Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.12785/amis/080601

Analysis of a Scheduler for Virtualization of Links with Performance Isolation

Andrzej Chydzinski[*] *and Blazej Adamczyk*

Silesian University of Technology, Institute of Informatics, Akademicka 16, 44-100 Gliwice, Poland

Received: 3 Oct. 2013, Revised: 1 Jan. 2014, Accepted: 2 Jan. 2014
Published online: 1 Nov. 2014

Abstract: In this paper we deal with a scheduling algorithm that creates several virtual links on one physical link by switching the service between the virtual links. The switching is done in such a way that the physical link is available for each virtual link for a constant time, what provides full performance isolation between virtual links. Firstly, we present a detailed analysis of the scheduler, including formulas for the queue sizes associated with virtual links and their packet loss ratios. Then we show numerical examples of performance characteristics of created virtual links. We also compare them with adequate non-virtual links in order to demonstrate what is the cost of virtualization.

Keywords: virtualization of links, scheduler, queue size distribution, performance isolation

1 Introduction

Virtualization of networking resources is considered as one of the foundations of future networking architectures, [1]. This is due to the fact that it offers the possibility that several overlay networks can use the same underlying physical infrastructure (physical links and nodes), being unaware of it. Therefore, it is easier to deploy new architectures, protocols and services using virtual, instead of physical, resources. The new architectures, protocols and services are in great demand, especially when thinking of the Internet, which suffers from important deficiencies, like no end-to-end quality of service, very complex management, poor extensibility of the IP protocol stack, to name a few (see [2] for more).

Due to the aforementioned reasons, the virtualization of networking resources has been considered as an important component of the Future Internet architecture in many scientific projects in the world, e.g. FIA MANA [3], AKARI [4], GENI [5], IIP [6], and others.

In this paper we are interested in virtualization of links. The basic idea of virtualization of links is depicted in Fig. 1. The streams of packets from several different virtual interfaces (vifs) are scheduled before entering the physical network interface (eth0). In this way, the physical link is shared between all virtual interfaces. In this scheme the most important part is the scheduler, which is responsible for the performance of the virtual links[1].

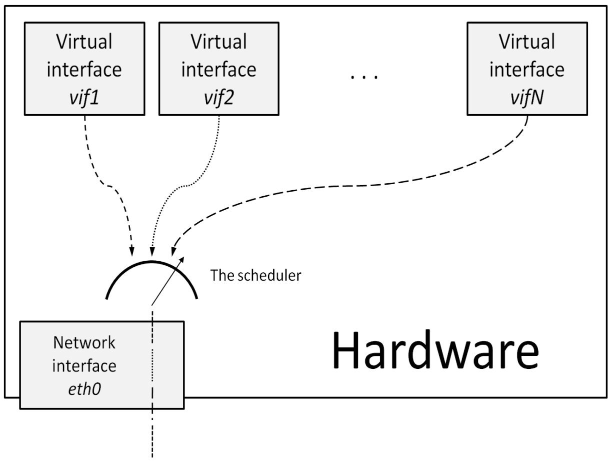


Fig. 1: The idea of virtualization of links.

[1] To obtain fully functional virtual links, a mechanism for classification of packets from different virtual links (e.g. via additional header) is also needed. It is not difficult to design and goes beyond the scope of this paper.

[*] Corresponding author e-mail: Andrzej.Chydzinski@polsl.pl

There are several possible requirements that may be imposed on the scheduler. For instance, we may demand limited queueing delay in (one or many) virtual links, guaranteed bandwidth of (one or many) virtual links, or optimal usage of the physical output link etc. However, for the purpose of future networking architectures, which include quality of service guarantees, the most important requirement is a full performance isolation. The performance isolation means that the volume and type of traffic offered to one of the virtual links has absolutely no impact on performance of other virtual links.

The performance isolation requirement is fulfilled by the following scheduler, proposed in [7] and implemented in a large experimental Future Internet architecture called the IIP System, [8], using the devices described in [9]. The scheduler works as follows. There are N separate buffers, one per each of N virtual links. After leaving the virtual interface (vif) a packet is stored in the buffer belonging to the appropriate virtual link. Therefore, in each buffer a queue of packets is formed. All these queues are served by the physical link in a cyclic manner. Namely, the packets from the first queue are transmitted for constant time W_1, then the packets from the second queue are transmitted for constant time W_2 etc. After the last, N-th queue, which is served for W_N time, the first queue is served again for time W_1 and the cycle repeats. At the end of each transmission phase (say W_i) one packet may not be transmitted before the end of the phase, due to lack of time (splitting packets is not allowed). This packet has to wait in its buffer for the next cycle and the physical link is idle until the end of the phase. Therefore we may observe some wastage of bandwidth of the physical link at the end of each transmission phase. Moreover, even if there is no traffic on one virtual link, say j-th, the scheduler has to wait for the appropriate time, W_j, without allowing any other virtual link to use the physical link.

We see at first glance, that this scheduler ensures full performance isolation between the virtual links. However, it is also non-work conserving, which means that sometimes the output link is idle even if there are packets waiting for transmission.

The presented scheduler has several properties that make its analysis and parameterization complicated. First of all, we have to deal with the aforementioned bandwidth wastage. For instance, let us consider the scheduler for creating two virtual links at a $1Gb/s$ physical link and set $W_1 = W_2 = 30\mu s$. Assume that on the first virtual link only 500-bytes-long packets will be transmitted, while on the second virtual link there will be 40% of 100-bytes-long packets and 60% of 1000-bytes-long packets[2]. Assume that the total traffic offered to each virtual link is $500Mb/s$. As we can guess, the achieved bitrate of each virtual link may be smaller than $500Mb/s$, due to the packets that could not be transmitted at the end of each work phase. But how much smaller in each case?

[2] Transmission of packets of sizes $100B$, $500B$ and $1000B$ take $0.8\mu s$, $4\mu s$ and $8\mu s$, respectively.

The problem of bandwidth wastage can be reduced by extending lengths of phases W_1 and W_2. However, such solution comes with the cost of large buffers that have to store the packets during the vacation phase and large queues building-up in these buffers, which cause a large delay before the actual transmission. On the other hand, short buffers would reduce queue sizes but at the cost of high number of losses.

Naturally, the presented trade-offs cannot be solved effectively without a precise analysis of the appropriate queueing model. The model, in addition to typical queueing parameters (buffer sizes, arrival and service rates), has to include the packet size distribution on each virtual link. Carrying out an analysis of such a model is the main purpose of this paper.

The full performance isolation causes that each queue in the discussed scheduler can be analyzed separately, as a model of a queueing system with vacations. In particular, a j-th virtual link consists of a queueing system with alternating work phases of length W_j and vacation phases of length $\sum_{i\neq j} W_i$.

The literature on vacation queues is vast. For long literature lists we may refer the reader either to monographs [10, 11] or to survey papers [12, 13]. The vacation queueing models considered in the literature differ in several important features, including:

- the arrival process model, e.g. [14, 15, 16],
- the service process model, e.g. [17, 14, 18],
- the capacity of buffers (finite or infinite), e.g. [15, 16],
- the service discipline (the number of jobs served during one work period).

Several different service disciplines are considered, for instance: exhaustive [19], gated [20], number-limited [21], or time-limited [22, 23, 24, 25]. From our point of view, the most interesting is the time-limited discipline, in which the server cannot attend a queue for longer than some time (called the maximum server attendance, MSA, time). However, the previously studied models with time-limited disciplines differ significantly from the model studied herein. Typically, it is assumed that the work phase ends immediately after the queue becomes empty, even if the MSA time has not expired yet, [22, 23, 24, 25]. Moreover, the preemptive-resume policy is often assumed for the last packet served in the work phase. This policy means that the service of the last packet can be preempted and resumed from the interrupted point in the next cycle, [22, 23, 24]. In addition to that, in [25] the MSA time is not a constant, but an exponentially distributed random variable.

All the aforementioned differences cause that the model studied herein has fundamentally different properties and requires a different analytical approach.

To the best of the authors knowledge, the results presented in this paper are new. The only paper we know of, which is devoted to the discussed scheduler, is [26]. It presents an interesting approach to buffer dimensioning and analysis of the mean waiting times. However, the formulas presented there are approximate and the model is simplified. Namely, a constant packet size is assumed in [26] and W_i is assumed as multiple of the constant transmission time. In this paper we present rigorous analysis with strict results and without any simplifications of the model. In fact, as it will be demonstrated, both the distribution of the packet size and the length of the work phase, which can be arbitrary chosen here, may have a deep impact on the system performance.

As regards the method used herein, an analysis of two-dimensional Markov chain, which stores the queue size and the size of the packet being transmitted, will be used as a framework. In addition to that, it will be required to compute two transient characteristics of the classic, single-server queue with finite buffer, namely the joint distribution of the queue size and the size of the packet being transmitted at time t, as well as the average number of packets lost in interval $(0,t)$. For both of these characteristics the method of [27] will be used. It is based on the Laplace transform technique combined with an application of special recurrent sequences for solving explicitly the set of equations for the characteristic of interest.

The remaining part of the paper is structured in the following way. In Section 2, a formal description of the model of the scheduler is presented. In Section 3, the actual analysis of the model is carried out. It is divided in finding the queue size distribution (subsection 3.2) and finding the packet loss ratio (subsection 3.3). The next two Sections, 4 and 5, are devoted to detailed calculations of two queueing characteristics needed to obtain the numerical results for the queue size and loss ratio. Then, in Section 6, numerical results are presented. They include studies on the dependence of the virtual link performance on the packet size distribution, the length of the work phase and the buffer size. In addition, in each case the performance of the virtual link is compared with the performance of an analogous non-virtual link, so that it could be seen what the cost payed for the virtualization is.

2 Model description

The model of the scheduler is depicted in Fig. 2. Namely, there are N separate Poisson arrival streams with rates $\lambda_1,\ldots,\lambda_N$, respectively. Each arrival stream has, besides the packet arrival rate λ_i, its own packet size distribution, D_i. There are N buffers of sizes $b_1,\ldots,b_N$ packets. In these buffers the packets from the arrival streams are stored, forming N separate queues. If upon a packet arrival the appropriate buffer is full, the packet is rejected and lost.

There is also an output link of capacity C b/s. All the queues are served by the output link in a cyclic manner, such that each queue is given a constant service time. In particular, the first queue is served for W_1 seconds, the second queue is served for W_2 seconds, etc. The last, N-th queue is served for W_N seconds. Then the cycle repeats, the first queue is served again for W_1 seconds, and so on.

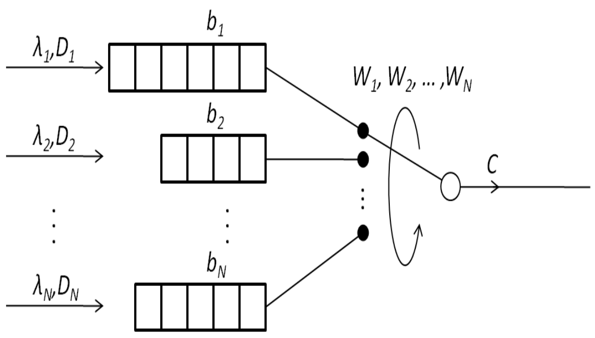


Fig. 2: The model of the scheduler.

By service of a packet we mean its transmission through the output link. Naturally, the service time of a packet depends on the packet size and is equal to d_j/C for a packet of size d_j. If the transmission of a packet is interrupted by the end of phase W_i, the whole packet remains in the buffer and the transmission of the whole packet is repeated in the next cycle. Such discipline is called *preemptive repeat-identical (PRI) discipline* (see [25]). Alternatively, we may prevent the transmission of a packet if there is not enough time by the end of the transmission phase, to complete it. This approach, more practical in real systems, makes no difference in the performance of the scheduler – both models are equivalent.

Within each queue the packets are stored and served in the arrival order (i.e. according to FIFO discipline).

As in the presented model we allow different packet sizes, one might ask why the buffers are dimensioned in packets, not in bytes. The reason is that in real networking devices the buffers are commonly dimensioned in packets. For more details, we refer the reader to [28]. In particular, it is explained there, why it is easier to implement the packet-based buffer policy than the byte-based one.

At the first glance, the presented queueing model (excluding vacations) might be perceived either as the $M/G/1/b$ queue, in which the customer (packet) size translates to the variable service time, or the $M^X/D/1/b$ queue, with constant service time and understanding ”packet” as ”batch of customers”. However, as splitting packets is prohibited and the buffers are dimensioned in packets, the more proper is the $M/G/1/b$ model - it will be used for finding some special characteristics in Sections 4 and 5.

3 Analysis

As it was already stated, the presented scheduler provides full performance isolation, so each queue can be analyzed separately. Therefore, in this section we will perform a detailed analysis of a queue associated with one virtual link (say j-th virtual link). For simplicity, we will be omitting index j from now on, so for this particular queue we have the following parameters:

C – the capacity of the physical link, in bits/s,
W – the duration of the work (transmission) phase,
V – the duration of the vacation phase, $V = \sum_{i\neq j} W_i$,
b – the buffer size, in packets,
λ – the rate of the arriving Poisson process,
D – the distribution of packet sizes in the arrival process.

In particular, D is given by M pairs:

$$D: \ (d_1, p_1), \ldots, (d_M, p_M), \qquad \sum_{i=1}^{M} p_i = 1, \qquad (1)$$

where d_i denotes a packet size, while p_i denotes the probability of size d_i. The average packet size is equal to

$$\overline{d} = \sum_{i=1}^{M} d_i p_i.$$

We have to make one natural assumption, namely

$$\max\{d_i/C : i = 1, \ldots, M\} \leq W,$$

which means that the service phase, W, is long enough for completing the transmission of the largest possible packet. Otherwise, the first occurrence of such packet would result in a total lock out of the virtual link.

The load offered to the queue is defined as:

$$\rho = \lambda \, \overline{d} \, \frac{V+W}{CW}.$$

From now on, by $X(t)$ we will denote the queue size at time t, including the service position. It is assumed that the service position is the first position in the buffer. As the buffer can store no more than b packets, we have $0 \leq X(t) \leq b$.

3.1 Auxiliary definitions

Before we start the actual analysis, note that during the work phase we in fact deal with the classic single-server queueing system with a finite buffer. For the analysis of the whole vacation model, we will need two important characteristics of this classic system, namely the joint distribution of the queue size and the size of the packet being served, as well as the average number of packets lost in the interval of length t.

To properly define these characteristics, in this subsection we assume that we have the classic queueing system with finite buffer for b packets and Poisson arrivals of rate λ. The packets arriving at the queue may have different sizes, with distribution given in (1), and they are served by the link of capacity C, without any vacations.

Let $S(t)$ denote the number of the size of the packet being served at time t. For instance, $S(10) = 3$ means that at time $t = 10$ a packet of size d_3 is being served. $S(t) = 0$ means that at time t the queue is empty.

Assume that $X(0) = n$, $0 \leq n \leq b$, and the sizes of packets present in the buffer at $t = 0$ are distributed according to (1).

By $\Theta_n(t,m,j)$ we will denote the probability that at time t the queue size is equal to m and a packet of size d_j is being transmitted. Namely, we have

$$\Theta_n(t,m,j) = \mathbb{P}\{X(t) = m, S(t) = j | X(0) = n\}.$$

Now, let $L(a,b)$ denote the number of packets lost in time interval (a,b) and $\Delta_n(t)$ be the average value of $L(0,t)$ provided $X(0) = n$, i. e.:

$$\Delta_n(t) = \mathbb{E}\{L(0,t) | X(0) = n\}. \qquad (2)$$

Both characteristics $\Theta_n(t,m,j)$ and $\Delta_n(t)$ will play important roles in the analysis of vacation queues associated with virtual links. They will be computed in Sections 4 and 5, respectively.

3.2 Queue size

Now we get back to the analysis of the vacation queue associated with one virtual link, defined at the beginning of Section 3. We will assume that the cycle starts with the vacation period ($t = 0$ is the beginning of the first vacation period). Let α_k be the beginning of k-th vacation period, i.e.:

$$\alpha_k = (k-1)(V+W), \quad k \geq 1.$$

When considering the queue size distribution, it is obvious at first glance that the stationary distribution of the queue size in the classic sense, i.e.:

$$\lim_{t\to\infty} \mathbb{P}\{X(t) = m\}, \quad m = 0, \ldots, b, \qquad (3)$$

does not exist. This is connected with the cyclic character of the model, with constant cycle duration. The queue size at the beginning of the vacation period, α_k, is significantly smaller than at the beginning of the work period, $\alpha_k + V$, no matter how large is k. This is due to the fact that during the vacation period we have only arrivals and no service. As a consequence

$$\lim_{k\to\infty} \mathbb{P}\{X(\alpha_k) = m\} \neq \lim_{k\to\infty} \mathbb{P}\{X(\alpha_k + V) = m\},$$

which prohibits the existence of limit (3).

Therefore, instead of (3), we will deal with the distribution of the queue size at the beginning of the vacation phase, namely

$$\bar{q}_m = \lim_{k\to\infty} \mathbb{P}\{X(\alpha_k) = m\}, \quad m = 0,\dots,b. \tag{4}$$

Now, let us denote

$$X_k = X(\alpha_k), \quad S_k = S(\alpha_k), \quad k \geq 1.$$

In particular, S_k is the number of the size of the first packet that could not be transmitted in the k-th work period, thus had to wait in the buffer for the next work period. $S_k = 0$ means that after the k-th work period the buffer was empty.

It is easy to see that pair (X_k, S_k) constitutes a two-dimensional Markov chain. Due to the fact that $S_k = 0$ if and only if $X_k = 0$, there are no states in the form $(0,i)$, $i > 0$, nor $(n,0)$, $n > 0$, and the chain has the following state space:

$$\Omega = \{(m,j) : m = 1,\dots,b, \quad j = 1,\dots,M\} \cup (0,0).$$

Our first goal will be finding the transition probabilities of this chain, i.e.:

$$Q_{n,i,m,j} = \mathbb{P}\{X_{k+1} = m, S_{k+1} = j | X_k = n, S_k = i\},$$

for $(n,i),(m,j) \in \Omega$.

To accomplish that, let us introduce the following sequence of moments in time:

$$\beta_k = \begin{cases} \alpha_k + V + d_i/C, & \text{if} \quad S_k = i, i \neq 0, \\ \alpha_k + V, & \text{if} \quad S_k = 0. \end{cases} \tag{5}$$

Namely, β_k is the moment of transmission completion of the first packet that reminded in the buffer from the previous work phase. If at the end of the previous work phase the buffer was empty, then β_k is just the beginning of the work phase.

Now, in each time interval (α_k, β_k) there are only Poisson arrivals and no transmission completion. Therefore, it is easy to compute the change of the queue size in this interval – the queue can only grow or stay unaltered. Namely, denoting

$$U_{n,i,l} = \mathbb{P}\{X(\beta_k-) = l | X_k = n, S_k = i\}, \ 0 \leq l \leq b, \ (n,i) \in \Omega, \tag{6}$$

and using Poisson distribution we have:

$$U_{n,i,l} = \begin{cases} \frac{e^{-\lambda V}(\lambda V)^l}{l!}, & \text{if} \quad n = 0, i = 0, n \leq l < b, \\ \sum_{j=b}^{\infty} \frac{e^{-\lambda V}(\lambda V)^j}{j!}, & \text{if} \quad n = 0, i = 0, l = b, \\ \frac{e^{-\lambda(V+d_i/C)}[\lambda(V+d_i/C)]^{l-n}}{(l-n)!}, & \text{if } n > 0, i > 0, n \leq l < b, \\ \sum_{j=b-n}^{\infty} \frac{e^{-\lambda(V+d_i/C)}[\lambda(V+d_i/C)]^j}{j!}, & \\ & \text{if} \quad n > 0, i > 0, l = b, \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

Using $U_{n,i,l}$ and functions $\Theta_n(t,m,j)$ we can calculate the transition probabilities $Q_{n,i,m,j}$. In particular, applying the total probability formula with respect to all possible queue sizes at time β_k we get the following theorem.

Theorem 1. *Transition probabilities for Markov chain* (X_k, S_k) *have the form:*

$$Q_{n,i,m,j} = \begin{cases} \sum_{l=0}^{b} U_{0,0,l}\Theta_l(W,m,j), & \textit{if} \quad n = 0, i = 0, \\ \sum_{l=n}^{b} U_{n,i,l}\Theta_{l-1}(W - d_i/C, m, j), & \\ & \textit{if} \quad n > 0, i > 0, \end{cases} \tag{8}$$

where $U_{n,i,l}$ *is given by* (7) *and* $\Theta_n(t,m,j)$ *is defined in subsection* 3.1.

As chain (X_k, S_k) is ergodic, we can calculate its stationary distribution

$$q_{n,i} = \lim_{k\to\infty} \mathbb{P}\{X_k = n, S_k = i\}, \quad (n,i) \in \Omega, \tag{9}$$

using the standard set of linear equations, i.e:

$$\sum_{n=1}^{b}\sum_{i=1}^{M} q_{n,i}Q_{n,i,m,j} + q_{0,0}Q_{0,0,m,j} = q_{m,j}, \tag{10}$$

where $1 \leq m \leq b$, $1 \leq j \leq M$, and

$$\sum_{n=1}^{b}\sum_{i=1}^{M} q_{n,i} + q_{0,0} = 1. \tag{11}$$

Then, using distribution $q_{n,i}$ we can obtain the stationary distribution of the queue size at the beginning of the vacation period:

$$\bar{q}_n = \lim_{k\to\infty} \mathbb{P}\{X_k = n\} = \begin{cases} q_{0,0}, & \text{if} \quad n = 0, \\ \sum_{i=1}^{M} q_{n,i}, & \text{if} \quad 0 < n \leq b, \end{cases} \tag{12}$$

and its average value:

$$\mathbb{E}(\bar{q}) = \sum_{n=0}^{b} n\bar{q}_n. \tag{13}$$

Similarly, we can obtain the stationary distribution of the size of the packet that was not transmitted at the end of the work phase and has to wait for the next work phase:

$$\tilde{q}_i = \lim_{k\to\infty} \mathbb{P}\{S_k = i\} = \begin{cases} q_{0,0}, & \text{if} \quad i = 0, \\ \sum_{n=1}^{b} q_{n,i}, & \text{if} \quad 0 < i \leq M. \end{cases} \tag{14}$$

According to our intuition, the latter distribution should differ significantly from the general packet size distribution in the arrival stream, D. Roughly speaking, large packets should more often, than small packets, remain in the buffer for the next work phase. This supposition is true indeed. It will be illustrated via numerical example presented in subsection 6.1.

Finally, note that all the formulas presented in this section can be easily used in numerical computations. Of course, the infinite sums appearing in (7) can be replaced by finite sums, e.g.:

$$\sum_{j=b}^{\infty}\frac{e^{-\lambda V}(\lambda V)^j}{j!}=1-\sum_{j=0}^{b-1}\frac{e^{-\lambda V}(\lambda V)^j}{j!}.$$

In fact, the most demanding part is computing probabilities $\Theta_n(t,m,j)$. All the necessary details will be presented in Section 4.

3.3 Loss ratio

In this section we will find the loss ratio, LR, of one virtual link. The loss ratio is defined as the long-run fraction of packets lost due to the buffer overflow.

We will start with calculations of the probability that in the time interval (α_k,β_k) there will be j losses and just after time β_k the queue size will be m. Namely, by $L(\alpha_k,\beta_k)$ we denote the number of packets lost in time interval (α_k,β_k) and define

$$Y_{n,i,m}(j)=\mathbb{P}\{L(\alpha_k,\beta_k)=j,X(\beta_k+)=m|X_k=n,S_k=i\},$$

where $(n,i)\in\Omega$, $m=0,\ldots,b$ and $j=0,1,\ldots$.

Using Poisson distribution we get

$$Y_{n,i,m}(j)=\begin{cases}\frac{e^{-\lambda V}(\lambda V)^m}{m!}, & \text{if}\quad n=0,i=0,0\le m\le b,j=0,\\ \frac{e^{-\lambda V}(\lambda V)^{b+j}}{(b+j)!}, & \text{if}\quad n=0,i=0,m=b,j>0,\\ \frac{e^{-\lambda(V+d_i/C)}[\lambda(V+d_i/C)]^{m-n+1}}{(m-n+1)!}, & \text{if}\quad n>0,i>0,n-1\le m\le b-1,j=0,\\ \frac{e^{-\lambda(V+d_i/C)}[\lambda(V+d_i/C)]^{b-n+j}}{(b-n+j)!}, & \text{if}\quad n>0,i>0,m=b-1,j>0,\\ 0, & \text{otherwise.}\end{cases}\tag{15}$$

Now, using the stationary distribution of chain (X_k,S_k), probabilities $Y_{n,i,m}(j)$ and function $\Delta_n(t)$, we can easily obtain the final formula.

Theorem 2. *The loss ratio for one virtual link is equal to*

$$LR=\frac{1}{\lambda(V+W)}\Bigg[\sum_{n=1}^{b}\sum_{i=1}^{M}\sum_{j=0}^{\infty}\sum_{m=0}^{b}q_{n,i}Y_{n,i,m}(j)\big(j+\Delta_m(W-d_i/C)\big)+\sum_{j=0}^{\infty}\sum_{m=0}^{b}q_{0,0}Y_{0,0,m}(j)\big(j+\Delta_m(W)\big)\Bigg],\tag{16}$$

where $Y_{n,i,m}(j)$ is given by (15) and $\Delta_n(t)$ is defined in subsection 3.1.

Again, note that formulas (15) and (16) can be used in numerical computations and the most demanding part is computing functions $\Delta_n(t)$. All the necessary details will be presented in Section 5.

Finally, note that the packet loss ratio (the fraction of packet lost) is equal to the data loss ratio (i.e. the overall loss ratio taking into account packet sizes). This is because the rejection or acceptance of a packet upon its arrival does not depend on the packet size. Therefore lost packets have the same size distribution as accepted packets.

4 Calculations of $\Theta_n(t,m,j)$

In this section we deal with the classic queueing system with finite buffer for b packets and Poisson arrivals of rate λ. The packets arriving at the queue may have different sizes with distribution (1) and they are served by the link of capacity C, with no vacations. Assume that $X(0)=n$, $0\le n\le b$, and sizes of packets present in the buffer at $t=0$ are distributed according to (1).

We are interested in the joint distribution of the queue size and the size of the packet being transmitted at time t, i.e.:

$$\Theta_n(t,m,j)=\mathbb{P}\{X(t)=m,S(t)=j|X(0)=n\}.$$

To find this distribution (and the distribution of $\Delta_n(t)$ in the next section) we will use the method systematized in [27]. The method is based on the Laplace transform technique combined with an application of recurrent sequences in form of (26). It can be used for solving various finite-buffer queueing systems with Markovian arrivals, including the classic systems (e.g. $M/G/1/N$, $M^X/G/1/N$, $PH/G/1/N$, $BMAP/G/1/N$) and some special models (e.g. [29,30,31]).

Let us assume first that the buffer is not empty at the beginning, i.e. $X(0)=n>0$. Using the total probability law with respect to the size of the first packet in the buffer we obtain the following equations:

$$\begin{aligned}\Theta_n(t,m,j)=&\sum_{i=1}^{M}p_iI(d_i/C<t)\\ &\cdot\Bigg[\sum_{k=0}^{b-n-1}\Theta_{n+k-1}(t-d_i/C,m,j)\frac{e^{-\lambda d_i/C}(\lambda d_i/C)^k}{k!}\\ &+\sum_{k=b-n}^{\infty}\Theta_{b-1}(t-d_i/C,m,j)\frac{e^{-\lambda d_i/C}(\lambda d_i/C)^k}{k!}\Bigg]\\ &+\zeta_n(t,m,j),\qquad 0<n\le b,\end{aligned}\tag{17}$$

where

$$\zeta_n(t,m,j)=I(j>0)p_jI(d_j/C\ge t)\cdot\begin{cases}0, & \text{if}\quad m<n,\\ \frac{e^{-\lambda t}(\lambda t)^{m-n}}{(m-n)!}, & \text{if}\quad n\le m<b,\\ \sum_{k=b-n}^{\infty}\frac{e^{-\lambda t}(\lambda t)^k}{k!}, & \text{if}\quad m=b,\end{cases}$$

and $I(\cdot)$ is defined as

$$I(A) = \begin{cases} 1, & \text{if } A \text{ is true,} \\ 0, & \text{otherwise.} \end{cases}$$

The first component of (17) corresponds to the case when the transmission of the first packet takes less than t time and during the transmission of this packet the buffer is not overflowed. The second component corresponds to the case when transmission of the first packet takes less than t time, but during the transmission of this packet the buffer gets overflowed. Finally, the third component of (17) corresponds to the case when the service of the first packet takes more than t time. As we have $X(0) = n > 0$, only $j > 0$ is possible in this case.

If we assume that the buffer is initially empty and use the total probability law with respect to the first arrival time, we get the following equation:

$$\Theta_0(t,m,j) = \int_0^t \Theta_1(t-u,m,j)\lambda e^{-\lambda u}du + e^{-\lambda t}I(m=0,j=0). \tag{18}$$

Naturally, the first component of (18) corresponds to the case when the first arrival time, u, is before t. The second component corresponds to the case when the first arrival time is after t.

Now we want to solve the system of equations (17) and (18). Denoting

$$\theta_n(s,m,j) = \int_0^\infty e^{-st}\Theta_n(t,m,j)dt$$

and applying the Laplace transform to (17) and (18) we obtain:

$$\begin{aligned}\theta_n(s,m,j) = &\sum_{k=0}^{b-n-1} a_k(s)\theta_{n+k-1}(s,m,j) \\ &+ \sum_{k=b-n}^{\infty} a_k(s)\theta_{b-1}(s,m,j) \\ &+ z_n(s,m,j), \qquad 0 < n \le b, \end{aligned} \tag{19}$$

$$\theta_0(s,m,j) = \frac{\lambda}{s+\lambda}\theta_1(s,m,j) + \frac{I(m=0,j=0)}{s+\lambda}, \tag{20}$$

where

$$z_n(s,m,j) = I(j>0)p_j \cdot \begin{cases} 0, & \text{if } \quad m<n, \\ g_{m-n}(s,j), & \text{if } n \le m < b, \\ (1-e^{-d_js/C})/s - \sum_{i=0}^{b-n-1} g_i(s,j), & \text{if } \quad m = b, \end{cases}$$

$$a_k(s) = \sum_{i=1}^{M} p_i e^{-(\lambda+s)d_i/C}(\lambda d_i/C)^k/k!, \tag{21}$$

$$g_k(s,i) = \frac{1}{k!}\lambda^k(s+\lambda)^{-1-k}\left[\Gamma(k+1) - \Gamma(k+1,(s+\lambda)d_i/C)\right], \tag{22}$$

$\Gamma(k)$ denotes the gamma function and $\Gamma(k,t)$ denotes the incomplete gamma function. Using the substitution

$$\tilde{\theta}_n(s,m,j) = \theta_{b-n}(s,m,j),$$

(19) and (20) yields:

$$\sum_{k=-1}^{n} a_{k+1}(s)\tilde{\theta}_{n-k}(s,m,j) - \tilde{\theta}_n(s,m,j) = \psi_n(s,m,j), \qquad 0 \le n < b, \tag{23}$$

$$\tilde{\theta}_b(s,m,j) = \frac{\lambda}{s+\lambda}\tilde{\theta}_{b-1}(s,m,j) + \frac{I(m=0,j=0)}{s+\lambda}, \tag{24}$$

where

$$\psi_n(s,m,j) = a_{n+1}(s)\tilde{\theta}_0(s,m,j) - \sum_{k=n+1}^{\infty} a_k(s)\tilde{\theta}_1(s,m,j) - z_{b-n}(s,m,j).$$

Using Lemma 3.2.1 of [32], we obtain the general solution of (23) in the form

$$\tilde{\theta}_n(s,m,j) = c(s,m,j)R_{n+1}(s) + \sum_{k=0}^{n} R_{n-k}(s)\psi_k(s,m,j), \quad n \ge 0, \tag{25}$$

where $c(s,m,j)$ is an unknown function which does not depend on n and

$$R_0(s) = 0, \quad R_1(s) = \frac{1}{a_0(s)},$$

$$R_k(s) = R_1(s)\left(R_{k-1}(s) - \sum_{i=0}^{k-1} R_{k-1-i}(s)a_{i+1}(s)\right), \quad k > 1. \tag{26}$$

Now we are left with the task of finding unknowns $c(s,m,j)$ and $\tilde{\theta}_0(s,m,j)$, $\tilde{\theta}_1(s,m,j)$ which occur in $\psi_n(s,m,j)$. From (25), for $n=0$ we get

$$c(s,m,j) = a_0(s)\tilde{\theta}_0(s,m,j). \tag{27}$$

From (23), for $n=0$ we get

$$\tilde{\theta}_1(s,m,j) = \left(\tilde{\theta}_0(s,m,j) - z_b(s,m,j)\right)/f(s), \tag{28}$$

where

$$f(s) = \sum_{k=0}^{\infty} a_k(s) = \sum_{i=1}^{M} p_i e^{-sd_i/C}.$$

Substituting (27) and (28) into (25) we obtain

$$\begin{aligned}\tilde{\theta}_n(s,m,j) =&\tilde{\theta}_0(s,m,j)\Bigg[R_{n+1}(s)a_0(s)\\&+\sum_{k=0}^{n}R_{n-k}(s)\big(a_{k+1}(s)-\frac{\sum_{i=k+1}^{\infty}a_i(s)}{f(s)}\big)\Bigg]\\&+\frac{1}{f(s)}\sum_{k=0}^{n}R_{n-k}(s)\Bigg(\sum_{i=k+1}^{\infty}a_i(s)z_b(s,m,j)\\&-f(s)z_{b-k}(s,m,j)\Bigg)\\=&\tilde{\theta}_0(s,m,j)u_n(s)+w_n(s,m,j),\end{aligned} \quad (29)$$

where

$$u_k(s)=R_{k+1}(s)a_0(s)+\sum_{i=0}^{k}R_{k-i}(s)b_i(s), \quad (30)$$

$$b_k(s)=a_{k+1}(s)+\frac{1}{f(s)}\sum_{i=0}^{k}a_i(s)-1, \quad (31)$$

$$\begin{aligned}w_k(s,m,j)=\sum_{i=0}^{k}R_{k-i}(s)\Bigg[&z_b(s,m,j)\big(1-\frac{1}{f(s)}\sum_{j=0}^{i}a_j(s)\big)\\&-z_{b-i}(s,m,j)\Bigg].\end{aligned} \quad (32)$$

Substituting $n=b$ and $n=b-1$ into (29) and using (24) yields

$$\begin{aligned}&\tilde{\theta}_0(s,m,j)=\theta_b(s,m,j)=\\&=\frac{\lambda w_{b-1}(s,m,j)-(s+\lambda)w_b(s,m,j)+I(m=0,j=0)}{(s+\lambda)u_b(s)-\lambda u_{b-1}(s)}.\end{aligned} \quad (33)$$

Finally, substituting (33) to (29) we have proven the following theorem.

Theorem 3. *The Laplace transform of the joint distribution of the queue size and the size of the packet being transmitted at time t is equal to*

$$\begin{aligned}&\theta_n(s,m,j)=\\&u_{b-n}(s)\frac{\lambda w_{b-1}(s,m,j)-(s+\lambda)w_b(s,m,j)+I(m=0,j=0)}{(s+\lambda)u_b(s)-\lambda u_{b-1}(s)}\\&+w_{b-n}(s,m,j), \qquad 0\le n\le b,\end{aligned} \quad (34)$$

where functions $u_k(s)$, $w_k(s,m,j)$ are defined in (30), (32), respectively.

For practical purposes we are interested in values of $\Theta_n(t,m,j)$, so that we could use them in (8). To obtain values of $\Theta_n(t,m,j)$ from (34), one of the methods for numerical inversion of the Laplace transform can be used. We use the method proposed in [33].

5 Calculations of $\Delta_n(t)$

It this section we deal again with the classic finite-buffer queue as defined at the beginning of the previous section, but now we are interested in the average number of losses. In particular, let $L(0,t)$ denote the number of packets lost in interval $(0,t)$ and let $\Delta_n(t)$ be its average value provided $X(0)=n$ (see (2)). We will find the Laplace transform of $\Delta_n(t)$:

$$\delta_n(s)=\int_0^{\infty}e^{-st}\Delta_n(t)dt. \quad (35)$$

Arguing similarly as in the previous section we obtain for the initially non-empty system:

$$\begin{aligned}&\Delta_n(t)\\&=\sum_{i=1}^{M}p_iI(d_i/C<t)\sum_{k=0}^{b-n-1}\Delta_{n+k-1}(t-d_i/C)\frac{e^{-\lambda d_i/C}(\lambda d_i/C)^k}{k!}\\&+\sum_{i=1}^{M}p_iI(d_i/C<t)\sum_{k=b-n}^{\infty}\Big[k-b+n+\Delta_{b-1}(t-d_i/C)\Big]\\&\qquad\cdot\frac{e^{-\lambda d_i/C}(\lambda d_i/C)^k}{k!}\\&+\sum_{i=1}^{M}p_iI(d_i/C\ge t)\sum_{k=b-n}^{\infty}(k-b+n)\frac{e^{-\lambda t}(\lambda t)^k}{k!}, \quad 0<n\le b,\end{aligned} \quad (36)$$

and for the initially empty system

$$\Delta_0(t)=\int_0^t\Delta_1(t-u)\lambda e^{-\lambda u}du. \quad (37)$$

Application of the Laplace transform to (36) and (37) gives:

$$\begin{aligned}\delta_n(s)=&\sum_{k=0}^{b-n-1}a_k(s)\delta_{n+k-1}(s)\\&+\sum_{k=b-n}^{\infty}a_k(s)\delta_{b-1}(s)+c_n(s), \quad 0<n\le b,\end{aligned} \quad (38)$$

$$\delta_0(s)=y(s)\delta_1(s), \quad (39)$$

with

$$y(s)=\frac{\lambda}{s+\lambda},$$

$$c_k(s)=\frac{1}{s}\sum_{i=b-k}^{\infty}(i-b+k)a_i(s)+\sum_{i=b-k}^{\infty}(i-b+k)e_i(s),$$

$$\begin{aligned}e_k(s)=&g_k(s,1)+\frac{1}{k!}\lambda^k(s+\lambda)^{-1-k}\\&\cdot\sum_{i=2}^{M}\big(1-\sum_{k=1}^{i-1}p_k\big)\Big[\Gamma(k+1,(s+\lambda)d_{i-1}/C)\\&-\Gamma(k+1,(s+\lambda)d_i/C)\Big],\end{aligned} \quad (40)$$

and $g_k(s,i)$ defined in (22). Changing indexes by substitution $\tilde{\delta}_n(s) = \delta_{b-n}(s)$ we obtain

$$\sum_{k=-1}^{n} a_{k+1}(s)\tilde{\delta}_{n-k}(s) - \tilde{\delta}_n(s) = \varphi_n(s), \qquad 0 \le n \le b-1, \tag{41}$$

$$\tilde{\delta}_b(s) = y(s)\tilde{\delta}_{b-1}(s), \tag{42}$$

where

$$\varphi_n(s) = a_{n+1}(s)\tilde{\delta}_0(s) - \sum_{k=n+1}^{\infty} a_k(s)\tilde{\delta}_1(s) - c_{b-n}(s).$$

Using Lemma 3.2.1 of [32], every solution of the system (41) has the form

$$\tilde{\delta}_n(s) = R_{n+1}(s)c(s) + \sum_{k=0}^{n} R_{n-k}(s)\varphi_k(s), \tag{43}$$

where $c(s)$ does not depend on n. It remains to find $c(s)$, $\tilde{\delta}_0(s)$ and $\tilde{\delta}_1(s)$. Formula (43) with $n = 0$ yields:

$$c(s) = a_0(s)\tilde{\delta}_0(s). \tag{44}$$

Substitution $n = 0$ in (41) gives:

$$\tilde{\delta}_0(s) = \sum_{k=0}^{\infty} a_k(s)\tilde{\delta}_1(s) + c_b(s). \tag{45}$$

Then, substituting $n = b$ and $n = b-1$ in (43) and utilizing (42) we get

$$\begin{aligned}\tilde{\delta}_0(s) =& \delta_b(s) = \\ &= \left[y(s)\sum_{k=0}^{b-1} R_{b-1-k}(s)v_k(s) - \sum_{k=0}^{b} R_{b-k}(s)v_k(s)\right] \Big/ \\ &\Big[R_{b+1}(s)a_0(s) + \sum_{k=0}^{b} R_{b-k}(s)b_k(s) - y(s)\Big[R_b(s)a_0(s) \\ &+ \sum_{k=0}^{b-1} R_{b-1-k}(s)b_k(s)\Big]\Big],\end{aligned} \tag{46}$$

where

$$v_k(s) = \Big(1 - \frac{1}{f(s)}\sum_{i=0}^{k} a_i(s)\Big)c_b(s) - c_{b-k}(s) \tag{47}$$

and $b_k(s)$ is defined in (31). Finally, using (43) again, we obtain the following theorem.

Theorem 4. *The Laplace transform of the average number of losses in interval* $(0,t)$ *has the following form:*

$$\begin{aligned}\delta_n(s) =& \sum_{k=0}^{b-n} R_{b-n-k}(s)v_k(s) + \delta_b(s)\Big[R_{b-n+1}(s)a_0(s) \\ &+ \sum_{k=0}^{b-n} R_{b-n-k}(s)b_k(s)\Big], \quad 0 \le n \le b,\end{aligned} \tag{48}$$

where $R_k(s)$, $v_k(s)$, $\delta_b(s)$, $a_0(s)$ *and* $b_k(s)$ *are given in (26), (47), (46), (21) and (31), respectively.*

Naturally, we can again use the inversion method of [33] for obtaining the values of $\Delta_n(t)$ and use them in (16).

6 Examples

In the following set of numerical examples we checked the performance of three virtual links created at a physical link of capacity[3]:

$$C = 1Gb/s.$$

If not stated otherwise, the total rate of the traffic offered to each of the virtual links was $330Mb/s$ and all three work phases were equal. Therefore, for each virtual link the offered load was:

$$\rho = 0.99.$$

However, the packet size distributions were different. Namely, to the first link only 256-bytes-long packets were offered, to the second link only 512-bytes-long packets were offered, while to the third link the packets of size 40bytes, 512bytes and 1500bytes were offered, according to the following distribution:

$$D: \quad d_1 = 40B, \; d_2 = 512B, \; d_3 = 1500B,$$

$$p_1 = 0.494, \; p_2 = 0.270, \; p_3 = 0.236, \tag{49}$$

As we can calculate, the average packet size in the third link was also

$$\bar{d} = 512B.$$

Taking into account the total arrival rates and the average packet sizes, the packet arrival rates were:

$$\lambda_1 = 161132.812, \quad \lambda_2 = \lambda_3 = 80566.406.$$

Translating these into the queueing terminology: the Poisson arrival rate to the first link was 161132.812 per second, while to the second and third link – 80566.406 per second. The service time of a customer in the first link was constant and equal to $2.048\mu s$, the service time of a customer in the second link was constant and equal to $4.096\mu s$, the service time in the third link could be $0.32\mu s$ or $2.048\mu s$ or $12\mu s$, with probabilities 0.494, 0.270 and 0.236, respectively.

6.1 Dependence on the packet size distribution

In the first example we set $W_1 = W_2 = W_3 = 32\mu s$, $b_1 = b_2 = b_3 = 20$ and checked the dependence of the loss ratio and the average queue size[4] on the packet size

[3] Herein we use the metric meaning of prefixes, i.e. $G = giga = 10^9$, $M = mega = 10^6$.

[4] Here and subsequently, the average queue size in virtual links will be calculated at the beginning of the vacation period.

distribution. The results for the three virtual links are shown in the 'virtual links' section of Tab. 1.

For comparison, in the 'non-virtual links' section of Tab. 1, the results for adequate non-virtual links of capacity $333.333 Mb/s$, with offered traffic of rate $330 Mb/s$, are shown. They were obtained using the formulas for the classic $M/G/1/b$ queue (see, for instance, [34], page 202), with the same packet size distribution and the buffer size as in the adequate virtual link case.

As we can see in Tab. 1, in the virtual links case, the loss ratio depends strongly on both the average packet size and the packet size distribution. Namely, for constant size $256B$ we have 5.23% losses, for constant size $512B$ – 9.79% losses, while for distributed size (with the same average, $512B$) as much as 16.91%. Surprisingly, the average queue size is slightly smaller for distributed packet sizes than for the constant sizes (11.59 versus 12.05).

Both these effects can be understood when we have a look at the distribution of the size of the packet whose transmission was not completed due to lack of time at the end of the work phase. For the third virtual link we have:

$$\tilde{q}_0 = 0.0458, \ \ \tilde{q}_1 = 0.0048, \ \ \tilde{q}_2 = 0.2833, \ \ \tilde{q}_3 = 0.6661.$$

As we have $\tilde{q}_3 >> p_3$, we can confirm now the intuition that the large packets ($1500B$) remain in the buffer for the next cycle much more frequently than they appear in the arrival stream. This explains a relatively small queue size at the end of the work phase - the queue (in packets) may be slightly shorter, but the first packet in the queue is usually large.

Now we can compare the performance of virtual and non-virtual links. As expected, in non-virtual links there are also some losses – they are connected with a random structure of arrivals and finite buffering space. However, in virtual links there are additional losses connected with the bandwidth wastage at the end of the work phase. Therefore, it could be convenient to define the virtualization cost as the percent of the physical bandwidth lost due to virtualization, or, in other words, the difference between the loss ratio in the virtual link and adequate non-virtual one, i.e.:

$$cost = LR_{\text{virtual}} - LR_{\text{non-virtual}}.$$

In the last column of Tab. 1. the virtualization costs for the three studied virtual links are presented. We have $cost_1 = 3.16\%$, $cost_2 = 7.72\%$ and $cost_3 = 11.93\%$, respectively. As we can see, the cost grows with the average packet size, as well as with the variability of the packet size.

6.2 Dependence on the work phase duration

In the second set of examples we kept $\rho = 0.99$, $b_1 = b_2 = b_3 = 20$, but changed the phases lengths.

In particular, in Tab. 2 the results for $W_1 = W_2 = W_3 = 64\mu s$ are shown. For links 2 and 3 the loss ratio dropped significantly comparing to $32\mu s$ phases, which was to be expected. However, for link 1 the loss ratio increased significantly, which might be surprising at first glance. This effect can be explained using the buffer sizes. If we are extending the phases lengths, from some point the buffers become insufficient to store the traffic incoming during the vacation phase. This is more visible when the packets are small – in link 1 there are twice more arriving packets than in links 2 and 3.

In Tab. 3 the results for short phases, $W_1 = W_2 = W_3 = 16\mu s$, are presented. For links 2 and 3 we have very high loss ratios due to the bandwidth wastage at the end of the work period. However, for link 1 the loss ratio is almost unaltered compared to phase $64\mu s$. Now the buffer is large enough, but the bandwidth wastage effect is more dominant, and the balance between these two effects led to a similar result.

Now, it must be stressed that the performance of a virtual link may depend in a highly variable, non-monotonic manner on the phase length, especially for constant packet sizes and short phase lengths. An example of the dependence of the loss ratio on the work phase length for link 2 is depicted in Fig. 3. As we can see, even a minor change of the phase length may cause an increase of the loss ratio from 0.03 to 0.3 and vice versa. This is again connected with the bandwidth wastage at the end of the work period. If the work period is slightly greater than a multiple of packet transmission time, the bandwidth wastage is low. If the work period is slightly smaller than a multiple of packet transmission time, the bandwidth wastage is high, especially if the phase is short.

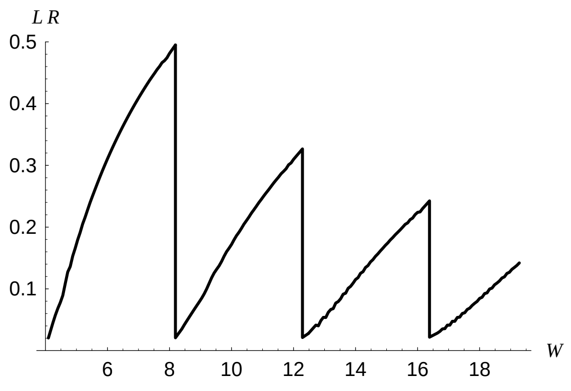


Fig. 3: The loss ratio versus the work period duration for link 2. $\rho = 0.99$, $b = 20$. The work period was W (in μs), the vacation period was $V = 2W$.

Table 1: The loss ratio and the average queue size for three virtual and three non-virtual links. $\rho = 0.99$, $b_1 = b_2 = b_3 = 20$, $W_1 = W_2 = W_3 = 32\mu s$.

	virtual links results			non-virtual links results			
link	loss ratio	average queue	stddev queue	loss ratio	average queue	stddev queue	cost
link 1, 256B	0.0523	5.960	3.909	0.0207	9.416	5.735	3.16%
link 2, 512B	0.0979	12.05	3.987	0.0207	9.416	5.735	7.72%
link 3, distr. D	0.1691	11.59	5.371	0.0498	9.775	6.091	11.93%

Table 2: The loss ratio and the average queue size for three virtual and three non-virtual links. $\rho = 0.99$, $b_1 = b_2 = b_3 = 20$, $W_1 = W_2 = W_3 = 64\mu s$.

	virtual links results			non-virtual links results			
link	loss ratio	average queue	stddev queue	loss ratio	average queue	stddev queue	cost
link 1, 256B	0.0967	0.889	1.403	0.0207	9.416	5.735	7.60%
link 2, 512B	0.0523	5.960	3.909	0.0207	9.416	5.735	3.16%
link 3, dist. D	0.1074	6.662	5.143	0.0498	9.775	6.091	5.76%

Table 3: The loss ratio and the average queue size for three virtual and three non-virtual links. $\rho = 0.99$, $b_1 = b_2 = b_3 = 20$, $W_1 = W_2 = W_3 = 16\mu s$.

	virtual links results			non-virtual links results			
link	loss ratio	average queue	stddev queue	loss ratio	average queue	stddev queue	cost
link 1, 256B	0.0979	12.05	3.987	0.0207	9.416	5.735	7.72%
link 2, 512B	0.2239	16.77	1.983	0.0207	9.416	5.735	20.32%
link 3, distr. D	0.3301	16.13	3.722	0.0498	9.775	6.091	28.03%

6.3 Dependence on the buffer size

In the third set of numerical examples we checked the dependence of the virtual links performance on the buffer size. In all examples $W_1 = W_2 = W_3 = 32\mu s$ was set. The results for the buffers of sizes 10 and 40 are presented in Tables 4 and 5, respectively. These tables are also to be compared with Tab. 1, where the results for buffers of size 20 are presented. As we can see, for small buffer ($b = 10$) the loss ratio is high, especially for small constant packets, or distributed sizes. For larger buffers, the loss ratio stabilizes – there are small differences between $b = 20$ and $b = 40$. This is confirmed in Fig. 4, in which the dependence of the loss ratio on the buffer size for link 2 is depicted.

As we can see in Tables 4, 1 and 5, the queue size changes in a different way - it grows linearly with buffer size. This is further depicted in Fig. 5, where the average queue size versus the buffer size for link 2 is depicted.

Now, both Figs. 4 and 5 indicate that the system would be unstable (the queue growing to infinity), if the buffer was infinite, even though we have $\rho < 1$. Once again, this is due to the bandwidth wastage at the end of each work period. Therefore, it is an interesting open question, which ρ would guarantee stability of the queue size in the case of an infinite buffer. For instance, for $\rho = 0.85$ ($\lambda_2 = 69173.177$) the system would be stable. It can be observed in Figs. 6 and 7. When the buffer size grows, the queue grows only up to a certain level and then remains unaltered. As for the loss ratio, it decreases to zero with the buffer size. This behaviour of the queue is different than observed in Figs. 4, 5. Therefore we may expect that somewhere in interval $(0.85, 0.99)$ there is the critical load, below which the system is stable, even with an infinite buffer.

7 Conclusions

We presented an analysis of a scheduling algorithm that creates several virtual links on one physical link by switching the service between the virtual links. In particular, we have proven formulas for calculating the queue size distribution and the loss ratio for queues associated with virtual links. We have shown numerical examples of the performance characteristics of created virtual links, including their dependence on the most important system parameters. We also presented the cost

Table 4: The loss ratio and the average queue size for three virtual and three non-virtual links. $\rho = 0.99$, $b_1 = b_2 = b_3 = 10$, $W_1 = W_2 = W_3 = 32\mu s$.

	virtual links results			non-virtual links results			
link	loss ratio	average queue	stddev queue	loss ratio	average queue	stddev queue	cost
link 1, 256B	0.1448	0.820	1.152	0.0463	4.901	2.895	9.85%
link 2, 512B	0.1197	3.630	2.187	0.0463	4.901	2.895	7.34%
link 3, distr. D	0.2107	3.994	3.011	0.0994	4.985	3.218	11.13%

Table 5: The loss ratio and the average queue size for three virtual and three non-virtual links. $\rho = 0.99$, $b_1 = b_2 = b_3 = 40$, $W_1 = W_2 = W_3 = 32\mu s$.

	virtual links results			non-virtual links results			
link	loss ratio	average queue	stddev queue	loss ratio	average queue	stddev queue	cost
link 1, 256B	0.0347	20.43	8.833	0.0083	17.45	11.34	2.64%
link 2, 512B	0.0953	31.56	4.957	0.0083	17.45	11.34	8.70%
link 3, distr. D	0.1571	30.05	7.663	0.0233	18.88	11.82	13.38%

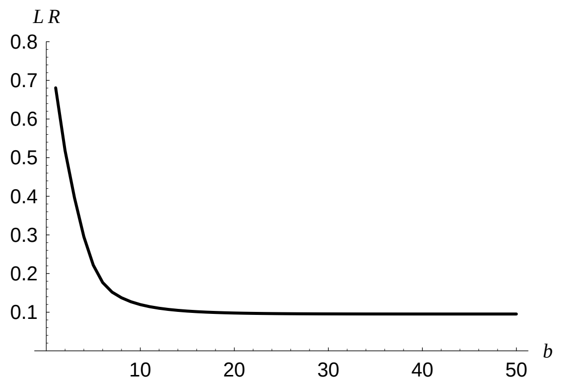


Fig. 4: The loss ratio versus the buffer size for link 2. $\rho = 0.99$, $W = 32\mu$s, $V = 64\mu$s.

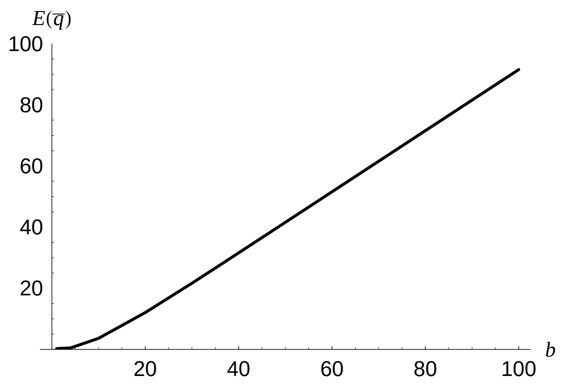


Fig. 5: The average queue size versus the buffer size for link 2. $\rho = 0.99$, $W = 32\mu$s, $V = 64\mu$s.

of link virtualization, i.e. the percent of bandwidth lost due to the usage of virtual, instead of physical, link. As we could see, this cost is not marginal and varies from a few to even thirty percent in some parameterizations. Naturally, this observation is not meant to depreciate the idea of link virtualization, but to demonstrate only, that creating virtual links is not for free.

As we could see also, manipulating the parameters of the system (durations of phases, buffer sizes, packet sizes) have a complex (and sometimes counterintuitive) impact on the system performance. We hope that the provided mathematical tools will be of help in the design and parameterization of virtual links.

As for the future work, there are two important goals to be accomplished. The first is finding the distribution of the queueing delay (waiting time) in the presented

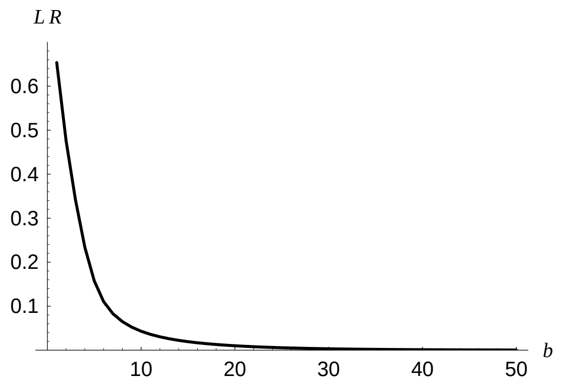


Fig. 6: The loss ratio versus the buffer size for link 2. $\rho = 0.85$, $W = 32\mu$s, $V = 64\mu$s.

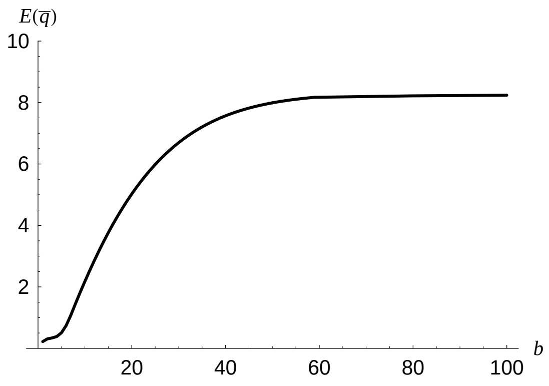


Fig. 7: The average queue size versus the buffer size for link 2. $\rho = 0.85$, $W = 32\mu$s, $V = 64\mu$s.

scheduler. The second is formulation of the conditions under which the scheduler with infinite buffers would be stable (as discussed at the end of section 6.3).

References

[1] T. Anderson, L. Peterson, S. Shenker, J. Turner, Overcoming the Internet Impasse through Virtualization, Computer, **38**, 34–41 (2005).

[2] D. Papadimitriou et al., Fundamental Limitations of current Internet and the path to Future Internet, EC FIArch Group technical report, (2011).

[3] A. Galis, et al., Management and Service-aware Networking Architectures (MANA) for Future Internet. System Functions, Capabilities and Requirements, Position Paper, ver. 6.0, (2009).

[4] M. Hirabaru et al., New Generation Network Architecture AKARI Conceptual Design (ver1.1). NICT technical report, (2008).

[5] L. Peterson et al., GENI: Global Environment for Network Innovation, GENI Design Principles GDD/06/08, technical report, (2006).

[6] W. Burakowski, A. Beben, J. Sliwiski, H. Tarasiuk, A. Binczewski, A. Grzech and T. Czachrski, The Future Internet Engineering project in Poland: goals and achievements, Towards and Beyond Europe 2020 - the Significance of Future Internet for Regional Development. Future Internet Week Report, Publishing House of the Institute for Sustainable Technologies - National Research Institute, (2012).

[7] W. Burakowski, H. Tarasiuk, A. Beben, W. Goralski and P. Wisniewski, Ideal device supporting virtualization of network infrastructure in System IIP (in Polish), Proc. of KSTiT'11, Lodz, Poland, 818–823, September 14-16 (2011).

[8] W. Burakowski, H. Tarasiuk, A. Beben and G. Danilewicz, Virtualized network infrastructure supporting co-existence of Parallel Internets, Proc. of SNPD'12, Kyoto, 679–684, August 8–10, (2012).

[9] A. Chydzinski, M. Rawski, P. Wisniewski, B. Adamczyk, I. Olszewski, P. Szotkowski, L. Chrost, P. Tomaszewicz, D. Parniewicz, Virtualization Devices for Prototyping of Future Internet, Proc. of SNPD'12, Kyoto, 672–678, August 8–10, (2012).

[10] H. Takagi, *Queueing analysis - Vacation and Priority Systems*, North-Holland, Amsterdam, (1991).

[11] N. Tian and Z. G. Zhang, *Vacation Queueing Models - Theory and Applications*, Springer, New York, (2006).

[12] B. T. Doshi, Queueing systems with vacation: A survey. Queueing Systems, **1**, 29-66 (1986).

[13] J. C. Ke, C. H. Wu and Z. G. Zhang, Recent developments in vacations queueing models: A short survey, International Journal of Operations Research, **7**, 3–8 (2010).

[14] S. Hur and S. Ahn, Batch arrival queues with vacations and server setup, Applied Mathematical Modelling, **29**, 1164–1181 (2005).

[15] U. C. Gupta, K. Sikdar, Computing queue length distributions in MAP/G/1/N queue under single and multiple vacation, Applied Mathematics and Computations, **174**, 1498–1525 (2006).

[16] J. Wu, Z. Liu and Y. Peng, On the BMAP/G/1 G-queues with second optional service and multiple vacations, Applied Mathematical Modelling, **33**, 4314–4325 (2009).

[17] K.-H. Wang, M.-C. Chan and J-C. Ke, Maximum entropy analysis of the $M^{[X]}/M/1$ queueing system with multiple vacations and server breakdowns, Computers and Industrial Engineering, **52**, 192–202 (2007).

[18] K. Sikdar, U.C. Gupta, On the batch arrival batch service queue with finite buffer under servers vacation: $M^{[X]}/G^{[Y]}/1/N$ queue, Computers & Mathematics with Applications, **56**, 2861–2873 (2008).

[19] T. T. Lee, M/G/1/N Queue with Vacation Time and Exhaustive Service Discipline, Operations Research, **32**, 774–784 (1984).

[20] T. Takine and T. Hasegawa, On the M/G/l queue with multiple vacatlions and gated service discipline, J. Oper. Res. Soc. Japan, **35**, 217–235 (1992).

[21] T. T. Lee, M/G/1/N Queue with vacation time and limited service discipline, Performance Evaluation, **9**, 181-190 (1989).

[22] K. K. Leung and M. Eisenberg, A Single Server Queue with Vacations and Gated Time-Limited Service, IEEE Transactions on Communications, **38**, 1454–1462 (1990).

[23] K. K. Leung and M. Eisenberg, A single-server queue with vacations and non-gated time-limited service, Performance Evaluation, **12**, 115–125 (1991).

[24] H. Takagi and K. K. Leung, Analysis of a discrete-time queueing system with time-limited service, Queueing Systems, **18**, 183–197 (1994).

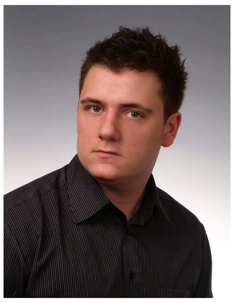
[25] T. Katayama, Waiting time analysis for a queueing system with time-limited service and exponential timer, Naval Research Logistics, **48**, 638–651 (2001).

[26] M. Sosnowski and W. Burakowski, Analysis of the system with vacations under Poissonian input stream and constant service times, Proc. of Polish Teletraffic Symposium, 9–13, Zakopane, December 6-7, (2012).

[27] A. Chydzinski, A Unified Method of Analysis for Queues with Markovian Arrivals, Mathematical Problems in Engineering, **2012**, ID 831956, 1-18 (2012).

[28] K. Rusek, L. Janowski and Z. Papir, Correct router interface modeling, Proc. International Conference on Performance Engineering, 97-102 (2011).
[29] A. Chydzinski, The M/G-G/1 oscillating queueing system, Queueing Systems, **42**, 255–268 (2002).
[30] A. Chydzinski, The oscillating queue with finite buffer, Performance Evaluation, **57**, 341–355 (2004).
[31] A. Chydzinski, Duration of the buffer overflow period in a batch arrival queue, Performance Evaluation, **63**, 493–508 (2006).
[32] A. Chydzinski, Time to reach buffer capacity in a BMAP queue, Stochastic Models, **23**, 195–209 (2007).
[33] J. Abate, G. L. Choudhury and W. Whitt, An introduction to numerical transform inversion and its application to probability models. Chapter in Computational Probability, W. Grassman (ed.), Kluwer, Boston, 257–323 (2000).
[34] H. Takagi, *Queueing analysis - Finite Systems*. North-Holland Amsterdam, (1993).

Andrzej Chydzinski received his MS (in applied mathematics), PhD (in computer science) and DSc (in computer science) degrees from the Silesian University of Technology, Gliwice, Poland, in 1997, 2002 and 2008, respectively. He is currently a professor in the Institute of Informatics of this university. His academic and professional interests are with computer networking, in particular with performance evaluation of computer networks, Future Internet design, active queue management in Internet routers, mathematical modelling, queueing theory and discrete-event network simulators. He authored and co-authored 4 books, about 40 conference papers and about 40 journal articles, including papers in Telecommunication Systems, Performance Evaluation, Pattern Recognition, Microprocessors and Microsystems, Queueing Systems, Stochastic Models, Mathematical Problems in Engineering, Applied Mathematical Modelling and other. He is also reviewing articles for several high-quality journals, including IEEE/ACM Transactions on Networking, Annals of Operations Research, Applied Mathematical Modelling, Queueing Systems, Mathematical Problems in Engineering, International Journal of Applied Mathematics and Computer Science, Journal of Network and Computer Applications, Performance Evaluation and other. He is a Technical Program Committee member for several conferences. He was (and is) a leader of several scientific projects founded by Polish state and European Union. Since January 2011 he has been an IEEE Senior Member.



Blazej Adamczyk received his MS and PhD degrees in computer science, from the Silesian University of Technology, Gliwice, Poland, in 2009 and 2013 respectively. He is currently an assistant professor in the Institute of Informatics at this university. His main research interests are computer networks, virtualization and security. He is an author or a co-author of 2 books, 8 conference papers, 3 technical reports and 6 journal articles, including papers in Mathematical Problems in Engieneering, Theoretical and Applied Informatics and International Journal On Advances in Networks and Services. He took part in several scientific projects founded by Polish state and European Union. He is also an IEEE member.