Interval-Availability Distribution of 2-State Systems with Exponential Failures and Phase-Type Repairs

B. Sericola
IRISA-INRIA, Rennes

Key Words — **Repairable computer system, Cumulative operation time, Interval availability, Markov process, Uniformization technique**

Reader Aids —
General purpose: Present a new method
Special math needed for explanations: Probability, Markov processes
Special math needed to use results: Same
Results useful to: Reliability analysts, designers of fault-tolerant computers

Summary & Conclusions — **Interval availability is a dependability measure defined as the fraction of time during which a system is in operation over a finite observation period. Usually, for computing systems, the models used to evaluate interval availability distribution are Markov models. Numerous papers using these models have been published, and only complex numerical methods have been proposed as solutions to this problem even in simple cases such as the 2-state Markov model. This paper proposes a new way to compute this distribution when the model is a 2-state semi-Markov process in which the holding times have an exponential distribution for the operational state and a phase-type distribution for the non-operational one.**

The main contribution of this paper is to define a new algorithm to compute the interval availability distribution for systems having only one operational state. The computational complexity depends weakly on the number of states of the system, and sometimes it can deal also with infinite state spaces. Moreover, simple closed expressions of this distribution are shown when repair periods are of the Erlang type with eventually absorbing states.

1. INTRODUCTION

Interval availability is important, especially for dependable computer systems. The papers on this topic give complex numerical solutions even for simple cases, *eg*, for a 2-state Markov model. The problem for a general Markov model is described by a linear hyperbolic system of partial differential equations in [1], and it is solved by explicit finite-difference methods in [2]. A *uniformization*[1] method that bounds the errors caused by truncation of an infinite series during the computation was proposed in [3]; this method was developed further in [4] to obtain a closed-form expression. Another technique [5] is based on numerical inversion of Laplace transforms.

This paper proposes a new algorithm (IAD-SU)[2] to compute the interval availability distribution for a 2-state semi-Markov model in which failures have an exponential distribution and repairs have a phase-type distribution. This measure can be interpreted as the fraction of time during the interval $(0,t)$, spent by a Markov process in its initial state. IAD-SU is derived from the work in [4], which is reviewed in section 2. Section 3 applies IAD-SU to the 2-state semi-Markov model. Section 4 considers particular cases of phase-type repairs such as exponential & Erlang repair. Section 5 gives 2 applications of IAD-SU: 1) A critical system with n components fails if any component fails. 2) The classical M/M/1 queueing system for which we compute the fraction of time in which the server is busy (system workload) during a given time-interval. Application #2 is interesting since the state space of the system is infinite.

Acronyms

IAD interval availability distribution
IAD-SU interval availability distribution — Sericola *uniformization* (algorithm).

Notation

X continuous-time homogeneous Markov process
X_t state of X at time t
E finite state space of X
α initial probability distribution of X
A infinitesimal generator of X
ν *uniformization* rate of X
P transition probability matrix of the *uniformized* Markov chain associated with X
B, L [subset, number] of operational (up) states
B^c subset of non-operational (down) states
α_B, α_{B^c} subvectors of α associated with partition $\{B, B^c\}$ of E
P_B, P_{BB^c}, P_{B^cB}, P_{B^c} submatrices of P associated with partition $\{B,B^c\}$ of E
P_j P_B, for $j=1$; $P_{BB^c}\cdot P_{B^c}^{j-2}\cdot P_{B^cB}$, for $j > 1$.
$\mathcal{I}(\cdot)$ $\mathcal{I}(\text{True}) \equiv 1$, $\mathcal{I}(\text{False}) \equiv 0$
$(0,t)$ interval of time
$[a;b]$ set of integers $\{a,a+1,\ldots,b\}$
$O(t)$ cumulative amount of operational time during $(0,t)$, a r.v.
$\text{IAV}(t)$ interval availability over $(0,t)$. a r.v.
$\mathbf{1}_B(n)$ column vector of 1's; dimension is $(n+1)\cdot L$
$N_B(n)$ number of visits to the states of B during the first n transitions of the *uniformized* Markov chain associated with X; a r.v.

[1]Appendix A.5 briefly explains *uniformization*.

[2]*Editors' note*: We have assigned this acronym IAD-SU (interval availability distribution — Sericola uniformization) for simple, clear, unique reference to the concept.

0018-9529/94/$4.00 ©1994 IEEE

$\beta(n)$ $(n+1)\cdot L$ row vector
$H(n)$ square $[(n+1)\cdot L]\times[(n+1)\cdot L]$ matrix
C, N integers used in truncation for *uniformization*, $0 < C \le N$
$\boldsymbol{l}$ $\{l_1,\ldots,l_m\}$ — $\boldsymbol{l}$ is usually constrained
$\Phi_{m,k}(\boldsymbol{l})$ $k!\cdot\left(\prod_{j=1}^{m} P_j^{l_j}\right) \Big/ \left(\prod_{j=1}^{m} l_j!\right)$
$\theta_{1:m}(\boldsymbol{l})$ $\sum_{j=1}^{m} l_j$
$\theta_{2:m}(\boldsymbol{l})$ $\sum_{j=1}^{m} j\cdot l_j$.

Other, standard notation is given in "Information for Readers & Authors" at the rear of each issue. All proofs are in the appendix.

2. INTERVAL AVAILABILITY DISTRIBUTION

Consider a continuous-time homogeneous Markov process, $X = \{X_t,\ t\ge 0\}$, over a finite state space E.

$$O(t) \equiv \int_0^t \mathcal{I}(X_u\in B)\ du$$

$$\text{IAV}(t) \equiv O(t)/t$$

The infinitesimal generator A of X verifies $A(i,i) = -\Sigma_{j\neq i} A(i,j)$. The transition probability matrix of the *uniformized* Markov chain associated with X [6] verifies:

$$P = I + A/\nu$$

$$\nu \ge \max(-A(i,i),\ i \in E).$$

Decompose P & α with respect to $\{B,B^c\}$.

$$P = \begin{pmatrix} P_B & P_{BB^c} \\ P_{B^cB} & P_{B^c} \end{pmatrix}$$

$$\alpha = (\alpha_B,\ \alpha_{B^c})$$

The main result in [4] is $\text{Cdf}\{O(t)\}$, $(0 \le s < t)$:

$$\Pr\{O(t) \le s\} = 1 - \sum_{n=0}^{+\infty} \text{poim}(n;\nu\cdot t)\cdot \sum_{k=0}^{n} \text{binm}(k;s/t,n) \cdot\beta(n)\cdot H(n)^k\cdot \mathbf{1}_B(n)$$

$$\beta(n) \equiv (\alpha_B,\ \alpha_{B^c}\cdot P_{B^cB},\ \alpha_{B^c}\cdot P_{B^c}\cdot P_{B^cB},\ \ldots,\ \alpha_{B^c}\cdot P_{B^c}^{n-1} \cdot P_{B^cB}),\ n > 0$$

$$\beta(0) = \alpha_B$$

$$H(n) \equiv \begin{pmatrix} 0 & P_1 & P_2 & P_3 & P_4 & \ldots & P_{n-1} & P_n \\ 0 & 0 & P_1 & P_2 & P_3 & \ldots & P_{n-2} & P_{n-1} \\ 0 & 0 & 0 & P_1 & P_2 & \ldots & P_{n-3} & P_{n-2} \\ 0 & 0 & 0 & 0 & P_1 & \ldots & P_{n-4} & P_{n-3} \\ \vdots & & & & & \ddots & & \vdots \\ 0 & 0 & 0 & 0 & 0 & \ldots & P_1 & P_2 \\ 0 & 0 & 0 & 0 & 0 & \ldots & 0 & P_1 \\ 0 & 0 & 0 & 0 & 0 & \ldots & 0 & 0 \end{pmatrix}$$

$$H(0) = 0$$

This theoretical result is used for a numerical algorithm. The $\text{Sf}\{N_B(n)\}$ is [4]:

$$\Pr\{N_B(n) > k\} = \beta(n)\cdot H(n)^k\cdot \mathbf{1}_B(n),\ \text{for } 0 \le k \le n.$$

3. A 2-STATE SEMI-MARKOV MODEL

Assumptions

1. The operational (up) state has exponential holding times.
2. The non-operational (down) state has phase-type holding times. ◄

Nomenclature

Entry: component (of a vector). ◄

This structure is equivalent to the Markov process depicted in section 2 with only 1 operational state, *viz*, with subset B reduced to 1 state. The formula for $\text{Cdf}\{O(t)\}$ can be simplified since the P_j, $j \ge 1$, are now reduced to real numbers, verifying $0 \le P_j \le 1$.

3.1 Derivation of Simpler Expression for $\beta(n)\cdot H(n)^k\cdot \mathbf{1}_B(n)$

For a fixed $n \ge 0$ and $0 \le k \le n$,

$$x_{n,k} \equiv H(n)^k\cdot \mathbf{1}_B(n). \tag{3-1}$$

For convenience, the first entry is denoted by $x_{n,k}(0)$ and its last entry by $x_{n,k}(n)$. To simplify the notation, let:

$$x_{n,k}(m) \equiv 0,\ \text{for } m > n.$$

Since $H(0) = 0$ and $H(0)^0 = 1$, we have as first values (in the $x_{n,k}$ sequence):

$$x_{0,0}(0) = 1;$$

$$x_{1,0}(i) = 1,\ i=0,1;$$

$x_{1,1}(0) = P_1$, $x_{1,1}(1) = 0$.

Then,

$$x_{n,k+1} = H(n) \cdot x_{n,k}$$

$$x_{n,k+1}(i) = \sum_{j=1}^{n} P_j \cdot x_{n,k}(j+i), \text{ for } 0 \le i \le n. \qquad (3\text{-}2)$$

Theorem 3.1 is the main result of this paper.

Theorem 3.1. For all n, $n \ge 0$; for all k, $0 \le k \le n$; for all i, $0 \le i \le n$:

$$x_{n,k}(i) = \sum_{\Omega 1} \Phi_{n,k}(l), \qquad (3\text{-}3)$$

$\Omega 1 \equiv \{l_1, l_2, \ldots, l_n \in [0;k] \,|\, \theta_{1:n}(l) = k,\ i + \theta_{2:n}(l) \le n\}$.◄

Theorem 3.1 implies the relation $(s < t)$:

$$\Pr\{O(t) \le s\} = 1 - \sum_{n=0}^{\infty} \text{poim}(n;\nu \cdot t) \cdot \sum_{k=0}^{n} \text{binm}(k;s/t,n) \cdot \sum_{i=0}^{n} \beta(i) \cdot x_{n,k}(i). \qquad (3\text{-}4)$$

In practice, the initial system-state is operational ($\alpha_B = 1$); thus $\beta = (1,0,\ldots,0)$.

Eq (3-4) reduces to:

$$\Pr\{O(t) \le s\} = 1 - \sum_{n=0}^{\infty} \text{poim}(n;\nu \cdot t) \cdot \sum_{k=0}^{n} \text{binm}(k;s/t,n) \cdot \sum_{\Omega 2} \Phi_{n,k}(l),$$

$\Omega 2 \equiv \{l_1, \ldots, l_n \in [0;k] \,|\, \theta_{1:n}(l) = k,\ \theta_{2:n}(l) \le n\}$.

The IAD $(0 \le u < 1)$ is:

$$\Pr\{\text{IAV}(t) \le u\} = 1 - \sum_{n-0}^{\infty} \text{poim}(n;\nu \cdot t) \cdot \sum_{k=0}^{n} \text{binm}(k;u,n) \cdot \sum_{\Omega 2} \Phi_{n,k}(l); \qquad (3\text{-}5)$$

$$\sum_{\Omega 2} \Phi_{n,k}(l) = x_{n,k}(0);\ x_{0,k}(0) = 1.$$

Notation

$y_{n,k}$ $\quad x_{n,k}(0)$.

(This notation simplifies the following presentation.)

To compute the Cdf$\{\text{IAV}(t)\}$, evaluate the $y_{n,k}$ for $n=0,\ldots,N$ and $k=0,\ldots,n$ where N, the truncation step of the infinite series, is chosen such that for a given error tolerance ϵ', the remainder of the series $e'(N)$ verifies:

$$e'(N) = \sum_{n=N+1}^{\infty} \text{poim}(n;\nu \cdot t) \cdot \sum_{k=0}^{n} \text{binm}(k;u,n) \cdot y_{n,k} \le \text{poifc}(N+1;\ \nu \cdot t).$$

The integer N is chosen such that poifc$(N+1;\nu \cdot t) \le \epsilon'$. Thus,

$$-\epsilon' \le \Pr\{\text{IAV}(t) \le u\} - \left[1 - \sum_{n=0}^{N} \text{poim}(n;\nu\ t) \cdot \sum_{k=0}^{n} \text{binm}(k;u,n) \cdot y_{n,k}\right] \le 0. \qquad (3\text{-}6)$$

Corollary 3.1 provides a recursion to compute the $y_{n,k}$.

Corollary 3.1

a. $y_{n,0} = 1$, for $n \ge 0$;

b. $y_{n,k} = \sum_{j=1}^{n-k+1} P_j \cdot y_{n-j,k-1}$, for $n \ge 1$, $1 \le k \le n$. ◄

Figure 1 illustrates the recursion to compute the real numbers $y_{n,k}$ in corollary 3.1; for instance, $y_{6,4} = P_1 \cdot y_{5,3} + P_2 \cdot y_{4,3} + P_3 \cdot y_{3,3}$. The $y_{n,k}$ is represented as the (n,k)-entry of a (N+1)-dimensioned lower triangular matrix. The C is introduced to perform another truncation over index k; see lemma 3.1.

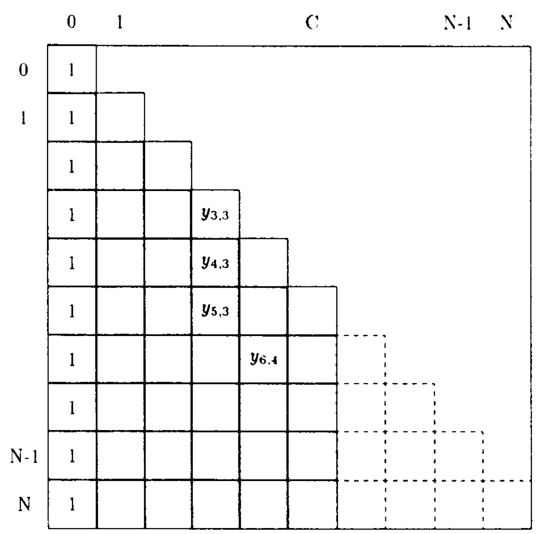


Figure 1. $y_{n,k} = P_1 \cdot y_{n-1,k-1} + P_2 \cdot y_{n-2,k-1} + \ldots + P_{n-k+1} \cdot y_{k-1,k-1}$

Lemma 3.1 induces an order relation between the real numbers $y_{n,k}$; this order relation determines C (see figure 1) for which a truncation over index k is feasible.

Lemma 3.1. For every $n \geq 0$ and $0 \leq k \leq n$, —

c. $y_{n,k+1} \leq y_{n,k}$;

d. $y_{n,k} \leq y_{n+1,k}$. ◄

Lemma 3.1 helps to show that when k becomes large enough ($k > C$), the values of $y_{n,k}$ can become very small. Formally —

$$\sum_{n=0}^{N} \text{poim}(n;\nu\cdot t)\cdot \sum_{k=0}^{n} \text{binm}(k;u,n)\cdot y_{n,k}$$

$$= \sum_{k=0}^{N} \sum_{n=k}^{N} \text{poim}(n;\nu\cdot t)\cdot\text{binm}(k;u,n)\cdot y_{n,k}$$

$$= \sum_{k=0}^{C} \sum_{n=k}^{N} \text{poim}(n;\nu\cdot t)\cdot\text{binm}(k;u,n)\cdot y_{n,k} + e''(N,C). \tag{3-7}$$

$$e''(N,C) \equiv \sum_{k=C+1}^{N} \sum_{n=k}^{N} \text{poim}(n;\nu\cdot t)\cdot\text{binm}(k;u,n)\cdot y_{n,k}$$

$$= \sum_{n=C+1}^{N} \text{poim}(n;\nu\cdot t)\cdot \sum_{k=C+1}^{n} \text{binm}(k;u,n)\cdot y_{n,k}$$

$$\leq \sum_{n=C+1}^{N} \text{poim}(n;\nu\cdot t)\cdot y_{n,C+1} \text{ — by lemma 3.1-c}$$

$$\leq y_{N,C+1} \text{ — by lemma 3.1-d}$$

That is, when computing the $y_{n,k}$, we try to find a C such that for a given error tolerance ϵ'', we have

$$y_{N,C+1} \leq \epsilon''. \tag{3-8}$$

The computation is made column by column as shown in figure 1. For each column k, compute $y_{N,k}$, using corollary 3.1-b, and test its value with respect to ϵ''. If $y_{N,k} \leq \epsilon''$, then take $C=k-1$; else compute the other elements of column k, that is $y_{N-1,k}, y_{N-2,k}, \ldots y_{k,k}$, and restart by computing $y_{N,k+1}$.

If such a C does not exist, then $C=N$ and $e''(N,C)=0$; the global error is ϵ'.

Using this last truncation, compute,

$$1 - \sum_{k=0}^{C} \sum_{n=k}^{N} \text{poim}(n;\nu\cdot t)\cdot\text{binm}(k;u,n)\cdot y_{n,k}. \tag{3-9}$$

If ϵ becomes the global error tolerance ($\epsilon = \epsilon' + \epsilon''$), then from (3-6) - (3-8),

$$-\epsilon \leq \Pr\{\text{IAV}(t) \leq u\} - \left[1 - \sum_{k=0}^{C} \sum_{n=k}^{N} \text{poim}(n;\nu\cdot t)\cdot\text{binm}(k;u,n)\cdot y_{n,k}\right] \leq 0.$$

IAD-SU for computing N & C is similar to the method in [3]. The main advantage of IAD-SU is that it stores only scalars ($y_{n,k}$). The algorithm in [3] requires storage of N vectors of dimension 'cardinality of the state space of the Markov process', even if the number of operational states is reduced to 1.

IAD-SU does require computing the P_j, $j=1,\ldots,N$; however, this can be done recursively in the following way. Recall that:

$$P_1 = P_B, \text{ and } P_j = P_{BB^c}\cdot P_{B^c}^{j-2}\cdot P_{B^cB} \text{ for } j \geq 2.$$

Define the row vectors:

$$Q_j \equiv P_{BB^c}\cdot P_{B^c}^{j-2}, \text{ for } j \geq 2. \text{ Then,}$$

$$Q_{j+1} = Q_j\cdot P_{B^c};$$

so only 1 supplementary vector is needed to store the successive values of Q_j.

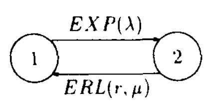
4. ERLANG PHASE-TYPE REPAIR

This section considers 2 phase-type repairs for which a simple closed expression for IAD can be obtained using (3-5).

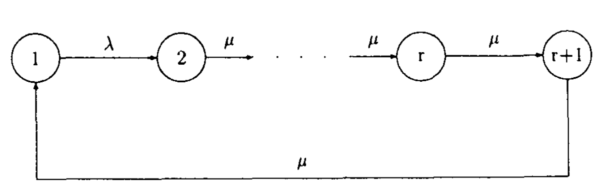
4.1 Irreducible Case

Assumptions

1. The model is a 2-state semi-Markov process.
2. The holding times in state 1 follow an exponential law with rate λ.
3. The holding times in state 2 follow an Erlang law with r stages and parameter μ.
4. The system starts in state 1 (the unique operational state). It then reaches state 2 after a failure, comes back to state 1 after repair, *etc.*
5. $\lambda \leq \mu$.



This semi-Markov process is equivalent to the following Markov process.



Assumption

Notation

r	number of Erlang stages
k^*	$\text{gilb}[(n-k)/r]$
k^{**}	$\text{gilb}[n/(r+1)]$.

Apply (3-5); choose $\nu=\mu$ which leads to:

$$P_1 = 1 - \lambda/\mu,\ P_2 = \ldots = P_r = 0,\ P_{r+1} = \lambda/\mu;\ P_j = 0,$$

$$\text{for } j \geq r+2.$$

This gives, if $p \equiv P_1$ and $q \equiv 1 - p$,

$$y_{n,k} = \sum_{\Omega A} (k!/(i!\cdot j!))\cdot p^i \cdot q^j$$

$$\Omega A \equiv \{i,j \in [0;k] \mid i+j = k,\ i + (r+1)\cdot j \leq n\}$$

or,

$$y_{n,k} = \sum_{\Omega B} \text{binm}(j;\ q,k)$$

$$\Omega B \equiv \{j \in [0;k] \mid k + r\cdot j \leq n\}$$

or,

$$y_{n,k} = \text{binf}(\min(k^*,k);\ q,k)$$

For fixed values of $r \geq 1$ and $n \geq 0$ —

$$\min(k^*,\ k) = \begin{cases} k, & \text{for } 0 \leq k \leq k^{**} \\ k^*, & \text{for } k^{**} < k \leq n \end{cases}$$

This leads to the closed expression:

$$\Pr\{\text{IAV}(t) \leq u\} = 1 - \sum_{n=0}^{\infty} \text{poim}(n;\mu\cdot t).\text{binf}(k^{**};\ u,n)$$

$$- \sum_{n=1}^{\infty} \text{poim}(n;\mu\cdot t)\cdot \sum_{k=k^{**}+1}^{n} \text{binm}(k;\ u,n)\cdot \text{binf}(k^*;\ q,k) \quad \text{(4-1)}$$

The exponential repair case $(r=1)$ reduces to:

$$\Pr\{\text{IAV}(t) \leq u\} = 1 - \sum_{n=0}^{\infty} \text{poim}(n;\mu\cdot t)$$

$$\cdot \text{binf}(\text{gilb}(n/2);\ u,n).$$

$$- \sum_{n=1}^{\infty} \text{poim}(n;\mu\cdot t)\cdot \sum_{k=\text{gilb}(n/2)+1}^{n} \text{binm}(k;\ u,n)$$

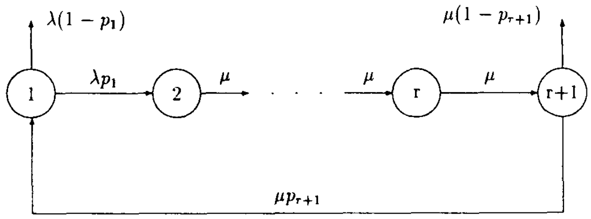
$$\cdot \text{binf}(n-k;\ q,k).$$

4.2 Absorbing Case

Assumptions

1. The system has 3 states.
2. One state is absorbing, such that it can be completely down either after an operational period (with probability, $1-p_1$) or after an unsuccessful repair period (with probability, $1-p_{r+1}$).
3. $\lambda \leq \mu$. ◄

We then obtain the following Markov process in which the two *up* arrows (without destination) are to the absorbing state.



State 1 (initial state) is the unique operational state. Apply (3-5). We choose $\nu = \mu$ and have:

$$P_1 = 1 - \lambda/\mu,\ P_2 = \ldots = P_r = 0,\ P_{r+1} = \lambda\cdot p_1$$

$$\cdot p_{r+1}/\mu;\ P_j = 0 \text{ for every } j \geq r+2.$$

Notation

p	P_1
q	P_{r+1}.

We obtain (4-1) even though q does not have the same value; here $p+q \neq 1$. If $p_1 = p_{r+1} = 1$, we obtain (4-1).

4.3 Discussion

In these two examples, the computation can be performed simply by truncating the infinite series as in (3-6).

5. APPLICATIONS

5.1 A Critical System

Assumptions

1. A hardware system has n components, and is 1-out-of-n:G (*series*).
2. Component failures are mutually s-independent.
3. Repair times are s-independent of component lives.
4. Maintenance policy is unrestricted, *ie*, the number of repairmen available is equal to the number of system components.
5. $\lambda_i \leq \mu_i$.
6. $\lambda_i = i/1000$ hours; $\mu_i = \mu = 1$/hour. ◄

Notation

n number of components in the system
i component index, $i = 1, \ldots, n$
x_i $\mathcal{I}$(component i is up)
x $(x_1,\ldots,x_n)$: binary vector
A_i transition rate matrix for component i
λ_i, μ_i [failure, repair] rate of component i
I_m m-dimensional identity matrix
Π_ρ, Σ_ρ implies the [product, sum] over ρ from 1 to n
$\mathfrak{N}$ set of non-negative integers.

These assumptions lead to a Markov model in which the number of system states be $M = 2^n$. Any system state can be represented by x. The only operational state is $(1,\ldots,1)$. The transition rate matrix $A(n)$ of the system can be easily generated using Kronecker algebra as follows.

$$A_i = \begin{bmatrix} -\lambda_i & \lambda_i \\ \mu_i & -\mu_i \end{bmatrix}$$

$$A(1) \equiv A_1,$$

$$A(n) = A(n-1) \oplus A_n =$$

$$\begin{bmatrix} A(n-1) - \lambda_n \cdot I_{n-1} & \lambda_n \cdot I_{n-1} \\ \mu_n \cdot I_{n-1} & A(n-1) - \mu_n \cdot I_{n-1} \end{bmatrix},$$

for $n \geq 2$.

The *uniformization* rate is:

$$\nu = \sum_\rho \mu(\rho).$$

All the repair rate's being equal does not simplify the IAD computation. Figure 2 shows $\Pr\{\text{IAV}(t) > 0.9\}$ *vs* time for several values of n.

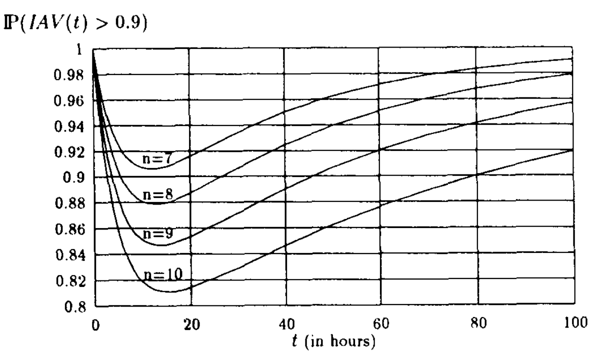


Figure 2. $\Pr\{\text{IAV}(t) > 0.9\}$ *vs* Time [$n = 7(1)10$].

All these values have been computed using $\epsilon = 10^{-5}$.

$$\text{IAV}(\infty) = \prod_\rho \mu_\rho/(\lambda_\rho + \mu_\rho) = 1/\prod_\rho (1 + \lambda_\rho),$$

$\text{IAV}(\infty) > 0.9$, for $n \leq 10$.

Thus, for $n \leq 10$, $\Pr\{\text{IAV}(t) > 0.9\} \to 1$ as $t \to \infty$.

System reliability is:

$$R(t) = Pr\{\text{IAV}(t) = 1\} = \exp(-\sum_\rho \lambda_\rho \cdot t). \quad (5\text{-}1)$$

For example, for $t > 60$ hours, figure 2 shows that such a system with $n=7$ components, is available at least 90% of the time with probability > 0.97. In contrast, $R(t)$ of the 7-component system < 0.19.

If $n=10$ and $t > 100$ hours, the system is available at least 90% of the time with probability $= 0.92$; while the reliability < 0.005.

5.2 Busy Fraction Time of the Server for M/M/1 Queue

Assumptions

1. The queueing system is M/M/1 with arrival rate λ and service rate μ.
2. State i, $i \in \mathfrak{N}$, of the system represents the number of customers waiting, including the one being served.
3. The initial state is 0. ◄

Notation

BPS(t) busy percent of server during $(0,t)$; a r.v.
IPS(t) percent of time during $(0,t)$ that the server is idle.

The non-zero entries of the infinitesimal generator A of the corresponding Markov process $X = \{X_t, t \geq 0\}$ are:

$$A(0,0) = -\lambda$$

$$A(i,i-1) = \mu, A(i,i) = -(\lambda+\mu), A(i,i+1) = \lambda, \text{ for } i \geq 1.$$

$$\text{BPS}(t) = (1/t) \cdot \int_0^t \mathcal{I}(X_s \geq 1)\, ds.$$

BPS(t) is also called 'system workload'.

$$\text{IPS}(t) = (1/t) \cdot \int_0^t \mathcal{I}(X_s = 0)\, ds.$$

$$\text{BPS}(t) + \text{IPS}(t) = 1.$$

$$\Pr\{\text{BPS}(t) > x\} = \Pr\{\text{IPS}(t) \leq 1-x\}, \text{ for } 0 < x < 1.$$

The transition probability matrix P of the *uniformized* Markov chain associated with X with respect to the rate $\nu = \lambda + \mu$ verifies:

$P = I + A/(\lambda + \mu)$.

Let,

$B \equiv \{0\}$, $B^c = \{i \in \mathfrak{N} | i \geq 1\}$,

$p \equiv \lambda/(\lambda + \mu)$, $q = 1-p$.

Then,

$P_B = q$, $P_{BB^c} = (p, 0, 0, \dots)$,

$P_{B^cB} = (q, 0, 0, \dots)$, and

the non-zero entries of matrix P_{Bc} are:

$P_{Bc}(i,i-1) = q$, and $P_{Bc}(i,i+1) = p$, for $i \geq 1$.

The Cdf{IPS(t)} or Sf{BPS(t)} is given by (3-8) with an error less than ϵ:

$$\Pr\{\text{BPS}(t) > x\} = \Pr\{\text{IPS}(t) \leq 1-x\}$$

$$\approx 1 - \sum_{k=0}^{C} \sum_{n=k}^{N} \text{poim}(n; (\lambda+\mu)\cdot t)\cdot\text{binm}(k; 1-x, n)\cdot y_{n,k};$$

the N & C are as in section 3. The values of $y_{n,k}$ are (for $n \geq 0$):

$$y_{n,k} = \begin{cases} 1, \text{ for } k = 0 \\ \sum_{j=1}^{n-k+1} P_j \cdot y_{n-j,k-1}, \text{ for } 1 \leq k \leq n. \end{cases}$$

So, we need only the values of P_j to compute Sf{BPS(t)}. These values are given by lemma 5.1.

Lemma 5.1. $P_1 = q$, and for all $j \geq 1$

$$P_{2j} = \binom{2(j-1)}{j-1}\cdot(p\cdot q)^j / j,$$

$$P_{2j+1} = 0.$$ ◄

The P_j can be easily computed recursively. Figure 3 shows the probability that the server is occupied for at least 95% of the time, as a function of the λ ($0 \leq \lambda \leq 2$) and t ($0 \leq t \leq 100$); the service rate $\mu = 1.0$.

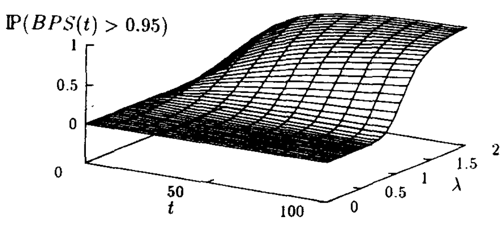


Figure 3. Pr{BPS(t) > 95%} *vs* λ & t

It is well-known that the steady-state workload of the M/M/1 queueing system is:

$$\text{BPS}(\infty) = \begin{cases} \lambda/\mu, \text{ for } \lambda < \mu \\ 1, \text{ otherwise} \end{cases}$$

Figure 3 shows this limiting behavior. For example, for $\lambda = 1$,

Pr{workload > 95%} = 0.28, for $t \in (0, 100)$,

Pr{workload > 95%} = 1, for $t \in (0, \infty)$.

APPENDIX

A.1 Proof of Theorem 3.1

For $n=0$, the result is trivial.

The proof can be made by induction on integer k, for fixed-integer $n \geq 1$.

For $k=0$, we obtain $x_{n,0}(i) = 1$ for every i, $0 \leq i \leq n$ which is in accord with (3-1).

For $k=1$, we obtain for $0 \leq i \leq n-1$,

$$x_{n,1}(i) = \sum_{j=1}^{n-i} P_j, \quad \text{(A-a)}$$

$$x_{n,1}(n) = 0.$$

Alternatively, (3-2) gives,

$$x_{n,1}(i) = \sum_{j=1}^{n} P_j \cdot x_{n,0}(j+i). \quad \text{(A-b)}$$

Eq (A-a) & (A-b) are the same using the convention that $x_{n,k}(m) \equiv 0$ for every $m > n$.

Let the result be true for integers $0, 1, \dots, k < n$; then compute $x_{n,k+1}(i)$ using (3-2) for every i, $0 \leq i \leq n$:

$$x_{n,k+1}(i) = \sum_{h=1}^{n} P_h \, x_{n,k}(h+i)$$

$$= \sum_{h=1}^{n} P_h \cdot \sum_{\Omega 3} \Phi_{n,k}(l)$$

$$= \sum_{h=1}^{n} \sum_{\Omega 3} P_h \cdot \Phi_{n,k}(l); \quad \text{(A-1)}$$

$$\Omega 3 \equiv \{l_1, l_2, \dots, l_n \in [0;k] | \theta_{1:n}(l) = k, \; i+h+\theta_{2:n}(l) \leq n\}.$$

In the $\sum_{\Omega 3}$ sum, change $l_h \rightarrow l_h + 1$.

$$\mathrm{x}_{\mathrm{n,k+1}}(\mathrm{i}) = \sum_{\mathrm{h=1}}^{\mathrm{n}} \sum_{\Omega 4} l_h \cdot \Phi_{n,k}(l);$$

$$\Omega 4 \equiv \{l_1,\ldots,l_{h-1},l_{h+1},\ldots,l_n \in [0;k] | l_h \in [1;k+1],$$

$$\theta_{1:n}(l) = k+1,\ i+\theta_{2:n}\ (l) \le n\}.$$

The l_h can start with 0 since all the corresponding terms will be 0; thus,

$$x_{n,k+1}(i) = \sum_{h=1}^{n} \sum_{\Omega 5} l_h \cdot \Phi_{n,k}(l);$$

$$\Omega 5 \equiv \{l_1,\ldots,l_{h-1},l_{h+1},\ldots,l_n \in [0;k] | l_h \in [0;k+1],$$

$$\theta_{1:n}(l) = k+1,\ i+\theta_{2:n}(l) \le n\}.$$

The r.h.s of the previous equation can be decomposed into two terms: 1) in which $l_h \in [0;k]$, and 2) in which $l_h = k+1$. In term 2, $l_h = k+1$ implies that all the other l_i are 0. Therefore,

$$x_{n,k+1}(i) = \sum_{h=1}^{n} \sum_{\Omega 6} l_h \cdot \Phi_{n,k}(l) + \sum_{h=1}^{n} \sum_{\Omega 7} l_h \cdot \Phi_{n,k}(l);$$

$$\Omega 6 \equiv \{l_1,\ldots,l_n \in [0;k] | \theta_{1:n}(l) = k+1,\ i+\theta_{2:n}(l) \le n\};$$

$$\Omega 7 \equiv \{l_1,\ldots,l_n \in [0;k] | \mathrm{l_p} = 0, \text{ for } p \ne h,\ l_h = k+1,$$

$$i+\theta_{2:n}(l) \le n,\}$$

In the first term, replace $\theta_{1:n}(l)$ by $k+1$; in the second term replace l_h by $k+1$; then,

$$x_{n,k+1}(i) = \sum_{\Omega 6} \Phi_{n,k+1}(l) + \sum_{h=1}^{n} \sum_{\Omega 7} \Phi_{n,k+1}(l);$$

The relation between disjoint sets is:

$$\{l_1,\ldots,l_\mathrm{n} \in [0;\mathrm{k}+1] | \theta_{1:n}(l) = k+1,\ i+\theta_{2:n}(l) \le n\}$$

$$= \{l_1,\ldots,l_n \in [0;k] | \theta_{1:n}(l) = k+1,\ i+\theta_{2:n}(l) \le n\}$$

$$\cup(\cup_{h=1}^{n}\{l_1,\ldots,l_n \in [0;k] | l_p = 0 \text{ for } p \ne h,$$

$$l_h = k+1,\ i+\theta_{2:n}(l) \le n\}).$$

Thus, from $k+1 \to k$, we obtain (3-3). *Q.E.D.*

A.2 Proof of Corollary 3.1

Relation #a is easily deduced from theorem 3.1.

Relation #b can be proved using (A-1), which can be written for integers $n \ge 1$, $1 < k \le n$ and $i=0$ as:

$$y_{n,k} = \sum_{h=1}^{n} P_h \cdot \sum_{\Omega 8} \Phi_{n,k-1}(l), \tag{A-2}$$

$$\Omega 8 \equiv \{l_1,l_2,\ldots,l_n \in [0;k-1] | \theta_{1:n}(l) = k-1,$$

$$\theta_{2:n}(l) \le n-h\}.$$

The conditions $\{\theta_{1:n}(l) = k-1\}$ and $\{\theta_{2:n}(l) \le n-h\}$ imply that $k-1 \le n-h$, *ie*, $h \le \mathrm{n}-\mathrm{k}+1$. So, if $h > n-k+1$, then the terms in the $\Sigma_{\Omega 8}$ sum of (A-2) are all 0. Therefore,

$$y_{n,k} = \sum_{h=1}^{n-k+1} P_h \cdot \sum_{\Omega 8} \Phi_{n,k-1}(l). \tag{A-3}$$

In the second sum of (A-3), since the integer $n-h+1$ verifies that $1 \le n-h+1 \le n$, then the condition,

$\{\theta_{2:n}(l) \le n-h\}$ implies that,

$$l_{n-h+1} = \ldots = l_n = 0.$$

Eq (A-3) then becomes,

$$y_{n,k} = \sum_{h=1}^{n-k+1} P_h \cdot \sum_{\Omega 8} \Phi_{n-h,k-1}(l).$$

Using theorem 3.1, we obtain,

$$y_{n,k} = \sum_{h=1}^{n-k+1} P_h \cdot y_{n-h,k-1}. \qquad Q.E.D.$$

A.3 Proof of Lemma 3.1

Since $y_{n,k} = \Pr\{N_B(n) > k\}$, then lemma 3.1-c is evident.

By definition of $N_B(n)$, we have $N_B(n) \le N_B(n+1)$. It follows that $N_B(n) > k$ implies $N_B(n+1) > k$; thus,

$$y_{n,k} = \Pr\{N_B(n) > k\} \le \Pr\{N_B(n+1) > k\} = y_{n+1,k}.$$

This proves lemma 3,1-d. *Q.E.D.*

A.4 Proof of Lemma 5.1

It is clear that $P_1 = q$ since $P_1 = P_B$.

For $j \ge 2$,

$$P_j \equiv P_{BB^c} \cdot P_{B^c}^{j-2} \cdot P_{B^cB},$$

which can be interpreted as,

$P_j = p \cdot \Pr\{$reaching state 0 after exactly $j-2$ transitions into $B^c | X_0 = 1\}$.

This last probability is clearly 0 when j is odd. It follows that for all $j \ge 1$, $P_{2j+1} = 0$. Furthermore, for every $j \ge 1$,

$P_{2j} = p \cdot \Pr\{$reaching state 0 after exactly $2(j-1)$ transitions into $B^c | X_0 = 1\}$

which is also

$$P_{2j} = p \cdot \Pr\{\text{'number of customers served in a busy period'} = j\}.$$

It is well known [7] that Pr{serving j customers in a busy period} is:

$$\binom{2(j-1)}{j-1} \cdot p^{j-1} \cdot q^{j}/j.$$

It follows, therefore, that, for every $j \geq 1$,

$$P_{2j} = \binom{2(j-1)}{j-1} \cdot (p \cdot q)^{j}/\, j. \qquad \textit{Q.E.D.}$$

A.5 Explanation of Uniformization

When studying the transient behavior of a Markov process (continuous time Markov chain), the solution to the Chappman forward/backward differential equations follows a matrix exponential, $\exp(A \cdot t)$, yielding the "transition functions" — analogous to the 1-step transition matrix for discrete-time chains. Generally, computation of the transition functions must be approached numerically, *eg*, eigen-analysis to compute $\exp(A \cdot t)$. However, it is possible to trade a complicated Markov process for one of simpler structure but of the same probability law. This simpler process is such that the subordinate point process (times between jumps) is Poisson (instead of the complicated non-renewal subordinate point process of the original continuous chain — an amazing result) and thus is independent of the imbedded (discrete) Markov chain governing state transitions.

Uniformization is the well-known technique for creating this simpler Markov process. An advantage in numerical computations is sometimes gained by appealing to the properties of Poisson processes and the straightforward computations required to study the transient behavior of the (discrete) imbedded chain.

REFERENCES

[1] A. Goyal, A. N. Tantawi, K. S. Trivedi, "A measure of guaranteed availability", *IBM Research Report RC 11341*, 1985; IBM T.J. Watson Research Center.

[2] A. Goyal, A. N. Tantawi, "A measure of guaranteed availability and its numerical evaluation", *IEEE Trans. Computers*, vol 37, 1988 Jan, pp 25-32.

[3] E. de Souza e Silva, H. R. Gail, "Calculating cumulative operational time distributions of repairable computer systems", *IEEE Trans. Computers*, vol C-35, 1986 Apr, pp 322-332.

[4] B. Sericola, "Closed form solution for the distribution of the total time spent in a subset of states of a homogeneous Markov process during a finite observation period", *J. Applied Probability*, vol 27, 1990 Sep, pp 713-719.

[5] V.G. Kulkarni, V.F. Nicola, R.M. Smith, K.S. Trivedi, "Numerical evaluation of performability and job completion time in repairable fault-tolerant systems", *Proc. IEEE 16th Fault-Tolerant Computing Symp*, 1986 Jul; Vienna, Austria.

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

[7] L. Kleinrock, *Queueing Systems*, vol 1, 1975; John Wiley & Sons.

AUTHOR

Dr. Bruno Sericola; IRISA-INRIA; Campus de Beaulieu; 35042 Rennes Cedex; FRANCE.

Bruno Sericola was born in 1959. He received a PhD (1988) in Computer Science from the University of Rennes, France. He then worked on an ESPRIT project at INRIA and in 1989 he obtained a Research position at INRIA. His research activity includes computer-system performance modeling, dependability evaluation of fault-tolerant computing systems, and applied stochastic processes.

Manuscript TR91-090 received 1991 June 7; revised 1992 April 27; 1992 November 18.

IEEE Log Number 92-10653

◄TR►

CORRECTION 1992 JUNE ISSUE CORRECTION 1992 JUNE ISSUE CORRECTION 1992 JUNE ISSUE CORRECTION

A Decomposition Method for Optimization of Large-System Reliability

There is an error in [1: p 187, (26a)]; that equation should be:

$$\lambda_c = 2/[\pi G_1 \bullet [1 + (G^*/G_1)^2]], \qquad (26a)$$

$$G^* \equiv 1 - \textstyle\sum_i G_i \bullet \tan(\tfrac{1}{2}\pi R_i).$$

REFERENCES

[1] D. Li, Y. Y. Haimes, "A decomposition method for optimization of large-system reliability", *IEEE Trans. Reliability*, vol 41, 1992 Jun, pp 183-189.

Correction received 1994 January 28
Original IEEE Log Number 92-00843

◄TR►