Paper

Iterative Algorithm for Threshold Calculation in the Problem of Routing Fixed Size Jobs to Two Parallel Servers

Mikhail Konovalov and Rostislav Razumchik

Institute of Informatics Problems, Federal Research Center "Computer Science and Control",
Russian Academy of Sciences, Moscow, Russia

Abstract—At present, solutions of many practical problems require significant computational resources and systems (grids, clouds, clusters etc.), which provide appropriate means are constantly evolving. The capability of the systems to fulfil quality of service requirements pose new challenges for the developers. One of the well-known approaches to increase system performance is the use of optimal scheduling (dispatching) policies. In this paper the special case of the general problem of finding optimal allocation policy in the heterogeneous *n*-server system processing fixed size jobs is considered. There are two servers working independently at constant but different speeds. Each of them has a dedicated queue (of infinite capacity) in front of it. Jobs of equal size arrive at the system. Inter-arrival times are i.i.d. random variables with general distribution with finite mean. Each job upon arrival must be immediately dispatched to one of the two queues wherefrom it will be served in FCFS manner (no pre-emption). The objective is the minimization of mean job sojourn time in the system. It is known that under this objective the optimal policy is of threshold type. The authors propose scalable fast iterative non-simulation algorithm for approximate calculation of the policy parameter (threshold). Numerical results are given.

Keywords—continuous MDP, discretization, job allocation, optimal policy, threshold.

1. Introduction

For high-performance processing systems, consisting of several servers working independently and in parallel one of the fundamental problems is the problem of optimal allocation (or routing) of arriving jobs. Allocation happens at instants of each job arrival and means that job is assigned to one of the servers where it will be served. This decision cannot be undone later. It is assumed that each server has a dedicated queue of infinite capacity where jobs assigned to this server can wait for service.

The optimal allocation (or optimal policy) is the one which provides optimal value of the value function. As the example of simple (but sometimes difficult to compute) value function one can imagine mean sojourn time in the system, tail of the sojourn time distribution. The optimal policy typically depends on value function, service discipline (FIFO, LIFO, PS, etc.) and on the amount of information about the state of the system, which is available at decision instants. One can identify are three main approaches for finding optimal policy for the type of problems described above. The first approach is to choose, based on preliminary qualitative system analysis, the most "promising" policy and then to check the "degree" of its optimality. According to the second approach one chooses the parametrized policy (for example, SITA policy in [1], [2]), then finds the value function under this policy and estimates the values of the policy parameters which provide optimal value of the value function. The third relied on ideas from Markov decision processes and is used in many jobs and resource allocation problems (see for example [3]–[9]). In the majority of the problems the system state space is very complex (for example, due to the need to track elapsed/remaining service times, allow infinite storage capacities, etc.). Thus the class of considered policies is usually reduced to static policies which allow sometimes decomposition of the system and its study in component-wise manner. The are also policies which allow look-ahead actions and still tractable solution (see, for example, [10]).

The problem of finding optimal allocation policy in a heterogeneous two-server system processing fixed size jobs, which is the subject of this paper, has already been considered before and the apparently latest results appear in [11]. In [11] the flow of jobs is Poisson and jobs are served in FCFS manner from queues. The objective is minimization of mean sojourn time in the system. Authors show that this problem is related to the well-known slow-server-problem ([11], [12]). From this observation they derive the following result: optimal allocation policy is of threshold type with one threshold i.e. if upon arrival of the job the amount of unfinished work at faster server (plus total work in its queue) minus the amount of unfinished work at slower server (plus total work in its queue) exceed the threshold value, job is allocated to slow server. The simplicity of the problem formulation and the known (but nonconstructive) answer makes even more sticking the fact that its analytic solution is not known: one can determine the threshold value only using numerical methods. In [11] authors provide one of such methods based on Markov decision processes and Monte-Carlo simulation and also provide several heuristic policies which show near optimal results for the wide range of initial system's values.

The use of Monte-Carlo simulation for the threshold value estimation in the considered problem is greatly complicated by the fact that the curvature of the value function in the neighborhood of its minimum is very low and thus it requires very long simulation time in order to achieve high accuracy.

In this paper a new method for estimation of the threshold value of the optimal policy is provided, which does not rely on any simulation results and is based only on probabilistic arguments and properties of threshold policy. In this respect from one point of view it is free from disadvantages inherent to simulation methods (like those in [11]) and from the other point of view it serves as a case study of efficient handling of Markov decision process problem with continuous state space by discretization.

The paper is organized as follows. In the Section 2 the description of the system is given and the question under study is formulated. Section 3 is devoted to detailed description of the solution method and in Section 4 some numerical results are presented. In conclusion, obtained results are briefly discussed.

2. Description of the System and Problem Formulation

Consider heterogeneous dispatching system with two parallel servers processing fixed size jobs. Jobs inter-arrival times are i.i.d. random variables with known distribution function $F(x)$ with finite mean. Servers are working independently and at constant rates: service rate of one server equals 1 and of the other equals $\nu > 1$. Henceforth, the server working at rate 1 will be referred as server I and to the server working at rate ν as server II. Clearly, time it takes server I and server II to complete one job equals 1 and ν^{-1} respectively. Each server has its own queue (of infinite capacity) and arriving job must be immediately upon arrival assigned (or routed) to one of the queues wherefrom it will be served. For the sake of brevity in what follows authors will refer to the decision to route a job to the queue in front of server I or server II by saying that action 1 or 2 was chosen. No jockeying between queues is allowed. Each server serves jobs only from its own queue on a first-come-first-served basis. Pre-emption is not allowed. The objective is to find the sequence of actions that minimizes mean job sojourn time in the system[1]. It is known that such sequence of actions is fully described by threshold-type policy (see details in, for example [11]). The most interesting is the non-simulation estimation of the value of the policy parameter, i.e. threshold value.

Let us denote by x current workload at server I which equals the number of jobs in the queue in front of server I plus the remaining service of the job in server I. Current workload at server II is denoted by y. Following queueing theory terminology x and y can be understood as virtual waiting times. The evolution of the system in time is fully described by changing values of the pair (x,y) with state space $S = \{(x,y),\ x \in [0,\infty),\ y \in [0,\infty)\}$.

Assume that upon arrival of a job the system is in the state $s = (x,y) \in S$. At this time instant the job must be routed to one of the two queues. If the job is routed to queue in front of the server I (i.e. action 1 is chosen), then at time instant of the next job arrival system's state will be s' equal to

$$s' = ((x+1-\tau)^+, (y-\nu\tau)^+),$$

where τ is the time until next job arrival and $a^+ = \max(0,a)$. The set of states to which transitions from state $s = (x,y)$ can occur is $A_1(s) = \{(x',y'),\ x' = (x+1-t)^+, y' = (y-\nu t)^+,\ t \geq 0\}$. The probability distribution that governs these transitions is denoted by $\mathbf{P}_1(s'|s)$, $s' \in A_1(s)$, $s \in S$. Note that given the distribution $F(x)$ of inter-arrival times, the distribution $\mathbf{P}_1(s'|s)$ can be calculated in straightforward manner.

In case the job is routed to server II (i.e. action 2 is chosen), then at time instant of the next job arrival the state of the system will be s' equal to

$$s' = ((x-\tau)^+, (y+1-\nu\tau)^+).$$

When action 2 is chosen the set of states to which transitions from state $s = (x,y)$ can occur is $A_2(s) = \{(x',y'),\ x' = (x-t)^+,\ y' = (y+1-\nu t)^+,\ t \geq 0\}$. Probability distribution that governs such transitions is denoted by $\mathbf{P}_2(s'|s)$, $s' \in A_2(s)$, $s \in S$. It can be calculated just like $\mathbf{P}_1(s'|s)$.

For fixed s both sets $A_1(s)$ and $A_2(s)$ are one-dimensional. Specifically, each of them is the composition of two line segments: one segment is part of the line with slope ν going through point (x,y) between point (x,y) and intersection of the line with one of the coordinate axes (segment AB in Fig. 1) and the other segment is part of the line from the intersection to point $(0,0)$ (segment OA in Fig. 1).

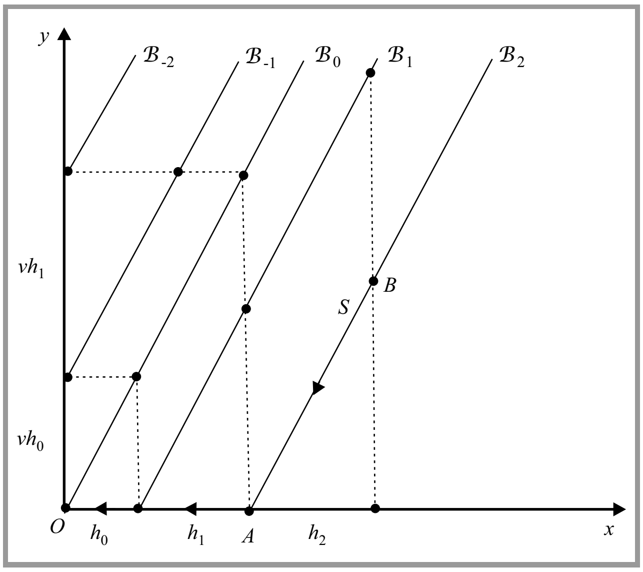

Fig. 1. Discretization of the state space.

[1] Sojourn time for a given job starts from the instant when it arrives at a queue and stops when its service is completed. It is assumed that the decision process does not incur any delay.

Let τ_n, $n \geq 1$, be the arrival instant of the n-th job. Denote by s_n the state of the system at time τ_n but *before* any action is chosen, i.e. before decision where to route the arrived job is made. The authors assume that threshold-type policy is implemented in the system. This implies that at time τ_n one must choose action 1 if

$$s_n \in S^{\xi} = \{(x,y) : \frac{y}{\nu} - x > \xi\} \tag{1}$$

and one must choose action 2 if

$$s_n \in \overline{S}^{\xi} = S \backslash S^{\xi} = \{(x,y) : \frac{y}{\nu} - x \leq \xi\}. \tag{2}$$

Here $\xi \geq 0$ is the parameter of the policy (i.e. threshold value). For detailed discussion of this threshold-type policy one can refer to [11]. Given initial system's state, say $s_1 = (0,0)$, the sequence of s_n, $n \geq 1$, constitutes the Markov chain with transition probabilities

$$\mathbf{P}_{\xi}(s'|s) = \begin{cases} \mathbf{P}_1(s'|s), & \text{if} \quad s \in S^{\xi}, \\ \mathbf{P}_2(s'|s), & \text{if} \quad s \in \overline{S}^{\xi}. \end{cases}$$

Denote the stationary distribution of this Markov chain by π_{ξ}. Given sufficient condition for the stability of the system $(\lambda/(1+\nu)) < 1$, where $\lambda = (\int xF(x))^{-1}$, is satisfied, the stationary distribution exists.

With each state of the Markov chain s_n, $n \geq 1$, one can associate a "reward" $g_n(s_n)$ equal to the sojourn time of the n-th job in the system. If $s_n = (x,y)$ then, due to the fact that jobs are served from queues on FCFS basis, we have

$$g_n(s_n) = \begin{cases} x+1, & \text{if} \quad s_n \in S^{\xi}, \\ \frac{y+1}{\nu}, & \text{if} \quad s_n \in \overline{S}^{\xi}. \end{cases}$$

The limiting expected reward or limiting expected sojourn time T_{ξ} in the system can be defined as

$$T_{\xi} = \int_S g(u)\pi_{\xi}(du).$$

As mentioned above, the most interested is the non-simulation estimation of the value ξ, which minimizes the value of T_{ξ}. Despite the fact that the value function depends only one parameter, the analytical solution of the optimization problem is not known. To author's knowledge there are no analytical results concerning the exact expression and properties of T_{ξ} such as monotonicity, concavity, unimodality, differentiability, which makes impossible the application of standard optimization methods. One of the main solution approaches is the use of simulation in conjunction with ideas of Markov decision processes. This was done in [11], where authors have thoroughly studied the behavior of T_{ξ} experimentally and proposed method for the estimation of the threshold value ξ. But the problem of minimization of T_{ξ} basing only on system's initial parameters ($F(x)$ and ν) without the use of simulation remains open and in the next section fast iterative algorithm is provided, which allows one to find solution with prescribed accuracy.

3. Iterative Algorithm

The idea of the iterative algorithm for computation of the approximate value of the optimal threshold is based on the following observation. Assume the system is in state $\hat{s} = (\hat{x}, \hat{y})$ such that $\frac{\hat{y}}{\nu} - \hat{x} = \xi_{opt}$, where ξ_{opt} is the optimal (still unknown) threshold value. Then the threshold policy introduced in the previous section tells us that action 2 must be chosen. But in fact it is irrelevant, which action one chooses when system is in the state $\hat{s}$. Otherwise the current threshold value is not the true optimal value, because we have to prefer one action to another (and thus the threshold value must be shifted and the value of the value function will be improved[2]). As we don't know the optimal threshold value we fix (almost) arbitrary value $\xi \geq 0$ and assume the system is in the state $\hat{s} = (\hat{x}, \hat{y})$ such that $\frac{\hat{y}}{\nu} - \hat{x} = \xi$. In state $\hat{s}$ two actions can be chosen. Denote by $\sigma_{\xi}^{(1)}$ the policy that chooses action 1 and then follows Eqs. (1)–(2) rule. The policy that at first chooses action 2 and then also follows Eqs. (1)–(2) rule denote by $\sigma_{\xi}^{(2)}$. Let us compare $\sigma_{\xi}^{(1)}$ and $\sigma_{\xi}^{(2)}$. Consider the difference

$$\Delta_{\xi} = g^{(1)} - g^{(2)} + \sum_{n=1}^{\infty} \left(\int_S g(u)\pi_n^{(1)}(du) - \int_S g(u)\pi_n^{(2)}(du) \right), \tag{3}$$

where $g^{(1)}$ and $g^{(2)}$ are rewards for the first action when system is in state $\hat{s}$, and $\pi_n^{(i)}$ is stationary distribution at n-th step of Markov chain (corresponding to fixed value of ξ) given that first action was i, $i = 1,2$. From definition of strategies $\sigma_{\xi}^{(i)}$ it follows that

$$g^{(1)} = \hat{x} + 1, \quad g^{(2)} = \frac{\hat{y}+1}{\nu}. \tag{4}$$

Thus $g^{(1)} - g^{(2)} \neq 0$ and the other terms in (3) are non-zero because the distibutions $\pi_n^{(1)}$ and $\pi_n^{(2)}$ are different for any n. But due to the fact that $\lim_{n\to\infty} \pi_n^{(1)} = \lim_{n\to\infty} \pi_n^{(2)} = \pi_{\xi}$ at exponential rate, the sum in Eq. (3) converges. If $\Delta_{\xi} = 0$ then the value of ξ is the value of the optimal threshold. If $\Delta_{\xi} \neq 0$ then the value of ξ must be increased or decreased depending on the sign of Δ_{ξ}.

The implementation of this idea heavily depends on the opportunity to compute distributions $\pi_n^{(i)}$. The obvious approach is to approximate Markov chain $\{s_n, n \geq 1\}$ by a finite-state Markov chain with transition probability matrix P and use the relation $\pi_n = \pi_{n-1}\mathrm{P}$. If one partitions state space by equal rectangles then the cost to compute with such an approach becomes too high. As experiments show the curvature of function T_{ξ} in the neighborhood of its minimum is very low and thus, in order to obtain suitable results, one has to use very high level of discretization. Eventually matrix P becomes too big (storage requirements become too high) making impossible to use relation $\pi_n = \pi_{n-1}\mathrm{P}$. In the next subsection a new discretization method based on non-uniform grid spacing, which does

[2] Here is implicitly assumed that T_{ξ} is a continuous function of ξ.

not require the calculation of matrix P and gives accurate results is proposed.

3.1. Discretization of the State Space and Construction of Approximating Finite-State Markov Chain

Before constructing finite-state Markov chain $\{\hat{s}_n, n \geq 1\}$, which approximates Markov chain $\{s_n, n \geq 1\}$ discretization of state space S must be performed.

In order to do this some notation need to be introduced. Denote by h_i be the sequence of numbers

$$h_i = h_0(1+\alpha)^i,\ i = 0, 1, \ldots, \tag{5}$$

where $h_0 > 0$ and $\alpha > 0$ are arbitrary small numbers and introduce the following sets:

$$\mathcal{B}_i = \{(x,y) : y = \nu(x - a_i)\},\ i = 0, \pm 1, \pm 2, \ldots, L,$$

where L is arbitrary big whole number and

$$a_i = \begin{cases} 0, & \text{if } i = 0, \\ a_{i-1} + h_{i-1}, & \text{if } i > 0, \\ a_{i+1} - h_{-i-1}, & \text{if } i < 0. \end{cases}$$

The sets $\mathcal{B}_i$ are straight lines with slope ν shifted along the x-axis. Denote also by $\mathcal{C}_j^+$ and $\mathcal{C}_j^-$ the following sets:

$$\begin{aligned} \mathcal{C}_j^+ &= \{(x,y) : x = a_j\},\ j = 1, 2, \ldots, L, \\ \mathcal{C}_j^- &= \{(x,y) : y = \nu a_j\},\ j = 1, 2, \ldots, L. \end{aligned}$$

Define the set of points $\tilde{S}^{\alpha,L}$ as union of the following sets

$$\tilde{S}^{\alpha,L} = \{\tilde{S}_{00}\} \cup \{\tilde{S}_{ij},\ i = 0, \pm 1, \pm 2, \ldots, L; j = 1, 2, \ldots, L\},$$

where $\tilde{S}_{00} = (0,0)$, $\tilde{S}_{ij} = \mathcal{B}_i \cap \mathcal{C}_j^+$ if $i \geq 0$ and $\tilde{S}_{ij} = \mathcal{B}_i \cap \mathcal{C}_j^-$ if $i < 0$. The set of points $\tilde{S}^{\alpha,L}$ consists of $(L+1)^2$ points and represents the grid, which covers the rectangle area of the first quadrant of the xy-plane. One vertex of the rectangle coincides with $(0,0)$ and sides along the x-axis and y-axis equal H and νH respectively, where

$$H = \sum_{i=0}^{L-1} h_i = \frac{h_0(1+\alpha)^L - h_0}{\alpha}. \tag{6}$$

The points in the set $\tilde{S}^{\alpha,L}$ are distributed non-uniformly (Fig. 1). As one moves towards the origin and line $y = \nu x$ (set $\mathcal{B}_0$) the concentration increases. As one move in the opposite direction the concentration goes down.

The set of points $\tilde{S}^{\alpha,L}$ is used to construct state space of approximating finite-state Markov chain $\{\hat{s}_n, n \geq 1\}$. The following argumentation follows from the description of the sets $A_1(s)$ and $A_2(s)$ given in the previous section.

Assume that *after* decision on the n-th step Markov chain $\{s_n, n \geq 1\}$ was in state $s = (x,y) \in \mathcal{B}_i$ (see Fig. 1). Then until the arrival instant of the next job system's state, i.e. values of pair (x,y) will "belong" to the line indicated with the arrow in Fig. 1. The start point of the route is s and finish point is $A = (0,0)$ which means that system is empty. By the arrival of $(n+1)$-th customer the system may be at any point *only* on this route. As the continuous state space of Markov chain $\{s_n, n \geq 1\}$ is discretized, then this route must consist of finite number of points.

Remark 1. The way in which the grid $\tilde{S}^{\alpha,L}$ was constructed tells that the length of any segment (either vertical or horizontal, or slanted) of arbitrary route equals h_i. It can easily be seen that time it takes system to pass a segment also equals h_i.

Now one needs to define the set of possible routes. In order to do this the following sets are defined:

$$\mathcal{A}_0 = \mathcal{B}_0 \cap \tilde{S}^{\alpha,L}, \tag{7}$$

$$\mathcal{A}_i = (\mathcal{B}_i \cap \tilde{S}^{\alpha,L}) \cup \{\tilde{S}_{i-1,i-1}, \ldots, \tilde{S}_{1,1}, \tilde{S}_{0,0}\},\ i > 0, \tag{8}$$

$$\mathcal{A}_i = (\mathcal{B}_i \cap \tilde{S}^{\alpha,L}) \cup \{\tilde{S}_{i+1,-i-1}, \ldots, \tilde{S}_{-1,1}, \tilde{S}_{0,0}\},\ i < 0. \tag{9}$$

In Eqs. (8)–(9) each set in parentheses contains points from the set $\tilde{S}^{\alpha,L}$ which belong to slanted segment $\mathcal{B}_i$. The set in braces contains points of the line connecting origin O and intersection of $\mathcal{B}_i$ with one of the coordinate axes. The routes $\mathcal{A}_i$ can be also represented in a different way:

$$\begin{aligned} \mathcal{A}_0 &= \{\tilde{S}_{0,L}, \tilde{S}_{0,L-1}, \ldots, \tilde{S}_{0,0}\}, \\ \mathcal{A}_i &= \{\tilde{S}_{i,L}, \tilde{S}_{i,L-1}, \ldots, \tilde{S}_{i,i}, \tilde{S}_{i-1,i-1}, \ldots, \tilde{S}_{1,1}, \tilde{S}_{0,0}\},\ i > 0, \\ \mathcal{A}_i &= \{\tilde{S}_{i,L}, \tilde{S}_{i,L-1}, \ldots, \tilde{S}_{i,-i}, \tilde{S}_{i+1,-i-1}, \ldots, \tilde{S}_{-1,1}, \tilde{S}_{0,0}\},\ i < 0. \end{aligned}$$

Any discretized route, just like OAB depicted in Fig. 1, is the subset of $\mathcal{A}_i$. The elements of $\mathcal{A}_i$ can be enumerated in a natural way, starting from point $\tilde{S}_{0,0}$. For $i \geq 0$ it holds that

$$\begin{aligned} S_{i,0} &= \tilde{S}_{0,0}, S_{i,1} = \tilde{S}_{1,1}, \ldots, S_{i,i-1} = \tilde{S}_{i-1,i-1}, \\ S_{i,i} &= \tilde{S}_{i,i}, \ldots S_{i,L-1} = \tilde{S}_{i,L-1}, S_{i,L} = \tilde{S}_{i,L}, \end{aligned}$$

and for $i < 0$

$$\begin{aligned} S_{i,0} &= \tilde{S}_{0,0}, S_{i,1} = \tilde{S}_{-1,1}, \ldots, S_{i,-i-1} = \tilde{S}_{i+1,-i-1}, \\ S_{i,-i} &= \tilde{S}_{i,-i}, \ldots S_{i,L-1} = \tilde{S}_{i,L-1}, S_{i,L} = \tilde{S}_{i,L}. \end{aligned}$$

Thus for any i the route $\mathcal{A}_i$ can be represented as $\mathcal{A}_i = \{S_{i,0}, S_{i,1}, \ldots, S_{i,L}\}$. As the state space $S^{\alpha,L}$ of the approximating finite-state Markov chain $\{\hat{s}_n, n \geq 1\}$ we will take the union of possible routes, i.e. $S^{\alpha,L} = \cup_{i=-L}^{L} \mathcal{A}_i$. The size of the set $S^{\alpha,L}$ is $(L+1)(2L+1)$, which is greater than the size of the set $\tilde{S}^{\alpha,L}$. This is due to the fact that some points of the grid $\tilde{S}^{\alpha,L}$ belong to different routes $\mathcal{A}_i$ at the same time. Such points are those which lie on coordinate axes (excluding extreme points). For example the route $\mathcal{A}_i$ includes point $S_{i,0} = \tilde{S}_{0,0}$ corresponding to empty state of the system. Such duplication may seem unnatural but, as will be shown further, it greatly simplifies the calculation of transition probabilities.

Now let us dwell on description of transitions of approximating Markov chain $\{\hat{s}_n, n \geq 1\}$. Let at the time of the n-th job arrival the system be in the state $\hat{s}_n = S_{ij} \in S^{\alpha,L}$ and assume that after a decision the system entered state

$S_{kl} \in S^{\alpha,L}$[3]. The state S_{ij} (point on the grid) to which transition from state S_{ij} occurs is completely defined by threshold policy and there is one-to-one correspondence between indexes k and i (l and j, as well). After transition to state S_{kl} system evolves deterministically until the next arrival. At next arrival instant system finds itself in the new state $\hat{s}_{n+1}$, which coincides with one of the points S_{kl}, $S_{k,l-1}$, …, S_{k0} of the grid. From description of the set $S^{\alpha,L}$ and *Remark 1* it follows that the transition probabilities $S_{kl} \to S_{km}$, $m = 0,1,\dots,l$, depend only on index l and do not depend on index k. Let us denote these probabilities by q_{lm}, i.e. $q_{lm} = \mathbf{P}\{S_{kl} \to S_{km}\}$. Clearly $q_{00} = 1$. Let $l = 1$. From system standpoint it means that there is unfinished work in the system equal to h_0. Due to the fact that the state space at the instant of the next job arrival have been discretized, there are only two options: either unfinished work in the system will be the same (say, with probability q_{11}), or the system will be empty (with probability $1 - q_{11} = q_{10}$). The value of q_{11} may be taken equal to probability that inter-arrival time does not exceed $0.5h_0$, i.e. $q_{11} = F(0.5h_0)$[4]. By the same argument the following expression for arbitrary value of $l = 1,\dots,L$ is obtained:

$$q_{lm} = F(H_{l-m+1}) - F(H_{l-m}),\ m = 0,\dots,l,$$

where

$$H_m = \begin{cases} 0, & \text{if } m=0,\\ h_{l-1}+h_{l-2}+\ldots+h_{l-m+1}+0.5h_{m-1}, & \text{if } m=1,\dots,l,\\ 1, & \text{if } m=l+1\,. \end{cases}$$

3.2. Description of the Iterative Procedure

In order to be able to compute transition probabilities of approximating Markov chain $\{\hat{s}_n, n \geq 1\}$ one has to know how to jump from bevel coordinates given by indexes of elements S_{ij} to rectangular coordinates $(x,y) \in S$ and back. This transform follows directly from the way the sets $S^{\alpha,L}$ was constructed. Let x and y be rectangular coordinates of point $S_{ij} \in S^{\alpha,L}$. If $i = 0$, then clearly $x = y = 0$. For $i > 0$ it holds that

$$x = h_0 \frac{(1+\alpha)^j - 1}{\alpha}, \tag{10}$$

$$y = \max\Big(0, \nu x - \nu h_0 \frac{(1+\alpha)^i - 1}{\alpha}\Big), \tag{11}$$

and for $i < 0$

$$x = \max\Big(0, \frac{y}{\nu} - h_0 \frac{(1+\alpha)^{-i} - 1}{\alpha}\Big), \tag{12}$$

$$y = \nu h_0 \frac{(1+\alpha)^j - 1}{\alpha}. \tag{13}$$

The inverse transform is not unique. This is because there are different ways in which one can choose point $S_{ij} \in S^{\alpha,L}$, which approximates point $(x,y) \in S$. For example, one can use the following rule:

$$i = \max\big(-L, \min(L, i')\big), \tag{14}$$

$$j = \max\big(-L, \min(L, j')\big), \tag{15}$$

where

$$i' = \operatorname{sign}\Big(x - \frac{y}{\nu}\Big) \left\lfloor \frac{\ln\big(1 + \frac{\alpha}{h_0}|x - \frac{y}{\nu}|\big)}{\ln(1+\alpha)} \right\rfloor,$$

$$j' = \begin{cases} \left\lfloor \frac{\ln\left(1+\frac{\alpha x}{h_0}\right)}{\ln(1+\alpha)} \right\rfloor, & \text{if } y < \nu x,\\ \left\lfloor \frac{\ln\left(1+\frac{\alpha y}{h_0 \nu}\right)}{\ln(1+\alpha)} \right\rfloor, & \text{if } y \geq \nu x, \end{cases}$$

where $\operatorname{sign}(a)$ denotes signum function and $\lfloor a \rfloor$ denotes integer part of a.

Assume Markov chain $\{s_n, n \geq 1\}$ generated by threshold policy ξ is in state $s = (x,y)$ at the time of n-th arrival. Then after a decision it will move to state

$$(\tilde{x},\tilde{y}) = \begin{cases} (x+1,y), & \text{if } \frac{y}{\nu} - x > \xi,\\ (x,y+1), & \text{if } \frac{y}{\nu} - x \leq \xi. \end{cases} \tag{16}$$

Let the approximating Markov chain $\{\hat{s}_n, n \geq 1\}$ be in state $\hat{s} = (\hat{x},\hat{y})$ such that $\frac{\hat{y}}{\nu} - \hat{x} = \xi$. Consider again policies $\sigma_\xi^{(1)}$ and $\sigma_\xi^{(2)}$ introduced at the beginning of Section 3 and denote by $\hat{\pi}_n^{(1)}$ and $\hat{\pi}_n^{(2)}$ respectively stationary distribution over the state space $S^{\alpha,L}$ under these policies. The discrete version of the difference Δ_ξ, introduced in Eq. (3), is given by

$$\Delta_\xi^{\alpha,L} = g^{(1)} - g^{(2)} + \sum_{n=1}^{\infty} \sum_i \sum_j g(i,j)\big(\hat{\pi}_n^{(1)}(i,j) - \hat{\pi}_n^{(2)}(i,j)\big),$$

where $g^{(1)}$ and $g^{(2)}$ are computed from Eq. (4), $g(i,j) = g(S_{ij})$, and $\hat{\pi}_n^{(i)}(i,j)$ are the values of the distributions $\hat{\pi}_n^{(1)}$ and $\hat{\pi}_n^{(2)}$ at point S_{ij}.

The step-by-step procedure for the update of the value $\Delta_\xi^{\alpha,L}$ is given below in Algorithm 1. It also shows how stationary distributions $\hat{\pi}_n^{(i)}$ can be calculated on the fly.

The x_{ij}, y_{ij} are rectangular coordinates of point S_{ij}, calculated from Eqs. (10)–(13), I_{xy}, J_{xy} are indexes of inverse transform calculated from Eqs. (14)–(15) and $\tilde{x} = \tilde{x}(x,y)$, $\tilde{y} = \tilde{y}(x,y)$ are given by Eq. (16).

Remark 2. Algorithm 1 is only the basic version which can be modified in order to improve its efficiency. For example, one can shift the area of the grid $\tilde{S}^{\alpha,L}$ where the most points are concentrated from the neighborhood of $(0,0)$ (which is the case in Algorithm 1) to the neighborhood of the more frequent states of the system. Such states can be determined using simulation.

Remark 3. Proposed algorithm allows one to check whether the chosen value of threshold ξ is the optimal value. Algorithm 1 does not contain the description of the exact

[3] Note that the authors are working under assumption that transitions $S_{ij} \to S_{kl}$ do not incur any delay.

[4] This value is taken by an agreement. The is no other reasoning behind this choice except for common sense.

Algorithm 1: Algorithm for computation of steady state probabilities and approximating value of Δ_ξ

```
Step 1
Initialize $\Delta_0 = g^{(1)} - g^{(2)}$;
if $i = i_{\hat{x},\hat{y}}$ and $j = j_{\hat{x},\hat{y}}$ then
    $\hat{\pi}_0^{(1)}(i,j) = 1$, $\hat{\pi}_0^{(2)}(i,j) = 0$;
else
    $\hat{\pi}_0^{(1)}(i,j) = 0$, $\hat{\pi}_0^{(2)}(i,j) = 1$;
end if
Step 2
$x = \hat{x} + 1$, $y = \hat{y}$;
$k = I_{xy}$, $l = J_{xy}$;
$\hat{\pi}_1^{(1)} = 0$;
for $m = 0$ to $l$ do
    $\hat{\pi}_1^{(1)}(k,m) = \hat{\pi}_1^{(1)}(k,m) + q_{lm}\hat{\pi}_0^{(1)}(k,l) = 0$; // Compute initial state probabilities after action 1
end for
$x = \hat{x}$, $y = \hat{y} + 1$;
$k = I_{xy}$, $l = J_{xy}$;
$\hat{\pi}_1^{(2)} = 0$;
for $m = 0$ to $l$ do
    $\hat{\pi}_1^{(2)}(k,m) = \hat{\pi}_1^{(2)}(k,m) + q_{lm}\hat{\pi}_0^{(2)}(k,l) = 0$; // Compute initial state probabilities after action 2
end for
$\Delta\hat{\pi}_1 = \hat{\pi}_1^{(1)}(k,m) - \hat{\pi}_1^{(2)}(k,m)$; // component-wise difference
$\Delta_1 = \Delta_0 + \sum_{n=1}^{\infty}\sum_{i=-L}^{L}\sum_{j=0}^{L} g(i,j)\Delta\hat{\pi}_1(i,j)$;
$n = 1$;
Step 3
$n = n + 1$;
for $i = -L$ to $L$ do
    for $j = 0$ to $L$ do
        $x = x_{ij}$, $y = y_{ij}$; // rectangular coordinates of point $S_{ij}$
        $k = i_{\tilde{x}\tilde{y}}$, $l = j_{\tilde{x}\tilde{y}}$; // index values after making decision
        $\Delta\hat{\pi}_n = 0$;
        for $m = 0$ to $l$ do
            $\Delta\hat{\pi}_n = \Delta\hat{\pi}_n + q_{lm}\hat{\pi}_{n-1}^{(1)}(k,l) - q_{lm}\hat{\pi}_{n-1}^{(2)}(k,l)$
        end for
    end for
end for
$\Delta_n = \Delta_{n-1} + \sum_{n=1}^{\infty}\sum_{i=-L}^{L}\sum_{j=0}^{L} g(i,j)\Delta\hat{\pi}_n(i,j)$;
if $|\Delta_n - \Delta_{n-1}| < \varepsilon$ then // $\varepsilon$ – parameter of the algorithm
    goto Step 3;
else
    $\Delta_\xi^{\alpha,L} = \Delta_n$.
end if
```

procedure for the calculation of the threshold because it can be performed in different ways. For example, one can choose (using qualitative analysis of the system behavior) interval which contains the (unknown) value of threshold ξ. For example, in current setting this interval is $(0, \nu^{-1})$. Then use bisection method can be applied.

4. Numerical Example

Let us give simple comparison of results, which were obtained from proposed algorithm with results obtained from Monte-Carlo simulation.

Let the service rate of server II be equal to $\nu = 2$. The threshold value of the optimal policy for two types of inter-arrival distributions is then computed: exponential with parameter $\lambda = 2.4$ and Pareto with scale $b = 0.21$ and shape $a = (1 - \lambda b)^{-1} \approx 2.016$. Both these distributions have equal mean inter-arrival times but their variances differ significantly. For exponential distribution the variance is $\lambda^{-2} \approx 0.417$ and for Pareto it is $ab^2/[(a-1)^2(a-2)]) \approx$ ≈ 25.43.

In order to construct the grid $\tilde{S}^{\alpha,L}$ let us fix the minimum and maximum grid spacing by letting $h_0 = 0.005$ and $h_{L-1} = 0.025$. Let the total length of the approximating area along the x-axis be $H = 10$. Given the value of h_0, h_{L-1} and Hm, other parameters of the grid can be calculated from Eqs. (5) and (6). That is

$$\alpha = \frac{h_{L-1} - h_0}{H - h_{L-1}} \approx 0.0001, \quad L = \left\lfloor \frac{\ln\left(1 + \frac{\alpha H}{h_0}\right)}{\ln(1+\alpha)} \right\rfloor \approx 1600.$$

The total number of states after discretization is

$$(L+1)(2L+1) \approx 5.2 \times 10^6.$$

Having applied iterative algorithm described in Section 3 we obtained that for exponential inter-arrival times the optimal threshold ξ_{opt} lies in the interval $(0.166, 0.167)$ and for Pareto inter-arrival times the interval is $(0.150, 0.151)$. In order to understand how accurate these results are, let us have a look at the value of value function T_ξ (estimated from Monte-Carlo simulation) for the threshold values ξ, which are in the neighborhood of the obtained intervals. The results are given in Table 1.

Table 1
(Approximate) values of the value function T_ξ in the neighborhood of the intervals, containing optimal threshold

Exponential inter-arrival times		Pareto inter-arrival times	
Threshold ξ	Mean T	Threshold ξ	Mean T
0.160	1.25459	0.144	0.93638
0.162	1.25458	0.146	0.93637
0.164	1.25456	0.148	0.93636
0.166	1.25454	0.150	0.93636
0.168	1.25454	0.152	0.93636
0.172	1.25454	0.154	0.93637
0.174	1.25455	0.156	0.93638

One can see from Table 1 that the proposed algorithm gives good results up to (and including) the third digit after the decimal point. In order to check the value of the fourth digit one has to be able to estimate value function T_ξ from Monte-Carlo simulation up to the sixth digit after the decimal point. Such estimation is far from being simple because estimation of T_ξ up to fifth digit already takes several hours on standard PC. Meanwhile the proposed algorithm finds the interval (up to third digit after decimal point), which contains optimal threshold value usually in 5–10 minutes.

5. Conclusion

As it is mentioned in many research papers quite a few problems which one may encounter in practice (for example, building schedulers in distributed processing systems) can be formulated in terms of flows, servers, queues. The considered problem is only the special case of far more general model which may encompass many details of real-life systems and the need for appropriate solution methods seems to be high. At present the most popular "attack" method for such problems is the use of heuristics and their validation using simulation. Even for the considered special case the non-simulation solution is far from being simple (and exact solution is not known at all). Analytic solution methods for arbitrary $n > 2$ number of servers have not yet been developed and the structure of optimal policy is not known. It must not necessarily be of threshold type. Although if one decides that threshold policy should be used in the n-server system, then the proposed algorithm can be scaled in a straightforward manner, but the obtained results may not be optimal. Here one of the appealing ways to check the quality of the solution is again the comparison with simulation. Our experiments show that Monte-Carlo simulation in combination with adaptive algorithms for partially observed Markov chains is the most suitable approach for this purpose.

Acknowledgements

This work was supported by the Russian Foundation for Basic Research (grant 15-07-03406).

References

[1] M. E. Crovella, M. Harchol-Balter, and C. D. Murta, "Task assignment in a distributed system: Improving performance by unbalancing load", in *Proc. of Joint Int. Conf. on Measur. and Model. of Comp. Syst. SIGMETRICS'98/PERFORMANCE '98*, Madison, Wisconsin, USA, 1998, pp. 268–269.

[2] M. Harchol-Balter, M. E. Crovella, and C. D. Murta, "On choosing a task assignment policy for a distributed server system", *J. of Parallel and Distrib. Comput.*, vol. 59, pp. 204–228, 1999.

[3] S. Bhulai, "On the value function of the M/Cox(r)/1 queue", *J. of Appl. Probab.*, vol. 43, no. 2, pp. 363–376, 2006.

[4] E. Hyytiä, J. Virtamo, S. Aalto, and A. Penttinen, "M/M/1-PS queue and size-aware task assignment", *Perform. Eval.*, vol. 68, no. 11, pp. 1136–1148, 2011.

[5] E. Hyytiä, A. Penttinen, and S. Aalto, "Size- and state-aware dispatching problem with queue-specific job sizes", *Eur. J. of Oper. Res.*, vol. 217, no. 2, pp. 357–370, 2012.

[6] E. Hyytiä, S. Aalto, and A. Penttinen, "Minimizing slowdown in heterogeneous size-aware dis-patching systems", in *Proc. of the 12th ACM SIGMETRICS/Performance Conf. Measur. Model. Comp. Syst.*, London, UK, 2012, *Eval. Rev.*, vol. 40, pp. 29–40, 2012.

[7] J. Kołodziej and F. Xhafa, "Enhancing the genetic-based scheduling in computational grids by a structured hierarchical population", *Future Gener. Comp. Syst.*, vol. 27, no. 8. pp. 1035–1046, 2011.

[8] K. R. Krishnan and T. J. Ott, "State-dependent routing for telephone traffic: Theory and results", in *Proc. 25th IEEE Conf. on Decision and Control*, Athens, Greece, 1986, vol. 25, pp. 2124–2128.

[9] M. G. Konovalov, "About one task of overload control", *Informatics and its Applications*, vol. 7, no. 4, pp. 34–43, 2013 (in Russian).

[10] E. Hyytiä, "Lookahead actions in dispatching to parallel queues", *Perform. Eval.*, vol. 70, no. 10, pp. 859–872, 2013.

[11] E. Hyytiä, "Optimal routing of fixed size jobs to two parallel servers", *INFOR: Inform. Syst. and Oper. Res.*, vol. 51, no. 4, pp. 215–224, 2013.

[12] R. L. Larsen, "Control of multiple exponential servers with application to computer systems", Ph.D. thesis, University of Maryland at College Park, College Park, MD, USA, 1981.

[13] K. R. Krishnan, "Markov decision algorithms for dynamic routing", *IEEE Commun. Mag.*, vol. 28, no. 10, pp. 66–69, 1990.

[14] J. Leeuwaarden, S. Aalto, and J. Virtamo, "Load balancing in cellular networks using first policy iteration", Tech. Rep., Networking Laboratory, Helsinki University of Technology, 2001.

[15] K. R. Krishnan, "Joining the right queue: a state-dependent decision rule", *IEEE Trans. on Autom. Contr.*, vol. 35, no. 1, pp. 104–108, 1990.

[16] S. A. E. Sassen, H. C. Tijms, and R. D. Nobel, "A heuristic rule for routing customers to parallel servers", *Statistica Neerlandica*, vol. 51, no. 1, pp. 107–121, 1997.

Mikhail Konovalov, Ph.D., is the head of the sector at Information Technologies Department at Institute of Informatics Problems FRC CSC RAS. His interests include adaptive control of random sequences, modeling and simulation of complex systems.

E-mail: mkonovalov@ipiran.ru
Institute of Informatics Problems
Federal Research Center "Computer Science and Control"
Russian Academy of Sciences
Vavilova st 44-2
119333 Moscow, Russia

Rostislav Razumchik received his Ph.D. degree in Physics and Mathematics in 2011. Since then, he has worked as a senior research fellow at Institute of Informatics Problems FRC CSC RAS. His current research activities focus on stochastic processes and queuing theory.

E-mail: rrazumchik@ipiran.ru
Institute of Informatics Problems
Federal Research Center "Computer Science and Control"
Russian Academy of Sciences
Vavilova st 44-2
119333 Moscow, Russia