Performability Modeling of Non-homogeneous Terminal Systems Using MOSEL

B. Almási, * G. Bolch, J. Sztrik
University of Debrecen, Debrecen, Hungary
* University of Erlangen, Erlangen, Germany
{almasi|jsztrik}@math.klte.hu, bolch@informatik.uni-erlangen.de

Abstract

The aim of this paper is to investigate the effect of the different service disciplines, such as FIFO, PS, Priority Processor Sharing, Polling, on the main performance measures, such as utilizations, response times, throughput, mean queue length. Using MOSEL it has been shown by numerical examples that even in the case of homogeneous sources and homogeneous failure and repair times the CPU utilization depends on the scheduling discipline contrary to the case of reliable terminal systems. All random variables involved in the model construction are supposed to be exponentially distributed and independent of each other.

1 Introduction

Several works have been devoted to the investigation of the utilization factor of the Central Processor Unit (CPU), and the number of jobs staying at the CPU. It has turned out that in the case when the involved random variables are exponentially distributed, the request's generation rates are the same, the processing rates are different, Asztalos [5], Kameda [7], Lehtonen [8], Van der Wal [12] have proved that the utilization of CPU is not influenced at all by any work-conserving scheduling rule, including First-In-First-Out (FIFO), Processor Sharing (PS), Priority Processor Sharing (PPS), Preemptive or Nonpreemptive priority, Shortest and Longest-Expected- Processing-Time-First disciplines. More precisely, it has been shown that the mean busy period length of the processor is the same for any of the above mentioned schedulings. Furthermore, the mean number of jobs staying at the CPU is minimized by giving higher preemptive priority to a job with less mean job size (so-called H-schedule). Consequently the overall utilization of the system, the sum of CPU and terminal utilizations, sometimes called as effective degree of multiprogramming, is maximized. Based on this fact Kameda [7] has investigated more practical models of multiprogramming systems to estimate the maximum processing capacity of the system.

In the case when the request's generation rates are also different by using different methods Koole [9] and Van der Wal [12] have shown that if preemptions of the resume

[1]Acknowledgment: The authors are very grateful to Helmut Herold for his helpful comments. Research is partially supported by Hungarian Scientific Research Fund under grants , OTKA-T30685/99, T034280/2000 and German-Hungarian Bilateral Intergovernmental Scientific Cooperation, No. 21-2000.

type are allowed, the CPU utilization is maximized by giving higher priority to the jobs of the faster thinking terminals irrespectively of the expected job sizes. Results for the overall device utilizations have not been mentioned.

However, in practice we can see that the terminals and the CPU are not always available for service. These situations could be considered as breakdowns, so the analysis of non-reliable terminal systems seems to be also important.

2 Modeling non-reliable non-homogeneous terminal systems.

2.1 The mathematical model.

Let us consider a terminal system consisting of n terminals connected with a Central Processing Unit (CPU). The model is a closed queueing network with 2 (multiple server) service stations (nodes) and finite number n of jobs. The first service station is the processor, consisting of only one server (the CPU itself), where jobs from the terminals may suffer from queueing delay. We consider three service disciplines at the CPU: FIFO, Priority Processor Sharing (PPS), which includes the Processor Sharing (PS) discipline too (see [10]), and the Polling discipline (see [3]).

The second service station is a collection of n terminals (i.e. multiple server station). At the terminals there is no queueing delay for the jobs (as the number of terminals is equal to the number of jobs). A user at terminal i has thinking (i.e. program generation) times, and processing (i.e. program running) times depending on index i. We assume, that each user generates only one job at a time, and he waits until the CPU services it.

There is a third service station (node), named repairman. We assume, that the busy equipments (terminals, CPU) are subject to random breakdowns, so giving duties to the repairman. The random working and repair times of the terminals are exponentially distributed with mean depending on the terminal index (non-homogeneous breakdowns). Furthermore we assume, that the CPU is responsible for the system's work, i.e. the service stops at the terminals, and at the CPU, when the CPU is down. The repairman gives preemptive priority to the CPU failure, and follows FIFO discipline for the terminal breakdowns.

Let us denote by λ_i, μ_i, γ_i, τ_i, w_i the parameters of the exponentially distributed thinking, processing, operating, repair times and weight for terminal i, i=1,...,n, respectively. Similarly, let α, β denote the failure and repair rate of the CPU, respectively. The random variables are assumed to be independent of each other.

To deal with the problem we have to introduce the following random variables:

$$\begin{aligned}
X(t) &= \begin{cases} 1, & \text{if the operating system is failed at time } t, \\ 0, & \text{otherwise,} \end{cases} \\
Y(t) &= \text{the failed terminals' indices at time } t \text{ in order of their failure,} \\
& \text{ or 0 if there is no failed terminal,} \\
Z(t) &= \text{the indices of the jobs residing at the CPU at time } t, \\
& \text{ or 0 if the CPU is idle.}
\end{aligned}$$

Depending on the service discipline the random variable Z(t) gives the order of service by the CPU, too. It can easily be seen that the stochastic process $M(t) = (X(t), Y(t), Z(t))$ is a Markov chain having a rather complex, and large state space.

To get its the steady-state probabilities an efficient recursive computational method has been introduced and used for different service rules mentioned earlier, c.f. [1], [11]. In the present case MOSEL has been used to obtain these probabilities [6].

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

$$p(q; i_1 \dots i_k; j_1, \dots, j_s) = \\ = \lim_{t\to\infty} p(X(t) = q; Y(t) = i_1, \dots, i_k; Z(t) = j_1, \dots, j_s)$$

Furthermore, let us denote by $p(q,k,s)$ the steady-state probability that the operating system is in state q, k terminals are failed and s jobs are at the CPU.

Knowing these probabilities the main performance measures can be obtained as follows:

(i) Mean number of jobs residing at the CPU

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

(ii) Mean number of good terminals

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

(iii) Average 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 the 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 the CPU

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

(vi) Utilization of terminal i, i=1,...,n

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

(vii) Expected response time of jobs for terminal i

$$T_i = \frac{Q_i}{\lambda_i U_i}$$

where $\delta(i,j) = \begin{cases} 1, & \text{if i=j,} \\ 0, & \text{otherwise,} \end{cases}$ and Q_i denotes the probability of staying at the CPU for terminal i, namely

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

2.2 Numerical results.

The results discussed in this section were introduced in [4], where the authors proved by numerical examples, that the utilization of the CPU depends on the service discipline (in the case of homogeneous sources), contrary to the reliable systems (see [5], [7]).

In Almási [4] the numerical result were obtained by a particular recurrence relations for solving the envolved steady-state equations. The advances of using MOSEL is in formulation of the problem in a very short and compact way.

In the following table we can see, that the MOSEL implementation confirms the results of [4], by producing the same results.

Input parameters

$n = 4$	$\alpha = 0.001$	$\beta = 999.0$

i	λ_i	μ_i	γ_i	τ_i	w_i
1	0.3500	0.4000	0.2000	0.3000	3.0
2	0.3500	0.8500	0.2000	0.3000	90.0
3	0.3500	0.5000	0.2000	0.3000	15.0
4	0.3500	0.9000	0.2000	0.3000	190.0

Performance measures

	FIFO	PS	POLLING	PPS
n_j	1.283976	1.230658	1.285014	1.137364
U_r	0.754239	0.767961	0.754007	0.791032
U_{CPU}	0.663056	0.660519	0.663111	0.655889
U_1	0.268280	0.249977	0.268550	0.211706
U_2	0.292608	0.313970	0.292608	0.345383
U_3	0.276354	0.269498	0.276542	0.268845
U_4	0.294113	0.318494	0.293307	0.360611
T_1	3.785136	4.397992	3.776645	6.074148
T_2	2.909238	2.322270	2.912437	1.566298
T_3	3.475490	3.650338	3.469850	3.563455
T_4	2.860429	2.210037	2.882578	1.288600

Table 1.

2.3 Summary of performance issues obtained from numerical examples

The calculations using MOSEL showed again that the utilization of the CPU depends on the service discipline (in the case of homogeneous sources), contrary to the reliable systems (see [5], [7]).

3 Conclusion

MOSEL software package has been used for modeling non-reliable non-homogeneous terminal systems. The advanced features of the tool made it possible to formulate the problem in a short and compact form. By giving counter examples it has been shown that contrary to the reliable systems the utilization of the CPU depends on the service discipline.

References

[1] Almási, B: A Queuing Model for a Processor-Shared Multi-Terminal System Subject to Breakdowns, *Acta Cybernetica* Vol. 10, Nr. 4, 273-282, (1996).

[2] Almási, B: Response Time for Finite Heterogeneous Non-reliable Queueing Systems, *Computers and Mathematics with Applications* Vol. 31, No. 11, 55-59, (1996).

[3] Almási, B.: A Queueing Model for a Non-homogeneous Polling System Subject to Breakdowns, *Annales Univ. Sci. Budapest. Sec. Comp.* Vol. 18, 11-23, (1999)

[4] Almási, B. and Sztrik, J.: Optimization problems on the performance of a non-reliable terminal system, *Computers and Mathematics with Applications* Vol. 38, 13-21, (1999).

[5] Asztalos, D.: Optimal control of finite-source priority queues with computer system applications, *Computers and Mathematics with Applications* 6, 425-431, (1980).

[6] Begain K., Bolch, G. and Herold, H.: *Practical Performance Modeling, Application of the MOSEL Language*, Kluwer Academic Publisher, Boston, 2001.

[7] Kameda, H.: A finite-source queue with different customers, *J. ACM* 29, 478-491, (1982).

[8] Lehtonen, T.: On the Optimal Policies of an Exponential Machine Repair Problem, *Naval Research Logistics Quarterly* 31, 173-181, (1984).

[9] Koole G. and Vrijenhoek M.: Scheduling a Repairman in a Finite Source System, *Mathematical Methods of Operations Research* 44, 333-344, (1996).

[10] Sztrik, J.: A probability model for priority processor-shared multiprogrammed computer systems, *Acta Cybernetica* 7, 329–340, (1986).

[11] Sztrik, J. and Gál, T.: A recursive solution of a queueing model for a multi-terminal system subject to breakdowns. *Performance Evaluation* 11, 1–7, (1990).

[12] Van der Wal J.: The maximization of CP utilization in an exponential CP-terminal system with different think times and different job sizes *Stochastic Processes and their Applications* 18, 277-289, (1984)