On the Transient Queue with the Dropping Function

Andrzej Chydzinski

Department of Computer Networks and Systems, Silesian University of Technology, Akademicka 16,
44-100 Gliwice, Poland; andrzej.chydzinski@polsl.pl

Received: 4 July 2020; Accepted: 23 July 2020; Published: 28 July 2020

Abstract: We deal with a queueing system, in which arriving packets are being dropped with the probability depending on the queue size. Such a scheme is used in several active queue management schemes proposed for Internet routers. In this paper, we derive and analyze a selected transient characteristic of the model, i.e., the probability that in a given time interval the queue size is kept under a predefined level. As the main purpose of the discussed queueing scheme is to maintain the queue size low, this is a natural characteristic to study. In addition to that, the average time to reach a given level is derived. Theoretical results for both characteristics are accompanied by numerical examples. Among other things, they demonstrate that the transient behavior of the queue may vary significantly with the shape of the dropping function, even if the steady-state performance remains unaltered.

Keywords: queueing model; dropping function; M/G/1; Internet routers; active queue management; transient analysis

1. Introduction

It is known that there is a partial divergence in the design goals of the network layer and the transport layer in the contemporary TCP/IP networks and the Internet. Namely, the buffers in the network layer (routers) were originally meant to store bursts of packets, occurring occasionally due to statistical multiplexing of different flows. Such buffers do not have to be large, and they do not introduce a substantial queueing delay. On the other hand, developed much later, congestion control for the transport layer uses the same buffers for a different purpose. Namely, the TCP protocol probes the throughput, currently available for a particular flow, by increasing its sending rate, until the buffer is full and something bad happens (e.g., a packet is lost or the queueing delay gets large). Such behavior of TCP makes the buffers overflowed for a significant fraction of time, no matter how large they are. As a consequence, the queueing delays are unnecessarily high. This is the so called bufferbloat phenomenon, well described in the networking literature, for example, [1,2].

As a cure for this, the Internet Engineering Task Force advises the application of active queue management in Internet routers, [3]. The general idea is that every packet arriving to the router can be dropped with some probability. Roughly speaking, this probability should be relatively high, when there are signs of forthcoming congestion. Such preventive packet dropping enables a substantial reduction of the average queue size and delay, and has other positive effects (e.g., desynchronization of TCP flows). The dropping probability, however, cannot be too high, as emptying the buffer completely would cause an underutilization of the link.

How exactly the dropping probability should evolve and on which factors it should depend, is a long debate among researchers. Several advanced algorithms for computing the dropping probability were proposed to date, see, for example [4–7] and the references given there. Some propositions are based on artificial neural networks (e.g., [8]), fuzzy logic (e.g., [9]) or genetic algorithms (e.g., [10]).

In others, the dropping probability is replaced by a deterministic decision about each arriving packet, i.e., whether to drop it, or not [11].

A very important subclass of algorithms proposed in the literature exploits an idea of the dropping function. Namely, an arriving packet can be dropped with the probability being a function of the queue size. Several candidates for such function have been studied, beginning with the linear function [12], through the doubly-linear one [13], exponential [14], quadratic [15], cubic [16], and the newest proposition being a mixture of linear and cubic functions [17]. The algorithms based on the dropping function may not always provide as good performance as those cited in the previous paragraph, but their greatest advantage is an extreme ease of implementation, accompanied by a decent performance, much better than no active management at all. Obviously, to provide a theoretical background for such algorithms, queueing models with the dropping function have to be studied. Initially, they were mainly studied via simulations. Recently, more efforts have been made on mathematical analysis, using tools and concepts of the queueing theory.

In this paper, we derive a selected transient characteristic of the queueing model with the dropping function—the probability that in a given time interval the queue size is kept under a predefined level. As argued above, the main purpose of the discussed queueing scheme is to maintain a low queue size. Therefore, such characteristic is very natural to deal with. In addition to that, the average time to reach a given level is studied. The importance of those characteristics is underlined in numerical examples, where it is shown that different dropping functions may induce a very different transient behavior of the system, even if they provide the same stationary behavior.

We assume Poisson arrivals and general service time distribution in the analysis. The dropping function is also assumed in a general form, i.e., every mentioned above function (and many others) can be studied using the obtained results. In other words, the model studied herein is the M/G/1 model in Kendall's notation, with the addition of the dropping function. Note that such a model is very general. It incorporates the classic M/G/1 model (when the dropping function equals 0 for every argument), the classic M/G/1/N model (when the dropping function assumes 1 for $n \geq N$ and 0 otherwise) and the M/G/1/N model with the dropping function (when the dropping function assumes 1 for $n \geq N$ and some other values otherwise).

To the best of the author's knowledge, the results presented herein are new. For studies on other characteristics (the queue size, loss probability, response time) of systems with the dropping function, or carried out under different assumptions on the arrival process and service times, we refer the reader to [18–28]. On the other hand, there are several papers on the time to reach a given level in classic queueing models, i.e., without the dropping function—see, for example, [29–33] and the references given there.

The analytical method used herein is based on formulating and solving a system of Volterra integral equations (see also, e.g., [34,35]). In this method, the system of equations is formulated first using probabilistic properties of the model. Then it is solved using the Laplace transform technique. Finally, in order to perform numerical calculations, a method for inverting the Laplace transform is needed. We recommend one such method, which combines a good speed with a decent accuracy in inverting probability distributions.

The rest of the paper is structured as follows. In Section 2, the queueing model is formally presented and accompanied by basic notations and conventions. In Section 3, an analysis of the probability of not reaching a given level in a time interval, as well as the average time to reach a given level, is carried out. The analysis ends with formulas for the two characteristics, presented in Theorem 1 and Corollary 1, respectively. In the same section, computational aspects are discussed, with the emphasis on obtaining numerical values by using the transform inversion. In Section 4, numerical examples are shown. They are focused on demonstrating the impact of the dropping function on the studied transient characteristics. Namely, five different dropping functions are shown to induce different transient behavior, even if each of them is parameterized to provide the same stationary behavior (the average response time). Additionally, an example on how we may design the

dropping function in such a way that it meets some performance goals, is presented. Finally, remarks concluding the paper are given in Section 5.

2. Model Description

We analyze the M/G/1 queueing model in Kendall's notation, with the addition of the dropping function in a general form.

Namely, the arrival process is Poisson with the rate λ , while the service time has distribution function $F(\cdot)$, which is not further specified, except for the fact, that its average value is finite, i.e.:

$$\int_0^{\infty} t dF(t) < \infty. \quad (1)$$

The buffer for packets is infinite.

Moreover, each arriving packet can be dropped (deleted) with the probability $d(n)$, where n is the length of the queue at the time of this packet arrival, including the packet being serviced, if applicable. The dropping function $d(n)$ may assume any value in $[0, 1]$ for every $n = 0, 1, 2, \dots$. The queueing discipline is irrelevant in the analysis, it can be FIFO, LIFO, etc.

The following notation will be used: $\mathbb{P}$ for probability, $\mathbb{E}$ for the average value of a random variable, $X(t)$ for the queue length at the time t , including the service position, if occupied. We use the convention that $X(t)$ is left-continuous, i.e. $X(t) = X(t-)$. If $X(0+) > 0$, then it is assumed that $t = 0$ is the beginning of the service time.

3. Analysis

The main characteristic of interest, i.e., the probability that the queue size does not reach level M by the time t , assuming it starts from the level n , will be denoted by $Y_{n,M}(t)$, i.e.:

$$Y_{n,M}(t) = \mathbb{P}(H_{n,M} > t), \quad (2)$$

where

$$H_{n,M} = \inf\{t > 0 : X(t) = M | X(0+) = n\}, \quad 0 \leq n < M, \quad (3)$$

is the time of reaching the level M .

Our first goal is to derive the formula for the Laplace transform of the function $Y_{n,M}(t)$, i.e.:

$$y_{n,M}(s) = \int_0^{\infty} e^{-st} Y_{n,M}(t) dt, \quad s > 0, \quad (4)$$

in the vector form:

$$y_M(s) = [y_{0,M}(s), \dots, y_{M-1,M}(s)]^T. \quad (5)$$

Let $Q_{n,k}(u)$ denote the probability, that in the time interval $(0, u]$ exactly k packets were accepted to the queue, assuming that it was $X(0+) = n$ and there was no service completion by the time u . In other words, $Q_{n,k}(u)$ is the number of effective arrivals in the interval $(0, u]$, after the filtration by the dropping function. In addition to $Q_{n,k}(u)$, the following transforms will be of use:

$$a_{n,k}(s) = \int_0^{\infty} e^{-su} Q_{n,k}(u) dF(u), \quad s > 0, \quad (6)$$

$$b_{n,k}(s) = \int_0^{\infty} e^{-su} Q_{n,k}(u) (1 - F(u)) du, \quad s > 0. \quad (7)$$

Using the law of total probability with respect to the first departure time u , for $1 \leq n \leq M-1$ we have:

$$Y_{n,M}(t) = \sum_{k=0}^{M-n-1} \int_0^t Q_{n,k}(u) Y_{n+k-1,M}(t-u) dF(u) + (1-F(t)) \sum_{k=0}^{M-n-1} Q_{n,k}(t). \quad (8)$$

In particular, the first part of (8) corresponds to the event that the first departure happens before t . In this case, the number of effective arrivals by the time u must be less than $M-n$. At the first departure time, u , the new queue size is clearly $n+k-1$, while the remaining time to hit the level M is $t-u$. The second part of (8) corresponds to the event that the first departure happens after t , which has probability $1-F(t)$. In this case, the number of effective arrivals must be less than $M-n$ by the time t .

For $n=0$, we can use the law of total probability with respect to the first arrival time, v . Namely, we obtain:

$$Y_{0,M}(t) = (1-d(0)) \int_0^t Y_{1,M}(t-v) \lambda e^{-\lambda v} dv + d(0) \int_0^t Y_{0,M}(t-v) \lambda e^{-\lambda v} dv + e^{-\lambda t}. \quad (9)$$

The first part of (9) corresponds to the case where a packet arrives by the time t and it is accepted to the system. Therefore, the new queue size at the time v is 1 and the remaining time to hit the level M is $t-v$. The second part of (9) corresponds to the case where a packet arrives by the time t , but it is dropped. Therefore, the queue size is still 0 at the time v . Finally, the third part of (9) corresponds to the case where there are no new arrivals by the time t . It means also that the queue size cannot hit the level M by the time t .

Applying the Laplace transform to (8) and (9), we get:

$$y_{n,M}(s) = \sum_{k=0}^{M-n-1} a_{n,k}(s) y_{n+k-1,M} + \sum_{k=0}^{M-n-1} b_{n,k}(s), \quad 1 \leq n \leq M-1, \quad (10)$$

and

$$y_{0,M}(s) = \frac{\lambda(1-d(0))}{s+\lambda} y_{1,M}(s) + \frac{d(0)\lambda}{s+\lambda} y_{0,M}(s) + \frac{1}{s+\lambda}, \quad (11)$$

respectively.

Now, introducing the matrix $R_M(s)$ defined as:

$$R_M(s) = [r_{i,j}(s)]_{i=0 \dots M-1, j=0 \dots M-1}, \quad (12)$$

$$r_{i,j}(s) = \begin{cases} 1 - \frac{d(0)\lambda}{s+\lambda}, & \text{if } i=j=0, \\ \frac{(d(0)-1)\lambda}{s+\lambda}, & \text{if } i=0, j=1, \\ a_{i,0}(s), & \text{if } i=j+1, \\ a_{i,1}(s) - 1, & \text{if } 1 \leq i=j \leq M-2, \\ a_{i,j-i+1}(s), & \text{if } 1 \leq i < j \leq M-2, \\ -1, & \text{if } i=j=M-1, \\ 0, & \text{otherwise,} \end{cases} \quad (13)$$

and vector:

$$w_M(s) = [w_{0,M}(s), \dots, w_{M-1,M}(s)]^T, \quad (14)$$

$$w_{0,M}(s) = \frac{1}{s+\lambda}, \quad (15)$$

$$w_{i,M}(s) = - \sum_{k=0}^{M-i-1} b_{i,k}, \quad 1 \leq i \leq M-1, \quad (16)$$

from (10) and (11) we have:

$$R_M(s)y_M(s) = w_M(s). \quad (17)$$

Finally, from (17) we obtain the following theorem.

Theorem 1. The Laplace transform of the probability that the queue size does not reach the level M by the time t is:

$$y_M(s) = R_M^{-1}(s)w_M(s), \quad (18)$$

where the matrix $R_M(s)$ is given in (13), while the vector $w_M(s)$ in (15) and (16).

It easy to see that:

$$\mathbb{E}H_{n,M} = \int_0^\infty \mathbb{P}(H_{n,M} > t)dt = \int_0^\infty Y_{n,M}(t)dt = \lim_{s \rightarrow 0^+} y_{n,M}(s). \quad (19)$$

Thus we have the following corollary.

Corollary 1. The average time to reach the level M by the queue size, if starting from the level n , is equal to:

$$\mathbb{E}H_{n,M} = \lim_{s \rightarrow 0^+} \left[R_M^{-1}(s)w_M(s) \right]_n, \quad (20)$$

where $[\cdot]_n$ denotes the n -th element of a vector.

In order to use Theorem 1 or Corollary 1 in practice, we have to be able to compute the values of $Q_{n,k}(u)$. Fortunately, $Q_{n,k}(u)$ has a known Laplace transform, namely:

$$q_{n,k}(s) = \frac{\prod_{i=0}^{k-1} \lambda(1 - d(n+i))}{\prod_{i=0}^k (s + \lambda - \lambda d(n+i))}, \quad n \geq 0, \quad k \geq 0, \quad (21)$$

where

$$q_{n,k}(s) = \int_0^\infty e^{-su} Q_{n,k}(u)du, \quad s > 0, \quad (22)$$

(see [21] for the proof).

Therefore, the values of $Q_{n,k}(u)$ can be computed using one of the available methods of inversion of the Laplace transform. In the following numerical examples, the Zakian method, [36], is used for this purpose. Namely, the original function $G(t)$, given by its Laplace transform $g(s)$, is equal approximately to:

$$G(t) \approx \frac{2}{t} \sum_{k=0}^4 \operatorname{Re} \left[\omega_k g \left(\frac{\alpha_k}{t} \right) \right], \quad (23)$$

with coefficients α_k and ω_k given in Table 1:

Table 1. Coefficients for the Zakian inversion algorithm.

Coefficient	Value
α_0	12.83767675 + i1.666063445
α_1	12.22613209 + i5.012718792
α_2	10.93430308 + i8.409673116
α_3	8.776434715 + i11.92185389
α_4	5.225453361 + i15.72952905
ω_0	-36902.08210 + i196990.4257
ω_1	61277.02524 - i95408.62551
ω_2	28916.56288 + i18169.18531
ω_3	4655.361138 - i1.901528642
ω_4	-118.7414011 - i141.3036911

The same inversion method can be used to invert (18) in order to obtain the probability that the queue size does not reach a given level.

4. Examples

In the examples, we focus on the impact of the shape and parameterization of the dropping function on the studied performance characteristics. Therefore, we assume simply that the arrival rate is 1, the service time is constant and equal to 1, and, as a consequence, $\rho = 1$. We assume also that $X(0+) = 0$. The assumption that initially the queue is empty is the most natural one, but all the calculations could be repeated for an arbitrary initial queue size, which is enabled by the formulas proven in the previous section.

The following five dropping functions are used in the examples (see also Figure 1):

- RED-type dropping function, [12]:

$$d_1(n) = \begin{cases} 0, & \text{if } n < 30, \\ 0.01026n - 0.30775, & \text{if } 30 \leq n < 60, \\ 0.30775, & \text{if } n \geq 60, \end{cases} \quad (24)$$

- GRED-type dropping function, [13]:

$$d_2(n) = \begin{cases} 0, & \text{if } n < 30, \\ 0.01012n - 0.30368, & \text{if } 30 \leq n < 45, \\ 0.03037n - 1.21470, & \text{if } 45 \leq n < 60, \\ 0.60735, & \text{if } n \geq 60, \end{cases} \quad (25)$$

- NLRED-type dropping function, [15]:

$$d_3(n) = \begin{cases} 0, & \text{if } n < 20, \\ 0.000178781(n - 20)^2, & \text{if } 20 \leq n < 60, \\ 0.28605, & \text{if } n \geq 60, \end{cases} \quad (26)$$

- TRED-type dropping function, [17]:

$$d_4(n) = \begin{cases} 0, & \text{if } n < 30, \\ 0.000208916\left(\frac{n-30}{30}\right)^3, & \text{if } 30 \leq n < 40, \\ 0.020892n - 0.62675, & \text{if } 40 \leq n < 50, \\ 0.000208916\left(\frac{n-60}{30}\right)^3 + 0.62675, & \text{if } 50 \leq n < 60, \\ 0.62675, & \text{if } n \geq 60, \end{cases} \quad (27)$$

- REM-type dropping function, [14]:

$$d_5(n) = \begin{cases} 0, & \text{if } n < 30, \\ 0.14950 - 0.14950e^{-(n-30)/10}, & \text{if } n \geq 30. \end{cases} \quad (28)$$

They represent the most popular dropping function classes proposed in the active queue management literature. Importantly, all these dropping functions were parameterized in [28] in such a way that they provide the average stationary response time of exactly 20.0 for $\rho = 1$. The average response time is arguably the single most important stationary performance measure of a queueing system, thus in some sense, these five dropping functions provide the same stationary behavior of the queue.

As we will see, it does not mean that the transient behavior is similar as well. Namely, in Table 2, the probability of not reaching a queue size of level 50 in 1000 seconds is shown for functions d_1 – d_5 . It is

accompanied by the average time to reach level 50. As we can see, the probability varies significantly, from 0.933 to 0.997, depending on the dropping function. The mean time to reach 50 varies even more—from about 8000 to over 200,000.

Figure 1. Dropping functions $d_1(n) - d_5(n)$.

Table 2. Probability of not reaching 50 in $(0,1000)$ and the average time to reach 50 for dropping functions $d_1 - d_5$.

Dropp. Fun.	$Y_{0,50}(1000)$	$\mathbb{E}H_{0,50}$
$d_1(n)$	0.96242	13,733
$d_2(n)$	0.96728	15,621
$d_3(n)$	0.93350	7973
$d_4(n)$	0.99746	200,534
$d_5(n)$	0.94178	10,002

The detailed dependence of the probability of not reaching 50 by the time t is depicted as a function of t in Figure 2, for three selected dropping functions. We see a very different transient performance in each case. For instance, probability 0.6 of not crossing 50 corresponds to the time interval of length about 8000 in the case of d_2 , of length about 4300 in the case of d_2 , and of length over 100,000 in the case of d_2 .

Figure 2. Probability of not reaching level 50 in interval $(0, t)$ as a function of t , for dropping functions d_3, d_2, d_4 , counting from the bottom.

In Figure 3, the average time to reach level M is depicted for the same three dropping functions, as a function of M . As we can see, for low M values, all three graphs are the same, which follows from the fact that all functions d_2 – d_4 vanish for small n . From M about 35, however, the graphs start to diverge, and the divergence becomes very fast for M over 42 (note that the scale on the vertical axis is logarithmic).

Figure 3. The average time to reach level M as a function of M , for dropping functions d_3 , d_2 , d_4 , counting from the bottom.

All of these considerations indicate clearly, that the transient behavior of the system, may differ by far for different dropping functions, even if the stationary average response time is the same for all of them.

In the final example, we show that we can design the dropping function in such a way that it meets some performance goals. For this purpose we can use, for instance, the following class of dropping functions:

$$d(p, n) = pd_5(n), \quad (29)$$

where p is a positive parameter. Sample functions from this class are shown in Figure 4.

Figure 4. Dropping function $d(p, n)$ for $p = 1, \dots, 6$, counting from the bottom.

Assume now that we have the following performance goals: the probability that the queue size does not reach 50 in one hour (3600s) has to be at least 0.99, while the overall packet loss ratio has to be as small as possible. (The loss ratio is defined, naturally, as the long-run fraction of dropped packets).

Using the bisection method and Theorem 1, we can easily obtain the value of p for which it holds $Y_{0.50}(3600) = 0.99$. It is $p = 2.237$. Moreover, for every n we have $d(p_1, n) \leq d(p_2, n)$ if $p_1 < p_2$. Thus it is obvious that the loss ratio must grow with p , when the class $d(p, n)$ of dropping functions is used.

Therefore, we conclude that the optimal dropping function in the class (29), with respect to the assumed performance goals, is $d(2.237, n)$.

5. Conclusions

We derived a formula for the probability that in a time interval of length t , the queue size is kept under a given level, in a queuing system with the dropping function. In addition, a formula for the average time to reach a given level was obtained. As the main reason to apply the dropping function mechanism is to maintain a low queue size, these are natural characteristics to study.

In numerical examples it was demonstrated that for a precise characterization of the system, stationary performance characteristics should be accompanied by the above-mentioned transient probabilities when the dropping function is applied. Namely, it was shown that different dropping functions may induce a very different transient behavior of the system, even if they provide the same stationary behavior (e.g., the average response time).

The study was motivated by active queue management in Internet routers. The results are presented, however, in universal terms of queueing theory, and are applicable in other areas as well.

Funding: This research was funded by National Science Centre, Poland, grant number 2017/25/B/ST6/00110.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Gettys, J.; Nichols, K. Bufferbloat: Dark buffers in the Internet. *Queue* **2011**, *9*, 40–54. [CrossRef]
2. Cerf, V.G. Bufferbloat and other Internet challenges. *IEEE Internet Comput.* **2014**, *18*, 79–80. [CrossRef]
3. Baker, F.; Fairhurst, G. (Eds.) *IETF Recommendations Regarding Active Queue Management*. Available online: <https://www.rfc-editor.org/pdf/rfc/rfc7567.txt.pdf> (accessed on 1 July 2020).
4. Nichols, K.; Jacobson, V. Controlling Queue Delay. *Queue* **2012**, *55*, 42–50.
5. Khoshnevisan, L.; Salmasi, F.R. A robust and high-performance queue management controller for large round trip time networks. *Int. J. Syst. Sci.* **2016**, *47*, 1–12. [CrossRef]
6. Wang, P.; Zhu, D.; Lu, X. Active queue management algorithm based on data-driven predictive control. *Telecommun. Syst.* **2017**, *64*, 103–111. [CrossRef]
7. Abbas, G.; Manzoor, S.; Hussain, M. A stateless fairness-driven active queue management scheme for efficient and fair bandwidth allocation in congested Internet routers. *Telecommun. Syst.* **2018**, *67*, 3–20. [CrossRef]
8. Li, F.; Sun, J.; Zukerman, M.; Liu, Z.; Xu, Q.; Chan, S.; Chen, G.; Ko, K.T. A comparative simulation study of TCP/AQM systems for evaluating the potential of neuron-based AQM schemes. *J. Netw. Comput. Appl.* **2014**, *41*, 274–299. [CrossRef]
9. Chen, J.V.; Chen, F.-C.; Tarn, J.M.; Yen, D.C. Improving network congestion: A RED-based fuzzy PID approach. *Comput. Stand. Interfaces* **2012**, *34*, 426–438. [CrossRef]
10. Chebli, S.; Elakkary, A. Multi-Objective Genetic Algorithm Optimization Using PID Controller for AQM/TCP Networks. *Int. Rev. Autom. Control* **2017**, *10*, 33–39. [CrossRef]
11. Chrost, L.; Chydzinski, A. On the deterministic approach to active queue management. *Telecommun. Syst.* **2016**, *63*, 27–44. [CrossRef]
12. Floyd, S.; Jacobson, V. Random early detection gateways for congestion avoidance. *IEEE/ACM Trans. Netw.* **1993**, *1*, 397–413. [CrossRef]
13. Rosolen, V.; Bonaventure, O.; Leduc, G. A RED discard strategy for ATM networks and its performance evaluation with TCP/IP traffic. *ACM SIGCOMM Comput. Commun. Rev.* **1999**, *29*, 23–43. [CrossRef]
14. Athuraliya, S.; Li, V.H.; Low, S.H.; Yin, Q. REM: Active queue management. *IEEE Netw.* **2001**, *15*, 48–53. [CrossRef]

15. Zhou, K.; Yeung, K.L.; Li, V. Nonlinear RED: A simple yet efficient active queue management scheme. *Comput. Netw.* **2006**, *50*, 3784. [\[CrossRef\]](#)
16. Augustyn, D.R.; Domanski, A.; Domanska, J. A choice of optimal packet dropping function for active queue management. *Commun. Comput. Inf. Sci.* **2010**, *7*, 199–206.
17. Feng, C.; Huang, L.; Xu, C.; Chang, Y. Congestion Control Scheme Performance Analysis Based on Nonlinear RED. *IEEE Syst. J.* **2017**, *11*, 2247–2254. [\[CrossRef\]](#)
18. Bonald, T.; May, M.; Bolot, J.-C. Analytic evaluation of RED performance. In Proceedings of the IEEE INFOCOM 2000, Tel Aviv, Israel, 26–30 March 2000; pp. 1415–1424.
19. Hao, W.; Wei, Y. An Extended $G^X/M/1/N$ Queueing Model for Evaluating the Performance of AQM Algorithms with Aggregate Traffic. *Lect. Notes Comput. Sci.* **2005**, *3619*, 395–414.
20. Kempa, W. On main characteristics of the $M/M/1/N$ queue with single and batch arrivals and the queue size controlled by AQM algorithms. *Kybernetika* **2011**, *47*, 930–943.
21. Chydzinski, A.; Chrost, L. Analysis of AQM queues with queue-size based packet dropping. *Appl. Math. Comput. Sci.* **2011**, *21*, 567–577. [\[CrossRef\]](#)
22. Kempa, W. A direct approach to transient queue-size distribution in a finite-buffer queue with AQM. *Appl. Math. Inf. Sci.* **2013**, *7*, 909–915. [\[CrossRef\]](#)
23. Tikhonenko, O.; Kempa, W. Performance evaluation of an $M/G/N$ -type queue with bounded capacity and packet dropping. *Appl. Math. Comput. Sci.* **2016**, *26*, 841–854. [\[CrossRef\]](#)
24. Mrozowski, P.; Chydzinski, A. Queues with dropping functions and autocorrelated arrivals. *Methodol. Comput. Appl. Probab.* **2018**, *20*, 97–115. [\[CrossRef\]](#)
25. Chydzinski, A.; Mrozowski, P. Queues with dropping functions and general arrival processes. *PLoS ONE* **2016**, *11*, e0150702. [\[CrossRef\]](#)
26. Tikhonenko, O.; Kempa, W. Erlang service system with limited memory space under control of AQM mechanism. *Commun. Comput. Inf. Sci.* **2017**, *718*, 366–379.
27. Chydzinski, A. Queues with the dropping function and non-Poisson arrivals. *IEEE Access* **2020**, *8*, 39819–39829. [\[CrossRef\]](#)
28. Chydzinski, A.; Adamczyk, B. Response time of the queue with the dropping function. *Appl. Math. Comput.* **2020**, *377*, 125164. [\[CrossRef\]](#)
29. Machihara, F. First passage times of $PH/PH/1/K$ and $PH/PH/1$ queues. *J. Oper. Res. Soc. Jpn.* **1987**, *30*, 1–26. [\[CrossRef\]](#)
30. Ross, S.M.; Seshadri, S. Hitting time in an $M/G/1$ queue. *J. Appl. Probability* **1999**, *36*, 934–940. [\[CrossRef\]](#)
31. Asmussen, S.; Jobmann, M.; Schwefel, H.-P. Exact buffer overflow calculations for queues via martingales. *Queueing Syst.* **2002**, *42*, 63–90. [\[CrossRef\]](#)
32. Chydzinski, A. Time to Buffer Overflow in an MMPP Queue. *Lect. Notes Comput. Sci.* **2007**, *4479*, 879–889.
33. Kempa, W.; Marjasz, R. Distribution of the time to buffer overflow in the $M/G/1/N$ -type queueing model with batch arrivals and multiple vacation policy. *J. Oper. Res. Soc.* **2020**, *71*, 447–455. [\[CrossRef\]](#)
34. Bratiychuk, M.; Chydzinski, A. On the loss process in a batch arrival queue. *Appl. Math. Model.* **2009**, *33*, 3565–3577. [\[CrossRef\]](#)
35. Chydzinski, A.; Wojcicki, R.; Hryn, G. On the number of losses in an MMPP queue. *Lect. Notes Comput. Sci.* **2007**, *4712*, 38–48.
36. Zakian, V. Numerical Inversion of Laplace Transform. *Electron. Lett.* **1969**, *5*, 120–121. [\[CrossRef\]](#)