

Received 27 October 2023, accepted 28 November 2023, date of publication 4 December 2023,
date of current version 13 December 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3339385

RESEARCH ARTICLE

Throughput of the Queue With Probabilistic Rejections

ANDRZEJ CHYDZINSKI 

Department of Computer Networks and Systems, Silesian University of Technology, 44-100 Gliwice, Poland
e-mail: andrzej.chydzinski@polsl.pl

ABSTRACT The queueing system with probabilistic job rejections based on the system occupancy has applications in engineering and logistics. It is also a natural extension of the most basic and commonly used FIFO queue with tail drop. In this paper, we analyze the throughput of such a system – a fundamental characteristic from a practical point of view. Specifically, we derive a formula for the number of jobs that the system processes in a time interval of arbitrary length (transient analysis), as well as a formula for the stationary throughput, i.e., the overall percentage of jobs passing successfully through the system. What is important, a general interarrival distribution is used in derivations, which enables modeling of a great variety of arrival streams. Theoretical results are accompanied by numeric calculations, in which the time-dependent and stationary throughput is calculated for different rejection probabilities, system loads, interarrival distributions, and initial system states.


INDEX TERMS Queueing system, probabilistic rejections, throughput, transient analysis, stationary analysis, computer networking.

I. INTRODUCTION

In many queueing systems used in engineering and logistics, the waiting room for the queue has limited capacity. This is especially true for all computer and electronic devices, where the limited waiting room (buffer) is a consequence of a limited physical memory for jobs or data packets waiting for processing.

In the great majority of such queueing systems, the waiting room is organized in the tail-drop manner. Namely, the queue can build up until the waiting room is fully occupied. When this takes place, any new incoming job is rejected—it cannot join the queue and exits the system unprocessed.

This natural and widely adopted policy comes with several drawbacks. Foremost, it allows no control of the performance of the system—the rejections happen in an uncontrolled manner. As a consequence, the waiting room may have a tendency to be nearly full most of the time, resulting in prolonged waiting times for jobs. Moreover, the rejections may usually happen consecutively, which may lead to long sequences of unserved jobs [1].

The associate editor coordinating the review of this manuscript and approving it for publication was Tawfik Al-Hadhrani .

The tail-drop policy is particularly detrimental in computer networking, where all the mentioned negative phenomena have been observed in queues of packets at routers/switches. Furthermore, some other subtle negative effects occur in networking due to the tail-drop algorithm, including the synchronization of periods of TCP flows passing through the router and highly unequal bandwidth assignment between different flows [2], [3], [4].

All the aforementioned negative phenomena can be circumvented by generalizing the tail-drop approach. Loosely speaking, jobs/packets ought to be rejected rather gradually, when the queue is building up. To acquire that, jobs can be rejected probabilistically, with the probability increasing as the number of jobs in the system grows. The fundamental role in this approach is occupied by function $d(i)$, which relates the job rejection probability to the current system occupancy, i . By manipulating $d(i)$, we can control nearly all queuing characteristics, including the number of rejected and processed jobs, the system occupancy, and its response time.

In networking literature, many forms of function $d(i)$ were postulated and tested in discrete-event simulations. These include the linear function [5], negative exponential [6], quadratic [7], cubic [8], [9] and composed from different

functions (linear, cubic, logarithmic, beta) on different intervals [10], [11], [12], [13].

Furthermore, the rejection mechanism based of function $d(i)$ was recently realized in a prototype of a networking device and examined extensively in an actual, operating network [14]. During these tests, lasting over a month, a large number of measurements of various performance characteristics were collected [14]. The results imply clearly that the $d(i)$ mechanism is very beneficial in TCP/IP networking, if compared with the FIFO queueing, commonly used now.

Mathematical insight in the performance of the $d(i)$ mechanism is the natural continuation of the study on the mechanism.

In this paper, the throughput of such mechanism is studied. Obviously, throughput is a fundamental performance characteristic in terms of practicality – it represents the number of jobs/packets that the mechanism can process in a given time.

The most important contribution of the paper consists of three formulae for the throughput and throughput-related characteristics of the system, specifically:

- a formula for the number of jobs that the system completes in a time interval of length t (transient case),
- a formula for the intensity of the output process at time t (transient case),
- a formula for the stationary throughput, i.e., the ratio of jobs that successfully traverse the system over a long period.

All these formulae are generic with respect to both function $d(i)$ and the job interarrival distribution, $G(t)$. The fact that both $d(i)$ and $G(t)$ can have arbitrary forms in the considered model is essential for its applicability. As already mentioned, several different forms of $d(i)$ have been suggested in the literature. The formulae proven here can be applied to all of them. This will be demonstrated in the examples. Similarly, the interarrival distribution may vary significantly in different queueing systems. It is known that some parameters of this distribution (e.g., the variance) may profoundly influence the performance of the system. Once again, the formulae proven here can be used for an arbitrary $G(t)$.

From the theoretical perspective, the approach founded upon function $d(i)$ is a generalized tail-drop approach. Specifically, if $d(i) = 0$ for $i < K$ and $d(i) = 1$ for $i \geq K$, then we have, in fact, tail dropping with a waiting room of capacity K . However, all the previously noted benefits occur when $d(i)$ does not have such a trivial form.

Mathematical results are illustrated here with numeric calculations. Specifically, the throughput of the system is calculated for various functions $d(i)$, system loads, initial system occupancies, and the standard deviations of the interarrival distribution. Using these results, we can study the influence of each of these factors on the throughput.

Finally, it can be pointed out that computer and networking systems are not the only possible applications of the queue

with rejections based on the system occupancy. Some other applications include call centers and everyday life queues [15], and vehicular traffic [16].

The article continues with the following organization. In Section II, prior research on related subjects is reviewed. In Section III, the queueing model with the $d(i)$ mechanism is described, along with necessary notations used in the remaining sections. Then, in Section IV, the actual analysis of the system throughput is conducted. The key outcome of this section is Theorem 1 on the number of jobs that the system completes in a time interval of length t . Building upon this theorem, the intensity of the output process at time t is derived in formula (23), and the stationary throughput is presented in formula (25). In Section V, results of numeric calculations are shown and discussed. These results encompass both stationary and transient throughput, obtained for numerous parameterizations of the system. Finally, in Section VI, some closing comments and a proposal for future work are offered.

II. PRIOR RESEARCH

Based on the author's knowledge, the results shown here are new.

To begin with, older studies on the $d(i)$ mechanism are based rather on simulations than the mathematical modelling, like herein. This, in particular, concerns the already referenced studies [6], [7], [8], [9], [10], [11], [12], [13].

Over time, more articles exploiting mathematical models of the system were published, e.g. [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], and [25]. The essential difference between these models and the model considered herein is that the arrival stream is Poisson or Markovian in all articles [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], and [25]. Herein, we model the arrivals using the general renewal stream, in which the interarrival distribution can have an arbitrary form. This, obviously, creates broad modelling capabilities. For instance, we can easily model particular variance, skewness and kurtosis of the interarrival distribution.

So far, there have been only three studies, in which the arrival stream is also of the general renewal type [26], [27], [28]. However, none of the papers [26], [27], and [28] deal with the throughput of the system, which is the fundamental performance characteristic, especially from the engineering point of view.

Finally, it should be mentioned that there are mechanisms proposed for IP networks in which the probability of rejection of a packet arriving to the router is a function of different factors, than the current system occupancy, e.g. [29], [30], [31], [32], [33], [34], [35], and [36]. These are, however, conceptually different mechanisms than the one studied herein.

III. MODEL

The following queueing model is considered.

Jobs arrive at the server following a general renewal stream, which means that interarrival times are random,

mutually independent, and their distribution is defined by distribution function $G(t)$. It is assumed that the mean interarrival time, m , is finite, i.e.:

$$m = \int_0^\infty (1 - G(t)) dt < \infty. \quad (1)$$

(Systems with $m = \infty$ carry little practical significance, because the mean queue size is zero in such systems).

Jobs arriving to the server when it is processing another jobs, are aggregated into a queue. The server processes jobs from this queue in the order they arrived (FIFO). The processing time of a job is random and follows an exponential distribution with parameter μ . The overall system capacity is restricted to accommodate a maximum of K jobs, which includes the processing position. If a job arrives while there are already K jobs in the system, the just-arrived job is rejected - it departs the system undone and never returns. (In networking, the arriving packet is simply deleted).

Furthermore, an arriving job can be rejected even if there are fewer than K jobs in the system, upon its arrival. This rejection takes place with probability $d(i)$, where i represents the number of jobs in the system when the new job enters.

Finally, we assume that if the system is not empty at $t = 0$, then the time origin aligns with the arrival time of a new job. This assumption does not limit the generality of the model, but simplifies some of the equations.

IV. SYSTEM THROUGHPUT

Let $C_n(t)$ be the mean number of jobs completed in $(0, t)$, assuming that the system occupancy at time origin was n .

We begin with assembling a system of integral equations for $C_n(t)$. Specifically, for $n = 1, \dots, K$ it holds:

$$\begin{aligned} C_n(t) &= \int_0^t \sum_{k=0}^{n-1} \frac{e^{-\mu z} (\mu z)^k}{k!} (1-d(n-k))(k+C_{n-k+1}(t-z))dG(z) \\ &+ \int_0^t \left(1 - \sum_{k=0}^{n-1} \frac{e^{-\mu z} (\mu z)^k}{k!}\right) (1-d(0))(n+C_1(t-z))dG(z) \\ &+ \int_0^t \sum_{k=0}^{n-1} \frac{e^{-\mu z} (\mu z)^k}{k!} d(n-k)(k+C_{n-k}(t-z))dG(z) \\ &+ \int_0^t \left(1 - \sum_{k=0}^{n-1} \frac{e^{-\mu z} (\mu z)^k}{k!}\right) d(0)(n+C_0(t-z))dG(z) \\ &+ (1-G(t)) \sum_{k=0}^{n-1} k \frac{e^{-\mu t} (\mu t)^k}{k!} \\ &+ (1-G(t))n \left(1 - \sum_{k=0}^{n-1} \frac{e^{-\mu t} (\mu t)^k}{k!}\right). \end{aligned} \quad (2)$$

System (2) is derived by conditioning on joint distribution of the initial arrival time, z , and the number of jobs processed by this time, k . Namely, the first term of (2) is associated with the event where less than n jobs are processed by z and the

job arriving at z is accepted. The second term is associated with the event where all n jobs are processed by z and the job arriving at z is accepted. Similarly, the third and the fourth terms are associated with the event where the job arriving at z is rejected. The fifth term of (2) encompasses the event where the initial arrival takes place after t and less than n jobs are processed by t . Finally, the sixth term encompasses the event where the initial arrival takes place after t , and all n jobs are processed by t .

Note also that for $n = K$ and $k = 1$, there is $C_{K+1}(t)$ summand in the first term of (2). Its value is, however, mathematically irrelevant, because it is multiplied by $1 - d(K)$, which is zero. The value of $C_{K+1}(t)$ is also irrelevant from a practical point of view, because the system occupancy will never be $K + 1$ under assumptions of the previous section. Therefore, we may assume simply, $C_{K+1}(t) = 0$.

Assume now that $n = 0$. We obtain then:

$$\begin{aligned} C_0(t) &= \int_0^t (1-d(0))C_1(t-z)dG(z) \\ &+ \int_0^t d(0)C_0(t-z)dG(z) \\ &+ (1-G(t)) \cdot 0. \end{aligned} \quad (3)$$

System (3) is derived by conditioning on the initial arrival time, z , only. The first term is associated with the event where $z < t$ and the initial arrival is accepted. Similarly, the second term is associated with the event where the initial arrival is rejected. Finally, the third term of (3) encompasses the event where the initial arrival takes place after t .

To derive $C_n(t)$ for arbitrary t , we need to solve integral equations (2) and (3). These equations, however, are intractable in their current forms in the time domain. Therefore, we will solve them in the Laplace transform domain. Such approach does not affect the applicability of the results, as there are many efficient inversion formulae available to transform the results back from the Laplace domain to the time domain.

From (2), we then have:

$$\begin{aligned} C_n^*(s) &= \sum_{k=0}^{n-1} (1-d(n-k))h_k(s) \left(\frac{k}{s} + C_{n-k+1}^*(s)\right) \\ &+ (1-d(0)) \left(g(s) - \sum_{k=0}^{n-1} h_k(s)\right) \left(\frac{n}{s} + C_1^*(s)\right) \\ &+ \sum_{k=0}^{n-1} d(n-k)h_k(s) \left(\frac{k}{s} + C_{n-k}^*(s)\right) \\ &+ d(0) \left(g(s) - \sum_{k=0}^{n-1} h_k(s)\right) \left(\frac{n}{s} + C_0^*(s)\right) \\ &+ \sum_{k=0}^{n-1} k f_k(s) \\ &+ n \left(y(s) - \sum_{k=0}^{n-1} f_k(s)\right), \quad n = 1, \dots, K, \end{aligned} \quad (4)$$

where

$$C_n^*(s) = \int_0^\infty e^{-st} C_n(t) dt, \tag{5}$$

$$h_k(s) = \frac{1}{k!} \int_0^\infty e^{-t(s+\mu)} (\mu t)^k dG(t), \tag{6}$$

$$g(s) = \int_0^\infty e^{-st} dG(t), \tag{7}$$

$$f_k(s) = \frac{1}{k!} \int_0^\infty e^{-t(s+\mu)} (\mu t)^k (1 - G(t)) t, \tag{8}$$

$$y(s) = \int_0^\infty e^{-st} (1 - G(t)) t. \tag{9}$$

Specifically, (4) has been obtained by employing the Laplace transform to (2) and making use of the Convolution Theorem, [37], which transforms two functions convoluted in the time domain into a product of their transforms. (We have four such convolutions in initial four terms of (2)).

In the similar manner we obtain:

$$C_0^*(s) = (1 - d(0))g(s)C_1^*(s) + d(0)g(s)C_0^*(s) \tag{10}$$

from (3).

We can rearrange (4) and (10) with respect to unknowns C_n^* . After that, we get from (4):

$$\begin{aligned} & d(0)a_n(s)C_0^*(s) + (1 - d(0)) a_n(s)C_1^*(s) \\ & + \sum_{k=0}^n d(k)h_{n-k}(s)C_k^*(s) \\ & + \sum_{k=0}^n (1 - d(k)) h_{n-k}(s)C_{k+1}^*(s) - C_n^*(s) = c_n(s), \\ & n = 1, \dots, K, \end{aligned} \tag{11}$$

where

$$a_n(s) = g(s) - \sum_{k=0}^n h_k(s), \tag{12}$$

$$b_n(s) = y(s) - \sum_{k=0}^n f_k(s), \tag{13}$$

$$c_n(s) = -\frac{1}{s} \sum_{k=0}^n kh_k(s) - \frac{n}{s} a_n(s) - \sum_{k=0}^n kf_k(s) - nb_n(s), \tag{14}$$

while (10) yields:

$$(1 - d(0)g(s))C_0^*(s) - (1 - d(0))g(s)C_1^*(s) = 0. \tag{15}$$

Clearly, (11) and (15) constitute now an ordered system of $K + 1$ linear equations with $K + 1$ variables $C_0^*(s), \dots, C_K^*(s)$. This system can be easily rewritten in the matrix form, and its solution can be presented using an inverted square matrix of size $(K + 1) \times (K + 1)$.

In this way, we get the following theorem.

Theorem 1: The transform of the number of jobs completed in $(0, t)$ has the following form:

$$C^*(s) = W^{-1}(s)c(s), \tag{16}$$

where

$$C^*(s) = [C_0^*(s), \dots, C_K^*(s)]^T, \tag{17}$$

$$c(s) = [0, c_1(s), \dots, c_K(s)]^T, \tag{18}$$

$$W(s) = [w_{ij}(s)]_{i=0, \dots, K; j=0, \dots, K}, \tag{19}$$

$$w_{ij}(s) = \begin{cases} 1 - d(0)g(s), & \text{if } i = 0, j = 0, \\ -(1 - d(0))g(s), & \text{if } i = 0, j = 1, \\ (1 - d(0))a_0(s) + d(1)h_0(s) - 1, & \text{if } i = 1, j = 1, \\ d(0)a_{i-1}(s), & \text{if } i \geq 1, j = 0, \\ (1 - d(0))a_{i-1}(s) + d(1)h_{i-1}(s), & \text{if } i \geq 2, j = 1, \\ (1 - d(i))h_0(s), & \text{if } i \geq 1, j = i + 1, \\ d(i)h_0(s) + (1 - d(i-1))h_1(s) - 1, & \text{if } i \geq 2, i = j, \\ d(j)h_{i-j}(s) + (1 - d(j-1))h_{i-j+1}(s), & \text{if } j \geq 2, i \geq j + 1, \\ 0, & \text{otherwise,} \end{cases} \tag{20}$$

$h_k(s)$ is given in (6), $g(s)$ in (7), $a_k(s)$ in (12), while $c_k(s)$ in (14).

Another interesting transient performance characteristic, $T_n(t)$, is defined as follows:

$$T_n(t) = m \frac{dC_n(t)}{dt}, \tag{21}$$

where m is the mean interarrival time. In other words, $T_n(t)$ is the intensity of the output process at the time t , which equals $\frac{dC_n(t)}{dt}$, normalized to the mean arrival rate $1/m$, given the initial occupancy was n . We have also:

$$T_n^*(s) = \int_0^\infty e^{-st} T_n(t) dt. \tag{22}$$

Having Theorem 1, it is an easy task to obtain $T_n^*(s)$. Specifically, using Derivative Theorem for the Laplace transform, [37], yields:

$$T_n^*(s) = m s C_n^*(s), \tag{23}$$

where $C_n^*(s)$ can be computed by means of Theorem 1.

Finally, we can also derive the stationary throughput, T , defined as the ratio of jobs that traverse successfully the system over a long period. By definition, we have:

$$T = \lim_{t \rightarrow \infty} T_0(t). \tag{24}$$

Note that $\lim_{t \rightarrow \infty} T_n(t)$ with any other n , instead of 0, can be employed in (24). This is thanks to the fact that the stationary characteristic does not depend on the initial system state and the limit in (24) is unaltered for every n .

Using Terminal-Value Theorem for the Laplace transform, [37], we have:

$$T = \lim_{s \rightarrow 0} s T_0^*(s) = m \lim_{s \rightarrow 0} s^2 C_0^*(s), \tag{25}$$

where $C_0^*(s)$ can be computed by means of Theorem 1.

It might prompt one to ask, if it is possible to present Theorem 1 in an explicit form, rather than using a matrix

inversion. Such explicit solutions are often presented for classic models, with no $d(i)$ function. Regrettably, this might be very hard or impossible when $d(i)$ is employed. The chief reason is that the model is not vertically homogeneous. A classic model is vertically homogeneous in the sense as follows: given some system occupancy $i > 0$, the probability that the occupancy will first increase by 1, before decreasing by 1, does not depend on i . Such homogeneity can be exploited to obtain explicit solutions. In the system with $d(i)$, the probability that the occupancy will first increase by 1, before decreasing by 1, depends on i , throughout function $d(i)$. Moreover, function $d(i)$ is arbitrary, hence no vertical repetitiveness can be established nor exploited to obtain the solution in explicit form.

A. SPECIAL CASES

To use formulae (16), (23), and (25), we need to know $h_k(s)$, $g(s)$, $f_k(s)$ and $y(s)$, given in (6)-(9), respectively.

For many popular forms of distribution $G(t)$, these functions can be derived symbolically. For instance, if $G(t)$ is a constant distribution at point D , then:

$$h_k(s) = \frac{1}{k!} e^{-D(s+\mu)} (D\mu)^k, \tag{26}$$

$$g(s) = e^{-Ds}, \tag{27}$$

$$f_k(s) = \frac{\mu^k}{k!(\mu+s)^{k+1}} (k! - \Gamma(k+1, D(\mu+s))), \tag{28}$$

$$y(s) = \frac{1 - e^{-Ds}}{s}, \tag{29}$$

where $\Gamma(a, z)$ is the incomplete gamma function.

If $G(t)$ is the uniform distribution on interval (A, B) , then:

$$h_k(s) = \frac{1}{k!(B-A)} [A(A\mu)^k E_{-k}(A(\mu+s)) - B(B\mu)^k E_{-k}(B(\mu+s))], \tag{30}$$

$$g(s) = \frac{e^{-As} - e^{-Bs}}{(B-A)s}, \tag{31}$$

$$f_k(s) = \frac{\mu^k}{k!(\mu+s)^{k+1}} (k! - \Gamma(k+1, A(\mu+s))) \tag{32}$$

$$y(s) = \frac{1}{s} + \frac{1}{k!(A-B)(\mu+s)} \cdot \{Ae^{-A(\mu+s)}(A\mu)^k - Be^{-B(\mu+s)}(B\mu)^k + (k - B(\mu+s) + 1)[A(A\mu)^k E_{-k}(A(\mu+s)) - B(B\mu)^k E_{-k}(B(\mu+s))]\}, \tag{33}$$

where $E_n(z)$ is the exponential integral function.

If $G(t)$ is exponential with the parameter λ , then:

$$h_k(s) = \frac{\lambda \mu^k}{(\lambda + \mu + s)^{k+1}}, \tag{34}$$

$$g(s) = \frac{\lambda}{\lambda + s}, \tag{35}$$

$$f_k(s) = \frac{\mu^k}{(\lambda + \mu + s)^{k+1}}, \tag{36}$$

$$y(s) = \frac{1}{\lambda + s}. \tag{37}$$

If $G(t)$ is the gamma distribution with the shape parameter α and the rate parameter β , then:

$$h_k(s) = \frac{\mu^k \beta^\alpha \Gamma[k + \alpha]}{k! \Gamma(\alpha) (\mu + s + \beta)^{k+\alpha}}, \tag{38}$$

$$g(s) = \left(\frac{\beta}{s + \beta}\right)^\alpha, \tag{39}$$

$$f_k(s) = \frac{\mu^k \Gamma(k + \alpha + 1) H(k + 1, k + \alpha + 1, k + 2, -\frac{\mu+s}{\beta})}{(k + 1)! \Gamma(\alpha) \beta^{k+1}}, \tag{40}$$

$$y(s) = \frac{1}{s} - \frac{1}{s} \left(\frac{\beta}{s + \beta}\right)^\alpha, \tag{41}$$

where $H(a, b, c, z)$ is the hypergeometric function.

For other forms of distribution G , quantities h_k , g , f_k and y can be obtained easily by numerical integration.

V. EXAMPLES OF NUMERIC SOLUTIONS

In these examples, the following shapes of function $d(i)$, suggested in the literature, are used:

- linear (see, e.g. [5]):

$$d_1(i) = \begin{cases} 0, & \text{if } i < 32, \\ \frac{i}{32} - 1, & \text{if } 32 \leq i < 64, \\ 1, & \text{if } i \geq 64. \end{cases} \tag{42}$$

- quadratic (see, e.g. [7]):

$$d_2(i) = \begin{cases} 0, & \text{if } i < 32, \\ \frac{i^2}{1024} - \frac{i}{16} + 1, & \text{if } 32 \leq i < 64, \\ 1, & \text{if } i \geq 64. \end{cases} \tag{43}$$

- cubic (see, e.g. [8], [9]):

$$d_3(i) = \begin{cases} 0, & \text{if } i < 32, \\ \frac{i^3}{32768} - \frac{3i^2}{1024} + \frac{3i}{32} - 1, & \text{if } 32 \leq i < 64, \\ 1, & \text{if } i \geq 64. \end{cases} \tag{44}$$

- negative exponential (see, e.g. [6]):

$$d_4(i) = \begin{cases} 0, & \text{if } i < 32, \\ \frac{1}{2} - \frac{e^{-(i-32)/10}}{2}, & \text{if } 32 \leq i < 64, \\ 1, & \text{if } i \geq 64. \end{cases} \tag{45}$$

TABLE 1. The stationary throughput for different functions $d(i)$ and std. deviations of the interarrival distribution. $\rho = 1$ in every case.

	$d_1(i)$	$d_2(i)$	$d_3(i)$	$d_4(i)$	$d_5(i)$
$S = 0$	0.987	0.988	0.989	0.986	0.988
$S = 0.1$	0.986	0.988	0.989	0.986	0.987
$S = 0.2$	0.986	0.987	0.988	0.985	0.987
$S = 0.5$	0.983	0.985	0.986	0.983	0.984
$S = 1$	0.974	0.978	0.979	0.974	0.976
$S = 2$	0.944	0.951	0.954	0.943	0.947
$S = 5$	0.802	0.819	0.827	0.809	0.805
$S = 10$	0.576	0.592	0.597	0.589	0.572

TABLE 2. The stationary throughput for different functions $d(i)$ and system loads. $S = 2$ in every case.

	$d_1(i)$	$d_2(i)$	$d_3(i)$	$d_4(i)$	$d_5(i)$
$\rho = 0.4$	0.999	1.000	1.000	0.999	1.000
$\rho = 0.6$	0.999	0.999	0.999	0.999	0.999
$\rho = 0.8$	0.994	0.996	0.997	0.993	0.995
$\rho = 0.9$	0.979	0.984	0.987	0.978	0.981
$\rho = 1.0$	0.944	0.951	0.954	0.943	0.947
$\rho = 1.2$	0.827	0.829	0.830	0.827	0.828
$\rho = 1.4$	0.713	0.714	0.714	0.713	0.713
$\rho = 1.6$	0.624	0.624	0.624	0.624	0.624

- compound, with cubic and linear parts (see, e.g. [11]):

$$d_5(i) = \begin{cases} 0, & \text{if } i < 32, \\ \frac{(i - 32)^3}{3872}, & \text{if } 32 \leq i < 43, \\ \frac{i}{32} - 1, & \text{if } 43 \leq i < 53, \\ \frac{(i - 64)^3}{3872} + 1, & \text{if } 53 \leq i < 64, \\ 1, & \text{if } i \geq 64. \end{cases} \quad (46)$$

We can notice that all these functions were adopted to the system capacity of $K = 64$. All of them operate effectively on the second half of the capacity, i.e. on $i = 32, \dots, 64$. Functions $d_1 - d_3$ can be ordered with respect to their rejectiveness. Specifically, d_2 is less rejective than d_1 (i.e. $d_1(i) \geq d_2(i)$ for every i), while d_3 is less rejective than d_2 . Functions d_4 - d_5 cannot be ordered in such a way, because they are less rejective in some intervals and more rejective in others.

The interarrival distribution is gamma with the same shape and rate parameters, $\alpha = \beta$. This setting gives a normalized mean interarrival time $m = 1$. However, altering α , we can obtain different values of the std. dev. of the interarrival distribution, S , without changing the system load. Specifically, we have:

$$S = \frac{1}{\sqrt{\alpha}}. \quad (47)$$

Similarly, altering the service rate μ , we can obtain an arbitrary system load:

$$\rho = \frac{1}{m\mu} = \frac{1}{\mu}. \quad (48)$$

A. STATIONARY EXAMPLES

We begin with examples of the stationary throughput, T .

In Tab. 1, the stationary throughput is presented, for all functions $d_1 - d_5$ and the std. deviations of the interarrival distribution varying from 0.1 to 10, while maintaining the load of 1. The following inferences can be drawn from this table.

Firstly, the std. deviation of the interarrival distribution has a tremendous effect on the system throughput. Even for a moderate $S = 2$, the throughput is reduced to about 95% in all the cases. For $S = 10$, the throughput is reduced radically,

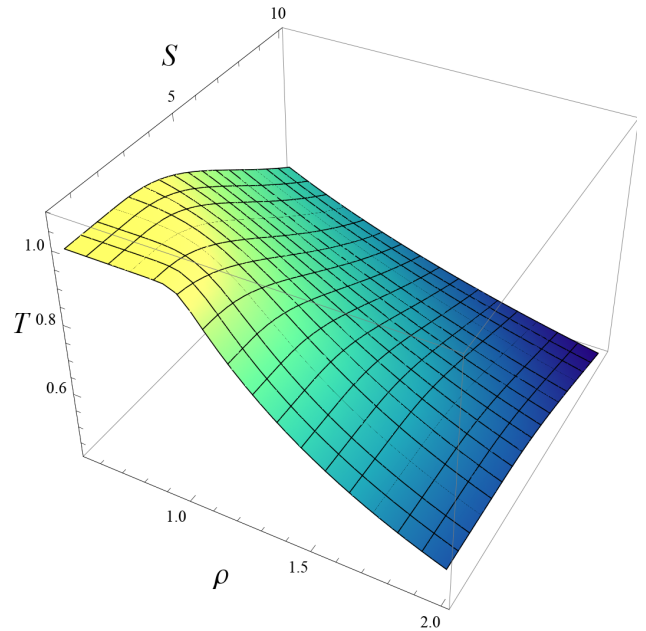


FIGURE 1. The stationary throughput as a function of the load and the std. deviation of the interarrival distribution. $d_1(i)$ is used.

to about 57%. Secondly, we see that $T_{d_1} \leq T_{d_2} \leq T_{d_3}$ for every S , which was to be anticipated seeing that $d_1(i) \geq d_2(i) \geq d_3(i)$ for every i . But if we compare the results for $d_4(i)$ and $d_5(i)$, the situation is not as straightforward. For small values of S , up to $S = 2$, $d_5(i)$ provides a better throughput than $d_4(i)$. Conversely, for high values of S , a better throughput is offered by $d_4(i)$. This can be linked to the fact that for $i \leq 42$ it holds $d_4(i) \geq d_5(i)$, while for $i > 42$ it holds $d_4(i) \leq d_5(i)$.

Finally, the throughput does not vary a lot with the form of function $d(i)$, regardless of the std. deviation of the interarrival distribution. It is essential to note, however, that all the considered functions $d(i)$ are non-decreasing and operating on the same interval, i.e., $i = 32, \dots, 64$. They differ primarily in terms of their shape and convexity. Using a broader range of different functions $d(i)$ would yield more diverse throughput results.

In Table 2, the stationary throughput is presented for all functions d_1 to d_5 with the load varying from 0.4 to 1.6, while maintaining the std. deviation of the interarrival distribution

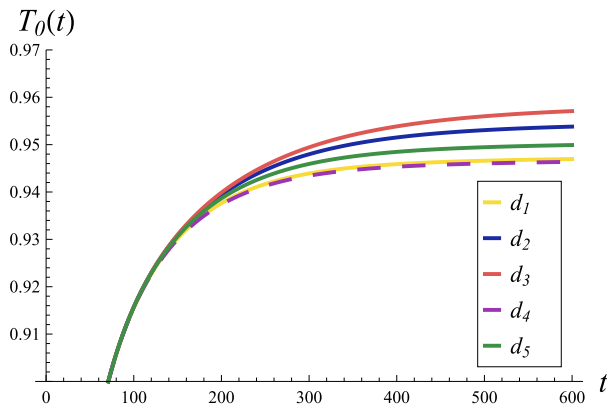


FIGURE 2. The intensity of the output proces in time for different functions $d(i)$ and the initial occupancy 0, $\rho = 1$ and $S = 2$.

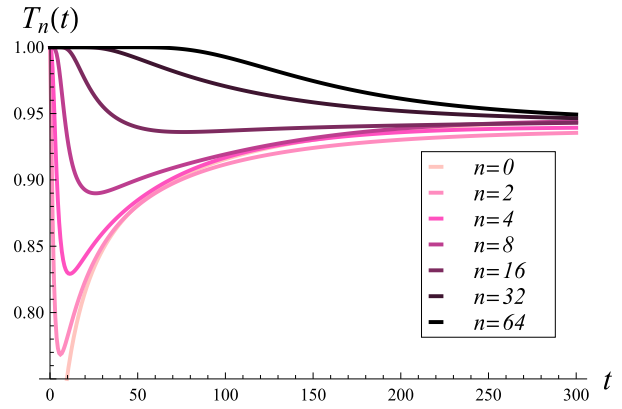


FIGURE 5. The intensity of the output proces in time for different initial system occupancies and $S = 2$, $\rho = 1$, $d_1(i)$.

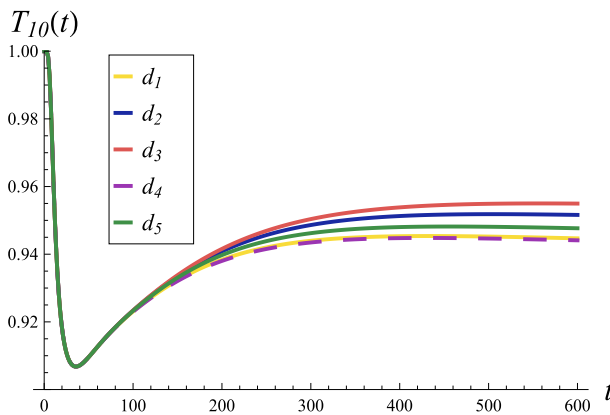


FIGURE 3. The intensity of the output proces in time for different functions $d(i)$ and the initial occupancy 10, $\rho = 1$ and $S = 2$.

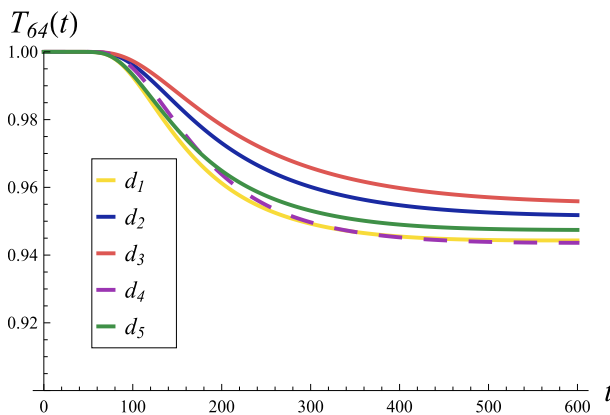


FIGURE 4. The intensity of the output proces in time for different functions $d(i)$ and the initial occupancy 64, $\rho = 1$ and $S = 2$.

at 2. It is evident that the load has a significant impact on the throughput. A nearly maximal throughput is observed for system loads up to about 0.8. However, when the load reaches 0.9, even though the system is still significantly underloaded, the throughput begins to deteriorate, losing about 2% for $\rho = 0.9$, 5% for $\rho = 1$ and 17% for $\rho = 1.2$.

Finally, in Figure 1, the stationary throughput is depicted as a function of both ρ and S for $d_1(i)$ rejections. We can see the yellow region of pairs (ρ, S) , where a relatively high throughput of the system is preserved.

B. TRANSIENT EXAMPLES

Now, we will proceed with transient examples, in which the intensity of the output process will be shown as a function of time.

In Figures 2, 3, and 4, the transient intensity of the output process is displayed for the initial system occupancies of 0, 10, and 64 jobs, respectively, with separate curves for functions $d_1 - d_5$. All three figures were generated with $\rho = 1$ and $S = 2$.

The following inferences can be drawn from these figures.

Firstly, achieving the stationary throughput, indicated by a flat curve, can take a relatively long time, up to 500-600 seconds. Secondly, the convergence time does not depend significantly on the initial occupancy. However, it is slightly influenced by the target stationary throughput. If the target throughput is higher, the convergence time tends to be slightly longer. For example, compare the d_3 curve with the d_1 curve. Thirdly, in the initial period of about 300 seconds, the course of $T_n(t)$ depends strongly on the initial occupancy, n . When $n = 64$ (Fig. 4), the server is busy for about 60 seconds, processing the initial queue. Therefore, even if the rejection rate is high during this period, the output process remains intense. Conversely, when $n = 0$ (Fig. 2), the server has to wait for new arriving jobs to process them, resulting in a lower initial output intensity. Finally, when $n = 10$ (Fig. 3), the server processes the initial queue for about 10 seconds, leading to a relatively high initial output intensity. After this initial processing, the server has to wait for new arriving jobs, causing the intensity to drop significantly and reach a local minimum. Afterward, the output intensity gradually increases until it reaches the stationary value.

This effect can be studied further in Fig. 5, in which the course of the intensity of the output proces in time is depicted for function $d_1(i)$ only, but with different initial

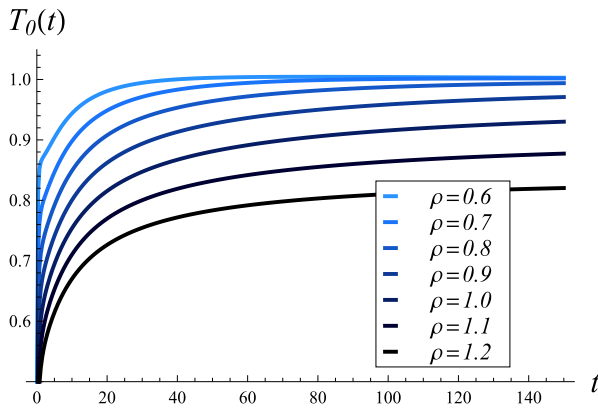


FIGURE 6. The intensity of the output proces in time for different system loads and the initial occupancy 0, $S = 2$ and $d_1(i)$.

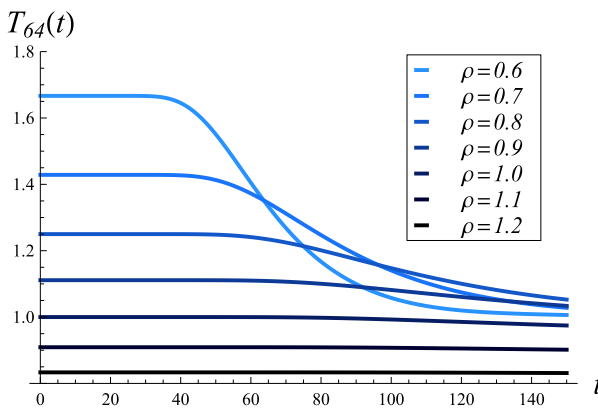


FIGURE 7. The intensity of the output proces in time for different system loads and the initial occupancy 64, $S = 2$ and $d_1(i)$.

system occupancies. The local minimum mentioned earlier can be seen for all non-zero initial occupancies up to about 32. The smaller the initial occupancy, the lower the minimum coordinate and the smaller the minimal intensity.

In the following two figures, we will investigate the transient course of the output intensity depending on the load. Specifically, in Figures 6 and 7, this intensity is depicted for the initial occupancy of 0 and 64, respectively. Both figures were obtained assuming $S = 2$ and using function $d_1(i)$.

As can be observed, the convergence to the stationary throughput is more rapid when ρ is significantly greater than 1 or significantly less than 1. When ρ is close to 1 or slightly less than 1, the convergence is slower.

It is also notable that for small values of ρ and non-zero initial system occupancy, the output intensity can temporarily exceed 1. This is caused by the fast service rate when ρ is small. Specifically, if there are jobs accumulated in the queue at some point, the fast service can process and depart them at a rate faster than the arrival rate. However, this situation is short-lived and the system eventually stabilizes.

In Figure 8, we can observe the transient performance of the output intensity depending on the std. deviation of the

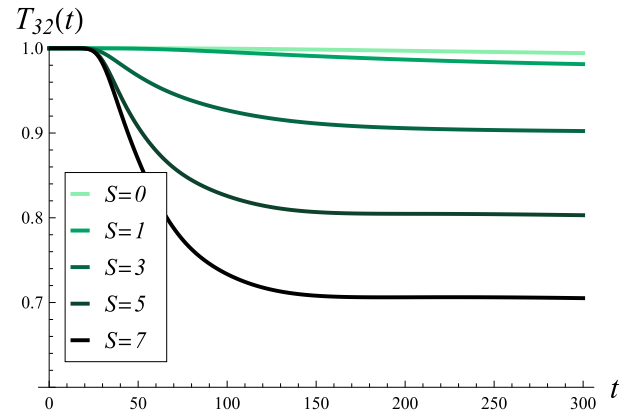


FIGURE 8. The intensity of the output proces in time for different std. deviations of the interarrival distribution and the initial occupancy 32, $\rho = 1$ and $d_1(i)$.

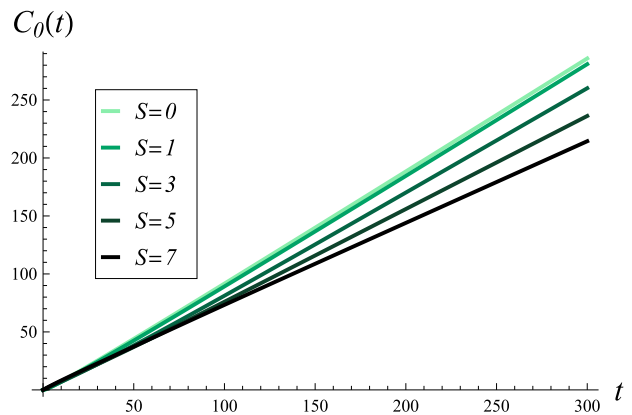


FIGURE 9. The mean number jobs completed by the time t , for different std. deviations of the interarrival distribution and the initial occupancy 0, $\rho = 1$ and $d_1(i)$.

interarrival distribution, for the initial occupancy of 32 jobs. We already know from Section V-A that high values of S induce a low stationary throughput. In Fig. 8 we may see additionally that the deterioration of the output intensity is rather quick for high S – after about 100 seconds, it is already very low.

Finally, in Figure 9 we observe how the input distribution influences the counting process, i.e. the mean number of jobs completed in $(0, t)$, starting from the initial occupancy of 0. Once again, different input distributions were used with the std. deviation varying from 0 to 7. Evidently, the growth rate is influenced by the std. deviation of the input distribution. Specifically, the smaller S , the steeper the curve. Due to this, substantial differences in the numbers of processed jobs, even in a relatively short time, may occur. For example, 214 jobs are completed on average during first 300 seconds when $S = 7$. Conversely, when $S = 0$, the number of completed jobs is 285 on average, in the same interval. This a significant discrepancy if we take into account that all the parameters, i.e. ρ , μ and $d(i)$, are the same, whereas the only difference is the std. deviation of the input distribution.

VI. CONCLUSION

We conducted a comprehensive analysis of the throughput for a queue with probabilistic job rejections based on the system occupancy. Both the stationary and transient cases were addressed. In the transient situation, we derived a formula for the number of jobs completed by the system within an interval of arbitrary length, and a formula for the intensity of the output process at time t . In the stationary situation, we found a formula for the stationary throughput.

Theoretical findings were supported by numeric calculations. These calculations covered various forms of function $d(i)$, system loads, std. deviations of the interarrival distribution, and initial system occupancies. They exposed the influence of each of these factors on the throughput.

Specifically, we observed that a significant impact on the stationary throughput is exerted by the std. deviations of the interarrival distribution and the load. For instance, even for a moderate deviation, $S = 2$, the throughput began to deteriorate for an underloaded system with $\rho = 0.9$. For a fully loaded system, $\rho = 1$, the throughput was reduced by 5%, and deteriorated rapidly as the load increased beyond that point.

Conversely, we observed that varying function $d(i)$ did not have a substantial impact on the throughput, at least when all the used functions operated on the same interval and were non-decreasing. However, the choice of $d(i)$ did have some subtle yet significant effects on the throughput. For instance, for certain forms of $d(i)$, the throughput was better when the std. deviation of the interarrival distribution was low and worse when it was high. For other forms of $d(i)$, the opposite effect was observed.

In the transient examples, we observed a relatively lengthy convergence to the stationary throughput. We noticed that the convergence time was shorter for both underloaded and overloaded systems, but longer for systems operating around $\rho = 1$ (fully loaded). Additionally, the convergence tended to be longer when the target stationary throughput was higher. Conversely, the convergence time did not exhibit a strong dependence on the initial occupancy.

An interesting avenue for future research would be analyzing mathematical models such that the rejection probability is not based on the current system occupancy, but on some other factor(s). For instance, in the mechanism proposed in [29], waiting time of jobs in the queue is used. Specifically, if the waiting time is above some parameter, for long enough, a job is rejected and the next rejection is scheduled in the future. Understanding mathematically how such waiting time-based rejection mechanism affects the system performance and how it compares to occupancy-based rejection mechanism could provide valuable insights.

REFERENCES

- [1] A. Chydzinski, D. Samociuk, and B. Adamczyk, "Burst ratio in the finite-buffer queue with batch Poisson arrivals," *Appl. Math. Comput.*, vol. 330, pp. 225–238, Aug. 2018.
- [2] F. Baker and G. Fairhurst, "Request for comments 7567," Internet Eng. Task Force, Wilmington, DE, USA, Tech. Rep. 7567, 2015.
- [3] J. Gettys and K. Nichols, "Bufferbloat: Dark buffers in the internet," *Queue*, vol. 9, no. 11, pp. 1–15, 2011.
- [4] V. G. Cerf, "Bufferbloat and other internet challenges," *IEEE Internet Comput.*, vol. 18, no. 5, p. 80, Sep. 2014.
- [5] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Netw.*, vol. 1, no. 4, pp. 397–413, 1993.
- [6] S. Athuraliya, S. H. Low, V. H. Li, and Q. Yin, "REM: Active queue management," *IEEE Netw.*, vol. 15, no. 3, pp. 48–53, May/Jun. 2001.
- [7] K. Zhou, K. L. Yeung, and V. O. K. Li, "Nonlinear RED: A simple yet efficient active queue management scheme," *Comput. Netw.*, vol. 50, no. 18, pp. 3784–3794, Dec. 2006.
- [8] D. R. Augustyn, A. Domanski, and J. Domanska, "A choice of optimal packet dropping function for active queue management," in *Proc. Int. Conf. Comput. Netw.*, vol. 79, 2010, pp. 199–206.
- [9] J. Domańska, D. R. Augustyn, and A. Domański, "The choice of optimal 3-rd order polynomial packet dropping function for NLRED in the presence of self-similar traffic," *Bull. Polish Acad. Sci., Tech. Sci.*, vol. 60, no. 4, pp. 779–786, Dec. 2012.
- [10] V. Rosolen, O. Bonaventure, and G. Leduc, "A RED discard strategy for ATM networks and its performance evaluation with TCP/IP traffic," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 29, no. 3, pp. 23–43, Jul. 1999.
- [11] C.-W. Feng, L.-F. Huang, C. Xu, and Y.-C. Chang, "Congestion control scheme performance analysis based on nonlinear RED," *IEEE Syst. J.*, vol. 11, no. 4, pp. 2247–2254, Dec. 2017.
- [12] S. Patel and Karmeshu, "A new modified dropping function for congested AQM networks," *Wireless Pers. Commun.*, vol. 104, no. 1, pp. 37–55, Jan. 2019.
- [13] A. Gimenez, M. A. Murcia, J. M. Amigo, O. Martinez-Bonastre, and J. Valero, "New RED-type TCP-AQM algorithms based on beta distribution drop functions," *Appl. Sci.*, vol. 12, no. 21, p. 11176, 2022.
- [14] M. Barczyk and A. Chydzinski, "AQM based on the queue length: A real-network study," *PLoS ONE*, vol. 17, no. 2, Feb. 2022, Art. no. e0263407.
- [15] A. Chydzinski, "Waiting time in a general active queue management scheme," *IEEE Access*, vol. 11, pp. 66535–66543, 2023.
- [16] A. Chydzinski, "On the stability of queues with the dropping function," *PLoS ONE*, vol. 16, no. 11, Nov. 2021, Art. no. e0259186.
- [17] T. Bonald, M. May, and J.-C. Bolot, "Analytic evaluation of RED performance," in *Proc. INFOCOM*, 2000, pp. 1415–1424.
- [18] W. M. Kempa, "On main characteristics of the M/M/1/N queue with single and batch arrivals and the queue size controlled by AQM algorithms," *Kybernetika*, vol. 47, no. 6, pp. 930–943, 2011.
- [19] O. Tikhonenko and W. M. Kempa, "The generalization of AQM algorithms for queueing systems with bounded capacity," in *Parallel Processing and Applied Mathematics (Lecture Notes in Computer Science)*, vol. 7204. Berlin, Germany: Springer-Verlag, 2012, pp. 242–251.
- [20] W. M. Kempa, "A direct approach to transient queue-size distribution in a finite-buffer queue with AQM," *Appl. Math. Inf. Sci.*, vol. 7, no. 3, pp. 909–915, May 2013.
- [21] O. Tikhonenko and W. M. Kempa, "Performance evaluation of an M/G/n-type queue with bounded capacity and packet dropping," *Int. J. Appl. Math. Comput. Sci.*, vol. 26, no. 4, pp. 841–854, Dec. 2016.
- [22] O. Tikhonenko and W. M. Kempa, "Erlang service system with limited memory space under control of AQM mechanism," in *Proc. Int. Conf. Comput. Netw.*, vol. 718, 2017, pp. 366–379.
- [23] A. Chydzinski, M. Barczyk, and D. Samociuk, "The single-server queue with the dropping function and infinite buffer," *Math. Problems Eng.*, vol. 2018, pp. 1–12, Oct. 2018.
- [24] P. Mrozowski and A. Chydzinski, "Queues with dropping functions and autocorrelated arrivals," *Methodol. Comput. Appl. Probab.*, vol. 20, no. 1, pp. 97–115, Mar. 2018.
- [25] A. Chydzinski, "Non-stationary characteristics of AQM based on the queue length," *Sensors*, vol. 23, no. 1, p. 485, Jan. 2023.
- [26] W. Hao and Y. Wei, "An extended G^X/M/1/N queueing model for evaluating the performance of AQM algorithms with aggregate traffic," in *Networking and Mobile Computing (Lecture Notes in Computer Science)*, vol. 3619. Berlin, Germany: Springer-Verlag, 2005, pp. 395–414.
- [27] W. M. Kempa, "Time-dependent queue-size distribution in the finite GI/M/1 model with AQM-type dropping," *Acta Electrotechnica et Inf.*, vol. 13, no. 4, pp. 85–90, Dec. 2013.
- [28] A. Chydzinski, "Queues with the dropping function and non-Poisson arrivals," *IEEE Access*, vol. 8, pp. 39819–39829, 2020.
- [29] K. Nichols and V. Jacobson, "Controlling queue delay," *Queue*, vol. 55, no. 7, pp. 42–50, (2012).

- [30] R. Pan, P. Natarajan, C. Piglione, M. S. Prabhu, V. Subramanian, F. Baker, and B. VerSteeg, "PIE: A lightweight control scheme to address the bufferbloat problem," in *Proc. IEEE 14th Int. Conf. High Perform. Switching Routing (HPSR)*, Jul. 2013, pp. 148–155.
- [31] G. Kahe and A. H. Jahangir, "A self-tuning controller for queuing delay regulation in TCP/AQM networks," *Telecommun. Syst.*, vol. 71, no. 2, pp. 215–229, Jun. 2019.
- [32] S. B. Alaoui, E. H. Tissir, and N. Chaibi, "Analysis and design of robust guaranteed cost active queue management," *Comput. Commun.*, vol. 159, pp. 124–132, Jun. 2020.
- [33] H. Wang, "Trade-off queuing delay and link utilization for solving bufferbloat," *ICT Exp.*, vol. 6, no. 4, pp. 269–272, Dec. 2020.
- [34] R. Hotchi, H. Chibana, T. Iwai, and R. Kubo, "Active queue management supporting TCP flows using disturbance observer and Smith predictor," *IEEE Access*, vol. 8, pp. 173401–173413, 2020.
- [35] C. Wang, X. Chen, J. Cao, J. Qiu, Y. Liu, and Y. Luo, "Neural network-based distributed adaptive pre-assigned finite-time consensus of multiple TCP/AQM networks," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 68, no. 1, pp. 387–395, Jan. 2021.
- [36] J. Shen, Y. Jing, and T. Ren, "Adaptive finite time congestion tracking control for TCP/AQM system with input-saturation," *Int. J. Syst. Sci.*, vol. 53, no. 2, pp. 253–264, Jan. 2022.
- [37] J. L. Schiff, *The Laplace Transform: Theory and Applications*. New York, NY, USA: Springer, 1999.



ANDRZEJ CHYDZINSKI received the M.Sc. degree (Hons.) in applied mathematics and the Ph.D. and D.Sc. degrees in computer science from the Silesian University of Technology, in 1997, 2002, and 2008, respectively. He is currently a Full Professor with the Silesian University of Technology and the Head of the Department of Computer Networks and Systems. He has been engaged as a Researcher in many other scientific initiatives. He has led six major scientific projects granted by research foundations. He has authored and coauthored two monographs, two academic textbooks, and 125 peer-reviewed articles, including contributions to numerous leading journals. His research interests include computer networking, specifically focusing on performance evaluation, queueing models, network virtualization, and network simulations. His contributions have been recognized through a recipient of five best paper awards for conference papers and a prestigious prize from POLITYKA, a well-regarded Polish weekly.

• • •