Mathematics

https://www.aimspress.com/journal/Math

AIMS Mathematics, 10(5): 10650–10674.
DOI: 10.3934/math.2025485
Received: 04 February 2025
Revised: 16 April 2025
Accepted: 23 April 2025
Published: 09 May 2025

Research article

Retrial tandem queueing system with correlated arrivals

Vladimir Vishnevsky[1], **Valentina Klimenok**[2], **Olga Semenova**[1,*], **and Minh Cong Dang**[3]

[1] Institute of Control Sciences of Russian Academy of Sciences, Moscow, Russia
[2] Belarusian State University, Minsk, Belarus
[3] Moscow Institute of Physics and Technology, Dolgoprudny, Moscow Region, Russia

* **Correspondence:** Email: olgasmnv@gmail.com; Tel: +7-495-198-17-20.

Abstract: In this paper, we investigate a retrial tandem queueing system with a finite number of queues. Each queue consists of a finite buffer and a single server with phase-type distributed service times. Customers arrive at the system according to a Markovian arrival process (MAP). At any queue, a customer that finds both the server busy and the buffer full enters a common orbit and makes repeated attempts to rejoin the first queue after exponentially distributed time intervals. This system models telecommunication networks with linear topology that implement retransmission protocols for lost data packets. We provide a complete mathematical analysis for the two-queue system with an infinite-capacity orbit, deriving the ergodicity condition, stationary state distribution, and key performance characteristics. For systems with an arbitrary number of queues, we develop a comprehensive solution approach that combines queueing theory methods, discrete-event simulation, and machine learning techniques to predict the mean sojourn time. We implement and compare multiple machine learning methods, evaluating their predictive performance through extensive numerical experiments.

Keywords: tandem queues; retrial; Markovian arrival process; phase-type distribution; stationary performance characteristics; machine learning
Mathematics Subject Classification: 60K25, 60K30, 68T07

1. Introduction

Modern telecommunication networks have become critical infrastructure components in contemporary society. The efficient operation of these networks relies heavily on accurate mathematical modeling techniques. Consequently, queueing theory methods have received increasing attention in telecommunications research. A particularly active research direction involves analyzing queueing networks by studying their fundamental building blocks under realistic assumptions about arrival processes and service mechanisms. Within this domain, two-stage queueing systems have attracted significant interest as they represent both an important subclass of linear-topology networks and effective models for network segments in general topologies [1, 2].

The literature contains numerous studies on two-stage queueing systems. Early work primarily examined systems with Poisson input processes (see, e.g., [3]). However, modern telecommunication traffic exhibits non-stationarity and correlation that cannot be adequately captured by Poisson models. More sophisticated mathematical models for such traffic were introduced by Neuts [4] and Lucantoni [5] through the batch Markovian arrival process (BMAP) and its single-arrival variant MAP. These have become the predominant models for correlated bursty traffic, leading to numerous analytical studies of tandem queueing systems with MAP arrivals [6–8], BMAP arrivals [9] or marked MAP (MMAP) arrivals [10] in recent decades.

While retrial mechanisms are inherent in many telecommunication systems (e.g., TCP's retransmission protocol), relatively few studies have addressed retrial tandem queueing systems with non-MAP arrivals [11, 12] or BMAP/MAP arrivals [13–15]. The mathematical analysis of retrial systems presents greater challenges than conventional queueing systems due to the spatial inhomogeneity of their underlying Markov chains [16]. Comprehensive reviews of early retrial queue research (pre-2008) can be found in [17, 18], with MAP/BMAP models proving particularly valuable for modern network design and performance evaluation [19–21]. Most existing work assumes retrials only occur from the first station, with blocked customers at subsequent stations being lost. Our work addresses the more general case of a common orbit serving all stations. To our knowledge, only [22–24] have considered such systems.

In [22], for a tandem queueing system with a finite arbitrary number of stations, the authors elaborate on the approximation procedures to find the mean sojourn time. No restrictions are imposed on the distributions characterizing the input arrivals, service times, and inter-retrial times. The assumption is that the blocking probability at every station is known and fixed. Based on results for the system with two stations, the authors conclude that the approximation works well under light traffic. The paper [23] is devoted to a tandem queueing system with two single server stations with a common orbit for retrials and no buffers. The service times at the first server have a general distribution, and the service times at the second server are distributed exponentially. Assuming that the retrial rate is extremely small, the authors prove that the scaled version of the number of customers in the orbit asymptotically follows a diffusion process, which is further utilized to obtain an approximation to the number of customers in the orbit in the stationary state.

The system in [24] is similar to one studied in [23], but the input is MMPP (Markov-modulated Poisson arrivals), and the service times at the first server are exponentially distributed. The authors obtain the necessary condition for the Markov chain describing the system behavior to be ergodic and the asymptotic distribution of the scaled version of the number of customers in the orbit in case the retrial rate is extremely small.

In the present paper we consider the tandem (multi-station linear topology) model with a common orbit and an arbitrary number of stations. In contrast to known publications, the paper proposes a new model of a multi-phase queuing system of large size with an orbit and retrial facility. The novelty of the model is presented by the following aspects: 1) The input is MAP, which is more general than MMPP; 2) service times have PH distribution; 3) the system stations have finite buffers. The common orbit mechanism handles all blocked customers uniformly: whether arriving externally at the first station or transitioning internally from previous stations, any customer encountering a full buffer joins the orbit and must restart service from the first station. This architecture models wireless networks with linear topologies, extending the framework of [25] with retrial capabilities while excluding cross traffic.

Unlike the papers listed above, we propose *the combined solutions* and analyze the model having an arbitrary number of stations by combining the methods of queueing theory, simulation tools and machine learning methods. As the investigation of retrial tandem queueing system with more than two stations is of considerable difficulty, the system analysis by means of queueing theory methods is given here for the case of two stations. The math part of the analysis is based on the matrix analytical method. In the second part, for the system with arbitrary finite number of stations, we study employing machine learning methods to predict the mean sojourn time of system. Machine learning applications in queueing theory are well-established [26–28]. A tandem queueing system closely related to our model has been studied using machine learning methods [29]; however, each system requires careful method selection and parameter tuning. The paper is organized as follows: Section 2 presents the problem formulation and mathematical model. In Section 3 we provide the closed-form solution for the case of two stations by means of queueing theory methods and present the algorithm to calculate the performance measures. In Section 4 we apply the machine learning methods to predict the mean sojourn time for the systems with an arbitrary number of stations. We also provide the comparison analysis of the analytical and machine learning results. Section 5 contains the conclusion to the paper.

2. Problem statement

In this paper we consider a tandem queuing system with a finite number of stations and a common orbit (Figure 1). The input arrivals to the system of MAP type; each station consists of a single server with a finite buffer in front of it, and the service times of each server have a phase-type (PH) distribution. The PH distribution provides a versatile modeling framework that generalizes many common distributions used in queueing theory. As demonstrated in [30], PH distributions can approximate any positive-valued distribution with arbitrary accuracy, making them particularly valuable for modeling realistic service time distributions.

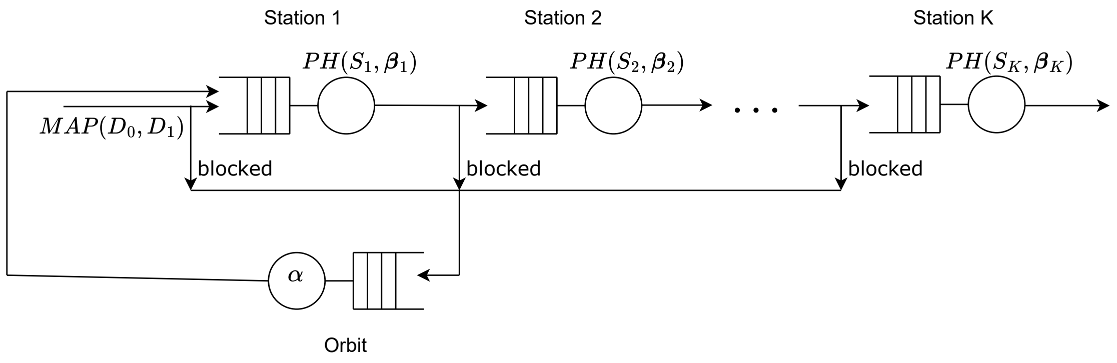

Figure 1. Tandem queueing system with a common retrial orbit.

A tandem queueing system consists of N stations and a common orbit for retrials, $N \geq 2$. Station i is a single server queueing system with finite buffer of size K_i. Customers arrive at the first station according to a Markovian arrival process (MAP) which is specified by the underlying process (Markov chain) ν_t, $t \geq 0$, with the state space $\{0, 1, \dots, W\}$ and $(W + 1) \times (W + 1)$ matrices D_0 and D_1. In the MAP, customers can arrive only at the transition moments of process ν_t, $t \geq 0$. The rates of ν_t, $t \geq 0$ transitions accompanied by the generation of a customer are specified by the matrix D_1. "Idle" transitions of the underlying process not accompanied by a generation of a customer are specified by the off-diagonal entries of the matrix D_0. The matrix $D_0 + D_1$ is an infinitesimal generator of the process $\nu_t, t \geq 0$. The average rate λ of arrivals is defined as $\lambda = \boldsymbol{\theta} D_1 \mathbf{e}$, where $\boldsymbol{\theta}$ is a vector of the stationary distribution of the process $\nu_t, t \geq 0$. This vector is calculated as the unique solution of the system $\boldsymbol{\theta}(D_0 + D_1) = \mathbf{0}, \boldsymbol{\theta}\mathbf{e} = 1$. Here and below, $\mathbf{e}$ is a column vector consisting of ones, and $\mathbf{0}$ is a row vector consisting of zeros. A more detailed description of MAP as a partial case of BMAP and its properties can be found in [5, 30].

The service time of a customer by the k-th server has a PH distribution with an irreducible representation $(\boldsymbol{\beta}_k, S_k), k = \overline{1, N}$. This time can be interpreted as time until the underlying Markov chain $m_t^{(k)}, t \geq 0$, with a finite state space $\{1, \dots, M_k, M_k + 1\}$ reaches the single absorbing state $M_k + 1$ given that the initial state of this process is chosen within the states $1, \dots, M_k$ according to the stochastic row vector $\boldsymbol{\beta}_k = (\beta_k^{(1)}, \dots, \beta_k^{(M_k)})$. Transition rates of the process $m_t^{(k)}$ within the set $\{1, \dots, M_k\}$ are defined by the sub-generator S_k, and transition rates into the absorbing state (which lead to a service completion) are given by the entries of the column vector $\boldsymbol{S}_0^{(k)} = -S_k \mathbf{e}$. The service rate is calculated as $\mu_k = -[\boldsymbol{\beta}_k S_k^{-1} \mathbf{e}]^{-1}$ and average service time is $b_k = \mu_k^{-1}, k = \overline{1, N}$. More information about PH distribution can be found in [30, 31].

The system operates under the following service discipline:

- **First Station Arrivals**:
  - If the server is idle, service begins immediately,
  - If the server is busy but buffer space is available, the customer joins the buffer queue,
  - If both server and buffer are occupied, the customer enters the infinite-capacity orbit.
- **Orbital Behavior**:
  - Customers in orbit make independent retrials after exponentially distributed time intervals,
  - Each retrial attempts to access either:
    - The first station's server (if available), or
    - Its buffer (if server busy but space available).
- **Inter-station Transitions** ($k = 1, ..., N - 1$):
  - Upon completing service at station k, the customer:
    - Immediately starts service at station $k + 1$ if its server is idle,
    - Joins station $k + 1$'s buffer if server busy but space available,
    - Enters the common orbit if station $k + 1$'s buffer is full.
- **Service Completion**:
  - Customers reaching station N exit the system after service completion,
  - Orbital customers persist until successfully completing all N service stages.

Thus, each customer in the orbit, no matter which station it came from, makes repeated attempts to get to the first station at random time intervals, exponentially distributed with the parameter α_i, where i is the total number of customers in orbit. We assume that $\alpha_0 = 0$, $\alpha_i \to \infty$ for $i \to \infty$. Such a dependence includes, in particular, the linear retrial strategy $\alpha_i = i\alpha, \alpha > 0$.

3. Analysis of two stations model

Our primary objective is to derive key performance metrics, particularly focusing on the mean sojourn time. The subsequent sections develop our analytical approach, where we analyze the model by means of queueing theory methods. Below we consider the stochastic process describing the system behavior and show that it is an asymptotically quasi-Toeplitz Markov chain (AQTMC) [32]. This allows us to apply the algorithm elaborated in [32] to calculate the stationary state distribution and get the performance characteristics. Due to the complexity of the analysis with an arbitrary K, below we present results for the system with two stations, $K = 2$. In Section 4, the results obtained for $K = 2$ are used to verify the simulation model for arbitrary K, and both results from the analytical solution and simulation are used for machine learning tools to get results for an arbitrary number of stations.

3.1. Markov Chain

Let at time t:

- i_t be a number of customers in the orbit, $i \geq 0$,
- j_t be a number of customers at the first station, $j_t = \overline{0, K_1}$,
- n_t be a number of customers at the second station, $n_t = \overline{0, K_2}$,
- ν_t be the state of the underlying process of the MAP, $\nu_t = \overline{0, W}$,
- $m_t^{(k)}$ be the state of the PH service process on the k-th station server, $m_t^{(k)} = \overline{1, M^{(k)}}$, $k = 1, 2$.

The process of the system operation is described by a regular irreducible Markov chain $\xi_t, t \geq 0$, with state space

$$\Omega = \{(i, j, n, \nu), i \geq 0, j = 0, n = 0, \nu = \overline{0, W}\} \bigcup$$
$$\{(i, j, n, \nu, m^{(1)}), i \geq 0, j = \overline{0, K_1}, n = 0, \nu = \overline{0, W}, m^{(1)} = \overline{1, M^{(1)}}\} \bigcup$$
$$\{(i, j, n, \nu, m^{(2)}), i \geq 0, j = 0, n = \overline{0, K_2}, \nu = \overline{0, W}, m^{(2)} = \overline{1, M^{(2)}}\} \bigcup$$
$$\{(i, j, n, \nu, m^{(1)}, m^{(2)}), i \geq 0, j = \overline{0, K_1}, n = \overline{0, K_2}, \nu = \overline{0, W}, m^{(1)} = \overline{1, M^{(1)}}, m^{(2)} = \overline{1, M^{(2)}}\}.$$

In what follows, we assume that the states of the Markov chain under consideration are ordered in lexicographical order and form an infinitesimal generator Q of the Markov chain. Let us denote by $Q_{i,i'}$ the matrix of transition rates of the Markov chain from the states corresponding to the level i of the countable component i_t to the states corresponding to the level i', $i, i' \geq 0$. Matrix $Q_{i,i'}$ consists of blocks $Q_{i,i'}(j, j')$, $j, j' = \overline{0, K_1}$, containing the rates of transitions from the states corresponding to the levels i, j of components i_t, j_t to the states corresponding to the levels i', j' of these components. Matrix $Q_{i,i'}(j, j')$ consists of blocks $Q_{i,i'}^{(n,n')}(j, j'), n, n' = \overline{0, K_2}$, containing the rates of transitions of the underlying process of the MAP and the PH service processes on the first and the second servers from the states corresponding to the levels i, j, n of components i_t, j_t, n_t to the states corresponding to the levels i', j', n' of these components. Then the infinitesimal generator Q has the following block structure:

$$Q = \left(Q_{i,i'}\right)_{i,i' \geq 0} = \left(Q_{i,i'}(j, j')\right)_{j,j'=\overline{0,K_1},\ i,i' \geq 0} = \left(Q_{i,i'}^{(n,n')}(j, j')\right)_{n,n'=\overline{0,K_2},\ j,j'=\overline{0,K_1},\ i,i' \geq 0}. \qquad (1)$$

In order to describe the matrix Q, we use the following notations:

- I is an identity matrix, O is a zero matrix, I_n is an identity matrix of size $n \times n$, O_n is a zero matrix of size $n \times n$, $O_{m\times n}$ is a zero matrix of size $m \times n$,
- $diag^{+}\{a_1, a_2, \dots, a_n\}$ is a matrix that consists of $(n + 1) \times (n + 1)$ blocks (not necessarily square) whose over-diagonal blocks are equal to the matrices listed in the brackets and the remaining blocks are zero,
- $diag^{-}\{a_1, a_2, \dots, a_n\}$ is a matrix that consists of $(n+1)\times(n+1)$ blocks whose sub-diagonal blocks are equal to the matrices listed in the brackets and the remaining blocks are zero,
- $\delta(i, j)$ is the Kronecker symbol.

The infinitesimal generator Q of the Markov chain $\xi_t, t \geq 0$ has a block tridiagonal structure:

$$Q = \begin{pmatrix} Q_{0,0} & Q_{0,1} & O & O & O & \cdots \\ Q_{1,0} & Q_{1,1} & Q_{1,2} & O & O & \cdots \\ O & Q_{2,1} & Q_{2,2} & Q_{2,3} & O & \cdots \\ O & O & Q_{3,2} & Q_{3,3} & Q_{3,4} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

with the non-zero blocks $Q_{i,i'}$ also have a block tridiagonal structure and are defined by the following matrices:

$$Q_{i,i-1}(j, j-1) = O, i \geq 1, j = \overline{1, K_1},$$

$$Q_{i,i-1}(j, j) = O, i \geq 1, j = \overline{0, K_1},$$

$$Q_{i,i-1}(0, 1) = \alpha_i diag\{I_W \otimes \boldsymbol{\beta}_1, \underbrace{I_W \otimes \boldsymbol{\beta}_1 \otimes I_{M_2}}_{K_2}\}, i \geq 1,$$

$$Q_{i,i-1}(j, j+1) = \alpha_i I_{WR}, i \geq 1, j = \overline{1, K_1 - 1},$$

$$Q_{i,i}(1, 0) = diag^{+}\{I_W \otimes \boldsymbol{S}_0^{(1)} \otimes \boldsymbol{\beta}_2, \underbrace{I_W \otimes \boldsymbol{S}_0^{(1)} \otimes I_{M_2}}_{K_2-1}\}, i \geq 0,$$

$$Q_{i,i}(j, j-1) = diag^{+}\{I_W \otimes \boldsymbol{S}_0^{(1)}\boldsymbol{\beta}_1 \otimes \boldsymbol{\beta}_2, \underbrace{I_W \otimes \boldsymbol{S}_0^{(1)}\boldsymbol{\beta}_1 \otimes I_{M_2}}_{K_2-1}\}, i \geq 0, j = \overline{2, K_1},$$

$$Q_{i,i}(0, 0) = diag\{D_0, \underbrace{D_0 \oplus S_2}_{K_2}\} + diag^{-}\{I_W \otimes \boldsymbol{S}_0^{(2)}, \underbrace{I_W \otimes \boldsymbol{S}_0^{(2)}\boldsymbol{\beta}_2}_{K_2-1}\} - \alpha_i I, i \geq 0,$$

$$Q_{i,i}(j, j) = diag\{D_0 \oplus S_1, \underbrace{D_0 \oplus S_1 \oplus S_2}_{K_2}\} +$$

$$diag^{-}\{I_{WM_1} \otimes \boldsymbol{S}_0^{(2)}, \underbrace{I_{WM_1} \otimes \boldsymbol{S}_0^{(2)}\boldsymbol{\beta}_2}_{K_2-1}\} - [1 - \delta(j, K_1)]\alpha_i I, i \geq 0, j = \overline{1, K_1},$$

$$Q_{i,i}(0, 1) = diag\{D_1 \otimes \boldsymbol{\beta}_1, \underbrace{D_1 \otimes \boldsymbol{\beta}_1 \otimes I_{M_2}}_{K_2}\}, i \geq 0,$$

$$Q_{i,i}(j, j+1) = diag\{D_1 \otimes I_{M_1}, \underbrace{D_1 \otimes I_{M_1} \otimes I_{M_2}}_{K_2}\}, i \geq 0, j = \overline{1, K_1 - 1},$$

$$Q_{i,i+1}(1, 0) = diag\{O_{WM_1\times W}, \underbrace{O_{WM_1M_2\times WM_2}}_{K_2-1}, I_W \otimes \boldsymbol{S}_0^{(1)} \otimes I_{M_2}\}, i \geq 0,$$

$$Q_{i,i+1}(j, j-1) = diag\{O_{W(R-M_1M_2)}, I_W \otimes \boldsymbol{S}_0^{(1)}\boldsymbol{\beta}_1 \otimes I_{M_2}\}, i \geq 0, j = \overline{2, K_1},$$

$$Q_{i,i+1}(j, j) = O, i \geq 0, j = \overline{1, K_1 - 1},$$

$$Q_{i,i+1}(K_1, K_1) = diag\{D_1 \otimes I_{M_1}, \underbrace{D_1 \otimes I_{M_1M_2}}_{K_2}\}, i \geq 0,$$

$$Q_{i,i+1}(j, j+1) = O, i \geq 0, j = \overline{1, K_1 - 1},$$

where $R = M_1(1 + M_2K_2)$.

Explanation: It is seen from formula (1) that an infinitesimal generator Q of the Markov chain $\xi_t, t \geq 0$ consists of zeroes and blocks $Q_{i,i'}(j, j')$ containing rates of transitions from the states corresponding to the levels i, j of the chain components i_t, j_t to the states corresponding to the levels i', j' of these components.

Blocks $Q_{i,i-1}(j, j')$ describe the rates of transitions of the Markov chain $\xi_t, t \geq 0$, which lead to a decrease in the number of customers in the orbit by one and a change in the number of customers at the first station from j to j'. Let $j' = j - 1$. As it follows from the description of the queue under consideration, the transitions from the states corresponding to the levels i, j of i_t, j_t to the states corresponding to the levels $i - 1, j - 1$ and $i - 1, j$ of these components are impossible. Therefore, $Q_{i,i-1}(j, j-1) = O$, $Q_{i,i-1}(j, j) = O$.

Blocks $Q_{i,i-1}(j, j+1)$ describe the rates of transitions of the Markov chain $\xi_t, t \geq 0$ which entails the successful retry from the orbit to the first station. If the first station is empty ($j = 0$), such transitions are accompanied by the starting of the initial phase for the service on the first server according to the probabilistic vector $\boldsymbol{\beta}_1$.

Blocks $Q_{i,i}(j, j')$ describe the rates of transitions of the Markov chain $\xi_t, t \geq 0$ that do not lead to a change of the number of customers in the orbit. Let $j' = j-1$. If $j = 1$, the transitions are accompanied by the completion of the service of a customer on the first server (the matrix $\boldsymbol{S}_0^{(1)}$) and the transmission of this customer to the second station. If the server of the second station is idle ($n = 0$), the customer goes to the initial phase of the PH service process on the second server according to the probabilistic vector $\boldsymbol{\beta}_2$ and starts its service. Otherwise, the customer is buffered. In the case $j > 1$ the released server of the first station is occupied by a customer from the buffer. This customer goes to the initial phase for the PH service process on the first server according to the probabilistic vector $\boldsymbol{\beta}_1$ and starts its service. In the case $j' = j$, transition rates of the Markov chain depend on the level j:

(1) $j = 0$. The matrix $Q_{i,i}(0, 0)$ contains the rates of transitions of the Markov chain $\xi_t, t \geq 0$ caused by:
- Idle transitions of the underlying process of the MAP (the matrix D_0), if $n = 0$;
- Idle transitions of the underlying processes of the MAP or the PH service on the second server (the matrix $D_0 \oplus S_2$) or transitions which are accompanied by the completion of the service on the second server (the matrix $\boldsymbol{S}_0^{(2)}$) , if $n = 1$;
- Idle transitions of the underlying processes of the MAP or the PH service on the second server (the matrix $D_0 \oplus S_2$) or transitions which are accompanied by the completion of the service on the second server and installation of the initial phase for the next service on this server (the matrix $\boldsymbol{S}_0^{(2)}\boldsymbol{\beta}_2$), if $n > 1$.

(2) $0 < j \leq K_1$. The matrix $Q_{i,i}(j, j)$ contains the rates of transitions of the Markov chain $\xi_t, t \geq 0$ caused by:
- Idle transitions of the underlying processes of the MAP or the PH service on the first server (the

matrix $D_0 \oplus S_1$), if $n = 0$;
- Idle transitions of the underlying processes of the MAP or the PH service on the first or the second server (the matrix $D_0 \oplus S_1 \oplus S_2$) or transitions which are accompanied by the completion of the service on the second server (the matrix $\boldsymbol{S}_0^{(2)}$), if $n = 1$;
- Idle transitions of the underlying processes of the MAP or the PH service on the first or the second server (the matrix $D_0 \oplus S_1 \oplus S_2$) or transitions which are accompanied by the completion of the service on the second server and installation of the initial phase for the next service on this server (the matrix $\boldsymbol{S}_0^{(2)}\boldsymbol{\beta}_2$), if $n > 1$. Note that $\boldsymbol{S}_0^{(2)}\boldsymbol{\beta}_2$ is the square matrix whose entry (i, j) is the product of the i-th entry of vector $\boldsymbol{S}_0^{(2)}$ by the j-th entry of vector $\boldsymbol{\beta}_2$,

$$\boldsymbol{S}_0^{(2)}\boldsymbol{\beta}_2 = \left(\left(\boldsymbol{S}_0^{(2)}\right)_i (\boldsymbol{\beta}_2)_j\right)_{i,j=\overline{1,M_2}}.$$

Blocks $Q_{i,i+1}(j, j')$ describe the rates of transitions of the Markov chain $\xi_t, t \geq 0$ that leads to an increase in the number of customers in the orbit by one.

Blocks $Q_{i,i+1}(j, j), Q_{i,i+1}(j, j+1), j = \overline{0, K_1 - 1}$, consist of zeroes. This follows from the description of the queue under consideration.

Blocks $Q_{i,i+1}(j, j-1), j = \overline{1, K_1}$, contain the rates of transitions of the Markov chain $\xi_t, t \geq 0$ that occur when the second station is completely occupied at the moment of the service completion of a customer on the first station (the column vector $\boldsymbol{S}_0^{(1)}$). Then this customer is forced to go into the orbit, and the initial phase for the next service is installed on the first server (the row vector $\boldsymbol{\beta}_2$).

Blocks $Q_{i,i+1}(K_1, K_1)$ contain the rates of transitions of the Markov chain $\xi_t, t \geq 0$ caused by an arrival of a customer in the MAP (the matrix D_1). This customer is forced to go into orbit since the first station buffer is full upon its arrival.

Corollary 1. *The Markov chain $\xi_t, t \geq 0$, is an asymptotically quasi-Toeplitz Markov chain (AQTMC) defined in [32].*

Proof. Let T_i be a diagonal matrix with diagonal entries defined as the absolute values of the diagonal entries of the matrix $Q_{i,i}, i \geq 0$. According to [32], the corollary will be proven if we show that limits

$$Y_0 = \lim_{i\to\infty} T_i^{-1} Q_{i,i-1},\ Y_1 = \lim_{i\to\infty} T_i^{-1} Q_{i,i} + I,\ Y_2 = \lim_{i\to\infty} T_i^{-1} Q_{i,i+1}$$

exist and matrix $Y_0 + Y_1 + Y_2$ is stochastic.

Note that the matrices $Q_{i,i}(K_1, K_1 - 1)$, $Q_{i,i}(K_1, K_1)$, $Q_{i,i+1}(K_1, K_1)$, and $Q_{i,i+1}(K_1, K_1 - 1)$ do not depend on i and K_1. Henceforth we will use the notations A, B, C for these matrices. Namely,

$$A = Q_{i,i}(K_1, K_1 - 1), B = Q_{i,i}(K_1, K_1) + Q_{i,i+1}(K_1, K_1), C = Q_{i,i+1}(K_1, K_1 - 1). \tag{2}$$

Then, after some algebra, we obtain the following expressions for the matrices Y_0, Y_1, Y_2:

$$Y_0 = \begin{pmatrix} O & I_{WR} & O & \dots & O \\ O & O & I_{WR} & \dots & O \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ O & O & O & \dots & I_{WR} \\ O & O & O & \dots & O \end{pmatrix},$$

$$Y_1 = \begin{pmatrix} O & \dots & O & O \\ \vdots & \ddots & \vdots & \vdots \\ O & \dots & O & O \\ O & \dots & T^{-1}A & T^{-1}C + I \end{pmatrix},$$

$$Y_2 = \begin{pmatrix} O & \dots & O & O \\ \vdots & \ddots & \vdots & \vdots \\ O & \dots & O & O \\ O & \dots & O & T^{-1}B \end{pmatrix}$$

where T is a diagonal matrix formed by the moduli of the diagonal entries of the matrix C.

It is easy to check that the matrix $Y_0 + Y_1 + Y_2$ is stochastic. Thus, the corollary is proved. □

3.2. Ergodicity condition

The ergodicity condition for the AQTMC $\xi_t, t \geq 0$, can be formulated in terms of the matrices Y_0, Y_1, Y_2. Following [32], we first obtain an expression for the generating function $Y(z)$ of these matrices.

Lemma 1. *The matrix generating function $Y(z) = Y_0 + Y_1 z + Y_2 z^2$ has the form*

$$Y(z) = \begin{pmatrix} O_{WR} & I_{WR} & O & \dots & O & O \\ O & O & I_{WR} & \dots & O & O \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ O & O & O & \dots & O & I_{WR} \\ O & O & O & \dots & T^{-1}Az & z[T^{-1}(C + Bz)] + zI \end{pmatrix}$$

where matrices A, B, C are defined in (2).

Theorem 1. *(i) The Markov chain $\xi_t, t \geq 0$ is ergodic if the following inequality holds:*

$$\lambda < \frac{1}{2}\Big\{[\boldsymbol{y}_0 + \sum_{n=1}^{K_2-1} \boldsymbol{y}_n(I_{M_1} \otimes \boldsymbol{e}_{M_2})]\boldsymbol{S}_0^{(1)} + \boldsymbol{y}_{K_2}(\boldsymbol{e}_{M_1} \otimes I_{M_2})\boldsymbol{S}_0^{(2)}\Big\} \tag{3}$$

where vector $\boldsymbol{y} = (\boldsymbol{y}_0, \boldsymbol{y}_1, \dots, \boldsymbol{y}_{K_2})$ is the unique solution of the system

$$\boldsymbol{y}V = 0, \ \boldsymbol{y}\boldsymbol{e} = 1 \tag{4}$$

where matrix V has the form

$$V = diag^{+}\{\boldsymbol{S}_0^{(1)}\boldsymbol{\beta}_1 \otimes \boldsymbol{\beta}_2, \underbrace{\boldsymbol{S}_0^{(1)}\boldsymbol{\beta}_1 \otimes I_{M_2}}_{K_2-1}\}$$

$$+diag\{S_1, \underbrace{S_1 \otimes S_2}_{K_2-1}, S_1 \otimes S_2 + \boldsymbol{S}_0^{(1)}\boldsymbol{\beta}_1 \otimes I_{M_2}\} + diag^{-}\{I_{M_1} \otimes \boldsymbol{S}_0^{(2)}, \underbrace{I_{M_1} \otimes \boldsymbol{S}_0^{(2)}\boldsymbol{\beta}_2}_{K_2-1}\};$$

(ii) The Markov chain $\xi_t, t \geq 0$ is non-ergodic if the inequality (3) taken with the opposite sign holds.

Proof. Matrix $Y(1)$ is reducible. Let us denote by $Y^{\{N\}}(1)$ its normal form (see [33]):

$$Y^{\{N\}}(z) = \begin{pmatrix} z[T^{-1}(C+Bz)]+zI & T^{-1}Az & O & \dots & O & O \\ I_{WR} & O & O & \dots & O & O \\ O & I_{WR} & O & \dots & O & O \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ O & O & O & \dots & I_{WR} & O \end{pmatrix}.$$

It is easy to see that matrix $Y^{\{N\}}(1)$ contains only one irreducible stochastic diagonal block. The corresponding block of matrix $Y^{\{N\}}(z)$ has the form

$$\tilde{Y}(z) = \begin{pmatrix} z[T^{-1}(C+Bz)]+zI & T^{-1}Az \\ I_{WR} & O \end{pmatrix}.$$

From ([32], Theorem 2), it follows that the Markov chain $\xi_t, t \geq 0$ is ergodic if

$$[\det(zI - \tilde{Y}(z))]'_{z=1} > 0. \tag{5}$$

Using the block structure of matrix $\tilde{Y}(z)$, we can reduce the determinant in (5) to the following form:

$$\det(zI - \tilde{Y}(z)) = \det(zT^{-1})\det[-z(C+Bz) - A]. \tag{6}$$

At $z = 1$, the second determinant on the right side of (6) is zero due to the properties of an infinitesimal generator. Note that $\det(zT^{-1}) > 0$ for $z = 1$. Therefore, inequality (5) is equivalent to the following inequality:

$$[\det(-z(C+Bz) - A)]'_{z=1} > 0. \tag{7}$$

Following the proof of Theorem 2, [32], we can show that inequality (7) is equivalent to

$$\boldsymbol{x}(C + 2B)\mathbf{e} < 0 \tag{8}$$

where $\boldsymbol{x}$ is the unique solution of the system

$$\boldsymbol{x}(C + B + A) = 0, \boldsymbol{x}\mathbf{e} = 1. \tag{9}$$

Consider vector $\boldsymbol{x}$ in the form

$$\boldsymbol{x} = (\boldsymbol{\theta} \otimes \boldsymbol{y}_0, \boldsymbol{\theta} \otimes \boldsymbol{y}_1, \dots, \boldsymbol{\theta} \otimes \boldsymbol{y}_{K_2}) \tag{10}$$

where vector $\boldsymbol{y}_0$ is of order M_1, and vectors $\boldsymbol{y}_1, \dots, \boldsymbol{y}_{K_2}$ are of order $M_1 M_2$.

Next, we substitute vector $\boldsymbol{x}$ in the form (10) and the expressions for matrices A, B, C defined in (2) into (9). Taking into account that $\boldsymbol{\theta}(D_0 + D_1) = 0$, $\boldsymbol{\theta}\mathbf{e} = 1$, after some algebra we obtain (4) for vector $\boldsymbol{y} = (\boldsymbol{y}_0, \boldsymbol{y}_1, \dots, \boldsymbol{y}_{K_2})$. Thus, system (9) for vector $\boldsymbol{x}$ is reduced to (4) for vector $\boldsymbol{y}$.

Now consider inequality (8). Substituting vector $\boldsymbol{x}$ in form (10) and matrices B, C defined by (2) into (8) and taking into account the relation $\boldsymbol{\theta} D_1 \mathbf{e} = \lambda$, after some algebraic transformations we obtain the inequality (3) equivalent to (8).

Thus, the statement (i) of the theorem is proven.

Taking into account the statement (i), we immediately prove statement (ii) of the theorem by using the results of [32], Theorem 2. □

Corollary 2. *In the case of the tandem system without buffers in front of the servers and exponential distributions of service times at both the stations:*

(i) the sufficient condition for the Markov chain $\xi_t, t \geq 0$ ergodicity is

$$\lambda < \frac{\mu_1\mu_2}{\mu_1 + \mu_2}; \tag{11}$$

(ii) the sufficient condition for the Markov chain $\xi_t, t \geq 0$ non-ergodicity is inequality (11) taken with the opposite sign.

Proof. of the corollary follows from Theorem 1, given that $K_1 = K_2 = 1, M_k = 1, S_k = -\mu_k, \boldsymbol{\beta}_k = 1, k = 1, 2$.

Note that the system, defined in Corollary 2, coincides with the system investigated in [24] if we assume that the input arrivals are not a MAP, but its special case – an MMPP. Paper [24] provides the necessary condition for the system stability in form (11). Here we have proven that inequality (11) is also sufficient for the Markov chain $\xi_t, t \geq 0$ to be ergodic.

Moreover, under the opposite sign inequality (11) is turning into the sufficient condition for non-ergodicity of the Markov chain. □

3.3. Stationary distribution and performance measures

Let us denote by $\mathbf{p}_i, i \geq 0$, vectors of the stationary probabilities of the Markov chain $\xi_t, t \geq 0$, corresponding to state i of the countable component. These vectors are of order $W[1 + M_2K_2 + (M_1 + M_1M_2K_2)K_1]$. To find the vectors $\mathbf{p}_i, i \geq 0$, we use a special algorithm for calculating the stationary distribution of asymptotically quasi-Toeplitz Markov chains presented in [32].

Algorithm 1. *(1) Calculate matrix G as the minimal non-negative solution of matrix equation $G = Y(G)$.*

(2) Calculate matrices $G_{i_0-1}, G_{i_0-2}, \ldots, G_0$ using the back recursion equation

$$G_i = (-Q_{i+1,i+1} - Q_{i+1,i+2}G_{i+1})^{-1}Q_{i+1,i},$$

$i = i_0 - 1, i_0 - 2, \ldots, 0$, with the boundary condition $G_i = G, i \geq i_0$, where i_0 is a non-negative integer such that the inequality $||G_{i_0} - G|| < \varepsilon$ holds for some positive ε (calculation accuracy).

(3) Calculate matrices $\bar{Q}_{i,i} = Q_{i,i} + Q_{i,i+1}G_i$, $\bar{Q}_{i,i+1} = Q_{i,i+1}, i \geq 0$, where $G_i = G, i \geq i_0$.

(4) Calculate matrices $F_i, i \geq 1$ using the recurrent formula $F_0 = I, F_i = F_{i-1}\bar{Q}_{i-1,i}(-\bar{Q}_{i,i})^{-1}, i \geq 1$.

(5) Calculate vector $\mathbf{p}_0$ as the unique solution to system $\mathbf{p}_0(-\bar{Q}_{0,0}) = 0$, $\mathbf{p}_0 \sum_{i=0}^{\infty} F_i\mathbf{e} = 1$.

(6) Calculate vectors $\mathbf{p}_i$ as $\mathbf{p}_i = \mathbf{p}_0F_i, i \geq 1$.

To calculate the vectors $\mathbf{p}_i, i \geq 0$, we need to truncate the number of vectors to calculate (step 5). Truncation level can be $\max\{i_0, N^*\}$, where N^* is chosen such that $|\mathbf{p}_{N^*} - \mathbf{p}_{N^*-1}| < \varepsilon$.

Having calculated the stationary distribution $\mathbf{p}_i, i \geq 0$, we can obtain a number of performance measures of the system under consideration and apply Little's Law to obtain the mean sojourn time in the whole system.

- Stationary distribution of the number of customers in the orbit $p_i = \mathbf{p}_i\mathbf{e}, i \geq 0$.
- Average number of customers in the orbit $L = \sum_{i=1}^{\infty} i\mathbf{p}_i\mathbf{e}$.

- The stationary distribution of the number of customers at the first station or at the second station can be calculated from the joint distribution $q_i(j,n), i \geq 0, j = \overline{0,K_1}, n = \overline{0,K_2}$ of the number of customers in the orbit at the first and second stations. Here, $q_i(j,n)$ is a probability that there are i customers in the orbit, j customers at the first station, and n customers at the second station. The expressions for probabilities $q_i(0,0), q_i(j,0), q_i(0,n), q_i(j,n)$ are calculated by using the following formulas:

$$q_i(0,0) = \mathbf{p}_i \begin{pmatrix} \mathbf{e}_W \\ \mathbf{0}^T_{W(M_2K_2+RK_1)} \end{pmatrix}, i \geq 0,$$

$$q_i(0,n) = \mathbf{p}_i \begin{pmatrix} \mathbf{0}^T_{W[1+M_2(n-1)]} \\ \mathbf{e}_{WM_2} \\ \mathbf{0}^T_{W[M_2(K_2-n)+RK_1]} \end{pmatrix}, i \geq 0, n = \overline{1,K_2},$$

$$q_i(j,0) = \mathbf{p}_i \begin{pmatrix} \mathbf{0}^T_{W[1+M_2K_2+R(j-1)]} \\ \mathbf{e}_{WM_1} \\ \mathbf{0}^T_{W[M_1M_2K_2+R(K_1-j)]} \end{pmatrix}, i \geq 0, j = \overline{1,K_1},$$

$$q_i(j,n) = \mathbf{p}_i \begin{pmatrix} \mathbf{0}^T_{W[1+M_2K_2+R(j-1)+M_1(1+M_2(n-1))]} \\ \mathbf{e}_{WM_1M_2} \\ \mathbf{0}^T_{W[M_1M_2(K_2-n)+R(K_1-j)]} \end{pmatrix}, i \geq 0, j = \overline{1,K_1}, n = \overline{1,K_2}.$$

- Stationary distribution of the number of customers at the first station

$$q_j^{(1)} = \sum_{i=0}^{\infty} p_i \sum_{n=0}^{K_2} q_i(j,n), \ j = \overline{0,K_1}.$$

- Average number of customers at the first station

$$L_1 = \sum_{j=1}^{K_1} j q_j^{(1)}.$$

- Stationary distribution of the number of customers at the second station

$$q_n^{(2)} = \sum_{i=0}^{\infty} p_i \sum_{j=0}^{K_1} q_i(j,n), \ n = \overline{0,K_2}.$$

- Average number of customers at the second station

$$L_2 = \sum_{n=1}^{K_2} n q_n^{(2)}.$$

- Average sojourn time of customers in the system, by Little's theorem: $W = (L + L_1 + L_2)\lambda^{-1}$.

4. Performance evaluation of tandem queueing systems with a common retrial orbit and arbitrary number of single server stations using machine learning methods

In practical applications involving computer networks, such as those discussed in [25], it is essential to consider systems with more than two stations. When dealing with multiple stations, mathematical analysis becomes challenging, necessitating the use of Monte Carlo discrete event simulation to model the queueing process. The accuracy of the simulation model is verified using numerical methods for the case of two stations. Depending on the parameters of the queuing system and the required number of samples, the simulation process can take a considerable amount of time to complete. In this section, we investigate the use of machine learning methods to predict the mean sojourn time of a tandem queueing system with a common orbit based on its parameters. Previously, a limited attempt at using machine learning methods for systems with Poisson arrivals and exponentially distributed service times had been studied in [34].

4.1. Generation of synthetic dataset

We have developed a simulation program for a tandem queuing system with a common orbit, utilizing Markov arrival processes (MAP) for input arrivals and phase-type (PH) distributed service times. Unlike the theoretical system described above, our simulated system can accommodate an arbitrary number of stations while incorporating a finite number of places in the orbit and a constant retrial policy. This setup closely mimics the retrial processes found in real computer networks. The simulation program requires the following input parameters to characterize the queuing system:

- Matrices D_0, D_1 represent MAP arrivals,
- Number N of stations,
- Matrices $S_i, \boldsymbol{\beta}_i$ represent PH service times at i-th station, $i = \overline{0, N}$,
- Buffer size K_i at i-th station, $i = \overline{0, N}$,
- Orbit retrial rate α,
- Number K_{orbit} of places at orbit

and outputs various performance characteristics of the queueing system, such as mean sojourn time, average packet loss rate, average number of packets in the system, etc.

In order to train machine learning models, we need a dataset with a large number of data points $(\boldsymbol{X}, \boldsymbol{y})$, where $\boldsymbol{X}$ corresponds to the input parameters of the simulation program and $\boldsymbol{y}$ is the mean sojourn time. This dataset is obtained by performing simulation on various randomly generated input parameters:

- Number N of stations – uniform discrete random variable on $[2, 15]$;
- Size of matrices D_0, D_1 – uniform discrete random variable on $[1, 16]$;
- Size of matrices $S_i, \boldsymbol{\beta}_i$ – uniform discrete random variable on $[1, 16]$;
- Buffer size K_i – uniform discrete random variable on $[0, 32]$;
- Number of places at orbit, K_{orbit} – uniform discrete random variable on $[0, 512]$.

For non-diagonal entries of stochastic matrices D_0, D_1, S_i (i.e. transition rates) and orbit retrial rate α, to make the dataset more varied, we use the following two ways:

- Method 1. The non-diagonal entries of matrices D_0, and D_1 are generated as uniform continuous random variables on the range $[1, 1024]$. The non-diagonal entries of matrices S_i, the transition rates $\boldsymbol{S}_i^{(0)}$ into absorption, and the orbit retrial rate α are generated as uniform continuous random variables on the range $[2, 2048]$;

- Method 2. The non-diagonal entries of matrices D_0, and D_1 are generated as e^p; the non-diagonal entries of matrices S_i, the transition rates $\boldsymbol{S}_i^{(0)}$ into absorption, and the orbit retrial rate α are generated as $2e^p$, where p is a uniform continuous random variable on the range $[-2.5, 2.5]$

Then, we regularize the input parameters, by dividing all matrices D_0, D_1, S_i, and orbit retrial rate α by a factor equal to λ (the average rate of arrivals). This step ensures all inputs will have the same arrival rate equal to 1. Note that by multiplying all transition rates by a common factor m, all non-temporal characteristics of the queueing process remain the same, only the temporal characteristics (such as mean sojourn time) are changed by a factor $\frac{1}{m}$. Therefore, there is no downside to this regularization step; and the upside is that the machine learning algorithms will work more effectively.

We then collect and divide the simulation results data into the following datasets:

- **Dataset300k**: Dataset of 200000 data points $(\boldsymbol{X}, \boldsymbol{y})$ generated by method 1 and 100000 data points generated by method 2 for training purpose;
- **Test60k**: A dataset of 60000 data points $(\boldsymbol{X}, \boldsymbol{y})$ generated by method 1 for testing purpose;
- **Sample**: A small non-random dataset of 32 data points $(\boldsymbol{X}, \boldsymbol{y})$, all have Poisson arrivals ($D_0 = \begin{bmatrix}-1\end{bmatrix}$, $D_1 = \begin{bmatrix}1\end{bmatrix}$), exponentially distributed service times ($S_i = \begin{bmatrix}-1\end{bmatrix}$, $\boldsymbol{\beta}_i = \begin{bmatrix}1\end{bmatrix}$), and orbit retrial rate $\alpha = 1$, for the purpose of testing the generalization ability of machine learning models on unseen data. In theory, the data points in this set can be randomly generated by the above procedures; however, the chance is improbable in practice.

The final step of data preparation is dimension reduction. As matrices are high-dimensional data, using them directly in training may lead to ineffective learning results. Therefore, we employ the approximation procedure described in [35] to reduce the matrices D_0, and D_1 into 5-dimension tuples $(m_1, m_2, m_3, l_1, l_2)$ and matrices $S_i, \boldsymbol{\beta}_i$ into 3-dimension tuples $(m_1^{(i)}, m_2^{(i)}, m_3^{(i)})$.

Where m_k is the k-th moment of MAP

$$m_k = k!\boldsymbol{\pi}(-D_0)^{-k}\mathbf{e},\ k \geq 1.$$

$\boldsymbol{\pi}$ is the solution of the system

$$\begin{cases}\boldsymbol{\pi}(-D_0)^{-1}D_1 = \boldsymbol{\pi},\\ \boldsymbol{\pi}\mathbf{e} = 1.\end{cases}$$

l_k is the k-th lags of MAP

$$l_k = \frac{\lambda^2\boldsymbol{\pi}(-D_0)^{-1}[(-D_0)^{-1}D_1]^k(-D_0)^{-1}\mathbf{e} - 1}{\lambda^2\boldsymbol{\pi}(-D_0)^{-2}\mathbf{e} - 1},\ k \geq 1.$$

$m_k^{(i)}$ is the k-th moment of PH distribution at i-th station

$$m_k^{(i)} = k!\boldsymbol{\beta}_i(-S)^{-k}\mathbf{e},\ k \geq 1.$$

We perform training with three types of machine learning methods: gradient boosting, random forest and artificial neural network. We do not experiment with support vector machine (SVM), which is also a powerful classical machine learning method [36], as the computational complexity of the SVM method can be from quadratic to cubic in relation to the number of data samples, and hence it is impractical for our case. Meanwhile, the training time complexity of gradient boosting and random forest methods is only $O(n\ \log(n))$, and of neural network is $O(n)$, where n is the number of data samples.

4.2. Gradient boosting

Boosting is a technique in machine learning, where an ensemble of weak prediction models are used to train stronger models (hence the name boosting) [37]. Gradient boosting is a machine learning method that is based on boosting technique and has applications in many fields. In this paper, we use the machine learning programming library Scikit-learn to train gradient boosting models for the problem.

As the experiments in [34] had shown that training models for prediction of sojourn time with mean squared error (MSE) loss function is not as effective as training with mean squared log error (MSLE), as sojourn times can have a wide range of values, from very small to very large. Therefore, in our experiments, we train two types of models, one that predicts the mean sojourn time $\boldsymbol{y}$ of the queueing system and one that predicts the natural logarithm of the mean sojourn time $\ln(\boldsymbol{y})$. Using the MSE criterion to train the second type has the same effect as using the MSLE criterion:

$$\mathrm{MSE}(\hat{\boldsymbol{y}}, \ln(\boldsymbol{y})) = \frac{1}{N}\sum_{j=1}^{N}(\hat{\boldsymbol{y}}_j - \ln(\boldsymbol{y}_j))^2 = \frac{1}{N}\sum_{j=1}^{N}(\ln(e^{\hat{\boldsymbol{y}}_j}) - \ln(\boldsymbol{y}_j))^2,$$

$$\mathrm{MLSE}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \frac{1}{N}\sum_{j=1}^{N}(\ln(\hat{\boldsymbol{y}}_j + 1) - \ln(\boldsymbol{y}_j + 1))^2.$$

Because the sojourn time is strictly positive, its logarithm is always well defined. As all machine learning programming libraries support the MSE loss function, but not every one of them has MSLE (for example, the gradient boosting implementation in Scikit-learn), training the model to predict the logarithm of the target value is a valid alternative.

Correspondingly, for the evaluation of trained models, we use the absolute value of the log accuracy ratio as the evaluation score: Score $= |\ln(\hat{\boldsymbol{y}}) - \ln(\boldsymbol{y})| = |\ln \frac{\hat{\boldsymbol{y}}}{\boldsymbol{y}}|$.

We train the gradient boosting models with default parameters, a learning rate = 0.01, least squared criterion, and 10000 training iterations. From the evaluation results in Table 1, we can see that the model, trained to predict the logarithm of mean sojourn time $\ln(\boldsymbol{y})$, works significantly better in all cases.

Table 1. Evaluation results of Gradient Boosting models.

Dataset	Training without Log-transform target		Training with Log-transform target	
	Mean Score	σ	Mean Score	σ
Dataset300k	0.5988	1.8435	0.0790	0.1351
Test60k	0.5659	1.8155	0.0900	0.2021
Sample	5.1667	5.2363	0.1807	0.1241

In Figure 2, we show the evaluation scores of gradient boosting model, trained with the logarithm transform of the target value, during the training process. We can see that the training process is stable, with the gradual decreasing of the error rate on both the training set (Dataset300k) and the test set (Test60k). For the sample set, due to its higher difference with the training set, the decreasing of the error rate is slower, but overall, there is no overfitting phenomenon (i.e., the error rate on the training set is decreasing, but the error rate on the test set is increasing).

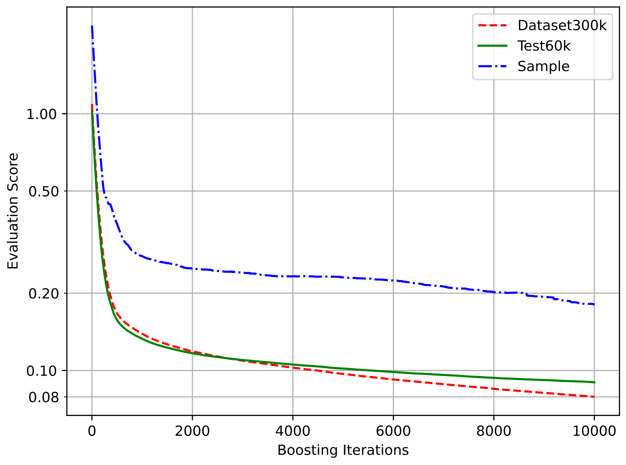

Figure 2. Evaluation results of Gradient Boosting model trained with Logarithm transform of target value during training process.

4.3. Random Forest

Random Forest is a popular method that belongs to the class of ensemble machine learning techniques. By combining the results of multiple randomized decision trees, the random forest method can achieve strong performance and reduce the tendency of overfitting (i.e., performing well on seen data but poorly on unseen data) that is often present in single decision trees.

Similar to gradient boosting, we will use the implementation available in the Scikit-learn library to train models using default parameters, with a learning rate of 0.01, the least squares criterion, and various numbers of estimators (or trees). The evaluation results are presented in Tables 2 and 3.

Table 2. Evaluation results of Random Forest models trained without Logarithm transform of target value.

Dataset	Models trained without Log-transform							
	Number of estimators							
	100		200		300		400	
	Score	σ	Score	σ	Score	σ	Score	σ
Dataset300k	0.0551	0.1603	0.0551	0.1591	0.0551	0.1587	0.0551	0.1581
Test60k	0.1176	0.2693	0.1179	0.2695	0.1181	0.2703	0.1177	0.2688
Sample	0.2696	0.2660	0.2675	0.2918	0.2768	0.2618	0.2623	0.2675
	500		600		700		800	
	Score	σ	Score	σ	Score	σ	Score	σ
Dataset300k	0.0551	0.1580	0.0551	0.1582	0.0552	0.1585	0.0551	0.1579
Test60k	0.1180	0.2694	0.1179	0.2696	0.1179	0.2694	0.1177	0.2686
Sample	0.2693	0.2586	0.2660	0.2563	0.2672	0.2579	0.2731	0.2693

Table 3. Evaluation results of Random Forest models with Logarithm transform of target value.

Dataset	Models trained with Log-transform							
	Number of estimators							
	100		200		300		400	
	Score	σ	Score	σ	Score	σ	Score	σ
Dataset300k	0.0467	0.0914	0.0460	0.0903	0.0458	0.0897	0.0457	0.0894
Test60k	0.1097	0.2357	0.1090	0.2348	0.1089	0.2344	0.1088	0.2349
Sample	0.2207	0.1823	0.2274	0.1766	0.2120	0.1790	0.2249	0.1834
	500		600		700		800	
	Score	σ	Score	σ	Score	σ	Score	σ
Dataset300k	0.0456	0.0893	0.0456	0.0891	0.0455	0.0892	0.0455	0.0891
Test60k	0.1085	0.2343	0.1086	0.2348	0.1084	0.2343	0.1085	0.2343
Sample	0.2259	0.1846	0.2261	0.1839	0.2287	0.1917	0.2193	0.1807

As shown in Tables 2 and 3, the models trained using the logarithmic transformation of the target value perform better with random forest, although the improvement is not as significant as that seen with gradient boosting. Additionally, increasing the number of estimators-which is akin to increasing the number of training iterations in gradient boosting-does not enhance the performance of the random forest models. Therefore, when training random forest models, it is advisable not to set the number of estimators too high, as this can result in very large model sizes (for example, 20 GB with 800 estimators) and consequently longer prediction times. Furthermore, a comparison of the results in Table 1 with those in Table 3 indicates that random forest exhibits a higher tendency for overfitting, as evidenced by its superior accuracy on the training set compared to gradient boosting ($\sim$ 4.6% vs. $\sim$ 7.9%), but worse on unseen datasets (Test60k and Sample). One of the downsides of using a random forest model is its significantly large size. For instance, with the default number of estimators set to 100, the model can reach a size of 2.6 GB. This large size is primarily due to the default parameters in the Scikit-learn library, which do not impose any limits on the depth of the decision trees (parameter maxDepth = 0). As a result, when working with a large training dataset, the decision trees grow extensively, leading to a considerable increase in the overall model size. To mitigate this issue and reduce the model's size, we can set the parameter maxDepth $>$ 0, at the cost of lower accuracy (Table 4).

Table 4. Random Forest models trained with Logarithm transform target value, number of estimators = 100, varying maxDepth.

Maximum Tree Depth	Dataset						Model size
	Dataset300k		Test60k		Sample		
	Score	σ	Score	σ	Score	σ	
4	0.5205	0.5744	0.4781	0.5450	0.6057	0.6072	243 KB
8	0.2601	0.3397	0.2419	0.3417	0.4361	0.3284	3.49 MB
16	0.1153	0.1429	0.1399	0.2500	0.2657	0.2069	209 MB
32	0.0476	0.0905	0.1098	0.2358	0.2028	0.1580	2.34 GB
64	0.0467	0.0915	0.1094	0.2350	0.2234	0.1580	2.60 GB
0 (Unlimited)	0.0467	0.0914	0.1097	0.2357	0.2207	0.1823	2.60 GB

4.4. Neural Network

Artificial neural networks (ANNs) are currently the most widely used machine learning method, applicable across various fields. Although ANNs are one of the oldest machine learning techniques, their practical training on large datasets has only become feasible in recent decades. This is largely due to advancements in computer hardware that enhance matrix computation capabilities. As a result, deep neural networks with a high number of parameters can now be trained effectively, allowing them to achieve superior predictive performance compared to other methods.

For training our neural network models, we utilize PyTorch, a library that is more specialized for developing neural network architectures, rather than using Scikit-learn as we did with gradient boosting and random forest. Our network is designed with three hidden layers, each consisting of 1,024 neurons (as shown in Figure 3). Based on previous experiments and findings from [34], which indicated that models using the logarithmic transformation of the target variable produced better predictions, we will only train one type of model this time (i.e., using the logarithm of the target value), instead of two types as in previous experiments.

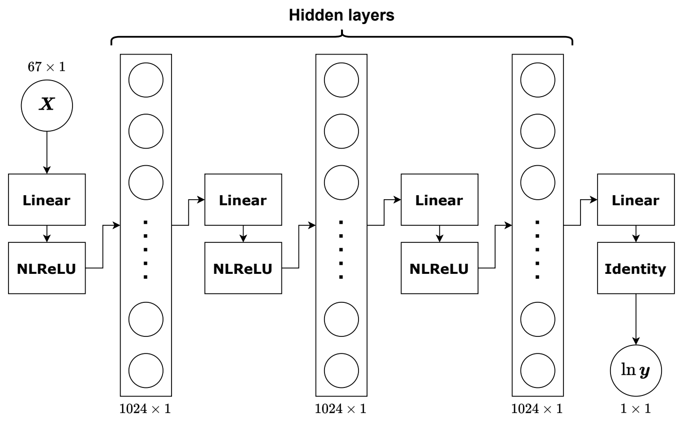

Figure 3. Neural Network model structure.

Initially, we used the standard ReLU activation function between each of the hidden layers. However, we encountered a problem during training: the loss value became excessively high due to gradient exploding, rendering the model untrainable. A straightforward solution to this issue is to change the activation function to a logarithmic variant of ReLU [38]:

$$\begin{aligned} \text{NLReLU}(x) &= \ln(\text{ReLU}(x) + 1) \\ &= \ln(\max(0, x) + 1) \end{aligned}$$

We train the neural network models using the AdamW optimizer and the mean squared rrror (MSE) loss function, with a constant learning rate of 10^{-4} and a batch size of 1024, over 10000 epochs. Throughout the training process, the loss value decreases steadily without any sudden spikes (see Figure 4). We save the model's state at regular intervals during training and then perform validation on both the Test60k and Sample datasets (refer to Figure 5).

The validation results indicate that the network generalizes well on the Test60k dataset. However, the validation scores on the sample dataset exhibit instability during training, although there are instances where the model achieves good predictive results. In Table 5, we present specific validation values from several training epochs.

From Table 5 and Figure 5, it is evident that the model at epoch 6912 is sufficiently robust to be considered as the final result, rather than the latest epoch, as further training beyond epoch 6912 does not yield improvements. Additionally, the results demonstrate that the neural network has a stronger generalization ability compared to gradient boosting and random forest, both of which fail to achieve an error rate of less than 10% on the sample dataset.

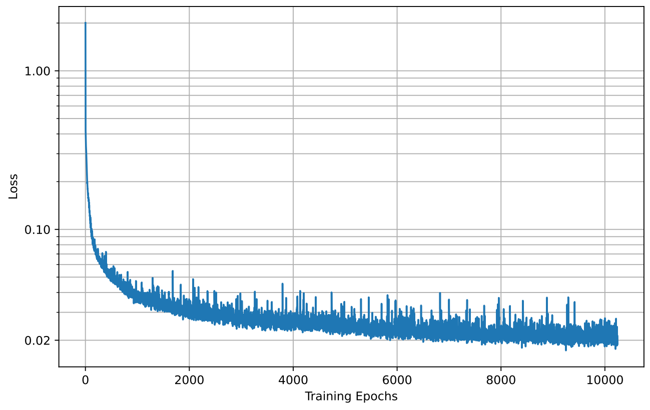

Figure 4. Neural Network training loss.

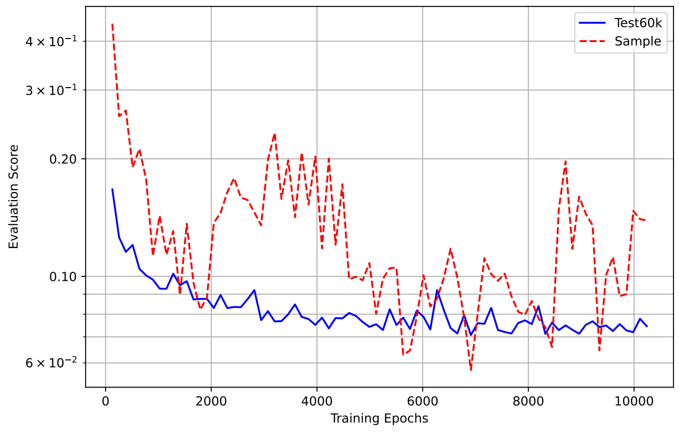

Figure 5. Evaluation results of Neural Network model at certain training epochs

Table 5. Evaluation results of Neural Network model at certain training epochs.

Dataset	Epoch									
	5670		6912		8448		9344		10112	
	Score	σ	Score	σ	Score	σ	Score	σ	Score	σ
Test60k	0.0733	0.2068	0.0707	0.1950	0.0761	0.2048	0.0741	0.1986	0.0779	0.1950
Sample	0.0646	0.0936	0.0576	0.1005	0.0659	0.1317	0.0646	0.0716	0.1402	0.0955

4.5. *Case of two stations system*

For comparison purposes, Tables 6 and 7 present the average sojourn time calculated using various methods for a simple system with two stations. The system features an input arrival process described by a Markov arrival process (MAP) (D_0, D_1), service time at the first station modeled as a phase-type (PH) distribution $(S_1, \boldsymbol{\beta}_1)$, a buffer size at the first station denoted as K_1, service time at the second station also modeled as a PH distribution $(S_2, \boldsymbol{\beta}_2)$, a buffer size at the second station represented as K_2, and a fixed retrial rate α. From the ten numerical examples provided, it is evident that machine learning methods frequently perform well in predicting the mean sojourn time for the two-station system with an infinite-size orbit (where the orbit size is sufficiently large to function similarly to an infinite one).

However, there are instances where the prediction errors from all three machine learning methods are notably high, such as in the third row of Table 6. Additionally, we observe that the random forest model often produces considerably larger prediction errors compared to the other methods, as shown in Table 7. In contrast, the neural network and gradient boosting models tend to exhibit fewer significant errors than the random forest model.

To investigate this further, we examined 1,000 examples, the results of which are summarized in Table 8. Overall, the neural network model outperformed the others. While the random forest model resulted in more predictions with small errors ($\leq 10\%$) than the gradient boosting model, it also had a higher number of samples with unacceptable errors ($\geq 90\%$). Interestingly, the gradient boosting model had the fewest samples with small errors, as well as the fewest samples with extremely high errors.

Table 6. Numerical results of various methods in the case of system with two stations.

D_0, D_1	$S_1, \boldsymbol{\beta}_1$	$S_2, \boldsymbol{\beta}_2$	α	K_1	K_2	Analytical	Simulation	Neural Network	Grad. Boosting	Rand. Forest
$\begin{bmatrix}-1.42 & 0.11\\ 6.66 & -13.0\end{bmatrix}$ $\begin{bmatrix}0.56 & 0.75\\ 1.70 & 4.64\end{bmatrix}$	$\begin{bmatrix}-18.2 & 9.87\\ 2.07 & -4.26\end{bmatrix}$ $\begin{bmatrix}0.532 & 0.468\end{bmatrix}$	$\begin{bmatrix}-4.30 & 3.16\\ 1.40 & -12.7\end{bmatrix}$ $\begin{bmatrix}0.528 & 0.472\end{bmatrix}$	5.37	29	7	1.2728	1.2643	1.2115	1.2595	1.1379
$\begin{bmatrix}-2.80 & 2.31\\ 1.02 & -1.42\end{bmatrix}$ $\begin{bmatrix}0.11 & 0.38\\ 0.27 & 0.13\end{bmatrix}$	$\begin{bmatrix}-19.6 & 2.07\\ 0.229 & -1.42\end{bmatrix}$ $\begin{bmatrix}0.797 & 0.203\end{bmatrix}$	$\begin{bmatrix}-2.64 & 1.52\\ 1.65 & -13.7\end{bmatrix}$ $\begin{bmatrix}0.286 & 0.714\end{bmatrix}$	9.77	26	13	0.5908	0.5852	0.5773	0.8313	0.6633
$\begin{bmatrix}-3.57 & 1.17\\ 0.12 & -0.43\end{bmatrix}$ $\begin{bmatrix}0.44 & 1.96\\ 0.20 & 0.11\end{bmatrix}$	$\begin{bmatrix}-19.62 & 0.45\\ 0.52 & -0.91\end{bmatrix}$ $\begin{bmatrix}0.238 & 0.762\end{bmatrix}$	$\begin{bmatrix}-11.2 & 10.9\\ 0.96 & -1.99\end{bmatrix}$ $\begin{bmatrix}0.745 & 0.255\end{bmatrix}$	16.70	16	3	5.6972	5.7862	6.5020	6.5473	7.4541
$\begin{bmatrix}-12.36 & 9.47\\ 0.53 & -1.80\end{bmatrix}$ $\begin{bmatrix}0.17 & 2.72\\ 0.80 & 0.46\end{bmatrix}$	$\begin{bmatrix}-7.39 & 0.38\\ 0.54 & -1.38\end{bmatrix}$ $\begin{bmatrix}0.471 & 0.529\end{bmatrix}$	$\begin{bmatrix}-7.65 & 6.21\\ 1.06 & -19.31\end{bmatrix}$ $\begin{bmatrix}0.578 & 0.422\end{bmatrix}$	3.72	17	28	2.4771	2.4712	4.0656	3.0351	3.1902
$\begin{bmatrix}-4.98 & 1.06\\ 0.16 & -1.39\end{bmatrix}$ $\begin{bmatrix}1.62 & 2.30\\ 1.02 & 0.21\end{bmatrix}$	$\begin{bmatrix}-7.01 & 6.83\\ 0.53 & -4.39\end{bmatrix}$ $\begin{bmatrix}0.095 & 0.905\end{bmatrix}$	$\begin{bmatrix}-4.83 & 1.62\\ 20.86 & -34.07\end{bmatrix}$ $\begin{bmatrix}0.870 & 0.130\end{bmatrix}$	0.48	28	17	1.3501	1.3516	1.4492	1.1897	1.3931

Table 7. Numerical results of various methods in the case of system with two stations (cont.).

$D_0; D_1$	$S_1; \beta_1$	$S_2; \beta_2$	α	K_1	K_2	Analytical	Simulation	Neural Network	Grad. Boosting	Rand. Forest
$\begin{bmatrix}-1.49 & 1.11\\ 0.37 & -2.09\end{bmatrix}$ $\begin{bmatrix}0.25 & 0.13\\ 1.03 & 0.69\end{bmatrix}$	$\begin{bmatrix}-15.01 & 0.65\\ 0.56 & -1.25\end{bmatrix}$ $\begin{bmatrix}0.336 & 0.664\end{bmatrix}$	$\begin{bmatrix}-5.17 & 2.74\\ 2.17 & -3.57\end{bmatrix}$ $\begin{bmatrix}0.512 & 0.488\end{bmatrix}$	4.58	3	9	3.1472	3.1502	3.6450	3.9505	6.7534
$\begin{bmatrix}-4.70 & 0.10\\ 0.79 & -4.28\end{bmatrix}$ $\begin{bmatrix}1.01 & 3.59\\ 1.58 & 1.91\end{bmatrix}$	$\begin{bmatrix}-9.94 & 0.47\\ 8.12 & -22.09\end{bmatrix}$ $\begin{bmatrix}0.764 & 0.236\end{bmatrix}$	$\begin{bmatrix}-16.33 & 0.17\\ 22.50 & -22.85\end{bmatrix}$ $\begin{bmatrix}0.526 & 0.474\end{bmatrix}$	0.20	8	15	0.2851	0.2850	0.5385	0.4195	0.8624
$\begin{bmatrix}-2.60 & 0.21\\ 4.95 & -6.09\end{bmatrix}$ $\begin{bmatrix}0.13 & 2.26\\ 0.62 & 0.52\end{bmatrix}$	$\begin{bmatrix}-2.95 & 2.64\\ 0.67 & -7.26\end{bmatrix}$ $\begin{bmatrix}0.769 & 0.231\end{bmatrix}$	$\begin{bmatrix}-10.73 & 10.39\\ 0.72 & -23.36\end{bmatrix}$ $\begin{bmatrix}0.869 & 0.131\end{bmatrix}$	0.54	8	27	3.3706	3.3382	3.0185	3.1438	5.6815
$\begin{bmatrix}-1.61 & 0.57\\ 2.78 & -3.38\end{bmatrix}$ $\begin{bmatrix}0.85 & 0.19\\ 0.39 & 0.21\end{bmatrix}$	$\begin{bmatrix}-30.94 & 20.60\\ 12.13 & -14.44\end{bmatrix}$ $\begin{bmatrix}0.653 & 0.347\end{bmatrix}$	$\begin{bmatrix}-10.58 & 8.67\\ 1.10 & -6.17\end{bmatrix}$ $\begin{bmatrix}0.959 & 0.041\end{bmatrix}$	2.05	32	18	0.5825	0.5831	0.5566	0.5317	0.5701
$\begin{bmatrix}-2.32 & 0.25\\ 0.14 & -2.38\end{bmatrix}$ $\begin{bmatrix}0.09 & 1.97\\ 2.00 & 0.25\end{bmatrix}$	$\begin{bmatrix}-9.52 & 5.35\\ 0.33 & -3.49\end{bmatrix}$ $\begin{bmatrix}0.598 & 0.402\end{bmatrix}$	$\begin{bmatrix}-10.56 & 0.21\\ 0.84 & -7.07\end{bmatrix}$ $\begin{bmatrix}0.816 & 0.184\end{bmatrix}$	5.36	25	27	0.9635	0.9804	0.7594	0.8355	1.4875

Table 8. Verification results on 1000 examples.

	Neural Network	Gradient Boosting	Random Forest
Number of examples with error $\leq 10\%$	532	347	474
Number of examples with error $\geq 50\%$	66	90	89
Number of examples with error $\geq 90\%$	35	25	43

4.6. Comparison of training and execution time

The primary motivation for applying machine learning methods to queueing theory problems is to enhance execution speed, albeit with some sacrifices in accuracy. For instance, generating a single sample in the datasets used in the aforementioned experiments takes an average of 4 seconds of simulation time. While this duration may seem minimal, simulating large datasets comprising hundreds of thousands or even millions of samples could take days or weeks. As illustrated in Table 9, even the slowest machine learning methods are significantly faster than traditional simulations.

However, this substantial speedup comes at a cost. Machine learning methods require more memory for loading models and incur higher computational costs during the training process. Table 10 provides a summary of model sizes and training time costs for each method. Most models, except for the neural network, were trained on an Intel Xeon E5-2680 CPU. The neural network model, on the other hand, was trained using an NVIDIA RTX 3070 Ti GPU.

Table 9. Comparison of average execution time of methods for one data point.

Method	Execution time (seconds)
Simulation	4
Gradient Boosting	2×10^{-4}
Random Forest	1×10^{-7}
Neural Network	3×10^{-5}

Table 10. Comparison of model size and training time of machine learning methods.

Method	Model size	Training iterations	Total training time
Gradient Boosting	34.5 MB	10000	143 secs
Random Forest	2.6 GB	100	306 secs
Neural Network (GPU)	24.8 MB	10000	13 hours

It is important to note that since gradient boosting and random forest are ensemble-based methods, the model size increases with the number of training iterations (i.e., the number of decision trees). In contrast, the size of the neural network model remains constant with respect to the number of training iterations; it is determined solely by the number of trainable parameters (layers and weights) in the model. As a result, we can train the neural network model for extended periods without impacting its size.

5. Conclusions

In this paper, we examined a multiphase queueing system with a finite number of stations and a common orbit. The incoming arrivals to the system are of the Markovian arrival process (MAP) type, and each station consists of a single server with a finite buffer. The service times for each server follow a phase-type (PH) distribution.

For the system with two stations and an infinite orbit, we derived the ergodicity condition, stationary distribution, and several performance indicators. In contrast, for the system with an arbitrary number of stations and a finite orbit, we employed various machine learning methods to predict the mean sojourn time of the system.

Overall, all methods demonstrated the capability to learn the problem effectively. Among these, gradient boosting and random forest-both ensemble methods based on decision trees—stand out for their ease of use, requiring minimal parameter tuning. Meanwhile, while neural networks necessitate more design choices to ensure a stable training process, they can potentially provide superior prediction performance.

Author contributions

Vladimir Vishnevsky: Conceptualization, Investigation, Formal analysis, Writing-review and editing, Supervision; Valentina Klimenok: Conceptualization, Investigation, Methodology, Formal analysis, Writing-review and editing, Supervision; Olga Semenova: Investigation, Formal analysis, Writing-original draft preparation, Writing-review and editing; Minh Cong Dang: Writing-original draft preparation, Software, Validation, Formal analysis, Writing-review and editing. All authors have read and approved the final version of the manuscript for publication.

Use of Generative-AI tools declaration

The authors declare that they have not used Artificial Intelligence (AI) tools in the creation of this article.

Conflict of interest

All authors declare no conflicts of interest in this paper.

References

1. A. Heindl, K. Mitchell, A. van de Liefvoort, Correlation bounds for second order MAPs with application to queueing network decomposition, *Perform. Eval.*, **63** (2006), 533–577. http://doi.org/10.1016/j.peva.2005.06.003
2. S. Balsamo, Queueing networks with blocking: Analysis, solution algorithms and properties, In: *Lecture Notes in Computer Science*, Berlin: Springer, **5233** (2011), 233–257. http://doi.org/10.1007/978-3-642-02742-0_11
3. B. W. Gnedenko, D. Konig, *Handbuch der bedienungstheorie*, Berlin: Akademie Verlag, 1983.
4. M. F. Neuts, A versatile Markovian point process, *J. Appl. Prob.*, **16** (1979), 764–779. http://doi.org/10.2307/3213143
5. D. M. Lucantoni, New results on the single server queue with a batch markovian arrival process, *Commun. Stat. Stoch. Models*, **7** (1991), 1–46. http://doi.org/10.1080/15326349108807174
6. S. A. Dudin, A. N. Dudin, O. S. Dudina, S. R. Chakravarthy, Analysis of a tandem queuing system with blocking and group service in the second node, *Int. J. Syst. Sci. Oper. Log.*, **10** (2023), 2235270. http://doi.org/10.1080/23302674.2023.2235270
7. Z. Lian, L. Liu, A tandem network with MAP inputs, *Oper. Res. Lett.*, **36** (2008), 189–195. http://doi.org/10.1016/j.orl.2007.04.004
8. A. Gomez-Corral, A tandem queue with blocking and Markovian arrival process, *Queueing Syst.*, **41** (2002), 343–370. http://doi.org/10.1023/A:1016235415066
9. C. Kim, A. Dudin, V. Klimenok, O. Taramin, A Tandem BMAP/G/1 $\rightarrow \bullet$/M/N/0 queue with group occupation of servers at the second station, *Math. Prob. Eng.*, **2012** (2012), 324604. http://doi.org/10.1155/2012/324604
10. C. Kim, S. Dudin, Priority tandem queueing model with admission control, *Comput. Indust. Eng.*, **61** (2011), 131–140. http://doi.org/10.1016/j.cie.2011.03.003
11. B. K. Kumar, R. Sankar, R. N. Krishnan, R. Rukmani, Performance analysis of multi-processor two-stage tandem call center retrial queues with non-reliable processors, *Methodol. Comput. Appl. Probab.*, **24** (2022), 95–142. http://doi.org/10.1007/s11009-020-09842-6

12. A. Dudin, A. Nazarov, On a tandem queue with retrials and losses and state dependent arrival, service and retrial rates, *Int. J. Oper. Res.*, **29** (2017), 170–182. http://doi.org/10.1504/IJOR.2017.083954

13. C. Kim, A. Dudin, V. Klimenok, Tandem retrial queueing system with correlated arrival flow and operation of the second station described by a Markov chain, In: *Computer Networks: 19th International Conference, CN 2012, Szczyrk, Poland*, **291** (2012), 370–382. http://doi.org/10.1007/978-3-642-31217-5_39

14. C. S. Kim, V. Klimenok, O. Taramin, A tandem retrial queueing system with two Markovian flows and reservation of channels, *Comput. Oper. Res,*, **37** (2010), 1238–1246. http://doi.org/10.1016/j.cor.2009.03.030

15. C. S. Kim, S. H. Park, A. Dudin, V. Klimenok, G. V. Tsarenkov, Investigaton of the BMAP/G/1 $\rightarrow$ •/PH/1/M tandem queue with retrials and losses, *Appl. Math. Model.*, **34** (2010), 2926–2940. http://doi.org/10.1016/j.apm.2010.01.003

16. A. N. Dudin, R. Manzo, R. Piscopo, Single server retrial queue with group admission of customers, *Comput. Oper. Res.*, **61** (2015), 89–99. https://doi.org/10.1016/j.cor.2015.03.008

17. G. Falin, J. G. C. Templeton, *Retrial queues*, Florida: CRC Press, 1997.

18. J. R. Artalejo, A. Gomez-Corral, *Retrial queueing systems*, Berlin: Springer, 2008.

19. V. Klimenok, O. Dudina, Retrial tandem queue with controllable strategy of repeated attempts, *Qual. Technol. Quant. Manag.*, **14** (2016), 74–93. http://doi.org/10.1080/16843703.2016.1189177

20. C. S. Kim, V. Klimenok, A. Dudin, Priority tandem queueing system with retrials and reservation of channels as a model of call center, *Comput. Indust. Eng.*, **96** (2016), 61–71. http://doi.org/10.1016/j.cie.2016.03.012

21. S. Dudin, A. Dudin, R. Manzo, L. Rarità, Analysis of semi-open queueing network with correlated arrival process and multi-server nodes, *Oper. Res. Forum*, **5** (2024), 99. https://doi.org/10.1007/s43069-024-00383-z

22. K. Avrachenkov, U. Yechiali, On tandem blocking queues with a common retrial queue, *Comput. Oper. Res.*, **37** (2010), 1174–1180. http://doi.org/10.1016/j.cor.2009.10.004

23. S. V. Paul, A. A. Nazarov, T. Phung-Duc, M. Morozova, Mathematical model of the tandem retrial queue M/GI/1/M/1 with a common orbit, In: *Communications in Computer and Information Science*, Berlin: Springer, **1605** (2022), 131–143. http://doi.org/10.1007/978-3-031-09331-9_11

24. A. A. Nazarov, S. V. Paul, T. Phung-Duc, M. Morozova, Analysis of tandem retrial queue with common orbit and MMPP incoming flow, In: *Lecture Notes Comput. Sci.*, Berlin: Springer, **13766** (2023), 270–283. http://doi.org/10.1007/978-3-031-23207-7_21

25. V. M. Vishnevsky, A. N. Dudin, D. V. Kozyrev, A. A. Larionov, Methods of performance evaluation of broadband wireless networks along the long transport routes, In: *Distributed Computer and Communication Networks. DCCN 2015. Communications in Computer and Information Science, Spinger*, **601** (2016), 72–85. http://doi.org/10.1007/978-3-319-30843-2_8

26. V. M. Vishnevsky, A. V. Gorbunova, Application of machine learning methods to solving problems of queuing theory, In: *International Conference on Information Technologies and Mathematical Modelling, Springer*, **1605** (2022), 304–316. http://doi.org/10.1007/978-3-031-09331-9_24

27. V. Vishnevsky, V. Klimenok, A. Solokov, A. Larionov, Performance evaluation of the priority multi-server system MMAP/PH/M/N using machine learning methods, *Mathematics*, **9** (2021), 3236. http://doi.org/10.3390/math9243236

28. V. Vishnevsky, V. Klimenok, A. Solokov, A. Larionov, Investigation of the fork-join system with markovian arrival process arrivals and phase-type service time distribution using machine learning methods, *Mathematics*, **12** (2024), 0659. http://doi.org/10.3390/math12050659

29. V. M. Vishnevsky, A. A. Larionov, A. A. Mukhtarov, A. M. Sokolov, Investigation of tandem queueing systems using machine learning methods, *Control Sci.*, **4** (2024), 10–21.

30. A. N. Dudin, V. I. Klimenok, V. M. Vishnevsky, *The theory of queuing systems with correlated flows*, Switzerland: Springer, 2020. http://doi.org/10.1007/978-3-030-32072-0

31. M. F. Neuts, *Structured stochastic matrices of M/G/1 type and their applications*, New York: Marcel Dekker, 1989.

32. V. Klimenok, A. Dudin, Multi-dimensional asymptotically quasi-Toeplitz Markov chains and their application in queueing theory, *Queueing Syst.*, **54** (2006), 245–259. http://doi.org/10.1007/s11134-006-0300-z

33. F. Gantmakher, *The matrix theory*, Moscow: Science, 1967.

34. M. C. Dang, Prediction of a multiphase queuing system performance using machine learning method, in Russian, *Trudy MFTI*, **16** (2024), 37–45.

35. V. Vishnevsky, A. Larionov, I. Roman, O. Semenova, Estimation of IEEE 802.11 DCF access performance in wireless networks with linear topology using PH service time approximations and MAP input, In: *IEEE 11th Int. Conf. on Application of Information and Communication Technologies*, 2017. http://doi.org/10.1109/ICAICT.2017.8687247

36. V. N. Vapnik, *The nature of statistical learning theory*, New York: Springer, 2000.

37. R. E. Schapire, The strength of weak learnability, *Mach. Learn.*, **5** (1990), 245–259. http://doi.org/10.1007/BF00116037

38. Y. Liu, J. Zhang, C. Gao, J. Qu, L. Ji, Natural-logarithm-rectified activation function in convolutional neural networks, In: *2019 IEEE 5th International Conference on Computer and Communications*, 2019, 2000–2008. http://doi.org/10.1109/ICCC47050.2019.9064398