Computers Math. Applic. Vol. 25, No. 4, pp. 105–111, 1993
Printed in Great Britain.

0097-4943/93 \$5.00 + 0.00

A QUEUEING MODEL FOR A NON-HOMOGENEOUS TERMINAL SYSTEM SUBJECT TO BR992

B. ALMASI AND J. SZTRIK
Lajos Kossuth University, Debrecen
P.O. Box 12, H-4010, Hungary

(Received May 1992)

Abstract—This paper deals with a non-homogeneous queueing model to describe the performance of a multi-terminal system subject to random breakdowns. All random variables involved here are independent and exponentially distributed. Although the stochastic process describing the system's behaviour is a Markov chain, the number of states becomes very large. The main contribution of this paper is a recursive computational approach (see (5)) to solve the steady-state equations concerning the problem. It further generalizes the homogeneous model treated in [1]. In equilibrium, the main performance characteristics of the system are obtained. Finally, some numerical results illustrate the problem in question.

1. INTRODUCTION

This paper deals with the analysis of a queueing system, which may be used as a model of a real life system, consisting of n terminals connected with a Central Processor Unit (CPU). The user at the terminal i has exponentially distributed think times with rate λ_i and generates jobs with processing time being exponentially distributed with rate μ_i. The service rule at the CPU is First-In, First-Out (FIFO). Let us suppose that the operational system is subject to random breakdowns stopping the service at the terminals and at the CPU. The failure-free operation times of the system are exponentially distributed random variables with rate α. The restoration times of the system are assumed to be exponentially distributed random variables with rate β. The busy terminals are also subject to random breakdowns not affecting the system's operation. The failure-free operation times of busy terminals are supposed to be exponentially distributed random variables with rate γ_i for the terminal i. The repair times of the terminal i are exponentially distributed random variables with rate τ_i. The breakdowns are serviced by a single repairman providing pre-emptive priority to the system's failure. We assume that each user generates only one job at a time, and he waits at the CPU before he starts thinking again, that is, the terminal is inactive while waiting at the CPU, and it can't break down. All random variables involved here are independent of each other.

As it can easily be seen, this model is a generalization of the classical 'machine interference problem' discussed, among others, in [2–4]. In recent years, finite-source models in different forms have been effectively used, for example, for mathematical description of a multiprogrammed computer system (see [1,5–8]).

This paper further generalizes the homogeneous model discussed in [1]. Using a similar computational approach as in [1], the steady-state equations are recursively solved. In equilibrium, the main performance of the system, such as the mean number of jobs residing at the CPU, the mean number of functional terminals, the expected response time of jobs, and utilizations are obtained. Finally, some numerical results illustrate the problem in question.

The authors are very grateful to Professor M. Arató for his helpful comments. Research is partially supported by Hungarian National Foundation for Scientific Research, under Grant OTKA 1648/91.

Typeset by $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX

2. THE MATHEMATICAL MODEL AND A COMPUTATIONAL APPROACH

Let us introduce the following random variables:

$$X(t) = \begin{cases} 1, & \text{if the operating system is failed at time } t, \\ 0, & \text{otherwise.} \end{cases}$$

$$\begin{aligned} Y(t) &= \text{the number of failed terminals at time } t, \\ YI(t) &= \text{the failed terminals' indices at time } t \text{ in order of their failure,} \\ &\quad \text{or } 0 \text{ if } Y(t) = 0, \\ Z(t) &= \text{the number of jobs residing at the CPU at time } t, \\ ZI(t) &= \text{the indices of jobs staying at the CPU at time } t \text{ in order of the} \\ &\quad \text{generation of their request, or } 0 \text{ if } Z(t) = 0. \end{aligned}$$

It is easy to see that the process

$$M(t) = (X(t), Y(t), Z(t), YI(t), ZI(t))$$

is a five-dimensional Markov chain with state space

$$S = ((q;\ k;\ s;\ i_1, \dots, i_k;\ j_1, \dots, j_s), \quad q = 0, 1; \quad k = 0, \dots, n; \quad s = 0, \dots, n-k),$$

where

$(i_1, \dots, i_k)$ is a permutation of k objects from the numbers $1, \dots, n$ or 0, if $k = 0$,
$(j_1, \dots, j_s)$ is a permutation of s objects from the remaining $n - k$ numbers or 0, if $s = 0$.

The event $(q; k; s; i_1, \dots, i_k; j_1, \dots, j_s)$ denotes that the operating system is in state $X(t) = q$, there are k failed terminals with indices $i_1, \dots, i_k$, and there are s jobs with indices $j_1, \dots, j_s$ at the CPU.

It can easily be seen that the dimension of the state space is

$$\dim(S) = 2 \sum_{k=0}^{n} \frac{(k+1)n!}{(n-k)!}.$$

Let us denote the steady-state distribution of $(M(t), t \geq 0)$ by

$$\begin{aligned} p(q; i_1 \dots i_k; j_1, \dots, j_s) &= \\ &= \lim_{t \to \infty} p(X(t) = q; Y(t) = k; YI(t) = i_1, \dots, i_k; Z(t) = s; ZI(t) = j_1, \dots, j_s), \end{aligned}$$

which exists and is unique (see [2–4]) if all the rates are positive. As usual, using the notion of probability flow, the global balance equations for $p(q; i_1 \dots i_k; j_1, \dots, j_s)$ are as follows

$$\begin{aligned} &\Big(\alpha + \tau_{i_1} + \mu_{j_1} + \sum_{\substack{r \neq i_1, \dots, i_k \\ r \neq j_1, \dots, j_s}} (\lambda_r + \gamma_r)\Big) p(0; i_1, \dots, i_k; j_1, \dots, j_s) \\ &= \beta\, p(1; i_1, \dots, i_k; j_1, \dots, j_s) \\ &+ \sum_{\substack{r \neq i_1, \dots, i_k \\ r \neq j_1, \dots, j_s}} \Big(\tau_r\, p(0; r, i_1, \dots, i_k; j_1, \dots, j_s) + \sum_{\substack{r \neq i_1, \dots, i_k \\ r \neq j_1, \dots, j_s}} \mu_r\, p(0; i_1, \dots, i_k; r, j_1, \dots, j_s)\Big) \qquad (1) \\ &+ \gamma_{i_k}\, p(0; i_1, \dots, i_{k-1}; j_1, \dots, j_s) + \lambda_{j_s}\, p(0; i_1, \dots, i_k; j_1, \dots, j_{s-1}), \\ &\text{for all } i_1, \dots, i_k;\ j_1, \dots, j_s;\ k = 0, \dots, n;\ s = 0, \dots, n-k, \\ &\beta\, p(1; i_1, \dots, i_k; j_1, \dots, j_s) = \alpha\, p(0; i_1, \dots, i_k; j_1, \dots, j_s), \qquad (2) \\ &\text{for all } i_1, \dots, i_k;\ j_1, \dots, j_s;\ k = 0, \dots, n;\ s = 0, \dots, n-k, \end{aligned}$$

where the probabilities of meaningless events and coefficients are defined to be zero.

For $k = 0$ and $s = 0$ i_k (and j_s) are not defined, so, e.g., $\gamma_{i_k}\, p(0; i_1, \ldots, i_{k-1}; j_1, \ldots, j_s)$ has no meaning, so it is defined to be zero. We have:

$$\left(\alpha + \sum_{i=1}^{n}(\lambda_i + \gamma_i)\right) p(0;0;0) = \beta p(1;0;0) + \sum_{i=1}^{n} \mu_i\, p(0;0;i) + \sum_{i=1}^{n} \tau_i\, p(0;i;0).$$

This system will be simpler if we substitute equation (2) to equation (1). Namely, we have

$$\begin{aligned}
&(\tau_{i_1} + \mu_{j_1} + \sum_{\substack{r\neq i_1,\ldots,i_k \\ r\neq j_1,\ldots,j_s}} (\lambda_r + \gamma_r))\, p(0; i_1, \ldots, i_k; j_1, \ldots, j_s) \\
&= \sum_{\substack{r\neq i_1,\ldots,i_k \\ r\neq j_1,\ldots,j_s}} (\tau_r\, p(0; r, i_1, \ldots, i_k; j_1, \ldots, j_s) + \sum_{\substack{r\neq i_1,\ldots,i_k \\ r\neq j_1,\ldots,j_s}} \mu_r\, p(0; i_1, \ldots, i_k; r, j_1, \ldots, j_s)) \qquad (3) \\
&+ \gamma_{i_k}\, p(0; i_1, \ldots, i_{k-1}; j_1, \ldots, j_s) + \lambda_{j_s}\, p(0; i_1, \ldots, i_k; j_1, \ldots, j_{s-1}), \\
&\text{for all } i_1, \ldots, i_k;\ j_1, \ldots, j_s;\ k = 0, \ldots, n;\ s = 0, \ldots, n-k, \\
&\beta p(1; i_1, \ldots, i_k; j_1, \ldots, j_s) = \alpha\, p(0; i_1, \ldots, i_k; j_1, \ldots, j_s), \qquad (4) \\
&\text{for all } i_1, \ldots, i_k;\ j_1, \ldots, j_s;\ k = 0, \ldots, n;\ s = 0, \ldots, n-k.
\end{aligned}$$

Our aim is to solve this system subject to the normalization condition

$$\sum_{q=0}^{1} \sum_{k=0}^{n} \sum_{s=0}^{n-k} p(q, k, s) = 1,$$

where

$$p(q, k, s) = \sum_{(i_1,\ldots,i_k)\in V_n^k} \quad \sum_{(j_1,\ldots,j_s)\in V_{n-k}^s} p(q; i_1, \ldots, i_k; j_1, \ldots, j_s),$$

V_n^k : The set of all $(i_1, \ldots, i_k)$ (as defined above),

V_{n-k}^s : The set of all $(j_1, \ldots, j_s)$ (as defined above).

In principle, this system of linear equations can easily be solved by standard computational methods. However, we must take into consideration that the unknowns are probabilities, and therefore, in the case of large state space, the round-off errors may have considerable effect on them (see [4,7,9,10]). In the following, an efficient recursive computational approach is given for determining the steady-state probabilities.

3. THE RECURSIVE SOLUTION

Let $\underline{Y}(m)$ denote the vector of the stationary probabilities for the states where the operating system is working. There are k failed terminals, and $l = m - k$ job is waiting at the CPU

$((k = 0, \ldots, m),\ m = 0, \ldots, n)$. That is,

$$\underline{Y}(m) = \begin{pmatrix} p(0;1,\ldots,m-1,m;0) \\ p(0;1,\ldots,m-1,m+1;0) \\ \vdots \\ p(0;n,\ldots,n-m+1;0) \\ p(0;1,\ldots,m-1;m) \\ p(0;1,\ldots,m-1;m+1) \\ \vdots \\ p(0;n,\ldots,n-m+2;n-m+1) \\ \vdots \\ \vdots \\ p(0;0;n,\ldots,n-m+1) \end{pmatrix}.$$

In other words, the elements of $\underline{Y}(m)$ are ordered as follows

(1) For $k = m$ and $l = 0$, the stationary probabilities are in the increasing order of indices.
(2) For $k = m - 1$ and $l = 1$, the stationary probabilities are in the increasing order of indices.

$\vdots$

$(m+1)$ For $k = 0$ and $l = m$, the stationary probabilities are in the increasing order of indices.

Similarly, let $\underline{Z}(m)$ denote the vector of stationary probabilities alike $\underline{Y}(m)$, but for the states, where the operating system is failed. From the definition, it can be seen easily that the dimension of $\underline{Y}(m)$ and $\underline{Z}(m)$ is $(m+1)n!/(n-m)!$.

Using these notations equations (3) and (4) can be written in matrix form as

$$\underline{Y}(0) = C(0)\,\underline{Y}(1), \tag{i}$$
$$\underline{Y}(j) = C(j)\,\underline{Y}(j+1) + D(j)\,\underline{Y}(j-1), \qquad j = 1, \ldots, n-1, \tag{ii}$$
$$\underline{Y}(n) = D(n)\,\underline{Y}(n-1), \tag{iii}$$
$$\underline{Z}(j) = F(j)\,\underline{Y}(j), \qquad j = 0, \ldots, n. \tag{iv}$$

The dimension of the matrices are $d(j) = (j+1)n!/(n-j)!$:

$$\begin{aligned} F(j) &: d(j) \times d(j), \\ C(j) &: d(j) \times d(j+1), \\ D(j) &: d(j) \times d(j-1). \end{aligned}$$

The elements of the matrices can be obtained from the equations (3) and (4). Now we have the following theorem.

THEOREM. *The solution of the equations (i)–(iv) is*

$$\begin{aligned} \underline{Y}(j) &= L(j)\,\underline{Y}(j-1), \quad j = 1, \ldots, n, \\ \underline{Z}(j) &= F(j)\,\underline{Y}(j), \qquad j = 0, \ldots, n, \end{aligned} \tag{5}$$

where $L(n) = D(n)$, $L(j) = (I - C(j)\,L(j+1))^{-1} D(j), \qquad j = 1, \ldots, n-1,$

so the system of equations can be solved uniquely up to a multiplicative constant, which can be obtained from the normalization condition.

PROOF.

$$\text{From (iv) we get } \underline{Z}(j) = F(j)\,\underline{Y}(j), \qquad j = 0,\dots,n.$$
$$\text{By using (iii) we obtain } \underline{Y}(n) = D(n)\,Y(n-1),$$
$$\text{which yields } \underline{Y}(n) = L(n)\,Y(n-1).$$

Assuming that $\underline{Y}(j+1) = L(j+1)\underline{Y}(j)$, from (ii) we have

$$\underline{Y}(j) = C(j)\,L(j+1)\,\underline{Y}(j) + D(j)\,\underline{Y}(j-1).$$

By simple calculation, we obtain that

$$\begin{aligned}
&(I - C(j)\,L(j+1))\,\underline{Y}(j) = D(j)\,\underline{Y}(j-1),\\
&\underline{Y}(j) = (I - C(j)\,L(j+1))^{-1} D(j)\,\underline{Y}(j-1),\\
&\underline{Y}(j) = L(j)\,Y(j-1).
\end{aligned}$$

Starting the recursion with any initial value, we can calculate—without normalization—the $p'(q; i_1,\dots,i_k; j_1,\dots,j_s)$ elements of $\underline{Y}'(m)$, $\underline{Z}'(m)\,(m = 0,\dots,n)$. The steady-state probabilities can be obtained from $\underline{Y}'(m)$, $\underline{Z}'(m)\,(m = 0,\dots,n)$, by using the normalization condition as follows

$$\underline{Y}(m) = \frac{\underline{Y}'(0)}{\sum_{q=0}^{1}\sum_{k=0}^{n}\sum_{s=0}^{n-k}\sum_{i_1,\dots,i_k\in V_n^k}\sum_{j_1,\dots,j_s\in V_{n-k}^s} p'(q; i_1,\dots,i_k; j_1,\dots,j_s)}\,\underline{Y}'(m),$$

$$\underline{Z}(m) = \frac{\underline{Y}'(0)}{\sum_{q=0}^{1}\sum_{k=0}^{n}\sum_{s=0}^{n-k}\sum_{i_1,\dots,i_k\in V_n^k}\sum_{j_1,\dots,j_s\in V_{n-k}^s} p'(q; i_1,\dots,i_k; j_1,\dots,j_s)}\,\underline{Z}'(m),$$

$m = 0,\dots,n.$

4. PERFORMANCE MEASURES

Let us introduce the following notation: $\delta(i,j) = \begin{cases} 1, & \text{if } i = j,\\ 0, & \text{otherwise.}\end{cases}$

The steady-state characteristics:

(These characteristics will be calculated in Tables 1–3.)

(i) Mean number of jobs residing at the CPU:

$$\overline{n}_j = \sum_{i=0}^{1}\sum_{k=0}^{n}\sum_{s=0}^{n-k} s\,p(i,k,s).$$

(ii) Mean number of functional terminals:

$$\overline{n}_g = n - \sum_{i=0}^{1}\sum_{k=0}^{n}\sum_{s=0}^{n-k} k\,p(i,k,s).$$

(iii) Mean number of busy terminals:

$$\overline{n}_b = \sum_{k=0}^{n}\sum_{s=0}^{n-k} (n-k-s)\,p(0,k,s).$$

(iv) Utilization of repairman:

$$U_r = \sum_{k=0}^{n} \sum_{s=0}^{n-k} p(1,k,s) + \sum_{k=1}^{n} \sum_{s=0}^{n-k} p(0,k,s).$$

(v) Utilization of CPU:

$$U_{CPU} = \sum_{k=0}^{n-1} \sum_{s=1}^{n-k} p(0,k,s).$$

(vi) Utilization of terminal i $/i = 1,\dots,n/$:

$$U_i = \sum_{k=0}^{n} \sum_{s=0}^{n-k} \sum_{r=1}^{k} \sum_{v=1}^{s} (1 - \delta(i,i_r) - \delta(i,j_v))\, p(0;i_1,\dots,i_k;j_1,\dots,j_s).$$

(vii) Expected response time of jobs for terminal i $/i = 1,\dots,n/$:

$$T_i = \frac{\sum_{q=0}^{1} \sum_{s=0}^{n-k} \sum_{r=1}^{s} \delta(i,j_r)\, p(q;i_1,\dots,i_k;j_1,\dots,j_s)}{\lambda_i U_i}.$$

5. NUMERICAL RESULTS

The algorithm generating these characteristics was implemented in FORTRAN'77 on an IBM PC/AT at the Institute of Mathematics, University of Debrecen, Hungary. The following properties of the involved matrices has greatly improved the calculations.

(a) Each row of the matrix $D(k)$ $/k = 1,\dots,n/$ has maximum two non-zero elements, so it is useful to store the non-zero elements and the column indices of these elements. It is true for the matrix $L(n)$ too, since $L(n) = D(n)$.

(b) The matrix $F(k)$ $/k = 0,\dots,n/$ contains non-zero elements only in the main diagonal, and these values are the same constant: α/β.

(c) The matrix equation (i) can be used to test the solution, because it is not used by the algorithm.

Examples

CASE 1. Failure-free system (See [11, p. 123]).

$$n = 4 \quad \alpha = 0.0001 \quad \beta = 9999.0$$
$$\overline{n}_j = 2.186 \quad \overline{n}_g = 4.0 \quad U_{CPU} = 0.903$$

Table 1.

i	λ_i	μ_i	γ_i	τ_i	U_i	T_i
1	0.500	0.900	0.0001	9999.0	0.383	3.229
2	0.400	0.700	0.0001	9999.0	0.416	3.506
3	0.300	0.600	0.0001	9999.0	0.469	3.771
4	0.200	0.500	0.0001	9999.0	0.546	4.155

This case can be used to test the results and to approximate a failure-free system described in [11]. The difference between these results and the ones in [11] is less than 0.01 for all calculated characteristics.

CASE 2. Terminal failure.

$$n = 4 \quad \alpha = 0.0001 \quad \beta = 9999.0$$
$$\overline{n}_j = 1.876 \quad \overline{n}_g = 2.645 \quad U_{CPU} = 0.63$$

Table 2.

i	λ_i	μ_i	γ_i	τ_i	U_i	T_i
1	0.500	0.900	0.3200	0.4500	0.291	2.196
2	0.400	0.700	0.1700	0.3400	0.364	2.289
3	0.300	0.600	0.2200	0.5000	0.375	2.574
4	0.200	0.500	0.1600	0.3000	0.427	2.866

In this example, we can see how terminal failures influence the performance measures. The response times and the number of good terminals are less than in Case 1. That is, the system works as if there were less terminals.

CASE 3. CPU failure.

$$n = 4 \quad \alpha = 0.25 \quad \beta = 0.45$$
$$\overline{n}_j = 2.186 \quad \overline{n}_g = 4.0 \quad U_{CPU} = 0.581$$

Table 3.

i	λ_i	μ_i	γ_i	τ_i	U_i	T_i
1	0.500	0.900	0.0001	9999.0	0.245	5.021
2	0.400	0.700	0.0001	9999.0	0.268	5.444
3	0.300	0.600	0.0001	9999.0	0.301	5.771
4	0.200	0.500	0.0001	9999.0	0.351	6.463

If we compare these results with Case 1, it can be seen that the failure of the CPU increases the response times and decreases the utilizations, as one can expect.

REFERENCES

1. J. Sztrik and T. Gál, A recursive solution of a queueing model for a multi-terminal system subject to breakdowns, *Performance Evaluation* **1**, 1–7 (1990).
2. W. Feller, *An Introduction to Probability Theory and its Applications*, Vol. I, Wiley-Chapman, New York, (1950).
3. R. Goodman, *Introduction to Stochastic Models*, Benjamin/Cummings, CA, (1988).
4. D. Gross and C.M. Harris, *Fundamentals of Queueing Theory*, 2nd ed., Wiley and Sons, New York, (1985).
5. E. Gelenbe and I. Mitrani, *Analysis and Synthesis of Computer Systems*, Academic Press, London, (1980).
6. L. Kleinrock, *Queueing Systems*, Vol. II, Computer Applications, Wiley-Interscience, New York, (1976).
7. S.S. Lavenberg, *Computer Performance Modeling Handbook*, (Edited by S.S. Lavenberg), Academic Press, New York, (1983).
8. J. Sztrik, A queueing model for multiprogrammed computer systems with I/O times, *Acta Cybernet.* **7**, 127–135 (1984).
9. P.M. Snyder and W.J. Stewart, Explicit and iterative approaches to solving queueing models, *Oper. Res.* **33**, 183–202 (1985).
10. H.C. Tijms, *Stochastical Modelling and Analysis, A Computational Approach*, Wiley and Sons, New York, (1986).
11. L. Csige and J. Tomkó, The machine interference for exponentially distributed operating and repair times, *Alk. Mat. Lapok* **8**, 107–124 (in Hungarian) (1982).