An International Journal
computers & mathematics with applications

PERGAMON Computers and Mathematics with Applications 39 (2000) 217–234
www.elsevier.nl/locate/camwa

Performability Analysis for Degradable Computer Systems

H. NABLI*
Faculté des Sciences de Monastir
5000 Monastir, Tunisia

B. SERICOLA
IRISA/INRIA, Campus de Beaulieu
35042 Rennes cedex, France

(Received November 1997; revised and accepted January 1999)

Abstract—Degradable performance of fault-tolerant computer systems has given rise to considerable interest in mathematical models for combined evaluation of performance and reliability. Most of these models are based upon Markov processes. Several methods have been proposed for the computation of the probability distribution of performability upon an interval of time $[0, t]$. In this paper, we present a new algorithm based on the uniformization technique to compute this distribution for block degradable models. The main advantage of this method is its low polynomial computational complexity and its numerical stability, since it only deals with a nonincreasing sequence of positive numbers bounded by 1. This important property allows us to determine new truncation steps which improve the execution time of the algorithm. We apply this method to a degradable computer system. © 2000 Elsevier Science Ltd. All rights reserved.

Keywords—Block degradable models, Degradable systems, Markovian models, Performability.

1. INTRODUCTION

Reliability and availability measures are of considerable interest for the evaluation of dependability of computer systems, and more specifically, of fault-tolerant computer systems. Thanks to redundancies in both software and hardware, the system is able to reconfigure itself when failures occur. Two types of fault-tolerant computer systems are usually identified: degradable (or nonrepairable) systems and repairable systems.

As recognized in a number of recent studies, the evaluation of fault-tolerant computer systems must simultaneously deal with aspects of both performance and reliability. As part of these studies, Meyer [1] introduces a unified measure called performability which combines the two aspects of performance and reliability. Performability is defined as the accumulated reward over a period of time $[0, t]$. Formally, the fault-repair behavior of the system is assumed to be modeled by a homogeneous Markov process. Its state space is divided into disjoint subsets, which represent the different configurations of the system. A performance level (or reward rate) is associated with each of these configurations. This reward rate quantifies the ability of the system to perform

*This author is a member of RSR Laboratory of the Ecole Nationale des Sciences de l'Informatique de Tunis.

correctly in the corresponding configuration. As a consequence, performability corresponds to the accumulated reward over the mission time.

The distribution of this random variable has been studied in previous papers. For cyclic models, Smith *et al.* [2] derive an algorithm to compute the distribution of performability using Laplace transforms and a numerical inversion procedure to obtain the result in the time domain. De Souza e Silva and Gail [3,4] propose a method based on the uniformization technique, however their method exhibits an exponential computational complexity in the number of reward rates. Using the same technique, Donatiello and Grassi [5,6] obtain an algorithm with a polynomial computational complexity. However, this algorithm seems to be numerically unstable since the coefficient computed in their recursion can have positive and negative signs and are unbounded which can lead to severe numerical errors and overflow problems. More recently, De Souza e Silva and Gail [7] also obtain a new algorithm based on the general methodology of [3,4]. The computational cost of their algorithm is linear in a parameter that is smaller than the number of rewards, but their algorithm seems to have the same unstability problem due the use of both positive and negative coefficients. In [8,9], we present a new algorithm also based on the uniformization technique. The main advantage of our method is that it uses only addition and multiplication of nonnegative quantities that are moreover bounded by one. By this way, we avoid numerical problems and improve the stability of the computation. Pattipati *et al.* [10] obtain the distribution of the accumulated reward for nonhomogeneous Markov processes as the solution of a system of linear hyperbolic partial differential equations which is numerically solved by using a discretization approach.

Regarding the acyclic models, Meyer [11] obtains a closed form expression for the distribution of the performability for a degradable computer system with N processors and a buffer with finite capacity. Furchtgott and Meyer [12] define i-resolvable vectors to characterize the trajectories of an acyclic semi-Markovian process corresponding to a certain performance level. By enumerating all the possible trajectories of the system, they derive an integral expression for performability which they solve numerically. However, the complexity of such an algorithm is exponential in the number of states of the process. Beaudry [13] gives a method for the computation of performability in a Markovian process until absorption. Ciardo *et al.* [14] generalize Beaudry's approach to a semi-Markov reward process and remove the restriction requiring only the absorbing states to be associated with a zero reward rate. Iyer *et al.* [15] propose an algorithm to compute recursively the moments of the accumulated reward over the mission time, with a polynomial computational complexity in the number of states. Goyal and Tantawi [16] derive a closed form expression (precisely a finite sum of exponential functions) for the performability of degradable heterogeneous systems. They also give an algorithm with a polynomial complexity $O(dM^3)$ in the number M of states and in the number d of components in the system. A method which follows an approach similar to the one used by Goyal and Tantawi is presented in [17]. The author derives an algorithm to compute the probability distribution of performability with a low polynomial computational complexity in comparison with the algorithm of Goyal and Tantawi.

In this paper, we propose a new algorithm to compute the distribution of the accumulated reward over a finite mission time for block acyclic Markov models. These models are more general than the acyclic one's since the restriction of the Markov process on each block can be cyclic[1]. Our method uses the uniformization technique and it is based on the main result of [8] which gives a performability solution for general Markov models. This approach consists of proving some interesting mathematical results for block acyclic performability models which lead us to use new truncation steps improving so the computational complexity of the algorithm. Also, our method is characterized by its numerical stability once it only deals with a nonincreasing sequence of positive numbers bounded by one.

[1]The infinitesimal generator for acyclic models is upper triangular and it is block upper triangular for block acyclic models.

The remainder of this paper is organized as follows. In the next section, we introduce the mathematical model of the class of degradable systems and we give some interesting results for the distribution of the performability. In Section 3, we present a simple algorithm towards performability evaluation. We also discuss the computational complexity of the proposed technique. A numerical example of degradable multiprocessor system is presented and solved for a given performability measure in Section 4. The main points are summarized in the concluding section.

2. MATHEMATICAL MODEL AND RESULTS

Degradable computer systems operate at various levels of performance: when a component fails, the system reconfigures itself and carries on functioning albeit with degraded performance. Because of changes in its structure, due to failures, the system has different configurations in a finite state space $E = \{1, 2, \dots, M\}$. A reward rate $\rho(i)$ which is independent of the time is associated with each state $i \in E$. This reward rate measures how well the system performs in the corresponding configuration. Since we consider degradable systems, it must be that $\rho(i) \geq \rho(j)$ if a transition is possible from state i to state j. Therefore, we can number the states so that $i \mapsto \rho(i)$ becomes an increasing function. The accumulated reward random variable over a finite period of time $[0, t]$ is of interest.

Let therefore $X = (X_s)_{s \geq 0}$ be a homogeneous Markov process over the state space E. Since two different states may have the same reward rate, we denote by $r_m > r_{m-1} > \cdots > r_0$ the $m+1$ different reward rates ($m < M$), and by B_i the set of states having r_i as reward rate, for $i \in \{0, \dots, m\}$. It is clear that the subsets $B_m, \dots, B_0$ are disjoint and their union gives in the state space E. The process X is entirely determined by its infinitesimal generator $A = (a_{ij})_{i,j \in E}$ and its initial probability distribution $\alpha = (\alpha_i)_{i \in E}$. We assume α to be a row vector of the form $\alpha = (\alpha_{B_m}, 0, \dots, 0)$. This means that the system starts in subset B_m with probability 1. Since the system is degradable and $i \mapsto \rho(i)$ is a wide sense increasing function, we have

$$a_{ij} = 0, \qquad \text{if } i \in B_l,\ j \in B_k, \text{ and } l < k. \tag{1}$$

Note that in the same subset B_l, we can have transitions between two different states i and j even if $i < j$, which means that the process restricted on each block B_l can be cyclic, in this case, the block B_l can be defined as an equivalence class. Also, if we consider each block B_l as a macrostate, the graph of the Markov process X will be indeed acyclic. Such models are called block degradable.

It is well known [18] that if the process X is uniformized with an intensity parameter $\lambda \geq \max_{i \in E}(-a_{ii})$, then $P = I + A/\lambda$ is the transition probability matrix associated to the uniformized chain where I is the $M \times M$ identity matrix. Using the decomposition of E with respect to the partition $\{B_m, \dots, B_0\}$ and equation (1), the matrix P can be written as follows:

$$P = \begin{pmatrix} P_{B_m B_m} & P_{B_m B_{m-1}} & \cdots & P_{B_m B_0} \\ 0_{B_{m-1} B_m} & P_{B_{m-1} B_{m-1}} & \cdots & P_{B_{m-1} B_0} \\ \vdots & \vdots & & \vdots \\ 0_{B_0 B_m} & 0_{B_0 B_{m-1}} & \cdots & P_{B_0 B_0} \end{pmatrix},$$

where the submatrix $P_{B_i B_j}$ (respectively, $0_{B_i B_j}$) contains the transition probabilities from states of B_i to states of B_j (respectively, is a zero matrix). With the above notations, the accumulated reward over the mission time $[0, t]$ is defined by

$$Y_t = \int_0^t \rho(X_s)\, ds = \sum_{i=0}^{m} r_i \int_0^t \mathbb{1}_{\{X_s \in B_i\}}\, ds,$$

where $\mathbb{1}_c = 1$ if condition c is true and 0 otherwise.

The random variable Y_t takes its values in the interval $[r_0 t, r_m t]$ and we wish to derive $\mathbb{P}\{Y_t > s\}$. The reward rates r_i are arbitrary real numbers, but we can assume $r_0 = 0$ without loss of generality. This is obtained by replacing r_i by $r_i - r_0$ and s by $s - r_0 t$. Therefore, we take $r_0 = 0$.

In [8], the distribution of performability over $[0, t]$ for general finite Markov processes is derived by a new method which essentially is based upon the following theorem.

THEOREM 2.1. *For all $j = 1, \dots, m$ and $s \in [r_{j-1}t, r_j t[$, we have*

$$\mathbb{P}\{Y_t > s\} = \sum_{n=0}^{\infty} e^{-\lambda t} \frac{(\lambda t)^n}{n!} \sum_{k=0}^{n} \binom{n}{k} s_j^k (1 - s_j)^{n-k} b^{(j)}(n, k),$$

where $s_j = (s - r_{j-1}t)/(r_j - r_{j-1})t$ and $b^{(j)}(n,k) = \alpha_{B_m} b^{(j)}_{B_m}(n,k)$. The values of the column vectors $b^{(j)}_{B_l}(n,k)$ are positive bounded by 1 and, for all $n \geq 0$, and $j = 1, \dots, m$, are given by the following set of recursive expressions:

- *for $j \leq l \leq m$ and $1 \leq k \leq n$,*

$$b^{(j)}_{B_l}(n, 0) = \begin{cases} \mathbf{1}_{B_l}, & \text{if } j = 1, \\ b^{(j-1)}_{B_l}(n, n), & \text{otherwise}, \end{cases}$$

$$b^{(j)}_{B_l}(n,k) = \frac{r_l - r_j}{r_l - r_{j-1}} b^{(j)}_{B_l}(n, k-1) + \frac{r_j - r_{j-1}}{r_l - r_{j-1}} \sum_{i=0}^{m} P_{B_l B_i} b^{(j)}_{B_i}(n-1, k-1);$$

- *for $0 \leq l \leq j - 1$ and $0 \leq k \leq n - 1$,*

$$b^{(j)}_{B_l}(n,k) = \frac{r_{j-1} - r_l}{r_j - r_l} b^{(j)}_{B_l}(n, k+1) + \frac{r_j - r_{j-1}}{r_j - r_l} \sum_{i=0}^{m} P_{B_l B_i} b^{(j)}_{B_i}(n-1, k),$$

$$b^{(j)}_{B_l}(n, n) = \begin{cases} \mathbf{0}_{B_l}, & \text{if } j = m, \\ b^{(j+1)}_{B_l}(n, 0), & \text{otherwise}. \end{cases}$$

$\mathbf{1}_{B_l}$ and $\mathbf{0}_{B_l}$ are column vectors of dimension $|B_l|$ and their values are, respectively, equal 1 and 0.

PROOF. See [9]. ∎

In the case where $m = 1$, the performability distribution mentioned in Theorem 2.1 is the same as the one given in [19] where the interval availability distribution is computed. It follows that the method presented in [8] is the natural extension of the method in [19].

Note that for $l \leq j \leq m$, we have $0 \leq (r_l - r_j)/(r_l - r_{j-1}) = 1 - (r_j - r_{j-1})/(r_l - r_{j-1}) \leq 1$ and for $0 \leq l \leq j-1$, we have $0 \leq (r_{j-1} - r_l)/(r_j - r_l) = 1 - (r_j - r_{j-1})/(r_j - r_l) \leq 1$. On the other hand, the matrix P is stochastic (i.e., $\sum_{i=0}^{m} P_{B_l B_i} \mathbf{1}_{B_i} = \mathbf{1}_{B_l}$) and the initial value of $b^{(1)}_{B_l}(n, 0)$ is equal to $\mathbf{1}_{B_l}$ for $l \geq 1$ and the final value of $b^{(m)}_{B_l}(n,n)$ is equal to $\mathbf{0}_{B_l}$ for $l \leq m - 1$. So, it is easily to obtain by recurrence that $\mathbf{0}_{B_l} \leq b^{(j)}_{B_l}(n,k) \leq \mathbf{1}_{B_l}$. Moreover, for every $j \in \{1, \dots, m\}$ and $s \in [r_{j-1}t, r_j t[$, we have $0 \leq s_j < 1$. These remarks are essential from the computational point of view since the manipulation of nonnegative quantities bounded by 1 allows us to avoid the unstability problems which may appear in the algorithm described in [5,7].

The interpretation, if any, of the $b^{(j)}_i(n,k)$s is still an open problem. For the availability case, where $m = 1$, the authors of [19] prove that the vectors $b^{(1)}_i(n,k)$ is the probability that the uniformized Markov chain Z visits more than k states of B_1 during the first n transitions, given that the initial state is $Z_0 = i$. For $m \geq 2$, it is more difficult to obtain a probabilistic interpretation of the vectors $b^{(j)}_i(n,k)$. For more details, see [20].

When the system is block degradable, new results for the sequence $b^{(j)}_{B_l}(n,k)$ can be obtained. For such models, the recursive expressions of the vectors $b^{(j)}_{B_l}(n,k)$ reduce to the following.

LEMMA 2.2. *For all $n \geq 0$ and $j = 1, \ldots, m$, we have*

- *for $0 \leq l \leq j-1$ and $0 \leq k \leq n$,*

$$b_{B_l}^{(j)}(n,k) = \mathbf{0}_{B_l},$$

- *for $j \leq l \leq m$ and $1 \leq k \leq n$,*

$$b_{B_l}^{(j)}(n,0) = \begin{cases} \mathbf{1}_{B_l}, & \text{if } j = 1, \\ b_{B_l}^{(j-1)}(n,n), & \text{otherwise,} \end{cases}$$

$$b_{B_l}^{(j)}(n,k) = \frac{r_l - r_j}{r_l - r_{j-1}} b_{B_l}^{(j)}(n,k-1) + \frac{r_j - r_{j-1}}{r_l - r_{j-1}} \sum_{i=j}^{l} P_{B_l B_i} b_{B_i}^{(j)}(n-1,k-1).$$

Furthermore, for $j = 1$, the coefficients $b_{B_l}^{(1)}(n,k)$ are independent of index n.

PROOF. See Appendix A. ▮

The previous lemma shows that $b_{B_l}^{(1)}(n,k)$ can be simplified in $b_{B_l}^{(1)}(k)$. The case $j = 1$ means that s belongs to $[0, r_1 t[$. If we apply Theorem 2.1 to this case, we can simplify the distribution of the performability. This is the object of the following corollary.

COROLLARY 2.3. *For all $s \in [0, r_1 t[$, we have*

$$\mathbb{P}\{Y_t > s\} = \sum_{k=0}^{\infty} e^{-\lambda t s_1} \frac{(\lambda t s_1)^k}{k!} b^{(1)}(k),$$

where $b^{(1)}(k) = \alpha_{B_m} b_{B_m}^{(1)}(k)$.

PROOF. See Appendix B. ▮

In practise, the value of s is very close to $r_m t$ since it is generally required that the random variable Y_t should be close to its maximum value $r_m t$ with a probability close to 1. The following corollary gives the distribution of the performability when $s \in [r_{m-1} t, r_m t[$.

COROLLARY 2.4. *For all $s \in [r_{m-1} t, r_m t[$, we have*

$$\mathbb{P}\{Y_t > s\} = \alpha_{B_m} e^{A_{B_m} s_m t} \left[\sum_{n=0}^{\infty} e^{-\lambda t (1 - s_m)} \frac{(\lambda t (1 - s_m))^n}{n!} b_{B_m}^{(m-1)}(n,n) \right].$$

PROOF. See Appendix C. ▮

Our analysis will consist in obtaining new truncation steps by using properties of monotony of the sequence $b_{B_l}^{(j)}(n,k)$. The next theorem states the relation between successive values of the $b_{B_l}^{(j)}(n,k)$. These properties allow us to improve the complexity of our algorithm as shown in the next section.

THEOREM 2.5. *For all $1 \leq j \leq m$, $n \geq 1$, and $j \leq l \leq m$, we have*

- *for $1 \leq k \leq n$,*

$$b_{B_l}^{(j)}(n,k) \leq b_{B_l}^{(j)}(n,k-1),$$

- *for $0 \leq k \leq n-1$,*

$$b_{B_l}^{(j)}(n,k) \leq b_{B_l}^{(j)}(n-1,k).$$

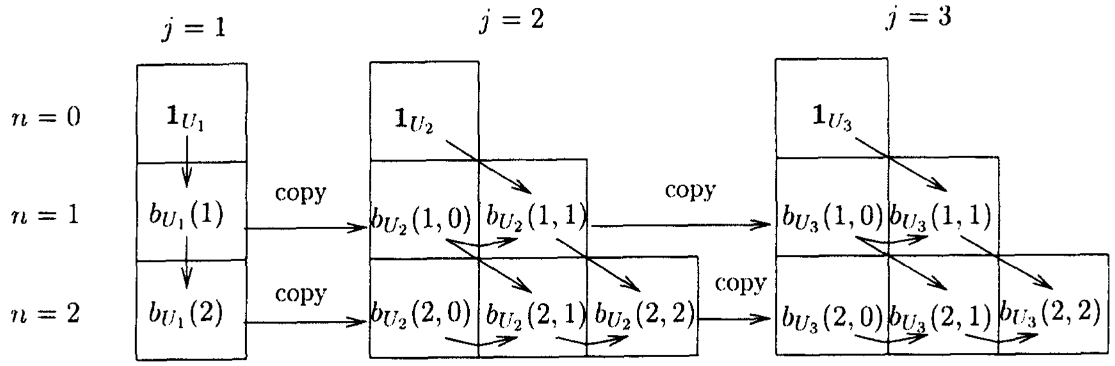

Figure 1. Computation of coefficients $b_{U_j}(n,k)$ for $m = 3$ and $n = 0, 1, 2$.

PROOF. See Appendix D. ∎

The first (respectively, the second) inequality of Theorem 2.5 expresses the decreasing of cells of the same line (respectively, column) in each triangle containing $b_{B_l}^{(j)}(.,.)$ elements (see Figure 1). Putting together these results, we obtain the following inequalities:

$$b_{B_l}^{(j)}(n,k) \leq b_{B_l}^{(j)}(n,k-1) \leq b_{B_l}^{(j)}(n-1,k-1). \tag{2}$$

In the next section, we describe the computational aspect of the vectors $b_{B_l}^{(j)}(n,k)$. Subsequently, we give a numerical method to compute the distribution of the performability. With different results obtained for different values of s, we, successively, deal with the following cases $s \in [0, r_1 t[$, $s \in [r_{j_0-1}t, r_{j_0}t[$, $1 < j_0 < m$, and $s \in [r_{m-1}t, r_m t[$. In each case, we also analyze the numerical complexity.

3. COMPUTATIONAL AND ALGORITHMIC ASPECTS

3.1. Computation of $b_{B_l}^{(j)}(n,k)$

Since coefficients $b_{B_l}^{(j)}(n,k)$ are equal to 0_{B_l} for $l < j$, we define for every $j = 1,\dots,m$, the subset U_j of the state space E as $U_j = B_m \cup B_{m-1} \cup \dots \cup B_j$. Note that the sets U_j decrease with respect to j in the sense of inclusion (i.e., $U_j \subset U_{j-1}$). We also define the following column vector of length $|U_j|$:

$$b_{U_j}(n,k) = \begin{pmatrix} b_{B_m}^{(j)}(n,k) \\ \vdots \\ b_{B_j}^{(j)}(n,k) \end{pmatrix}.$$

With this notation, Figure 1 illustrates the sequence of computations (drawn only for $m = 3$ and $n = 0, 1, 2$) needed to evaluate the $b_{B_l}^{(j)}(n,k)$s for $l \geq j$. The computation of each cell (n,k), $k \geq 1$ depends on the two vectors associated with cells $(n, k-1)$ and $(n-1, k-1)$. This dependence is pictured in Figure 1 with arrows. Figure 2 makes clear this dependence and shows how we calculate a coefficient $b_{B_l}^{(j)}(n,k)$. Note that the symbol "copy" means that each diagonal cell of each triangle must be reported in the corresponding cell of the first column in the next triangle (i.e., $b_{B_l}^{(j-1)}(n,n) = b_{B_l}^{(j)}(n,0)$ for all $l = j,\dots,m$). The use of relations in Lemma 2.2 leads us to perform the evaluation of the $b_{U_j}(n,k)$s in a line by line manner as shown in Figure 1.

Property (2) allows us to define a new truncation step for computing the infinite sum. Formally, for a tolerance error ε specified by the user and a given positive number x, we define the integer $N(x) = \min\{n \in \mathbb{N} : \sum_{i=0}^{n} e^{-x}(x^i/i!) > 1 - \varepsilon\}$ ($N(x)$ is the truncation step of a Poisson series [21]). We denote by d the connectivity degree of matrix P which is defined by $d = \max_{i\in E}\{NZ(i)\}$, where $NZ(i)$ is the number of nonzero entries in row i of P.

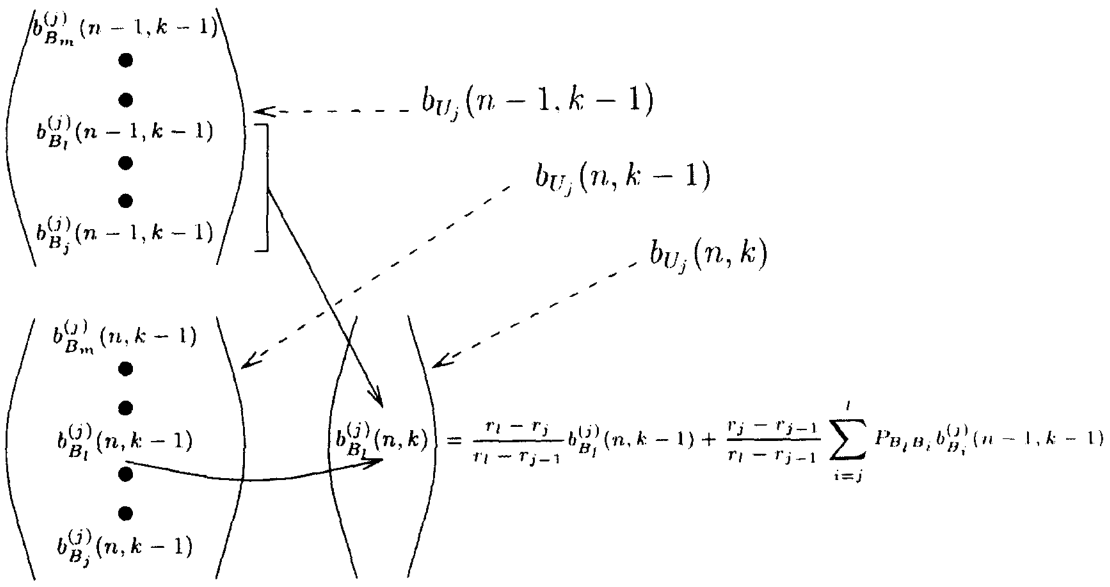

Figure 2. Computation of a coefficient $b^{(j)}_{B_l}(n,k)$.

The main computational effort required for the computation of the distribution of Y_t is due to the computation of the numbers $b_i^{(j)}(n,k)$. In order to evaluate the complexity of our algorithm, we give an approximate count of the number of multiplications involved in the computation of the $b_i^{(j)}(n,k)$s. To this end, we denote by $\mu(H)$ the complexity of an algorithm which requires approximately H multiplications.

3.2. Case $s \in [0, r_1 t[$

According to Corollary 2.3, the distribution of the performability Y_t is given by the following expression:

$$\mathbb{P}\{Y_t > s\} = \sum_{k=0}^{\infty} e^{-\lambda t s_1} \frac{(\lambda t s_1)^k}{k!} b^{(1)}(k).$$

This distribution can also be written as

$$\mathbb{P}\{Y_t > s\} = \sum_{k=0}^{N(\lambda t s_1)} e^{-\lambda t s_1} \frac{(\lambda t s_1)^k}{k!} b^{(1)}(k) + e(N(\lambda t s_1)),$$

where $e(N(\lambda t s_1))$ satisfies

$$\begin{aligned} e(N(\lambda t s_1)) &= \sum_{k=N(\lambda t s_1)+1}^{\infty} e^{-\lambda t s_1} \frac{(\lambda t s_1)^k}{k!} b^{(1)}(k) \\ &\le \sum_{k=N(\lambda t s_1)+1}^{\infty} e^{-\lambda t s_1} \frac{(\lambda t s_1)^k}{k!}, \qquad \text{since } b^{(1)}(k) \le 1, \\ &= 1 - \sum_{k=0}^{N(\lambda t s_1)} \frac{(\lambda t s_1)^k}{k!} \\ &\le \varepsilon. \end{aligned}$$

Moreover, if we take into account the inequality $b_{U_1}(k) \le b_{U_1}(k-1)$ given by Theorem 2.5, it is interesting to use another truncation step n_1 defined by

$$n_1 = \min\left(N(\lambda t s_1), \min\left\{k \in \mathbb{N} \mid b^{(1)}(k+1) \le \varepsilon\right\}\right).$$

By considering this new truncation, we obtain

$$\mathbb{P}\{Y_t > s\} = \sum_{k=0}^{n_1} e^{-\lambda t s_1} \frac{(\lambda t s_1)^k}{k!} b^{(1)}(k) + e(n_1).$$

The error introduced in the evaluation of the distribution of Y_t remains smaller than ε. In fact,

$$\begin{aligned} e(n_1) &= \sum_{k=n_1+1}^{\infty} e^{-\lambda t s_1} \frac{(\lambda t s_1)^k}{k!} b^{(1)}(k) \\ &\le \left[\sum_{k=n_1+1}^{\infty} e^{-\lambda t s_1} \frac{(\lambda t s_1)^k}{k!} \right] \varepsilon, \qquad \text{since } b^{(1)}(k) \le \varepsilon, \quad \text{for } k \ge n_1 + 1, \\ &\le \varepsilon. \end{aligned}$$

The computation of the distribution of Y_t mainly involves computing the coefficients $b^{(1)}(k)$. Moreover, the complexity in computing of a vector $b_{U_1}(k)$, which is one cell pictured in Figure 3, is $\mu(d(M - |B_0|))$. Therefore, as the total number of required vectors $b_{U_1}(k)$ is $n_1 + 1$, the complexing in evaluating $F_{B_m}(s,t)$ is $\mu(d(n_1+1)(M - |B_0|))$. On the other hand, the total computational effort required with the method of [8] is $\mu(dM[C(N - C) + mC^2/2])$, where C and N are, respectively, equal to $N(\lambda t s_1)$ and $N(\lambda t)$. Therefore, the algorithm in this section compares favorably with the method of [8].

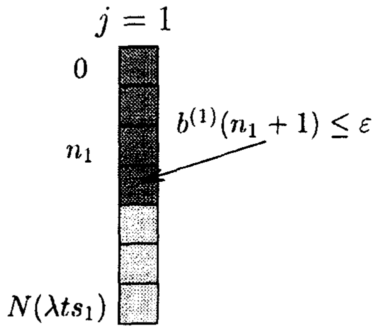

Figure 3. In dark, the computed vectors $b_{U_1}(.)$. In light, cells such that $b^{(1)}(.) \le \varepsilon$.

3.3. Case $s \in [r_{j_0-1}t, r_{j_0}t[$, $2 \le j_0 \le m-1$

According to Theorem 2.1, the distribution of Y_t is given by

$$\mathbb{P}\{Y_t > s\} = \sum_{n=0}^{\infty} e^{-\lambda t} \frac{(\lambda t)^n}{n!} \sum_{k=0}^{n} \binom{n}{k} s_{j_0}^k (1 - s_{j_0})^{n-k} b^{(j_0)}(n,k).$$

To compute the infinite sum, we define a truncation step n_{j_0} as follows:

$$n_{j_0} = \min\left(N(\lambda t), \min\left\{n \in \mathbb{N} \mid b^{(j_0)}(n+1, 0) \le \varepsilon\right\}\right).$$

Since $b_{B_m}^{(j_0)}(n,0) = b_{B_m}^{(j_0-1)}(n,n)$, we have

$$n_{j_0} = \min\left(N(\lambda t), \min\left\{n \in \mathbb{N} \mid b^{(j_0-1)}(n+1, n+1) \le \varepsilon\right\}\right).$$

Using inequalities (2), it is easy to establish that all terms $b^{(j_0)}(n,k)$ are smaller than ε, for $n \ge n_{j_0} + 1$ and $0 \le k \le n$ (see Figure 4). It follows that

$$\mathbb{P}\{Y_t > s\} = \sum_{n=0}^{n_{j_0}} e^{-\lambda t} \frac{(\lambda t)^n}{n!} \sum_{k=0}^{n} \tbinom{n}{k} s_{j_0}^k (1 - s_{j_0})^{n-k} b^{(j_0)}(n,k) + e(n_{j_0}),$$

where $e(n_{j_0})$ satisfies

$$\begin{aligned} e(n_{j_0}) &= \sum_{n=n_{j_0}+1}^{\infty} e^{-\lambda t}\frac{(\lambda t)^n}{n!}\sum_{k=0}^{n}\tbinom{n}{k}s_{j_0}^k(1-s_{j_0})^{n-k}b^{(j_0)}(n,k) \\ &\le \left[\sum_{n=n_{j_0}+1}^{\infty} e^{-\lambda t}\frac{(\lambda t)^n}{n!}\sum_{k=0}^{n}\tbinom{n}{k}s_{j_0}^k(1-s_{j_0})^{n-k}\right]\varepsilon \\ &= \left[\sum_{n=n_{j_0}+1}^{\infty} e^{-\lambda t}\frac{(\lambda t)^n}{n!}\right]\varepsilon \\ &\le \varepsilon. \end{aligned}$$

Note that in the case where all coefficients $b^{(j_0-1)}(n+1,n+1)$ are strictly greater than ε for all $n \le N(\lambda t)$, n_{j_0} is equal to $N(\lambda t)$. In this case, $e(n_{j_0}) = e(N(\lambda t))$ remains smaller than ε.

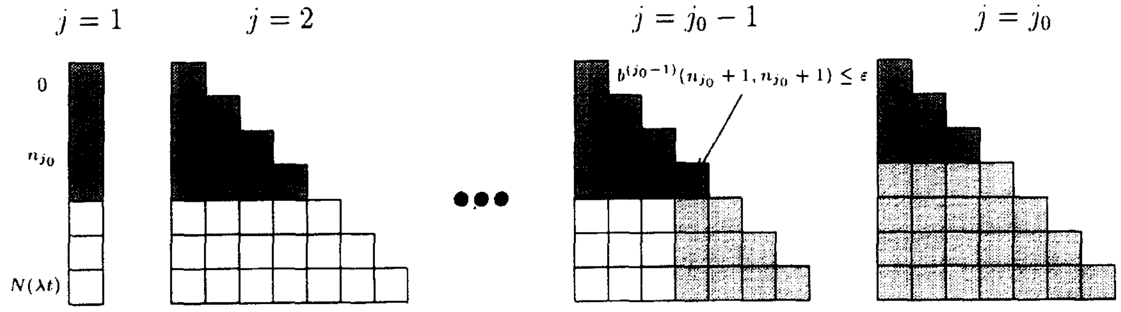

Figure 4. In dark, the computed vectors. In light, cells such that $b^{(j)}(.,.) \le \varepsilon$.

Since the cardinality $|U_j|$ is less than $M - j$, the computation of each vector $b_{U_j}(n,k)$ requires at most $\mu(d(M-j))$ (see equations $b_{U_j}(n,k)$ in Lemma 2.2 and Figure 2). According to Figure 4, the number of cells that have to be computed in the triangle associated with index j is equal to

$$\begin{aligned} &n_{j_0} + 2, && \text{if } j = 1, \\ &\frac{(n_{j_0}+2)(n_{j_0}+3)}{2}, && \text{if } 1 < j < j_0, \\ &\frac{(n_{j_0}+1)(n_{j_0}+2)}{2}, && \text{if } j = j_0. \end{aligned}$$

Since we have $\sum_{j=2}^{j_0}(M-j) = (j_0-1)(M-(j_0+2)/2)$ and can neglect n_{j_0} with respect to $n_{j_0}^2$, we obtain a complexity for the algorithm of

$$\mu\left(d\left(M - \frac{j_0+2}{2}\right)(j_0-1)\frac{n_{j_0}^2}{2}\right).$$

As opposed to the method relative of [8], we observe that the numerical complexity of this method depends on the index j_0, and therefore on the value of s. Another improvement came from the new truncation step n_{j_0} which is less than $N(\lambda t)$.

3.4. Case $s \in [r_{m-1}t, r_m t[$

According to Theorem 2.1, we have

$$\mathbb{P}\{Y_t > s\} = \alpha_{B_m} e^{A_{B_m} s_m t}\left[\sum_{n\ge 0} e^{-\lambda t(1-s_m)}\frac{(\lambda t(1-s_m))^n}{n!} b_{B_m}^{(m-1)}(n,n)\right].$$

Similarly to the previous section, we define the truncation step n_m as follows:

$$n_m = \min\left(N\left(\lambda t\left(1 - s_m\right)\right), \min\left\{n \in \mathbb{N} \mid b_{B_m}^{(m-1)}(n+1, n+1) \leq \varepsilon \mathbf{1}_{B_m}\right\}\right).$$

Using inequalities (2) again, it is easy to establish that all terms $b_{B_m}^{(m-1)}(n,n)$ are smaller than ε when $n \geq n_m + 1$. It follows that

$$\mathbb{P}\{Y_t > s\} = \alpha_{B_m} e^{A_{B_m} s_m t} \left[\sum_{n=0}^{n_m} e^{-\lambda t(1-s_m)} \frac{(\lambda t(1-s_m))^n}{n!} b_{B_m}^{(m-1)}(n,n)\right] + e(n_m),$$

where $e(n_m)$ satisfies

$$\begin{aligned} e(n_m) &= \alpha_{B_m} e^{A_{B_m} t s_m} \left[\sum_{n \geq n_m+1} e^{-\lambda t(1-s_m)} \frac{(\lambda t(1-s_m))^n}{n!} b_{B_m}^{(m-1)}(n,n)\right] \\ &\leq \alpha_{B_m} e^{A_{B_m} t s_m} \left[\sum_{n \geq n_m+1} e^{-\lambda t(1-s_m)} \frac{(\lambda t(1-s_m))^n}{n!}\right] \varepsilon \mathbf{1}_{B_m} \\ &\leq \alpha_{B_m} e^{A_{B_m} t s_m} \varepsilon \mathbf{1}_{B_m} \\ &\leq \varepsilon, \qquad \text{since } e^{A_{B_m} t s_m} \text{ is a substochastic matrix.} \end{aligned}$$

The global computational scheme using the truncation step n_m is shown in Figure 5, where only the dark part has to be computed. The number of dark cells is

$$n_m + (m-2)\frac{n_m(n_m+1)}{2}.$$

Similarly to the previous case, we prove that computing the sum $\sum_{n=0}^{n_m} e^{-\lambda t(1-s_m)}((\lambda t(1 - s_m))^n/n!) b_{B_m}^{(m-1)}(n,n)$ has complexity

$$\mu\left(d\left(M - \frac{m+1}{2}\right)(m-2)\frac{n_m^2}{2}\right).$$

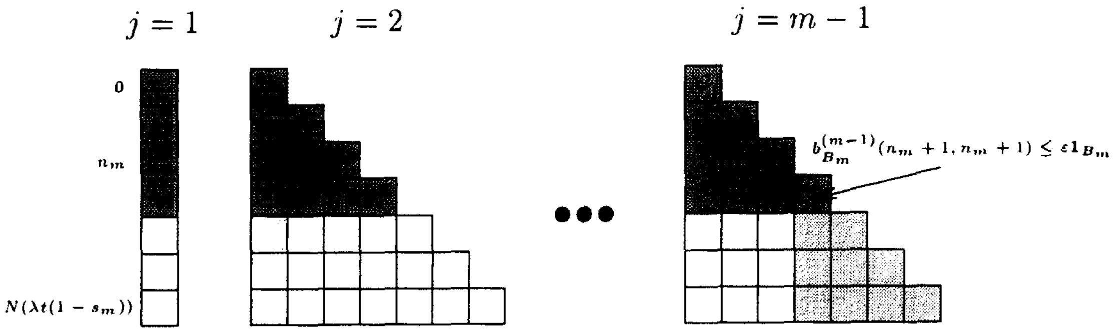
$j = 1$
$j = 2$
$j = m - 1$
0
n_m
$N(\lambda t(1 - s_m))$
$b_{B_m}^{(m-1)}(n_m + 1, n_m + 1) \leq \varepsilon \mathbf{1}_{B_m}$

Figure 5. In dark, the computed vectors. In light, cells such that $b_{B_m}^{(m-1)}(.,.) \leq \varepsilon \mathbf{1}_{B_m}$.

Next, to compute the matrix $e^{A_{B_m} s_m t}$, we use the uniformization technique. Therefore, we first denote, respectively, by λ_m and d_m the uniformization rate (i.e., $\lambda_m = \max_{i \in B_m}(-a_{ii})$) and the degree of connectivity of matrix A_{B_m}. It is easy to verify that $\lambda_m \leq \lambda$ and $d_m \leq d$. Then, the computational effort required to multiply the matrix A_{B_m} by a given column vector has complexity $\mu(d_m|B_m|)$. Moreover, it is well known that the truncation step in evaluating

$e^{A_{B_m} s_m t}$ is equal to $N(\lambda_m s_m t)$. Thus, computing the product of matrix $e^{A_{B_m} s_m t}$ by a column vector of dimension $|B_m|$ has a total cost of $\mu(d_m |B_m| N(\lambda_m s_m t))$.

We can, therefore, conclude in this case that the numerical complexity to compute the distribution of the performability is

$$\mu\left(d\left(M - \frac{m+1}{2}\right)(m-2)\frac{n_m^2}{2} + d_m |B_m| N(\lambda_m s_m t)\right).$$

Knowing that the evaluation of matrix $e^{A_{B_m} s_m t}$ introduces an error less than ε, it is easy to verify that the total error in computing $\mathbb{P}\{Y_t > s\}$, $s \geq r_{m-1} t$, is less than 2ε. On the other hand, the numerical complexity of the method presented in [8] is $\mu(dM[C'(N - C') + mC'/2])$ where C' and N are, respectively, equal to $N(\lambda t(1 - s_m))$ and $N(\lambda t)$. In practice, the truncation step n_m is very small in comparison to C'. This result leads to a low computational time comparing with the algorithm of [8].

4. A NUMERICAL EXAMPLE

We consider in this section a multiprocessor system described and modeled in [16]. The system contains N processors each with its local memory, SM shared memory modules, and B busses to communicate among the processors and the memory modules. We study the multiprocessor system when $N = \mathrm{SM} = B = 2$. The processors, shared memory modules, and busses are subject to random failures independently of each other. When one of these components fail, the number of available components decreases and consequently the processing power of the system decreases. The lifetime of each processor (respectively, bus) is assumed to be exponentially distributed with rate λ_p (respectively, λ_b). The lifetime of each shared memory module is assumed to have a probability distribution of phase type with two states, with initial probability distribution $\beta = (1, 0, 0)$ and infinitesimal generator T given by

$$T = \begin{pmatrix} -(\mu_{1,2} + \mu_{1,0}) & \mu_{1,2} & \mu_{1,0} \\ \mu_{2,1} & -(\mu_{2,1} + \mu_{2,0}) & \mu_{2,0} \\ 0 & 0 & 0 \end{pmatrix}.$$

The performance measure used here is the processing power of the system, that is, the processor utilization rate multiplied by the number of available processors. The values of the processing power have been taken from [16] and are the following. We define $pp(n, sm, b)$ as the processing power of the system when there are n processors, sm shared memory modules, and b busses available. The values of $pp(n, sm, b)$ have been taken from [16] and are the following:

$$pp(1, sm, b) = 0.8, \qquad \text{for } 1 \leq sm, b \leq 2,$$
$$pp(2, sm, 1) = 1.56, \qquad \text{for } 1 \leq sm \leq 2,$$
$$pp(2, 1, 2) = 1.56, \qquad \text{and} \qquad pp(2, 2, 2) = 1.58.$$

When one of the three n, sm, and b is 0 we set $pp(n, sm, b) = 0$. The values of the failure rates λ_p and λ_b are also chosen as in [16], that is, $\lambda_p = 6 \times 10^{-5}$ per hour and $\lambda_b = 4 \times 10^{-5}$ per hour. The parameters of the phase type distribution of the lifetime of each memory modules are given by

$$\mu_{1,2} = 14 \times 10^{-5}, \qquad \mu_{1,0} = 6 \times 10^{-5}, \qquad \mu_{2,1} = \left(\frac{4}{7}\right) \times 10^{-5}, \qquad \mu_{2,0} = \left(\frac{24}{7}\right) \times 10^{-5}.$$

With these values, we obtain a mean lifetime for each shared memory module equal to 0.25×10^5 hours which is the same expected value as in [16].

In order to obtain a Markov process to describe the behaviour of the system, we take as state description the vector $(n, (ph_1, ph_2), b)$ where $1 \leq n, b \leq 2$ and ph_i, $i = 1, 2$, denotes the number of memory modules in phase i. The value $sm = ph_1 + ph_2$ gives the number of available memory modules $(0 \leq ph_1, ph_2 \leq 2)$. All the states corresponding to the down state of the system, that is, when $n = 0$ or $sm = 0$ or $b = 0$, are lumped into only one absorbing state having a processing power equal to 0. Thus, we compute a Markov process with 21 states, one of them being an absorbing state. There are four distinct reward rates: $r_3 = 1.58$, $r_2 = 1.56$, $r_1 = 0.8$, and $r_0 = 0$ and we have $(|B_3|, |B_2|, |B_1|, |B_0|) = (3, 7, 10, 1)$, see Figure 6, where the transitions in dashed lines without destination are to the absorbing state 0. The value of ε is fixed to 10^{-6}.

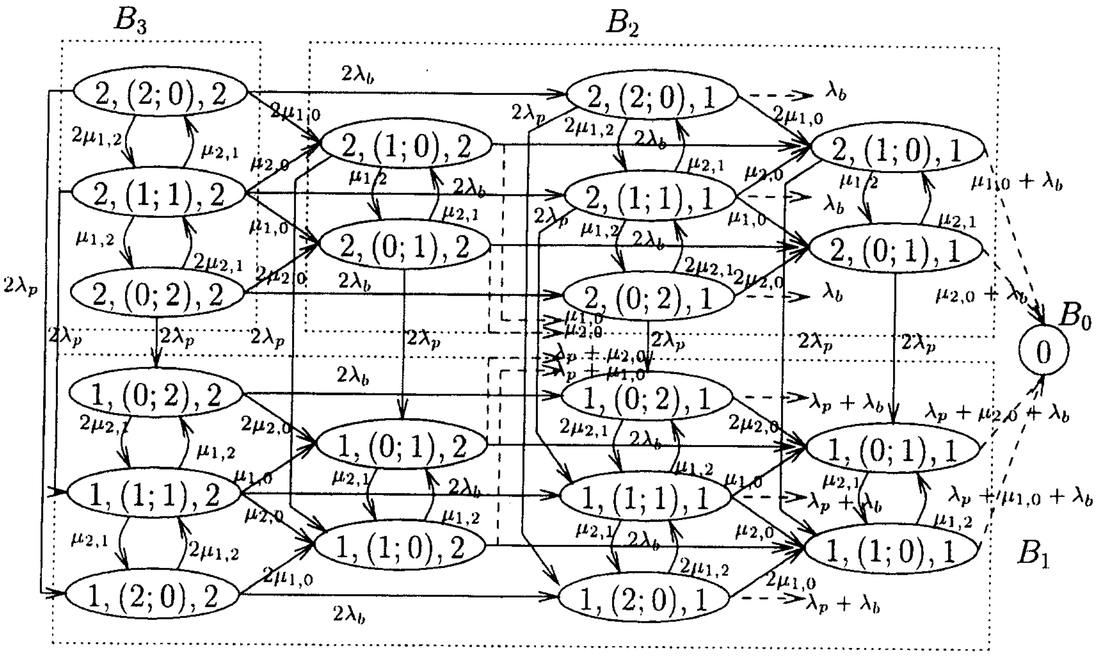

Figure 6. The Markov chain graph.

Figure 7 shows the probability to get a cumulative reward rate over the specified period $t = 10^5$ hours greater than the corresponding value on the s axis. It is interesting to note that for $s = 5 \times 10^4$, the truncation step n_1 is equal to 59 while $C = N(\lambda t s_1)$ is equal to 96. Figure 8 shows the probability that the processing power during $(0, t)$ averaged over time is greater than 99% of the maximum processing power (that is, 1.58) of the system.

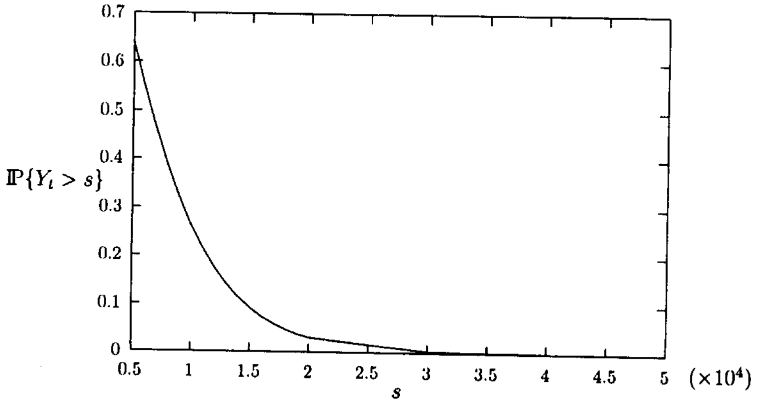

Figure 7. Distribution of Y_t for $t = 10^5$ hours.

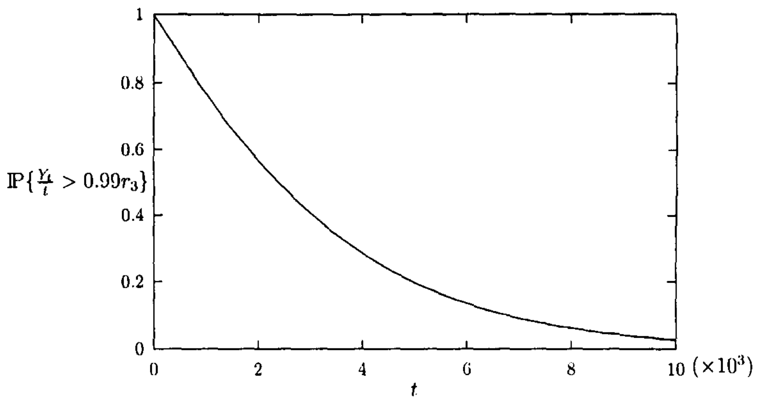

Figure 8. Distribution of Y_t/t.

Table 1. Comparison of CPU times.

s (×10⁶)	1	1.25	1.5	1.56	1.57	1.575
Algol	79.4	79.4	79.4	79.4	37.3	18.5
Algo2	0.2	0.2	0.2	0.1	0.1	0.1

Table 1 compares the computation times in seconds obtained by the execution of the general algorithm given in [8] and referred as "Algol" and of the algorithm developed here which is referred to as "Algo2". We have chosen $t = 10^6$ hours. As expected, we easily see that "Algo2" performs better than "Algol". These low computational times of "Algo2" are due to low truncation steps. For instance, for $s = 1.56 \times 10^6$, we obtain $C' = N(\lambda t(1 - s_m)) = 724$ and $n_m = 41$. All these results have been obtained on SPARCstation 5 Model 110 (Sun).

5. CONCLUSION

The current method to evaluate the performability distribution for degradable computer systems, based on the uniformization technique, leads to a new algorithm with a low polynomial computational complexity. Its main advantage involves a complexity at most quadratic in a truncation step which is smaller than the Poisson's one usually used in all methods based on uniformization. Another advantage of this algorithm is an improvement in its stability because only positive numbers bounded by 1 are involved.

APPENDIX A

Proof of Lemma 2.2. Using the recurrence relationship given by the Theorem 2.1, we can prove, by recurrence on n and k, that $b_{B_l}^{(j)}(n,k)$ is equal to 0_{B_l} for all index $l < j$. If we also use equality (1) (i.e., $P_{B_lB_k} = 0_{B_l}$ for $l < k$), the first part of the lemma is immediately proved. For the case $j = 1$, we can show by recurrence that $b_{B_1}^{(1)}(n,k) = P_{B_1B_1}^k \mathbf{1}_{B_1}$. Thus, the coefficients $b_{B_l}^{(1)}(n,k)$ do not depend on index n for $l = 1$. We suppose now that this property is true up to $l - 1$, that is,

$$\forall 1 \le i \le l-1,\ \forall n \ge 1, \text{ and } \forall 1 \le k \le n, \qquad b_{B_i}^{(1)}(n,k) \text{ is independent of } n.$$

To prove the property for index l, we proceed by recurrence on k. In fact, for $k = 0$, the term $b_{B_l}^{(1)}(n,0)$ is always equal to $\mathbf{1}_{B_l}$, so it is independent of n. On the other hand, since

$$b_{B_l}^{(1)}(n,k) = \frac{r_l - r_1}{r_l} b_{B_l}^{(1)}(n,k-1) + \frac{r_1}{r_l} \sum_{i=1}^{l} P_{B_lB_i} b_{B_i}^{(1)}(n-1,k-1),$$

we get: $b_{B_l}^{(1)}(n,k)$ is independent of n. ∎

APPENDIX B

PROOF OF COROLLARY 2.3. Using Theorem 2.1 and Lemma 2.2, we can write

$$\begin{aligned}\mathbb{P}\{Y_t > s\} &= \sum_{n=0}^{\infty} e^{-\lambda t}\frac{(\lambda t)^n}{n!}\sum_{k=0}^{n}\tbinom{n}{k}s_1^k(1-s_1)^{n-k}b^{(1)}(k)\\ &= \sum_{k=0}^{\infty}\sum_{n=k}^{\infty} e^{-\lambda t}\frac{(\lambda t)^n}{n!}\tbinom{n}{k}s_1^k(1-s_1)^{n-k}b^{(1)}(k)\\ &= e^{-\lambda t}\sum_{k=0}^{\infty}\frac{(\lambda t s_1)^k}{k!}b^{(1)}(k)\sum_{n=k}^{\infty}\frac{(\lambda t(1-s_1))^{n-k}}{(n-k)!}.\end{aligned}$$

But $\sum_{n=k}^{\infty}((\lambda t(1-s_1))^{n-k}/(n-k)!) = e^{\lambda t(1-s_1)}$, then

$$\mathbb{P}\{Y_t > s\} = e^{-\lambda t}e^{\lambda t(1-s_1)}\sum_{k=0}^{\infty}\frac{(\lambda t s_1)^k}{k!}b^{(1)}(k) = \sum_{k=0}^{\infty}e^{-\lambda t s_1}\frac{(\lambda t s_1)^k}{k!}b^{(1)}(k). \qquad ∎$$

APPENDIX C

PROOF OF COROLLARY 2.4. Since

$$b_{B_m}^{(m)}(n,0) = b_{B_m}^{(m-1)}(n,n) \qquad \text{and} \qquad b_{B_m}^{(m)}(n,k) = P_{B_m B_m} b_{B_m}^{(m)}(n-1,k-1),$$

we get

$$b_{B_m}^{(m)}(n,k) = P_{B_m B_m}^k b_{B_m}^{(m)}(n-k,0) = P_{B_m B_m}^k b_{B_m}^{(m-1)}(n-k,n-k).$$

Applying Theorem 2.1 for $j = m$, we obtain

$$\begin{aligned}\mathbb{P}\{Y_t > s\} &= \sum_{n\ge 0} e^{-\lambda t}\frac{(\lambda t)^n}{n!}\sum_{k=0}^{n}\tbinom{n}{k}s_m^k(1-s_m)^{n-k}\alpha_{B_m}b_{B_m}^{(m)}(n,k)\\ &= \sum_{n\ge 0} e^{-\lambda t}\frac{(\lambda t)^n}{n!}\sum_{k=0}^{n}\tbinom{n}{k}s_m^k(1-s_m)^{n-k}\alpha_{B_m}P_{B_m}^k b_{B_m}^{(m-1)}(n-k,n-k)\\ &= \sum_{k\ge 0}\sum_{n\ge k} e^{-\lambda t}\frac{(\lambda t)^n}{n!}\binom{n}{k}s_m^k(1-s_m)^{n-k}\alpha_{B_m}P_{B_m}^k b_{B_m}^{(m-1)}(n-k,n-k)\\ &= e^{-\lambda t}\sum_{k\ge 0}\frac{(\lambda t s_m)^k}{k!}\alpha_{B_m}P_{B_m}^k\sum_{n\ge k}\frac{(\lambda t(1-s_m))^{n-k}}{(n-k)!}b_{B_m}^{(m-1)}(n-k,n-k).\end{aligned}$$

If we write $e^{-\lambda t}$ as the product of $e^{-\lambda t s_1}$ and $e^{-\lambda t(1-s_1)}$, we obtain

$$\begin{aligned}\mathbb{P}\{Y_t > s\} &= \alpha_{B_m}\left[\sum_{k\ge 0}e^{-\lambda t s_m}\frac{(\lambda t s_m)^k}{k!}P_{B_m}^k\right]\left[\sum_{n\ge k}e^{-\lambda t(1-s_m)}\frac{(\lambda t(1-s_m))^{n-k}}{(n-k)!}b_{B_m}^{(m-1)}(n-k,n-k)\right]\\ &= \alpha_{B_m}\left[\sum_{k\ge 0}e^{-\lambda t s_m}\frac{(\lambda t s_m)^k}{k!}P_{B_m}^k\right]\left[\sum_{n\ge 0}e^{-\lambda t(1-s_m)}\frac{(\lambda t(1-s_m))^{n}}{n!}b_{B_m}^{(m-1)}(n,n)\right]\\ &= \alpha_{B_m}e^{\lambda t s_m(P_{B_m}-I_{B_m})}\left[\sum_{n\ge 0}e^{-\lambda t(1-s_m)}\frac{(\lambda t(1-s_m))^{n}}{n!}b_{B_m}^{(m-1)}(n,n)\right].\end{aligned}$$

Since $P = A/\lambda + I$, then $A_{B_m} = \lambda(P_{B_m} - I_{B_m})$. So

$$\mathbb{P}\{Y_t > s\} = \alpha_{B_m}e^{A_{B_m}t s_m}\left[\sum_{n\ge 0}e^{-\lambda t(1-s_m)}\frac{(\lambda t(1-s_m))^{n}}{n!}b_{B_m}^{(m-1)}(n,n)\right].$$

The proof of Corollary 2.4 is then completed. ∎

APPENDIX D

To prove Theorem 2.5, we first need two technical lemmas which can be proved by recurrence. To simplify notations, we denote the fraction $(r_l - r_j)/(r_l - r_{j-1})$ by $\alpha_{l,j}$. Note that $(r_j - r_{j-1})/(r_l - r_{j-1}) = 1 - \alpha_{l,j}$ and $0 \leq \alpha_{l,j} < 1$ if $l \geq j$.

LEMMA D.1. *For all $1 \leq l \leq m$ and $k \geq 1$, we have*

$$b_{B_l}^{(1)}(k) \leq b_{B_l}^{(1)}(k-1).$$

PROOF. We prove this lemma by recurrence on l. For $l = 1$, we easily obtain $b_{B_1}^{(1)}(k) \leq b_{B_1}^{(1)}(k-1)$ for all $k \in \mathbb{N}^*$ by considering the following system:

$$\begin{aligned} b_{B_1}^{(1)}(1) &\leq b_{B_1}^{(1)}(0) = \mathbf{1}_{B_1}, \\ b_{B_1}^{(1)}(k) &= P_{B_1} b_{B_1}^{(1)}(k-1), \\ P_{B_1}\mathbf{1}_{B_1} &\leq \mathbf{1}_{B_1}. \end{aligned}$$

We suppose now that the property is true up to $l-1$. To prove the property for index l, we proceed by recurrence on k. The initial condition for $k = 1$ is verified since $b_{B_l}^{(1)}(1) \leq b_{B_l}^{(1)}(0) = \mathbf{1}_{B_l}$. The property for $k-1$ is equivalent to

$$b_{B_l}^{(1)}(k-1) \leq b_{B_l}^{(1)}(k-2) \tag{D1}$$

The Lemma 2.2 can be written, for $j = 1$, as

$$\begin{aligned} b_{B_l}^{(1)}(k) &= \alpha_{l,1} b_{B_l}^{(1)}(k-1) + (1-\alpha_{l,1}) \sum_{i=1}^{l} P_{B_l B_i} b_{B_i}^{(1)}(k-1) \\ &= \alpha_{l,1} b_{B_l}^{(1)}(k-1) + (1-\alpha_{l,1}) \left[\sum_{i=1}^{l-1} P_{B_l B_i} b_{B_i}^{(1)}(k-1) + P_{B_l B_l} b_{B_l}^{(1)}(k-1) \right]. \end{aligned}$$

According to the recurrence hypothesis on index $l-1$, we have

$$\forall 1 \leq i \leq l-1, \qquad b_{B_i}^{(1)}(k-1) \leq b_{B_i}^{(1)}(k-2).$$

Since the P matrix is positive and $\alpha_{l,1} \in [0,1[$, we obtain

$$(1-\alpha_{l,1}) \sum_{i=1}^{l-1} P_{B_l B_i} b_{B_i}^{(1)}(k-1) \leq (1-\alpha_{l,1}) \sum_{i=1}^{l-1} P_{B_l B_i} b_{B_i}^{(1)}(k-2).$$

The recurrence hypothesis (D1) gives

$$P_{B_l B_l} b_{B_l}^{(1)}(k-1) \leq P_{B_l B_l} b_{B_l}^{(1)}(k-2) \qquad \text{and} \qquad \alpha_{l,1} b_{B_l}^{(1)}(k-1) \leq \alpha_{l,1} b_{B_l}^{(1)}(k-2).$$

It follows that

$$\begin{aligned} b_{B_l}^{(1)}(k) &\leq \alpha_{l,1} b_{B_l}^{(1)}(k-2) + (1-\alpha_{l,1}) \sum_{i=1}^{l} P_{B_l B_i} b_{B_i}^{(1)}(k-2) \\ &= b_{B_l}^{(1)}(k-1). \end{aligned}$$

The proof of the lemma is completed. ∎

REMARK. According to Lemma 2.2, we observe that the coefficient $b_{B_l}^{(j)}(n,k)$ is a convex combination of the two vectors $b_{B_l}^{(j)}(n,k-1)$ and $\sum_{i=j}^{l} P_{B_l B_i} b_{B_i}^{(j)}(n-1,k-1)$. So, we obtain the following equivalence:

$$b_{B_l}^{(j)}(n,k) \leq b_{B_l}^{(j)}(n,k-1) \iff \sum_{i=j}^{l} P_{B_l B_i} b_{B_i}^{(j)}(n-1,k-1) \leq b_{B_l}^{(j)}(n,k-1). \tag{D2}$$

This remark above will be exploited in the following lemma.

LEMMA D.2. *For $n \geq 1$, $2 \leq j \leq m$, and $j \leq l \leq m$, we have*

$$b_{B_l}^{(j-1)}(n,n) \leq b_{B_l}^{(j-1)}(n,n-1) \Longrightarrow b_{B_l}^{(j)}(n,1) \leq b_{B_l}^{(j)}(n,0).$$

PROOF. Since we have

$$b_{B_l}^{(j)}(n,0) = b_{B_l}^{(j-1)}(n,n) = \alpha_{l,j-1} b_{B_l}^{(j-1)}(n,n-1) + (1-\alpha_{l,j-1}) \sum_{i=j-1}^{l} P_{B_l B_i} b_{B_i}^{(j-1)}(n-1,n-1),$$

then

$$\sum_{i=j}^{l} P_{B_l B_i} b_{B_i}^{(j)}(n-1,0) \leq b_{B_l}^{(j)}(n,0)$$

$$\Updownarrow$$

$$\sum_{i=j}^{l} P_{B_l B_i} b_{B_i}^{(j)}(n-1,0) \leq \alpha_{l,j-1} b_{B_l}^{(j-1)}(n,n-1) + (1-\alpha_{l,j-1}) \sum_{i=j-1}^{l} P_{B_l B_i} b_{B_i}^{(j-1)}(n-1,n-1)$$

$$\Updownarrow$$

$$0_{B_l} \leq \alpha_{l,j-1} \left[b_{B_l}^{(j-1)}(n,n-1) - \sum_{i=j-1}^{l} P_{B_l B_i} b_{B_i}^{(j-1)}(n-1,n-1) \right] + P_{B_l B_{j-1}} b_{B_{j-1}}^{(j-1)}(n-1,n-1).$$

So, if $b_{B_l}^{(j-1)}(n,n) \leq b_{B_l}^{(j-1)}(n,n-1)$, then, according to (D2), the vector between hooks is greater than 0_{B_l}. This is equivalent to $\sum_{i=j}^{l} P_{B_l B_i} b_{B_i}^{(j)}(n-1,0) \leq b_{B_l}^{(j)}(n,0)$. By taking into account equivalence (D2), we get $b_{B_l}^{(j)}(n,1) \leq b_{B_l}^{(j)}(n,0)$. ∎

PROOF OF THE FIRST INEQUALITY OF THEOREM 2.5. To prove this first inequality, we proceed by recurrence on the indexes j, n, and k. For $1 \leq j \leq m$ fixed, we consider the following property $\Re(j)$:

$$\Re(j) : \left[\forall n \geq 1,\ \forall 1 \leq k \leq n, \text{ and } \forall j \leq l \leq m,\ b_{B_l}^{(j)}(n,k) \leq b_{B_l}^{(j)}(n,k-1) \right].$$

For $j = 1$, according to Lemma D.1, the vectors $b_{B_l}^{(1)}(n,k)$ are independent of n, so $\Re(1)$ is verified. We suppose that $\Re(j-1)$ is true. In order to prove the property $\Re(j)$, we proceed by recurrence on n. For $n = 1$, the recurrence hypothesis $\Re(j-1)$ leads to $b_{B_l}^{(j-1)}(1,1) \leq b_{B_l}^{(j-1)}(1,0)$, which gives according to Lemma D.2

$$b_{B_l}^{(j)}(1,1) \leq b_{B_l}^{(j)}(1,0).$$

We suppose now that the property holds for $n-1$. This means that the property $\Re(j,n-1)$ is true, where

$$\Re(j,n-1) \text{ is } \left[\forall 1 \leq k \leq n-1 \text{ and } \forall j \leq l \leq m,\ b_{B_l}^{(j)}(n-1,k) \leq b_{B_l}^{(j)}(n-1,k-1) \right].$$

In order to prove $\Re(j,n)$, we proceed by recurrence on k. For $k = 1$, the hypothesis $\Re(j-1)$ allows us to write $b_{B_l}^{(j-1)}(n,n) \leq b_{B_l}^{(j-1)}(n,n-1)$, which gives according to Lemma D.2,

$$b_{B_l}^{(j)}(n,1) \leq b_{B_l}^{(j)}(n,0).$$

At present, we suppose the property true for $k-1$. That means that property $\Re(j,n,k-1)$ is true, where

$$\Re(j,n,k-1) \text{ is } \left[\forall j \leq l \leq m,\ b_{B_l}^{(j)}(n-1,k) \leq b_{B_l}^{(j)}(n-1,k-1) \right].$$

According to Lemma 2.2, $b_{B_l}^{(j)}(n,k)$ and $b_{B_l}^{(j)}(n,k-1)$ can be written as

$$b_{B_l}^{(j)}(n,k) = \alpha_{l,j} b_{B_l}^{(j)}(n,k-1) + (1-\alpha_{l,j}) \sum_{i=j}^{l} P_{B_l B_i} b_{B_i}^{(j)}(n-1,k-1),$$

$$b_{B_l}^{(j)}(n,k-1) = \alpha_{l,j} b_{B_l}^{(j)}(n,k-2) + (1-\alpha_{l,j}) \sum_{i=j}^{l} P_{B_l B_i} b_{B_i}^{(j)}(n-1,k-2).$$

Moreover, according to $\Re(j,n-1)$, we have $b_{B_i}^{(j)}(n-1,k-1) \le b_{B_i}^{(j)}(n-1,k-2)$, and, since the matrix P is positive and $0 \le \alpha_{l,j} < 1$, we get

$$(1-\alpha_{l,j}) \sum_{i=j}^{l} P_{B_l B_i} b_{B_i}^{(j)}(n-1,k-1) \le (1-\alpha_{l,j}) \sum_{i=j}^{l} P_{B_l B_i} b_{B_i}^{(j)}(n-1,k-2).$$

The hypothesis $\Re(j,n,k-1)$ and the previous expressions for $b_{B_l}^{(j)}(n,k)$ and for $b_{B_l}^{(j)}(n,k-1)$ lead to the required inequality

$$b_{B_l}^{(j)}(n,k) \le b_{B_l}^{(j)}(n,k-1).$$

The proof of the the first part of Theorem 2.5 is then completed. ∎

Proof of the Second Inequality of Theorem 2.5. For $j = 1$, the sequence $(b_{B_l}^{(1)}(n,k))_{n \ge k}$ is independent of n. We suppose now that the property holds for $j-1$. We must prove it for the the index j. The elements $b_{B_l}^{(j)}(n,0)$ and $b_{B_l}^{(j)}(n-1,0)$ are, respectively, equal to $b_{B_l}^{(j-1)}(n,n)$ and $b_{B_l}^{(j-1)}(n-1,n-1)$. According to Theorem 2.5, we have $b_{B_l}^{(j-1)}(n,n) \le b_{B_l}^{(j-1)}(n,n-1)$, and according to the recurrence hypothesis, we have $b_{B_l}^{(j-1)}(n,n-1) \le b_{B_l}^{(j-1)}(n-1,n-1)$. So, we get $b_{B_l}^{(j-1)}(n,n) \le b_{B_l}^{(j-1)}(n-1,n-1)$. It follows that

$$b_{B_l}^{(j)}(n,0) \le b_{B_l}^{(j)}(n-1,0).$$

This inequality proves the result for $k = 0$. We suppose now that the property holds for $k-1$ and we show that $b_{B_l}^{(j)}(n,k) \le b_{B_l}^{(j)}(n-1,k)$. For this purpose, we write

$$\begin{aligned} b_{B_l}^{(j)}(n,k) &= \alpha_{l,j} b_{B_l}^{(j)}(n,k-1) + (1-\alpha_{l,j}) \sum_{i=j}^{l} P_{B_l B_i} b_{B_i}^{(j)}(n-1,k-1) \\ &\le \alpha_{l,j} b_{B_l}^{(j)}(n-1,k-1) + (1-\alpha_{l,j}) \sum_{i=j}^{l} P_{B_l B_i} b_{B_i}^{(j)}(n-2,k-1) \\ &= b_{B_l}^{(j)}(n-1,k), \end{aligned}$$

which completes the proof of Theorem 2.5. ∎

REFERENCES

1. J.F. Meyer, On evaluating the performability of degradable computing systems, *IEEE Transactions on Computers* **C.29** (8), 720–731, (August 1980).
2. R.M. Smith, K.S. Trivedi and A.V. Ramesh, Performability analysis: Measures, an algorithm, and a case study, *IEEE Transactions on Computers* **37** (4), 406–417, (April 1988).
3. E. de Souza e Silva and H.R. Gail, Calculating availability and performability measures of repairable-computer systems using randomization, *J. ACM* **36** (1), 171–193, (January 1989).
4. E. de Souza e Silva and H.R. Gail, Performability analysis of computer systems: From model specification to solution, *Performance Evaluation*, Performability Modelling of Computer and Communication Systems **14** (3/4), 157–196, (February 1992).
5. L. Donatiello and V. Grassi, On evaluating the cumulative performance distribution of fault-tolerant computer systems, *IEEE Transactions on Computers* **C-40** (11), 1301–1307, (November 1991).

6. L. Donatiello and V. Grassi, Sensitivity analysis of performability, *Performance Evaluation,* Performability Modelling of Computer and Communication Systems **14** (3/4), 227–238, (February 1992).
7. E. de Souza e Silva and H.R. Gail, Calculating transient distributions of cumulative reward, Technical Report CDS-930033, UCLA, University of California, Los Angeles, (September 1993).
8. H. Nabli and B. Sericola, Performability analysis: A new algorithm, *IEEE Transactions on Computers* **45** (4), 491–494, (April 1996).
9. H. Nabli and B. Sericola, Performability Analysis of fault-tolerant computer systems, Technical Report 2254, INRIA, Campus de Beaulieu, (May 1994); `ftp://ftp.inria.fr/INRIA/tech-reports/publi-ps-gz/RR/RR-2254.ps.gz`.
10. K.R. Pattipati, Y. Li and H.A.P. Blom, A unified framework for the preformability evaluation of fault-tolerant computer systems, *IEEE Transactions on Computers* **42**, 312–326, (March 1993).
11. J.F. Meyer, Closed-form solutions of performability, *IEEE Transactions on Computers* **C-31** (7), 648–657, (July 1982).
12. J.F. Meyer, D.G. Furchtgott and L.T. Wu, Performability evaluation of the SIFT computer, *IEEE Transactions on Computers* **C-29** (6), 501–509, (June 1980).
13. M.D. Beaudry, Performance-related reliability measures for computing systems, *IEEE Transactions on Computers* **C-27**, 540–547, (June 1978).
14. G. Ciardo, R. Marie, B. Sericola and K.S. Trivedi, Performability analysis using semi-Markov reward processes, *IEEE Transactions on Computers* **C-39**, 1251–1264, (October 1990).
15. B.R. Iyer, L. Donatiello and P. Heidelberger, Analysis of performability for stochastic models of fault-tolerant systems, *IEEE Transactions on Computers* **C-35** (10), 902–907, (1986).
16. A. Goyal and A.N. Tantawi, Evaluation of performability for degradable computer systems, *IEEE Transactions on Computers* **C-36** (6), 738–744, (June 1987).
17. H. Nabli, Performability measure for acyclic Markovian models, *Computers Math. Applic.* **35** (8), 41–51, (1998).
18. S.M. Ross, *Stochastic Processes*, John Wiley and Sons, (1983).
19. G. Rubino and B. Sericola, Interval availability distribution computation, In *Proceedings IEEE 23rd Fault-Tolerant Computing Symposium*, Toulouse, France, pp. 49–55, (1993).
20. H. Nabli, Mesure de performabilité sur des processus de Markov homogènes à espace d'états fini, Thèse de Doctorat, Vol. 1378, pp. 136–150, Université de Rennes I, (September 1995); `http://www.irisa.fr/model/theses.html`.
21. P.N. Bowerman, R.G. Nolty and E.M. Scheuer, Calculation of the Poisson cumulative distribution function, *IEEE Transactions on Computers* **39** (2), 158–161, (June 1990).