Queues with the dropping function and general service time

Andrzej Chydzinski*, Blazej Adamczyk

Institute of Informatics, Silesian University of Technology, Gliwice, Poland

* andrzej.chydzinski@polsl.pl

Abstract

We present an analysis of queueing systems with the dropping function, infinite buffer and general distribution of the service time. Firstly, a stability condition, more general than the well-known $\rho < 1$, is proven. Secondly, the formulas for the queue size distribution, loss ratio and mean duration of the busy period, are derived. Thirdly, numerical examples are given, including optimizations of the shape of the dropping function with regard to the combined cost of the queue size and loss ratio.

1 Introduction

A queueing system with the dropping function is a system, in which an arriving packet (job, customer) may not be allowed to the queue (dropped) with probability $d(n)$, where $n$ is the number of packets present in the queue on the new packet arrival.

Perhaps the simplest queueing system, in which some arriving packets are lost, is the commonly used system with a finite buffer and tail dropping. In such a system, an arriving packet is dropped, when the buffer is full, upon this packet arrival. Obviously, this is equivalent to the application of a trivial dropping function, in the form of the step function:

$$d(n) = \begin{cases} 0, & \text{if } n < N, \\ 1, & \text{if } n \geq N, \end{cases}$$

where $N$ is the buffer size.

Allowing the function $d(n)$ to have an arbitrary, non-trivial form, can be viewed as a natural generalization of the tail-drop queueing model.

There are a few good reasons, practical and theoretical, to deal with queues with such non-trivial dropping functions.

It has been observed that the tail-drop queueing of packets in an IP network causes several problems and degrades the network performance. These include long queueing delays, synchronization of TCP flows, interflow unfairness, bursty losses (long series of losses one after another), and others. Therefore, Internet Engineering Task Force recommends in [1] the usage of active queue management (AQM), which means basically that the packets should be dropped far before the buffer gets full, or even far before the queue gets long. The dropping policy can be based on several factors, including the queue size, the loss rate, the idle periods...
and others. One of the most popular approaches, justified by simplicity, ease of implementation and decent performance, is the probabilistic dropping, based on non-trivial dropping functions.

Several different forms of the dropping function have been proposed for active queue management so far, starting from the simple linear function in [2], ending with the newest three-range mixture of cubic and linear functions in [3]. Other dropping functions used in active queue management include the doubly linear function [4], the exponential function [5], the quadratic function [6], orthogonal polynomials [7] and the cubic function [8]. Some AQM algorithms use dropping functions which consider additional parameters to the current queue size, e.g. the current trend of the queue’s occupancy or the packet’s flow. In the model studied herein, the dropping function is a function of the queue size only, but over a long enough time and large number of packets, the impact of the other parameters is likely to be smoothed and averaged, so the important parameter for the steady-state is the queue size. Finally, some AQM algorithms are based on a completely different, non-probabilistic approach. For an account of such algorithms we refer the reader to [9] and the references given there.

Besides networking, this study may be of some interest in economics, e.g. for the pricing of toll roads, or for defining the quality of service in call centers. In both of these examples, the buffers are virtually infinite, but a high congestion leads customers to being dropped before getting serviced. A driver may choose to drive in an alternative road, a customer may quit the call before getting a human response or may be rejected by the system upon calling, when the queue is long. Using the results presented here, it is possible to control and optimize the customer loss process, rather than leaving it to happen spontaneously. For instance, by rejecting a place in the queue to some customers when the queue is long, we may achieve a good trade-off between the average waiting time and the customer loss ratio.

Therefore, in this paper we deal with the classic M/G/1 queueing model with the addition of a dropping mechanism based on a non-trivial dropping function. The importance of the M/G/1 system can be seen in virtually all queueing theory textbooks—this is one of the most frequently solved and discussed models. From the networking perspective, the usefulness of this model is connected with the fact, that it allows an arbitrary distribution of the service time. In most TCP/IP networks, the packet size (and the resulting service time) varies significantly, but not according to the exponential distribution—most of the probability mass is concentrated on the maximal and minimal size (e.g. 40 and 1500 bytes).

We begin the analysis with formulating and proving a sufficient condition for the system stability. This is an absolutely necessary step in an analysis of all systems, in which the queue can potentially grow to infinity. As pointed out in earlier works, the classic stability condition, \( \rho < 1 \), is far too restrictive, when dealing with the systems with the dropping function. It is easy to find models with dropping functions which are stable even when \( \rho \) is greater than one. Consider, for instance, a simple system with \( \rho = 5 \) and \( d(n) = 0.9 \) for every \( n \). For the M/M/1 model with the dropping function, several stability conditions were given in [10]. However, the model of [10] assumes that the queue size process is Markovian. Herein, the queue size process is not Markovian, so finding and proving a stability condition requires a totally different approach. On the other hand, the general service time is necessary in modeling of TCP/IP networks, as the packet processing time is far from exponential.

Then the formulas for three important characteristics—the queue size distribution, the mean busy period and the loss ratio—are derived. The distribution of the queue size is the first, most important characteristic in an analysis of every queueing system, usually accompanied by an analysis of the busy period. The loss ratio, \( L \), defined as the long-run fraction of dropped packets, is computed only for systems with losses, as the one considered herein. Obviously, the loss ratio is equivalent to the loss probability—both terms can be interchanged. The
loss ratio is especially important in networking, where the fraction of lost packets has a deep impact on the communication quality. Therefore, it has been studied extensively from networking perspective using measurements [11–15] and mathematical modeling [16–18]. Besides the loss ratio, the statistical structure of packet losses has been investigated using theoretical and experimental tools in [19, 20].

We give also several numerical examples. Firstly, a cost function combining the queue size and the loss ratio is proposed. Then, it is minimized using a class of dropping functions, depending on a parameter, thus providing a good trade-off between the two contradicting goals—a short queue and a small loss ratio. Then, the dependence of the system performance on the standard deviation of the service time, and on the system load, is studied.

The rest of the paper is organized as follows. In Section 2, the previous work on queues with dropping functions and queues with state-dependent rates is recalled and characterized briefly. In Section 3, the queueing model and notation is introduced. Then, in Section 4, the stability condition is proven. In Section 5, the most important performance characteristics are shown, including the queue size distribution, the loss ratio and the average duration of the busy period. In Section 6, numerical results motivated by a need for a dropping function that balances the average queue size and the loss ratio, are presented first. Then the impact of the standard deviation of the service time and the system load is tested. Finally, conclusions are gathered in Section 7.

2 Related work

We are unaware of any previous work studying the combination of an M/G/1 queue with an infinite buffer and a non-trivial dropping function.

As far as we know, there is only one analytical article [10], on the queueing model with the dropping function and infinite buffer. But the service time, as well as the interarrival time, are both exponential in [10]. As mentioned above, these make the model of [10] to be fully Markovian, with the birth-and-death process structure, and allows for application of several well-known analytical tools. Herein, the situation is different. Assuming the general type of the service time distribution ruins the Markovian structure and makes all derivations much more difficult. For instance, the main stability theorem requires applying advanced results of the renewal theory and computing some special characteristics of the M/G/1 queue, which is not needed in the proof of Theorem 1 of [10].

The queueing literature on models with the dropping function, but a finite buffer, is much wider. The studies began with an approximate solution of the queue with batch Poisson arrivals and a linear dropping function in [21]. Then, an approximate solution of the system with general batch arrivals and exponential service time was obtained in [22]. The exact solution of the system with Poisson arrivals and exponential service time was found in [23]. The stationary solution of the model with the general service time was shown in [24]. In [25], the transient solution of the same model was derived, while in [26], a case with multiple service stations was analyzed. In [27, 28], queues with a more complex arrival structure were analyzed, including the autocorrelation of the interarrival times in [27], and the full structure of the batch Markovian arrival process in [28]. Finally, in [29] a generalization of the queueing system with the dropping function was proposed and analyzed. This generalization assumes continuous job sizes and buffer size.

The model studied herein is somewhat similar to the model with state-dependent rates, see [30–32] and the references given there. There are, however, several differences between the two models. These differences are fundamental and make it impossible to translate one model to the other. Namely, in [30–32], the arrival and service rates depend on the system
workload, i.e. the total time needed to service the whole queue. What is important, the service time of a new customer becomes known immediately upon arrival of this customer, so the system workload is always known. For this reason, the arrival rate can be changed to adopt to the current workload. We, however, assume no knowledge about the current system workload. The service time of a customer becomes known upon departure of this customer, not upon arrival. The only thing always known herein is the queue size. Therefore, the arrival rate depends on the queue size, not the unknown workload. Moreover, we assume that the arrival rate cannot be controlled directly, so the dropping function is used for thinning the arrival process, at the cost of losses. There are no losses in the model of [30–32], except perhaps for those caused by a finite buffer, if assumed (both variants are analyzed, with a finite and infinite buffer). Finally, the assumption of knowing the current system workload might not be practical in some applications of the queueing system. In the mentioned call center example, it is obviously impossible to know in advance how long it will take to service the current queue. In a TCP/IP network node, it is also hard to predict in advance the total packet service time (see [33] for more details). For instance, all the packets of one flow may need forwarding only, while all the packets of another flow may first require a deep packet inspection, IPSec processing and, finally, forwarding. Also, the first and the last packet in a flow may need a longer processing time, as the processor has to open or close connection states. Prediction of the total packet service time is possible in theory, but requires a lot of computations, which make it impractical.

In the proof of the stability condition (Section 4), we exploit an approach based on the renewal theory in general, and the key renewal theorem in particular. This approach has proven to be useful in finding the limiting distributions in classic queues, like in [34], more complex queues, like in [35], and in other engineering problems, like in [36].

3 Description of the model

We deal with a single-server queueing model. The arrival stream is Poisson with rate $\lambda$. The service time is distributed according to distribution function $F(\cdot)$, which is not further specified.

The buffer (waiting room) is infinite.

Additionally, a dropping mechanism based on the dropping function $d(\cdot)$ is applied, which means that every arriving packet may be dropped with probability $d(n)$, where $n$ is the queue size at time of this packet arrival, including the packet being serviced, if applicable. The dropping function $d(n)$ is not further specified and may assume any value in interval $[0, 1)$ for every $n = 0, 1, 2, \ldots$. We exclude the possibility that $d(n) = 1$ for some $n$, as it is equivalent to having a finite, rather than infinite, buffer.

The queueing discipline is irrelevant to this study, it can be FIFO, LIFO, etc.

The load offered to the queue (or simply, load) is defined as:

$$\rho = \lambda m,$$

where

$$m = \int_0^\infty tdF(t)$$

is the average service time.

Let $X(t)$ denote the queue size at time $t$, including the service position, if occupied. If not stated otherwise, it is assumed that at $t = 0$ the system is empty, i.e. $X(0) = 0$. Let $P$ denote probability and $E$—the average value of a random variable.
4 Stability of the system

We consider a queueing system to be stable if and only if for every \( n = 0, 1, \ldots \), there exist the limit:

\[
\lim_{t \to \infty} \mathbb{P}(X(t) = n) = p_n,
\]

and it holds:

\[
\sum_{n=0}^{\infty} p_n = 1.
\]

If the limit does not exist for some \( n \), or the sum in (5) is not equal to 1, we consider the system to be unstable.

A single-server queue with an infinite buffer is stable if \( \rho < 1 \) \([34, 37]\). When the dropping function is applied, this assumption becomes excessive. Obviously, if \( \rho < 1 \), then the system with the dropping function is stable—it would have been stable even without dropping of packets. But following the example given in the introduction, we can easily construct stable systems with the dropping function and arbitrary large \( \rho \), for instance 1000.

The following theorem provides a tighter stability condition.

**Theorem 1.** If

\[
m < \infty,
\]

and

\[
c = \limsup_{n \to \infty} \rho(1 - d(n)) < 1,
\]

then the M/G/1 queue with the dropping function \( d(n) \) is stable.

Proof. The proof will be divided into several distinct conceptual parts (Part 1, Part 2, etc.), starting with an outline, which gives a high-level overview of the proof.

**Outline.** In Part 1, a simple dropping function, \( d_0 \), which is smaller or equal to \( d \) for every \( n \), is proposed. This function has a form of the step function and assumes only two values. As this function drops less packets than \( d \) for every possible queue size, the stability of the system under study follows from the stability of the system with \( d_0 \) instead of \( d \), and other parameters unaltered. Therefore, in Part 2 the stability of the system with \( d_0 \) is proven. This is the longest part of the proof and is divided into several pieces (Part 2a, Part 2b, etc.). In Part 2a, the key renewal theorem is used to obtain the limiting distribution of the queue size in the system with \( d_0 \). It has the form \( q_n = k_n + l_n \), where \( K_n \) and \( L_n \) are some special characteristics of the classic M/G/1 queue, while \( C \) is a special characteristic of the system with the dropping function \( d_0 \).

The rest of Part 2 is devoted to show that \( K_n, L_n \), and \( C \) are finite, given the assumptions of Theorem 1, and that the resulting limiting distribution \( q_n \) is not defective, i.e. \( \sum_{n=0}^{\infty} q_n = 1 \). In Part 2b, using some known results on the M/G/1 queue, it is shown that \( K_n \) is finite. Similarly, in Part 2c it is shown that \( L_n \) is finite. In Part 2d it is proven that \( C \) is finite and \( \sum_{n=0}^{\infty} q_n = 1 \) holds. Finally, Part 3 concludes the whole proof by gathering all partial results together.

**Part 1.** Let us first find a new, simpler dropping function \( d_0 \), which is smaller or equal to \( d \) for every \( n \) and such that it makes the system stable.

If (7) is fulfilled, then there must exist natural number \( n_0 \) such that:

\[
\rho(1 - d(n)) < \frac{c + 1}{2} < 1, \quad n > n_0.
\]
Let us define the new dropping function in the following way:

\[
d_0(n) = \begin{cases} 
1 - \frac{n - n_0}{2r}, & \text{if } n > n_0, \\
0, & \text{otherwise.}
\end{cases}
\]  
(9)

From (8) it follows that:

\[
d_0(n) \leq d(n), \quad n = 0, 1, 2, \ldots
\]  
(10)

Now we have to show that the new queueing system, with the dropping function \(d_0\) and other parameters unaltered, is stable. We will denote by \(X_0(t)\) the queue size at time \(t\) in the new system. This will make it possible to distinguish it from the queue size in the original system, denoted by \(X(t)\).

**Part 2a.** Let us first define the following sequence of moments in time:

\[0 = t_0 < t_1 < t_2 < t_3 < \ldots\]

as the consecutive moments of reaching by the queue size levels \(n_0\) and \(n_0 + 1\), alternately. Namely, if \(X_0(0) \leq n_0\), then \(t_1\) is the first time when the queue size reaches \(n_0 + 1\), then \(t_2\) is the first time after \(t_1\) when the queue size reaches \(n_0\), then \(t_3\) is the first time after \(t_2\) when the queue size reaches \(n_0 + 1\), and so on. If \(X_0(0) > n_0\), then \(t_1\) is the first time when the queue size reaches \(n_0\), then \(t_2\) is the first time after \(t_1\) when the queue size reaches \(n_0 + 1\), then \(t_3\) is the first time after \(t_2\) when the queue size reaches \(n_0\), and so on.

Formally, if \(X(0) \leq n_0\), then \(t_k\)'s are defined as:

\[t_1 = \inf \{t > 0 : X(t) = n_0 + 1\},\]
\[t_2 = \inf \{t > t_1 : X(t) = n_0\},\]
\[t_3 = \inf \{t > t_2 : X(t) = n_0 + 1\},\]

and so on. If \(X(0) > n_0\), then \(t_k\)'s are defined as:

\[t_1 = \inf \{t > 0 : X(t) = n_0\},\]
\[t_2 = \inf \{t > t_1 : X(t) = n_0 + 1\},\]
\[t_3 = \inf \{t > t_2 : X(t) = n_0\},\]

and so on.

Using the notation:

\[G(x) = P(t_2 < x | X_0(0) = n_0),\]

\[G^k(u) = 1, \quad G^k(u) = \int_0^u G^{(k-1)}(u-v) dG(v), \quad k \geq 1,\]  
(11)

and the renewal function:

\[M(u) = \sum_{k=0}^{\infty} G^k(u),\]  
(12)
we can derive the distribution of the queue size in the new system at arbitrary time. We have:

\[
\mathbb{P}(X_0(t) = l | X_0(0) = n_0) = \sum_{k=0}^{\infty} \mathbb{P}(X_0(t) = l, t_{k+1} \leq t < t_{k+2} | X_0(0) = n_0)
\]

\[
= \sum_{k=0}^{\infty} \int_0^t \mathbb{P}(X_0(t) = l, t_k \leq t < t_{k+1} | X_0(0) = n_0) dG^x(u)
\]

\[
= \sum_{k=0}^{\infty} \int_0^t \mathbb{P}(X_0(t-u) = l, t_z > t-u | X_0(0) = n_0) dG^x(u)
\]

\[
= \int_0^t \mathbb{P}(X_0(t-u) = l, t_z > t-u | X_0(0) = n_0) dM(u).
\]

Using the latter formula and the key renewal theorem [34], p. 102, yields:

\[
q_l = \lim_{t \to \infty} \mathbb{P}(X_0(t) = l) = \frac{1}{C} \int_0^\infty \mathbb{P}(X_0(t) = l, t_z > t | X_0(0) = n_0) dt, \quad l \geq 0,
\]

where \(C\) is the expected value of \(t_{2k}\) if the system starts from level \(n_0\), i.e.:

\[
C = \mathbb{E}(t_z | X_0(0) = n_0).
\]

From (13) it follows then:

\[
q_l = \frac{1}{C} \int_0^\infty \mathbb{P}(X_0(t) = l, t_z > t, t_{z+1} > t | X_0(0) = n_0) dt
\]

\[
= \frac{1}{C} \int_0^\infty \mathbb{P}(X_0(t) = l, t \leq t, t_z > t | X_0(0) = n_0) dt
\]

\[
= \frac{1}{C} \int_0^\infty \mathbb{P}(X_0(t) = l, t_z > t | X_0(0) = n_0) dt
\]

\[
+ \frac{1}{C} \int_0^\infty dt \int_0^t \mathbb{P}(X_0(t) = l, t_z > t | t \leq t_z, X_0(0) = n_0) d\mathbb{P}(t \leq u | X_0(0) = n_0)
\]

\[
= \frac{1}{C} \int_0^\infty \mathbb{P}(X_0(t) = l, t_z > t | X_0(0) = n_0) dt
\]

\[
+ \frac{1}{C} \int_0^{l_0} \mathbb{P}(X_0(t) = l, t_z > t | X_0(0) = n_0 + 1) dt.
\]

which gives the final result of this part, namely:

\[
q_l = \frac{K_l + L_l}{C},
\]

with

\[
K_l = \int_0^\infty \mathbb{P}(X_0(t) = l, t_z > t | X_0(0) = n_0) dt,
\]

\[
L_l = \int_0^{l_0} \mathbb{P}(X_0(t) = l, t_z > t | X_0(0) = n_0 + 1) dt,
\]

and \(C\) defined in (14).
To prove the stability of the system with the dropping function $d_0$, it is sufficient to show that $C$, $K_l$ and $L_l$ are finite, and that:

$$\sum_{l=0}^{\infty} q_l = 1. \quad (16)$$

**Part 2b.** It is easily seen that the system with $d_0$ operates in the same way as the classic M/G/1 queue, except for the fact, that the effective arrival rate is $\lambda$ when the queue size is less or equal to $n_0$, and $m = \lambda + 1$, when the queue size is greater than $n_0$. Moreover, $K_l$ and $L_l$ are defined for one, unaltered arrival rate only—the former for $\lambda$, the latter for $\mu$. Therefore, $K_l$ and $L_l$ constitute some special characteristics of the classic M/G/1 queueing system. Fortunately, derivations of both of these characteristics of the M/G/1 queue were already done and can be found in literature.

To study characteristic $K_l$, which is a characteristic of the classic M/G/1 queue with arrival rate $\lambda$, the results of [35] can be used. Namely, under assumption (6), adapting the notation and repeating the derivations of Section 4 of [35] give:

$$K_l = \frac{\sum_{k=0}^{n_0} d_{l-n_0+1-k}(R_{n_0+1-k} - R_{n_0-k}) + I(l = 0)}{\sum_{k=0}^{n_0} r_k(R_{n_0+1-k} - R_{n_0-k})} < \infty, \quad (17)$$

where

$$r_k = \int_0^{\infty} e^{-\lambda t} \frac{(\lambda t)^k}{k!} dt,$$  

$$d_k = I(k \geq 0) \int_0^{\infty} e^{-\lambda t} \frac{(\lambda t)^k}{k!}(1 - F(t)) dt,$$  

$$R_0 = 0, \quad R_1 = \frac{1}{r_0}, \quad R_{k+1} = R_k \left(1 - \sum_{j=0}^{k} r_{k+1} R_{k-j}\right), \quad k \geq 1, \quad (20)$$

and $I$ is an indicator function defined as: $I(A) = 1$ if $A$ is fulfilled, $I(A) = 0$ otherwise.

**Part 2c.** To obtain $L_l$, the results of the same paper can be used, but another assumption, besides (6), is required for this:

$$\rho' = m\mu < 1. \quad (21)$$

This assumption is fulfilled, as we have:

$$m\mu = m\lambda \frac{c + 1}{2\rho} = \frac{c + 1}{2} < 1. \quad (22)$$

Therefore, adapting the notation and repeating the derivations of Section 6 of [35] yields:

$$L_l = \begin{cases} \varphi_l \left[\frac{q_{l-n_0}}{q_l} + \sum_{j=0}^{l-n_0} (\sum_{j=0}^{k} b_j - 1)Q_{l-n_0-k}\right] - \sum_{k=0}^{l-n_0-1} f_{k+1} Q_{l-n_0-k}, & \text{if } l > n_0+1, \\ \varphi_l, & \text{if } l = n_0 + 1, \\ 0, & \text{otherwise}, \end{cases} \quad (23)$$
where

\[ \varphi_i = \frac{\sum_{h=0}^{n-1} f_{i-h} Q_{i-n+1-h}}{Q_{i-n+1}/Q_1 + \sum_{h=0}^{n-1} (\sum_{j=0}^{i-h} b_j - 1) Q_{i-n+1-h}} , \quad (24) \]

\[ b_k = \int_0^\infty \frac{e^{-\mu t} \mu^k}{k!} dF(t), \quad (25) \]

\[ f_k = I(k \geq 0) \int_0^\infty \frac{e^{-\mu t} \mu^k}{k!} (1 - F(t)) dt, \quad (26) \]

\[ Q_0 = 0, \quad Q_1 = \frac{1}{b_0}, \quad Q_{k+1} = Q_k - \sum_{i=0}^{k} b_{i+1} Q_{k-i}, \quad k \geq 1. \quad (27) \]

Part 2d. To compute \( C \), it can be observed that it is equal to the sum of the average time that takes the queue size to reach level \( n_0 + 1 \) from level \( n_0 \), plus the average time back, from level \( n_0 + 1 \) to \( n_0 \). Writing this formally we get:

\[ C = \mathbb{E}(t_2|X_0(0) = n_0) = \mathbb{E}(t_1|X_0(0) = n_0) + \mathbb{E}(t_1|X_0(0) = n_0 + 1). \quad (28) \]

Then we obtain:

\[ \mathbb{E}(t_1|X_0(0) = n_0) = \int_0^\infty \mathbb{P}(t_1 > t|X_0(0) = n_0) dt = \sum_{i=0}^{n_0} K_i, \quad (29) \]

which is finite, as a finite sum of finite summands. As for the second summand of (28), we have:

\[ \mathbb{E}(t_1|X_0(0) = n_0 + 1) = \int_0^\infty \mathbb{P}(t_1 > t|X_0(0) = n_0 + 1) dt = \sum_{i=n_0+1}^{\infty} L_i. \quad (30) \]

In Section 6 of [35], the latter sum was proven to be finite under assumption (21) and obtained in the following simple form:

\[ \mathbb{E}(t_1|X_0(0) = n_0 + 1) = \frac{m}{1 - m\mu}. \quad (31) \]

Thus we can conclude that:

\[ C = \mathbb{E}(t_2|X_0(0) = n_0) = \sum_{i=0}^{n_0} K_i + \frac{m}{1 - m\mu} < \infty. \quad (32) \]

Therefore, it suffices to show that (16) is fulfilled. From (15), (28), (29) and (30) it follows that:

\[ \sum_{l=0}^{\infty} q_l = C \sum_{l=0}^{\infty} (K_i + L_i) = \frac{1}{\sum_{l=0}^{n_0} K_i + \sum_{l=n_0+1}^{\infty} L_i} \sum_{l=0}^{\infty} (K_i + L_i) = 1, \quad (33) \]

The latter equality follows from the fact that \( K_i = 0 \) for \( l > n_0 \) and \( L_i = 0 \) for \( 0 \leq l < n_0 \).

Part 3. Summarizing, in (9) a simple dropping function \( d_0 \), smaller or equal to \( d \) was proposed. It was shown that the limiting queue size distribution for \( d_0 \) has form (15). In (17), (23) and (32) it was proven that all parts of (15) are finite, given the assumptions of the theorem.
Then, in (33) it was shown that the limiting distribution of the queue size for \( d_0 \) is not defective. This completes the proof of the stability of the system with the dropping function \( d_0 \).
Finally, using (10) completes the proof of Theorem 1.

5 Queue size, loss ratio and busy period

Before the stationary distribution of the queue size, \( p_n \), can be computed, we have to introduce some notation. Namely, let the queueing system be stable and \( X_n, n = 1, 2, \ldots \), denote the queue size just after the \( n \)-th departure time. We define \( \pi_k \) as:

\[
\pi_k = \lim_{n \to \infty} \mathbb{P}(X_n = k), \quad k = 0, 1, \ldots \tag{34}
\]

We will derive first formulas for the stationary distribution of the queue size and the loss ratio as functions of \( \pi_k \). Then it will be shown, how \( \pi_k \) values can be computed.

Let \( L \) denote the loss ratio, i.e. the long-run fraction of dropped packets.

**Theorem 2.** If the M/G/1 system with the dropping function \( d(\cdot) \) is stable then its loss ratio equals:

\[
L = 1 - \frac{1}{\rho} + \frac{\pi_0}{\rho^2(1-d(0)) + \rho \pi_0},
\]

while the stationary distribution of the queue size equals:

\[
p_n = (1-L) \frac{\pi_n}{1-d(n)}, \quad n = 0, 1, \ldots \tag{36}
\]

Proof. Firstly, we apply Burke’s theorem from p. 7 of [38]. It states that for a stochastic process, whose all trajectories are step functions with unit jumps, the stationary distribution for moments just after the step down must be equal to the stationary distribution for moments just before the step up, if either of the distributions exists.

Namely, let \( \hat{t}, \tilde{t}_1, \tilde{t}_2, \tilde{t}_3, \ldots \) denote consecutive moments, when a packet is accepted to the queue, and let \( \hat{p}_n \) denote the stationary distribution of the queue size just before the packet acceptance, i.e.:

\[
\hat{p}_n = \lim_{k \to \infty} \mathbb{P}(X(\hat{t}_k^-) = n), \quad n = 0, 1, \ldots \tag{37}
\]

Applying Burke’s theorem yields then:

\[
\hat{p}_n = \pi_n, \quad n = 0, 1, \ldots \tag{38}
\]

Secondly, recalling the PASTA property of the Poisson process, [39], it can be concluded that the stationary probability of the queue size \( n \) at the packet arrival epoch is equal to \( p_n \) for every \( n = 0, 1, \ldots \). The only assumption of PASTA is fulfilled here in an obvious way—the future increments of the Poisson process after time \( t \) do not depend on the queue size before \( t \).

The latter conclusion can be used then to establish the relation between \( \hat{p}_n \) and \( p_n \). Let \( t_a \) denote an arbitrary Poisson arrival time and let \( A(t_a) \) denote the event that the packet arriving at time \( t_a \) is accepted. From the definition of \( L \) we have \( \mathbb{P}(A(t_a)) = 1 - L \), while the definition of \( \hat{p}_n \) yields \( \hat{p}_n = \mathbb{P}(X(t_a) = n|A(t_a)) \). Thus we have:

\[
\hat{p}_n = \mathbb{P}(X(t_a) = n|A(t_a)) = \frac{\mathbb{P}(X(t_a) = n, A(t_a))}{\mathbb{P}(A(t_a))} = \frac{p_n(1-d(n))}{1-L}, \quad n = 0, 1, \ldots \tag{39}
\]

In the last step, the fact that probability of the queue size \( n \) at the packet arrival epoch is \( p_n \) was used.
Now, combining (39) with (38) and solving the resulting equation with respect to \( p_n \), we can finish the proof of (36). Note that the assumption \( d(n) < 1 \) is needed to derive \( p_n \) from (39) and (38).

To compute the loss ratio, we can use a formula valid for general class of single-server queueing systems with losses, which can be found for instance in [28], namely:

\[
L = 1 - \frac{1 - p_0}{\rho},
\]

where \( p_0 \) is the stationary empty queue probability. Substituting \( n = 0 \) to (36) gives:

\[
p_0 = (1 - L) \frac{\pi_0}{1 - d(0)}.
\]

Combining (40) with (41) yields:

\[
p_0 = \frac{\pi_0}{\rho(1 - d(0)) + \pi_0}.
\]

Finally, substituting (42) to (40) we obtain (35), which completes the proof of the theorem.

In Theorem 2 it was assumed that \( d(n) < 1 \) for every \( n \). If the dropping function is 1 for some \( n_0 \), then the system is in fact equivalent to the system with a finite buffer of size \( n_0 \). The distribution of the queue size and the loss ratio for the finite-buffer M/G/1/N system were obtained in [24].

To use effectively the proven formulas, the vector:

\[
\pi = (\pi_0, \pi_1, \pi_2, \ldots)
\]

has to be first computed. If the queueing system is stable, then the sequence \( X_n \) constitutes an ergodic Markov chain, with transition probabilities:

\[
p_{j,k} = \mathbb{P}(X_{n+1} = k | X_n = j), \quad j, k = 0, 1, 2, \ldots,
\]

and transition matrix:

\[
P = [p_{j,k}]_{j=0,1,2,\ldots}.
\]

Therefore \( \pi \), which is the stationary vector for \( P \), has to fulfill the standard system of equations:

\[
\pi P = \pi,
\]

and

\[
\sum_{j=0}^{\infty} \pi_j = 1.
\]

This system is infinite, but this is not a problem. In [40] it was shown, that the solution \( (\pi) \) of the truncated system:

\[
(\pi) P = (\pi),
\]

converges elementwise to \( \pi \) as \( n \to \infty \), where \( (\pi) P \) is the \( n \times n \) northwest corner truncation of the matrix \( P \). Therefore, the stationary vector \( \pi \) can be obtained with a high precision by solving a finite linear system of equations.

What is left, is to show how transition probabilities \( p_{j,k} \) can be computed up to some arbitrary large \( j \) and \( k \). This problem has already been solved in [24] in the case of the finite-buffer
system with the dropping function. This solution is applicable here as well. Firstly, let us denote by $Q_{n,k}(u)$ the probability, that in interval $(0, u]$, $k$ packets are accepted to the queue, provided that $X(0) = n$ and there is no service completion by time $u$. Using $Q_{n,k}(u)$, we can compute probability $a_{n,k}$ that $k$ packets are accepted to the queue during the service time, provided that $X(0) = n$. Namely, we have:

$$a_{n,k} = \int_0^\infty Q_{n,k}(u)dF(u). \tag{49}$$

Using $a_{n,k}$ values, transition probabilities:

$$p_{j,k} = \begin{cases} 
a_{j,k}, & \text{if } j = 0, k \geq 0, \\
a_{j-k+1,k}, & \text{if } j \geq 1, k \geq j - 1, \\
0, & \text{otherwise},
\end{cases} \tag{50}$$

can be easily obtained. Finally, $Q_{n,k}(u)$ can be computed from its Laplace transform:

$$q_{n,k}(s) = \int_0^\infty e^{-su}Q_{n,k}(u)du, \tag{51}$$

which in [24] was proven to be equal:

$$q_{n,k}(s) = \frac{\prod_{i=0}^{k-1} \lambda(1 - d(n + i))}{\prod_{i=0}^{n} (s + \lambda - \lambda d(n + i))}, \quad n \geq 0, \ k \geq 0. \tag{52}$$

In numerical calculations, (52) has to be inverted using an inversion method for the Laplace transform. Many such methods are available. We used the Zakian method of [41] to obtain results presented in Section 6.

We finish derivations of the most important characteristics of the model with the busy period.

**Theorem 3.** If the M/G/1 system with the dropping function is stable, then its average duration of the busy period is:

$$\mathbb{E}b = \frac{\rho}{\lambda \pi_0}. \tag{53}$$

Proof. It is easily seen that an empty system remains empty for an exponentially distributed time with parameter $(1 - d(0))\lambda$. Therefore, the expected duration of the idle period of the system is equal to:

$$\frac{1}{\lambda(1 - d(0))}. \tag{54}$$

On the other hand, the expected duration of the busy period, divided by the expected duration of the idle period must be the same, as the non-empty system probability, divided by the empty system probability. This yields:

$$\frac{\mathbb{E}b}{\frac{1}{\lambda(1 - d(0))}} = \frac{1 - p_0}{p_0}. \tag{55}$$
Therefore,

\[ E_b = \frac{1 - p_0}{\lambda(1 - \bar{d}(0))p_0}, \]  

(56)

Finally, substituting (42) to (56) completes the proof of (53).

6 Numerical examples

Optimization problems

Before the numerical examples are given, we can ask, what properties a “good” dropping function should have? Unfortunately, such characterization is very hard to obtain given the current state of the analysis of the M/G/1 system with the dropping function.

First of all, there is no way to tell that one dropping function is better than another in general. In fact, one dropping function can be better than another only with respect to a particular cost function. Such cost function may depend, for instance, on the average queue size \( \mathbb{E}X \) and the loss ratio \( L \), but have variable forms, e.g. \( C = \mathbb{E}X^a \cdot L^b \) or \( C = a\mathbb{E}X + bL \). It may depend also on some other variables, e.g. on the standard deviation of the queue size: \( C = \mathbb{E}X^a \cdot L^b \cdot \text{DX}^c \). Finally, it may be parameterized in several different ways. The final choice of the cost function is arbitrary and depends on our needs. Of course, the shape of a “good” dropping function depends on the choice of the cost function.

Now, even assuming that the cost function is given, it is quite hard to characterize a “good” dropping function. For instance, the common sense tells us that \( d(n) \) should be non-decreasing. Intuitively, there is no reason that the dropping probability should increase, when the queue size decreases. However, it is not easy to prove this formally, given some particular form of the cost function.

All of these do not mean that we cannot effectively use dropping functions in practice to solve optimization problems, at least in some domain. To do this, we have to choose first a cost function and parameterize it. It will be explained below, how this function can be parameterized. Then we have to choose arbitrarily a class of dropping functions depending on a parameter. Then we may find the optimal parameter with regard to the chosen cost function. The obtained solution will not be optimal in general, i.e. in the space of all possible dropping functions, but it will be optimal in the chosen class of dropping functions. Finally, we may repeat the procedure using different classes of dropping functions, e.g. of different shapes and convexity, and compare the costs obtained for them.

Herein, we will use a simple cost being a function of the average queue size and the loss ratio:

\[ C = w \mathbb{E}X + (1 - w)L, \]  

(57)

where \( w \in (0, 1) \) is the weight parameter. To set \( w \), we have to decide how bad (costful) it is when the queue size grows by \( x \), compared to when the loss ratio grows by \( y \). For instance, we may decide that it is equally bad when the queue size grows by 50, as when the loss ratio grows by 1%. In such example, we obtain immediately that \( w \) should be about 0.0002, as we have 0.0002 \cdot 50 \approx (1 - 0.0002) \cdot 1%.

Note that the decision how we value losses versus the queue size is arbitrary by nature. For instance, in one computer network we may value a short queue size more (e.g. for gaming, VoIP) while in other, a small loss ratio (e.g. for reliable transfer of documents, files, etc.).
When the cost function is properly defined, we may start searching for the optimal dropping function in some class. The first class used herein is the following:

\[
d_1(n) = \begin{cases} 
1 - e^{-\frac{n}{v}}, & \text{if } n > v, \\
0, & \text{otherwise}.
\end{cases}
\]  

(58)

It depends on shift parameter \( v \). A few sample functions from this class are depicted in Fig 1.

We assume that the weight parameter is set to \( w = 0.0002 \), the system is critically loaded, \( \rho = 1 \), the service time is constant, and the queue is stable, which follows from Theorem 1.

In Fig 2, the dependence of cost (57) on the parameter \( v \) is depicted for \( v \in (0, 120) \). The obtained numerically minimal cost is reached for \( v = 67.9 \), which gives the optimal dropping function from the class \( d_1 \). Detailed performance characteristics for the queueing system with the dropping function \( d_1 \) and \( v = 67.9 \) are also presented in the fourth column of Table 1.

![Fig 1. The dropping function \( d_1(n) \) for different values of the shift parameter \( v \).](https://doi.org/10.1371/journal.pone.0219444.g001)

![Fig 2. The cost function versus the shift parameter \( v \) for the class \( d_1(n) \).](https://doi.org/10.1371/journal.pone.0219444.g002)
Obviously, we may search for the optimal solution in other classes of dropping functions, for example in:

\[
d_2(n) = \begin{cases} 
1 - e^{-\left(\frac{n}{u}\right)^2}, & \text{if } n > 60, \\
0, & \text{otherwise.}
\end{cases}
\] (59)

where \( u \) is now the shape parameter. A few functions from this class are depicted in Fig 3.

The resulting dependence of cost (57) on \( u \in (0, 50) \) is presented in Fig 4. The obtained numerically optimal parameter value is \( u = 26.6 \). The performance of the queue for this value of \( u \) is also summarized in the last but one column of Table 1.

The third considered class of dropping functions is the following:

\[
d_3(n) = \frac{1}{2} + \frac{1}{2} \text{sgn}(z(n - 80))(1 - e^{-|z(n-80)|}),
\] (60)

where \( \text{sgn}(x) \) denotes the sign of \( x \), and \( z \) is another shape parameter. A few functions from the class \( d_3 \), for different values of \( z \), are shown in Fig 5.

In Fig 6, dependence of cost (57) on \( z \in (0, 1) \) is depicted. The obtained numerically minimum is reached for \( z = 0.259 \). Detailed performance for this value of \( z \) is also presented in the last column of Table 1.

Finally, performance characteristics obtained for six different system configurations are gathered in Table 1. In the first three columns, the class \( d_1 \) was used to achieve different trade-offs between the queue size and the loss ratio. Namely, the weight parameter \( w \) varied from 0.0001 to

| \( d_1 \) | \( d_2 \) | \( d_3 \) |
|---|---|---|
| \( v = 97.9 \) | \( v = 28.8 \) | \( v = 7.1 \) |
| \( w = 0.0001 \) | \( w = 0.001 \) | \( w = 0.01 \) |
| \( u = 67.9 \) | \( u = 26.6 \) | \( u = 0.259 \) |
| \( z = 0.0002 \) | \( z = 0.0002 \) | \( z = 0.0002 \) |

| avg. queue, \( EX \) | 50.65 | 16.06 | 5.28 | 35.64 | 34.86 | 36.58 |
| stddev., \( \Delta X \) | 29.20 | 9.37 | 3.47 | 20.56 | 20.16 | 21.20 |
| loss ratio, \( L \) | 0.496% | 1.58% | 5.01% | 0.707% | 0.723% | 0.690% |
| busy period, \( E_b \) | 200.5 | 62.2 | 18.9 | 140.5 | 137.3 | 143.9 |
| cost function, \( C \) | 0.0100 | 0.032 | 0.102 | 0.0142 | 0.0142 | 0.0142 |

Table 1. System performance for different dropping functions and their parameterizations. In every case, the best trade-off between \( EX \) and \( L \) was achieved using weight \( w \) from the first row.

https://doi.org/10.1371/journal.pone.0219444.t001

Fig 3. The dropping function \( d_2(n) \) for different values of the shape parameter \( u \).
0.01, resulting in different optimal values of the parameter $v$ of the function $d_1$. As was to be expected, the weight has a deep impact on the queue size, the loss ratio and the optimal cost—they vary significantly in the first three columns of Table 1. On the other hand, in the last three columns, a common weight was used, but with three different classes of dropping functions. As we can see, the optimal results are close to each other. Obviously, this would not happen for any parameter-dependent class of dropping functions. The final, optimal cost depends on the optimization potential of the particular class of functions, and can be higher (or, maybe, lower) than achieved 0.0142. Close to each other results obtained for $d_1$, $d_2$ and $d_3$ classes mean that these classes have similar potential to solve the chosen optimization problem.

Dependence on the service time distribution

The theoretical results presented herein differ from [10] in the distribution of the service time. Therefore, it is reasonable to ask, how the exponential service time, assumed in [10], influences the performance of the system.
From the classic theory of the M/G/1 system it follows that the average queue size should depend strongly on the standard deviation of the service time, $D(F)$. (It is rather small in the case of the exponential distribution.) We will study now this dependence.

To accomplish this, the two-point service time distribution is used, with values $\{0.1, M\}$, and probabilities $p_{0.1} = \frac{M - 1}{M - 0.1}$ and $p_M = \frac{0.9}{M - 0.1}$, respectively. As it can be easily checked, the average value of this distribution is 1 for every $M > 1$. However, by manipulating $M$ in interval $(1, \infty)$, we can obtain an arbitrary large standard deviation. We assume that the arrival rate is 1, so that the system load is 1 for every value of the parameter $M$. This allows us to study the bare impact of the standard deviation of the service time on the system performance.

The resulting dependence of the average queue size on the standard deviation of the service time is depicted in Fig 7 for dropping functions $d_1$ and $d_3$ with parameters $v = 67.9$ and $z = 0.1$, respectively. As we can see, the dependence has rather irregular form in both cases. Moreover, the average queue size varies significantly. For $D(F) \in (0, 30)$ and $d_1$, it varies from 35.6 to 50.2. For $d_3$, it varies from 27.4 to 52.6.
We computed also the average queue size in the case of the exponential service time. For the dropping function $d_1$ it equals 36.0, while for $d_3$ it equals 30.3. Comparing these numbers with intervals mentioned in the previous paragraph, we can conclude that the standard deviation of the service time has a deep impact on the average queue size.

In Fig 8, the dependence of the loss ratio on the standard deviation of the service time is shown. This time the shapes of both curves are almost identical, but the values vary even more than in the case of the queue size. For both dropping functions, the loss ratio varies from about 1%, up to 44% (!). For comparison, the loss ratio for $d_1$ and the exponential service time is only 1.4%, while for $d_3$, it is only 1.7%.

Summarizing, the model with the exponential service time may produce a large error when the real distribution is far from exponential.

**Dependence on the load**

In the final set of examples, we check the sensitivity of the average queue size and the loss ratio on the load offered to the system. In the calculations, the variable load in interval $(0, 2)$ is obtained by manipulating the arrival rate. The dropping function $d_1$ is used with three different parameters: $\nu = 5$, $\nu = 30$ and $\nu = 70$.

The resulting dependence of the average queue size on the system load is depicted in Fig 9. As we can see, the plot depends significantly on the parameterization of the dropping function, especially for $\rho > 1$. Moreover, it can be observed that the dependence is convex for $\rho < 1$ and concave for $\rho > 1$.

The dependence of the loss ratio on the system load is depicted in Fig 10. As previously, the shape depends on the parameterization of the dropping function, but less profoundly than in the case of the queue size. As regards convexity, the inflection point is not $\rho = 1$ anymore and its location depends on the parameterization of the dropping function.

**7 Conclusions**

We presented an analysis of the classic M/G/1 queueing model, but with an additional dropping mechanism, based on the dropping function. The study is motivated by the application of the model in networking, namely active queue management in Internet routers. It can be also applied in different areas of computer science, engineering and economics.
We first proved a more general stability condition than the classic $\rho < 1$, which is not useful in the considered model, as far too restrictive. Then, we derived the formulas for the queue length distribution, the loss ratio and the mean duration of the busy period. Finally, we presented numerical results. They were aimed at the optimization of the shape of the dropping function with regard to the combined cost of the queue size and loss ratio, and at demonstrating the dependence of the system performance on the load and the standard deviation of the service time.

There are several possible directions of future work.

Firstly, it would be solving the classic G/M/1 queue with the dropping function, i.e. the system with general renewal arrivals, exponential service, infinite buffer and the dropping function.

Secondly, it would be proving that the dropping function optimal with regard to cost (57), or some other cost, must have some properties, e.g. must be non-decreasing.

Thirdly, it would be proving that the dependence of the average queue size on $\rho$ is convex for $\rho < 1$ and concave for $\rho > 1$, as observed in Fig 9.

![Fig 9. The average queue size vs the system load for the dropping function $d_1$ with different values of the parameter $v$.](https://doi.org/10.1371/journal.pone.0219444.g009)

![Fig 10. The loss ratio vs the system load for the dropping function $d_1$ with different values of the parameter $v$.](https://doi.org/10.1371/journal.pone.0219444.g010)
Acknowledgments

This work was conducted within project 2017/25/B/ST6/00110, funded by National Science Centre, Poland.

Author Contributions

Conceptualization: Andrzej Chydzinski.
Formal analysis: Andrzej Chydzinski, Blazej Adamczyk.
Investigation: Andrzej Chydzinski.
Methodology: Andrzej Chydzinski, Blazej Adamczyk.
Software: Andrzej Chydzinski, Blazej Adamczyk.
Supervision: Andrzej Chydzinski.
Visualization: Andrzej Chydzinski.
Writing – original draft: Andrzej Chydzinski.
Writing – review & editing: Andrzej Chydzinski.

References

1. Internet Engineering Task Force. Request for Comments 7567. Eds. Baker F. and Fairhurst G., (2015).
2. Floyd S. and Jacobson V. Random early detection gateways for congestion avoidance. IEEE/ACM Transactions on Networking 1(4), pp. 397–413, (1993). https://doi.org/10.1109/90.251892
3. Feng C., Huang L., Xu C. and Chang Y. Congestion Control Scheme Performance Analysis Based on Nonlinear RED. IEEE Systems Journal 11(4), pp. 2247–2254, (2017). https://doi.org/10.1109/JSYST.2014.2375314
4. Rosolen V., Bonaventure O. and Leduc G. A RED discard strategy for ATM networks and its performance evaluation with TCP/IP traffic. ACM SIGCOMM Computer Communication Review 29(3), pp. 23–43 (1999). https://doi.org/10.1145/505724.505728
5. Athuraliya S., Li V.H., Low S.H. and Yin Q. REM: active queue management. IEEE Network 15, pp. 48–53, (2001). https://doi.org/10.1109/65.923940
6. Zhou K, Yeung K. L and Li V. Nonlinear RED: asimple yet efficient active queue management scheme, Computer Networks 50, pp. 3784–3784, (2006). https://doi.org/10.1016/j.comnet.2006.04.007
7. Augustyn D. R., Domanski A. and Domanska J. A choice of optimal packet dropping function for active queue management, Communications in Computer and Information Science 79, pp. 199–206, (2010).
8. Domanska J., Augustyn D.R and Domanski A. The choice of optimal 3-rd order polynomial packet dropping function for NLRED in the presence of self-similar traffic, Bulletin of the Polish Academy of Sciences: Technical Sciences, Vol. 60, Issue 4, pp. 779–786, (2012). https://doi.org/10.2478/v10175-012-0090-x
9. Chrost L. and Chydzinski A. On the deterministic approach to active queue management. Telecommunication Systems 63(1), pp. 27–44, (2016). https://doi.org/10.1007/s11235-015-9969-9
10. Chydzinski A., Barczyk M. and Samociuk D. The Single-Server Queue with the Dropping Function and Infinite Buffer. Mathematical Problems in Engineering, vol. 2018, ID 3260428, pp. 1–12, (2018). https://doi.org/10.1155/2018/3260428
11. Benko P. and Veres A. A passive method for estimating end-to-end TCP packet loss. Proc. of IEEE GLOBECOM, pp. 2609–2613, (2002).
12. Bolot J. End-to-end packet delay and loss behavior in the Internet. Proc of ACM SIGCOMM, pp. 289–298, (1993).
13. Coates M. and Nowak R. Network loss inference using unicast end-to-end measurement. Proc. of ITC Conference on IP Traffic, Measurement and Modeling, pp. 282–289, (2000).
14. Duffield N., Presti F. L., Paxson V. and Towsley D. Inferring link loss using striped unicast probes. Proc. of IEEE INFOCOM, pp. 915–923, (2001).
15. Sommers J., Barford P., Duffield N. and Ron A. Improving accuracy in end-to-end packet loss measurement. ACM SIGCOMM Computer Communication Review, vol. 35(4), pp. 157–168, (2005). https://doi.org/10.1145/1090191.1080111

16. Takagi H. Queueing analysis—Finite Systems. North-Holland Amsterdam, (1993).

17. Chydzinski A., Wojcicki R. and Hryn G. On the Number of Losses in an MMPP Queue. Lecture Notes in Computer Science, vol. 4712, pp. 38–48, (2007).

18. Chydzinski A. and Adamczyk B. Transient and stationary losses in a finite-buffer queue with batch arrivals. Mathematical Problems in Engineering, vol. 2012, ID 326830, pp. 1–17, (2012). https://doi.org/10.1155/2012/326830

19. Cidon I., Khamisy A. and Sidi M. Analysis of packet loss processes in high-speed networks. IEEE Transactions on Information Theory, vol. 39(1), pp. 98–108, (1993). https://doi.org/10.1109/18.179347

20. Bratlychuk M. and Chydzinski A. On the loss process in a batch arrival queue. Applied Mathematical Modelling, Vol. 33, Iss. 9, pp. 3565–3577, (2009). https://doi.org/10.1016/j.apm.2008.11.015

21. Bonald T., May M. and Bolot J.-C. Analytic evaluation of RED performance, Proc. of IEEE INFOCOM, pp. 1415–1424, (2000).

22. Hao W. and Wei Y. An Extended $G^X/M/1/N$ Queueing Model for Evaluating the Performance of AQM Algorithms with Aggregate Traffic. Lecture Notes in Computer Science 3618, pp. 395–414, (2005).

23. Kempa W.M. On main characteristics of the $M/M/1/N$ queue with single and batch arrivals and the queue size controlled by AQM algorithms, Kybernetika 47(6), pp. 930–943, (2011).

24. Chydzinski A. and Chrost L. Analysis of AQM queues with queue-size based packet dropping. Applied Mathematics and Computer Science (21)3, pp. 567–577, (2011).

25. Kempa W.M. A direct approach to transient queue-size distribution in a finite-buffer queue with AQM. Applied Mathematics & Information Sciences 7(3), pp. 909–915, (2013). https://doi.org/10.12785/amis/070308

26. Tikhonenko O. and Kempa W.M. Performance evaluation of an $M/G/N$-type queue with bounded capacity and packet dropping. Applied Mathematics and Computer Science 26(4), pp. 841–854, (2016).

27. Mrozowski P. and Chydzinski A. Queues with dropping functions and autocorrelated arrivals. Methodology and Computing in Applied Probability 20(1), pp. 97–115, (2018). https://doi.org/10.1007/s11009-016-9534-3

28. Chydzinski A. and Mrozowski P. Queues with dropping functions and general arrival processes, PLoS ONE 11(3), pp. 1–23, (2016). https://doi.org/10.1371/journal.pone.0150702

29. Tikhonenko O. and Kempa W.M. Erlang service system with limited memory space under control of AQM mechanism. Communications in Computer and Information Science 718, pp. 366–379, (2017). https://doi.org/10.1007/978-3-319-59767-6_29

30. Bekker R., Borst S.C., Boxma O.J. and Kella O. Queues with workload-dependent arrival and service rates. Queueing Systems 46, pp. 537–556, (2004). https://doi.org/10.1023/B:QUEUES.0000027998.95375.ee

31. Bekker R. Finite-buffer queues with workload-dependent service and arrival rates. Queueing Systems 50, pp. 231–253, (2005). https://doi.org/10.1007/s11134-005-0730-z

32. Bekker R. Queues with state-dependent rates. Technische Universiteit Eindhoven, (2005).

33. Shpiner A., Keslassy I. and Cohen R. Scaling Multi-Core Network Processors Without the Reordering Bottleneck. Proc. of IEEE Int. Conf. on High Performance Switching and Routing, pp. 146–153, Vancouver, 1–4 July, (2014).

34. Cohen J.W. The Single Server Queue. Revised edition, North-Holland Publishing Company, Amsterdam, (1982).

35. Chydzinski A. The $M/G-G/1$ oscillating queueing system. Queueing Systems, vol. 42:3, pp. 255–268, (2002). https://doi.org/10.1023/A:1020523931025

36. Battaila J.M. et al. Adaptive video streaming: rate and buffer on the track of minimum re-buffering. IEEE Journal on Selected Areas in Communications 34(8), pp. 2154–2167, (2016). https://doi.org/10.1109/JSAC.2016.2577360

37. Kleinrock L. Queueing Systems: Theory. John Wiley and Sons New York, (1975).

38. Takagi H. Queueing analysis—Vacation and Priority System. North-Holland, Amsterdam, 1991.

39. Wolff R.W. Poisson Arrivals See Time Averages. Operations Research 30(2), pp. 223–231, (1982). https://doi.org/10.1287/opre.30.2.223

40. Seneta E. Computing the stationary distributions for infinite Markov chains. Linear Algebra and its Applications 34, pp. 259–267, (1980). https://doi.org/10.1016/0024-3795(80)90168-8

41. Zakian V. Numerical Inversion of Laplace Transform. Electronic Letters 5(6), pp. 120–121, (1969). https://doi.org/10.1049/el:19690090