Computers & Operations Research 40 (2013) 510–519

Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

Two-server parallel system with pure space sharing and Markovian arrivals

S.R. Chakravarthy [a], H.D. Karatza [b,*]

[a] Department of Industrial and Manufacturing Engineering, Kettering University, Flint, MI 48504, USA
[b] Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki 541 24, Greece

ARTICLE INFO

Available online 16 August 2012

Keywords:
Markovian arrival process
Parallel systems
Rigid job
Optimization
Algorithmic probability

ABSTRACT

We consider a parallel system with two identical servers and pure space sharing among rigid jobs. The parallel system is modeled as an MAP/M/2 queue with two types of jobs. While one type of jobs requires only one server, the other type needs both the servers before leaving the system. Using matrix–analytic methods, we analyze the queueing system in steady state. We report some interesting performance measures as well as illustrative examples to bring out the qualitative nature of the model under study.

1. Introduction and model description

In parallel and distributed systems, the scheduling of jobs on processors has been shown to be a critical factor in achieving efficient parallel execution. Users expect their individual jobs to achieve excellent performance. The main issue is how to share system resources among competing jobs in a way that satisfies the demands of jobs and produces a good overall performance. These objectives raise a number of scheduling policy issues with respect to workload type and resource heterogeneity.

Good scheduling policies can maximize system and individual application performance and avoid unnecessary delays. To evaluate the performance of these systems, researchers often apply simulation models to obtain results [8,9]. With simulation it is possible to simulate the system under study in detail. However, in many cases simulation models do not give insight into the exact way a factor affects the performance of the system. Hence, approaching the analysis of parallel and distributed systems via analytical modeling is preferred. This paper belongs to this category. It analyzes a parallel system model with two servers and a specific job type known as rigid. In addition, it is assumed that jobs in service share the servers according to pure space sharing.

An $M/M/2$ parallel system model with pure space sharing among rigid jobs has been studied analytically in [6]. Two types of jobs were considered and closed-form expressions for performance measures of interest of the parallel system were provided. Also, the authors in [6] validated their expressions via simulation. In this paper we model the parallel system as an $MAP/M/2$ queue. Using matrix–analytic methods, we analyze the queueing system in steady state.

In [7], the author considers a multi-server queueing system with Poisson arrivals and exponential services wherein the customers require a random number of servers. However, here the author assumes that the servers (all working on the same customer) are not released at the same time. However, in [3], the authors study a similar model considered in [7] (i.e., a multi-server queueing system with Poisson arrivals and exponential services such that the customers require a random number of servers) but assume that the servers end service concurrently. They employ a system point approach for obtaining the waiting time distribution for each customer type (a type is defined based on the number of servers required by a customer). For the case of a two-server system they derive explicit solutions. While the model in [7] is totally different from the one studied here due to the nature of how the servers are released, the model studied in this paper generalizes the one studied in [3] for the case of a two-server system using totally a different approach.

This model studied in this paper can be applied to dual-processor parallel computer systems which are very common computing platforms today (for example two processor PCs). The model can also be applied widely to manufacturing problems where a job may need two servers (e.g. machines or people) simultaneously.

Our parallel system model consists of two homogeneous servers that share a single unbounded queue. Jobs arrive at the system according to a Markovian arrival process (*MAP*), a versatile point process introduced by Neuts [13]. A brief description of this process is given below. The service times of the jobs are exponentially distributed with parameter μ. Each job needs a certain number of servers to start the execution. Specifically, a job may require one server with probability p_1 or two servers with probability $p_2 = 1 - p_1$, $0 \le p_1 \le 1$. Since the extreme cases, $p_1 = 0$

* Corresponding author. Tel.: +30 2310 997974.
E-mail addresses: schakrav@kettering.edu (S.R. Chakravarthy), karatza@csd.auth.gr (H.D. Karatza).

0305-0548/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cor.2012.08.002

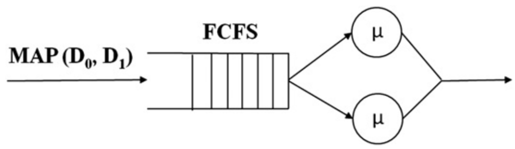

Fig. 1. Queueing model.

and $p_1 = 1$, reduce, respectively, to the classical *MAP/M/*1 and *MAP/M/*2 queues, our focus is primarily on p_1 inside the unit interval. The queuing discipline for all jobs is *FCFS*. A job at the head of the queue can start execution only if the number of servers it requires is available. Otherwise, it is blocked until all servers it needs are released. No other job behind a blocked one can start execution because of the *FCFS* scheduling policy. A pictorial description of the queueing model under study is displayed in Fig. 1.

The number of processors a job requires is fixed and is specified by the user at the submission time. It does not change during its execution. Furthermore, the jobs are *rigid* [1]. That is, they acquire all the processors they need at the same time and release them simultaneously soon after execution. We employ *pure space sharing* [1]. That is, we assume that each job is executed exclusively and no time sharing is employed.

Now we will briefly describe the versatile point process introduced by Neuts. A *MAP* is a tractable class of Markov renewal processes. It should be noted that by appropriately choosing the parameters of the *MAP* the underlying arrival process can be made as a renewal process. The *MAP* is a rich class of point processes that includes many well-known processes such as Poisson, PH-renewal processes, and Markov-modulated Poisson process. One of the most significant features of the *MAP* is the underlying Markovian structure and fits ideally in the context of matrix–analytic solutions to stochastic models. Matrix–analytic methods were first introduced and studied by Neuts [14]. As is well known, Poisson processes are the simplest and most tractable ones used extensively in stochastic modeling. The idea of the *MAP* is to significantly generalize the Poisson processes and still keep the tractability for modeling purposes. Furthermore, in many practical applications, notably in communications engineering, production and manufacturing engineering, the arrivals do not usually form a renewal process. So, *MAP* is a convenient tool to model both renewal and non-renewal arrivals. While *MAP* is defined for both discrete and continuous times, here we will need only the continuous time case.

The *MAP* in continuous time is described as follows. Let the underlying Markov chain be irreducible and let Q^* be the generator of this Markov chain. At the end of a sojourn time in state i, that is exponentially distributed with parameter λ_i, one of the following two events could occur: with probability $p_{ij}^{(1)}$ the transition corresponds to an arrival and the underlying Markov chain is in state j with $1 \le i,\ j \le m$; with probability $p_{ij}^{(0)}$ the transition corresponds to no arrival and the state of the Markov chain is j, $j \ne i$. Note that the Markov chain can go from state i to state i only through an arrival. Define matrices $D_0 = (d_{ij}^{(0)})$ and $D_1 = (d_{ij}^{(1)})$ such that $d_{ii}^{(0)} = -\lambda_i$, $1 \le i \le m$, $d_{ij}^{(0)} = \lambda_i p_{ij}^{(0)}$, for $j \ne i$ and $d_{ij}^{(1)} = \lambda_i p_{ij}^{(1)}$, $1 \le i,\ j \le m$. By assuming D_0 to be a nonsingular matrix, the interarrival times will be finite with probability one and the arrival process does not terminate. Hence, we see that D_0 is a stable matrix. The generator Q^* is then given by $Q^* = D_0 + D_1$.

Thus, D_0 governs the transitions corresponding to no arrival and D_1 governs those corresponding to an arrival. It can be shown that *MAP* is equivalent to Neuts' versatile Markovian point process. The point process described by the *MAP* is a special class of semi-Markov processes with transition probability matrix given by

$$\int_0^x e^{D_0 t}\, dt D_1 = [I - e^{D_0 x}](-D_0)^{-1} D_1, \quad x \ge 0. \tag{1}$$

For use in sequel, let $\boldsymbol{e}(r)$, $\boldsymbol{e}_j(r)$ and I_r denote, respectively, the (column) vector of dimension r consisting of 1's, column vector of dimension r with 1 in the jth position and 0 elsewhere, and an identity matrix of dimension r. When there is no need to emphasize the dimension of these vectors we will suppress the suffix. Thus, $\boldsymbol{e}$ will denote a column vector of 1's of appropriate dimension. The notation $\otimes$ will stand for the Kronecker product of two matrices. Thus, if A is a matrix of order $m \times n$ and if B is a matrix of order $p \times q$, then $A \otimes B$ will denote a matrix of order $mp \times nq$ whose (i,j)th block matrix is given by $a_{ij}B$. For more details on Kronecker products we refer the reader to [12].

Let $\boldsymbol{\eta}$ be the stationary probability vector of the Markov process with generator Q^*. That is, $\boldsymbol{\eta}$ is the unique (positive) probability vector satisfying

$$\boldsymbol{\eta} Q^* = \mathbf{0}, \quad \boldsymbol{\eta}\boldsymbol{e} = 1. \tag{2}$$

Let ξ be the initial probability vector of the underlying Markov chain governing the *MAP*. Then, by choosing ξ appropriately we can model the time origin to be (a) an arbitrary arrival point; (b) the end of an interval during which there are at least k arrivals; (c) the point at which the system is in specific state such as the busy period ends or busy period begins. The most interesting case is the one where we get the stationary version of the *MAP* by $\boldsymbol{\xi} = \boldsymbol{\eta}$. The constant $\lambda = \boldsymbol{\eta} D_1 \boldsymbol{e}$, referred to as the *fundamental rate* gives the expected number of arrivals per unit of time in the stationary version of the *MAP*.

Often, in model comparisons, it is convenient to select the time scale of the *MAP* so that λ has a certain value. That is accomplished, in the continuous *MAP* case, by multiplying the coefficient matrices D_0 and D_1, by the appropriate common constant. For further details on *MAP* and their usefulness in stochastic modeling, we refer to [11,15,16] and for a review and recent work on *MAP* we refer the reader to [2,4,5].

2. The steady-state analysis

In this section we will analyze the queueing model described in Section 1 in steady state. Let $N(t)$ $J_1(t)$,$J_2(t)$, and $J_3(t)$ denote, respectively, the number of customers in the system[1] (not including the ones identified in $J_1(t)$,$J_2(t)$, if any), the number of servers required by the oldest job in the system, the number of servers required by the second oldest job in the system, and the phase of the arrival process at time t. The process $\{(N(t),J_1(t),J_2(t),J_3(t)) : t \ge 0\}$ is a continuous-time Markov chain and the state space is given by

$$\begin{aligned}\Omega = &\{(r,j_3) : 0 \le r \le 2, 1 \le j_3 \le m\}\\ &\cup \{(i,j_1,j_2,j_3) : i \ge 0, 1 \le j_1,j_2 \le 2, 1 \le j_3 \le m\}.\end{aligned}$$

Note that

- The level $\hat{\mathbf{0}} = \{(0,k) : 1 \le k \le m\}$ denotes the system is idle and the arrival process is in phase k.
- The level $\hat{\mathbf{1}} = \{(1,k) : 1 \le k \le m\}$ denotes the system corresponding to only one server being busy and no job is waiting in the queue with the arrival process is in phase k.

[1] $N(t)$ cannot be taken as the number of jobs in the queue at time t as the second oldest job may still be waiting for service and is not accounted in this.