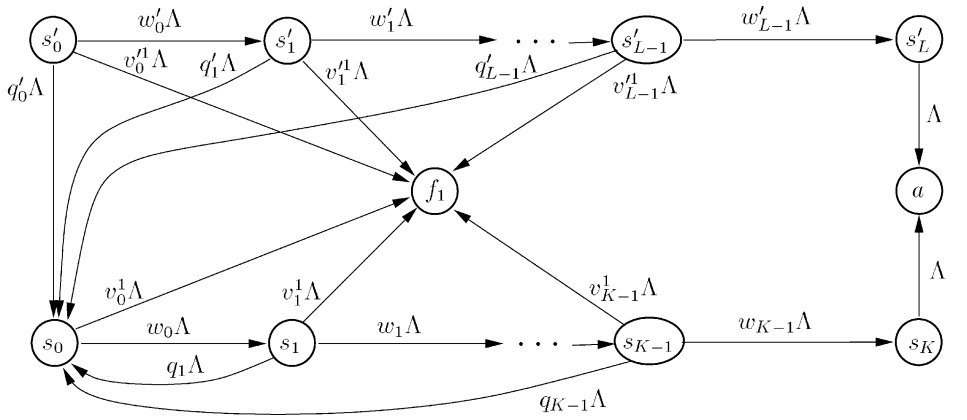

Fig. 3. State transition diagram of the CTMC $V_{K,L}$ for the case $A = 1$.

$$= I_{\alpha_{S'}>0}\sum_{k=0}^{\infty}\frac{\sum_{i\in S'} r_i\pi'_i(k)}{a'(k)}P[V(t)=s'_k] + \sum_{k=0}^{\infty}\frac{\sum_{i\in S} r_i\pi_i(k)}{a(k)}P[V(t)=s_k]$$

$$+\sum_{i=1}^{A} r_{f_i}P[V(t)=f_i]$$

$$= I_{\alpha_{S'}>0}\sum_{k=0}^{\infty} b'(k)P[V(t)=s'_k] + \sum_{k=0}^{\infty} b(k)P[V(t)=s_k] + \sum_{i=1}^{A} r_{f_i}P[V(t)=f_i]$$

$$= ETRR^V(t).$$

Finally, using $EARR(t) = \int_0^t ETRR(\tau)\,\mathrm{d}\tau/t$ and $EARR^V(t) = \int_0^t ETRR^V(\tau)\,\mathrm{d}\tau/t$,

$$EARR(t) = \frac{1}{t}\int_0^t ETRR(\tau)\,\mathrm{d}\tau = \frac{1}{t}\int_0^t ETRR^V(\tau)\,\mathrm{d}\tau = EARR^V(t). \qquad \square$$

Approximate values for $ETRR(t)$ and $EARR(t)$ can be obtained by truncating V. For the case $\alpha_{S'} > 0$, the truncated CTMC is called $V_{K,L}$ and is obtained from V by keeping the states s_k up to s_K, $K \geqslant 1$ and the states s'_k up to s'_L, $L \geqslant 1$ and directing to an absorbing state a the transitions rates from states s_K and s'_L. The initial probability distribution of $V_{K,L}$ is the same as that of V and its state transition diagram is illustrated in Fig. 3 for the case $A = 1$. Formally, $V_{K,L}$ can be defined from V as

$$V_{K,L}(t) = \begin{cases} V(t) & \text{if, by time } t,\ V \text{ has not exited state } s_K \text{ or state } s'_L, \\ a & \text{otherwise.} \end{cases} \tag{12}$$

For the case $\alpha_{S'} = 0$, the truncated CTMC is called V_K and is obtained from V by keeping the states s_k up to s_K, $K \geqslant 1$ and directing to an absorbing state a the transition rates from s_K. The initial probability distribution of V_K is the same as that of V and its state transition diagram is the same

as that of $V_{K,L}$ without the upper part, corresponding to states s'_k, $0 \leqslant k \leqslant L$. Formally, V_K can be defined from V as

$$V_K(t) = \begin{cases} V(t) & \text{if, by time } t, \ V \text{ has not exited state } s_K, \\ a & \text{otherwise.} \end{cases}$$

For the case $\alpha_{S'} > 0$, the approximate values, $ETRR^{K,L,a}(t)$ and $EARR^{K,L,a}(t)$, for, respectively, $ETRR(t)$ and $EARR(t)$, are the expected transient reward rate and expected averaged reward rate of $V_{K,L}$ with reward rate structure $r''_{s_k} = b(k)$, $r''_{s'_k} = b'(k)$, $r''_{f_i} = r_{f_i}$, and $r''_a = 0$. For the case $\alpha_{S'} = 0$, the approximate values, $ETRR^{K,a}(t)$ and $EARR^{K,a}(t)$, for, respectively, $ETRR(t)$ and $EARR(t)$, are the expected transient reward rate and the expected averaged reward rate of V_K with reward rate structure $r''_{s_k} = b(k)$, $r''_{f_i} = r_{f_i}$, and $r''_a = 0$.

The following theorem gives upper bounds for the model truncation errors for the $ETRR(t)$ measure.

Theorem 2. *Let* $r_{\max} = \max_{i\in\Omega} r_i$. *For the case* $\alpha_{S'} > 0$, $ETRR(t) - ETRR^{K,L,a}(t) \leqslant r_{\max} P[V_{K,L}(t) = a] = ETRR^{K,L,e}(t)$. *For the case* $\alpha_{S'} = 0$, $ETRR(t) - ETRR^{K,a}(t) \leqslant r_{\max} P[V_K(t) = a] = ETRR^{K,e}(t)$.

Proof. See the appendix. □

Regarding the measure $EARR(t)$ we have the following result:

Theorem 3. *Let* $r_{\max} = \max_{i\in\Omega} r_i$. *For the case* $\alpha_{S'} > 0$, $EARR(t) - EARR^{K,L,a}(t) \leqslant (r_{\max}/t) \int_0^t P[V_{K,L}(\tau)=a]\,\mathrm{d}\tau = EARR^{K,L,e}(t)$. *For the case* $\alpha_{S'}=0$, $EARR(t) - EARR^{K,a}(t) \leqslant (r_{\max}/t) \int_0^t P[V_K(\tau)= a]\,\mathrm{d}\tau = EARR^{K,e}(t)$.

Proof. For the case $\alpha_{S'} > 0$, using $EARR(t) = \int_0^t ETRR(\tau)\,\mathrm{d}\tau/t$ and $EARR^{K,L,a}(t) = \int_0^t ETRR^{K,L,a}(\tau)\,\mathrm{d}\tau/t$, and Theorem 2, we have

$$\begin{aligned} EARR(t) - EARR^{K,L,a}(t) &= \frac{1}{t}\int_0^t ETRR(\tau)\,\mathrm{d}\tau - \frac{1}{t}\int_0^t ETRR^{K,L,a}(\tau)\,\mathrm{d}\tau \\ &= \frac{1}{t}\int_0^t (ETRR(\tau) - ETRR^{K,L,a}(\tau))\,\mathrm{d}\tau \leqslant \frac{r_{\max}}{t}\int_0^t P[V_{K,L}(\tau) = a]\,\mathrm{d}\tau. \end{aligned}$$

The result for the case $\alpha_{S'} = 0$ can be proved similarly. □

The generalized regenerative randomization method uses upper bounds for $ETRR^{K,L,e}(t)$, $ETRR^{K,e}(t)$, $EARR^{K,L,e}(t)$, and $EARR^{K,e}(t)$ which can be computed inexpensively to control the error associated with the truncation of the transformed model V. The upper bounds for $ETRR^{K,L,e}(t)$ and $ETRR^{K,e}(t)$ are given by the following Theorem.

Theorem 4. *Let* $r_{\max} = \max_{i\in\Omega} r_i$. *For the case* $\alpha_{S'} > 0$, $ETRR^{K,L,e}(t) \leqslant r_{\max} a'(L) \sum_{k=L+1}^{\infty} \mathrm{e}^{-\Lambda t}(\Lambda t)^k/k! + r_{\max}\alpha_S a(K) \sum_{k=K+1}^{\infty}(k-K)\mathrm{e}^{-\Lambda t}(\Lambda t)^k/k!$. *For the case* $\alpha_{S'}=0$, $ETRR^{K,e}(t) \leqslant r_{\max}\alpha_S a(K) \sum_{k=K+1}^{\infty}(k-K)\mathrm{e}^{-\Lambda t}(\Lambda t)^k/k!$.

Proof. Since the expressions for $ETRR^{K,L,e}(t)$ and $ETRR^{K,e}(t)$ are formally identical to the expressions for, respectively, $m^e_{K,L}(t)$ and $m^e_K(t)$ in [30] (with the generalization to the case $A = 0$), the result follows from Theorem 2 of [30]. □

Regarding $EARR^{K,L,e}(t)$ and $EARR^{K,e}(t)$ we have

Theorem 5. *Let* $r_{\max} = \max_{i\in\Omega} r_i$. *For the case* $\alpha_{S'} > 0$, $EARR^{K,L,e}(t) \leqslant (r_{\max}a'(L))/(\Lambda t)\sum_{k=L+2}^{\infty}(k - L - 1)e^{-\Lambda t}(\Lambda t)^k/k! + (r_{\max}\alpha_S a(K))/(\Lambda t)\sum_{k=K+2}^{\infty}(((k - K)(k - K - 1))/2)e^{-\Lambda t}(\Lambda t)^k/k!$. *For the case* $\alpha_{S'} = 0$, $EARR^{K,e}(t) \leqslant (r_{\max}\alpha_S a(K))/(\Lambda t)\sum_{k=K+2}^{\infty}(((k - K)(k - K - 1))/2)e^{-\Lambda t}(\Lambda t)^k/k!$.

Proof. For the case $\alpha_{S'} > 0$, using Theorems 2–4

$$EARR^{K,L,e}(t) = \frac{r_{\max}}{t}\int_0^t P[V_{K,L}(\tau) = a]\,d\tau = \frac{1}{t}\int_0^t ETRR^{K,L,e}(\tau)\,d\tau$$

$$\leqslant \frac{r_{\max}a'(L)}{t}\sum_{k=L+1}^{\infty}\int_0^t e^{-\Lambda\tau}\frac{(\Lambda\tau)^k}{k!}\,d\tau$$

$$+ \frac{r_{\max}\alpha_S a(K)}{t}\sum_{k=K+1}^{\infty}(k-K)\int_0^t e^{-\Lambda\tau}\frac{(\Lambda\tau)^k}{k!}\,d\tau.$$

Using $\int_0^t e^{-\Lambda\tau}(\Lambda\tau)^k/k!\,d\tau = (1/\Lambda)\sum_{l=k+1}^{\infty} e^{-\Lambda t}(\Lambda t)^l/l!$, the first term is equal to

$$\frac{r_{\max}a'(L)}{\Lambda t}\sum_{k=L+1}^{\infty}\sum_{l=k+1}^{\infty} e^{-\Lambda t}\frac{(\Lambda t)^l}{l!} = \frac{r_{\max}a'(L)}{\Lambda t}\sum_{k=L+2}^{\infty}(k-L-1)e^{-\Lambda t}\frac{(\Lambda t)^k}{k!}.$$

The second term is equal to

$$\frac{r_{\max}\alpha_S a(K)}{\Lambda t}\sum_{k=K+1}^{\infty}(k-K)\sum_{l=k+1}^{\infty} e^{-\Lambda t}\frac{(\Lambda t)^l}{l!}$$

$$= \frac{r_{\max}\alpha_S a(K)}{\Lambda t}\sum_{k=K+2}^{\infty}\left(\sum_{l=K+1}^{k-1}(l-K)\right)e^{-\Lambda t}\frac{(\Lambda t)^k}{k!}$$

$$= \frac{r_{\max}\alpha_S a(K)}{\Lambda t}\sum_{k=K+2}^{\infty}\frac{(k-K)(k-K-1)}{2}e^{-\Lambda t}\frac{(\Lambda t)^k}{k!},$$

proving the result for the case $\alpha_{S'} > 0$. The result for the case $\alpha_{S'} = 0$ can be proved similarly. □

It is proved in [30] that the upper bounds for $ETRR^{K,L,e}(t)$ and $ETRR^{K,e}(t)$ given by Theorem 4 are increasing with t. Regarding the upper bounds for $EARR^{K,e}(t)$ and $EARR^{K,e}(t)$ given by Theorem 5, since they are the averaged values in the interval $[0,t]$ of the upper bounds for, respectively, $ETRR^{K,L,e}(t)$ and $ETRR^{K,e}(t)$ given by Theorem 4, they are also increasing with t.

Inputs: X, A, r_i, $i \in \Omega$, $\boldsymbol{\alpha}$, r, ε, n, $t_1, t_2, \ldots, t_n$
Outputs: $\widetilde{ETRR}(t_1), \widetilde{ETRR}(t_2), \ldots, \widetilde{ETRR}(t_n)$
$r_{\max} = \max_{i\in\Omega} r_i$;
$t_{\max} = \max\{t_1, t_2, \ldots, t_n\}$;
$\Lambda = (1 + 10^{-4}) \max_{i\in S} \lambda_i$;
Obtain $\mathbf{P}$;
for $(i \in S)$ $P_{i,S'} = \sum_{j\in S', P_{i,j}\neq 0} P_{i,j}$; $\alpha_{S'} = \sum_{i\in S'} \alpha_i$; $\alpha_S = \alpha_r + \alpha_{S'}$;
if $(\alpha_{S'} > 0)$ $tol\_K = \varepsilon/4$; else $tol\_K = \varepsilon/2$;
$\boldsymbol{\pi} = (I_{i=r})_{i\in S}$; $a = 1$; $K = 0$;
do {
  for $(j = 1; j \le A; j{+}{+})$ $v_K^j = \sum_{i\in S, P_{i,f_j}\neq 0} \pi_i P_{i,f_j}/a$;
  $q_K = \sum_{i\in S, P_{i,r}\neq 0} \pi_i P_{i,r}/a$; $w_K = \sum_{i\in S} \pi_i P_{i,S'}/a$; $b(K) = \sum_{i\in S} \pi_i r_i/a$;
  $\boldsymbol{n\pi} = \boldsymbol{\pi}\mathbf{P}_Z$; $\boldsymbol{\pi} = \boldsymbol{n\pi}$;
  $K{+}{+}$;
  $a = \sum_{i\in S} \pi_i$;
}
until $(r_{\max}\alpha_S a \sum_{k=K+1}^{\infty}(k-K)e^{-\Lambda t_{\max}}(\Lambda t_{\max})^k/k! \le tol\_K)$;
$b(K) = \sum_{i\in S} \pi_i r_i/a$;
if $(\alpha_{S'} > 0)$ {
  $\boldsymbol{\pi}' = (\alpha_i)_{i\in S'}$; $a' = \alpha_{S'}$; $L = 0$;
  do {
    for $(j = 1; j \le A; j{+}{+})$ $v_L'^j = \sum_{i\in S', P_{i,f_j}\neq 0} \pi_i' P_{i,f_j}/a'$;
    $q_L' = \sum_{i\in S', P_{i,r}\neq 0} \pi_i' P_{i,r}/a'$; $w_L' = \sum_{i\in S'} \pi_i' P_{i,S'}/a'$; $b'(L) = \sum_{i\in S'} \pi_i' r_i/a'$;
    $\boldsymbol{n\pi}' = \boldsymbol{\pi}'\mathbf{P}_{Z'}$; $\boldsymbol{\pi}' = \boldsymbol{n\pi}'$;
    $L{+}{+}$;
    $a' = \sum_{i\in S'} \pi_i'$;
  }
  until $(r_{\max} a' \sum_{k=L+1}^{\infty} e^{-\Lambda t_{\max}}(\Lambda t_{\max})^k/k! \le \varepsilon/4)$;
  $b'(L) = \sum_{i\in S'} \pi_i' r_i/a'$;
}
$N = \min\{m \ge 0 : r_{\max} \sum_{k=m+1}^{\infty} e^{-\Lambda t_{\max}}(\Lambda t_{\max})^k/k! \le \varepsilon/2\}$;
if $(\alpha_{S'} > 0)$
  Give N steps to $\widehat{V}_{K,L}$ and compute $d(k) = \sum_{l=0}^{K} b(l) P[(\widehat{V}_{K,L})_k = s_l]$
  $+ \sum_{l=0}^{L} b'(l) P[(\widehat{V}_{K,L})_k = s_l'] + \sum_{i=1}^{A} r_{f_i} P[(\widehat{V}_{K,L})_k = f_i]$, $k = 0, 1, \ldots, N$;
else
  Give N steps to $\widehat{V}_K$ and compute $d(k) = \sum_{l=0}^{K} b(l) P[(\widehat{V}_K)_k = s_l]$
  $+ \sum_{i=1}^{A} r_{f_i} P[(\widehat{V}_K)_k = f_i]$, $k = 0, 1, \ldots, N$;
for $(i = 1; i \le n; i{+}{+})$
  for $(k = 0, \widetilde{ETRR}(t_i) = 0; k \le N; k{+}{+})$ $\widetilde{ETRR}(t_i)$ += $d(k) e^{-\Lambda t_i}(\Lambda t_i)^k/k!$;

Fig. 4. Algorithmic description of the regenerative randomization method for $ETRR(t)$.

Algorithmic descriptions of the generalized regenerative randomization method are given in Figs. 4 and 5, correspond, respectively, to the measures $ETRR(t)$ and $EARR(t)$. The algorithms have as inputs the CTMC X, the number A of absorbing states f_i, the reward rates r_i, $i \in \Omega$, an initial probability distribution row vector $\boldsymbol{\alpha} = (\alpha_i)_{i\in S}$, the regenerative state r, the allowed error ε, the number of time points n at which estimates for the measures have to be computed, and the time points $t_1, t_2, \ldots, t_n$. The algorithms have as outputs the estimates for the measure at the time points t_i. Of the allowed error, ε, a portion $\varepsilon/2$ is allocated for the error associated with the truncation of the transformed model and a portion $\varepsilon/2$ is allocated for the error associated with the solution of

Inputs: X, $\underset{\sim}{A}$, r_j, $i \in \Omega$, $\underset{\sim}{\alpha}$, r, ε, n, $t_1, t_2, \ldots, t_n$
Outputs: $\widetilde{EARR}(t_1), \widetilde{EARR}(t_2), \ldots, \widetilde{EARR}(t_n)$
$r_{\max} = \max_{i\in\Omega} r_i$;
$t_{\max} = \max\{t_1, t_2, \ldots, t_n\}$;
$\Lambda = (1 + 10^{-4}) \max_{i\in S} \lambda_i$;
Obtain $\mathbf{P}$;
for $(i \in S)$ $P_{i,S'} = \sum_{j\in S', P_{i,j}\neq 0} P_{i,j}$; $\alpha_{S'} = \sum_{i\in S'} \alpha_i$; $\alpha_S = \alpha_r + \alpha_{S'}$;
if $(\alpha_{S'} > 0)$ $tol\_K = \varepsilon/4$; else $tol\_K = \varepsilon/2$;
$\boldsymbol{\pi} = (I_{i=r})_{i\in S}$; $a = 1$; $K = 0$;
do {
  for $(j = 1; j \le A; j{+}{+})$ $v_K^j = \sum_{i\in S, P_{i,f_j}\neq 0} \pi_i P_{i,f_j}/a$;
  $q_K = \sum_{i\in S, P_{i,r}\neq 0} \pi_i P_{i,r}/a$; $w_K = \sum_{i\in S} \pi_i P_{i,S'}/a$; $b(K) = \sum_{i\in S} \pi_i r_i/a$;
  $\boldsymbol{n\pi} = \boldsymbol{\pi}\mathbf{P}_Z$; $\boldsymbol{\pi} = \boldsymbol{n\pi}$;
  $K{+}{+}$;
  $a = \sum_{i\in S} \pi_i$;
}
until $(((r_{\max}\alpha_S a)/(\Lambda t_{\max}))$
  $\sum_{k=K+2}^{\infty}((k-K)(k-K-1)/2)e^{-\Lambda t_{\max}}(\Lambda t_{\max})^k/k! \le tol\_K)$;
$b(K) = \sum_{i\in S} \pi_i r_i/a$;
if $(\alpha_{S'} > 0)$ {
  $\boldsymbol{\pi}' = (\alpha_i)_{i\in S'}$; $a' = \alpha_{S'}$; $L = 0$;
  do {
    for $(j = 1; j \le A; j{+}{+})$ $v_L'^j = \sum_{i\in S', P_{i,f_j}\neq 0} \pi_i' P_{i,f_j}/a'$;
    $q_L' = \sum_{i\in S', P_{i,r}\neq 0} \pi_i' P_{i,r}/a'$; $w_L' = \sum_{i\in S'} \pi_i' P_{i,S'}/a'$; $b'(L) = \sum_{i\in S'} \pi_i' r_i/a'$;
    $\boldsymbol{n\pi}' = \boldsymbol{\pi}'\mathbf{P}_{Z'}$; $\boldsymbol{\pi}' = \boldsymbol{n\pi}'$;
    $L{+}{+}$;
    $a' = \sum_{i\in S'} \pi_i'$;
  }
  until $((r_{\max}a'/(\Lambda t_{\max}))\sum_{k=L+2}^{\infty}(k-L-1)e^{-\Lambda t_{\max}}(\Lambda t_{\max})^k/k! \le \varepsilon/4)$;
  $b'(L) = \sum_{i\in S'} \pi_i' r_i/a'$;
}
$N = \min\{m \ge 0 : r_{\max}\sum_{k=m+1}^{\infty} e^{-\Lambda t_{\max}}(\Lambda t_{\max})^k/k! \le \varepsilon/2\}$;
if $(\alpha_{S'} > 0)$
  Give N steps to $\widehat{V}_{K,L}$ and compute $d(k) = \sum_{l=0}^{K} b(l)P[(\widehat{V}_{K,L})_k = s_l]$
  $+\sum_{l=0}^{L} b'(l)P[(\widehat{V}_{K,L})_k = s_l'] + \sum_{i=1}^{A} r_{f_i}P[(\widehat{V}_{K,L})_k = f_i]$, $k = 0, 1, \ldots, N$;
else
  Give N steps to $\widehat{V}_K$ and compute $d(k) = \sum_{l=0}^{K} b(l)P[(\widehat{V}_K)_k = s_l]$
  $+\sum_{i=1}^{A} r_{f_i}P[(\widehat{V}_K)_k = f_i]$, $k = 0, 1, \ldots, N$;
for $(k = 1, c = 0; k \le N+1; k{+}{+})$ {
  $c \mathrel{+}= d(k-1)$;
  $c(k) = c$;
}
for $(i = 1; i \le n; i{+}{+})$
  for $(k = 1, \widetilde{EARR}(t_i) = 0; k \le N+1; k{+}{+})$ $\widetilde{EARR}(t_i) \mathrel{+}= c(k)e^{-\Lambda t_i}(\Lambda t_i)^k/k!/(\Lambda t_i)$;

Fig. 5. Algorithmic description of the regenerative randomization method for $EARR(t)$.

the truncated transformed model by standard randomization. Since the upper bounds for the error associated with the truncation of the transformed model given by Theorems 4 and 5 increase with t, that error is controlled for $t_{\max} = \max\{t_1, t_2, \ldots, t_n\}$. For the case $\alpha_{S'} > 0$, the error allocated for the truncation of V, $\varepsilon/2$, is divided equally between the two contributions of the model truncation error bound. For both measures, the error upper bound associated with the solution of the truncated

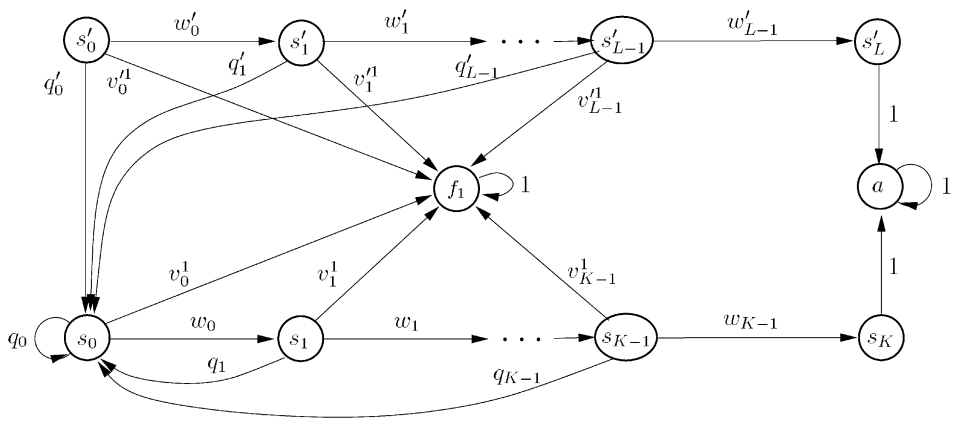

Fig. 6. State transition diagram of the CTMC $\hat{V}_{K,L}$ for the case $A = 1$.

transformed model by standard randomization, $r_{\max} \sum_{k=N+1}^{\infty} \mathrm{e}^{-\Lambda t}(\Lambda t)^k/k!$, where N is the truncation point, increases with t, since it is $r_{\max} \geqslant 0$ times the probability that by time t there have been more than N arrivals in a Poisson process with arrival rate Λ, and that error is also controlled for the largest t, $t_{\max}$. Solution of the truncated transformed model by standard randomization involves stepping the randomized DTMC $\hat{V}_{K,L}$ ($\hat{V}_K$) of $V_{K,L}$ (V_K) with randomization rate Λ. Fig. 6 shows the state transition diagram of $\hat{V}_{K,L}$ for the case $A = 1$. The state transition diagram of $\hat{V}_K$ is identical to the state transition diagram of $\hat{V}_{K,L}$ but without its upper part, corresponding to states s'_k, $k \geqslant 0$.

We deal next with some low-level implementation details not explicitly shown in Figs. 4 and 5. The method requires the computation of the summations $S(m) = \sum_{k=m+1}^{\infty} \mathrm{e}^{-\Lambda t}(\Lambda t)^k/k!$, $S'(m) = \sum_{k=m+1}^{\infty}(k-m)\mathrm{e}^{-\Lambda t}(\Lambda t)^k/k!$, and $S''(m) = \sum_{k=m+2}^{\infty}((k-m)(k-m-1)/2)\mathrm{e}^{-\Lambda t}(\Lambda t)^k/k!$ for $t = t_{\max}$ and increasing values of m. Efficient and numerically stable algorithms for computing $S(m)$ and $S'(m)$ for increasing values of m are given in [30]. An efficient and numerically stable algorithm for computing $S''(m)$ for increasing values of m is described next.

Assuming $M + 1 > \Lambda t$ and $M \geqslant m + 3$, we have

$$S''(m) = \sum_{k=m+2}^{M-1} \frac{(k-m)(k-m-1)}{2} \mathrm{e}^{-\Lambda t} \frac{(\Lambda t)^k}{k!} + \sum_{k=M}^{\infty} \frac{(k-m)(k-m-1)}{2} \mathrm{e}^{-\Lambda t} \frac{(\Lambda t)^k}{k!}$$

$$< \sum_{k=m+2}^{M-1} \frac{(k-m)(k-m-1)}{2} \mathrm{e}^{-\Lambda t} \frac{(\Lambda t)^k}{k!}$$

$$+ \mathrm{e}^{-\Lambda t} \frac{(\Lambda t)^M}{M!} \sum_{k=M}^{\infty} \frac{(k-m)(k-m-1)}{2} \left(\frac{\Lambda t}{M+1}\right)^{k-M}$$

$$= \tilde{S}''(m, M) + S''^{e}(m, M),$$

with

$$\tilde{S}''(m,M) = \sum_{k=m+2}^{M-1} \frac{(k-m)(k-m-1)}{2} \mathrm{e}^{-\Lambda t} \frac{(\Lambda t)^k}{k!},$$

$$S''^{e}(m,M) = \mathrm{e}^{-\Lambda t} \frac{(\Lambda t)^M}{M!} \sum_{k=M}^{\infty} \frac{(k-m)(k-m-1)}{2} \left(\frac{\Lambda t}{M+1}\right)^{k-M}$$

$$= \mathrm{e}^{-\Lambda t} \frac{(\Lambda t)^M}{M!} \sum_{k=0}^{\infty} \frac{(k+M-m)(k+M-m-1)}{2} \left(\frac{\Lambda t}{M+1}\right)^{k}$$

$$= \mathrm{e}^{-\Lambda t} \frac{(\Lambda t)^M}{M!} \left[\frac{(M-m)(M-m-1)}{2} \sum_{k=0}^{\infty} \left(\frac{\Lambda t}{M+1}\right)^{k} + \left(M-m-\frac{1}{2}\right) \sum_{k=0}^{\infty} k \left(\frac{\Lambda t}{M+1}\right)^{k} + \frac{1}{2} \sum_{k=0}^{\infty} k^2 \left(\frac{\Lambda t}{M+1}\right)^{k} \right]$$

$$= \mathrm{e}^{-\Lambda t} \frac{(\Lambda t)^M}{M!} \left[\frac{(M-m)(M-m-1)/2}{1-\Lambda t/(M+1)} + \left(M-m-\frac{1}{2}\right) \frac{\Lambda t/(M+1)}{(1-\Lambda t/(M+1))^2} + \frac{1}{2} \frac{\Lambda t/(M+1)(1+\Lambda t/(M+1))}{(1-\Lambda t/(M+1))^3} \right],$$

where it has been used $\sum_{k=0}^{\infty} k a^k = a/(1-a)^2$ and $\sum_{k=0}^{\infty} k^2 a^k = a(1+a)/(1-a)^3$, $0 < a < 1$ (for the first result, see, for instance, [12, Formula 19.7]; the second result can easily be obtained from the first one by taking the derivative of $\sum_{k=0}^{\infty} k a^k = a/(1-a)^2$ with respect to a). The quantity $\tilde{S}''(m,M) + S''^{e}(m,M)$ can be taken as a pessimistic (larger) approximation for $S''(m)$ with error upper bounded by $S''^{e}(m,M)$. Let

$$\tilde{S}'(m,M) = \sum_{k=m+1}^{M-1} (k-m) \mathrm{e}^{-\Lambda t} \frac{(\Lambda t)^k}{k!},$$

$$\tilde{S}(m,M) = \sum_{k=m+1}^{M-1} \mathrm{e}^{-\Lambda t} \frac{(\Lambda t)^k}{k!}.$$

It is easy to check that $\tilde{S}''(m,M) = \tilde{S}''(m-1,M) - \tilde{S}'(m,M)$ and $\tilde{S}'(m,M) = \tilde{S}'(m-1,M) - \tilde{S}(m-1,M)$. Also, $S''^{e}(m,M)$ decreases with m. Those observations justify the following algorithm. Being v a small quantity representing the desired relative error for the computation of the $S''(m)$'s (for instance, $v = 10^{-6}$), the algorithm starts by selecting for the initial m for which $S''(m)$ has to be computed, m_0, the smallest integer M with $M+1 > \Lambda t$, $M \geqslant m_0 + 3$ satisfying $S''^{e}(m_0,M)/\tilde{S}''(m_0,M) \leqslant v/10$, sets

$S''_{\text{last}} = \tilde{S}''(m_0, M) + S''^e(m_0, M)$, $S''_{\text{old}} = S''_{\text{last}}$, $S'_{\text{last}} = \tilde{S}'(m_0, M)$, $S'_{\text{old}} = S'_{\text{last}}$, $S_{\text{last}} = \tilde{S}(m_0, M)$, and $S_{\text{old}} = S_{\text{last}}$, and approximates $S''(m)$ with S''_{last}. Then, each time m is incremented, if $m \leqslant M-3$, the algorithm computes $S'_{\text{last}} = S'_{\text{last}} - S_{\text{last}}$, unless $S'_{\text{last}}/S'_{\text{old}}$ becomes < 0.1, in which case it computes $S'_{\text{last}} = \tilde{S}'(m, M)$ and sets $S'_{\text{old}} = S'_{\text{last}}$; then, after S'_{last} has been obtained, the algorithm computes $S''_{\text{last}} = S''_{\text{last}} - S'_{\text{last}}$ and, if $S''_{\text{last}}/S''_{\text{old}} \geqslant 0.1$, the algorithm approximates $S''(m)$ with S''_{last}, and before continuing computes $S_{\text{last}} = S_{\text{last}} - \mathrm{e}^{-\Lambda t}(\Lambda t)^m/m!$ (so that it be $\tilde{S}(m, M)$), unless $S_{\text{last}}/S_{\text{old}}$ becomes < 0.1, in which case the algorithm computes $S_{\text{last}} = \tilde{S}(m, M)$, and sets $S_{\text{old}} = S_{\text{last}}$. If m becomes $> M-3$ or, being $m \leqslant M-3$, $S''_{\text{last}}/S''_{\text{old}}$ becomes < 0.1, the algorithm obtains a new M as the smallest integer with $M+1 > \Lambda t$, $M \geqslant m+3$ satisfying $S''^e(m, M)/\tilde{S}''(m, M) \leqslant \nu/10$, sets $S''_{\text{last}} = \tilde{S}''(m, M) + S''^e(m, M)$, $S''_{\text{old}} = S''_{\text{last}}$, $S'_{\text{last}} = \tilde{S}'(m, M)$, $S'_{\text{old}} = S'_{\text{last}}$, $S_{\text{last}} = \tilde{S}(m, M)$, and $S_{\text{old}} = S_{\text{last}}$, approximates $S''(m)$ with S''_{last}, and continues.

We note that, in the generalized regenerative randomization method, the transient probabilities of $\hat{V}_{K,L}$ ($\hat{V}_K$) are determined, once $\mathbf{P}$ has been computed, by adding always positive numbers smaller than 1 and, therefore, regenerative randomization has the same excellent numerical stability as standard randomization. In addition, the computation error is well-controlled and can be specified in advance. Thus, the generalized regenerative randomization method has the same good properties as standard randomization.

We analyze next the memory overhead of regenerative randomization with respect to standard randomization. Given relationships (4)–(6), (8)–(10) between the transition probabilities of, respectively, Z and Z' and the transition probability matrix $\mathbf{P}$ of $\hat{X}$, it is not necessary to store $\mathbf{P}_Z$ and $\mathbf{P}_{Z'}$ explicitly. In addition, vectors $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$ and vectors $\boldsymbol{n\pi}$ and $\boldsymbol{n\pi}'$ can share the same storage, and a similar storage is required by standard randomization. The memory overhead is then basically restricted to the space needed to store the vector of size $|S|$, $(P_{i,S'})_{i \in S}$, the transition probabilities of $\hat{V}_{K,L}$ ($\hat{V}_K$) v_k^i, q_k, w_k, $0 \leqslant k \leqslant K-1$, $1 \leqslant i \leqslant A$ and, if $\alpha_{S'} > 0$, $v_k'^i$, q'_k, w'_k, $0 \leqslant k \leqslant L-1$, $1 \leqslant i \leqslant A$, and the quantities $b(k)$, $0 \leqslant k \leqslant K$ and, if $\alpha_{S'} > 0$, $b'(k)$, $0 \leqslant k \leqslant L$.

3. Theoretical properties

As discussed in Section 1, standard randomization requires a number of steps of the DTMC $\hat{X}$ which, for large Λt, is approximately equal to Λt. The model truncation error bounds for $ETRR(t)$ of regenerative randomization are formally identical to the model truncation error bounds for the less general measure considered in [30] and, then, for $ETRR(t)$, we have the following result:

Theorem 6. *For the case $\alpha_{S'} > 0$, the number of steps K on Z and the number of steps L on Z' required in regenerative randomization for the measure $ETRR(t)$ are, respectively, $O(\log(\Lambda t/\varepsilon))$ and $O(\log(1/\varepsilon))$. For the case $\alpha_{S'} = 0$, the number of steps K on Z required in regenerative randomization for the measure $ETRR(t)$ is $O(\log(\Lambda t/\varepsilon))$.*

A similar result holds for the measure $EARR(t)$:

Theorem 7. *For the case $\alpha_{S'} > 0$, the number of steps K on Z and the number of steps L on Z' required in regenerative randomization for the measure $EARR(t)$ are, respectively, $O(\log(\Lambda t/\varepsilon))$ and*

$O(\log(1/\varepsilon))$. *For the case* $\alpha_{S'}=0$, *the number of steps* K *on* Z *required in regenerative randomization for the measure* $EARR(t)$ *is* $O(\log(\Lambda t/\varepsilon))$.

Proof. Note that the model truncation error bounds for $EARR(t)$ are the averaged values in the interval $[0,t]$ of the respective model truncation error bounds for $ETRR(\tau)$, $0 \leqslant \tau \leqslant t$. Then, since the model truncation error bounds for $ETRR(\tau)$ increase with τ, the model truncation error bounds for $EARR(t)$ are no greater than the respective model truncation error bounds for $ETRR(t)$ and the result follows from Theorem 6. □

Theorems 6 and 7 assert that, contrary to standard randomization, the number of steps required in regenerative randomization is, for large Λt, a smooth function of t. That property is called "benign behavior". A consequence of Theorems 6 and 7 is that, for large enough Λt, the number of steps on Z and Z' required in regenerative randomization will be significantly smaller than the number of steps on $\hat{X}$ required in standard randomization, implying that the cost of the first phase of regenerative randomization (generation of the truncated transformed model) will be significantly smaller than the cost of standard randomization. In addition, for large enough X, the truncated transformed model will be significantly smaller than X, and, since the maximum output rate of the truncated transformed model is only slightly larger than $\max_{i\in\Omega}\lambda_i$ and, then, for large t, the truncation point N of the standard randomization method applied to the solution of the truncated transformed model would be almost identical to the truncation point N of standard randomization applied to X, the second phase of regenerative randomization (solution of the truncated transformed model by standard randomization) will have significantly smaller cost than standard randomization. In summary, for large enough X and Λt, regenerative randomization will be significantly faster than standard randomization.

The performance of regenerative randomization depends, of course, on the selection of the regenerative state r, since that selection influences the behavior of $a(k)$ and $a'(k)$ and the required values for the truncation parameters K and L. Ideally, the state r should be chosen so that $a(k)$ and $a'(k)$ decrease as fast as possible. For as wide class of models as covered by the generalized regenerative randomization method, automatic selection of r does not seem to be easy in general, and, then, the method relies on user's intuition to select an appropriate state r. However, a model class C′ can be considered for which a natural selection for the regenerative state exists and, for those models, theoretical results are available assessing the performance of regenerative randomization in terms of "visible" model characteristics.

The model class C′ includes all CTMCs X with the properties described in Section 1 for which a partition $S_0 \cup S_1 \cup \cdots \cup S_{N_C}$ for S exists satisfying the following two properties:

P1. $S_0 = \{o\}$ (i.e. $|S_0| = 1$).

P2. $\max_{0\leqslant k\leqslant N_C} \max_{i\in S_k} \lambda_{i,S_k-\{i\}\cup S_{k+1}\cup\cdots\cup S_{N_C}}$ is significantly smaller than $\min_{0<k\leqslant N_C} \min_{i\in S_k} \lambda_{i,S_0\cup\cdots\cup S_{k-1}\cup\{f_1,\ldots,f_A\}} > 0$.

Class C′ covers failure/repair models with exponential failure and repair time distributions and repair in every state with failed components when failure rates are significantly smaller than repair rates (the typical case). For those models, a partition for which properties P1 and P2 are satisfied is $S_k = \{\text{states in } S \text{ with } k \text{ failed components}\}$. The class also covers failure/repair models with exponential failure time distributions, repair times with acyclic phase-type distributions [31] (which can

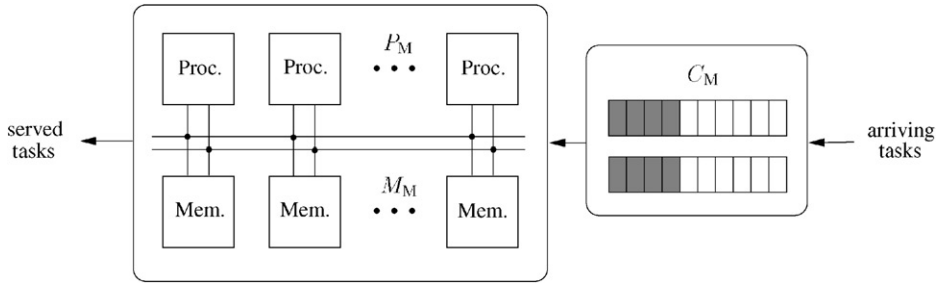

Fig. 7. Structure of the fault-tolerant multiprocessor system.

be used to fit distributions of non-exponential positive random variables [32]), and repair in every state with failed components, provided that the transition rates of the transient CTMCs defining the phase-type distributions are sufficiently large compared with failure rates.

Since, for class C′ models, X moves "fast" to either state o or an absorbing state f_i, a natural selection for the regenerative state for those models is $r = o$. Let $R = \max_{i\in S} \lambda_i / \min_{i\in S'} \lambda_i$ (R is a "visible" parameter, i.e. one that can be easily predicted by the user). Then, it is shown in [30] that, with $r = o$, both $a(k)$ and $a'(k)$ are upper bounded by functions of the form $B\binom{k}{p-1}\rho^k$, $B > 0$, p integer $\geqslant 1$, for $k \to \infty$, where $\rho \approx 1 - 1/R$ and, then, the number of steps K and L required in regenerative randomization will be mainly determined by R: the smaller R, the smaller K and L. As a rule of thumb, for $R \gg 1$, the required K and L can be roughly upper bounded by $30R$. Those rough upper bounds can be used to anticipate, for class C′ models with the selection $r = o$, when regenerative randomization can be expected to be significantly faster than standard randomization.

4. A fault-tolerant multiprocessor example

The theoretical results for class C′ models given in the previous section clearly indicate that regenerative randomization can be much faster than standard randomization for models of that class. In this section, we will consider a large performability model of a fault-tolerant multiprocessor system not belonging to class C′ to show that regenerative randomization can be significantly faster than standard randomization, also for other models.

The fault-tolerant multiprocessor system is made up of a multiprocessor comprising P_M processors, M_M memories, two redundant busses, and two redundant buffers with capacity to hold up to C_M tasks. Tasks arrive to the system following a Poisson process with rate ϕ and are held in all unfailed buffers until their service by the multiprocessor is finished. The structure of the system is depicted in Fig. 7. The system is up when at least two processors and two memories are believed to be unfailed and one bus and one buffer are unfailed. Being up, the multiprocessor may serve tasks in two modes: dual mode and simplex mode. In dual mode, processors and memories which are believed to be unfailed are used to build two clusters of maximum size containing the same number of processors and memories and both clusters execute the same task and compare the results. Processors and memories executing a task are called active. If the results agree, the service of the task is completed and, if there are more tasks in the buffers, a new task is served. If the results issued by the clusters disagree,

the task is re-executed in the hope that the cause of the disagreement was a transient fault. If the new results agree, the service of the task is completed. If the results disagree again, a permanent fault is thought to have been the cause, and the multiprocessor system is diagnosed. Diagnosis is assumed perfect and failed components will be detected with probability one. In simplex mode, the whole multiprocessor serves a single task using all processors and memories that are believed to be unfailed. Those processors and memories are called active. If a fault occurs in simplex mode, the task is erroneously served.

The multiprocessor system is configured in dual mode if the number of tasks in the buffers is $< N_T$ and in simplex mode if the number of tasks in the buffers is $\geqslant N_T$, where N_T is a threshold parameter with $0 \leqslant N_T \leqslant C_M + 1$. If a task arrives when the buffers are full, the task is rejected. The threshold parameter N_T can be optimized to minimize the rate at which tasks are either lost (because they arrive when the buffers are full, they arrive when both buffers are failed, or they are lost because a buffer failed when the other buffer was also failed) or erroneously served. Task service times are assumed to have an exponential distribution. Let ψ the rate at which a task would be served by a single processor and a single memory. The speedups resulting from multiprocessing are computed using the model proposed in [33][6] (however, if available, measured speedups could be easily incorporated). In addition, the model accounts for an overhead in dual mode due to comparison of the results. Then, the service rate in simplex mode is $\psi R_M(1-(1-1/R_M)^{R_m})$, where $R_m = \min\{P_a, M_a\}$ and $R_M = \max\{P_a, M_a\}$, P_a and M_a being, respectively, the numbers of active processors and memories. In dual mode, the task service rate is $\psi R_M(1-(1-1/R_M)^{R_m})/(1+\beta)$, where R_m and R_M are as before (in this case P_a and M_a are, respectively, the numbers of active processors and memories in each cluster) and β is a parameter accounting for the overhead due to comparison of the results. Faults in busses, buffers and inactive processors and memories (i.e. not serving a task) are assumed to be detected immediately; however, permanent faults in active processors and memories will not be detected except when the system is diagnosed. Diagnosis is performed after a second disagreement in dual mode and when a critical fault in a bus or a buffer (i.e. a fault which takes the system down) occurs while the multiprocessor is serving a task. Detected failed memories and processors and failed busses and buffers are repaired with rate μ_R by a single repairman who gives priority first to busses, next to buffers, next to processors, and last to memories. When the system is serving a task and a processor or memory gets repaired, it is not considered a candidate to become part of the active configuration until a new task has to be served. Diagnosis is performed at rate μ_D. The other parameters of the model are the processor transient fault rate λ_{PT}, the memory transient fault rate λ_{MT}, the processor permanent fault rate λ_{PP}, the memory permanent fault rate λ_{MP}, the buffer fault rate λ_{BF}, and the bus fault rate λ_{BS}.

The measures of interest are the rate at time t at which tasks are either lost or erroneously served and the average rate in the interval $[0,t]$ at which tasks are either lost or erroneously served. These measures can be formalized as, respectively, $ETRR(t)$ and $EARR(t)$. The required reward rate structure can be obtained by adding two reward rate structures. The first one deals with task losses and is ϕ in the states in which two buffers are failed, and is obtained, otherwise, by adding $q_B\lambda_{BF}$, where q_B is the number of tasks in the buffer, for the states in which one buffer is failed, and ϕ for the states in which the number of tasks in the buffer is equal to C_M. The second one deals with tasks erroneously served and has value $\psi R_M(1-(1-1/R_M)^{R_m})$, $R_m = \min\{P_a, M_a\}$, $R_M = \max\{P_a, M_a\}$,

[6] The model has been used in [34] in a context similar to ours.

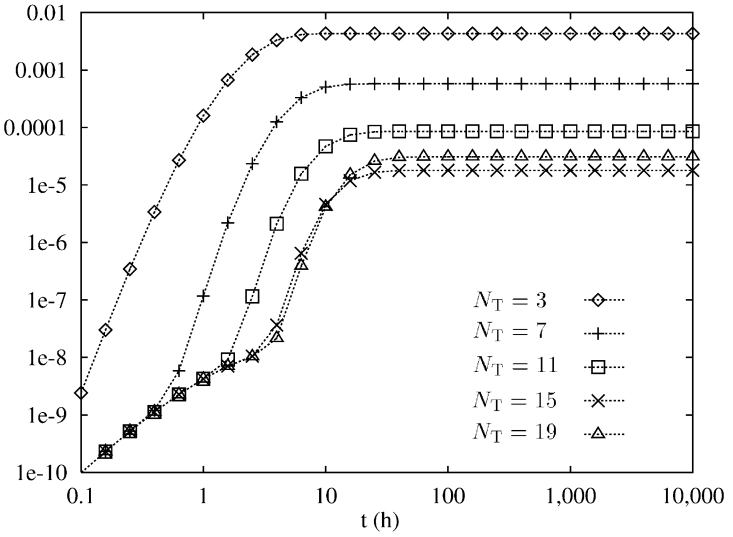

Fig. 8. Behavior of the lower bound for $ETRR(t)$.

P_a and M_a being, respectively, the number of active processors and memories serving the task, in the states in which the multiprocessor is configured in simplex mode and either a transient fault occurred during the service of the task or there is some active processor or memory in permanent fault.

We will use the set of model parameter values $P_M = 16$, $M_M = 16$, $C_M = 20$, $\lambda_{PT} = 1.5 \times 10^{-3}\ h^{-1}$, $\lambda_{PP} = 1.5 \times 10^{-4}\ h^{-1}$, $\lambda_{MT} = 6 \times 10^{-4}\ h^{-1}$, $\lambda_{MP} = 6 \times 10^{-5}\ h^{-1}$, $\lambda_{BS} = 5 \times 10^{-6}\ h^{-1}$, $\lambda_{BF} = 5 \times 10^{-5}\ h^{-1}$, $\mu_R = 1\ h^{-1}$, $\mu_D = 5\ h^{-1}$, $\phi = 1.5\ h^{-1}$, $\psi = 0.5\ h^{-1}$, and $\beta = 0.05$, and will vary the threshold parameter N_T. The complete CTMCs have unmanageable number of states and, thus, the example illustrates the applicability of bounding models. We include in S the states with up to N_F failed components. Then, the bounding models have state space $S \cup \{f_1\}$, where f_1 is an absorbing state into which the bounding model enters when the complete model would exit S. The lower bound for both $ETRR(t)$ and $EARR(t)$ is obtained by assigning a reward rate 0 to f_1. The upper bound for both $ETRR(t)$ and $EARR(t)$ is obtained by assigning a reward rate $\psi R_2(1 - (1 - 1/R_2)^{R_1}) + C_M \lambda_{BF} + \phi$, $R_1 = \min\{P_M, M_M\}$, $R_2 = \max\{P_M, M_M\}$ to f_1, which upper bounds the reward rate associated with any state of the complete model ($\psi R_2(1-(1-1/R_2)^{R_1})$ upper bounds the second reward rate structure, which increases with both R_m and R_M). After some experimentation, we found that the minimum N_F for which the bounds were tight for all considered values of t (up to $10,000$ h) was $N_F = 4$, and we will report results for that value of N_F. In addition, both $ETRR(t)$ and $EARR(t)$ will be computed assuming that the initial probability distribution of X is concentrated in the state in which no component is failed and the buffers are empty. That state will be taken as the regenerative state r. Thus, we will only illustrate the particular case $\alpha_{S'} = 0$.

Fig. 8 plots the lower bound for $ETRR(t)$ (the upper bound was close enough to consider $ETRR(t)$ well determined) as a function of t for several values of N_T. Several comments are in order. First, the "steady state" value is reached relatively fast. However, note that there is no rigorous way to take advantage of this fact to save computations because, in fact, the steady state is concen-

Table 1
Bounds for $ETRR(t)$ for $N_T = 16$

t (h)	$[ETRR(t)]_{lb}$	$[ETRR(t)]_{ub}$
0.1	9.86060×10^{-11}	9.86060×10^{-11}
0.2	3.49924×10^{-10}	3.49924×10^{-10}
0.5	1.60598×10^{-9}	1.60598×10^{-9}
1	4.22992×10^{-9}	4.22993×10^{-9}
2	8.70868×10^{-9}	8.70890×10^{-9}
5	9.52465×10^{-8}	9.52523×10^{-8}
10	3.42316×10^{-6}	3.42319×10^{-6}
20	1.27237×10^{-5}	1.27238×10^{-5}
50	1.60857×10^{-5}	1.60859×10^{-5}
100	1.61162×10^{-5}	1.61166×10^{-5}
200	1.61162×10^{-5}	1.61172×10^{-5}
500	1.61162×10^{-5}	1.61187×10^{-5}
1000	1.61162×10^{-5}	1.61213×10^{-5}
2000	1.61162×10^{-5}	1.61264×10^{-5}
5000	1.61162×10^{-5}	1.61418×10^{-5}
10,000	1.61162×10^{-5}	1.61674×10^{-5}

trated in the absorbing state of the model, that true steady state is reached much more slowly, and by then the bounds are extremely coarse and of no interest. Second, as N_T increases, the "steady state" is reached more slowly. The explanation for this is the following. As N_T increases, the value of $ETRR(t)$ becomes more dominated by task rejections due to the buffers being full and, thus, the steady state is reached when the distribution of the number of tasks in the buffers reaches its steady state, and this is done more slowly as N_T increases. Third, for small values of t and not too small N_T, the curves collapse and follow a behavior different from the one corresponding to larger values of t. This is due to the fact that for small values of t and not too small N_T, the probability of the multiprocessor working in simplex mode or the buffers being full is very small, and $ETRR(t)$ is basically due to queued tasks lost when a second buffer fails and tasks which are rejected when both buffers are failed. As N_T increases, more tasks have to be queued for the multiprocessor system to switch to simplex mode, in which tasks may be erroneously served, and the above behavior is sustained for larger values of t. Finally, we note that the optimum value for N_T depends on t. This is illustrated by the crosspoint between the curves corresponding to $N_T = 15$ and $N_T = 19$: for t smaller than about 10 h, $N_T = 19$ is better; for t larger than about 10 h, $N_T = 15$ is better. For $t = 1000$ h the optimum value for N_T is $N_T = 16$, yielding $ETRR(t) \approx 1.61 \times 10^{-5}$. For that value of N_T, X has 83,972 states and 534,873 transitions. Table 1 gives the bounds obtained for $ETRR(t)$ for $N_T = 16$ and several values of t, ranging from 0.1 to 10,000 h. The results were obtained with $\varepsilon = 10^{-17}$. The bounds are tight for all values of t but become less tight as t increases.

Fig. 9 plots the lower bound for $EARR(t)$ (the upper bound was close enough to consider $EARR(t)$ well determined) as a function of t for several values of N_T. The $EARR(t)$ measure takes longer to "stabilize" than the $ETRR(t)$ measure. The behavior of $EARR(t)$ with respect to t and N_T is similar

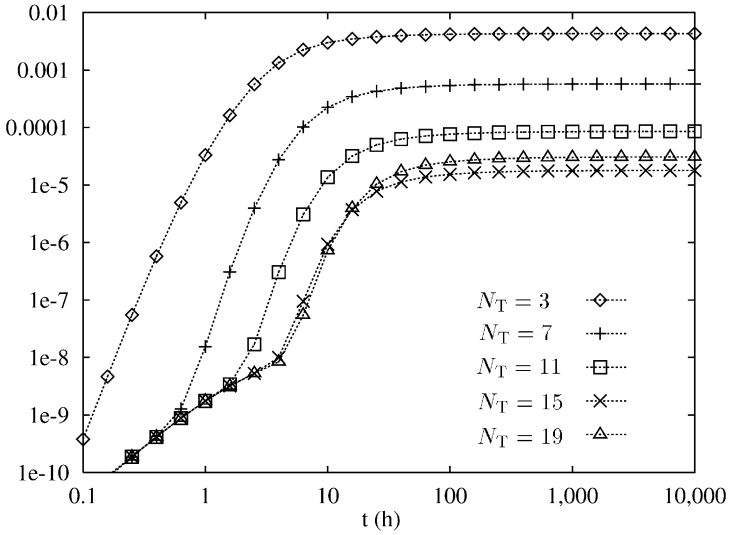

Fig. 9. Behavior of the lower bound for $EARR(t)$.

Table 2
Bounds for $EARR(t)$ for $N_T = 16$

t (h)	$[EARR(t)]_{lb}$	$[EARR(t)]_{ub}$
0.1	3.39464×10^{-11}	3.39464×10^{-11}
0.2	1.23912×10^{-10}	1.23912×10^{-10}
0.5	6.09495×10^{-10}	6.09495×10^{-10}
1	1.75770×10^{-9}	1.75770×10^{-9}
2	4.18602×10^{-9}	4.18606×10^{-9}
5	1.75127×10^{-8}	1.75141×10^{-8}
10	6.60466×10^{-7}	6.60475×10^{-7}
20	4.67714×10^{-6}	4.67717×10^{-6}
50	1.11129×10^{-5}	1.11130×10^{-5}
100	1.36126×10^{-5}	1.36128×10^{-5}
200	1.48644×10^{-5}	1.48649×10^{-5}
500	1.56154×10^{-5}	1.56167×10^{-5}
1000	1.58658×10^{-5}	1.58683×10^{-5}
2000	1.59910×10^{-5}	1.59961×10^{-5}
5000	1.60661×10^{-5}	1.60789×10^{-5}
10000	1.60911×10^{-5}	1.61167×10^{-5}

to the behavior of the $ETRR(t)$ measure and similar comments can be made. Table 2 gives the bounds obtained for $EARR(t)$ for $N_T = 16$ and several values of t, ranging from 0.1 to 10,000 h. The results were obtained with $\varepsilon = 10^{-17}$. The bounds are tight for all values of t, but become less tight as t increases.

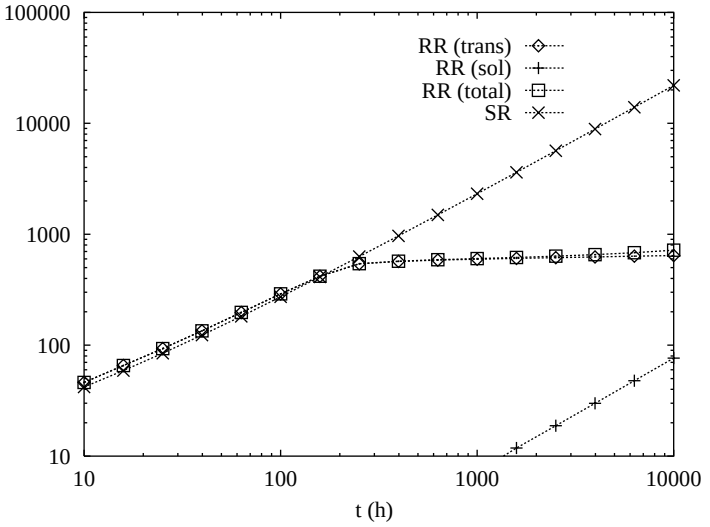

Fig. 10. CPU times in seconds required by regenerative randomization (RR) and standard randomization (SR) to compute the lower bound for $ETRR(t)$ for $N_T = 16$ and $\varepsilon = 10^{-12}$.

We compare next the performances of regenerative randomization and standard randomization. Standard randomization was implemented with $\Lambda = \max_{i\in\Omega} \lambda_i$. Fig. 10 plots the CPU times consumed by regenerative randomization and standard randomization to compute the lower bound for $ETRR(t)$ for $N_T = 16$ and $\varepsilon = 10^{-12}$ as a function of the target time t (the algorithms were run with a single time target t on a 167 MHz, 128 MB UltraSPARC 1 workstation and the CPU times for standard randomization for large t were estimated using the required number of steps on the randomized DTMC). The CPU times consumed by regenerative randomization are decomposed into the times consumed to obtain the truncated transformed model (trans) and the times consumed to solve that model by standard randomization (sol). For small values of t both methods require about the same number of steps (i.e. the required K under regenerative randomization is approximately equal to the required number of steps N under standard randomization) and have similar performances. However, beyond $t = 200$ h, the benign behavior of regenerative randomization predicted by Theorem 6 enters into play and the required CPU time increases very smoothly with t, whereas the CPU time required under standard randomization increases approximately linearly with t. This makes regenerative randomization significantly faster than standard randomization for large t. Thus, for $t = 10,000$ h, regenerative randomization requires $K = 1961$ and a CPU time of 719.3 s (about 12 min), whereas standard randomization requires $N = 78,916$ steps and a CPU time of $21,964$ s (about 6 h), making regenerative randomization about 31 times faster than standard randomization for that t. For regenerative randomization, the CPU times consumed in the solution of the truncated transformed model are significantly smaller than the CPU times consumed in the generation of the truncated transformed model for all considered values of t, but the first increases faster than the second and, for large enough t, the total CPU time consumed by regenerative randomization would be dominated by that component. Fig. 11 plots the CPU times consumed by regenerative randomization and standard randomization to compute the lower bound for $EARR(t)$ for $N_T = 16$ and $\varepsilon = 10^{-12}$ as

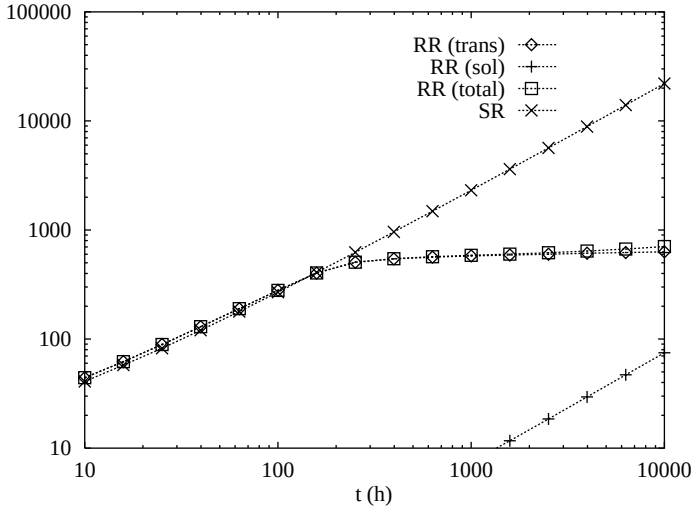

Fig. 11. CPU times in seconds required by regenerative randomization (RR) and standard randomization (SR) to compute the lower bound for $EARR(t)$ for $N_T = 16$ and $\varepsilon = 10^{-12}$.

a function of the target time t. The behavior is very similar to that encountered for the CPU times consumed by both methods for the computation of the lower bound for $ETRR(t)$.

5. Conclusions

We have generalized the regenerative randomization method described in [30] for the transient analysis of continuous time Markov models. The generalized method allows to compute two transient measures (the expected transient reward rate and the expected averaged reward rate) of rewarded continuous time Markov models with a structure covering bounding models which are useful when a complete, exact model has unmanageable size. The method has the same good properties as the well-known (standard) randomization method: numerical stability, well-controlled computation error, and ability to specify the computation error in advance, and, for large enough models and long enough times, can be significantly faster than that method. The method requires the selection of a regenerative state and its performance depends on that selection. For a class of models, class C′, including typical failure/repair models with exponential failure and repair time distributions and repair in every state with failed components, a natural selection for the regenerative state exists, and results are available assessing approximately the performance of the method for that natural selection in terms of "visible" model characteristics. Those results can be used to anticipate when the method can be expected to be significantly faster than standard randomization for class C′ models. For models not in class C′, selection of an appropriate regenerative state can be made based on user's intuition. The potentially superior efficiency of the regenerative randomization method compared to standard randomization for models not in class C′ has been illustrated using a large performability model of a fault-tolerant multiprocessor system. The generalized method allows a numerically stable, with

well-controlled and specifiable-in-advance error, solution of some large rewarded continuous time Markov models in affordable CPU times.

Acknowledgements

This work was supported by the "Comisión Interministerial de Ciencia y Tecnología" (CICYT) of the Ministry of Science and Technology of Spain under the research grants TIC95-0707-C02-02 and TAP1999-0443-C05-05.

Appendix.

Proof (Proposition 1). Using the probabilistic identity between $X = \{X(t); t \geqslant 0\}$ and $\{\hat{X}_{Q(t)}; t \geqslant 0\}$, where Q is a Poisson process with arrival rate Λ independent of $\hat{X}$, and noting that $\hat{X}_k \in S$ implies $\hat{V}_k \in \{s'_k, s_0, s_1, \ldots, s_k\}$, we have

$$P[X(t) = i] = \sum_{k=0}^{\infty} P[\hat{X}_k = i] P[Q(t) = k]$$

$$= \sum_{k=0}^{\infty} \left(P[\hat{X}_k = i \wedge \hat{V}_k = s'_k] + \sum_{l=0}^{k} P[\hat{X}_k = i \wedge \hat{V}_k = s_l] \right) P[Q(t) = k],$$

$$i \in S. \tag{A.1}$$

The probabilities $P[\hat{X}_k = i \wedge \hat{V}_k = s'_k]$, $i \in S$ can be expressed as follows. By the definition of $\hat{V}$ (11), we clearly have

$$P[\hat{X}_k = r \wedge \hat{V}_k = s'_k] = P[\hat{X}_{0:k} \in S' \wedge \hat{X}_k = r] = 0 \tag{A.2}$$

and, by the definition of $\hat{V}$ and Z' (7), taking into account that f_i are absorbing,

$$P[\hat{X}_k = i \wedge \hat{V}_k = s'_k] = P[\hat{X}_{0:k} \in S' \wedge \hat{X}_k = i] = P[Z'_k = i] = \pi'_i(k), \ i \in S'. \tag{A.3}$$

But, taking into account that $P[Z'_k = i] = 0$, $i \in S'$ for $\alpha_{S'} = 0$ (because states f_i and a are absorbing in Z' and $P[Z'_0 \in \{f_1, f_2, \ldots, f_A, a\}] = 1 - \alpha_{S'}$), from (A.3),

$$P[\hat{X}_k = i \wedge \hat{V}_k = s'_k] = 0, \quad i \in S' \text{ for } \alpha_{S'} = 0. \tag{A.4}$$

Noting that $\hat{V}_k = s'_k$ implies $\hat{X}_k \in S'$, and taking into account the definition of $\hat{V}$ and Z', and that f_i are absorbing

$$P[\hat{V}_k = s'_k] = \sum_{i \in S'} P[\hat{V}_k = s'_k \wedge \hat{X}_k = i] = \sum_{i \in S'} P[\hat{X}_{0:k} \in S' \wedge \hat{X}_k = i] = \sum_{i \in S'} P[Z'_k = i]$$

$$= \sum_{i \in S'} \pi'_i(k) = a'(k). \tag{A.5}$$

Assuming $\alpha_{S'} > 0$, which implies $a'(k) > 0$, and dividing (A.3) and (A.5):

$$P[\hat{X}_k = i \wedge \hat{V}_k = s'_k] = \frac{\pi'_i(k)}{a'(k)} P[\hat{V}_k = s'_k], \quad i \in S' \text{ for } \alpha_{S'} > 0. \tag{A.6}$$

Regarding the probabilities $P[\hat{X}_k = i \wedge \hat{V}_k = s_l]$, $i \in S'$, $1 \leqslant l \leqslant k$, assuming $P[\hat{X}_{k-l} = r] > 0$, given the definitions of $\hat{V}$ and Z (3), taking into account that f_i are absorbing, we have

$$\begin{aligned} P[\hat{X}_k = i \wedge \hat{V}_k = s_l] &= P[\hat{X}_{k-l} = r \wedge \hat{X}_{k-l+1:k} \in S' \wedge \hat{X}_k = i] \\ &= P[\hat{X}_{k-l} = r] P[\hat{X}_{k-l+1:k} \in S' \wedge \hat{X}_k = i | \hat{X}_{k-l} = r] \\ &= P[\hat{X}_{k-l} = r] P[\hat{X}'_{1:l} \in S' \wedge \hat{X}'_l = i] \\ &= P[\hat{X}_{k-l} = r] P[Z_l = i] \\ &= P[\hat{X}_{k-l} = r] \pi_i(l), \quad i \in S',\ 1 \leqslant l \leqslant k \text{ for } P[\hat{X}_{k-l} = r] > 0. \end{aligned} \tag{A.7}$$

Noting that $\hat{V}_k = s_l$, $1 \leqslant l \leqslant k$, implies (11) $\hat{X}_k \in S'$, assuming $P[\hat{X}_{k-l} = r] > 0$, taking into account the definition of $\hat{V}$ and Z, that f_i are absorbing, and that $\pi_r(l) = 0$ for $l \geqslant 1$, which implies $\sum_{i \in S'} \pi_i(l) = a(l)$,

$$\begin{aligned} P[\hat{V}_k = s_l] &= \sum_{i \in S'} P[\hat{V}_k = s_l \wedge \hat{X}_k = i] = \sum_{i \in S'} P[\hat{X}_{k-l} = r \wedge \hat{X}_{k-l+1:k} \in S' \wedge \hat{X}_k = i] \\ &= \sum_{i \in S'} P[\hat{X}_{k-l+1:k} \in S' \wedge \hat{X}_k = i \,|\, \hat{X}_{k-l} = r] P[\hat{X}_{k-l} = r] \\ &= \sum_{i \in S'} P[\hat{X}_{k-l} = r] P[\hat{X}'_{1:l} \in S' \wedge \hat{X}'_l = i] = \sum_{i \in S'} P[\hat{X}_{k-l} = r] P[Z_l = i] \\ &= P[\hat{X}_{k-l} = r] \sum_{i \in S'} P[Z_l = i] = P[\hat{X}_{k-l} = r] \sum_{i \in S'} \pi_i(l) \\ &= P[\hat{X}_{k-l} = r] a(l) > 0, \quad 1 \leqslant l \leqslant k \text{ for } P[\hat{X}_{k-l} = r] > 0, \end{aligned} \tag{A.8}$$

and dividing (A.7) and (A.8):

$$P[\hat{X}_k = i \wedge \hat{V}_k = s_l] = \frac{\pi_i(l)}{a(l)} P[\hat{V}_k = s_l], \quad i \in S',\ 1 \leqslant l \leqslant k \text{ for } P[\hat{X}_{k-l} = r] > 0. \tag{A.9}$$

Since, according to the definition of $\hat{V}$, the event $\hat{X}_{k-l}=r$ includes the event $\hat{V}_k=s_l$, $P[\hat{X}_{k-l}=r]=0$ implies $P[\hat{V}_k = s_l] = 0$ and $P[\hat{X}_k = i \wedge \hat{V}_k = s_l] = 0$, which allows to extend (A.9) to the case $P[\hat{X}_{k-l} = r] = 0$, i.e. we have

$$P[\hat{X}_k = i \wedge \hat{V}_k = s_l] = \frac{\pi_i(l)}{a(l)} P[\hat{V}_k = s_l], \quad i \in S',\ 1 \leqslant l \leqslant k. \tag{A.10}$$

For $l = 0$, $P[\hat{V}_k = s_l] = P[\hat{V}_k = s_0] = P[\hat{X}_k = r]$, $P[\hat{X}_k = i \wedge \hat{V}_k = s_l] = P[\hat{X}_k = i \wedge \hat{V}_k = s_0] = P[\hat{X}_k = i \wedge \hat{X}_k = r] = I_{i=r} P[\hat{X}_k = r]$, and $\pi_i(l)/a(l) = \pi_i(0)/a(0) = I_{i=r}$, which allows to extend (A.10) to the case $l = 0$, i.e. we have

$$P[\hat{X}_k = i \wedge \hat{V}_k = s_l] = \frac{\pi_i(l)}{a(l)} P[\hat{V}_k = s_l], \quad i \in S',\ 0 \leqslant l \leqslant k. \tag{A.11}$$

Finally, for $i=r$, $\hat{X}_k=i=r$ if and only if $\hat{V}_k=s_0$, and $P[\hat{X}_k=i\wedge\hat{V}_k=s_l]=I_{l=0}P[\hat{V}_k=s_l]$. Also, given the structure of Z, $\pi_r(l)=I_{l=0}$, and $a(0)=1$, which allows to extend (A.11) to the case $i=r$, resulting in

$$P[\hat{X}_k=i\wedge\hat{V}_k=s_l]=\frac{\pi_i(l)}{a(l)}P[\hat{V}_k=s_l],\quad i\in S,\ 0\leqslant l\leqslant k. \tag{A.12}$$

Plugging (A.2), (A.4), (A.6) and (A.12) into (A.1), splitting terms, and exchanging the indices k and l in the second summation,

$$\begin{aligned}P[X(t)=i]&=\sum_{k=0}^{\infty}\left(I_{\alpha_{S'}>0\wedge i\neq r}\frac{\pi'_i(k)}{a'(k)}P[\hat{V}_k=s'_k]+\sum_{l=0}^{k}\frac{\pi_i(l)}{a(l)}P[\hat{V}_k=s_l]\right)P[Q(t)=k]\\&=I_{\alpha_{S'}>0\wedge i\neq r}\sum_{k=0}^{\infty}\frac{\pi'_i(k)}{a'(k)}P[\hat{V}_k=s'_k]P[Q(t)=k]\\&\quad+\sum_{k=0}^{\infty}\frac{\pi_i(k)}{a(k)}\sum_{l=k}^{\infty}P[\hat{V}_l=s_k]P[Q(t)=l],\quad i\in S.\end{aligned} \tag{A.13}$$

Now, using the probabilistic identity between $V=\{V(t);t\geqslant 0\}$ and $\{\hat{V}_{Q(t)};t\geqslant 0\}$, where Q is a Poisson process with arrival rate Λ independent of $\hat{V}$, and considering that $P[\hat{V}_l=s'_k]=0$ for $l\neq k$ and $P[\hat{V}_l=s_k]=0$ for $l<k$,

$$P[V(t)=s'_k]=\sum_{l=0}^{\infty}P[\hat{V}_l=s'_k]P[Q(t)=l]=P[\hat{V}_k=s'_k]P[Q(t)=k], \tag{A.14}$$

$$P[V(t)=s_k]=\sum_{l=0}^{\infty}P[\hat{V}_l=s_k]P[Q(t)=l]=\sum_{l=k}^{\infty}P[\hat{V}_l=s_k]P[Q(t)=l]. \tag{A.15}$$

Finally, combining (A.14) and (A.15) with (A.13),

$$P[X(t)=i]=I_{\alpha_{S'}>0\wedge i\neq r}\sum_{k=0}^{\infty}\frac{\pi'_i(k)}{a'(k)}P[V(t)=s'_k]+\sum_{k=0}^{\infty}\frac{\pi_i(k)}{a(k)}P[V(t)=s_k],\quad i\in S.\qquad\square$$

Proof (Theorem 2). Using Theorem 1, $b(k),b'(k),r_{f_i}\leqslant r_{\max}$, the inequalities $P[V_{K,L}(t)=s_k]\leqslant P[V(t)=s_k]$, $P[V_{K,L}(t)=s'_k]\leqslant P[V(t)=s'_k]$ and $P[V_{K,L}(t)=f_i]\leqslant P[V(t)=f_i]$, which follow from (12), and $P[V_{K,L}(t)=a]=1-P[V_{K,L}(t)\in S_{K,L}]$, where $S_{K,L}=\{s_0,s_1,\ldots,s_K\}\cup\{s'_0,s'_1,\ldots,s'_L\}\cup\{f_1,f_2,\ldots,f_A\}$,

$$\begin{aligned}&ETRR(t)-ETRR^{K,L,a}(t)=ETRR^{V}(t)-ETRR^{K,L,a}(t)\\&=\sum_{k=0}^{\infty}b(k)P[V(t)=s_k]+\sum_{k=0}^{\infty}b'(k)P[V(t)=s'_k]+\sum_{i=1}^{A}r_{f_i}P[V(t)=f_i]\\&\quad-\left(\sum_{k=0}^{K}b(k)P[V_{K,L}(t)=s_k]+\sum_{k=0}^{L}b'(k)P[V_{K,L}(t)=s'_k]+\sum_{i=1}^{A}r_{f_i}P[V_{K,L}(t)=f_i]\right)\end{aligned}$$

$$
\begin{aligned}
&= \sum_{k=0}^{K} b(k)(P[V(t)=s_k] - P[V_{K,L}(t)=s_k]) + \sum_{k=K+1}^{\infty} b(k)P[V(t)=s_k] \\
&\quad + \sum_{k=0}^{L} b'(k)(P[V(t)=s'_k] - P[V_{K,L}(t)=s'_k]) + \sum_{k=L+1}^{\infty} b'(k)P[V(t)=s'_k] \\
&\quad + \sum_{i=1}^{A} r_{f_i}(P[V(t)=f_i] - P[V_{K,L}(t)=f_i]) \\
&\leqslant r_{\max} \sum_{k=0}^{K} (P[V(t)=s_k] - P[V_{K,L}(t)=s_k]) + r_{\max} \sum_{k=K+1}^{\infty} P[V(t)=s_k] \\
&\quad + r_{\max} \sum_{k=0}^{L} (P[V(t)=s'_k] - P[V_{K,L}(t)=s'_k]) + r_{\max} \sum_{k=L+1}^{\infty} P[V(t)=s'_k] \\
&\quad + r_{\max} \sum_{i=1}^{A} (P[V(t)=f_i] - P[V_{K,L}(t)=f_i]) \\
&= r_{\max} \left(\sum_{k=0}^{\infty} P[V(t)=s_k] + \sum_{k=0}^{\infty} P[V(t)=s'_k] + \sum_{i=1}^{A} P[V(t)=f_i] \right. \\
&\quad \left. - \sum_{k=0}^{K} P[V_{K,L}(t)=s_k] - \sum_{k=0}^{L} P[V_{K,L}(t)=s'_k] - \sum_{i=1}^{A} P[V_{K,L}(t)=f_i] \right) \\
&= r_{\max}(1 - P[V_{K,L}(t) \in S_{K,L}]) = r_{\max} P[V_{K,L}(t)=a].
\end{aligned}
$$

For the case $\alpha_{S'} = 0$, the result $ETRR(t) - ETRR^{K,a}(t) \leqslant r_{\max} P[V_K(t) = a]$ can be proved similarly. □

References

[1] Malhotra M, Muppala JK, Trivedi KS. Stiffness-tolerant methods for transient analysis of stiff Markov Chains. Microelectronics and Reliability 1994;34(11):1825–41.

[2] Malhotra M. A computationally efficient technique for transient analysis of repairable Markovian systems. Performance Evaluation 1995;1–2:311–31.

[3] Reibman A, Trivedi KS. Numerical transient analysis of Markov models. Computers and Operations Research 1988;15:19–36.

[4] Grassman WK. Rounding errors in certain algorithms involving Markov chains. ACM Transactions on Mathematical Software 1993;19(4):496–508.

[5] Grassman WK. Transient solutions in Markovian queuing systems. Computers and Operations Research 1977;4: 47–53.

[6] Gross D, Miller DR. The randomization technique as a modelling tool and solution procedure for transient Markov processes. Operations Research 1984;32:343–61.
[7] Béounes C, Aguéra M, Arlat J, Bachman S, Bourdeu C, Doucet JE, Kanoun K, Laprie JC, Metge S, Moreira de Souza J, Powell D, Spiesser P. SURF-2: A program for dependability evaluation of complex hardware and software systems. Proceedings of the 23rd IEEE International Symposium on Fault-Tolerant Computing (FTCS-23), Toulouse, June 1993. p. 142–50.
[8] Ciardo G, Muppala J, Trivedi K. SPNP: Stochastic Petri Net Package. Proceedings of the Third IEEE International Workshop on Petri Nets and Performance Models, PNPM89, Kyoto, December 1989. p. 142–51.
[9] Couvillon J, Freire R, Johnson R, Obal II W, Qureshi A, Rai M, Sanders W, Tvedt J. Performability modelling with UltraSAN. IEEE Software, 1991;8(5):69–80.
[10] Goyal A, Carter WC, de Souza e Silva E, Lavenberg SS. The system availability estimator. Proceedings of the 16th IEEE International Symposium on Fault-Tolerant Computing (FTCS-16), July 1986. p. 84–9.
[11] Kijima M. Markov processes for stochastic modeling. Cambridge: University Press, 1997.
[12] Spiegel MR. Mathematical handbook of formulas and tables. New York: McGraw-Hill, 1970.
[13] Bowerman PN, Nolty RG, Schener EM. Calculation of the Poisson cumulative distribution function. IEEE Transactions on Reliability 1990;39(2):158–61.
[14] Fox BL, Glynn PW. Computing Poisson probabilities. Communications of the ACM 1988;31:440–5.
[15] Knüsel L. Computation of the chi-square and Poisson distribution. SIAM Journal of Scientific and Statistical Computing 1986;7(3):1022–36.
[16] Moorsel AP, Sanders WH. Transient solution of Markov models by combining adaptive and standard uniformization. IEEE Transactions on Reliability 1997;46(3):430–40.
[17] Abramowitz M, Stegun IA, editors. Handbook of mathematical functions. New York: Dover, 1964.
[18] Ross SM. Stochastic processes. New York: Wiley, 1983.
[19] Rubino G, Sericola B. Interval availability distribution computation. Proceedings of the 23rd International Symposium on Fault-Tolerant Computing (FTCS-23), Toulouse, June 1993. p. 48–55.
[20] Rubino G, Sericola B. Interval availability analysis using denumerable Markov processes: application to multiprocessor subject to breakdowns and repair. IEEE Transactions on Computers 1995;44(2):286–91.
[21] de Souza e Silva E, Gail HR. Calculating cumulative operational time distributions of repairable computer systems. IEEE Transactions on Computers 1986;35:322–32.
[22] Nabli H, Sericola B. Performability analysis: a new algorithm. IEEE Transactions on Computers 1996;45(4):491–4.
[23] Qureshi MA, Sanders WH. Reward model solution methods with impulse and rate rewards: an algorithm and numerical results. Performance Evaluation 1994;20:413–36.
[24] de Souza e Silva E, Gail HR. Calculating availability and performability measures of repairable computer systems using randomization. Journal of the ACM 1989;34:179–99.
[25] de Souza e Silva E, Gail HR. An algorithm to calculate transient distributions of cumulative rate and impulse based reward. Communications in Statistics–Stochastic Models 1998;14(3):509–36.
[26] Miller DR. Reliability calculation using randomization for Markovian fault-tolerant computing systems. Proceedings of the 13th IEEE International Symposium on Fault-Tolerant Computing (FTCS-13), June 1983. p. 284–9.
[27] Melamed B, Yadin M. Randomization procedures in the computation of cumulative-time distributions over discrete state Markov processes. Operations Research 1984;32:926–44.
[28] Moorsel AP, Sanders WH. Adaptive uniformization. Communications in Statistics–Stochastic Models 1994;10(3):619–47.
[29] Sericola B. Availability analysis of repairable computer systems and stationarity detection. IEEE Transactions on Computers 1999;48(11):1166–72.
[30] Carrasco JA. Transient analysis of large Markov models with absorbing states using regenerative randomization. Technical Report DMSD_99_2, Universitat Politècnica de Catalunya, February 1999, available at ftp://ftp-eel.upc.es/techreports.
[31] Neuts F. Matrix-geometric solutions in stochastic models. An algorithmic approach. New York: Dover Publications Inc., 1994.
[32] Bobbio A, Telek M. A benchmark for PH estimation algorithms: results for acyclic-PH. Communications in Statistics—Stochastic Models 1994;10(3):661–77.

[33] Bhandarkar D. Analysis of memory interference in multiprocessors. IEEE Transactions on Computers 1975;C-24: 897–908.
[34] Smith RM, Trivedi KS, Ramesh AV. Performability analysis: measures, an algorithm, and a case study. IEEE Transactions on Computers 1988;37(4):406–17.

Juan A. Carrasco received the Engineer degree in Industrial Engineering from the Polytechnical University of Catalonia, Barcelona, Spain, in 1982, the M.Sc. degree in Computer Science from Stanford University in 1987, and the Ph.D. degree in Industrial Engineering from UPC, also in 1987. He is currently an associate professor in the Electronics Engineering Department of UPC. He is currently coleader of a research team working on reliability and fault-tolerance of electronic systems. His research interests encompass methodologies for the modeling and evaluation of fault-tolerant systems, including bounding methods, numerical techniques, and combinatorial methods, topics in which he has published more than 40 papers in refereed journals and conference proceedings. He is a Senior member of the IEEE.