Optimal Stopping in the Balls-and-Bins Problem[*]

Anna A. Ivashko

Institute of Applied Mathematical Research,
Karelian Research Centre RAS,
Pushkinskaya str. 11, Petrozavodsk, 185910, Russia,
E-mail: aivashko@krc.karelia.ru
WWW home page: http://www.krc.karelia.ru/HP/aivashko
URL: http://mathem.krc.karelia.ru

Abstract This paper considers a multistage balls-and-bins problem with optimal stopping connected with the job allocation model. There are N steps. The player drops balls (tasks) randomly one at a time into available bins (servers). The game begins with only one empty bin. At each step, a new bin can appear with probability p. At step n $(n = 1, \ldots, N)$, the player can choose to stop and receive the payoff or continue the process and move to the next step. If the player stops, then he/she gets 1 for every bin with exactly one ball and loses $1/2$ for every bin with two or more balls. Empty bins do not count. At the last step, the player must stop the process. The player's aim is to find the stopping rule which maximizes the expected payoff. The optimal payoffs at each step are calculated. An approximate strategy depending on the number of steps is proposed. It is demonstrated that the payoff when using this strategy is close to the optimal payoff.

Keywords: optimal stopping, job allocation, balls-and-bins problem

1. Introduction

The job allocation problem is of high relevance in networking technologies. There are different schemes that describe the arrival of jobs or computing tasks to servers for the following processing.

Problems from the probability theory, game theory, and optimal stopping theory are often used for modelling different schemes. In real-life job allocation situations, it is often necessary to stop the process, for example to rearrange the system. Optimal stopping problems can be useful for such models.

Let there be a system in which computing tasks (jobs) arrive to servers randomly, one at a time. There is a total of N time instances. We assume that one new server can arrive in the system at each instant. The time instant has to be chosen for stopping the process. The gain during a stop depends on the system loading, i.e. we get 1 for each server with exactly one job and lose $1/2$ for each server with two or more jobs. Idle servers are disregarded in calculating the payoff. The aim is to stop the process so as to obtain the maximum expected payoff.

This problem is connected with classical probabilistic problems, such as the ballot problem and the balls-and-bins problem.

In the balls-and-bins problem (Kolchin, 1978) we have m balls and n boxes (or "bins"). Each time, a single ball is placed into one of the bins. After all balls are in the bins, we look at the number of balls in each bin. We call this number the load on the bin and we want to find the maximum load on a single bin.

The balls-and-bins models have many interesting applications to random allocation, hashing and load balancing in computer science.

Another noteworthy class of probabilistic schemes in which the player's payoff depends on the state of the system in previous periods is urn schemes. One of them is the ballot problem.

In the Ballot problem (Shep, 1969; Tamaki, 2001) we have an urn containing balls with $+1$ and -1 in it. We draw balls randomly, one at a time, without replacement until we wish to stop. We know the values of m (balls with -1) and p (balls with $+1$) and are also allowed not to draw at all. Payoff is the sum of values of balls.

One of the applications for the ballot problem is the bond-selling problem.

Tijms, 2017 studied the following optimal stopping model for the balls-and-bins problem. Balls are dropped one at a time and any ball will land at random into one of b bins. You can stop dropping balls whenever you wish. At the end of the game, you win 1 for every bin with exactly one ball and you lose half a dollar for every bin containing $m \geq 2$ balls. Empty bins do not count. We need to maximize the expected payoff.

Below, a new statement of the balls-and-bins problem with optimal stopping is proposed. The model suggested by Tijms, 2017 is generalized to a multistage problem in which one new empty bin can arrive at each step. Optimality equations are made for each step of the problem. We propose a new form of strategy and present the numerical results showing that this strategy is near-optimal.

This paper is structured as follows. Section 2 gives the statement of the balls-and-bins problem. In Sections 3 and 4 suggest an analytical solution of the problem for the last two steps and describe the solution procedure for the remaining steps. Construction of a strategy approximated to the optimal one and numerical modeling are reported in Section 5. Finally, in Section 6, we present the findings and conclusions, and draft plans for the future.

2. Optimal stopping for a multistage balls-and-bins problem

Consider an optimal stopping bins-and-balls problem connected with the job allocation problem. Let there be a system with an arriving flow of computing tasks to servers.

We present a multistage model in which at each step the player drops balls (tasks) randomly one at a time into available bins (servers). The game begins with only one empty bin. At each step, a new bin can appear with probability p. At step n ($n = 1, ..., N$), the player can choose to stop and receive the payoff or continue the process and move to the next step. If the player stops, then he/she gets 1 for every bin with exactly one ball and loses $1/2$ for every bin with two or more balls. Empty bins do not count. At the last step, the player must stop the process. The player's aim is to find the stopping rule which maximizes the expected net gain.

Consider payoffs at each step of the game.

At each step n ($n = 1, ..., N$) we have the finite set of states $S = \{(i_0, i_1; k) : i_0, i_1 \geq 0, i_0 + i_1 \leq k\}$, $k = 1, ..., n + 1$. Let state $(i_0, i_1; k)$ means that there are k bins in the game: i_0 empty bins, i_1 bins with exactly one ball and $k - i_0 - i_1$ bins with two or more balls.

We write $v_n(i_0, i_1; k)$, $k = 1, ..., n+1, n = 1, ..., N$ to denote the payoff at step n in this problem. Let $u_n(i_0, i_1; k)$ be the payoff if the player stops and $\bar{v}_n(i_0, i_1; k)$ be the expected payoffs if the player continues the process. Then,

$$v_n(i_0, i_1; k) = \max\left\{ u_n(i_0, i_1; k); \bar{v}_n(i_0, i_1; k) \right\}, \ n = 1, ..., N-1, \ k = 1, ..., n+1;$$
(1)

$$v_N(i_0, i_1; k) = i_1 - \frac{1}{2}(k - i_0 - i_1), \ k = 1, ..., N+1; \ 0 \le i_0 + i_1 \le k,$$

where
$$u_n(i_0, i_1; k) = i_1 - \frac{1}{2}(k - i_0 - i_1),$$

$$\bar{v}_n(i_0, i_1; k) = p\left[\frac{i_0 + 1}{k + 1} v_{n+1}(i_0, i_1 + 1; k + 1) + \frac{i_1}{k + 1} v_{n+1}(i_0 + 1, i_1 - 1; k + 1) \right.$$
$$+ \frac{k - i_0 - i_1}{k + 1} v_{n+1}(i_0 + 1, i_1; k + 1) \Bigg] + (1 - p)\left[\frac{i_0}{k} v_{n+1}(i_0 - 1, i_1 + 1; k) \right.$$
$$+ \frac{i_1}{k} v_{n+1}(i_0, i_1 - 1; k) + \frac{k - i_0 - i_1}{k} v_{n+1}(i_0, i_1; k) \Bigg].$$

The optimal value function at the beginning of the process has the form

$$v_0(1, 0; 1) = p v_1(1, 1; 2) + (1 - p) v_1(0, 1; 1).$$

3. Optimal strategy at step $N - 1$

Let us consider the last step in the problem and find the player's optimal behavior.

The game ends at step N, and it is necessary to stop. Then, the payoff is

$$v_N(i_0, i_1; k) = u_N(i_0, i_1; k) = i_1 - \frac{1}{2}(k - i_0 - i_1), \ k = 1, ..., N+1.$$

At step $N - 1$, it is necessary to decide whether to stop or to continue the game. If we stop, the payoff is

$$u_{N-1}(i_0, i_1; k) = i_1 - \frac{1}{2}(k - i_0 - i_1), \ k = 1, ..., N,$$

and if we continue, we gain

$$\bar{v}_{N-1}(i_0, i_1; k) = p\left[\frac{i_0 + 1}{k + 1} v_N(i_0, i_1 + 1; k + 1) + \frac{i_1}{k + 1} v_N(i_0 + 1, i_1 - 1; k + 1) \right.$$
$$+ \frac{k - i_0 - i_1}{k + 1} v_N(i_0 + 1, i_1; k + 1) \Bigg] + (1 - p)\left[\frac{i_0}{k} v_N(i_0 - 1, i_1 + 1; k) \right.$$
$$+ \frac{i_1}{k} v_N(i_0, i_1 - 1; k) + \frac{k - i_0 - i_1}{k} v_N(i_0, i_1; k) \Bigg],$$

$k = 1, ..., N$.
Since $v_N(i_0 + 1, i_1; k + 1) = v_N(i_0, i_1; k)$, we get

$$\bar{v}_{N-1}(i_0, i_1; k) = v_N(i_0 - 1, i_1 + 1; k) \left[p\frac{i_0 + 1}{k + 1} + (1 - p)\frac{i_0}{k} \right]$$

$$+ v_N(i_0, i_1 - 1; k) \left[p\frac{i_1}{k+1} + (1-p)\frac{i_1}{k} \right] + v_N(i_0, i_1; k) \left[p\frac{k - i_0 - i_1}{k + 1} + (1 - p)\frac{k - i_0 - i_1}{k} \right]$$

$$= v_N(i_0 - 1, i_1 + 1; k)\frac{pk + i_0(k + 1 - p)}{k(k + 1)} + v_N(i_0, i_1 - 1; k)\frac{i_1(k + 1 - p)}{k(k + 1)}$$

$$+ v_N(i_0, i_1; k)\frac{(k - i_0 - i_1)(k + 1 - p)}{k(k + 1)}.$$

Eventually, we get

$$v_{N-1}(i_0, i_1; k) = \max\left\{ i_1 - \frac{1}{2}(k - i_0 - i_1); v_N(i_0 - 1, i_1 + 1; k)\frac{pk + i_0(k + 1 - p)}{k(k + 1)} \right.$$

$$\left. + v_N(i_0, i_1 - 1; k)\frac{i_1(k + 1 - p)}{k(k + 1)} + v_N(i_0, i_1; k)\frac{(k - i_0 - i_1)(k + 1 - p)}{k(k + 1)} \right\} \qquad (2)$$

$$= i_1 - \frac{1}{2}(k - i_0 - i_1) + \max\left\{ 0; \frac{pk + \left(i_0 - \frac{3}{2}i_1 \right)(k + 1 - p)}{k(k + 1)} \right\}.$$

Here $0 \leq i_0 + i_1 \leq k$, $k = 1, 2, ..., N$.

Notice that $\max\left\{ 0; \dfrac{pk + \left(i_0 - \dfrac{3}{2}i_1 \right)(k + 1 - p)}{k(k + 1)} \right\} < 1$

and $\dfrac{pk + \left(i_0 - \dfrac{3}{2}i_1 \right)(k + 1 - p)}{k(k + 1)} > 0$ for $\dfrac{3}{2}i_1 - i_0 < \dfrac{pk}{k + 1 - p}$.

Hence, stopping at step $N - 1$ is optimal if

$$i_1 \geq \frac{2}{3}\left(i_0 + \frac{pk}{k + 1 - p} \right).$$

4.   Optimal strategy at step n, $n = 1, ..., N - 2$

To determine the optimal strategy at each step n, $n = 1, ..., N - 2$ consider the payoffs from stopping or continuing the process.

The payoff at step $N - 2$ has the form

$$v_{N-2}(i_0, i_1; k) = \max\left\{ u_{N-2}(i_0, i_1; k); \bar{v}_{N-2}(i_0, i_1; k) \right\}, \ k = 1, ..., N - 1.$$

In such case, the payoff if we stop is

$$u_{N-2}(i_0, i_1; k) = i_1 - \frac{1}{2}(k - i_0 - i_1), \ k = 1, ..., N - 1,$$

and the payoff if we continue is calculated by the following formula:

$$\bar{v}_{N-2}(i_0, i_1; k) = p\left[\frac{i_0+1}{k+1}v_{N-1}(i_0, i_1+1; k+1) + \frac{i_1}{k+1}v_{N-1}(i_0+1, i_1-1; k+1)\right.$$
$$+\frac{k-i_0-i_1}{k+1}v_{N-1}(i_0+1, i_1; k+1)\bigg] + (1-p)\left[\frac{i_0}{k}v_{N-1}(i_0-1, i_1+1; k)\right.$$
$$+\frac{i_1}{k}v_{N-1}(i_0, i_1-1; k) + \frac{k-i_0-i_1}{k}v_{N-1}(i_0, i_1; k)\bigg].$$

Having applied formula (2) and performed the subsequent transformation, we get

$$\bar{v}_{N-2}(i_0, i_1; k) = i_1 - \frac{1}{2}(k-i_0-i_1) + A_6 + p\frac{i_0+1}{k+1}\max\{0; A_1\}$$
$$+(1-p)\frac{i_0}{k}\max\{0; A_2\} + p\frac{i_1}{k+1}\max\{0; A_3\} + (1-p)\frac{i_1}{k}\max\{0; A_4\} \qquad (3)$$
$$+p\frac{k-i_0-i_1}{k+1}\max\{0; A_5\} + (1-p)\frac{k-i_0-i_1}{k}\max\{0; A_6\},$$

where
$$A_1 = \frac{p(k+1)+\left(i_0-\frac{3}{2}(i_1+1)\right)(k+2-p)}{(k+1)(k+2)}; \; A_2 = \frac{pk+\left(i_0-1-\frac{3}{2}(i_1+1)\right)(k+1-p)}{k(k+1)};$$
$$A_3 = \frac{p(k+1)+\left(i_0+1-\frac{3}{2}(i_1-1)\right)(k+2-p)}{(k+1)(k+2)}; \; A_4 = \frac{pk+\left(i_0-\frac{3}{2}(i_1-1)\right)(k+1-p)}{k(k+1)};$$
$$A_5 = \frac{p(k+1)+\left(i_0+1-\frac{3}{2}i_1\right)(k+2-p)}{(k+1)(k+2)}; \; A_6 = \frac{pk+\left(i_0-\frac{3}{2}i_1\right)(k+1-p)}{k(k+1)}.$$

Let us now consider different cases:

1) If $\frac{3}{2}i_1 - i_0 \geq \frac{p(k+1)}{k+2-p} + \frac{5}{2}$ (or $i_1 \geq \frac{2}{3}i_0 + \frac{2}{3}\frac{p(k+1)}{k+2-p} + \frac{5}{3}$), then all maxima in formula (3) are equal to 0. Then,

$$\bar{v}_{N-2}(i_0, i_1; k) = i_1 - \frac{1}{2}(k-i_0-i_1) + A_6.$$

Note that in this interval $A_6 < 0$. Hence, in this case it is optimal to stop. The payoff is then equal to

$$v_{N-2}(i_0, i_1; k) = i_1 - \frac{1}{2}(k-i_0-i_1).$$

2) If $\frac{2}{3}i_0 + \frac{2}{3}\frac{pk}{k+1-p} + 1 \leq i_1 < \frac{2}{3}i_0 + \frac{2}{3}\frac{p(k+1)}{k+2-p} + \frac{5}{3}$, then all maxima are equal to 0 except for $\max\{0; A_3\}$. In this case

$$v_{N-2}(i_0, i_1; k) = i_1 - \frac{1}{2}(k-i_0-i_1) + \max\left\{0; A_6 + p\frac{i_1}{k+1}A_3\right\}.$$

Here, $A_6 < 0$ and $A_3 > 0$. Hence, stopping is optimal if $A_6 + p\frac{i_1}{k+1}A_3 \geq 0$.

3) If $\frac{2}{3}i_0 + \frac{2}{3}\frac{p(k+1)}{k+2-p} + \frac{2}{3} \leq i_1 < \frac{2}{3}i_0 + \frac{2}{3}\frac{pk}{k+1-p} + 1$, then all maxima in formula (2) are equal to 0 except for $\max\{0; A_3\}$ and $\max\{0; A_4\}$. Therefore,

$$v_{N-2}(i_0, i_1; k) = i_1 - \frac{1}{2}(k - i_0 - i_1) +$$
$$+ \max\left\{0; A_6 + p\frac{i_1}{k+1}A_3 + (1-p)\frac{i_1}{k}A_4\right\}.$$

Here, $A_6 < 0$, while $A_3 > 0$ and $A_4 > 0$. Hence, stopping is optimal if $A_6 + p\frac{i_1}{k+1}A_3 + (1-p)\frac{i_1}{k}A_4 \geq 0$.

4) If $\frac{2}{3}i_0 + \frac{2}{3}\frac{pk}{k+1-p} \leq i_1 < \frac{2}{3}i_0 + \frac{2}{3}\frac{p(k+1)}{k+2-p} + \frac{2}{3}$, then all maxima are equal to 0 except for $\max\left\{0; A_3\right\}$, $\max\{0; A_4\}$, and $\max\{0; A_5\}$. Therefore,

$$v_{N-2}(i_0, i_1; k) = i_1 - \frac{1}{2}(k - i_0 - i_1) +$$
$$+ \max\left\{0; A_6 + p\frac{i_1}{k+1}A_3 + (1-p)\frac{i_1}{k}A_4 + p\frac{k - i_0 - i_1}{k+1}A_5\right\}.$$

5) If $i_1 \leq \frac{2}{3}i_0 + \frac{2}{3}\frac{pk}{k+1-p}$, it is optimal to continue the process and move to the next step since $A_6 \geq 0$ and the payoff if we continue is greater than the payoff if we stop the process.

We can continue in the same way to calculate the payoffs for $n = 1, ..., N-2$ steps by formula (1) using the dynamic programming method.

Before starting the game, we find the expected payoff

$$v_0(1, 0; 1) = pv_1(1, 1; 2) + (1-p)v_1(0, 1; 1).$$

Payoffs at the beginning of the game are given in Table 1 for different N and p values. As the Table shows, payoffs increase in line with N and p.

Table 1. Optimal payoffs $v_0(1, 0; 1)$.

p	0	0.1	0.3	0.5	0.7	1
$N = 1$	1	1	1	1	1	1
$N = 2$	1	1	1	1	1.02917	1.16667
$N = 3$	1	1.00467	1.041	1.11111	1.21233	1.41667
$N = 4$	1	1.01692	1.07624	1.16865	1.29837	1.55833
$N = 5$	1	1.02273	1.09462	1.20738	1.3696	1.65972
$N = 10$	1	1.03022	1.13274	1.3353	1.71902	2.30653
$N = 20$	1	1.03169	1.15306	1.64507	2.21508	3.39184
$N = 30$	1	1.03183	1.18372	1.82622	2.70821	4.50654
$N = 40$	1	1.03185	1.26061	1.98056	3.17587	5.66208
$N = 50$	1	1.03186	1.30767	2.14869	3.65483	6.84376

Figure 1 illustrates the behavior of payoffs depending on the p and N values. For small p values, payoffs grow slowly, whereas for $p \geq 1/2$ payoffs grow at a much faster rate.
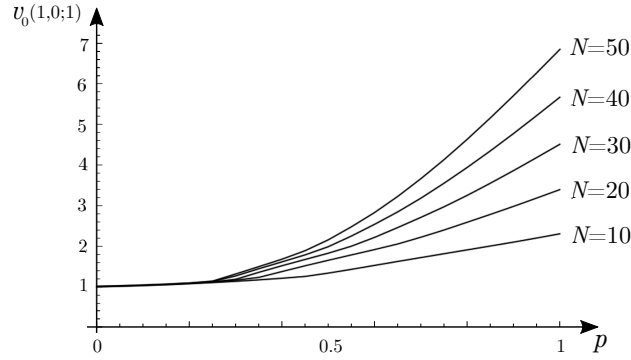
Fig. 1. Optimal payoffs $v_0(1,0;1)$ for different values N and p.

5. Approximation of the optimal strategy

Calculating payoffs by formula (1) is a fairly complicated task, because payoffs depend on the state of the system at previous steps and depend on the number of bins at each step.

In this section we propose a strategy, which has a simple form and gives a good approximation of the optimal strategy for the problem.

Tijms, 2017 proposed the so-called one-stage-look-ahead rule (OSLA) as a good approximation for the optimal stopping problem from that paper. According to this rule, the current payoff is compared against the expected reward in the case the game is continued one step further.

In our problem, the OSLA rule can be derived by comparing how the payoff will change if we stop and if we continue and then stop. If you decide at step n in state $s = (i_0, i_1; k)$ to drop one more ball and then stop rather than to stop now, then the expected change in your final reward is

$$p\left(\frac{i_0+1}{k+1}\cdot 1 - \frac{i_1}{k+1}\left(\frac{1}{2}+1\right) + \frac{k-i_0-i_1}{k+1}\cdot 0\right) + (1-p)\left(\frac{i_0}{k}\cdot 1 - \frac{i_1}{k}\left(\frac{1}{2}+1\right) + \frac{k-i_0-i_1}{k}\cdot 0\right)$$

$$= \left(i_0 - \frac{3}{2}i_1\right)\frac{k+1-p}{k(k+1)} + \frac{p}{k+1}.$$

This strategy is the same as the strategy at step $N-1$ (in Section 3).

Hence, the set of states in which it is at least as good to stop now in state s as to continue for one more step and then stop is given by

$$B = \left\{(i_0, i_1; k) : i_1 \geq \frac{2}{3}\left(i_0 + \frac{pk}{k+1-p}\right)\right\}.$$

In our problem, however, the OSLA rule (Table 2, the first column) is near-optimal only for small N values.

Numerical results show that the optimal strategy in our problem is close to the following one: to stop the process at step n, if $i_1 \geq \frac{2}{3}i_0 + p\sqrt{N-n}$ — we denote it by $\bar{C}(p)$. Then, the optimal strategy can be denoted as $C^*(p)$.

Table 2 shows the numerical results for payoffs if we use the OSLA rule and the $\bar{C}(p)$ strategy for different N and p values.

Table 2. Payoffs at the beginning of the game using OSLA rule and strategy $\bar{C}(p)$.

	OSLA $p = 1$	$\bar{C}(p)$			
		$p = 0.1$	$p = 0.5$	$p = 0.7$	$p = 1$
$N = 1$	1	1	1	1	1
$N = 2$	1.16667	1	0.979169	1.02917	1.16667
$N = 3$	1.41667	1	1.11111	1.21233	1.41667
$N = 4$	1.55833	1	1.1672	1.26531	1.49167
$N = 5$	1.65417	1	1.20668	1.36744	1.65975
$N = 10$	1.89157	1	1.32533	1.71597	2.25949
$N = 20$	2.07615	1.03133	1.53018	2.19653	3.37614
$N = 30$	2.169	1.03178	1.726	2.65138	4.49405
$N = 40$	2.23192	1.03182	1.88779	3.13837	5.66013
$N = 50$	2.28056	1.03183	2.03105	3.62872	6.84295

Figure 2 illustrates the optimal strategy $C^*(p)$ (bold line) and regions for stopping (shaded area) and continuing (white area). The aria above the dashed line shows the approximated $\bar{C}(p)$ strategy for $N = 50$, $p = 1$, $n = 40$, $k = 40$.
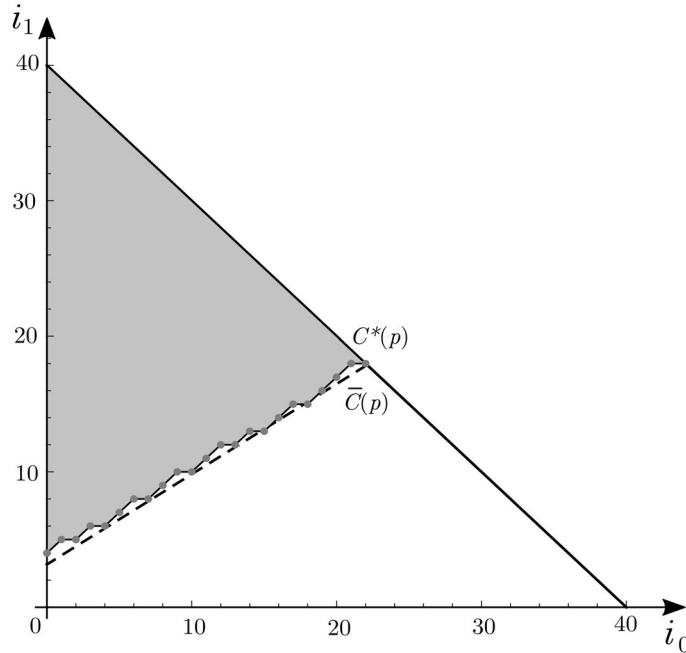

Fig. 2. Optimal strategy $C^*(p)$ and $\bar{C}(p)$ strategy for $N = 50$, $p = 1$, $n = 40$, $k = 40$.

Strategies of this kind, which involve the square root of the number of steps, can be seen in optimal stopping problems. For example, the paper Bearden, 2006 studied an extension of the secretary problem to the incomplete information and demonstrated that the strategy representing the square root of the number of steps is optimal.

Thus, players in the job allocation problem need to monitor the number of servers with only one job and with two or more jobs. If the number of servers with only one job exceeds the threshold, continuing the allocation process becomes inexpedient for optimal functioning of the system and the process should be stopped.

6. Conclusion

A new model for allocating computing resources to servers was proposed. The corresponding multistage balls-and-bins problem with optimal stopping was considered to determine the optimal strategy. For this problem, optimality equations were composed and the construction of optimal payoffs was described. Optimal payoffs and strategies were numerically modeled for different parameters of the problem. A simple-form strategy approximated to the optimal one was suggested. Finding an approximated strategy with a simple form is important to enable application of the results to job allocation problems.

The problem of finding the optimal stopping strategy for large values of N has not yet been studied. It is important both from the theoretical viewpoint and in practical applications, since a job allocation system can contain a large number of tasks and servers.

References

Bearden, J. N. (2006). *A new secretary problem with rank-based selection and cardinal payoffs*. Journal of Mathematical Psychology, **50(1)**, 58–59.

Kolchin, V. F. (1978). *Random allocations*. Winston [usw.]: Washington.

Shepp, L. A. (1969). *Explicit solutions to some problems of optimal stopping*. Ann. Math. Statist., **40**, 993–1010.

Tamaki, M. (2001). *Optimal stopping on trajectories and the ballot problem*. Journal of Applied Probability, **38**, 946–959.

Tijms, H. (2017). *One-Step Improvement Ideas and Computational Aspects*. In: Markov Decision Processes in Practice. International Series in Operations Research & Management Science (Boucherie, R. and van and N. Dijk, eds), Vol. 248, pp. 3–32. Springer, Cham.