


Analysis of Centralized Splitting Fork-Join Queueing Systems with Heterogeneous Servers and Threshold Control Policy

Oleg Osipov^{} and Ekaterina Rogachko^{}

Saratov State University, 83 Astrakhanskaya Street, Saratov 410012, Russia
oleg.alex.osipov@gmail.com, rogachkoes@info.sgu.ru

Abstract. In this paper, we consider a Markovian centralized splitting fork-join queueing system with heterogeneous servers and an infinite capacity queue. Jobs arrive at the system according to a Poisson process, a job consists of multiple homogeneous tasks. Tasks of a job are served independently, when all the tasks associated with the job are finished, the job service is completed. The system operates under a threshold control policy. The control consists in sending tasks to one of idle servers or to the queue. The threshold policy uses the slow servers only when the queue length reaches certain threshold levels. We perform an exact analysis which is based on the matrix-geometric method and obtain expressions for the performance measures. Some numerical examples are presented. The results can be used for the performance analysis of multiprocessor systems and other modern distributed systems.

Keywords: Fork-Join Queueing Systems · Heterogeneous Servers · Threshold Control Policy · Matrix Geometric Method · Performance Measures · Distributed Computing System

1 Introduction

Fork-join queueing systems are widely used to model parallel and distributed systems. Some examples of these systems include RAID, distributed web applications, MapReduce, GRID, multipath routing networks, multiprocessor systems and etc. In these systems, arriving jobs are split for service by numerous servers and joined before departure.

In papers [1,2,3], fork-join queueing systems are applied to the analysis of MapReduce clusters and multipath routing. Results on modeling of RAID arrays and distributed databases are provided in [4,5]. Papers [6,7] describe an application of fork-join queueing systems to the analysis of multiprocessor systems and parallel algorithms. Fork-join queueing systems also arise in the performance analysis of automated manufacturing systems. In these systems, parallelism can be found in processes such as product assembly and logistic operations that involve multiple suppliers.

Paper [4] presents an overview of the main results for fork-join and related queueing models. Most of the papers consider parallel-server fork-join queueing systems [8,9,10,11,12].

The parallel-server fork-join queueing system consists of homogeneous servers [9,10,13] or heterogeneous servers [1,2,8,11,12,14]. An arriving job is split into a number of tasks, one for each server. Previous research in this area reports three basic types of parallel-server fork-join queueing systems [15], such as the centralized splitting model, the distributed splitting model, and the split-merge model.

In [11] upon arrival, a job of size S bringing tasks to $S \leq K$ of K servers is immediately split so that each of its S tasks is allocated to exactly one server. In [14] each job is split into a number of tasks, one for each free server. Paper [2] considers both deterministic and probabilistic scheduling strategies. Within probabilistic strategy, each task chooses a server with some probability. Instead, the present paper considers the allocation of tasks between the heterogeneous servers according to a threshold control policy. Under a threshold policy, a task is accepted at server i if and only if the number of tasks in queue is at or above a given threshold q_i before acceptance.

The threshold control policy is used as optimal solution for the job routing problem named the slow server problem [16,17]. This problem focuses on routing jobs to heterogeneous servers so as to minimize the average response time of jobs or to minimize the average number of jobs in the system. As it shown in [16], the optimal policy which minimizes the average response time of jobs in the system with two heterogeneous servers is of threshold type, i.e., the fast server should always be used (as long as there are jobs in the queue), and that the slower server should only be used if the number of jobs in queue exceeds a certain threshold value. The generalization of this rule was obtained in [17,18] for the case of multi-server queueing system with heterogeneous servers. The optimal control policy has the monotonicity and the threshold property, i.e., there exists a queue length threshold such that a new server (with the minimal service cost and with the maximal service rate) should be used only after the queue reaches this threshold. Later, the results were extended to retrial queueing systems with heterogeneous servers [19], systems with unreliable servers [20] and with heterogeneous servers in speed and in quality of resolution [21].

In this paper, we consider a centralized splitting fork-join queueing model [15] with heterogeneous servers and a threshold control policy. We apply the matrix-geometric approach to analyze the considered system. A similar approach was used in [10,14].

The remainder of the paper is organized as follows. Section 2 describes the fork-join queueing system under consideration. In Sections 3 and 4 we obtain the stationary distribution and the main performance measures. Section 5 provides an illustration of the above results, we also discuss some control policies and compare them. Finally, a section of conclusions commenting the main research contributions of this paper is presented.

2 The Model

We consider a system which consists of M heterogeneous servers S_i , $i = 1, \dots, M$, and a FCFS infinite capacity queue as shown in Fig. 1. Jobs arrive at the system according to a Poisson process with rate λ . Each job is split into d homogeneous tasks, $d \leq M$. An arriving task is placed in the queue. The service times at server S_i have exponential distribution with parameter μ_i , $i = 1, \dots, M$, and $\mu_1 > \mu_2 > \dots > \mu_M$.

Let b denote the number of waiting tasks in the queue (*queue length*). At the arrival times of new tasks the control consists in sending tasks from the queue to the *fastest idle server* or leaving them in the queue. A task from the front of the queue is assigned to a fastest idle server S_i , $i \in \{1, \dots, M\}$, if the queue length immediately after the arrival is not less than threshold level q_i , i.e., $q_i \leq b + 1$, ($1 = q_1 \leq q_2 \leq \dots \leq q_M < q_{M+1} = \infty$).

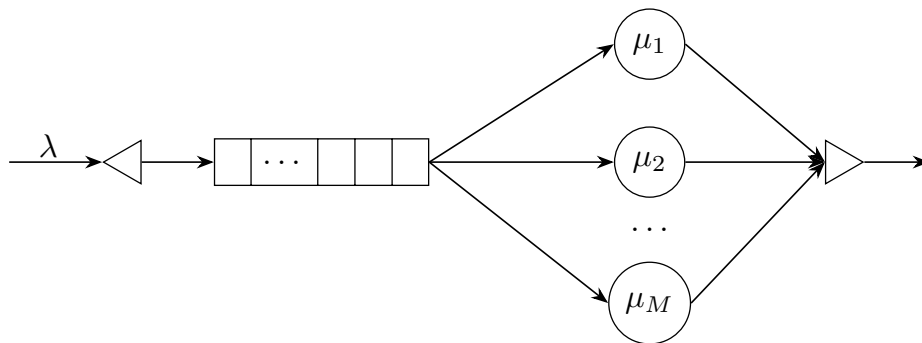


Fig. 1. Centralized splitting fork-join queueing system with heterogeneous servers.

When a task finishes its service at server S_i , a task from the queue will be assigned to server S_i if the queue length satisfies $b \geq q_i$. Otherwise server S_i will be idle. Tasks of a job are served independently, when all the tasks associated with the job are finished, the job service is completed.

3 The Stationary Distribution

The system state is described by a vector $\mathbf{x} = (r, n_0, n_1, \dots, n_M)$, where $r = r(b) = b \operatorname{div} d$ denotes the number of waiting jobs in the queue, $n_0 = n_0(b) = b \operatorname{mod} d$ is the number of tasks for the served job in the queue,

$$n_i = \begin{cases} 0, & \text{if server } S_i \text{ is idle,} \\ 1, & \text{if server } S_i \text{ is busy.} \end{cases}$$

The process $\{\mathbf{x}(t), t \geq 0\}$ is a $(M + 2)$ -dimensional continuous-time Markov chain on the state space \mathcal{X} .

For each state \mathbf{x} , we denote by $\mathcal{F}(\mathbf{x})$ and $\bar{\mathcal{F}}(\mathbf{x})$ the sets of idle and busy server indices, respectively,

$$\mathcal{F}(\mathbf{x}) = \{i > 0: n_i = 0\}, \quad \bar{\mathcal{F}}(\mathbf{x}) = \{i > 0: n_i = 1\}.$$

Denote by $\gamma(b)$ the mandatory (minimal) number of busy servers for queue length b . We can write

$$\gamma(b) = \begin{cases} 0, & b = 0, \\ M, & b \geq q_M, \\ \max\{i: q_i \leq b\}, & \text{otherwise.} \end{cases}$$

Define $\varphi(\mathbf{x})$ by

$$\varphi(\mathbf{x}) = \begin{cases} \text{the minimal element of } \mathcal{F}(\mathbf{x}), & \text{if } \mathcal{F}(\mathbf{x}) \neq \emptyset, \\ \infty, & \text{otherwise.} \end{cases}$$

If there are idle servers in the system, a task will be assigned to server $S_{\varphi(\mathbf{x})}$.

Let us define now the transition rate $q(\mathbf{x}, \mathbf{x}')$ from state $\mathbf{x} \in \mathcal{X}$ to state $\mathbf{x}' \in \mathcal{X}$. Each transition is associated with an event in the system, there are two types of events.

1. *A job arrives and splits into d tasks*
 - (a) if $b \geq q_M$,

$$q(\mathbf{x}, (r+1, n_0, n_1, \dots, n_M)) = \lambda, \quad (1)$$

- (b) if $b < q_M$,

$$q(\mathbf{x}, \mathbf{x}') = \lambda, \quad (2)$$

where $\mathbf{x}' = \mathbf{x}^{(d)}$ can be obtained from the following recurrence relation

$$\mathbf{x}^{(i+1)} = \begin{cases} (r(b^{(i)} + 1), n_0(b^{(i)} + 1), n_1^{(i)}, \dots, n_M^{(i)}), & \varphi(\mathbf{x}^{(i)}) > \gamma(b^{(i)} + 1), \\ \mathbf{x}^{(i)} + \mathbf{e}_{2+\varphi(\mathbf{x}^{(i)})}, & \text{otherwise,} \end{cases}$$

with the initial value $\mathbf{x}^{(0)} = \mathbf{x}$, $i = 0, \dots, d-1$.

Here, $\mathbf{x}^{(i)} = (r^{(i)}, n_0^{(i)}, \dots, n_M^{(i)})$ and $b^{(i)} = dr^{(i)} + n_0^{(i)}$; \mathbf{e}_j denotes the vector of the appropriate dimension with all components equal to 0, except the j th, which is 1.

2. *A task finishes its service at server S_i ($n_i = 1$)*
 - (a) if $b < q_i$,

$$q(\mathbf{x}, (r, n_0, n_1, \dots, n_{i-1}, 0, n_{i+1}, \dots, n_M)) = \mu_i, \quad (3)$$

- (b) if $n_0 > 0$, $\{i \in \bar{\mathcal{F}}(\mathbf{x}) : b \geq q_i\} \neq \emptyset$,

$$q(\mathbf{x}, (r, n_0 - 1, n_1, \dots, n_M)) = \sum_{i \in \bar{\mathcal{F}}(\mathbf{x}): b \geq q_i} \mu_i, \quad (4)$$

(c) if $r > 0$, $n_0 = 0$, $\{i \in \bar{\mathcal{F}}(\mathbf{x}) : b \geq q_i\} \neq \emptyset$,

$$q(\mathbf{x}, (r-1, d-1, n_1, \dots, n_M)) = \sum_{i \in \bar{\mathcal{F}}(\mathbf{x}): b \geq q_i} \mu_i. \quad (5)$$

Let state space \mathcal{X} be lexicographically ordered. The subset \mathcal{X}_l is called level l , $l = 0, 1, \dots$, and defined as

$$\mathcal{X}_l = \{(r, n_0, n_1, \dots, n_M) \in \mathcal{X} : r = l\}.$$

The cardinality L of \mathcal{X}_0 is

$$L = 2^M + (q_2 - q_1)2^{M-1} + (q_3 - q_2)2^{M-2} + \dots + (d - q_{\sigma_0})2^{M-\sigma_0},$$

where $\sigma_0 = \gamma(d-1)$.

Let \bar{r} be the minimal number $r \in \{1, 2, \dots\}$ such that $rd \geq q_M$. The cardinalities of \mathcal{X}_l , $l \geq \bar{r}$ are d .

The cardinalities of \mathcal{X}_l , $0 < l < \bar{r}$ are

$$K_l = 2^{M-\sigma_l^-} (q_{\sigma_l^-+1} - ld) + 2^{M-\sigma_l^- - 1} (q_{\sigma_l^-+2} - q_{\sigma_l^-+1}) + \dots + 2^{M-\sigma_l^+ + 1} (q_{\sigma_l^+} - q_{\sigma_l^+-1}) + 2^{M-\sigma_l^+} (ld + d - q_{\sigma_l^+}),$$

where

$$\sigma_l^- = \gamma(ld), \quad \sigma_l^+ = \gamma(ld + d - 1).$$

The system states are presented in Table 1.

Table 1. System states.

States $(r, n_0, n_1, \dots, n_M)$			Number of states
r	n_0	n_1, \dots, n_M	
0	0	n_1, \dots, n_M	2^M
	$q_1 \leq n_0 < q_2$	$1, n_2, \dots, n_M$	$(q_2 - q_1)2^{M-1}$

	$q_{\sigma_0} \leq n_0 \leq d-1$	$1, \dots, 1, n_{\sigma_0+1}, \dots, n_M$	$(d - q_{\sigma_0})2^{M-\sigma_0}$
$0 < l < \bar{r}$	$0 \leq n_0 < q_{\sigma_l^-+1} - ld$	$1, \dots, 1, n_{\sigma_l^-+1}, \dots, n_M$	$2^{M-\sigma_l^-} (q_{\sigma_l^-+1} - ld)$
	$q_{\sigma_l^-+1} - ld \leq n_0, n_0 < q_{\sigma_l^-+2} - ld$	$1, \dots, 1, n_{\sigma_l^-+2}, \dots, n_M$	$2^{M-\sigma_l^- - 1} (q_{\sigma_l^-+2} - q_{\sigma_l^-+1})$

	$q_{\sigma_l^+} - ld \leq n_0 < d$	$1, \dots, 1, n_{\sigma_l^+}, \dots, n_M$	$2^{M-\sigma_l^+} (ld + d - q_{\sigma_l^+})$
$l \geq \bar{r}$	$0 \leq n_0 \leq d-1$	$1, \dots, 1$	d

It is easily shown that for \mathcal{X}_l , $l > 0$, there are transitions to \mathcal{X}_{l-1} and \mathcal{X}_{l+1} . Thus, the Markov chain $\{\mathbf{x}(t), t \geq 0\}$ is a quasi-birth-and-death (QBD) process,

the generator matrix \mathbf{Q} of the process has the following form

$$\mathbf{Q} = \begin{pmatrix} \mathbf{B}_{00} & \mathbf{B}_{01} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{B}_{10} & \mathbf{B}_{11} & \mathbf{B}_{12} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{B}_{21} & \mathbf{B}_{22} & \mathbf{B}_{23} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{B}_{\bar{r}-1, \bar{r}-2} & \mathbf{B}_{\bar{r}-1, \bar{r}-1} & \mathbf{B}_{\bar{r}-1, \bar{r}} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{B}_{\bar{r}, \bar{r}-1} & \mathbf{B}_{\bar{r}, \bar{r}} & \mathbf{A}_0 & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{A}_2 & \mathbf{A}_1 & \mathbf{A}_0 & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{A}_2 & \mathbf{A}_1 & \mathbf{A}_0 & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{A}_2 & \mathbf{A}_1 & \ddots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \ddots \end{pmatrix},$$

where blocks \mathbf{A}_2 , \mathbf{A}_1 , \mathbf{A}_0 are square matrices of order d , \mathbf{B}_{00} is the square matrix of order L , blocks \mathbf{B}_{01} , \mathbf{B}_{10} are $L \times K_1$, $K_1 \times L$ matrices respectively, $\mathbf{B}_{l,l}$, $l = 1, \dots, \bar{r}$, are square matrices of order K_l (we set $K_{\bar{r}} = d$), blocks $\mathbf{B}_{l,l-1}$, $l = 2, \dots, \bar{r}$, are $K_l \times K_{l-1}$ matrices, and blocks $\mathbf{B}_{l,l+1}$, $l = 1, \dots, \bar{r} - 1$, are $K_l \times K_{l+1}$ matrices.

The elements of matrices \mathbf{B}_{01} , $\mathbf{B}_{l,l+1}$, $l = 1, \dots, \bar{r} - 1$, and \mathbf{A}_0 are determined by (1) and (2). The elements of matrices \mathbf{B}_{10} , $\mathbf{B}_{l,l-1}$, $l = 2, \dots, \bar{r}$, and \mathbf{A}_2 are determined by (5). Note that $\mathbf{A}_0 = \lambda \mathbf{I}$, where \mathbf{I} is an identity matrix of the appropriate order; \mathbf{A}_2 is the square matrix of order d , where the $(1, d)$ th element of the matrix is equal to $\sum_{i=1}^M \mu_i$, and other elements are zero.

The non-diagonal elements of matrices \mathbf{B}_{00} , $\mathbf{B}_{l,l}$, $l = 1, \dots, \bar{r}$, and \mathbf{A}_1 are determined by (2)–(4); the diagonal elements of these matrices are determined from conditions (we set $\mathbf{B}_{\bar{r}, \bar{r}+1} = \mathbf{A}_0$):

$$\begin{aligned} \mathbf{B}_{00} \mathbf{1} + \mathbf{B}_{01} \mathbf{1} &= \mathbf{0}, & \mathbf{B}_{l,l-1} \mathbf{1} + \mathbf{B}_{l,l} \mathbf{1} + \mathbf{B}_{l,l+1} \mathbf{1} &= \mathbf{0}, \\ \mathbf{A}_0 \mathbf{1} + \mathbf{A}_1 \mathbf{1} + \mathbf{A}_2 \mathbf{1} &= \mathbf{0}, \end{aligned}$$

where $\mathbf{1}$ is an ones column vector of the appropriate order. Note that \mathbf{A}_1 is the bidiagonal square matrix of order d , where the (j, j) th element of the matrix is equal to $-(\lambda + \sum_{i=1}^M \mu_i)$, and the $(j, j - 1)$ th element is equal to $\sum_{i=1}^M \mu_i$.

It is known [22], that the QBD process is ergodic if and only if $\pi_A \mathbf{A} \mathbf{1} < \pi_A \mathbf{A}_2 \mathbf{1}$, where $\pi_A \mathbf{A} = \mathbf{0}$, $\pi_A \mathbf{1} = 1$, $\mathbf{A} = \mathbf{A}_0 + \mathbf{A}_1 + \mathbf{A}_2$. The condition implies

$$\lambda d < \mu_1 + \cdots + \mu_M.$$

The stationary distribution $\boldsymbol{\pi} = (\boldsymbol{\pi}_0, \boldsymbol{\pi}_1, \dots)$ of the QBD process can be computed using the matrix-geometric method [22]. The row vector $\boldsymbol{\pi}_l$, $l = 0, 1, \dots$, defines probabilities of states for level l , according to the lexicographic order, $\boldsymbol{\pi}_l = (\boldsymbol{\pi}(\mathbf{x}) : \mathbf{x} \in \mathcal{X}_l)$.

We solve linear system $\boldsymbol{\pi}\mathbf{Q} = \mathbf{0}$ and $\boldsymbol{\pi}\mathbf{1} = 1$ to find the stationary probabilities. Taking the advantage of the block structure in \mathbf{Q} , we obtain

$$\begin{aligned}
& \boldsymbol{\pi}_0\mathbf{B}_{00} + \boldsymbol{\pi}_1\mathbf{B}_{10} = \mathbf{0}; \\
& \boldsymbol{\pi}_0\mathbf{B}_{01} + \boldsymbol{\pi}_1\mathbf{B}_{11} + \boldsymbol{\pi}_2\mathbf{B}_{21} = \mathbf{0}; \\
& \boldsymbol{\pi}_1\mathbf{B}_{12} + \boldsymbol{\pi}_2\mathbf{B}_{22} + \boldsymbol{\pi}_3\mathbf{B}_{32} = \mathbf{0}; \\
& \dots\dots\dots \\
& \boldsymbol{\pi}_{i-1}\mathbf{B}_{i-1,i} + \boldsymbol{\pi}_i\mathbf{B}_{i,i} + \boldsymbol{\pi}_{i+1}\mathbf{B}_{i+1,i} = \mathbf{0}; \\
& \dots\dots\dots \\
& \boldsymbol{\pi}_{\bar{r}-2}\mathbf{B}_{\bar{r}-2,\bar{r}-1} + \boldsymbol{\pi}_{\bar{r}-1}\mathbf{B}_{\bar{r}-1,\bar{r}-1} + \boldsymbol{\pi}_{\bar{r}}\mathbf{B}_{\bar{r},\bar{r}-1} = \mathbf{0}; \\
& \boldsymbol{\pi}_{\bar{r}-1}\mathbf{B}_{\bar{r}-1,\bar{r}} + \boldsymbol{\pi}_{\bar{r}}\mathbf{B}_{\bar{r},\bar{r}} + \boldsymbol{\pi}_{\bar{r}+1}\mathbf{A}_2 = \mathbf{0}; \\
& \boldsymbol{\pi}_{l-1}\mathbf{A}_0 + \boldsymbol{\pi}_l\mathbf{A}_1 + \boldsymbol{\pi}_{l+1}\mathbf{A}_2 = \mathbf{0}, \quad l = \bar{r} + 1, \bar{r} + 2, \dots
\end{aligned} \tag{6}$$

We know [22], the solution of (6) has the matrix-geometric form given by

$$\boldsymbol{\pi}_{\bar{r}+l} = \boldsymbol{\pi}_{\bar{r}}\mathbf{R}^l, \quad l = 1, 2, \dots,$$

where \mathbf{R} is the minimal non-negative solution to equation $\mathbf{A}_0 + \mathbf{R}\mathbf{A}_1 + \mathbf{R}^2\mathbf{A}_2 = \mathbf{0}$, vectors $\boldsymbol{\pi}_0, \boldsymbol{\pi}_1, \dots, \boldsymbol{\pi}_{\bar{r}}$ satisfy

$$\begin{aligned}
& (\boldsymbol{\pi}_0, \dots, \boldsymbol{\pi}_{\bar{r}}) \begin{pmatrix} \mathbf{B}_{00} & \mathbf{B}_{01} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \mathbf{B}_{10} & \mathbf{B}_{11} & \mathbf{B}_{12} & \dots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_{21} & \mathbf{B}_{22} & \dots & \mathbf{0} & \mathbf{0} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{B}_{\bar{r}-1,\bar{r}-1} & \mathbf{B}_{\bar{r}-1,\bar{r}} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{B}_{\bar{r},\bar{r}-1} & \mathbf{B}_{\bar{r},\bar{r}} + \mathbf{R}\mathbf{A}_2 \end{pmatrix} = (\mathbf{0}, \dots, \mathbf{0}), \\
& \boldsymbol{\pi}_0\mathbf{1} + \boldsymbol{\pi}_1\mathbf{1} + \dots + \boldsymbol{\pi}_{\bar{r}-1}\mathbf{1} + \boldsymbol{\pi}_{\bar{r}}(\mathbf{I} - \mathbf{R})^{-1}\mathbf{1} = 1.
\end{aligned}$$

4 Performance Measures

Once stationary distribution $\boldsymbol{\pi}$ is computed, a variety of other performance measures may be obtained.

The average number B of jobs in the queue

$$\begin{aligned}
B &= \sum_{l=1}^{\infty} l\boldsymbol{\pi}_l\mathbf{1} = \sum_{l=1}^{\bar{r}} l\boldsymbol{\pi}_l\mathbf{1} + \sum_{l=1}^{\infty} (\bar{r} + l)\boldsymbol{\pi}_{\bar{r}}\mathbf{R}^l\mathbf{1} = \\
&= \sum_{l=1}^{\bar{r}} l\boldsymbol{\pi}_l\mathbf{1} + \boldsymbol{\pi}_{\bar{r}}\mathbf{R} [\bar{r}(\mathbf{I} - \mathbf{R})^{-1} + (\mathbf{I} - \mathbf{R})^{-2}] \mathbf{1}.
\end{aligned}$$

The average job waiting time W in the queue (the average time that the job waits in the queue before the first of its tasks is assigned to a server)

$$W = B/\lambda.$$

Denote by T the average response time, then

$$T = W + V,$$

where V is the average job service time, which is defined as the average total time required to process all of the tasks associated with the job once the first task is assigned to a server.

The average number N of jobs in the system is

$$N = \lambda T.$$

To obtain the average job service time, we consider the service process of all d tasks of an arbitrarily chosen “tagged” job. There are two cases considered.

1. We assume that when the first task is assigned, the queueing system is in a state \mathbf{x} such that $r > \bar{r}$. Then all servers are busy, and tasks of the tagged job sequentially occupy available servers.

The service process of the tagged job is described by an absorbing Markov chain $\nu = \{\nu(t), t \geq 0\}$. Denote the state of chain ν by (k, \mathcal{S}) , where k denotes the number of tasks of the tagged job in the queue, \mathcal{S} denotes the set of indices for busy servers that process tasks of the tagged job. State $(0, \emptyset)$ is absorbing. The state space \mathcal{N} of Markov chain ν is given by

$$\mathcal{N} = \bigcup_{j=1}^d \mathcal{N}^j \cup \{(0, \emptyset)\},$$

where

$$\mathcal{N}^j = \{(k, \mathcal{S}) : k \in \{0, 1, \dots, d-j\}, \mathcal{S} \in \mathcal{S}^j\},$$

\mathcal{S}^j denotes the set of all j -element subsets of $\{1, \dots, M\}$,

$$|\mathcal{N}| = \sum_{j=1}^d j \binom{M}{d-j+1} + 1.$$

The initial probability distribution defined on state space \mathcal{N} is given by the vector $(\alpha_0, \boldsymbol{\alpha})$, where $\alpha_0 = \alpha_0(0, \emptyset) = 0$, $\boldsymbol{\alpha} = (\alpha(k, \mathcal{S}) : (k, \mathcal{S}) \in \mathcal{N} \setminus \{(0, \emptyset)\})$,

$$\alpha(d-1, \{i\}) = \frac{\mu_i}{\sum_{i=1}^M \mu_i}, \quad i = 1, \dots, M,$$

and other initial probabilities are zero.

Markov chain ν is determined by generator matrix \mathbf{Q}_ν with non-diagonal elements $q_\nu((k, \mathcal{S}), (k', \mathcal{S}'))$

$$q_\nu((k, \mathcal{S}), (k', \mathcal{S}')) = \begin{cases} \mu_i, & \text{if } k' = k = 0, \mathcal{S}' = \mathcal{S} \setminus \{i\}, i \in \mathcal{S}, \\ \mu_i, & \text{if } k' = k - 1 \geq 0, \mathcal{S}' = \mathcal{S} \cup \{i\}, i \notin \mathcal{S}, \\ \sum_{i \in \mathcal{S}} \mu_i, & \text{if } k' = k - 1 \geq 0, \mathcal{S}' = \mathcal{S}, \\ 0, & \text{otherwise.} \end{cases}$$

We denote by \mathbf{T} the square matrix of order $|\mathcal{N}| - 1$ which is a submatrix of matrix \mathbf{Q}_ν and contains transition rates among the transient states.

The absorption time ξ for Markov chain ν is a random variable which has a PH-distribution with PH-representation $(\boldsymbol{\alpha}, \mathbf{T})$ [22]. The expected absorption time is

$$\mathbb{E}[\xi] = -\boldsymbol{\alpha}\mathbf{T}^{-1}\mathbf{1}.$$

2. We assume that when the first task is assigned, the queueing system is in a state \mathbf{x} such that $0 \leq r \leq \bar{r}$. Then service of the tagged job starts right after the events:
 - the job arrives,
 - a task finishes and leaves the system,
 - a job arrives and this leads to a task of the job is assigned to an idle server.

In this case, the service process of the tagged job is described by an absorbing Markov chain $\zeta = \{\zeta(t), t \geq 0\}$. Denote the state of chain ζ by $(k, \mathcal{S}, \mathbf{x})$, where k denotes the number of tasks of the tagged job in the queue, \mathcal{S} denotes the set of indices for busy servers that process tasks of the tagged job, \mathbf{x} denotes the system state, $\mathbf{x} \in \hat{\mathcal{X}}$, $\hat{\mathcal{X}} = \{\mathbf{x} \in \mathcal{X} : r \leq \bar{r}\}$. States $(0, \emptyset, \mathbf{x})$, $\mathbf{x} \in \hat{\mathcal{X}}$ are absorbing. The state space \mathcal{Z} of Markov chain ζ is given by

$$\mathcal{Z} = \bigcup_{k=0}^{d-1} \mathcal{Z}^k,$$

where

$$\mathcal{Z}^0 = \bigcup_{\mathbf{x} \in \hat{\mathcal{X}}} \{(0, \mathcal{S}, \mathbf{x}) : \mathcal{S} \in \mathcal{S}(\mathbf{x}, 0)\},$$

$$\mathcal{Z}^k = \bigcup_{\mathbf{x} \in \hat{\mathcal{X}} : n_0=k} \{(k, \mathcal{S}, \mathbf{x}) : \mathcal{S} \in \mathcal{S}(\mathbf{x}, k)\}, \quad k = 1, \dots, d-1,$$

$$\mathcal{S}(\mathbf{x}, k) = \{\mathcal{S} \subseteq \bar{\mathcal{F}}(\mathbf{x}) : |\mathcal{S}| + k \leq d\}.$$

The initial probability distribution defined on state space \mathcal{Z} is given by the vector $(\boldsymbol{\beta}_0, \boldsymbol{\beta})$, where $\boldsymbol{\beta}_0 = (\beta_0(0, \emptyset, \mathbf{x}) : \mathbf{x} \in \hat{\mathcal{X}}) = \mathbf{0}$, $\boldsymbol{\beta} = (\beta(k, \mathcal{S}, \mathbf{x}') : (k, \mathcal{S}, \mathbf{x}') \in \mathcal{Z} \setminus \mathcal{Z}^0)$:

$$\beta(k, \mathcal{S}, \mathbf{x}') = \left(\lambda \sum_{\mathbf{x} \in \mathcal{A}(\mathbf{x}')} \pi(\mathbf{x}) + \sum_{i=1}^M \mu_i \sum_{\mathbf{x} \in \mathcal{D}_i(\mathbf{x}')} \pi(\mathbf{x}) \right) G^{-1},$$

$\mathbf{x} \in \mathcal{X}$ presents states from which the system moves to state \mathbf{x}' as a result of a job arrival ($\mathbf{x} \in \mathcal{A}(\mathbf{x}')$), or as a result of a task completion at server S_i ($\mathbf{x} \in \mathcal{D}_i(\mathbf{x}')$), $i = 1, \dots, M$; G denotes the normalizing constant.

The Markov chain ζ is determined by generator matrix \mathbf{Q}_ζ with non-diagonal elements $q_\zeta((k, \mathcal{S}, \mathbf{x}), (k', \mathcal{S}', \mathbf{x}'))$.

Suppose there is a transition from \mathbf{x} to \mathbf{x}' as a result of an event described in Section 3. So we have the corresponding transition from $(k, \mathcal{S}, \mathbf{x})$ to $(k', \mathcal{S}', \mathbf{x}')$, where k' and \mathcal{S}' will be described below.

Assume the system moves from state \mathbf{x} to state \mathbf{x}' as a result of a job arrival, we denote by $\mathcal{H}(\mathbf{x}, \mathbf{x}') = \bar{\mathcal{F}}(\mathbf{x}') \setminus \bar{\mathcal{F}}(\mathbf{x})$ the index set of idle servers which will be busy right after the event. Let $\mathcal{H}(\mathbf{x}, \mathbf{x}') = \{i_1, i_2, \dots, i_m\}$, $i_1 < i_2 < \dots < i_m$, then $\mathcal{H}_j(\mathbf{x}, \mathbf{x}') \subseteq \mathcal{H}(\mathbf{x}, \mathbf{x}')$ denotes the j -element set

$$\mathcal{H}_j(\mathbf{x}, \mathbf{x}') = \{i_1, \dots, i_j\}.$$

Thus, we can write

(a) *A job arrives*

$$q_\zeta((k, \mathcal{S}, \mathbf{x}), (k', \mathcal{S}', \mathbf{x}')) = \lambda,$$

where $k' = \max\{k - |\mathcal{H}(\mathbf{x}, \mathbf{x}')|, 0\}$, $\mathcal{S}' = \mathcal{S} \cup \mathcal{H}_{k-k'}(\mathbf{x}, \mathbf{x}')$,

(b) *A task finishes its service at S_i*

i. if $k > 0$, $n'_i = 1$, $i \notin \mathcal{S}$,

$$q_\zeta((k, \mathcal{S}, \mathbf{x}), (k-1, \mathcal{S} \cup \{i\}, \mathbf{x}')) = \mu_i,$$

ii. if $k > 0$,

$$q_\zeta((k, \mathcal{S}, \mathbf{x}), (k-1, \mathcal{S}, \mathbf{x}')) = \sum_{i \in \mathcal{S}: n'_i=1} \mu_i,$$

iii. if $k > 0$, $n'_i = 0$,

$$q_\zeta((k, \mathcal{S}, \mathbf{x}), (k, \mathcal{S} \setminus \{i\}, \mathbf{x}')) = \mu_i,$$

iv. if $k = 0$,

$$q_\zeta((0, \mathcal{S}, \mathbf{x}), (0, \mathcal{S} \setminus \{i\}, \mathbf{x}')) = \mu_i.$$

We denote by η the absorption time for Markov chain ζ . The expected absorption time $\mathbb{E}[\eta]$ is computed similarly to $\mathbb{E}[\xi]$.

Finally, we have

$$V = \frac{\mathbb{E}[\xi]c_1 + \mathbb{E}[\eta]c_2}{c_1 + c_2},$$

where $c_1 = \omega_1$, $c_2 = \beta \mathbf{1}G$,

$$(\omega_1, \dots, \omega_d) = (\pi_{\bar{r}}(\mathbf{I} - \mathbf{R})^{-1} - \pi_{\bar{r}} - \pi_{\bar{r}+1}) \sum_{i=1}^M \mu_i.$$

5 Numerical Results

Consider a fork-join queueing system which consists of $M = 3$ heterogeneous servers, where $\mu_1 = 5$, $\mu_2 = 2$, $\mu_3 = 1$.

The performance measures depend on threshold levels q_i , $i = 2, \dots, M$. Let q_i^* be the optimal threshold levels with respect to the minimization of the average response time T . They can be obtained by the exhaustive search method. Assume that $\lambda = 1$ and $d = 3$, then the optimal threshold levels are $q_2^* = 2$, $q_3^* = 8$, and $T = 0.838$.

To illustrate the advantages of the threshold control policy, we consider the fastest free server policy with $q_i = 1$, $i = 1, \dots, M$. Figure 2 presents the average response times for the optimal threshold control policy and the fastest free server policy for several values of the arrival rate λ .

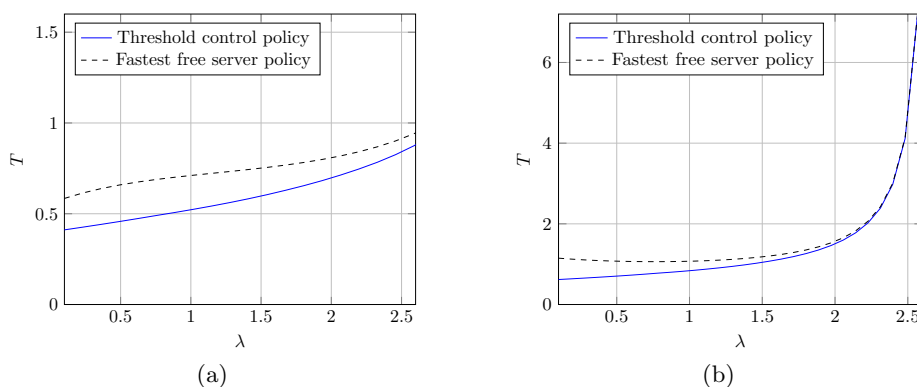


Fig. 2. The average response times for (a) $d = 2$ and (b) $d = 3$.

It may be seen that the optimal threshold control policy provides lower values of the average response time T . When the arrival rate λ is quite large, the difference between the policies can be neglected. As λ increases, the threshold levels decrease that leads to the fastest free server policy is near optimal one in this case.

6 Conclusions

We studied a Markovian fork-join queueing system with heterogeneous servers and threshold control policy. Applying a matrix-geometric approach, we obtained the stationary distribution of the system and performance measures. The results can be used for the performance analysis of multiprocessor systems and other modern distributed systems. An interesting topic for future research is to obtain the optimal threshold control policy which minimizes the long-run average number of jobs in the system or the average response time.

References

1. Rizk, A., Poloczek, F., Ciucu, F.: Computable bounds in fork-join queueing systems. *SIGMETRICS Perform. Eval. Rev.* **43**(1), 335–346 (2015). <https://doi.org/10.1145/2796314.2745859>
2. KhudaBukhsh, W.R., Rizk, A., Frömmgen, A., Koepl, H.: Optimizing stochastic scheduling in fork-join queueing models: bounds and applications. In: *IEEE INFOCOM 2017*, pp. 1–9. IEEE, New York (2017). <https://doi.org/10.1109/INFOCOM.2017.8057013>
3. Glushkova, D., Jovanovic, P., Abello, A.: MapReduce performance model for Hadoop 2.x. *Information Systems* **79**, 32–43 (2019). <https://doi.org/10.1016/j.is.2017.11.006>
4. Thomassian, A.: Analysis of fork/join and related queueing systems. *ACM Computing Surveys* **47**(2), 17:1–17:71 (2014). <https://doi.org/10.1145/2628913>
5. Joshi, G., Liu, Y., Soljanin, E.: On the delay-storage trade-off in content download from coded distributed storage. *IEEE Journal on Selected Areas on Communications* **32**(5), 989–997 (2014). <https://doi.org/10.1109/JSAC.2014.140518>
6. Heidelberger, P., Trivedi, K.S.: Analytic queueing models for programs with internal concurrency. *IEEE Trans. Comp.* **C-3**(1), 73–82 (1983). <https://doi.org/10.1109/TC.1983.1676125>
7. Gorbunova, A.V., Zaryadov, I.S., Matyushenko, S.I., Samouylov, K.E., Shorgin, S.Ya.: The approximation of response time of a cloud computing system. *Informatics and Applications* **9**(3), 31–38 (2015) (in Russian). <https://doi.org/10.14357/19922264150304>
8. Flatto, L., Hahn, S.: Two parallel queues created by arrivals with two demands I. *SIAM Journal of Applied Mathematics* **44**(5), 1041–1053 (1984). <https://doi.org/10.1137/0144074>
9. Nelson, R., Tantawi, A.N.: Approximate analysis of fork/join synchronization in parallel queues. *IEEE Trans. Comp.* **37**(6), 739–743 (1988). <https://doi.org/10.1109/12.2213>
10. Squillante, M.S., Zhang, Y., Sivasubramaniam, A., Gautam, N.: Generalized parallel-server fork-join queues with dynamic task scheduling. *Annals of Operations Research* **160**(1), 227–255 (2008). <https://doi.org/10.1007/s10479-008-0312-7>
11. Baccelli, F., Makowski, A.M., Shwartz, A.: The fork-join queue and related systems with synchronization constraints: stochastic ordering and computable bounds. *Adv. Appl. Probab.* **21**(3), 629–660 (1989). <https://doi.org/10.1017/S0001867800018851>
12. Ko, S.-S., Serfozo, R.F.: Sojourn times in G/M/1 fork-join networks. *Naval Research Logistics (NRL)* **55**(5), 432–443 (2008). <https://doi.org/10.1002/nav.20294>
13. Kumar, A., Shorey, R.: Performance analysis and scheduling of stochastic fork-join jobs in a multicomputer system. *IEEE Transactions on Parallel and Distributed Systems* **10**(4), 1147–1164 (1993). <https://doi.org/10.1109/71.246075>
14. Osipov, O.A.: A heterogeneous fork-join queueing system in which each job occupy all free servers. *RUDN Journal of Mathematics, Information Sciences and Physics* **26**(1), 28–38 (2018) (in Russian). <https://doi.org/10.22363/2312-9735-2018-26-1-28-38>
15. Narahari, Y., Sundarajan, P.: Performability analysis of fork-join queueing systems. *Journal of the Operational Research Society* **46**(10), 1237–1249 (1995). <https://doi.org/10.1057/jors.1995.171>
16. Lin, W., Kumar, P.: Optimal control of a queueing system with two heterogeneous servers. *IEEE Transactions on Automatic Control* **29**(8), 696–703 (1984). <https://doi.org/10.1109/TAC.1984.1103637>

17. Rykov, V.V., Efrosinin, D.V.: On the slow server problem. *Automation and Remote Control* **70**(12), 2013–2023 (2009). <https://doi.org/10.1134/S0005117909120091>
18. Rykov, V.: Monotone control of queueing systems with heterogeneous servers. *Queueing Sys.* **37**(4), 391–403 (2001). <https://doi.org/10.1023/A:1010893501581>
19. Efrosinin, D., Breuer, L.: Threshold policies for controlled retrial queues with heterogeneous servers. *Ann. Oper. Res.* **141**, 139–162 (2006). <https://doi.org/10.1007/s10479-006-5297-5>
20. Özkan, E., Kharoufeh, J.: Optimal control of a two-server queueing system with failures. *Probability in the Engineering and Informational Sciences* **28**(10), 489–527 (2014). <https://doi.org/10.1017/S0269964814000114>
21. Legros, B., Jouini, O.: Routing in a queueing system with two heterogeneous servers in speed and in quality of resolution. *Stochastic Models* **33**(3), 392–410 (2017). <https://doi.org/10.1080/15326349.2017.1303615>
22. He, Q.-M.: *Fundamentals of Matrix-Analytic Methods*. Springer, New York (2014). <https://doi.org/10.1007/978-1-4614-7330-5>