MINIMIZING MEAN RESPONSE TIME IN NON-OBSERVABLE DISTRIBUTED SYSTEMS WITH PROCESSOR SHARING NODES

Mikhail Konovalov
Institute of Informatics Problems
of the FRC CSC RAS, Moscow, Russia,
Email: mkonovalov@ipiran.ru

Rostislav Razumchik
Institute of Informatics Problems
of the FRC CSC RAS, Moscow, Russia,
Peoples' Friendship University of Russia
(RUDN University), Moscow, Russia
Email: rrazumchik@ipiran.ru,
razumchik-rv@rudn.ru

KEYWORDS

processor sharing, server farm, load balancing, customer assignment, dispatching, mean response time

ABSTRACT

Consider a non-observable distributed processing system with $N \geq 2$ single server queues operating in parallel, each under the processor sharing discipline. Jobs arrive one by one to the dispatcher, which immediately routes it to one of the queues. When making a routing decision the dispatcher does not have any online information about the system (current queues' sizes, size of the arriving job etc.) The only information available to the dispatcher is: job size distribution, job's inter-arrival time distribution, server's speeds, time instants of previously arrived jobs and previous routing decisions. Under these conditions, one is interested in the routing policies which minimize the job's long-run mean response time. Two new class of policies are being proposed, which, according to the numerical experiments, may significantly outperform the optimal probabilistic policy and the Round Robin policy.

INTRODUCTION

Consideration is given to the problem of optimal scheduling in parallel non-observable single server queues each with processor sharing (PS) discipline. In such systems, which are sometimes also referred to as dispatching systems or server farms or distributed processing systems, there is a dispatcher (broker, scheduler, load balancer), which has to route arriving jobs (immediately[1]) to one of the queues. The non-observability means that the dispatcher has only static information about the system: cumulative distribution function (CDF) of job's inter-arrival times, CDF of job's size and servers' speeds. Any online information about the system's state (like queue sizes, remaining work etc.) is not available to the dispatcher. Under these assumptions, one in interested in the routing policies, which minimize the job's long-run mean response[2] time.

The literature on non-observable dispatching systems is not rich; a short but recent review can be found in Konovalov and Razumchik (2018). Among the recent papers (more or less) related to the problem[3] of scheduling in unobservable queues one can mention Anselmi (2017); Hassin and Snitkovsky (2017); Burnetas, Economou and Vasiliadis (2017); Lingenbrink and Krishnamurthy (2017). The concept of the unobservable M/M/1 queue is covered in detail in (Hassin, 2016, Section 3). For a dispatching system with N servers in Anselmi (2017) the authors prove the optimality (in the regime when $N \to \infty$ and the system's load approaches 1) of a subset of deterministic and periodic policies within a wide set of (open-loop[4]) policies that can be randomized or deterministic and can be dependent on the arrival process at the dispatcher. A system with the Poisson input, two interdependent queues (one is observable and the other is unobservable) and the Bernoulli scheduling policy is considered in Hassin and Snitkovsky (2017). Here the authors study the interrelation between the equilibrium strategy and the socially optimal strategy. The decision problem whether to join or not the single-server (almost[5]) unobservable system upon arrival and whether to stay or renege later is being solved in Burnetas, Economou and Vasiliadis (2017). Another single-server unobservable queue offering a service at a known fixed price to delay sensitive jobs is studied in Lingenbrink and Krishnamurthy (2017) from the standpoint of signalling mechanism and service provider's price which maximize the revenue.

To the best of our knowledge in the previous research papers, devoted to the optimal routing in unobservable dispatching systems, it was assumed that the service discipline employed in each queue is FIFO. Already under this assumption only two class of routing policies are

[1] I.e. the dispatcher does not have a queue to store the jobs.

[2] Or, equivalently, job's mean sojourn time in the system.

[3] According to the proceedings Ayesta et al. (2016) this topic has also been discussed to some extent during the second European Conference on Queueing Theory. So far some of the presented talks have not been published as separate papers.

[4] Open-loop control means any dispatching algorithm that does not rely on feedback information dynamically flowing from the queues back to the controller.

[5] Since it is assumed that the system administrator announces to the customers their positions in the system in a stochastic manner.

Communications of the ECMS, Volume 33, Issue 1,
Proceedings, ©ECMS Mauro Iacono, Francesco Palmieri,
Marco Gribaudo, Massimo Ficco (Editors)
ISBN: 978-3-937436-65-4/978-3-937436-66-1(CD) ISSN 2522-2414

available to the dispatcher[6]: probabilistic and deterministic (see Hordijk and van der Laan (2004)). When one switches the scheduling in queues from FIFO to PS, the choice is narrowed down to the probabilistic routing and the Round Robin[7].

According to a probabilistic (Bernoulli, random) routing a job is routed to the queue m, $1 \le m \le N$, with the probability p_m independently of the previous decisions. For finding p_m there exist efficient numerical procedures (see, for example, Combe and Boxma (1994); Bell and Stidham (1983); Neely and Modiano (2005); Sethuraman and Squillante (1999)). If the arrival flow is Poisson[8] with rate λ, then the arrival process to each server remains Poissonian and each server behaves as the M/G/1-PS queue with the arrival rate $p_m\lambda$. The optimal probabilistic routing policy[9], further referred to as RND-opt, is the probability distribution $(p_1, p_2, \dots, p_N)$ that minimizes the job's mean response time

$$\sum_{m=1}^{N} p_m \frac{\mathbf{E}(X/v^{(m)})}{1 - p_m\lambda\mathbf{E}(X/v^{(m)})} \tag{1}$$

under the constraint $0 \le p_m\lambda\mathbf{E}(X/v^{(m)}) < 1$ for each m. Here $\mathbf{E}X$ denotes the mean job size. The solution can be found in many sources, for example, in Bell and Stidham (1983); Haviv and Roughgarden (2007); Altman, Ayesta and Prabhu (2011); Hyytiä, Virtamo et al. (2011). For the sake of completeness it is reproduced below according to (Bell and Stidham, 1983, Eq. (28)):

$$p_m = \begin{cases} 0, & \text{if } 0 < \lambda \le r_m, \\ \frac{1}{\lambda}\left(\frac{v^{(m)}}{\mathbf{E}X} - \frac{\sqrt{\frac{v^{(m)}}{\mathbf{E}X}}}{\sum_{i=1}^{N^*}\sqrt{\frac{v^{(i)}}{\mathbf{E}X}}}\left(\sum_{i=1}^{N^*}\frac{v^{(i)}}{\mathbf{E}X} - \lambda\right)\right), & \text{else,} \end{cases} \tag{2}$$

where

$$r_k = \sum_{i=1}^{k}\left(\frac{v^{(i)}}{\mathbf{E}X} - \sqrt{\frac{v^{(i)}}{\mathbf{E}X}\frac{v^{(k)}}{\mathbf{E}X}}\right), \quad r_{N+1} = \sum_{i=1}^{N}\frac{v^{(i)}}{\mathbf{E}X},$$

and $N^* = \min_{1\le k\le N}(r_k < \lambda \le r_{k+1})$. We will not discuss the computation of the job's mean response time under the Round Robin (RR) policy and just mention that when the arrival flow is Poisson, the inter-arrival times into each queue have an Erlang-N distribution, and thus each queue is E_N/G/1–PS. In some cases the computation of the mean response time can be performed analytically; for example, when the job size distribution is deterministic, the job's mean response time can be computed from the relation (3.3) in Brandt and Brandt (2006).

In this paper two new class of routing policies are proposed, which can outperform both the optimal probabilistic policy and the Round Robin policy. In addition to the assumptions made above, the new policies only require that the dispatcher can memorize its previous routing decisions and time instants at which those decisions were made. The new policies are based on the estimations of some average values related to the queues and, in some cases, can be not much worse than the online Join-the-Shortest-Queue (JSQ) policy.

The paper is organized as follows. In the next section the detailed description of model is given. The third section contains the overview of the new policies, that is followed by the section with the numerical comparison of the new policies with the RND-opt, RR and JSQ policies. In the concluding section one discusses pros and cons of the proposed policies and points out directions for further research.

MODEL DESCRIPTION AND ASSUMPTIONS

The system consists of $N \ge 2$ single server infinite capacity queues, operating in parallel. The queues are numbered from 1 to N. The server's speed of queue m is denoted by $v^{(m)}$, $1 \le m \le N$. The service discipline employed in each queue is processor sharing. Service preemption and jockeying between queues is not allowed. Inter-arrival times between jobs, which arrive one by one, and their sizes are i.i.d. with the known CDF $A(x)$ and $B(x)$ respectively. Upon receiving a job the dispatcher must immediately route it to one of the queues.

Fix an arbitrary integer $n \ge 1$. Let $0 \le t_1 < \cdots < t_n$ denote the arrival instants of the first n jobs and let $y_1, y_2, \dots, y_{n-1}$ be the first $n-1$ routing decisions i.e. y_j is the server whereto the job arrived at instant t_j was routed. Each y_j takes a value from the set $\{1, 2, \dots, N\}$. For the n^{th} job arrived at t_n, the dispatcher in order to make a routing decision may use only the following information: (i) the values of $t_1, t_2, \dots, t_n$, (ii) the values of $y_1, y_2, \dots, y_{n-1}$, (iii) the inter-arrival time distribution $A(x)$ and the jobs size distribution $B(x)$, (iv) the values $v^{(1)}, v^{(2)}, \dots, v^{(N)}$ of the servers' speeds. Online information (like the arriving job size, current queues' sizes etc.) is not available. The objective of the dispatcher is to route jobs in such a way that the job's (long-run) mean response time is minimal.

OVERVIEW OF THE NEW POLICIES

Based on the system description, two new class of policies are being suggested. The first class of policies implies that the dispatcher routes an arriving job to the

[6]This is in sharp contrast with the partially and fully observable dispatching systems for which a good selection of routing policies exits (see Bonomi (1990), Fu et al. (2016), Hyytiä, Penttinen et al. (2011), Harchol-Balter, Crovella and Murta (1999)), (Harchol-Balter, 2013, Section 24.2).

[7]That is the special case of a deterministic routing. In fact, general deterministic policies are also applicable here, but we are unaware of any procedures for finding optimal or near optimal deterministic sequences for PS server farms.

[8]The assumption of Poisson flow of jobs allows one to consider more general dispatching systems as those described, for example, in Section III.E in Hyytiä, Penttinen et al. (2011). New policies proposed in this paper are applicable in such general cases as well.

[9]Whenever the dispatcher has more information about the system's current state (for example, current backlogs of the queues and/or the size of the arriving job), other (state-dependent) policies can do better. For example, JSQ or FPI-ρ policy as described in Hyytiä, Penttinen et al. (2011).

queue, for which the probability of being least loaded[10] at the moment is maximum. The decision rule is the following: send the first job to the queue with the fastest server; send the n-th job to the queue y_n, where

$$y_n = \text{argmax}_m \left\{ \mathbf{E} 1_{\{A_n^m\}},\ 1 \le m \le N \right\},\ n \ge 2,$$

where $\mathbf{E}$ denotes the expectation operator and $1_{\{A_n^m\}}$ denotes the indicator function of the event A_n^m, which is defined as $A_n^m = \{N_n^{(m)} \le N_n^{(1)}, N_n^{(m)} \le N_n^{(2)}, \ldots, N_n^{(m)} \le N_n^{(N)}\}$, where $N_n^{(m)}$ denotes the number of jobs in the queue m upon arrival of the n^{th} job, conditioning of the fact that the previous $(n-1)$ jobs arrived at instants $t_1, t_2, \ldots, t_{n-1}$ and were routed to queues $y_1, y_2, \ldots, y_{n-1}$. Ties are always broken in favour of the queue with the fastest server or, if there is none, randomly. Since the exact values of $\mathbf{E} 1_{\{A_n^m\}}$ are unavailable, computer-aided simulation is used to estimate them. The first option (further referred to as CAA policy; see the pseudocode in the Algorithm 1) is to take the part of the previous trajectory ($d \ge 1$ previous values of the inter-arrival times $t_{n-1}, \ldots, t_{n-d}$ and routing decisions $y_{n-1}, \ldots, y_{n-d}$) and to simulate the system's trajectory $r \ge 1$ times starting from the *empty* system (and sampling service times from $B(x)$). At the end of each run (i.e. at instant t_n) $1_{\{A_n^m\}}$ is evaluated and $\mathbf{E} 1_{\{A_n^m\}}$ is the average over r runs.

The second option (further referred to as CA policy; see the pseudocode in the Algorithm 2) is to take only $d \ge 1$ previous routing decisions $y_{n-1}, \ldots, y_{n-d}$ and to simulate the system's trajectory $r \ge 1$ times starting from the *empty* system (and sampling *both* service and inter-arrival times). Again, at the end of each run (i.e. at instant t_n) $1_{\{A_n^m\}}$ is evaluated and $\mathbf{E} 1_{\{A_n^m\}}$ is the average over r runs.

Algorithm 1 High-level description of the implementation procedure for the CAA policy

function NextDecision($N, v^{(1)}, \ldots, v^{(N)}, B(x), t_n, t_{n-1}, \ldots, t_{n-d}, y_{n-1}, \ldots, y_{n-d}$)
  for $i = 1 \to r$ **do**
    z_i = SimulateSystem($t_n, \ldots, t_{n-d}, y_{n-1}, \ldots, y_{n-d}$)
  end for
  Find $i^* : \sum_{i=1}^r \delta_{i^*,z_i} = \max(\sum_{i=1}^r \delta_{1,z_i}, \ldots, \sum_{i=1}^r \delta_{N,z_i})$
  return $y_n = i^*$
end function

[a] SimulateSystem($t_n, \ldots, t_{n-d}, y_{n-1}, \ldots, y_{n-d}$) denotes a function, which returns the result of a single simulation run of the considered system under the following conditions: the system starts *empty* at the instant t_{n-d} and has job arrivals at the instants $t_{n-d}, t_{n-d+1}, \ldots, t_{n-1}$, which are routed to the servers $y_{n-d}, y_{n-d+1}, \ldots, y_{n-1}$. The job service times are sampled from $B(x)$. At the instant t_n the simulation is stopped and the number of the server, which currently has the minimum number of jobs in it, is returned.
[b] The positive integer values d and r are the parameters of the algorithm, which must be set in advance. Their values are mostly influenced by the inhomogeneity of the system (different servers' speeds), by the job size distribution and the system's load.
[c] δ_{i,z_i} is the Kronecker symbol, i.e. $\delta_{i,z_i} = 1$ if $i = z_i$ and $\delta_{i,z_i} = 0$ otherwise.

[10] The load is measured in terms of the total number of customers in the queue.

Algorithm 2 High-level description of the implementation procedure for the CA policy

function NextDecision($N, v^{(1)}, \ldots, v^{(N)}, B(x), y_{n-1}, y_{n-2}, \ldots, y_{n-d}$)
  for $i = 1 \to r$ **do**
    Generate($t_n, \ldots, t_{n-d}$)
    z_i = SimulateSystem($t_n, \ldots, t_{n-d}, y_{n-1}, \ldots, y_{n-d}$)
  end for
  Find $i^* : \sum_{i=1}^r \delta_{i^*,z_i} = \max(\sum_{i=1}^r \delta_{1,z_i}, \ldots, \sum_{i=1}^r \delta_{N,z_i})$
  return $y_n = i^*$
end function

[a] SimulateSystem($t_n, \ldots, t_{n-d}, y_{n-1}, \ldots, y_{n-d}$) denotes the same function as in the Algorithm 1.
[b] Generate($t_n, \ldots, t_{n-d}$) denotes a function, which generates $d+1$ consecutive job arrival instants, labelled by $t_n, \ldots, t_{n-d}$. Sampling is performed from $A(x)$.

The second class of policies, which can be used in the given context, is the modification of the policy proposed in Konovalov and Razumchik (2018) for the minimization of the job's mean response time in non-observable queues with FIFO scheduling[11]. The dispatching rule is the following: send the first job to the queue with the fastest server; send the n-th job to the queue y_n, where

$$y_n = \text{argmin}_m \left\{ \mathbf{E} N_n^{(m)},\ 1 \le m \le N \right\},\ n \ge 1.$$

Since the exact sizes of the arriving jobs are not observed, the exact values of $N_n^{(m)}$ cannot be computed. Yet in some special cases, approximations for $\mathbf{E} N_n^{(m)}$ are possible. Such case is, for example[12], when $B(x) = 1 - e^{-\mu x}$. The key to obtaining the values of $\mathbf{E} N_n^{(m)}$ is the discrete-time setting. All time-related quantities in the model are discretized into fixed-length intervals (slots) of such length Δ, so that $\mu\Delta < 1$. Then $q_{ij} = C_i^j (\mu\Delta/i)^j (1 - \mu\Delta/i)^{i-j}$ approximates the probability that the service of j jobs will be completed after one slot, given that at the beginning of a slot there where i jobs in the queue[13]. Then, if initially the system is empty, the distribution $\vec{p}_n^{(m)}$ of the number of jobs in the queue m upon arrival of the n^{th}, $n \ge 1$, job can be computed recursively from the relations:

$$\vec{p}_1^{(m)} = (1 - \delta_{m,y_1}, \delta_{m,y_1}),$$

$$\vec{p}_2^{(m)} = \vec{p}_1^{(m)} \left(\tilde{\mathbf{P}}_1 (\mathbf{P}_1)^{\frac{t_2 - t_1}{\Delta} - 1} \delta_{m,y_1} + \mathbf{I}_2 \cdot (1 - \delta_{m,y_1}) \right),$$

[11] The following observation is forth noticing here. While there is no rational reason to use the arrival-aware (AA) policy, proposed in Konovalov and Razumchik (2018), in a PS server farm (when trying to minimize the mean response time or mean waiting time), it becomes reasonable as soon as the system's load is high and in the queues, instead of PS, the limited processor sharing policy (LPS) with the same/different multi-programming levels (MPL) is used. The intuition behind this idea is the following. Under high system load it is quite probable that the total number of customers in each queue is greater than the MPL of the LPS policy. When this happens LPS policy starts to behave as FIFO, for which the AA policy shows good results.

[12] Phase-type distribution for $B(x)$ is also allowed but this case requires a separate study.

[13] Here, for simplicity, it is assumed that all servers have equal speeds equal to 1.

$$\vec{p}_n^{(m)} = \vec{p}_{n-1}^{(m)} \Big(\tilde{\mathbf{P}}_{n-1} (\mathbf{P}_{n-1})^{\frac{t_n - t_{n-1}}{\Delta} - 1} \delta_{m,y_{n-1}} + \mathbf{I}_{n-1,n} \cdot (1 - \delta_{m,y_{n-1}}) \Big),$$

where $\mathbf{I}_n$ denotes the identity square matrix of size n, $\mathbf{I}_{n-1,n}$ denotes the identity square matrix of size $n-1$ with the zero column appended to the right,

$$\mathbf{P}_1 = \tilde{\mathbf{P}}_1 = \begin{pmatrix} 1 & 0 \\ \mu\Delta & 1 - \mu\Delta \end{pmatrix},$$

the square matrix $\mathbf{P}_n$ is obtained from the matrix $\mathbf{P}_{n-1}$ by appending to it one zero column to the right and then the row $(q_{n,n}, q_{n,n}, \ldots, q_{n,0})$ to the bottom and the rectangular matrix $\tilde{\mathbf{P}}_n$ is obtained from the matrix $\mathbf{P}_n$ by deleting its first row. Once the distributions $\vec{p}_n^{(m)}$ are computed, the queue with the minimum mean number of jobs can be chosen. The implementation of the above algorithm is not straightforward since right now the dimension of each vector $\vec{p}_n^{(m)}$ is equal to the number of arrivals to the system up to instant t_n. Thus special procedures for keeping the vector dimensions low are required. We leave the discussion of this issue as well as the choice of Δ for the future and proceed in the next section to the numerical examples from which some conclusions about the performance of the CAA and CA policies can be drawn.

NUMERICAL EXPERIMENT

Extensive numerical experiments show, that the new routing policies generally lead to visibly better results than the RND-opt and RR policies. Exceptions, when it may not be easy (or not possible at all) to get the gain, are the cases with highly variable job size distributions.

Numerical results presented below show the values of the job's mean response time under five different dispatching policies – RND-opt, RR, JSQ, CAA and CA. For the demonstration purposes four special cases of the system structure are considered: the system with 2 servers (see Table 1), the system with 4 servers (see Table2), the system with 8 servers (see Table 3) and the system with 16 servers (see Table 4). In each case the system is heterogeneous (servers have different speeds), the incoming flow of jobs is Poisson and the mean job size is equal to 1. The considered job size distributions are: deterministic, uniform, normal, exponential, hyper-exponential and Pareto.

Although in some cases the values of the job's mean response time can be computed explicitly from the available analytic results, the values displayed in the tables were obtained from the simulation (the simulation framework is described in Konovalov and Razumchik (2014); Konovalov (2014)). In the implementation of the JSQ that was used, the ties were broken according to the rule: if two or more queues have the same number of jobs, choose the queue with the fastest server; if there is no such queue, choose the random one.

The numerical results evidence that the CAA policy (see Algorithm 1), which is based both on the inter-arrival times and the decision history, usually outperforms both both the RND-opt and the RR policy. The relative gain (with respect to the RND-opt policy) is higher for the low variable job size distributions and (in the presented cases) vanishes with the increase of the job size variability. The surprising fact, that can be seen from the tables, is that the CA policy (see Algorithm 2), which is based only on the decision history and does not utilize any information about the inter-arrival times, also does better than RND-opt and RR policy in many cases. As expected, the CAA policy is worse than the online JSQ policy. Yet for low variable job size distributions their performance is quite close.

SUMMARY

The results in the Tables 1–4 demonstrate that the new policies, memorizing the inter-arrival times and the routing decisions made, can improve the performance of an non-observable system with processor sharing single server queues, working in parallel and independently. For job size distributions with low variance the gain may be quite big, whereas for highly variable distributions the gain is usually low or absent. In general, the gain depends on the properties of the job size distribution, on the system structure (number of queues, server's speeds) and the values of the policy parameters r and d. More understanding is needed here. As numerical experiments evidence, the new policies sometimes can do almost as good as the JSQ policy.

Generally, the performance improvement that is promised in the Tables 1–4, comes at price. The optimal RND policy and the RR policy can be implemented in the dispatcher at very limited costs, but the new policies require computational efforts and parameter (number of simulation repetitions r and history depth d) tuning . Although the implementation algorithm for the new policies is simple, the rules for choosing the values for r and d are most likely not. This can be seen already from the Tables 3 and 4: the job's mean response time, as the function of d, is not monotonic. Further study is needed here. The question of ranking the RND-opt, RR and the new policies with respect to the job's mean waiting time and stochastic order (see Bonomi (1990); Winston (1977); Weber (1978)) remains open as well.

Acknowledgements The reported study was funded by RFBR according to the research project №18-07-00692. The publication has been prepared with the support of the "RUDN University Program 5-100".

REFERENCES

Anselmi, J. 2017. Asymptotically optimal open-loop load balancing. Queueing Systems. Vol. 87. No. 3-4. Pp. 245–267.

Altman, E., Ayesta, U., Prabhu, B. 2011. Load balancing in processor sharing systems. Telecommunication Systems. Vol. 47. No. 1. Pp. 35–48.

Table 1: Job's mean response time in the system with 2 servers ($N = 2$). The server's speeds are $v^{(1)} = 2$ and $v^{(2)} = 1$. The job arrival flow is Poisson with rate $\lambda = 1$. Mean job size $\mathbf{E}X = 1$. The offered load to the whole system is $\lambda \mathbf{E}X / \sum_{i=1}^{2} v^{(i)} \approx 0{,}33$. For the CAA and CA policies the number of simulation repetitions is $r = 10$.

		DET	U[0,5; 1,5]	N(1; 0,1)	Exp	H_2	Pareto
JSQ		0,721	0,724	0,724	0,732	0,734	0,727
RND-opt		0,915	0,913	0,914	0,914	0,914	0,914
Round Robin		0,946	0,971	0,974	1,096	1,133	1,002
CA	$d = 5$	0,864	0,877	0,877	0,939	0,964	0,893
CAA	$d = 5$	0,722	0,769	0,775	0,894	0,923	0,811
	$d = 10$					0,907	

[a] DET – deterministic, fixed job size equal to 1.
[b] U[0,5; 1,5] – uniformly distributed on [0,5; 1,5] job size.
[c] N[1; 0,1] – normally distributed job size.
[d] Exp – exponentially distributed job size with mean 1.
[e] H_2 – hyper-exponential job size distribution with mean 1, service rates $(2/3, 2)$ and equal phase probabilities.
[f] Pareto – Pareto-distributed job size with shape 3 and scale $2/3$.

Table 2: Job's mean response time in the system with 4 servers ($N = 4$). The server's speeds are $v^{(1)} = 4$, $v^{(2)} = 3$, $v^{(3)} = 2$, $v^{(4)} = 1$. The job arrival flow is Poisson with rate 5. The offered load to the whole system is 0,5. For the CAA and CA policies the number of simulation repetitions is $r = 10$.

		DET	U[0,5; 1,5]	N(1; 0,1)	Exp	H_2	Pareto
JSQ		0,411	0,414	0,414	0,420	0,422	0,415
RND-opt		0,711	0,711	0,711	0,711	0,711	0,711
Round Robin		∞	∞	∞	∞	∞	∞
CA	$d = 5$	0,594	0,662	0,671	1,129	1,316	0,771
CAA	$d = 5$	0,421	0,508	0,525	0,963	1,105	0,596
	$d = 10$				0,737	0,852	
	$d = 15$				0,684	0,767	
	$d = 20$					0,741	

Table 3: Job's mean response time in the system with 8 servers ($N = 8$). The server's speeds are $v^{(1)} = 5$, $v^{(2)} = v^{(3)} = v^{(4)} = 2$, $v^{(5)} = v^{(6)} = v^{(7)} = v^{(8)} = 1$. The job arrival flow is Poisson with rate 5. The offered load to the whole system is $\approx 0{,}33$. For the CAA and CA policies the number of simulation repetitions is $r = 10$.

		DET	U[0,5; 1,5]	N(1; 0,1)	Exp	H_2	Pareto
JSQ		0,383	0,383	0,383	0,385	0,385	0,384
RND-opt		0,595	0,596	0,597	0,598	0,596	0,596
Round Robin		0,745	0,779	0,786	1,074	1,160	0,865
CA	$d = 5$	0,556	0,570	0,574	0,685	0,711	0,593
CAA	$d = 5$	0,384	0,446	0,456	0,629	0,666	0,500
	$d = 10$					0,698	
	$d = 15$				0,634		

Ayesta, U., Boon, M., Prabhu, B., Righter, R., Verloop, M. 2016. European conference on queueing theory. 72p. https://hal.archives-ouvertes.fr/hal-01368218

Bell, C. H., Stidham S. 1983. Individual versus social optimization in the allocation of customers to alternative servers. Management Science. Vol. 29. No. 7. Pp. 831–839.

Bonomi, F. 1990. On job assignment for a parallel system of processor sharing queues. IEEE Transactions on Computers. Vol. 39. No. 7. Pp. 858–869.

Brandt, A., Brandt, M. 2006. A sample path relation for the sojourn times in G/G/1–PS systems and its applications. Queueing Systems. Vol. 52. Pp. 281–286.

Burnetas, A., Economou, A., Vasiliadis, G. 2017. Strategic customer behavior in a queueing system with delayed observations. Queueing Systems. Vol. 86. No. 3-4. Pp. 389–418.

Combe, M. B., Boxma O. J. 1994. Optimization of static traffic allocation policies. Theor. Comput. Sci. Vol. 125. No. 1. Pp. 17–43.

Fu, J., Moran, B., Guo, J., Wong, E.W.M., Zukerman, M. 2016. Asymptotically Optimal Job Assignment for Energy-

Table 4: Job's mean response time in the system with 16 servers ($N = 16$). The server's speeds are $\nu^{(1)} = \nu^{(2)} = \nu^{(3)} = 3$, $\nu^{(4)} = \nu^{(5)} = \nu^{(6)} = \nu^{(7)} = \nu^{(8)} = 2$, $\nu^{(9)} = \nu^{(10)} = \nu^{(11)} = \nu^{(12)} = \nu^{(13)} = \nu^{(14)} = \nu^{(15)} = \nu^{(16)} = 1$. The job arrival flow is Poisson with rate 10. The offered load to the whole system is $\approx 0{,}37$. For the CAA and CA policies the number of simulation repetitions is $r = 10$.

		DET	U[0,5; 1,5]	N(1; 0,1)	Exp	H_2	Pareto
JSQ		0,413	0,412	0,413	0,413	0,413	0,413
RND-opt		0,816	0,816	0,816	0,816	0,816	0,816
Round Robin		0,725	0,747	0,752	1,042	1,130	0,839
CA	$d = 5$	0,454	0,476	0,480	0,617	0,658	0,513
CAA	$d = 5$	0,427	0,461	0,468	0,616	0,658	0,509
	$d = 6$		0,451				
	$d = 7$		0,445	0,450			0,495
	$d = 8$		0,455	0,461			
	$d = 10$	0,413					

Efficient Processor-Sharing Server Farms. IEEE Journal on Selected Areas in Communications. Vol. 34. No. 12. Pp. 4008–4023.

Hassin, R. 2016. Rational Queueing. CRC Press, Boca Raton.

Hassin, R., Snitkovsky, R.I. 2017. Strategic customer behavior in a queueing system with a loss subsystem. Queueing Systems. Vol. 86. No. 3-4. Pp. 361–387.

Haviv, M., Roughgarden, T. 2007. The price of anarchy in an exponential multi-server. Operations Research Letters. Vol. 35. No. 4. Pp. 421–426.

Harchol-Balter, M., Crovella, M.E., Murta, C.D. 1999. On choosing a task assignment policy for a distributed server system. Journal of Parallel and Distributed Computing. Vol. 59. Pp. 204–228.

Harchol-Balter, M. 2013. Performance Modeling and Design of Computer Systems: Queueing Theory in Action (1st ed.). Cambridge University Press, New York, NY, USA.

Hordijk, A., van der Laan D. A. 2004. Periodic routing to parallel queues and billiard sequences. Math. Method. Oper. Res., 2004. Vol. 59. No. 2. Pp. 173–192.

Hyytiä, E., Penttinen, A., Aalto, S., Virtamo J. 2011. Dispatching problem with fixed size jobs and processor sharing discipline. 23rd International Teletraffic Congress, San Fransisco, USA. Pp. 190—197.

Hyytiä, E., Virtamo, J., Aalto, S., Penttinen, A. 2011. M/M/1-PS Queue and Size-Aware Task Assignment. Performance Evaluation. Vol. 68. No. 11. Pp. 1136–1148.

Konovalov, M. G. 2014. Building a simulation model for solving scheduling problems of computing resources. Systems and Means of Informatics. Vol. 24. No. 4. Pp. 45–62. (in Russian)

Konovalov, M., Razumchik R. 2014. Simulation Of Task Distribution In Parallel Processing Systems. Proceedings of the 6th International Congress on Ultra Modern Telecommunications and Control Systems. Pp. 657–663.

Konovalov, M.G., Razumchik, R.V. 2018 Improving routing decisions in parallel non-observable queues. Computing. Vol. 100. No. 10. Pp. 1059–1079.

Lingenbrink, D., Iyer K. 2017. Optimal Signaling Mechanisms in Unobservable Queues with Strategic Customers. In Proceedings of the 2017 ACM Conference on Economics and Computation, New York, NY, USA. Pp. 347–347.

Neely, M. J., Modiano E. 2005. Convexity in queues with general inputs. IEEE Transactions on Information Theory. Vol. 51. No. 2. Pp. 706–714.

Sethuraman, J., Squillante M. S. 1999. Optimal stochastic scheduling in multi-class parallel queues. SIGMETRICS. Pp. 93–102.

Weber, R.W. 1978. On optimal assignment of customers to parallel servers. Journal of Applied Probability. Vol. 15. No. 2. Pp. 406–413.

Winston, W. 1977. Optimality of the shortest line discipline. Journal of Applied Probability. Vol. 14, No. 1. Pp. 181–189.

AUTHOR BIOGRAPHIES

MIKHAIL KONOVALOV is a Doctor of Sciences in Technics and holds position of the principal scientist at Information Technologies Department at Institute of Informatics Problems of the Federal Research Center "Computer Science and Control" of the Russian Academy of Sciences. His research activities are focused on adaptive control of random sequences, modelling and simulation of complex systems. His email address is `mkonovalov@ipiran.ru`.

ROSTISLAV RAZUMCHIK received his Ph.D. degree in Physics and Mathematics in 2011. Since then, he has worked as a leading research fellow at Institute of Informatics Problems of the Federal Research Center "Computer Science and Control" of the Russian Academy of Sciences (FRC CSC RAS). Currently he also holds the associate professor position at Peoples' Friendship University of Russia (RUDN University). His current research activities are focused on queueing theory and its applications for performance evaluation of stochastic systems. His email address is `rrazumchik@ipiran.ru`