Computer Science Department Technical Report
University of California
Los Angeles, CA 90024-1596

CALCULATING TRANSIENT DISTRIBUTIONS OF CUMULATIVE REWARD

E. de Souza e Silva
H. R. Gail

October 1993
CSD-930033

Calculating Transient Distributions of Cumulative Reward

Edmundo de Souza e Silva[1]
UCLA Computer Science Department
Los Angeles, CA 90024

H. Richard Gail
IBM Thomas J. Watson Research Center
Yorktown Heights, NY 10598

Abstract

Markov reward models have been employed to obtain performability measures of computer and communication systems. In these models, a continuous time Markov chain is used to represent changes in the system structure, usually caused by faults and repairs of its components, and reward rates are assigned to states of the model to indicate some measure of accomplishment at each structure. A procedure to calculate numerically the distribution of the reward accumulated over a finite observation period is presented. The development is based solely on probabilistic arguments, and the final recursion is quite simple. The algorithm has a low computational cost in terms of model parameters, and, in fact, it is linear in a parameter that is smaller than the number of rewards. The approach follows the general methodology previously developed by the authors to calculate transient measures using uniformization, but the complexity of the final algorithm to calculate the distribution of cumulative reward is drastically reduced.

[1]This work was done while E. de Souza e Silva was on leave from the Federal University of Rio de Janeiro partially supported by grants from CNPq(Brazil) and NSF CCR-9215064.

1 Introduction

Modeling and analysis plays an important role in the design of computer and communication systems. It allows the designer to predict the behavior of these systems and evaluate design alternatives, in order to increase the efficiency and reliability of the system being built. In the past ten years, much attention has been given to *performability* models, which represent the performance a system can achieve during an observation period as its structure evolves over time.

In the seminal work of Meyer [17], a modeling framework for the definition and evaluation of performability measures was introduced. In broad terms, Meyer discussed determining the probability that a system performs at a given level of accomplishment during an observation period. Many measures can be mapped into Meyer's definition of performability. As an example, consider a Markov process that is used to model the changes in the structure of the system as time evolves. These changes are usually caused by the failure of the system components and the repairs made to the failed units. The system performance is integrated into the model by assigning rewards, obtained from a separate performance model, to the states. Each reward represents the steady state performance of the system when in a particular configuration. In [19] there is an extensive discussion of many concepts related to performability, and in [5] several performability measures are considered and solution techniques for their calculation are surveyed. Many examples which illustrate the use of performability models are presented in [23].

This paper is concerned with the calculation of an important performability measure, the distribution of cumulative reward over a finite interval for which reward rates are associated with states of a Markov model. In fact, this is also an interesting theoretical problem that has applications beyond the performability area. Formally, consider a homogeneous continuous time Markov process $\mathcal{X} = \{X(t) : t \geq 0\}$ with finite state space $\mathcal{S} = \{s_i : i = 1, \cdots, M\}$. To each state $s \in \mathcal{S}$ we assign a reward rate from a given set of rewards $\{r_1, \ldots, r_{K+1}\}$. The random variable $\mathrm{IR}(t)$, the instantaneous reward at time t, is simply $\mathrm{IR}(t) = r_{c(s)}$ if $X(t) = s$, where $c(s)$ is the index of the reward rate associated with state s. The cumulative reward during an observation period $(0, t)$ averaged over t is

$$\mathrm{ACR}(t) = \frac{1}{t} \int_0^t \mathrm{IR}(\tau) d\tau.$$

The problem of calculating the cumulative reward distribution has been studied by many researchers. In one of the first papers on performability [1], Beaudry developed a method to calculate the distribution of accumulated reward until absorption occurs for a Markov chain with absorbing states. Subsequently, several solution techniques were developed to calculate the distribution of cumulative reward when the generator matrix $\mathbf{Q}$ of the Markov process that describes changes in system structure is lower triangular (e.g., models for nonrepairable systems). Such techniques are discussed in [8, 9, 10, 18].

Determining the distribution of cumulative reward over a finite interval for Markov models with a general **Q** matrix is a more difficult problem. For these models, Laplace transform methods have been applied [14, 16, 22]. In [4] a methodology for calculating performability measures was developed using the uniformization technique [15]. In particular, a recursive algorithm to calculate the distribution of cumulative reward was developed based on probabilistic arguments. The procedure is best suited to cases for which the number of rewards is small, since its computational complexity is combinatorial with the number of distinct rewards of the model. Donatiello and Grassi [7] combined uniformization and the basic methodology of the Laplace transform techniques cited above to obtain a recursive expression for the distribution function of the cumulative reward. They derived a double Laplace transform equation for the distribution and found a way to analytically invert it, and thus no numerical inversion is necessary in their method unlike previous Laplace transform work. Furthermore, the computational complexity is polynomial with the number of states and the number of rewards. More recently, Pattipati *et al* [20] determined the distribution of cumulative reward by numerically solving a partial differential equation that they obtained.

In this work we develop a new algorithm for the distribution of cumulative reward based on the general methodology of [4]. The cost of the new algorithm is drastically reduced in comparison with the one in [4], and, in fact, it is linear in a parameter that is smaller than the number of distinct rewards. Furthermore, the algorithm is very simple to describe and implement, and the method has a nice probabilistic interpretation.

The remainder of the paper is organized as follows. In Section 2 the notation used throughout the paper is introduced and the necessary background needed is presented. In Section 3 an overview of the methodology of [4] for calculating transient measures based on uniformization is given. In Section 4 the algorithm is developed and computational issues related to its implementation are addressed. A simple example of the application of the algorithm is presented in Section 5. Section 6 concludes the paper.

2 Notation and Background Material

In this section we present the background material and introduce the notation that will be used throughout the paper. We start by presenting the uniformization technique, which was introduced by Jensen in [15] and is very useful for calculating transient measures. This technique, also called randomization or Jensen's method, has been widely used by many authors, and it has been covered in books and survey articles [2, 5, 11, 12, 13, 21].

Consider a homogeneous continuous time Markov process $\mathcal{X} = \{X(t) : t \geq 0\}$ with finite state space $\mathcal{S} = \{s_i : i = 1, \ldots, M\}$ and transition rate matrix

$$\mathbf{Q} = \begin{bmatrix} -q_1 & q_{12} & \cdots & q_{1M} \\ q_{21} & -q_2 & \cdots & q_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ q_{M1} & q_{M2} & \cdots & -q_M \end{bmatrix},$$

where q_{ij} is the exponential transition rate from state s_i to s_j and $q_i = \sum_{j \neq i} q_{ij}$ is the total rate out of state s_i. The uniformization method involves performing a simple transformation on the original process, so that the state probabilities at the end of an interval $(0, t)$ can be obtained by analyzing a discrete time Markov chain.

The rates q_i are uniformly bounded since the state space is finite, so we can always find $\Lambda \geq \max_{1 \leq i \leq M}\{q_i\}$. Now transform the original process $\mathcal{X}$ by allowing transitions back to the same state so that the residence time in state s_i, $i = 1, \ldots, M$, before a transition (out of s_i or back to s_i) is exponentially distributed with parameter Λ. Furthermore, we preserve the probability of jumping to a state s_j other than s_i. This is accomplished by adding fictitious self transitions so that with probability q_{ij}/Λ the process jumps to state s_j, $j \neq i$, and with probability q_i/Λ the process immediately returns to s_i after a transition from that state.

Since the residence time in any state before a transition occurs is exponential with the same rate Λ, the transitions in an interval $(0, t)$ are governed by a Poisson process with this rate. In more detail, $\mathcal{X}$ can be considered as a discrete time Markov chain subordinated to a Poisson process as follows. Let $\mathcal{Z} = \{Z_n : n = 0, 1, \ldots\}$ be a discrete time Markov chain with finite state space $\mathcal{S}$ and transition matrix $\mathbf{P} = \mathbf{I} + \mathbf{Q}/\Lambda$ (where $\mathbf{I}$ is the $M \times M$ identity matrix), and let $\mathcal{N} = \{N(t) : t \geq 0\}$ be a Poisson process with rate Λ that is independent of $\mathcal{Z}$. Then $X(t) = Z_{N(t)}$ for $t \geq 0$, which indicates that the transition times occur according to the Poisson process $\mathcal{N}$ and the probability of being in state s_j after a jump from s_i is given by the entry p_{ij} of $\mathbf{P}$. As a consequence, if we condition on the number of transitions n in $(0, t)$, we can obtain the probability $p_i(t)$ that the process $\mathcal{X}$ is in state s_i at t as

$$\mathbf{p}(t) = \sum_{n=0}^{\infty} e^{-\Lambda t} \frac{(\Lambda t)^n}{n!} \boldsymbol{\pi}(0) \mathbf{P}^n, \tag{1}$$

where $\mathbf{p}(t) = \langle p_1(t), \ldots, p_M(t) \rangle$ and $\boldsymbol{\pi}(0)$ is the initial probability vector. Equation (1) is the basic equation of the uniformization method. In order to evaluate (1) numerically, the infinite series is truncated to a value N, and the resulting error can be easily bounded from the remaining Poisson terms.

One of the main advantages of the uniformization technique is its probabilistic interpretation. In fact many measures of interest can be obtained based on this interpretation (for a survey of the measures that can be calculated using uniformization see [5]). In the next section we review a general methodology for obtaining the distribution of accumulated reward over a finite interval.

3 A Methodology for Calculating Transient Measures

We consider the problem of calculating the distribution of accumulated reward over a finite observation period. The basic methodology we use was developed in [4] (see also [5]) and is surveyed below.

Assume that the continuous time Markov chain $\mathcal{X}$ of the reward model being analyzed has been uniformized. Therefore, as indicated in the previous section, the transitions of the chain $\mathcal{X}$ are governed by a Poisson process. Assume that n transitions occur during the observation period $(0,t)$ at times $0 < \tau_1 < \cdots < \tau_n < t$. These events split $(0,t)$ into $n+1$ intervals with lengths $Y_1 = \tau_1, Y_2 = \tau_2 - \tau_1, \ldots, Y_{n+1} = t - \tau_n$. Figure 1 illustrates the intervals and their lengths. The state of the process $\mathcal{X}$ during each of these intervals

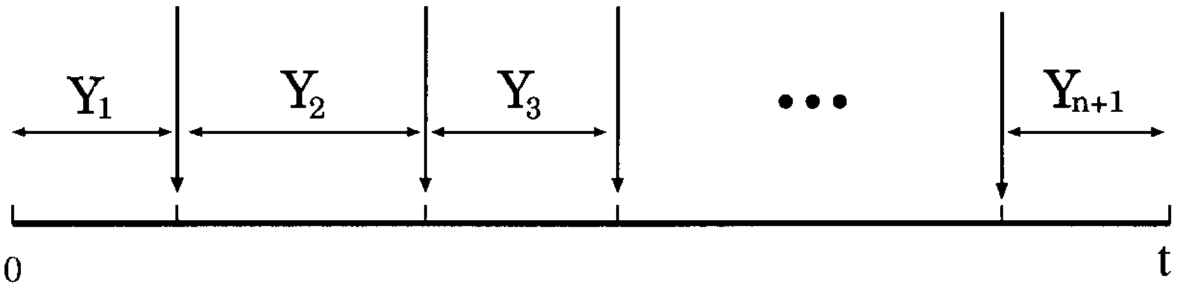


Figure 1: Interval lengths Y_i.

is given by the uniformized discrete time chain $\mathcal{Z}$. The probability that the process is in s_i during the kth interval is equal to $P[z_k = s_i] = \pi_i(k)$, where $\boldsymbol{\pi}(k) = \boldsymbol{\pi}(0)\mathbf{P}^{k-1}$ with $\boldsymbol{\pi}(0)$ the initial state probability distribution vector and $\mathbf{P}$ the probability transition matrix of $\mathcal{Z}$.

Each interval is associated with a reward which is based on the state of the process during the interval. We assume that there are $K+1$ distinct rewards $r_1 > r_2 > \cdots > r_{K+1} \geq 0$ associated with states of the process $\mathcal{X}$. Given that n transitions have occurred in $(0,t)$, let k_i, $i = 1, \ldots, K+1$, be the number of intervals that are associated with reward r_i, and define $\mathbf{k} = \langle k_1, \ldots, k_{K+1} \rangle$ and $||\mathbf{k}|| = k_1 + \cdots + k_{K+1}$. Note that $||\mathbf{k}|| = n+1$ given n transitions, and there are $\binom{K+n+1}{n+1}$ vectors such that $||\mathbf{k}|| = n+1$. We refer to a specific vector $\mathbf{k}$ as a *coloring*, since assigning rewards to intervals may be thought of as coloring the intervals with a different color for each distinct reward.

Let $M(t)$ be the measure we wish to calculate. For example, $M(t)$ may be the probability that the reward accumulated during $(0, t)$ averaged over t is greater than a given value. If we condition on n transitions and a coloring $\mathbf{k}$, we have

$$M(t) = \sum_{n=0}^{\infty} e^{-\Lambda t} \frac{(\Lambda t)^n}{n!} \sum_{||\mathbf{k}||=n+1} \Gamma[n, \mathbf{k}] M(t, n, \mathbf{k}) \tag{2}$$

where

$$\Gamma[n, \mathbf{k}] = P[\text{coloring } \mathbf{k} | n \text{ transitions}] \tag{3}$$

and

$$M(t, n, \mathbf{k}) = M(t) | n \text{ transitions, coloring } \mathbf{k}. \tag{4}$$

In order to apply (2), both $\Gamma[n, \mathbf{k}]$ and $M(t, n, \mathbf{k})$ have to be calculated in an efficient manner. For many measures of interest (see [4, 5]), simple recursive expressions can be found for $\Gamma[n, \mathbf{k}]$. Furthermore, certain measures $M(t, n, \mathbf{k})$ can be determined by using the property that a particular sample path of the discrete time chain $\mathcal{Z}$ influences the measure only through the coloring of the intervals and not, for instance, through the order in which the rewards appear in the sample path. More formally, let $G_{\mathbf{k}} \subset \mathcal{S}^{n+1}$ be the set of all possible sample paths of $\mathcal{Z}$ such that the first n transitions yield the coloring $\mathbf{k}$. Then, for many measures, $M(t, n, \mathbf{k}) = M(t, n, \mathbf{k}, \nu)$, where $M(t, n, \mathbf{k}, \nu)$ is the measure further conditioned on a path $\nu \in G_{\mathbf{k}}$. In order to show this result, the probabilistic interpretation of the uniformization procedure is exploited. That is, the independence of the Poisson process $\mathcal{N}$ and the discrete time chain $\mathcal{Z}$ is used as well as a property of the Y_i called *exchangeability.*

In the following section we use this methodology to obtain the distribution of accumulated reward during $(0, t)$ averaged over t. The recursion we find is linear with the number of states, linear in a parameter that is smaller than the number of rewards (colors), and cubic with the truncation point N. This is a major improvement over the results presented in [4], for which the recursion was combinatorial with the number of colors. However, the approach still preserves the probabilistic development in [4].

4 The Algorithm

Our interest is in calculating the distribution of the total reward accumulated during $(0, t)$ averaged over t, where r_i is the reward accumulated per unit time in any state associated with it. That is, we wish to calculate $M(t) = P[\text{ACR}(t) > r]$. We may assume that $r_1 \geq r \geq r_{K+1}$ to avoid trivial cases.

4.1 Preliminary Results

Using the methodology reviewed in the previous section, we first determine $\Gamma[n, \mathbf{k}]$. Let $\Gamma_s[n, \mathbf{k}]$ be the probability of a coloring $\mathbf{k}$ given n transitions *and* the state visited after the last transition is s. Then $\Gamma_s[n, \mathbf{k}]$ can be obtained recursively as follows. If s' and s are the states visited after the $(n-1)$st and nth transitions, respectively, then the coloring $\mathbf{k}$ after the nth transition is equal to the previous coloring except that the entry corresponding to the color associated with s has been incremented by 1. Conditioning on the state visited after the $(n-1)$st transition and recalling that state transitions are governed by $\mathcal{Z}$, we have

$$\Gamma_s[n, \mathbf{k}] = \sum_{s' \in \mathcal{S}} \Gamma_{s'}[n-1, \mathbf{k} - \mathbf{1}_{c(s)}] p_{s's}. \tag{5}$$

Here $c(s)$ is the index of the reward (color) associated with state s, and $\mathbf{1}_{c(s)}$ is a unit vector of length $K+1$ with 1 in position $c(s)$. The initial conditions are

$$\Gamma_s[0, \mathbf{1}_i] = \begin{cases} \pi_s(0) & \text{if } i = c(s) \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

where $\pi_s(0)$ is the probability that the process starts in state s. From the definition of $\Gamma_s[n, \mathbf{k}]$ we have

$$\Gamma[n, \mathbf{k}] = \sum_{s \in \mathcal{S}} \Gamma_s[n, \mathbf{k}]. \tag{7}$$

This recursion for $\Gamma[n, \mathbf{k}]$ was obtained in [4].

We now determine $M(t, n, \mathbf{k}) = P[\text{ACR}(t) | n, \mathbf{k}]$. Given n transitions and a coloring $\mathbf{k}$, let the random variable ζ_i be the sum of the lengths of intervals associated with reward r_i. Figure 2 illustrates the intervals corresponding to two different rewards for a particular sample path $\nu \in G_{\mathbf{k}}$. Clearly we have

$$\text{ACR}(t) | n, \mathbf{k} = \frac{1}{t} \sum_{i=1}^{K+1} r_i \zeta_i. \tag{8}$$

Let $U_1, U_2, \ldots, U_n$ be independent and identically distributed random variables uniform on $(0,1)$, and let $U_{(1)}, U_{(2)}, \ldots, U_{(n)}$ be their order statistics (for notational convenience set $U_{(0)} = 0$ and $U_{(n+1)} = 1$). It is well-known that, conditioned on n events during $(0, t)$, the time of the ith transition τ_i has the same distribution as $tU_{(i)}$. Therefore, we may make the identification $Y_1 = tU_{(1)}, Y_2 = t(U_{(2)} - U_{(1)}), \ldots, Y_{n+1} = t(1 - U_{(n)})$. Although the Y_i are dependent random variables, they are *exchangeable* [3], i.e., the joint distribution of the Y_i is invariant under any permutation of $1, \ldots, n+1$.

As a consequence of the exchangeability property of the Y_i and the fact that the Poisson process $\mathcal{N}$ and the discrete time chain $\mathcal{Z}$ are independent, we may group intervals with the

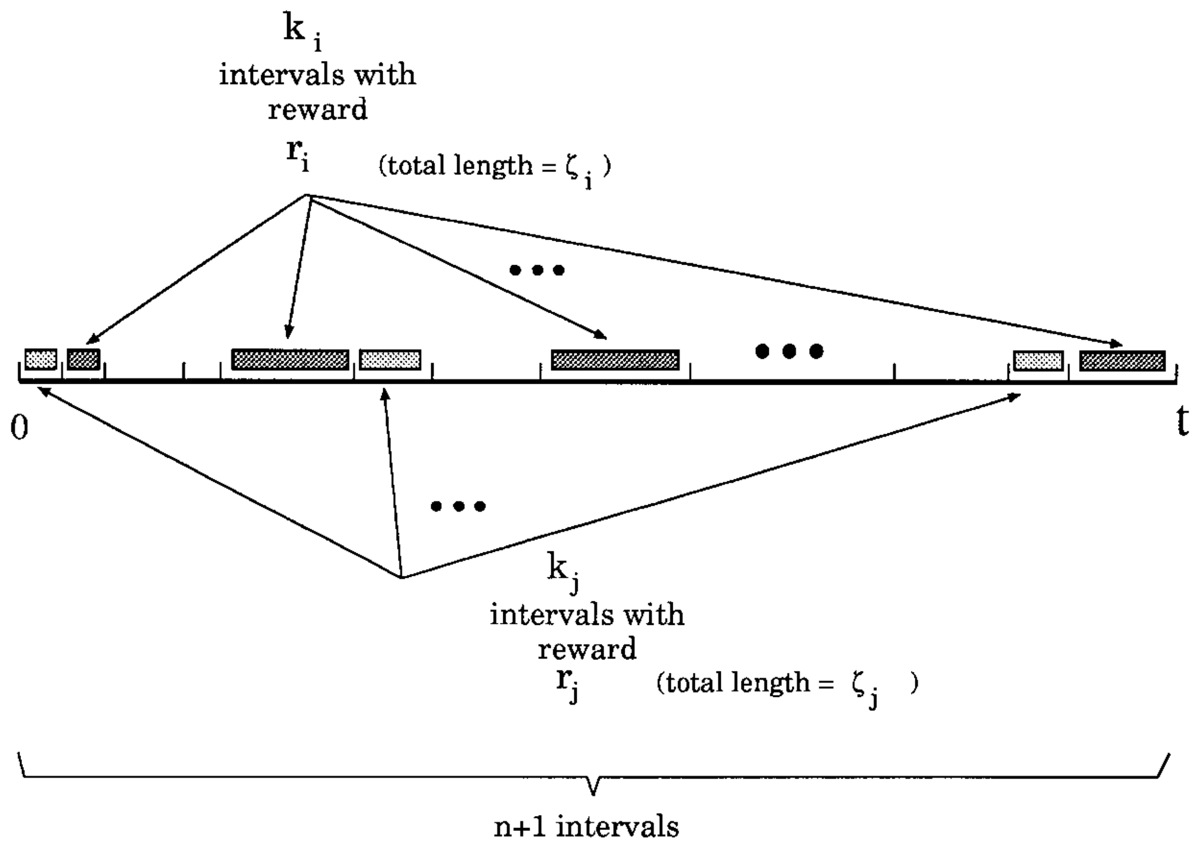


Figure 2: Rewards associated with intervals for a sample path ν given n transitions.

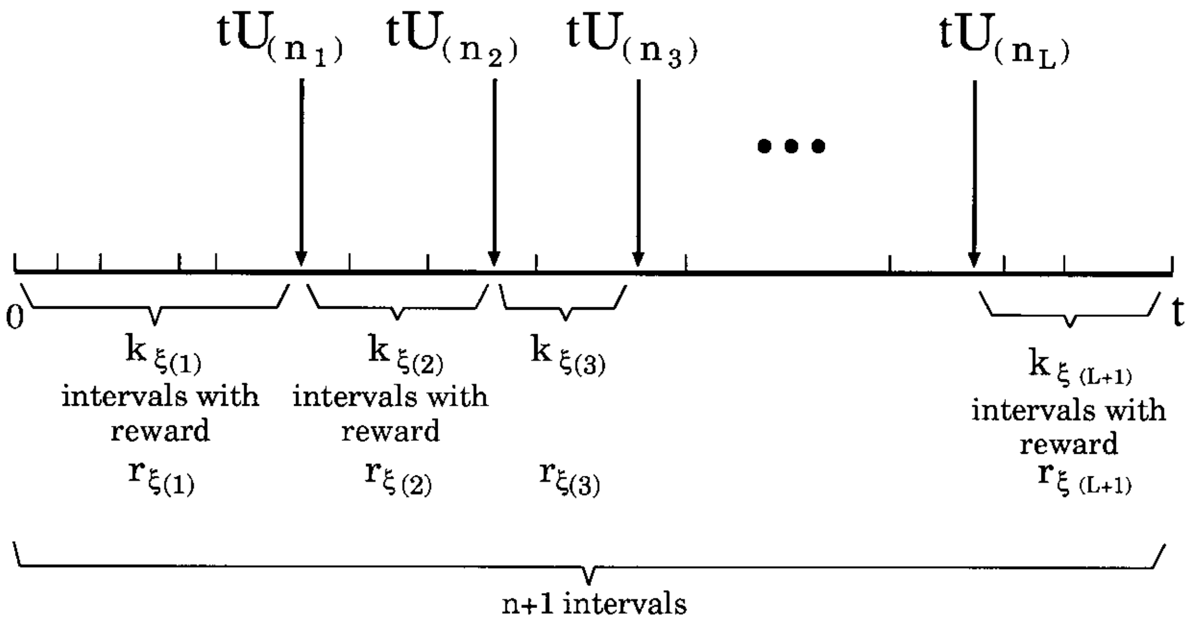
same reward in sequence when determining the distribution of ACR(t) given n transitions and a coloring $\mathbf{k}$, so that the first k_1 intervals have reward r_1, the next k_2 intervals have reward r_2, and so on. Note that if $k_i = 0$, then no interval has reward r_i. Suppose that the vector $\mathbf{k}$ has $L+1$ nonzero entries, i.e., only $L+1$ distinct colors out of the possible $K+1$ were used to color the $n+1$ intervals. Let $\xi(1) < \cdots < \xi(L+1)$ be the indices of these colors, and define $n_j = \sum_{l=1}^{j} k_{\xi(l)}$ for $j = 1, \ldots, L$. Figure 3 illustrates the notation used and shows the intervals and this coloring.

From the above discussion we may make the identification

$$\mathrm{ACR}(t)|n, \mathbf{k} =$$
$$\frac{1}{t}\left[r_{\xi(1)}tU_{(n_1)} + r_{\xi(2)}t(U_{(n_2)} - U_{(n_1)}) + \cdots + r_{\xi(L)}t(U_{(n_L)} - U_{(n_{L-1})}) + r_{\xi(L+1)}t(1 - U_{(n_L)})\right].$$

Therefore,

$$\mathrm{ACR}(t)|n, \mathbf{k} = \sum_{j=1}^{L}(r_{\xi(j)} - r_{\xi(j+1)})U_{(n_j)} + r_{\xi(L+1)}, \tag{9}$$



$tU_{(n_1)}$
$tU_{(n_2)}$
$tU_{(n_3)}$
$tU_{(n_L)}$
0
t
$k_{\xi(1)}$ intervals with reward $r_{\xi(1)}$
$k_{\xi(2)}$ intervals with reward $r_{\xi(2)}$
$k_{\xi(3)}$
$r_{\xi(3)}$
$k_{\xi(L+1)}$ intervals with reward $r_{\xi(L+1)}$
n+1 intervals

Figure 3: Intervals given n and $\mathbf{k}$.

so that

$$P[\mathrm{ACR}(t) > r|n,\mathbf{k}] = P\left[\sum_{j=1}^{L}(r_{\xi(j)} - r_{\xi(j+1)})U_{(n_j)} > r - r_{\xi(L+1)}\right]. \tag{10}$$

From (10) we see that in order to find $M(t,n,\mathbf{k}) = P[\mathrm{ACR}(t) > r|n,\mathbf{k}]$, we need to obtain the distribution of a linear combination of uniform order statistics on $(0,1)$. A solution to this problem was found by Weisberg [24] (see also [4]). The following is a direct consequence of the results of Weisberg.

Lemma 1 *For $n \geq 0$ and $\|\mathbf{k}\| = n+1$, we have*

$$P[\mathrm{ACR}(t) > r|n,\mathbf{k}] = \sum_{i=1}^{m} \frac{f_i^{(k_i-1)}(r_i,\mathbf{k})}{(k_i-1)!}, \tag{11}$$

where m is the largest index i such that $r_i \geq r$ and $f_i^{(k_i-1)}$ is the (k_i-1)st derivative of the function

$$f_i(x,\mathbf{k}) = \frac{(x-r)^n}{\prod_{\substack{j=1\\ j\neq i}}^{K+1}(x-r_j)^{k_j}} \tag{12}$$

(we define $f_i^{(0)} = f_i$ and $f_i^{(l)} = 0$ for $l < 0$).

Proof: Recall that $\xi(1) < \cdots < \xi(L+1)$ are the indices corresponding to the nonzero entries of $\mathbf{k}$. If $\xi(1) > m$, then $r_{\xi(L+1)} < \cdots < r_{\xi(1)} < r$. Thus $P[\mathrm{ACR}(t) > r|n,\mathbf{k}] = 0$, and equation (11) holds since $f_i^{(k_i-1)} = 0$ for $i = 1, \ldots, m$. Now consider the case for which $\xi(1) \le m$. When $r_{\xi(L+1)} \le r$, then Weisberg's result applies and is

$$P[\mathrm{ACR}(t) > r|n, \mathbf{k}] = \sum_{\substack{j=1 \\ \xi(j) \le m}}^{L} \frac{g_{\xi(j)}^{(k_{\xi(j)}-1)}(r_{\xi(j)} - r_{\xi(L+1)}, \mathbf{k})}{(k_{\xi(j)} - 1)!}, \tag{13}$$

where, for $\xi(j) \le m$, $g_{\xi(j)}$ is the function

$$g_{\xi(j)}(x, \mathbf{k}) = \frac{\left[x - (r - r_{\xi(L+1)})\right]^n}{\prod_{\substack{i=1 \\ i \ne j}}^{L+1} \left[x - (r_{\xi(i)} - r_{\xi(L+1)})\right]^{k_{\xi(i)}}}. \tag{14}$$

Since $k_i = 0$ for $i \notin \{\xi(1), \ldots, \xi(L+1)\}$, it is easy to see that evaluating $g_{\xi(j)}^{(l)}(x, \mathbf{k})$ (for $l = 0, 1, \ldots$) at the point $r_{\xi(j)} - r_{\xi(L+1)}$ is equivalent to evaluating $f_{\xi(j)}^{(l)}(x, \mathbf{k})$ at the point $r_{\xi(j)}$. Also, the sum of the $f_i^{(k_i-1)}$ can be taken over all $i = 1, \ldots, m$ and not just over the $\xi(j) \le m$, since $f_i^{(k_i-1)} = 0$ when $k_i = 0$. Thus (11) holds in this case.

When $r_{\xi(L+1)} > r$, then $P[\mathrm{ACR}(t) > r|n, \mathbf{k}] = 1$ since all $n+1$ intervals in $(0, t)$ have rewards that are greater than r (see Figure 3). To show that this case is covered by (11), first suppose that r is one of the $K+1$ rewards. Then, by the definition of m, it is the mth reward, i.e., $r_m = r$ (and also $k_m = 0$). Consider the vector $\mathbf{k} + \mathbf{1}_m$. The rewards corresponding to the nonzero entries of $\mathbf{k} + \mathbf{1}_m$ all have value at least r, so that $P[\mathrm{ACR}(t) > r|n+1, \mathbf{k} + \mathbf{1}_m] = 1$. Thus we need only show that this distribution satisfies (11). Since m is the largest index with a nonzero entry in $\mathbf{k} + \mathbf{1}_m$ and $r_m = r$, Weisberg's result applies. The mth entry of $\mathbf{k} + \mathbf{1}_m$ is 1, so

$$P[\mathrm{ACR}(t) > r|n+1, \mathbf{k} + \mathbf{1}_m] = \sum_{i=1}^{m-1} \frac{f_i^{(k_i-1)}(r_i, \mathbf{k} + \mathbf{1}_m)}{(k_i - 1)!} + f_m(r, \mathbf{k} + \mathbf{1}_m). \tag{15}$$

For $i \ne m$, we have

$$f_i(x, \mathbf{k} + \mathbf{1}_m) = \frac{(x-r)^{n+1}}{(x-r)\prod_{\substack{j=1 \\ j \ne i,m}}^{K+1} (x - r_j)^{k_j}} = \frac{(x-r)^n}{\prod_{\substack{j=1 \\ j \ne i}}^{K+1} (x - r_j)^{k_j}} = f_i(x, \mathbf{k})$$

(all derivatives are also identical). Furthermore, $f_m^{(k_m-1)}(x, \mathbf{k}) = 0$ since $k_m = 0$, and

$$f_m(r, \mathbf{k} + \mathbf{1}_m) = \left. \frac{(x-r)^{n+1}}{\prod_{\substack{j=1 \\ j \ne m}}^{K+1} (x - r_j)^{k_j}} \right|_{x=r} = 0.$$

Substituting these results into (15), we see that (11) holds.

If r is not one of the rewards, we consider a new system with $K+2$ rewards, namely, the original $K+1$ rewards plus the additional reward r. Using the notation $*$ for quantities associated to the new system, we see that $m^* = m+1$, $r^*_{m^*} = r$, and $r^*_i = r_i$ for $i = 1, \ldots, m$. Let $\mathbf{k}^*$ be the $K+2$ vector such that $k^*_i = k_i$ for $i = 1, \ldots, m$ and $k^*_i = 0$ otherwise (also recall that $k_i = 0$ for $i > m$). Then the cumulative reward in the new system given n and $\mathbf{k}^*$ is clearly the same as the cumulative reward in the original system given n and $\mathbf{k}$, and so $P[\mathrm{ACR}(t) > r|n, \mathbf{k}] = P[\mathrm{ACR}^*(t) > r|n, \mathbf{k}^*]$. Applying the previous result to the new system, we have

$$P[\mathrm{ACR}^*(t) > r|n, \mathbf{k}^*] = \sum_{i=1}^{m^*} \frac{f_i^{*(k_i^*-1)}(r_i^*, \mathbf{k}^*)}{(k_i^*-1)!}.$$

Now $k^*_{m^*} = 0$, so the term corresponding to m^* in the above sum is 0. Furthermore, $r^*_i = r_i$ and $k^*_i = k_i$ for $i = 1, \ldots, m$, and $k_i = 0$, $k^*_i = 0$ otherwise. Thus, for $i = 1, \ldots, m$, we have

$$f_i^*(x, \mathbf{k}^*) = \frac{(x-r)^n}{\prod_{\substack{j \neq i \\ k_j^* > 0}} \left(x - r_j^*\right)^{k_j^*}} = \frac{(x-r)^n}{\prod_{\substack{j \neq i \\ k_j > 0}} (x - r_j)^{k_j}} = f_i(x, \mathbf{k}).$$

Therefore, (11) also holds in this case. □

The functions $f_i^{(l)}(x, \mathbf{k})$ can be evaluated from the following recursion.

Lemma 2 *Let j be an index such that $k_j \geq 1$. For $\|\mathbf{k}\| = n+1 > 1$ $(n \geq 1)$*

$$\frac{f_i^{(l)}(x,\mathbf{k})}{l!} = \begin{cases} \dfrac{1}{x-r_j}\left(\dfrac{f_i^{(l-1)}(x,\mathbf{k}-\mathbf{1}_j)}{(l-1)!} - \dfrac{f_i^{(l-1)}(x,\mathbf{k})}{(l-1)!}\right) + \left(\dfrac{x-r}{x-r_j}\right)\dfrac{f_i^{(l)}(x,\mathbf{k}-\mathbf{1}_j)}{l!} & i \neq j \\[2ex] \dfrac{f_i^{(l-1)}(x,\mathbf{k}-\mathbf{1}_j)}{(l-1)!} + (x-r)\dfrac{f_i^{(l)}(x,\mathbf{k}-\mathbf{1}_j)}{l!} & i = j \end{cases} \tag{16}$$

For $\|\mathbf{k}\| = n+1 = 1$ $(n = 0,\ k_j = 1)$ and $l > 0$

$$\frac{f_i^{(l)}(x,\mathbf{k})}{l!} = \begin{cases} -\left(\dfrac{1}{x-r_j}\right)\dfrac{f_i^{(l-1)}(x,\mathbf{k})}{(l-1)!} & i \neq j \\[2ex] 0 & i = j \end{cases} \tag{17}$$

The initial conditions are (for $n = 0$, $k_j = 1$, $l = 0$)

$$f_i^{(0)}(x,\mathbf{k}) = \begin{cases} \dfrac{1}{x-r_j} & i \neq j \\[2ex] 1 & i = j \end{cases} \tag{18}$$

Proof: First consider the case $n \geq 1$. We prove (16) by induction on l. We have from the definition of $f_i(x, \mathbf{k})$ in (12) that

$$f_i^{(0)}(x, \mathbf{k}) = \begin{cases} \left(\dfrac{x - r}{x - r_j}\right) f_i^{(0)}(x, \mathbf{k} - \mathbf{1}_j) & i \neq j \\ \\ (x - r) f_i^{(0)}(x, \mathbf{k} - \mathbf{1}_j) & i = j \end{cases} \tag{19}$$

which is (16) for $l = 0$.

Now we assume that (16) is valid for $l - 1$ and prove it for l. Rewriting (16) for the case $i \neq j$ we have

$$(x - r_j)\frac{f_i^{(l-1)}(x, \mathbf{k})}{(l-1)!} + \frac{f_i^{(l-2)}(x, \mathbf{k})}{(l-2)!} = (x - r)\frac{f_i^{(l-1)}(x, \mathbf{k} - \mathbf{1}_j)}{(l-1)!} + \frac{f_i^{(l-2)}(x, \mathbf{k} - \mathbf{1}_j)}{(l-2)!}. \tag{20}$$

Differentiating both sides of (20) gives

$$\begin{aligned} &\frac{f_i^{(l-1)}(x, \mathbf{k})}{(l-1)!} + (x - r_j)\frac{f_i^{(l)}(x, \mathbf{k})}{(l-1)!} + \frac{f_i^{(l-1)}(x, \mathbf{k})}{(l-2)!} = \\ &\qquad \frac{f_i^{(l-1)}(x, \mathbf{k} - \mathbf{1}_j)}{(l-1)!} + (x - r)\frac{f_i^{(l)}(x, \mathbf{k} - \mathbf{1}_j)}{(l-1)!} + \frac{f_i^{(l-1)}(x, \mathbf{k} - \mathbf{1}_j)}{(l-2)!}. \end{aligned} \tag{21}$$

Combining terms yields

$$l\frac{f_i^{(l-1)}(x, \mathbf{k})}{(l-1)!} + (x - r_j)\frac{f_i^{(l)}(x, \mathbf{k})}{(l-1)!} = l\frac{f_i^{(l-1)}(x, \mathbf{k} - \mathbf{1}_j)}{(l-1)!} + (x - r)\frac{f_i^{(l)}(x, \mathbf{k} - \mathbf{1}_j)}{(l-1)!}. \tag{22}$$

Finally, dividing both sides of (22) by $l(x - r_j)$ and rearranging terms, we obtain (16) for $i \neq j$.

Equation (16) for $i = j$ is proved in a similar manner.

Now consider the case $n = 0$. The initial conditions (18) are obtained directly from (12). Then equation (17) is trivially obtained by taking the derivatives of (18). □

Equations (5) and (16) are sufficient to evaluate $M(t)$ recursively from (2). However, using these equations would lead to computational requirements that are combinatorial in the number of different rewards, since both $\Gamma[n, \mathbf{k}]$ and $M(t, n, \mathbf{k})$ would have to be evaluated for all combinations of vectors $\mathbf{k}$ such that $\|\mathbf{k}\| = n + 1$ for $n = 0, \ldots, N$ (recall that N is the truncation point of the infinite series in (2)). In what follows we show that a simple recursion with computational requirements that are only linear in the number of states, linear in a parameter smaller than the number of colors, and cubic in N can be found.

4.2 The Recursion for $P[\mathrm{ACR(t)} > r]$

For $n = 0, 1, \ldots$, let $\mathcal{K}_n = \{\mathbf{k} : \|\mathbf{k}\| = n+1\}$. We first define subsets of $\mathcal{K}_n$ that give a grouping of vectors $\mathbf{k}$ and are the key to obtaining the new recursion for $P[\mathrm{ACR(t)} > r]$.

Definition 1 *For $n \geq 0$, $i = 1, \ldots, m$, let*

$$G_g[i,n] = \{\mathbf{k} \in \mathcal{K}_n : k_i = g\}, \qquad g = 0, \ldots, n+1.$$

It follows from the definition that $G_g[i,n]$, $g = 0, \ldots, n+1$, is a partition of $\mathcal{K}_n$ for each i. As an example, if $n = 1$, $i = 1$ and $K = 2$ (the case of 3 rewards), we have

$$\begin{aligned} G_0[1,1] &= \{\langle 0,2,0\rangle, \langle 0,1,1\rangle, \langle 0,0,2\rangle\} \\ G_1[1,1] &= \{\langle 1,1,0\rangle, \langle 1,0,1\rangle\} \\ G_2[1,1] &= \{\langle 2,0,0\rangle\}. \end{aligned}$$

Definition 2 *For $n \geq 0$, $i = 1, \ldots, m$, $s \in \mathcal{S}$, let*

$$F_{g,s}[i,n] = \{\mathbf{k} \in G_g[i,n] : k_{c(s)} > 0\}, \qquad g = 0, \ldots, n+1.$$

In the above definition recall that $c(s)$ is the index of the reward associated with state s.

From these definitions the following equalities hold.

(a) For $c(s) = i$ and for any g, we have

$$\{\mathbf{k} + \mathbf{1}_{c(s)} : \mathbf{k} \in G_g[i,n]\} = G_{g+1}[i,n+1] = F_{g+1,s}[i,n+1]. \tag{23}$$

(b) For $c(s) \neq i$ and for any g, we have

$$\{\mathbf{k} + \mathbf{1}_{c(s)} : \mathbf{k} \in G_g[i,n]\} = F_{g,s}[i,n+1]. \tag{24}$$

To prove (a) and (b), first consider (23). If $c(s) = i$, adding 1 to the ith entry of all vectors $\mathbf{k} \in G_g[i,n]$ gives all possible vectors in $G_{g+1}[i,n+1]$, where n also increases by 1 when k_i is incremented. This yields the first equality in (23). The second equality holds since, by definition, $k_{c(s)} = k_i = g+1 > 0$ for vectors $\mathbf{k} \in G_{g+1}[i,n+1]$. If $c(s) \neq i$, adding 1 to the $c(s)$ entry of a vector $\mathbf{k} \in G_g[i,n]$ does not alter the value of k_i. Thus the new set of vectors obtained in this manner are associated with the same value of g. However, these vectors satisfy $k_{c(s)} > 0$, and so they belong to $F_{g,s}[i,n+1]$.

The basic idea of the algorithm we develop is to find recursions over sets of vectors $\mathbf{k}$, instead of recursions that consider each vector in isolation. This motivates the following definition.

Definition 3 *For $n \geq 0$, $i = 1, \ldots, m$, $s \in \mathcal{S}$, $g = 0, \ldots, n+1$, $l \geq 0$, let*

$$\Omega_s[g,i,n,l] = \sum_{\mathbf{k} \in G_g[i,n]} \Gamma_s[n,\mathbf{k}] \frac{f_i^{(l)}(r_i,\mathbf{k})}{l!}. \tag{25}$$

Recall that $\Gamma_s[n,\mathbf{k}]$ is the probability of the coloring $\mathbf{k}$ and the last state visited is s given n transitions. Therefore, if $k_{c(s)} = 0$, then $\Gamma_s[n,\mathbf{k}] = 0$. But $k_{c(s)} = 0$ if $\mathbf{k} \in G_g[i,n]$ and $\mathbf{k} \notin F_{g,s}[i,n]$. As a consequence, (25) can be written as

$$\Omega_s[g,i,n,l] = \sum_{\mathbf{k} \in F_{g,s}[i,n]} \Gamma_s[n,\mathbf{k}] \frac{f_i^{(l)}(r_i,\mathbf{k})}{l!}. \tag{26}$$

We now use the recursions for $\Gamma_s[n,\mathbf{k}]$ and $f_i^{(l)}(r_i,\mathbf{k})$ given by equations (5) and (16), respectively, in equation (26) to show that the vector

$$\boldsymbol{\Omega}[g,i,n,l] = \langle \Omega_{s_1}[g,i,n,l], \ldots, \Omega_{s_M}[g,i,n,l] \rangle \tag{27}$$

can be calculated recursively. In the following we define $\Omega_s[g,i,n,l] = 0$ for $n < 0$ or $g < 0$, and we also note that $\Omega_s[g,i,n,l] = 0$ for $l < 0$ from the definition of $f_i^{(l)}(x,\mathbf{k})$.

Lemma 3 *Let $s \in \mathcal{S}$, $i = 1, \ldots, m$, $g = 0, \ldots, n+1$. For $n \geq 1$, and for $n = 0$ and $l > 0$*

$$\Omega_s[g,i,n,l] = \begin{cases} \left(\{\boldsymbol{\Omega}[g,i,n-1,l-1] + \omega_i \boldsymbol{\Omega}[g,i,n-1,l]\}\, \mathbf{P}[:s] - \Omega_s[g,i,n,l-1]\right)/\omega_{i,c(s)} & c(s) \neq i \\ \{\boldsymbol{\Omega}[g-1,i,n-1,l-1] + \omega_i \boldsymbol{\Omega}[g-1,i,n-1,l]\}\, \mathbf{P}[:s] & c(s) = i \end{cases} \tag{28}$$

where $\omega_{i,j} = r_i - r_j$ for $i \neq j$, $\omega_i = r_i - r$, and $\mathbf{P}[:s]$ is the sth column of $\mathbf{P}$.
The initial conditions are (for $n = 0$, $l = 0$)

$$\Omega_s[g,i,0,0] = \begin{cases} \pi_s(0)/\omega_{i,c(s)} & c(s) \neq i,\ g = 0 \\ 0 & c(s) \neq i,\ g = 1 \\ 0 & c(s) = i,\ g = 0 \\ \pi_s(0) & c(s) = i,\ g = 1 \end{cases} \tag{29}$$

where $\pi_s(0)$ is the initial probability of state s.

Proof: First assume that $n \geq 1$. We have to consider the two cases $c(s) \neq i$ and $c(s) = i$. For $c(s) \neq i$, using (5) and (16) (for $i \neq j$), we have

$$\begin{aligned}
\Omega_s[g,i,n,l] &= \sum_{\mathbf{k}\in F_{g,s}[i,n]} \sum_{s'\in\mathcal{S}} \Gamma_{s'}[n-1,\mathbf{k}-\mathbf{1}_{c(s)}]p_{s's}\left(\frac{1}{\omega_{i,c(s)}}\right)\frac{f_i^{(l-1)}(r_i,\mathbf{k}-\mathbf{1}_{c(s)})}{(l-1)!} \\
&+ \sum_{\mathbf{k}\in F_{g,s}[i,n]} \sum_{s'\in\mathcal{S}} \Gamma_{s'}[n-1,\mathbf{k}-\mathbf{1}_{c(s)}]p_{s's}\left(\frac{\omega_i}{\omega_{i,c(s)}}\right)\frac{f_i^{(l)}(r_i,\mathbf{k}-\mathbf{1}_{c(s)})}{l!} \\
&- \sum_{\mathbf{k}\in F_{g,s}[i,n]} \Gamma_s[n,\mathbf{k}]\left(\frac{1}{\omega_{i,c(s)}}\right)\frac{f_i^{(l-1)}(r_i,\mathbf{k})}{(l-1)!}. \qquad (30)
\end{aligned}$$

Exchange the order of summation in the two first terms of equation (30) and note that, since $c(s) \neq i$, $\{\mathbf{k}-\mathbf{1}_{c(s)} : \mathbf{k} \in F_{g,s}[i,n]\} = G_g[i,n-1]$ from (24). Therefore, we can rewrite equation (30) as

$$\begin{aligned}
\Omega_s[g,i,n,l] &= \left(\frac{1}{\omega_{i,c(s)}}\right)\sum_{s'\in\mathcal{S}} p_{s's} \sum_{\mathbf{k}\in G_g[i,n-1]} \Gamma_{s'}[n-1,\mathbf{k}]\frac{f_i^{(l-1)}(r_i,\mathbf{k})}{(l-1)!} \\
&+ \left(\frac{\omega_i}{\omega_{i,c(s)}}\right)\sum_{s'\in\mathcal{S}} p_{s's} \sum_{\mathbf{k}\in G_g[i,n-1]} \Gamma_{s'}[n-1,\mathbf{k}]\frac{f_i^{(l)}(r_i,\mathbf{k})}{l!} \\
&- \left(\frac{1}{\omega_{i,c(s)}}\right)\sum_{\mathbf{k}\in F_{g,s}[i,n]} \Gamma_s[n,\mathbf{k}]\frac{f_i^{(l-1)}(r_i,\mathbf{k})}{(l-1)!}. \qquad (31)
\end{aligned}$$

Equation (28) for $c(s) \neq i$ is found by applying (25) and (26) to the three terms of equation (31) and rewriting the result in terms of vector operations.

For $c(s) = i$ the development is similar. Using (5) and (16) (for $i = j$) yields

$$\begin{aligned}
\Omega_s[g,i,n,l] &= \sum_{\mathbf{k}\in F_{g,s}[i,n]} \sum_{s'\in\mathcal{S}} \Gamma_{s'}[n-1,\mathbf{k}-\mathbf{1}_{c(s)}]p_{s's}\frac{f_i^{(l-1)}(r_i,\mathbf{k}-\mathbf{1}_{c(s)})}{(l-1)!} \\
&+ \sum_{\mathbf{k}\in F_{g,s}[i,n]} \sum_{s'\in\mathcal{S}} \Gamma_{s'}[n-1,\mathbf{k}-\mathbf{1}_{c(s)}]p_{s's}\omega_i\frac{f_i^{(l)}(r_i,\mathbf{k}-\mathbf{1}_{c(s)})}{l!}. \qquad (32)
\end{aligned}$$

Exchange the order of summation in the two terms of equation (32) and note that, since $c(s) = i$, $\{\mathbf{k}-\mathbf{1}_{c(s)} : \mathbf{k} \in F_{g,s}[i,n]\} = G_{g-1}[i,n-1]$ from (23) to obtain

$$\begin{aligned}
\Omega_s[g,i,n,l] &= \sum_{s'\in\mathcal{S}} p_{s's} \sum_{\mathbf{k}\in G_{g-1}[i,n-1]} \Gamma_{s'}[n-1,\mathbf{k}]\frac{f_i^{(l-1)}(r_i,\mathbf{k})}{(l-1)!} \\
&+ \omega_i \sum_{s'\in\mathcal{S}} p_{s's} \sum_{\mathbf{k}\in G_{g-1}[i,n-1]} \Gamma_{s'}[n-1,\mathbf{k}]\frac{f_i^{(l)}(r_i,\mathbf{k})}{l!}. \qquad (33)
\end{aligned}$$

Finally, apply (25) and write the result in terms of vector operations to obtain (28) for $c(s) = i$.

Now consider the case $n = 0$. Since n is the number of transitions of the uniformized process during the observation period, $n = 0$ indicates that no transitions occurred and the process remained in its initial state. Note that $\mathcal{K}_0 = \{\mathbf{1}_1, \ldots, \mathbf{1}_{K+1}\}$, $G_1[i, 0] = \{\mathbf{1}_i\}$ and $G_0[i, 0] = \{\mathbf{1}_j : j \neq i\}$. From (6) and (25) we obtain

$$\Omega_s[g, i, 0, l] = \begin{cases} \pi_s(0)\dfrac{f_i^{(l)}(r_i, \mathbf{k})}{l!} & \text{for } c(s) \neq i,\ g = 0 \text{ and for } c(s) = i,\ g = 1 \\ \\ 0 & \text{for } c(s) = i,\ g = 0 \text{ and for } c(s) \neq i,\ g = 1 \end{cases} \tag{34}$$

Since $\Omega_s[g, i, n, l] = 0$ for $n < 0$ by definition, then equation (28) (for $n = 0$ and $l > 0$) follows from (17) and (34). The initial conditions (29) follow immediately from (18) and (34). □

It should be observed that $\Omega_s[g, i, n, l] = 0$ for certain values of s, g, i, n and l. For example, it was already shown that $\Omega_s[g, c(s), 0, l] = 0$ for $l > 0$ and arbitrary g (see (28)). Now assume that $c(s) = i$ and $g = 0$ with n and l arbitrary. In this case $k_i \geq 1$, and so $G_0[i, n] = \emptyset$ from Definition 1. As a consequence, $\Omega_s[0, c(s), n, l] = 0$. Using similar arguments, we also see that $\Omega_s[n + 1, i, n, l] = 0$ for the case $c(s) \neq i$.

From the values of the vector $\boldsymbol{\Omega}[g, i, n, l]$, we can calculate the measure $P[\text{ACR}(t) > r]$. Let $\|\boldsymbol{\Omega}[g, i, n, l]\| = \sum_{s \in \mathcal{S}} \Omega_s[g, i, n, l]$ for all g, i, n, l.

Theorem 1 *The distribution of the total reward accumulated during $(0, t)$ averaged over t is given by*

$$P[\text{ACR}(t) > r] = \sum_{n=0}^{\infty} e^{-\Lambda t}\frac{(\Lambda t)^n}{n!} \sum_{i=1}^{m} \sum_{g=1}^{n+1} \|\boldsymbol{\Omega}[g, i, n, g - 1]\|. \tag{35}$$

Proof: With $M(t) = P[\text{ACR}(t) > r]$, from (2) we have

$$P[\text{ACR}(t) > r] = \sum_{n=0}^{\infty} e^{-\Lambda t}\frac{(\Lambda t)^n}{n!} \sum_{\mathbf{k} \in \mathcal{K}_n} \Gamma[n, \mathbf{k}] P[\text{ACR}(t) > r | n, \mathbf{k}]. \tag{36}$$

Now we evaluate the sum over the set $\mathcal{K}_n$ on the right hand side of (36). Substituting (7) and (11) into (36), the sum can be rewritten as

$$\begin{aligned} \sum_{\mathbf{k} \in \mathcal{K}_n} \Gamma[n, \mathbf{k}] P[\text{ACR}(t) > r | n, \mathbf{k}] &= \sum_{\mathbf{k} \in \mathcal{K}_n} \sum_{s \in \mathcal{S}} \Gamma_s[n, \mathbf{k}] \sum_{i=1}^{m} \frac{f_i^{(k_i - 1)}(r_i, \mathbf{k})}{(k_i - 1)!} \\ &= \sum_{i=1}^{m} \sum_{s \in \mathcal{S}} \sum_{\mathbf{k} \in \mathcal{K}_n} \Gamma_s[n, \mathbf{k}] \frac{f_i^{(k_i - 1)}(r_i, \mathbf{k})}{(k_i - 1)!}. \end{aligned} \tag{37}$$

The exchange of summation in the last equality of (37) is valid, since, by definition, the value of m depends only on the values of the rewards r_i, $i = 1, \ldots, K+1$, and is independent of the set $\mathcal{K}_n$ and the state space $\mathcal{S}$.

From Definition 1, for any $i = 1, \ldots, m$ and $n \geq 0$, the sets $G_g[i,n]$, $g = 0, \ldots, n+1$, form a partition of $\mathcal{K}_n$. Therefore,

$$\begin{aligned}\sum_{i=1}^{m}\sum_{s\in\mathcal{S}}\sum_{\mathbf{k}\in\mathcal{K}_n}\Gamma_s[n,\mathbf{k}]\frac{f_i^{(k_i-1)}(r_i,\mathbf{k})}{(k_i-1)!} &= \sum_{i=1}^{m}\sum_{s\in\mathcal{S}}\sum_{\mathbf{k}\in\bigcup_{g=0}^{n+1}G_g[i,n]}\Gamma_s[n,\mathbf{k}]\frac{f_i^{(k_i-1)}(r_i,\mathbf{k})}{(k_i-1)!} \\ &= \sum_{i=1}^{m}\sum_{s\in\mathcal{S}}\sum_{g=0}^{n+1}\sum_{\mathbf{k}\in G_g[i,n]}\Gamma_s[n,\mathbf{k}]\frac{f_i^{(k_i-1)}(r_i,\mathbf{k})}{(k_i-1)!}. \end{aligned} \tag{38}$$

Since $k_i = 0$ for $\mathbf{k} \in G_0[i,n]$ and $f_i^{(l)}(x,\mathbf{k}) = 0$ for $l < 0$, the terms corresponding to the index $g = 0$ vanish. Hence, recognizing Ω_s from equation (25), we have

$$\sum_{\mathbf{k}\in\mathcal{K}_n}\Gamma[n,\mathbf{k}]P[\mathrm{ACR}(t) > r|n,\mathbf{k}] = \sum_{i=1}^{m}\sum_{s\in\mathcal{S}}\sum_{g=1}^{n+1}\Omega_s[g,i,n,g-1]. \tag{39}$$

Exchanging the order of summation and substituting the result into (36) gives (35). □

Equation (35) can be simplified by recalling that, when $g = n+1$, the only $\Omega_s[g,i,n,l]$ that are nonzero are those for which $c(s) = i$, so that

$$P[\mathrm{ACR}(t) > r] = \sum_{n=0}^{\infty} e^{-\Lambda t}\frac{(\Lambda t)^n}{n!}\sum_{i=1}^{m}\left\{\sum_{g=1}^{n}\|\boldsymbol{\Omega}^*[g,i,n,g-1]\| + \sum_{s:c(s)=i}\Omega_s^*[n+1,i,n,n]\right\}. \tag{40}$$

In many situations evaluating the tail of the distribution of ACR(t) is of interest. That is, we wish to calculate the probability that the accumulated reward averaged over the observation period is greater than r, where r is close to the largest reward r_1. In fact, if $r_1 > r > r_2$, then $m = 1$. In this case, from Definition 3 and the definition of $f_i^{(l)}(x,\mathbf{k})$, it is easy to see that $\|\boldsymbol{\Omega}[g,i,n,g-1]\|$ is the probability that ACR$(t) > r$ when the model spends g intervals in states with the largest reward given n transitions.

4.3 Computational Requirements

We now discuss the computational complexity of evaluating (35) in order to calculate the distribution of cumulative reward over a finite interval $(0,t)$ averaged over t. From (35) we observe that the main computational effort is spent in the evaluation of $\Omega_s[g,i,n,g-1]$, for

$i = 1, \ldots, m$, $g = 1, \ldots, n+1$, $n = 0, \ldots, N$, where we recall that N is the truncation point of the infinite series in (35) in order to obtain the desired error tolerance.

From (28) we see that $\Omega_s[g, i, n, l]$ can be evaluated using a simple recursion. For a given value of i, the recursion should be carried out from $n = 0$ to N and, for each n, from $g = 0$ to $n + 1$, starting from $\Omega_s[g, i, n, 0]$ (for any s), i.e., for a given i, n and g, the first elements to be evaluated are those for which $l = 0$. Each value of $\Omega_s[g, i, n, l]$ requires a simple vector by vector multiplication, where one of the vectors is a column of the matrix $\mathbf{P}$. Since $\mathbf{P}$ is usually sparse, this operation requires roughly d multiplications, where d is the average number of nonzero entries in a column of $\mathbf{P}$. Thus, to obtain the vector $\boldsymbol{\Omega}[g, i, n, l]$, we need $O(dM)$ multiplications, where recall that M is the dimension of the state space.

Considering all necessary values of $\boldsymbol{\Omega}$ for a given i, clearly we have a total of $O(dMN^3)$ multiplications. Finally, considering $i = 1, \ldots, m$, we have $O(mdMN^3)$ multiplications, where $m < K + 1$. Therefore, the combinatorial complexity in the number of rewards of the algorithm in [4] has been reduced to a linear complexity. We also note the low polynomial order of the other model parameters, namely N^3, M and d. Furthermore, as mentioned above, often our interest is in the tail of the distribution of ACR(t). That is, for many performability models, we are interested in evaluating the probability that the system has a total reward that is close to the maximum possible reward obtained, for instance, the reward received when all components are fully operational. In this case, from its definition, m is close to the index of the largest reward, so m is equal to or near 1.

We now consider the storage requirements. We note that the computation of $\boldsymbol{\Omega}$ can be done independently for any value of i. Figure 4 illustrates the recursion for $\boldsymbol{\Omega}$. In this figure, each dot represents a vector $\boldsymbol{\Omega}[g, i, n, l]$, and the arrows indicate the values needed to calculate a vector. It is easy to see that, for a given n, only the values of $\boldsymbol{\Omega}$ for $n - 1$ need to be stored. Therefore, a total of N^2 vectors of dimension M are necessary, *independent* of the number of rewards of the model.

4.4 Implementation Issues

In section 4.2 we presented the details of an algorithm to calculate the distribution of cumulative reward averaged over the observation period. In this section we discuss some useful implementation details.

Although the rewards associated with the states can be equal to any real number, it is convenient to scale the rewards such that they are all nonnegative and the smallest reward is equal to 0. This can be done by replacing r_i (and r) by $r_i - r_{K+1}$ (and $r - r_{K+1}$).

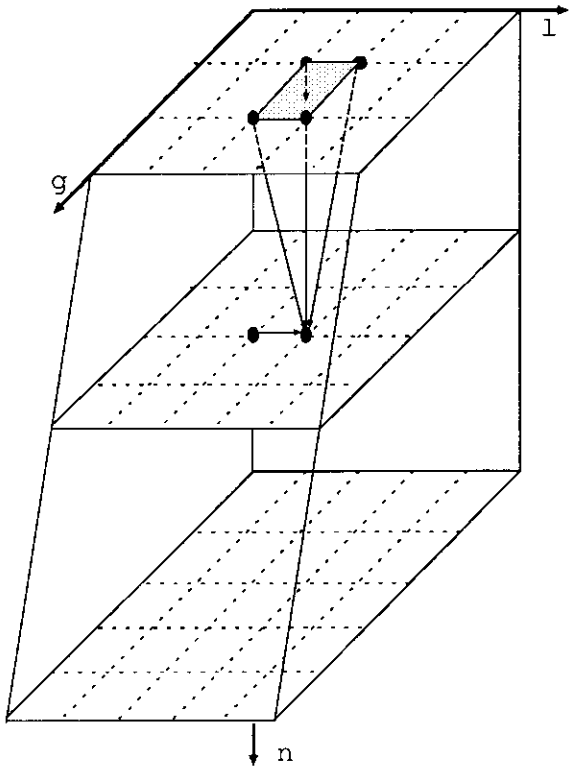


Figure 4: Recursion for Ω.

From Lemma 3 it is clear that $\omega_{i,j}$ can have arbitrary small values (and so $1/\omega_{i,j}$ in (28) is large) if $\omega_1 = r_1 - r$ is small (say, $\omega_1 \leq 1$). Conversely, ω_1 can be large if the values of $\omega_{i,j}$ are large. This may cause overflow problems for large values of N. In order to alleviate this problem, our objective is to find a scaling such that the multipliers in (28) are always less than or equal to 1. To this end, we scale the values in recursion (28) as follows. We first scale the rewards by dividing all values r_i and r by ω_1, and then let $\omega_i^* = \omega_i/\omega_1$ for $i = 1, \ldots, m$. Note that ω_1^*, the largest value of the ω_i^*, is equal to 1. Now we make the following definition.

Definition 4 *Let*

$$\Omega_s^*[g,i,n,l] = \omega^{[(n+1)-(g-l)]}\Omega_s[g,i,n,l] \tag{41}$$

and

$$\omega_{i,c(s)}^* = \frac{\omega}{\omega_{i,c(s)}}, \tag{42}$$

where $\omega = \min_{i,j}\{\omega_{i,j}\}$, $1 \leq i \leq m$, $1 \leq j \leq K+1$, $i \neq j$.

Note that $\omega_i^* \leq 1$ and $\omega_{i,c(s)}^* \leq 1$, $i = 1, \ldots, m$, after rescaling. It can easily be shown that the recursion in (28) can be rewritten as (see [6] for details)

$$\Omega_s^*[g,i,n,l] =$$
$$\begin{cases} \omega_{i,c(s)}^* \left(\{\omega \boldsymbol{\Omega}^*[g,i,n-1,l-1] + \omega_i^* \boldsymbol{\Omega}^*[g,i,n-1,l]\} \mathbf{P}[:s] - \Omega_s^*[g,i,n,l-1] \right) & i \neq c(s) \\ \{\omega \boldsymbol{\Omega}^*[g-1,i,n-1,l-1] + \omega_i^* \boldsymbol{\Omega}^*[g-1,i,n-1,l]\} \mathbf{P}[:s] & i = c(s) \end{cases} \quad (43)$$

The initial conditions are similar to those in equation (29), except that $1/\omega_{i,c(s)}$ is replaced by $\omega_{i,c(s)}^*$. From the values of $\boldsymbol{\Omega}^*$, we obtain $P[\text{ACR}(t) > r]$ as follows. Let α be the smallest positive integer such that $\omega^{-N/\alpha}$ is smaller than the overflow value. For $n = 0, \ldots, N$, let

$$\Theta(n) = \sum_{i=1}^{m} \left\{ \sum_{g=1}^{n} \|\boldsymbol{\Omega}^*[g,i,n,g-1]\| + \sum_{s:c(s)=i} \Omega_s^*[n+1,i,n,n] \right\}.$$

Then $P[\text{ACR}(t) > r|n] = \omega^{-n}\Theta(n)$ by (40) and (41). Now define the function $\varphi_n(\kappa)$ recursively as $\varphi_n(\kappa) = \omega^{-n/\alpha}\varphi_n(\kappa - 1)$ where $\varphi_n(0) = \Theta(n)$. Clearly, $P[\text{ACR}(t) > r|n] = \varphi_n(\alpha)$, and so $\varphi_n(\alpha) \leq 1$. Note that $\omega^{-n/\alpha} \geq 1$ and is less than the overflow value, and $\varphi_n(\kappa)$ is a nondecreasing function of κ which is upper bounded by 1. Thus all the quantities involved in the operations always remain below the overflow value.

Clearly, the value of ω can be made arbitrarily small (and so ω^{-n} can be made arbitrarily large) if the values of two distinct rewards that are greater than r approach each other. However, in this case, we can merge the rewards into a single value and obtain bounds for the final solution. Intuitively, if the reward values hardly differ, the bounds should be tight. In other words, suppose that the normalized values r_p and r_q (for $p < q < m$) differ by, say, a few percent. If we set both rewards to r_p we obtain an upper bound on the final solution. On the other hand, setting both rewards to r_q gives a lower bound. Details concerning these and other implementation issues are presented in [6].

5 Example

In this section an example of the application of the algorithm to a simple performability model of a repairable computer system is presented. The system consists of two processor clusters, each containing two processors, connected to three memory units. The processors operate independently of each other and generate requests for accessing data stored in the memory modules. At most one outstanding request from each processor is allowed at any time, i.e., a processor has to wait for a request to be satisfied before making a new memory access. The processors are assumed to be fast enough so that they generate a new memory request as soon as the previous one is satisfied. Furthermore, the probability that a processor generates a request for data stored in a particular memory module is the same for all modules, and the time to complete a memory access is exponential.

Each component fails independently of the other components in the system, and no failures are possible once the system is down. However, when a processor in a cluster fails, it affects the operation of the other processor in the same cluster, and the entire cluster becomes inoperative. The (exponential) rate at which processor and memory failures occur is 1 per 120 hours and 1 per 240 hours, respectively. Repairs are done to the failed components one at a time with (exponential) rate $\mu = 0.125$ components per hour, and priority is given to the repair of a processor. The system is considered up when at least one processor cluster and one memory unit is operational. Our goal is to calculate the total number of memory requests completed during an interval $(0, t)$ averaged over t.

The model that captures the changes in the structure of the system over time has 11 states, and 6 of the states represent an operational system, each with a different number of processors and memories working. For each configuration of processors and memories, a performance model is built to obtain the reward rates, which are the number of requests completed per unit time. The performance model is a simple single chain closed queueing network, with the number of queues equal to the number of working memory modules and the number of customers equal to the number of operational processors. The 6 distinct rewards $r_1 = 2.00$, $r_2 = 1.60$, $r_3 = 1.50$, $r_4 = 1.33$, $r_5 = 1.00$, $r_6 = 0.00$, are obtained (these values are scaled with respect to the average memory access time).

Figure 5 shows $P[\mathrm{ACR}(t) > \alpha]$, the probability that the total number of memory requests processed in $(0, t)$ averaged over time is at least α, where α is given as a percentage of the throughput achieved when the system is fully operational. In that figure the results are plotted for two values of t (120 hours and 240 hours) when the repair rate is μ and 2μ. Note that the probability of achieving more than 94% of the maximum capacity of the system does not vary much for the two values of t plotted, but it increases substantially when the repair rate doubles (the values increase from 0.659 to 0.916 for $t = 120$ and from 0.621 to 0.949 for $t = 240$, respectively). However, the probability of achieving more than 98% of the maximum system capacity is very low, and it remains below 0.5 even when the repair rate is doubled.

For comparison, the cumulative operational time distribution averaged over time is also plotted for $t = 240$ for the two values of μ (these curves are the dotted lines in the figure). In this case, α is the percentage of the total observation period during which the system remains operational. This distribution is calculated by assigning a reward of 1 to the operational states in the model and a reward of 0 otherwise. From the figure note that, although the probability that the system is operational for more than 98% of the time is high (above 0.95 for the highest repair rate), the probability that it achieves more than 98% of the maximum throughput during the observation period is very low, and it is even below 0.4 when $t = 240$.

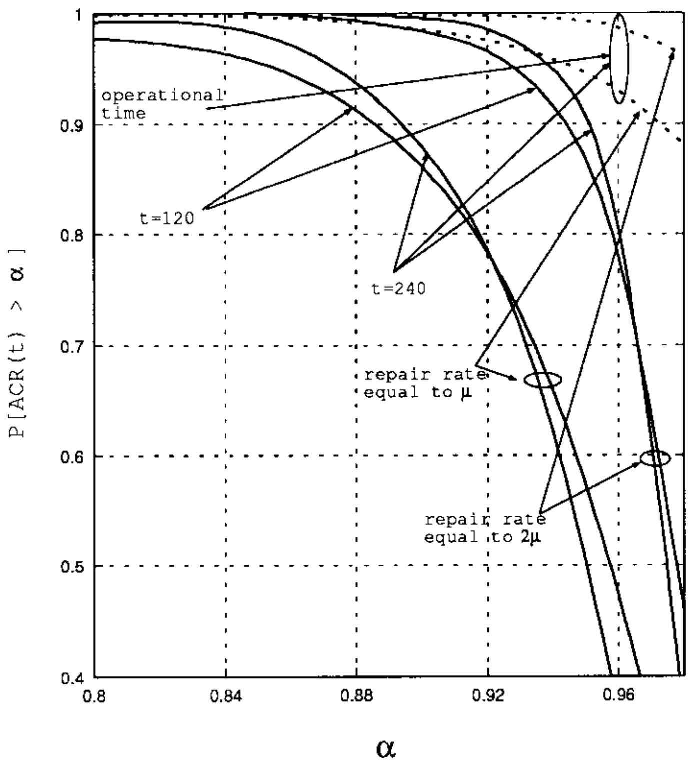


Figure 5: Distributions of cumulative reward and cumulative operational time.

However, the probability of achieving more than 90% of the total throughput is high. This illustrates the difference between the distribution of cumulative operational time and the distribution of cumulative reward.

Finally, note that the value of m in equation (35) is 1 for most of the points that are plotted and at most 2, since our focus is on the tail of the distribution. This corroborates the previous observation in Section 4.

6 Conclusions

The main contribution of this paper is to develop a new polynomial algorithm for calculating the distribution of cumulative reward over a finite observation period using only probabilistic arguments. The development is based on the uniformization technique and the general methodology presented in [4] to calculate performability measures. The new algorithm represents a drastic computational improvement over the previous algorithm reported in [4]. In addition to the probabilistic foundations of the algorithm, the recursions are very simple, have low computational cost and are easy to implement. In fact, the algorithm has been included as part of a set of tools based on uniformization to evaluate performability measures.

References

[1] M.D. Beaudry. Performance-related reliability measures for computing systems. *IEEE Trans. on Computers*, C-27(6):540–547, 1978.

[2] E. Çinlar. *Introduction to Stochastic Processes.* Prentice-Hall, 1975.

[3] H.A. David. *Order Statistics, 2nd Ed.* John Wiley & Sons, 1981.

[4] E. de Souza e Silva and H.R. Gail. Calculating availability and performability measures of repairable computer systems using randomization. *Journal of the ACM*, 36(1):171–193, 1989.

[5] E. de Souza e Silva and H.R. Gail. Performability analysis of computer systems: from model specification to solution. *Performance Evaluation*, 14:157–196, 1992.

[6] E. de Souza e Silva, H.R. Gail, and R. Vallejos Campos. An algorithm to calculate transient distributions of cumulative reward. Technical report, IBM Research Report, Yorktown Heights, N. Y., 1993.

[7] L. Donatiello and V. Grassi. On evaluating the cumulative performance distribution of fault-tolerant computer systems. *IEEE Trans. on Computers*, 40(11):1301–1307, 1991.

[8] L. Donatiello and B.R. Iyer. Analysis of a composite performance reliability measure for fault-tolerant systems. *Journal of the ACM*, 34(1):179–199, 1987.

[9] D.G. Furchtgott and J.F. Meyer. Closed-form solutions of performability. *IEEE Trans. on Computers*, C-33(6):550–554, 1984.

[10] A. Goyal and A.N. Tantawi. Evaluation of performability for degradable computer systems. *IEEE Trans. on Computers*, C-36(6):738–744, 1987.

[11] W.K. Grassmann. Numerical solutions for Markovian event systems. In P. Kall *et al.*, editor, *Quantitative Methoden in den Wirtschaftswissenschaften*, pages 73–87. Springer, 1989.

[12] W.K. Grassmann. Finding transient solutions in Markovian event systems through randomization. In W.J. Stewart, editor, *Numerical Solution of Markov Chains*, pages 357–371. Marcel Dekker, Inc., 1991.

[13] D. Gross and D.R. Miller. The randomization technique as a modeling tool and solution procedure for transient Markov processes. *Operations Research*, 32(2):343–361, 1984.

[14] B.R. Iyer, L. Donatiello, and P. Heidelberger. Analysis of performability for stochastic models of fault-tolerant systems. *IEEE Trans. on Computers*, C-35(10):902–907, 1986.

[15] A. Jensen. Markoff chains as an aid in the study of Markoff processes. *Skandinavsk Aktuarietidskrift*, 36:87–91, 1953.

[16] V.G. Kulkarni, V.F. Nicola, R.M. Smith, and K.S. Trivedi. Numerical evaluation of performability and job completion time in repairable fault-tolerant systems. In *Proceedings of FTCS-16*, pages 252–257, 1986.

[17] J.F. Meyer. On evaluating the performability of degradable computing systems. *IEEE Trans. on Computers*, C-29(8):720–731, 1980.

[18] J.F. Meyer. Closed-form solutions of performability. *IEEE Trans. on Computers*, C-31(7):648–657, 1982.

[19] J.F. Meyer. Performability: A retrospective and some pointers to the future. *Performance Evaluation*, 14:139–156, 1992.

[20] K.R. Pattipati, Y. Li, and H.A.P. Blom. A unified framework for the performability evaluation of fault-tolerant computer systems. *IEEE Trans. on Computers*, 42(3):312–326, 1993.

[21] S.M. Ross. *Stochastic Processes.* John Wiley & Sons, 1983.

[22] R.M. Smith, K.S. Trivedi, and A.V. Ramesh. Performability analysis: Measures, an algorithm and a case study. *IEEE Trans. on Computers*, 37(4):406–417, 1988.

[23] K.S. Trivedi, J.K. Muppala, S.P. Woolet, and B.R. Haverkort. Composite performance and dependability analysis. *Performance Evaluation*, 14:197–215, 1992.

[24] H. Weisberg. The distribution of linear combinations of order statistics from the uniform distribution. *Annals Math. Stat.*, 42:704–709, 1971.