# Estimation of the Optimal Threshold Policy in a Queue with Heterogeneous Servers Using a Heuristic Solution and Artificial Neural Networks 

Dmitry Efrosinin ${ }^{1,2, *(D)}$ and Natalia Stepanova ${ }^{3}$ (D)<br>1 Insitute for Stochastics, Johannes Kepler University Linz, 4030 Linz, Austria<br>2 Department of Information Technologies, Faculty of Mathematics and Natural Sciences, Peoples' Friendship University of Russia (RUDN University), 117198 Moscow, Russia<br>3 Laboratory N17, Trapeznikov Institute of Control Sciences of RAS, 117997 Moscow, Russia; natalia0410@rambler.ru<br>* Correspondence: dmitry.efrosinin@jku.at

Citation: Efrosinin, D.; Stepanova, N. Estimation of the Optimal Threshold Policy in a Queue with
Heterogeneous Servers Using a Heuristic Solution and Artificial Neural Networks. Mathematics 2021, 9, 1267. https://doi.org/10.3390/ math9111267

Academic Editors: Vladimir M Vishnevsky and Mark Kelbert

Received: 26 March 2021
Accepted: 25 May 2021
Published: 31 May 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https:// creativecommons.org/licenses/by/ 4.0/).


#### Abstract

This paper deals with heterogeneous queues where servers differ not only in service rates but also in operating costs. The classical optimisation problem in queueing systems with heterogeneous servers consists in the optimal allocation of customers between the servers with the aim to minimise the long-run average costs of the system per unit of time. As it is known, under some assumptions the optimal allocation policy for this system is of threshold type, i.e., the policy depends on the queue length and the state of faster servers. The optimal thresholds can be calculated using a Markov decision process by implementing the policy-iteration algorithm. This algorithm may have certain limitations on obtaining a result for the entire range of system parameter values. However, the available data sets for evaluated optimal threshold levels and values of system parameters can be used to provide estimations for optimal thresholds through artificial neural networks. The obtained results are accompanied by a simple heuristic solution. Numerical examples illustrate the quality of estimations.


Keywords: heterogeneous servers; policy-iteration algorithm; heuristic solution; artificial neural networks

## 1. Introduction

Many queueing systems are analysed for their dynamic and optimal control related to system access, resource allocation, changing service area characteristics and so on. Sets of computerised tools and procedures provide large data sets which can be useful to expand potential of classical optimisation methods. The paper deals with a known model of a multiserver queueing system with controllable allocation of customers between heterogeneous servers which are differentiated by their service and cost attributes. For the queueing system with two heterogeneous servers it has been shown in [1] by using a dynamic programming approach that to minimise the mean sojourn time of customers in the system, the faster server must be always used and the customer has to be assigned to the slower server if and only if the number of customers in the queue exceeds the certain threshold level. Furthermore, this result was obtained independently in more simple form in [2,3]. In [4], the author has analysed a multi-server version of such a system and confirmed a threshold nature of the optimal policy as well.

The problem of an optimal allocation of customers between heterogeneous servers in queueing systems with additional costs with the aim to minimise the long-run average cost per unit of time is notoriously more difficult. Some progress has been made after the appearance of a review paper [5]. In [6,7], the authors studied a model with set-up costs using a hysteretic control rule, thereby stressing the algorithmic aspects of the optimal control structure. The same system has been discussed in [8], where a direct method that
provides a closed-form expression for the stationary occupancy distribution was proposed. In [9,10], the authors have used theoretical study and exhaustive numerical analysis to show that for some specified servers, ordering the optimal allocation policy which minimises the long-run average cost belongs to a set of structural policies. In other words, for the servers' enumeration (1), the allocation control policy denoted by $f$ can be defined through a sequence of threshold levels $1=q_{1} \leq q_{2} \leq \cdots \leq q_{K}<\infty$. With respect to the defined policy, the server operating at highest rate should remain busy by non-empty queueing system. The $k$ th server $(k \geq 2)$ is used only if the first $k-1$ servers are busy and the queue length reaches a threshold level $q_{k}>0$. In the general case, the optimal threshold levels can depend on states of slower server and formally the optimal policy $f$ is not of a pure threshold type. However, since the $k$ th threshold value may vary by at most one when the state of slower server changes and it has a weak effect on the average cost, such influence can be neglected. Hence the optimal allocation policy for multi-server heterogeneous queueing system can be treated as a threshold one.

Searching for the optimal values of $q_{2}, \ldots, q_{K}$ by direct minimising the average cost function can be expensive, especially when $K$ is large. To calculate the optimal threshold levels we can use a policy-iteration algorithm [11-13] which constructs a sequence of improved policies that converges to optimal one. This algorithm is a fairly versatile tool for solving various optimisation problems. Unfortunately, as is usually the case in practice, this algorithm is not without some limitations, such as the difficulties associated with convergence when the traffic is close to loaded, limitation on the process dimension and, consequently, on the number of states. Thus, we would like to compensate for some of the weaknesses of this algorithm with other methods for calculating the optimal control policy. The contribution of this paper can be briefly described in two conceptual parts. In the first part, we propose a heuristic solution (HS) to obtain functional relationships for optimal thresholds based on a simple discrete approximation of the system's behaviour. The second part is devoted to the alternative machine learning technique such as artificial neural networks (NN) [14-16] which is used again for the estimation of the optimal threshold levels. The policy-iteration algorithm is used in the paper to generate the data sets needed both to verify the quality of the proposed optimal threshold estimation methods and to train the neural networks. We strongly believe that the trained neural network can be successfully used to calculate the optimal thresholds for those system parameters for which alternative numerical methods are difficult or impossible to use, for example, in heavy traffic case, or, in general, to reconstruct the areas of optimality without usage of timeexpensive algorithms and procedures. There are some number of papers on prediction of the stochastic behaviour of queueing systems and networks using machine learning algorithms, see e.g., $[17,18]$ and references therein. However, we unsuccessfully tried to find published works where heuristics and machine learning methods would be used to solve a similar optimisation problem for heterogeneous queueing systems and therefore we consider this paper relevant.

This paper is organised as follows. In Section 2, we briefly discuss a mathematical model. Section 3 introduces some heuristic choices for threshold levels that turn out to be nearly optimal. Section 4 presents results when the trained neural network was ran on verification data of the policy-iteration algorithm.

## 2. Mathematical Model

We summarise briefly the model under study. The queueing system is of the type $M / M / K$ with infinite-capacity buffer and $K$ heterogeneous servers. This system is shown schematically in Figure 1. The Poisson arrival stream has a rate $\lambda$ and the exponential distribution of the service time at server $j$ has a rate $\mu_{j}$. We assume that the service in the system is without preemption, when customer in service cannot change the server. The random variables of the inter-arrival times and the service times of the servers are assumed to be independent. An additional cost structure is introduced, consisting of the operating
$\operatorname{cost} c_{j}>0$ per unit of time of service on server $j$ and the holding $\operatorname{cost} c_{0}>0$ of waiting in the queue. Assume that the servers are enumerated in a way

$$
\begin{equation*}
\mu_{1} \geq \cdots \geq \mu_{K}, c_{1} \mu_{1}^{-1} \leq \cdots \leq c_{K} \mu_{K}^{-1} \tag{1}
\end{equation*}
$$

where $c_{j} \mu_{j}^{-1}$ stands for the mean operating cost per customer for the $j$ th server.


Figure 1. Controllable multi-server queueing system with heterogeneous servers and operating costs.
The controller has full information about the system's state and, based on this information, can make control actions on the system at the decision epochs when certain state transitions occur, following the prescription of the policy $f$. In our case, the controller selects the control action at the time when a new customer enters the system and at the service completion times, if the queue is not empty. When a new customer arrives, it joins the queue and at the same time, the controller sends another customer from the head of the queue to one of the idle servers or leaves it in the queue. At the service completion, the customer leaves the corresponding server, and at the same time the controller takes the next customer from the head of the queue, if it is not empty, and dispatches it to one of idle servers or can leave it in the queue as well. The service completion in the system without waiting customers does not require the controller to perform any control action.

The fact that the optimal policy for the problem of minimising the long-run average cost per unit of time belongs to a set of threshold-based policies for the multi-server heterogeneous queueing systems with costs were proved first in [10] and further conformed for systems with heterogeneous groups of servers in [19]. The corresponding optimal thresholds can in the general case depend on the states of slower servers. However, according to obtained numerical results in [9], we can neglect the weak influence of the slower servers' states on the optimal allocation policy for the faster servers. This phenomena was discussed additionally in Example 2. Therefore, we may assume that the optimal policy belongs to the class of a pure threshold policy when the use of a certain server depends solely on the number of waiting customers in the queue. Specifically, for the system under study, such a policy is defined by the following sequence of threshold levels:

$$
\begin{equation*}
1=q_{1} \leq q_{2} \leq \cdots \leq q_{K}<\infty . \tag{2}
\end{equation*}
$$

The policy prescribes the use of the $k$ fastest servers whenever the number of customers waiting in the queue satisfies the condition $q_{k} \leq q \leq q_{k+1}-1$.

To calculate optimal thresholds we need to formulate the introduced optimisation problem in terms of a Markov decision process. This process is based on a $K+1$-dimensional continuous-time Markov chain

$$
\begin{equation*}
\{X(t)\}_{t \geq 0}=\left\{Q(t), D_{1}(t), \ldots, D_{K}(t)\right\}_{t \geq 0} \tag{3}
\end{equation*}
$$

with an infinitesimal matrix $\Lambda^{f}$ which depends on the policy $f$. Here the component $Q(t) \in \mathbb{N}_{0}$ stands for the number of waiting customers at time $t$ and

$$
D_{j}(t)= \begin{cases}0 & \text { if } j \text { th server is idle } \\ 1 & \text { if } j \text { th server is busy }\end{cases}
$$

The state space of the process $\{X(t)\}_{t \geq 0}$ operating under some policy $f$ is $E^{f}=\{x=$ $\left.\left(q(x), d_{1}(x), \ldots, d_{K}(x)\right)\right\} \subseteq \mathbb{N}_{0} \times\{0,1\}^{K}$, where the notations $q(x)$ and $d_{j}(x)$ are used respectively for the queue length and for the state of $j$ th server in state $x \in E^{f}$.

The possible server states are partitioned as follows:

$$
J_{0}(x)=\left\{j: d_{j}(x)=0\right\}, J_{1}(x)=\left\{j: d_{j}(x)=1\right\}
$$

The sets $J_{0}(x)$ and $J_{1}(x)$ denote the sets of idle and busy servers in state $x \in E^{f}$, respectively. The set of control actions $a$ is $A=\{0,1, \ldots, K\}$. If $a=0$, the controller allocates a customer to the queue. Otherwise, if $a \neq 0$, the controller instructs a customer to occupy the server with a number $a$. In addition, we can define the subsets $A(x)=J_{0}(x) \cup\{0\} \subseteq A$ of admissible actions in state $x$ The policy $f$ specifies the choice of a control action at any decision epoch and the infinitesimal matrix $\Lambda^{f}=\left[\lambda_{x y}(a)\right]$ of the Markov-chain (3) has then the following elements,

$$
\lambda_{x y}(a)= \begin{cases}\lambda & y=x+\mathbf{e}_{a}, j \in A(x) \\ \mu_{j} & y=x-\mathbf{e}_{j}, j \in J_{1}(x), q(x)=0 \\ \mu_{j} & y=x-\mathbf{e}_{j}-\mathbf{e}_{0}+\mathbf{e}_{a}, a \in A\left(x-\mathbf{e}_{j}-\mathbf{e}_{0}\right), q(x)>0\end{cases}
$$

where $\mathbf{e}_{j}$ is defined as $K+1$-dimensional unit vector with each element equal to zero except the $j$ th position $(j=0,1, \ldots, K)$.

We will search for the optimal control policy among the set of stationary Markov policies $f$ that guarantee ergodicity of the Markov chain $\{X(t)\}_{t \geq 0}$. The corresponding stability condition is obviously defined as $\lambda<\sum_{j=1}^{K} \mu_{j}$. It follows from the fact, that if number of customers exceeds a threshold $q_{K}$, then the queueing systems behaves like a $M / M / 1$ queue with an arrival rate $\lambda$ and total service rate $\mu_{1}+\cdots+\mu_{K}$. As it is known, see e.g., [13], the ergodic Markov chain with costs implies the equality of the long-run average cost per unit of time for the policy $f$ and the corresponding assemble average, that can be written in the form

$$
\begin{equation*}
g^{f}=\limsup _{t \rightarrow \infty} \frac{1}{t} V^{f}(x, t)=\sum_{y \in E^{f}} c(y) \pi_{y}^{f} \tag{4}
\end{equation*}
$$

where $c(y)=c_{0} q(y)+\sum_{j=1}^{K} c_{j} d_{j}(y)$ is an immediate cost in state $y \in E^{f}$. The cost function $V^{f}(x, t)$ is given by

$$
V^{f}(x, t)=\mathbb{E}^{f}\left[\int_{0}^{t}\left(c_{0} Q(t)+\sum_{j=1}^{K} c_{j} D_{j}(t)\right) d t \mid X(0)=x\right]
$$

This function describes the total average cost up to time $t$ given the initial state is $x$ and $\pi_{y}^{f}=\mathbb{P}^{f}[X(t)=y]$ is a stationary state distribution for the policy $f$. The policy $f^{*}$ is said to be optimal when for $g^{f}$ defined in (4) we evaluate

$$
\begin{equation*}
g^{*}=\inf _{f} g^{f}=\min _{q_{2}, \ldots, q_{K}} g\left(q_{2}, \ldots, q_{K}\right) \tag{5}
\end{equation*}
$$

To evaluate optimal threshold levels and optimised value for the mean average cost per unit of time the policy-iteration Algorithm 1 is used. This algorithm constructs a sequence of improved policies until the average cost optimal is reached. It consists of three main steps: value evaluation, policy improvement and threshold evaluation. The Value evaluation is based on solving, for a given policy $f$, a system of linear equations

$$
\begin{equation*}
v^{f}(x)=\frac{1}{\lambda_{x}(a)}\left(c(x)+\sum_{y \neq x} \lambda_{x y}(a) v^{f}(y)-g^{f}\right) . \tag{6}
\end{equation*}
$$

```
Algorithm 1 Policy-iteration algorithm
    procedure \(\operatorname{PIA}\left(K, W, \lambda, \mu_{j}, c_{j}, j=1,2, \ldots, K, c_{0}\right)\)
        \(f^{(0)}(x)=\operatorname{argmin}_{j \in J_{0}(x)}\left\{\frac{c_{j}}{\mu_{j}}\right\} \quad \triangleright\) Initial policy
        \(n \leftarrow 0\)
        \(g^{f^{(n)}}=\lambda v^{f^{(n)}}\left(\mathbf{e}_{1}\right) \quad \triangleright\) Value evaluation
        for \(x=(0,1,0, \ldots, 0)\) to \((N, 1,1, \ldots, 1)\) do
            \(v^{f^{(n)}}(x)=\frac{1}{\lambda+\sum_{j \in J_{1}(x)} \mu_{j}}\left[c(x)-g^{f^{(n)}}+\lambda v^{f^{(n)}}\left(x+\mathbf{e}_{f^{(n)}(x)}\right)\right.\)
            \(+\sum_{j \in J_{1}(x)} \mu_{j} v^{f^{(n)}}\left(x-\mathbf{e}_{j}\right) 1_{\{q(x)=0\}}\)
    \(\left.+\sum_{j \in J_{1}(x)} \mu_{j} v^{f^{(n)}}\left(x-\mathbf{e}_{j}-\mathbf{e}_{0}+\mathbf{e}_{f^{(n)}\left(x-\mathbf{e}_{j}-\mathbf{e}_{0}\right)}\right) 1_{\{q(x)>0\}}\right]\)
        end for
                                    \(\triangleright\) Policy improvement
```

$$
f^{(n+1)}(x)=\operatorname{argmin}_{a \in A(x)} v^{f^{(n)}}\left(x+\mathbf{e}_{a}\right)
$$

        if \(f^{(n+1)}(x)=f^{(n)}(x), x \in E^{f}\) then return \(f^{(n+1)}(x), v^{f^{(n)}}(x), g^{f^{(n)}}\)
        else \(n \leftarrow n+1\), go to step 4
        end if
    $$
q_{k}: f^{(n+1)}\left(q, 1, \ldots, 1,0, d_{k+1}, \ldots, d_{K}\right)=\left\{\begin{array}{ll}
0 & q \leq q_{k}-2 \\
k & q>q_{k}-2
\end{array}, k=2, \ldots, K\right.
$$

end procedure

For the dynamic-programming value function $v^{f}: E^{f} \rightarrow \mathbb{R}$, which indicates a transition effect of an initial state $x$ to the total average cost and satisfies the following asymptotic relation,

$$
V^{f}(x, t)=g^{f} t+v^{f}(x)+o(1), t \rightarrow \infty, x \in E^{f}
$$

In order to make the system (6) solvable, one of the values $v(x)$ must be set to zero, e.g., for $x_{0}=(0, \ldots, 0)$ we set $v\left(x_{0}\right)=0$. Since in our case $c\left(x_{0}\right)=0$, the first equation of the system (6) is of the form $g^{f}=\sum_{y \neq x_{0}} \lambda_{x_{0} y}(a) v^{f}(y)$. In the policy improvement step a new policy $f^{\prime}$ is calculated by minimising the value function $v\left(x+\mathbf{e}_{a}\right)$ for any state $x \in E^{f}$ and any admissible control action $a \in A(x)$. The algorithm converges if the policies $f$ and $f^{\prime}$ on neighbouring iterations are equal. In the threshold evaluation we calculate the optimal thresholds $q_{k}, k=2, \ldots, K$, based on optimal policy $f$. As an initial policy we select the policy which prescribes in any state the usage of a server $j$ with the minimal value of the mean operating cost $\frac{c_{j}}{\mu_{j}}$ per customer. More detailed information on deriving the dynamic programming equations for the heterogeneous queueing system and calculating the corresponding optimal allocation control policy can be found in [9]. For existence of
an optimal stationary policy and convergence of the policy-iteration algorithm we refer to [12,20-22].

To realise the policy-iteration algorithm we convert the $K+1$-dimensional state space $E^{f}$ of the Markov decision process to a one-dimensional equivalent state space. Let $\Delta$ : $E^{f} \rightarrow \mathbb{N}_{0}$ be a one-to-one mapping of the vector state $x=\left(q(x), d_{1}(x), \ldots, d_{K}(x)\right) \in E^{f}$ to a value from $\mathbb{N}_{0}$ which is of the form

$$
\begin{equation*}
\Delta(x)=q(x) 2^{K}+\sum_{i=1}^{K} d_{i}(x) 2^{i-1} \tag{7}
\end{equation*}
$$

A new state after transition involving the addition or removal of customer in some state $x \in E^{f}$, in a one-dimensional state space is calculated by

$$
\begin{aligned}
& \Delta\left(x \pm \mathbf{e}_{0}\right)=(q(x) \pm 1) 2^{K}+\sum_{i=1}^{K} d_{i}(x) 2^{i-1}=\Delta(x) \pm 2^{K} \\
& \Delta\left(x \pm \mathbf{e}_{j}\right)=q(x) 2^{K}+\sum_{i=1}^{K} d_{i}(x) 2^{i-1} \pm 2^{j-1}=\Delta(x) \pm 2^{j-1}
\end{aligned}
$$

Further in the algorithm, an infinite buffer system must be approximated by an equivalent system where the number of waiting places is finite but at the same time is sufficiently large. As a truncation criterion, we use the loss probability which should not exceed some small value $\varepsilon>0$.

Remark 1. If the buffer size is $W$, the number of states is

$$
\left|E^{f}\right|=2^{K}(W+1)
$$

In case the number of waiting customers is getting larger as the level $q_{K}$, all servers must be occupied and the system dynamics is the same as in a classical queue $M / M / 1$ with arrival rate $\lambda$ and service rate $\sum_{j=1}^{K} \mu_{j}$. The stationary state probabilities for the states $x$ where the component $q(x) \geq q_{K}$ satisfy the following difference equation

$$
\lambda \pi_{(q-1,1, \ldots, 1)}-\left(\lambda+\sum_{j=1}^{K} \mu_{j}\right) \pi_{(q, 1, \ldots, 1)}+\sum_{j=1}^{K} \mu_{j} \pi_{(q+1,1, \ldots, 1)}=0,
$$

 theoretical substantiation see, e.g., [23]. Note that the value of $q_{K}$ included in this formula can be estimated by a heuristic solution (9). Then the truncation parameter $W$ of the buffer size can be evaluated from the following constraint for the loss probability

$$
\sum_{q=W}^{\infty} \pi_{(q, 1, \ldots, 1)}=\pi_{q_{K}} \sum_{q=W}^{\infty} \rho^{q-q_{K}} \leq \sum_{q=W}^{\infty} \rho^{q-q_{K}}=\frac{\rho^{W-q_{K}}}{1-\rho}<\varepsilon
$$

where $\rho=\frac{\lambda}{\sum_{j=1}^{K} \mu_{j}}$. After simple algebra, it implies

$$
W>\frac{\log \varepsilon(1-\rho)}{\log (\rho)}+q_{K}
$$

Example 1. Consider the system $M / M / 5$ with $K=5$ and $\lambda=15$. All other parameters take the following values

| $j$ | 0 | 1 | 2 | 3 | 4 | 5 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| $c_{j}$ | 1 | 5 | 4 | 3 | 2 | 1 |
| $\mu_{j}$ | - | 20 | 8 | 4 | 3 | 1 |
| $c_{j} \mu_{j}^{-1}$ | - | 0.25 | 0.50 | 0.75 | 0.67 | 1.00 |

The truncation parameter $W$ of the buffer size is chosen at value 80 which for $\varepsilon=0.0001$ guarantees that $W>\frac{\log 0.0001(1-14 / 36)}{\log (14 / 36)}+q_{5}=22.2734$. Here $q_{5}=12$ was calculated by (9). In a control table, we summarise the functions $f(x)$ which specify the control actions at time of arrivals to a certain state $x$ :

| $\begin{gathered} \text { System State } x \\ d=\left(d_{1}, d_{2}, d_{3}, d_{4}, d_{5}\right) \end{gathered}$ | Queue Length $q(x)$ |  |  |  |  |  |  |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | $\ldots$ |
| ( $0,{ }^{*}, *, *, *$ ) | $\underline{1}$ | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| $(1,0, *, *, *)$ | 0 | 0 | $\underline{2}$ | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| (1,1,0,*,*) | 0 | 0 | 0 | $\underline{3}$ | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| (1,1,1,0,*) | 0 | 0 | 0 | 0 | $\underline{4}$ | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 |
| (1,1,1,1,0) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | $\underline{5}$ | 5 | 5 |
| (1,1,1,1,1) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Threshold levels $q_{k}, k=1, \ldots, K=5$, can be evaluated by comparing the optimal actions $f(q, \underbrace{1, \ldots, 1}_{k-1}, \underbrace{0, \ldots, 0}_{K-k+1})<f(q+1, \underbrace{1, \ldots, 1}_{k-1}, \underbrace{0, \ldots, 0}_{K-k+1})$ for $q=0, \ldots, W-1$. In this example the optimal policy $f^{*}$ is defined here through a sequence of threshold levels $\left(q_{2}, q_{3}, q_{4}, q_{5}\right)=(3,4,5,12)$ and $g^{*}=4.92897$. The bold and underline format in a control table is used to label the change of the control action in a certain system state.

In the next example we give some arguments that allow us to work further only with the threshold-based control policies.

Example 2. Consider the system $M / M / 3$ with $K=3$ servers. The aim of this example consists in the following: With respect to the system states $x=(q, 1,0,0)$ and $y=(q, 1,0,1)$ the assignment to the second server can in general depend not only on the number of customers in the queue but also on the state of the third server. In this example it is optimal to make an assignment in state $x$ but not in state $y$. We solve optimisation problem for the following parameters:

- $\lambda=0.238, \mu_{1}=0.621, \mu_{2}=0.071$ and $\mu_{3}=0.070$,
- $\lambda=0.477, \mu_{1}=0.356, \mu_{2}=0.096$ and $\mu_{3}=0.070$.

The of optimal solution for the first and second group of system parameters are represented in Tables 1 and 2, respectively.

Table 1. Control table.

| System State $\boldsymbol{x}$ | Queue Length $q(x)$ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| $\left(d_{1}, d_{2}, d_{3}\right)$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | $\ldots$ |
| $(0,0,0)$ | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| $(1,0,0)$ | 0 | 0 | 0 | 0 | 0 | $\underline{2}$ | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| $(0,1,0)$ | $\underline{1}$ | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| $(1,1,0)$ | 0 | 0 | 0 | 0 | 0 | $\underline{3}$ | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| $(0,0,1)$ | $\underline{1}$ | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| $(1,0,1)$ | 0 | 0 | 0 | 0 | $\underline{2}$ | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| $(0,1,1)$ | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| $(1,1,1)$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

We notice that for most parameter values the optimal decision can be made independently of the states of the slower servers. However, it is interesting to consider the reasons for such possible dependence. It is evident that in our optimisation problem, the optimal policy assigns a customer to
the fastest free server in states for which this would not be optimal if there were no arrivals. This is because the system should be ready for possible arrivals, which, if they occur, will wish to see a less congested system.

Table 2. Control table.

| System State $x$ | Queue Length $q(x)$ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| $\left(d_{1}, d_{2}, d_{3}\right)$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |  |
| $(0,0,0)$ | $\underline{1}$ | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| $(1,0,0)$ | 0 | $\underline{2}$ | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| $(0,1,0)$ | $\underline{1}$ | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| $(1,1,0)$ | 0 | $\underline{3}$ | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| $(0,0,1)$ | $\underline{1}$ | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| $(1,0,1)$ | $\underline{2}$ | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| $(0,1,1)$ | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| $(1,1,1)$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Consider now the system with three servers in the states $x+\mathbf{e}_{0}+\mathbf{e}_{1}$ and $x+\mathbf{e}_{1}+\mathbf{e}_{2}$, where $x=(0,0,0,0)$. Let us consider the case of potential service completion at the second server, taking into account a large number $q$ of accompanied arrivals. Because of large $q$, it is optimal to occupy all accessible idle servers. The states mentioned above become $x+(q-1) \mathbf{e}_{0}+\mathbf{e}_{1}+\mathbf{e}_{2}+\mathbf{e}_{3}$ and $x+(q-2) \mathbf{e}_{0}+\mathbf{e}_{1}+\mathbf{e}_{2}+\mathbf{e}_{3}$. Thus, the difference $v\left(x+(q-1) \mathbf{e}_{0}+\mathbf{e}_{1}+\mathbf{e}_{2}+\mathbf{e}_{3}\right)-v(x+(q-$ 2) $\mathbf{e}_{0}+\mathbf{e}_{1}+\mathbf{e}_{2}+\mathbf{e}_{3}$ ) of value functions measures the advantage that will be obtained in the case of the assignment to the second processor $x+\mathbf{e}_{0}+\mathbf{e}_{1} \rightarrow x+\mathbf{e}_{1}+\mathbf{e}_{2}$. The events of service completion on the second server provide the incentive to make an assignment to the second server. However, if the two initial states are $x+\mathbf{e}_{0}+\mathbf{e}_{1}+\mathbf{e}_{3}$ and $x+\mathbf{e}_{1}+\mathbf{e}_{2}+\mathbf{e}_{3}$, the measure of advantage if service completion takes place is $v\left(x+q \mathbf{e}_{0}+\mathbf{e}_{1}+\mathbf{e}_{2}+\mathbf{e}_{3}\right)-v\left((q-1) \mathbf{e}_{0}+\mathbf{e}_{1}+\mathbf{e}_{2}+\mathbf{e}_{3}\right)$. Since we expect that the value function $v\left(q \mathbf{e}_{0}+\mathbf{e}_{1}+\mathbf{e}_{2}+\mathbf{e}_{3}\right)$ is convex in $q$, it is plausible that the incentive to make an assignment to the second server is greater in state $x+\mathbf{e}_{0}+\mathbf{e}_{1}+\mathbf{e}_{3}$ than in $x+\mathbf{e}_{0}+\mathbf{e}_{1}$. Numerical examples proposed in Table 3 confirm our expectations.

Table 3. Value function for system states.

| System State $x$ | Value Function $\boldsymbol{v}(\boldsymbol{x})$ |  |
| :---: | :---: | :---: |
| $\left(q, d_{1}, d_{2}, d_{3}\right)$ | example 1 | example 2 |
| $(0,0,0,0)$ | 0 | 0 |
| $(0,1,0,0)$ | 2.6034 | 19.4480 |
| $(0,0,1,0)$ | 14.0865 | 28.3810 |
| $(0,0,0,1)$ | 14.2872 | 33.7009 |
| $(1,1,0,0)$ | 7.7979 | 51.3142 |
| $(0,1,1,0)$ | 16.6905 | 51.4444 |
| $(0,1,0,1)$ | 16.8910 | 55.9981 |
| $(0,0,1,1)$ | 28.3747 | 65.9866 |
| $(2,1,0,0)$ | 15.5520 | 96.1454 |
| $(1,1,1,0)$ | 21.8874 | 90.3521 |
| $(1,1,0,1)$ | 22.0873 | 93.2714 |
| $(0,1,1,1)$ | 30.9798 | 93.2581 |
| $(3,1,0,0)$ | 25.7823 | 154.6580 |
| $(2,1,1,0)$ | 29.6487 | 142.7630 |
| $(2,1,0,1)$ | 29.8469 | 145.4230 |
| $(1,1,1,1)$ | 36.1809 | 140.4050 |
| $\ldots$ | $\ldots$ | - |
| $(6,1,0,0)$ | 68.3382 | - |
| $(5,1,1,0)$ | 66.8622 | - |
| $(5,1,0,1)$ | 66.9946 | - |
| $(4,1,1,1)$ | 66.9830 | - |
| $(7,1,0,0)$ | 85.9322 | - |
| $(6,1,1,0)$ | 82.9672 | - |
| $(6,1,0,1)$ | 83.0730 | - |
| $(5,1,1,1)$ | 81.9234 |  |
|  |  | - |

The further numerical examples show that the threshold levels have a very weak dependence of slower servers' states. According to our observations, the optimal threshold may vary by at most 1 when the state of a slower server changes.

The data needed either to verify the heuristic solution or for training and verification of the neural network was generated by a policy-iteration algorithm in form of the list

$$
\begin{align*}
S= & \left\{\left(\lambda, \mu_{1}, \ldots, \mu_{K}, c_{0}, c_{1}, \ldots, c_{K}\right) \rightarrow\left(q_{2}, \ldots, q_{K}\right):\right.  \tag{8}\\
& \lambda \in[1,45], \mu_{1}, \ldots, \mu_{K} \in[1,40], c_{0} \in[1,3], c_{1}, \ldots, c_{K} \in[1,5], \\
& \left.\lambda<\sum_{j=1}^{K} \mu_{j}, \mu_{1} \geq \cdots \geq \mu_{K}, c_{1} \mu_{1}^{-1} \leq \cdots \leq c_{K} \mu_{K}^{-1}\right\} .
\end{align*}
$$

Example 3. Some elements of the list $S$ for the $M / M / 5$ queueing system are

$$
\begin{aligned}
& (1,20,8,4,2,1,1,1,1,1,1,1) \rightarrow(2,5,13,30),(10,20,8,4,2,1,1,1,1,1,1,1) \rightarrow(1,4,9,21) \\
& (1,20,8,4,2,1,1,5,4,3,2,1) \rightarrow(5,12,20,20),(10,20,8,4,2,1,1,5,4,3,2,1) \rightarrow(3,8,13,13)
\end{aligned}
$$

## 3. Heuristic Solution

In this section, we want to obtain a heuristic solution (HS) to calculate the optimal thresholds $q_{k}, k=2, \ldots, K$ for the arbitrary $K$ in explicit form. For this purpose, we will use a simple deterministic approximation for the dynamic behaviour of the number of customers in the queue as illustrated in Figure 2.


Figure 2. Queue length approximation.
Let $q_{k}$ is an optimal threshold used to dispatch the customer to server $k$ in state $(q_{k}-1, \underbrace{1, \ldots, 1}_{k-1}, \underbrace{0, \ldots, 0}_{K-k+1})$, where the first $k-1$ servers are busy. Now we compare the queues of the system given initial state is $x_{0}=(q_{k}, \underbrace{1, \ldots, 1}_{k-1}, 0, \underbrace{0, \ldots, 0}_{K-k})$, where the $k$ th server is not used for a new customer, and $y_{0}=(q_{k}-1, \underbrace{1, \ldots, 1}_{k-1}, 1, \underbrace{0, \ldots, 0}_{K-k})$, where the $k$ th server is occupied by a waiting customer. It is assumed that the stability condition holds. The initial queue lengths are labelled in Figure 2 by $A=q_{k}$ and $B=q_{k}-1$. The proposed deterministic approximation is based on an assumption that the queue length of the system with the first $k-1$ busy servers decreases with the rate $\sum_{j=1}^{k-1} \mu_{j}-\lambda$. When this rate is keeping until the queue is empty, it occurs at time points $D=\frac{q_{k}}{\sum_{j=1}^{k-1} \mu_{j}-\lambda}$ and $C=\frac{q_{k}-1}{\sum_{j=1}^{k-1} \mu_{j}-\lambda}$ respectively for the given initial queue length $A$ and $B$. The total (accumulated) holding
times of all customers in the queue with lengths $q_{k}$ and $q_{k}-1$ are equal respectively to the number of square blocks of dimension $1 \times \frac{1}{\sum_{j=1}^{k-1} \mu_{j}-\lambda}$ within the areas $A O D$ and $B O C$ multiplied by the mean service time of the approximated model:

$$
\begin{aligned}
& F_{A O D}=\left(q_{k}+\left(q_{k}-1\right)+\left(q_{k}-2\right)+\cdots+1\right) \frac{1}{\sum_{j=1}^{k-1} \mu_{j}-\lambda}=\frac{q_{k}\left(q_{k}+1\right)}{2} \cdot \frac{1}{\sum_{j=1}^{k-1} \mu_{j}-\lambda} \text { and } \\
& F_{B O C}=\left(\left(q_{k}-1\right)+\left(q_{k}-2\right)+\cdots+1\right) \frac{1}{\sum_{j=1}^{k-1} \mu_{j}-\lambda}=\frac{q_{k}\left(q_{k}-1\right)}{2} \cdot \frac{1}{\sum_{j=1}^{k-1} \mu_{j}-\lambda} .
\end{aligned}
$$

The mean operating cost of the first $k-1$ servers during the time period until the queue becomes empty given the initial state is $x_{0}$ can be calculated by

$$
q_{k}\left(\frac{c_{1}}{\mu_{1}} \frac{\mu_{1}}{\sum_{j=1}^{k-1} \mu_{j}}+\cdots+\frac{c_{k-1}}{\mu_{k-1}} \frac{\mu_{k-1}}{\sum_{j=1}^{k-1} \mu_{j}}\right)=q_{k} \frac{\sum_{j=1}^{k-1} c_{j}}{\sum_{j=1}^{k-1} \mu_{j}} .
$$

The expression $\frac{\mu_{i}}{\sum_{j=1}^{k-1} \mu_{j}}$ means the probability of the service completion at the $i$ th server, and the mean operating cost given the initial state is $y_{0}$, which can be defined as $\left(q_{k}-1\right) \frac{\sum_{j=1}^{k-1} c_{j}}{\sum_{j=1}^{k-1} \mu_{j}}$.

Now using the deterministic approximation we can formulate the following proposition.
Proposition 1. The optimal thresholds $q_{k}, k=2, \ldots, K$, are defined by

$$
\begin{equation*}
q_{k} \approx \hat{q}_{k}=\min \left\{1,\left\lfloor\frac{\sum_{j=1}^{k-1} \mu_{j}-\lambda}{c_{0}}\left[\frac{c_{k}}{\mu_{k}}-\frac{\sum_{j=1}^{k-1} c_{j}}{\sum_{j=1}^{k-1} \mu_{j}}\right]\right]\right\} \tag{9}
\end{equation*}
$$

Proof. Let $V(x)$ be the overall average system cost until the system becomes empty given the initial state is $x \in E^{f}$. This value can be represented as a sum of the total holding cost of customers waiting in the queue and mean operating cost of all servers which remain busy in state $x$. Assume that the controller performs a decision to allocate the customer to the $k$ th server in state $(q_{k}-1, \underbrace{1, \ldots, 1}_{k-1}, \underbrace{0, \ldots, 0}_{K-k+1})$. As a result, it leads to a reduction of the overall system costs according to the proposed deterministic approximation, i.e.,

$$
\begin{equation*}
V\left(x_{0}\right)-V\left(y_{0}\right)>0 \tag{10}
\end{equation*}
$$

where

$$
\begin{gather*}
V\left(x_{0}\right)=c_{0} F_{A O D}+q_{k} \frac{\sum_{j=1}^{k-1} c_{j}}{\sum_{j=1}^{k-1} \mu_{j}}+V(0, \underbrace{1, \ldots, 1}_{k-1}, \underbrace{0, \ldots, 0}_{K-k+1}),  \tag{11}\\
V\left(y_{0}\right)=\frac{c_{k}}{\mu_{k}}+V(q_{k}-1, \underbrace{1, \ldots, 1}_{k-1}, 0, \underbrace{0, \ldots, 0}_{K-k}) \\
=\frac{c_{k}}{\mu_{k}}+c_{0} F_{B O C}+\left(q_{k}-1\right) \frac{\sum_{j=1}^{k-1} c_{j}}{\sum_{j=1}^{k-1} \mu_{j}}+V(0, \underbrace{1, \ldots, 1}_{k-1}, \underbrace{0, \ldots, 0}_{K-k+1}) .
\end{gather*}
$$

After substitution of (11) into (10) we get

$$
\begin{aligned}
& c_{0}\left(F_{A O D}-F_{B O C}\right)+\frac{\sum_{j=1}^{k-1} c_{j}}{\sum_{j=1}^{k-1} \mu_{j}}-\frac{c_{k}}{\mu_{k}} \\
& =c_{0} \frac{q_{k}}{\sum_{j=1}^{k-1} \mu_{j}-\lambda}+\frac{\sum_{j=1}^{k-1} c_{j}}{\sum_{j=1}^{k-1} \mu_{j}}-\frac{c_{k}}{\mu_{k}}>0
\end{aligned}
$$

Now, expressing $q_{k}$ after some simple manipulations we obtain the heuristic solution for the optimal value of $q_{k}$ in form (9).

Example 4. Consider a queueing system from the previous example for $K=5$. We select randomly from the data set $S$ (8) a list of system parameters $\vec{\alpha}=\left(\lambda, \mu_{1}, \ldots, \mu_{K}, c_{0}, c_{1}, \ldots, c_{K}\right)$ and calculate by means of the HS (9) threshold levels $q_{k}, k=1, \ldots, K$. Figure 3 illustrates the efficiency of the proposed heuristic solution respectively for threshold levels $\left(q_{2}, q_{3}, q_{4}, q_{5}\right)$ by confusion matrices. The matrix row represents the elements including a predicted value while each column represents the elements for an actual value. As a metric for the closeness of the measurements to a specific value and to the interval with possible deviation of threshold by $\pm 1$ from the real value, the overall accuracy and accuracy $\pm 1$ are used. The results are summarised in Table 4.

Table 4. Accuracy for prediction with HS.

| HS | $q_{2}$ | $q_{3}$ | $q_{4}$ | $q_{5}$ |
| :--- | :--- | :--- | :--- | :--- |
| Accuracy | 0.8430 | 0.8778 | 0.7899 | 0.6282 |
| Accuracy $\pm 1$ | 0.9861 | 0.9884 | 0.9871 | 0.9769 |


(c)

(b)

(d)


Figure 3. Confusion matrices (a-d) for prediction of $q_{2}, q_{3}, q_{4}$ and $q_{5}$ using HS.

## 4. Artificial Neural Networks

Artificial Neural Networks (NN) belong to a set of supervised machine learning methods. It is most popular in different applied problems including data classification, pattern recognition, regression, clustering and time series forecasting. Here we show that the NN can give even more positive results compared to the HS that indicates the possibility to use it for predicting the structural control policies.

The data set $S(8)$ is used to explore predictions for the optimal threshold levels through the NN. The multilayer neural network is used for the data classification. It can be formally defined as a function $f: \vec{\alpha} \rightarrow \vec{y}$, which maps an input vector $\vec{\alpha}$ of dimension $2 m+1$ to an estimate output $\vec{y} \in \mathbb{R}^{N_{c}}$ of the class number $N=1, \ldots, N_{c}$. The network is decomposed into 6 layers as illustrated in Figure 4, each of which represents a different function mapping vectors to vectors. The successive layers are: a linear layer with an output vector of size $k$, a nonlinear elementwise activation layer, other three linear layers with output vectors of size $k$ and a nonlinear normalisation layer.


Figure 4. Architecture of the neural network.
The first layer is an affine transformation

$$
\vec{q}_{1}=W_{1} \vec{\alpha}+\vec{b}_{1},
$$

where $\vec{q}_{1}=\mathbb{R}^{2 m+1}$ is the output vector, $W \in \mathbb{R}^{2 m+1 \times k=30}$ is the weight matrix, $\vec{b}_{1} \in \mathbb{R}^{2 m+1}$ is the bias vector. The rows in $W_{1}$ are interpreted as features that are relevant for differentiating between corresponding classes. Consequently, $W_{1} \vec{\alpha}$ is a projection of the input $\vec{\alpha}$ onto these features. The second layer is an elementwise activation layer which is defined by the nonlinear function $\vec{q}_{2}=\max \left(0, \vec{q}_{1}\right)$ setting negative entries of $q_{1}$ to zero and uses only positive entries. The next three layers are other affine transformations,

$$
\vec{q}_{i}=W_{i} \vec{q}_{i-1}+\vec{b}_{i},
$$

where $\vec{q}_{i} \in \mathbb{R}^{k}, W_{i} \in \mathbb{R}^{k \times k}$, and $b_{i} \in \mathbb{R}^{k}, i=3,4,5$. The last layer is the normalisation layer $\vec{y}=\operatorname{softmax}\left(\vec{q}_{5}\right)$, whose componentwise is of the form

$$
y_{N}=\frac{e^{q_{5 N}}}{\sum_{N} e^{q_{5 N}}}, N=1, \ldots, N_{c}
$$

The last layer normalises the output vector $\vec{y}$ with the aim to get the values between 0 and 1 . The output $\vec{y}$ can be treated as a probability distribution vector, where the $N$ th element $y_{N}$ represents the likelihood that $\vec{\alpha}$ belongs to class $N$.

We use $70 \%$ of the same data $S$ which was not used to verify the quality of the HS in a training phase of the NN and the rest of $S$-as validation data. We train a multilayer (6-layer) NN using an adaptive moment estimation method [24] and the neural network toolbox in Mathematica® of the Wolfram Research. Then we verify the approximated function

$$
\hat{q}_{k}:=\hat{q}_{k}\left(\lambda, \mu_{1}, \ldots, \mu_{K}, c_{0}, c_{1}, \ldots, c_{K}\right)
$$

which should be accurate enough to be used to predict new output from verification data. The algorithm was ran many times on samples and networks with different sizes. In all cases the results were quite positive and indicate the potential of machine learning methodology for optimisation problems in the queueing theory.

Example 5. The results of estimations of the optimal threshold values using the trained NN are summarised again in form of confusion matrices, as is shown in Figure 5. The overall accuracy of classification and accuracies for the values with deviations are given in Table 5. We can see that the NN methodology exhibits even more accurate estimations for the optimal thresholds if the results are compared with the corresponding HS.

Table 5. Accuracy for prediction with NN.

| $\mathbf{N N}$ | $\boldsymbol{q}_{\mathbf{2}}$ | $\boldsymbol{q}_{3}$ | $\boldsymbol{q}_{4}$ | $\boldsymbol{q}_{5}$ |
| :--- | :--- | :--- | :--- | :--- |
| Accuracy | 0.9700 | 0.8785 | 0.8708 | 0.7977 |
| Accuracy $\pm 1$ | 0.9991 | 0.9951 | 0.9874 | 0.9962 |



Figure 5. Confusion matrices (a-d) for prediction of $q_{2}, q_{3}, q_{4}$ and $q_{5}$ using NN.

## 5. Conclusions

We combine classic methodology of analysing controllable queues with a heuristic solution and machine learning to study the possibility to estimate the values of optimal thresholds. Due to the fact that the results were quite positive, we can make the following general conclusion. With this study we confirm that the analysis of controlled queueing systems and the solution of optimisation problems using classical Markov decision theory can be successfully combined with machine learning techniques. These approaches do not contradict each other; on the contrary, combining them provides new results.

Author Contributions: Conceptualization, D.E.; formal analysis, investigation, methodology, software and writing, D.E. and N.S. Both authors have read and agreed to the published version of the manuscript.

Funding: Open Access Funding by the University of Linz. This research has been supported by the RUDN University Strategic Academic Leadership Program (recipient D. Efrosinin).
Institutional Review Board Statement: Not applicable.
Informed Consent Statement: Not applicable.
Data Availability Statement: Not applicable.
Acknowledgments: This paper has been supported by the RUDN University Strategic Academic Leadership Program (recipient D. Efrosinin), Open Access Funding by the University of Linz.

Conflicts of Interest: The authors declare no conflict of interest.

## References

1. Lin, W.; Kumar, P.R. Optimal control of a queueing system with two heterogeneous servers. IEEE Trans. Autom. Control 1984, 29, 696-703. [CrossRef]
2. Koole, G. A simple proof of the optimality of a threshold policy in a two-server queueing system. Syst. Control. Lett. 1995, 26, 301-303. [CrossRef]
3. Walrand, J. A note on: "Optimal control of a queuing system with two heterogeneous servers". Syst. Control Lett. 1984, 4, 131-134. [CrossRef]
4. Rykov, V. Monotone Control of Queueing Systems with Heterogeneous Servers. QUESTA 2001, 37, 391-403.
5. Crabill, T.; Gross, D.; Magazine, M.J. A classified bibliography of research on optimal design and control of queues. Oper. Res. 1977, 25, 219-232. [CrossRef]
6. Nobel, N. Hysteretic and Heuristic Control of Queueing Systems. Ph.D. Thesis, Vrije University Amsterdam, Amsterdam, The Netherlands, November 1998.
7. Nobel, R.; Tijms, H.C. Optimal control of a queueing system with heterogeneous servers and set-up costs. IEEE Trans. Autom. Control 2000, 45, 780-784. [CrossRef]
8. Le Ny, L.-M.; Tuffin, B. A Simple Analysis of Heterogeneous Multi-Server Threshold Queues with Hysteresis; Institut National de Recherche en Informatique: Nancy, France, 2000.
9. Efrosinin, D. Controlled Queueing Systems with Heterogeneous Servers: Dynamic Optimization and Monotonicity Properties of Optimal Control Policies in Multiserver Heterogeneous Queues; VDM Verlag: Saarbrücken, Germany, 2008.
10. Rykov, V.; Efrosinin, D. On the slow server problem. Autom. Remote. Control 2010, 70, 2013-2023. [CrossRef]
11. Howard, R. Dynamic Programming and Markov Processes; Wiley Series; Wiley: London, UK, 1960.
12. Puterman, M.L. Markov Decision Process; Wiley Series in Probability and Mathematical Statistics; Wiley: London, UK, 1994.
13. Tijms, H.C. Stochastic Models. An Algorithmic Approach; John Wiley and Sons: New York, NY, USA, 1994.
14. Gershenson, C. Artificial Neural Networks for Beginners; 2003. Available online: http://arxiv.org/abs/cs/0308031 (accessed on 20 August 2003).
15. Rätsch, G. A Brief Introduction into Machine Learning; Friedrich Miescher Laboratory of the Max Planck Society: Tuebinger, Germany, 2004.
16. Russel, S.J.; Norvig, P. Artificial Intelligence. A Modern Approach; Prentice-Hall, Inc.: Upper Saddle River, NJ, USA, 1995.
17. Kyritsis, A.I.; Deriaz, M. A machine learning approach to waiting time prediction in queueing scenarios. In Proceedings of the Second International Conference on Artificial Intelligence for Industries, Laguna Hills, CA, USA, 25-27 September 2019; pp. 17-21.
18. Stintzing, J.; Norrman, F. Prediction of Queuing Behaviour through the Use of Artificial Neural Networks. Available online: http://www.diva-portal.se/smash/get/diva2:1111289/FULLTEXT01.pdf (accessed on 18 June 2017).
19. Xia, L.; Zhang, Z.G.; Li, Q.-L.; Glynn, P.W. A $c / \mu$-Rule for Service Resource Allocation in Group-Server Queues. arXiv 2018, arXiv:1807.05367.
20. Aviv, Y.; Federgruen, A. The value-iteration method for countable state Markov decision processes. Oper. Res. Lett. 1999, 24, 223-234. [CrossRef]
21. Özkan, E.; Kharoufeh, J.P. Optimal control of a two-server queueing system with failures. Probab. Eng. Inform. Sci. 2014, 28, 489-527. [CrossRef]
22. Sennott, L.I. Stochastic Dynamic Programming and the Control of Queueing Systems; Wiley: New York, NY, USA, 1999.
23. Efrosinin, D.; Sztrik, J. An algorithmic approach to analyzing the reliability of a controllable unreliable queue with two heterogeneous servers. Eur. J. Oper. Res. 2018, 271, 934-952. [CrossRef]
24. Kingma, D.P.; Adam, J.B. A Method for Stochastic Optimization; 2015. Available online: https:/ / arxiv.org/abs/1412.6980 (accessed on 30 January 2017).
