PERGAMON

Computers & Operations Research 26 (1999) 211–218

computers &
operations
research

Point and expected interval availability analysis with stationarity detection

Haïscam Abdallah*[1], Raymond Marie[2], Bruno Sericola[3]

IRISA, Campus de Beaulieu, 35042 Rennes Cedex, France

Received June 1996; received in revised form June 1998

Scope and purpose

The advent of fault-tolerant computing systems has led to increased interest in analytic techniques for prediction of dependability measures such as the availability over a given period. Interval availability is defined by the fraction of time during which a system is in operation over a finite observation period. By modeling the system behaviour by a Markov process, we transform the problem of evaluating dependability cumulative measures into the computation of the cumulative measures on a Markov process. In this note we are interested in the expected interval availability. Generally, we are faced with the problem of the execution time, especially when the Markov model is stiff, i.e., when we have a highly available system. This note proposes a new technique which deals efficiently with such a class of processes.

Abstract

Interval availability is a dependability measure defined by the fraction of time during which a system is in operation over a finite observation period. The system is assumed to be modeled by a finite Markov process. Because the computation of the distribution of this random variable is very expensive, it is

* Corresponding author.

[1] Haïscam Abdallah received the B.S. degree in Mathematics from the Libanese University, Lebanon, and the Ph.D. degree in Computer Science from the University of Rennes I, France. Since 1990, he is a Maître de Conférence in Computer Science at the University of Rennes II. His active research interests include dependability measures evaluation of critical computing systems and numerical analysis.

[2] Raymond A. Marie received the Doctorat d'Ingénieur and the Doctorat d'Etates-Sciences Mathématiques from the University of Rennes I, France. He is a Research Manager of an INRIA project in modelling and a Professor at the Computer Science Department of the University of Rennes I. His active research interests include performance evaluation of computer systems and reliability computation of complex systems.

[3] Bruno Sericola received the Ph.D. degree in Computer Science from the University of Rennes I in 1988. He is with INRIA (Institute National de Recherche en Informatique et Automatique, a public research French laboratory). His main research activity is in computer and communication systems performance evaluation, dependability and perfomability of fault-tolerant architectures and applied stochastic processes.

PII: S0305-0548(98)00057-4

common to compute only its expectation. In this note, we propose a new algorithm to compute the expected interval availability and we compare it with respect to the standard uniformization technique from an execution time point of view. This new method is particularly interesting if the Markov chain is stiff.

Moreover, a new algorithm for the stationarity detection is proposed in order to avoid excessive computation.

Keywords: Availability; Markov process; Uniformization; Stationarity detection

1. Introduction

In the dependability analysis of repairable computing systems, there is an increasing interest in evaluating cumulative measures such as the availability over a given period. This note deals with the computation of the interval availability which is defined by the fraction of time during which a system is in operation over a finite observation period. For a general presentation of the interest of such a subject, we refer to [1–6].

In this note we develop a new method to compute the expectation of the interval availability. It is based on the technique of the uniformized power [7]. The interest of this new approach is that it will save computation time if the state space is not too large and the mission time is long; especially, if the Markov chain is stiff. Moreover, a new algorithm for the stationarity detection is proposed in order to avoid excessive computation. While the previous stationary detection methods are based on heuristics [8], our algorithm is based on a theorem which bounds the error.

The remainder of the note is organized as follows. In the following section, we present our approach based on the uniformized power technique. In Section 3, we give a formal criterion for stationarity detection and the corresponding algorithm. Section 4 is devoted to a numerical illustration of the proposed algorithm and a comparison of the computation time of our method with respect to standard uniformization technique.

2. Expected interval availability analysis

Consider a system modeled by a continuous-time homogeneous Markov process, say $X = \{X_t, t \geqslant 0\}$, defined over a finite state space denoted by S and with cardinality M, infinitesimal generator $\mathbf{A}$ and initial probability distribution α. The set S is partitioned into two disjoint subsets denoted by U and D, containing, respectively, the *up* states and *down* states. The transition probability matrix at time t is denoted by $\mathbf{P}(t)$, which is the exponential of the matrix $\mathbf{A}t$. Let $\mathbf{1}_U$ be the column vector such that its ith entry is equal to 1 if $i \in U$ and 0 if $i \in D$. We then have the Point Availability $PAV(s)$

$$PAV(s) = \mathbb{P}(X_s \in U) = \alpha \mathbf{P}(s) \mathbf{1}_U = \alpha e^{\mathbf{A}s} \mathbf{1}_U.$$

The Standard Uniformization approach (SU) [9, 6] allows to obtain the expected interval availability over $(0, t)$ as

$$\mathbb{E}[IAV(t)] = \frac{1}{t}\int_0^t PAV(s)\,\mathrm{d}s = \sum_{n=0}^{+\infty} p(n, vt)\,\frac{1}{n+1}\sum_{k=0}^{n} \alpha\mathbf{P}^k\mathbf{1}_U, \tag{1}$$

where $p(n, vt) = \mathrm{e}^{-vt}(v^n t^n/n!)$, $v \geqslant \max\{-a_{ii}, i \in S\}$, $\mathbf{P} = \mathbf{I} + \mathbf{A}/v$, and $\mathbf{I}$ is the identity matrix.

Let ε be a given error tolerance, we can write

$$\mathbb{E}[IAV(t)] = \sum_{n=0}^{N} p(n, vt)\,\frac{1}{n+1}\sum_{k=0}^{n} \alpha\mathbf{P}^k\mathbf{1}_U + e(N), \tag{2}$$

where $e(N) \leqslant \varepsilon$ if N is chosen such that $1 - \sum_{n=0}^{N} p(n, vt) \leqslant \varepsilon$.

For large values of t, the computation of formula (2) is expensive when the product vt is large since N is always greater than vt.

In order to present our approach based on the Uniformized Power technique (UP) [7], we first define the stochastic matrix $\mathbf{C}(t) = (c_{ij}(t))$ as $\mathbf{C}(t) = (1/t)\int_0^t \mathbf{P}(s)\,\mathrm{d}s$. Let $m \in \mathbb{N}$ and $t_0 > 0$ such that $t_0 = t/2^m$.

By a recurrence relation, we will obtain $\mathbf{C}(t)$ by starting from $\mathbf{C}(t_0)$ and finally compute $\mathbb{E}[IAV(t)]$ as $\mathbb{E}[IAV(t)] = \alpha\mathbf{C}(t)\mathbf{1}_U$.

For that, we first construct a recurrence relation between $\mathbf{C}(2^{l+1}t_0)$ and $\mathbf{C}(2^l t_0)$. For all integer $l \geqslant 0$, we have

$$\begin{aligned}\mathbf{C}(2^{l+1}t_0) &= \frac{1}{2^{l+1}t_0}\int_0^{2^{l+1}t_0}\mathbf{P}(s)\,\mathrm{d}s = \frac{1}{2^{l+1}t_0}\left[\int_0^{2^l t_0}\mathbf{P}(s)\,\mathrm{d}s + \int_{2^l t_0}^{2^{l+1}t_0}\mathbf{P}(s)\,\mathrm{d}s\right]\\ &= \frac{1}{2}\left[C(2^l t_0) + \frac{1}{2^l t_0}\mathbf{P}(t_0)^{2^l}\int_0^{2^l t_0}\mathbf{P}(s)\,\mathrm{d}s\right]\\ &= \left[\frac{\mathbf{P}(t_0)^{2^l} + \mathbf{I}}{2}\right]\mathbf{C}(2^l t_0)\end{aligned} \tag{3}$$

The matrices $\mathbf{P}(t_0)$ and $\mathbf{C}(t_0)$ verify (see Eq. (1))

$$\mathbf{P}(t_0) = \sum_{n=0}^{+\infty} p(n, vt_0)\,\mathbf{P}^n \text{ and } \mathbf{C}(t_0) = \sum_{n=0}^{+\infty} p(n, vt_0)\frac{1}{n+1}\sum_{k=0}^{n}\mathbf{P}^k.$$

We consider the classical $\|\cdot\|_\infty$ norm which is defined, for a square matrix $\mathbf{H} = (h_{ij})$ of order M as $\|\mathbf{H}\|_\infty = \max_{1\leqslant i\leqslant M}\sum_{j=1}^{M}|h_{ij}|$. For fixed values of t_0 and N_0, we define $e = 1 - \sum_{n=0}^{N_0} p(n, vt_0)$,

$$\mathbf{P}^*(l) = \frac{1}{(1-e_0)^{2_l}}\left(\sum_{n=0}^{N_0} p(n, vt_0)\,\mathbf{P}^n\right)^{2^l} \text{ and } \mathbf{C}^*(0) = \frac{1}{1-e_0}\sum_{n=0}^{N_0} p(n, vt_0)\,\frac{1}{n+1}\sum_{k=0}^{n}\mathbf{P}^k,$$

and recursively for all integers $l \geqslant 0$,

$$\mathbf{C}^*(l+1) = \left[\frac{\mathbf{P}^*(l) + \mathbf{I}}{2}\right]\mathbf{C}^*(l).$$

The following lemma gives a useful error bound.

Lemma 2.1. *For every integer* $l \geqslant 0$,

$$\| \mathbf{P}(2^l t_0) - \mathbf{P}^*(l) \|_\infty \leqslant 2^{l+1} e_0 \text{ and } \| \mathbf{C}(2^l t_0) - \mathbf{C}^*(l) \|_\infty \leqslant 2^{l+1} e_0$$

Proof. See [10]. □

The choice of vt_0 in $]0, 1[$ implies that $p(n, vt_0)$ decreases when n increases. Because the time complexity of the UP method is $O(N_0 + m)\,(M^3 + M^2)$, the chosen values of N_0 and m are those that minimize $N_0 + m$. Experimentally, we have found that taking m as the first integer such that $vt_0 = vt/2^m \leqslant 0.1$ gives the lowest value of $N_0 + m$ while keeping $N_0 < 10$ and $e_0 \leqslant \varepsilon_0$ where ε_0 is the machine epsilon.

3. Stationarity detection

Let the vector $\pi = (\pi_j)$ denote the stationary probability distribution of the Markov process X. This vector verifies $\pi \mathbf{A} = 0$, $\pi \mathbf{P} = \pi$ and so for every $l \geqslant 0$, $\pi \mathbf{P}^*(l) = \pi$. It follows that if we define $s(l) = \min_{i \in S} \sum_{j \in U} p^*_{ij}(l)$ and $S(l) = \max_{i \in S} \sum_{j \in U} p^*_{ij}(l)$, we have

$$s(l) \leqslant \sum_{j \in U} \pi_j \leqslant S(l)$$

It is easy to show that the sequence $s(l)$ increases with respect to l and the sequence $S(l)$ decreases with respect to l. The following results give a criterion to stop the computation when the stationary behaviour is reached. For every $l \geqslant 0$, we define $h(l) = \alpha \mathbf{C}^*(l) \mathbf{1}_U$ and for $k \leqslant m$,

$$h^*(k) = \frac{1}{2^{m-k}}\, h(k) + \left(1 - \frac{1}{2^{m-k}}\right) \frac{S(k) + s(k)}{2}\,.$$

Theorem 3.1. *Let* $\varepsilon_s > 0$ *and* $k < m$ *such that* $S(k) - s(k) \leqslant \varepsilon_s$ *we then have*

$$|h(m) - h^*(k)| \leqslant \left(1 - \frac{1}{2^{m-k}}\right) \frac{\varepsilon_s}{2}\,.$$

Proof. See [10]. □

Corollary 3.2. *Let* $\varepsilon_s > 0$ *and* $k \leqslant m$ *such that* $S(k) - s(k) \leqslant \varepsilon_s$. *We then have*

$$|\mathbb{E}[IAV(t)] - h^*(k)| \leqslant 2^{m+1} e_0 + \left(1 - \frac{1}{2^{m-k}}\right) \frac{\varepsilon_s}{2}\,. \tag{4}$$

Proof. We have $|\mathbb{E}[IAV(t)] - h^*(k)| \leqslant |\mathbb{E}[IAV(t)] - h(m)| + |h(m) - h^*(k)|$ and the result follows using Lemma 2.1 and Theorem 3.1. □

We now present the algorithm to compute the point availability at time t and the expected interval availability over $(0, t)$. For every $l \geqslant 0$, we define the column vector $u_l = \mathbf{C}^*(l) \mathbf{1}_U$. In the algorithm, the value of ε_s is fixed to $2^{m-2}\varepsilon_0$. The user can give another value of ε_s.

```
input: $t$
output: $PAV(t)$, $\mathbb{E}[IAV(t)]$
Compute $m = \min\{k/vt/2^k \leqslant 0.1\}$, $t_0 = t/2^m$, and $\varepsilon_s$ such that $\varepsilon_s = 2^{m-2}\varepsilon_0$
Compute $N_0$ such that $e_0 \leqslant \varepsilon_0$, $\mathbf{P}^*(0)$, $\mathbf{u}_0$, $s(0)$ and $S(0)$
$k = 0$
while $(k < m)$ and $(S(k) - s(k) > \varepsilon_s)$ do
    $k = k + 1$
    $\mathbf{u}_k = \left[\dfrac{\mathbf{P}^*(k-1) + \mathbf{I}}{2}\right]\mathbf{u}_{k-1}$
    $h(k) = \alpha\mathbf{u}_k$
    $\mathbf{P}^*(k) = \mathbf{P}^*(k-1)\,\mathbf{P}^*(k-1)$
    Compute $s(k)$ and $S(k)$
endwhile
if $k = m$ then
        $\mathbb{E}[IAV(t)] = h(m) \pm 2^{m+1}e_0$
        $PAV(t) = \alpha\mathbf{P}^*(m)\mathbf{1}_U \pm 2^{m+1}e_0$
        $s(m) \leqslant PAV(+\infty) = \mathbb{E}[IAV(+\infty)] \leqslant S(m)$
endif
if $S(k) - s(k) \leqslant \varepsilon_s$ then
        $\mathbb{E}[IAV(t)] = h^*(k) \pm (2^{m+1}e_0 + (1 - (1/2^{m-k}))(\varepsilon_s/2))$
        $PAV(t) = (S(k) + s(k))/2 \pm (\varepsilon_s/2)$
        $\mathbb{E}[IAV(+\infty)] = PAV(+\infty) = (S(k) + s(k))/2 \pm (\varepsilon_s/2)$
endif
```

Let K be the final value of k in the algorithm ($K \leqslant m$). For a given value of t, the UP method requires $(N_0 + K)$ matrix–matrix products and $(N_0 + K)$ matrix–vector products in order to compute $h^*(K)$. The time complexity of this method is $O((N_0 + K)(M^3 + M^2))$. The time complexity of the SU method is $O(N\beta)$ where β is the number of non-null entries in $\mathbf{A}$ and N is the truncation step (see Eq. (2)). So, the ratio of time complexities is

$$R = \frac{(N_0 + K)(M^3 + M^2)}{N\beta}, \quad N > vt$$

It is clear that the UP method is faster than the SU method ($R < 1$) for large values of t corresponding to stiff Markov models and for moderated size of the cardinality M of the state space.

4. Numerical illustration

We present in this section a numerical illustration of our algorithm. The model used here is a system with n components [11]. Assume that the system components are binary, i.e. assume they have only two states, "down" and "up". Then the number of system states is $M = 2^n$. We assume that system components fail independently, that repair times are stochastically independent of component lives and that maintenance policy is either unrestricted (there are as many available repairmen as system components) or such that repairs in progress are independent of each other. If we consider the model as a reliability model, we assume system components are all repairable and separately maintained. Let λ_i and μ_i be, respectively, the failure and the repair rate of component i, $i = 1, \dots, n$. We assume that the system is down as soon as more than one component is down. So we have $n + 1$ operational states. For our experiments, we suppose $\lambda_i = \lambda$ and $\mu_i = \mu$ for all i. Therefore, the formal expression of $\mathbb{E}[IAV(t)]$ is given by

$$\mathbb{E}[IAV(t)] = \frac{1}{t}\int_0^t [(a_0(s))^n + n(a_0(s))^{n-1}\, a_1(s)]\,\mathrm{d}s,$$

where

$$a_0(s) = \frac{\mu}{\lambda + \mu} + \left(\frac{\lambda}{\lambda + \mu}\right)\mathrm{e}^{-(\lambda+\mu)s} \quad \text{and} \quad a_1(s) = \frac{\lambda}{\lambda + \mu} - \left(\frac{\lambda}{\lambda + \mu}\right)\mathrm{e}^{-(\lambda+\mu)s}.$$

This formal expression allows us to use the Maple package in order to compute the reference numerical values of $\mathbb{E}[IAV(t)]$. For the experimentations, the parameter values have been set to $n = 4$, $\lambda = 10^{-3}$, and $\mu = 1$. This algorithm has been run on a SUN4 workstation in double precision arithmetic ($\varepsilon_0 \approx 10^{-16}$) for $t = 100 \times 2^i$, $i = 0, 1, \dots, 20$. For these 21 values of t, N_0 stays constant ($N_0 = 9$). When t is evaluated from 100×2^i to $100 \times 2^{i+1}$, the value of vt_0 remains constant ($vt_0 = 9.765 \times 10^{-2}$) and m is increased by one. The real error is always lower than its theoretical bound.

For the same example, Fig. 1 shows the evolution of $S(l) - s(l)$ for $t = 100 \times 2^{20}$ as function of l. The values of l changes from 1 to 10 because the stationarity is detected at $t_d = 12.5 = 2^9 t_0$ ($\varepsilon_s = 2^{31} \times 10^{-16} \approx 2.15 \times 10^{-7}$).

However, the user can run computations for values of t such that the stationarity is detected for k smaller than m. For example, if we consider the same model, the value of K corresponding to

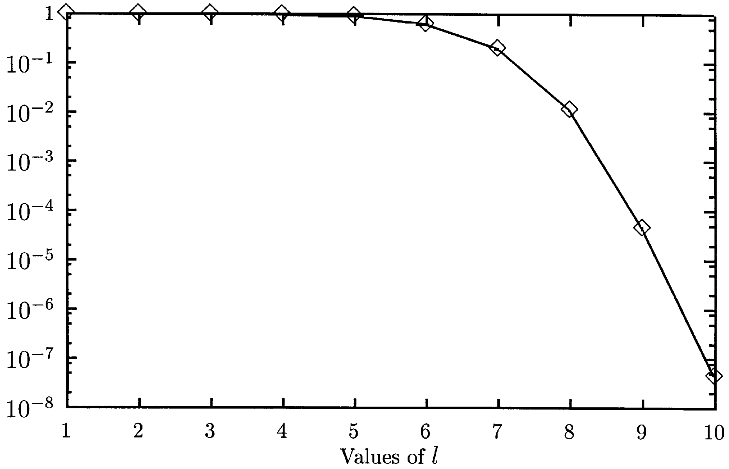

Fig. 1. Evolution of $S(l) - s(l)$ as function of l.

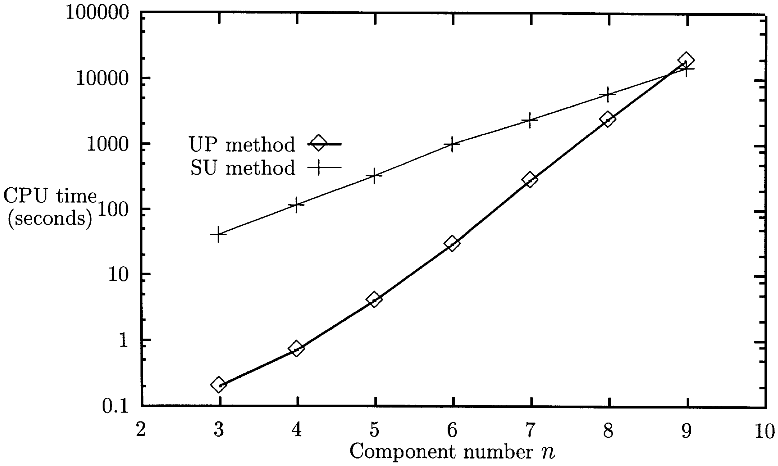

Fig. 2. CPU times function of the component number n for $t = 10^5$.

$m = 30$ ($t = 13107200$) is equal to 10. In order to compare the time complexity of the UP and SU methods, we fix a value of t and give the CPU time as a function of the component number n, with $\lambda = 10^{-8}$, $\mu = 1$, and $t = 10^5$. Fig. 2 shows the CPU times required by the SU and UP method in such a case.

For the SU method, we have used $\varepsilon = 10^{-13}$. Let us remark that in this example, the UP method remains faster than the SU method as long as the CPU time is lower than 3 h. Moreover, if t is very large, because of the detection of the stationarity by our method, the UP method remains faster than the SU method even for M of the order of 10^3. In fact, in such a case, the SU method has a prohibitive CPU time. For this example, if t was changed from 10^5 to 10^8, then the UP method would take the same CPU time (because K would not change) while the SU method would require a CPU time roughly 1000 times longer.

References

[1] Goyal A, Tantawi AN. A measure of guaranteed availability and its numerical evaluation. IEEE Transactions on Computers 1998;C-37(1):25–32.

[2] Rubino G, Sericola B. Interval availability analysis using operational periods. Performance Evaluation 1992;14:257–72.

[3] de Souza e Silva E, Gail HR. Calculating cumulative operational time distributions of repairable computer systems. IEEE Transactions on Computers 1986;C-35:322–32.

[4] Rubino G, Sericola B. Interval availability distribution computation. Proceedings IEEE 23-th Fault-Tolerant Computing Symposium, Toulouse, France, 1993;49–55.

[5] Reibman A, Smith R, Trivedi K. Markov and Markov reward model transient analysis: an overview of numerical approaches. European Journal of Operational Research 1989;40:257–67.

[6] Reibman A, Trivedi K. Transient Analysis of Cummulative Measures of Markov Model Behavior. Communication Statistics-Stochastic Models 1989;5(4):683–710.

[7] Abdallah H, Marie R. The uniformized power method for transient solutions of Markov processes. Computers and Operations Research 1993;20(5):515–26.

[8] Malhotra M, Muppala JK, Trivedi K. Stiffness-tolerant methods for transient analysis of stiff Markov chains. Microelectronic and Reliability 1994;34(11):1825–41.

[9] Grassmann WK. Transient Solutions in Markovian Queueing Systems. Computers and Operations Research 1977;4:47–56.

[10] Abdallah H, Marie R, Sericola B. Point and expected interval availability analysis with stationarity detection. Technical Report, IRISA, Campus de Beaulieu, Rennes, France, 1996;1014.

[11] Clarotti CA. The Markov approach to calculating system reliability: computational problems. International School of Physics Enrico Fermi 1986;55–66.