Ethernet PON (ePON): Design and Analysis of an Optical Access Network.

Glen Kramer
Department of Computer Science
University of California, Davis, CA 95616, USA
Tel: 1.530.297.5217; Fax: 1.530.297.5301
E-mail: *kramer@cs.ucdavis.edu*

Biswanath Mukherjee
Department of Computer Science
University of California, Davis, CA 95616, USA
Tel: 1.530.752.5129; Fax: 1.530.752.4767
E-mail: *mukherjee@cs.ucdavis.edu*

Gerry Pesavento
Advanced Technology Lab.
Alloptic, Inc.
Livermore, CA 94550, USA
Tel: 1.925.245.7600; Fax: 1.925.245.7601
E-mail: *gerry.pesavento@alloptic.com*

August, 2000

Abstract

With the expansion of services offered over the Internet, the “last mile” bottleneck problems continue to exacerbate. A Passive Optical Network (PON) is a technology viewed by many as an attractive solution to this problem.

In this study, we propose the design and analysis of a PON architecture which has an excellent performance-to-cost ratio. This architecture uses the time-division multiplexing (TDM) approach to deliver data encapsulated in Ethernet packets from a collection of Optical Network

Units (ONUs) to a central Optical Line Terminal (OLT) over the PON access network. The OLT, in turn, is connected to the rest of the Internet. A simulation model is used to analyze the system's performance such as bounds on packets delay and queue occupancy. Then, we discuss the possibility of improving the bandwidth utilization by means of timeslot size adjustment, and by packet scheduling.

Keywords: access network, local loop, passive optical network, PON, time-division multiple access, TDMA, self-similar traffic

1. Introduction

Passive Optical Networks (PON) are point-to-multipoint optical networks with no active elements in the signals' path from source to destination. The only interior elements used in such networks are passive combiners, couplers, and splitters.

PON technology is getting more and more attention by the telecommunication industry as the "last mile" solution [1,2]. Advantages of using a PON for local access networks are numerous:

- A PON allows for longer distances between central offices and customer premises. While with the Digital Subscriber Line (DSL) the maximum distance between the central office and the customer is only 18000 feet (approximately 5.5 km), a PON local loop can operate at distances of over 20 km.
- A PON minimizes fiber deployment in both the local exchange and the local loop.
- A PON provides higher bandwidth due to deeper fiber penetration. While the fiber-to-the-building (FTTB), fiber-to-the-home (FTTH), or even fiber-to-the-PC (FTTPC) solutions have the ultimate goal of fiber reaching all the way to customer premises, fiber-to-the-curb (FTTC) may be the most economical deployment today.
- As a point-to-multipoint network, a PON allows for downstream video broadcasting.
- A PON eliminates the necessity of installing multiplexers and demultiplexers in the splitting locations, thus relieving network operators from the gruesome task of maintaining them and providing power to them. Instead of active devices in these locations, a PON has passive components that can be buried into the ground at the time of deployment.

- A PON allows easy upgrades to higher bit rates or additional wavelengths.

The clear advantages of using PON technology in access networks dictate that we make important design decisions in implementing it. Because an access network aggregates traffic from a relatively small number of subscribers (compared to metro or regional networks), it is very cost sensitive. Therefore, a PON design should not require over-provisioning and should allow for incremental deployment.

In this study, we propose the design and analysis of a PON architecture that has an excellent performance-to-cost ratio. This architecture (Section 2) uses the time-division multiplexing (TDM) approach to deliver data encapsulated in Ethernet packets from a collection of Optical Network Units (ONUs) to a central Optical Line Terminal (OLT) over the PON access network. The OLT, in turn, is connected to the rest of the Internet. A simulation model described in Section 3 is used to analyze the system's performance such as bounds on packets delay and queue occupancy.

The simulation analysis was performed using Bellcore traces that exhibit the property of *self-similarity* [3]. Self-similar (or fractal) traffic has the same or similar degree of burstiness observed at a wide range of time scales. Using self-similar traffic is extremely important as it provides realistic bounds on packets delay and queue occupancy. (See Section 4.)

We continue our investigation of the bandwidth utilization of our proposed model and considered two ways to improve it. In Section 5 we consider the timeslot size adjustment to achieve the best operating parameters (utilization and delay). We employ the *power function* [4] as a convenient measure of system performance. In Section 6 we consider an alternative approach to improve utilization; specifically, we examine packet scheduling. In doing so, we pay special attention to the effect of packet reordering on TCP/IP connection behavior.

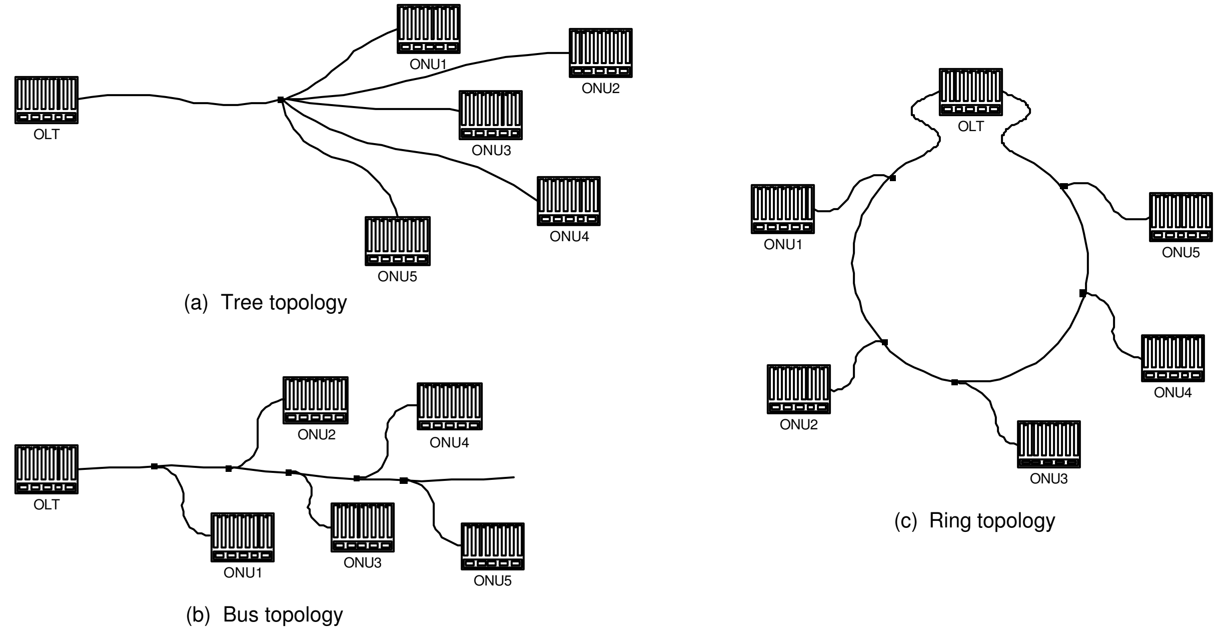
Section 7 concludes this study.

2. PON Design Alternatives

There are several topologies suitable for the access network: tree, ring, or bus (Fig. 1). A PON can also be deployed in redundant configuration as double ring or double tree; or redundancy may be added only to a part of the PON, say the trunk of the tree (Fig. 2). For the rest of this article, we will focus our attention on the tree topology; however, most of the conclusions made are equally relevant to other topologies.

All transmissions in a PON are performed between Optical Line Terminal (OLT) and Optical Network Units (ONU). Therefore, in the downstream direction (from OLT to ONUs), a PON is a point-to-multipoint network, and in the upstream direction it is a multipoint-to-point network.

The OLT resides in the local exchange (central office), connecting the optical access network to an IP, ATM, or SONET backbone. The ONU is located either at the curb (FTTC solution), or at the end-user location (FTTH, FTTB solutions), and provides broadband voice, data, and video services.



(a) Tree topology

(b) Bus topology

(c) Ring topology

Fig. 1. PON topologies.

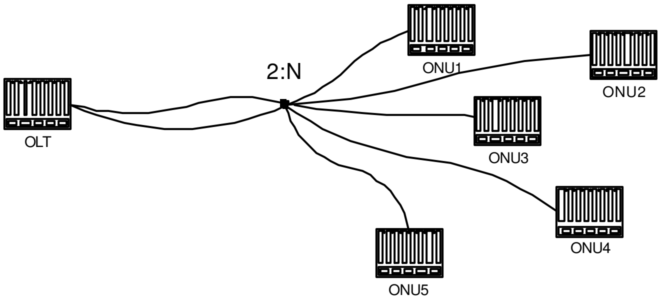


Fig. 2. Tree with a redundant trunk.

Access networks based on PON technology face several design challenges, regardless of the physical topology. The first design decision to be made is the data-link technology. The Table 1 summarizes the advantages and disadvantages of different data-link technologies.

Data Link	Advantage	Disadvantage
SONET	Fault tolerance, fault management, large installed base.	Expensive hardware – too expensive for the local loop. Also not efficient for data traffic.
ATM	Queues in the OLT and ONUs can easily implement various QoS policies and guarantees providing better support for real-time traffic (voice and video).	At the customer side and at the network side, data has the form of IP packets. In order to traverse the PON, IP packets should be broken into cells and reassembled at the other end. This introduces additional complexity and cost.
Ethernet	Very convenient to carry IP packets (see ATM disadvantage); ubiquitous and cheap hardware; scalable (100 Mbps, 1 Gbps, 10 Gbps).	Requires QoS techniques for real-time traffic.

Table 1. Advantages and disadvantages of data-link technologies in PON.

Another design challenge is the separation of upstream channels belonging to different ONUs. Without such separation, two ONUs may start transmitting (not necessarily simultaneously) such that their transmissions, when they reach the trunk (combiner), may overlap (most likely, only partially) and thus will collide. The available solutions for multiplexing are WDM, TDM, and CDM. Table 2 describes the advantages and disadvantages of each solution.

	Advantage	Disadvantage
WDM	Provides high bandwidth. This could be the best approach as it is very simple to implement.	Cost and scalability: the OLT has to have a transmitter array with one transmitter for each ONU. Then, adding a new ONU could be a problem, unless transmitters were overprovisioned in advance. Each ONU must have a wavelength-specific laser.
TDM	Allows each ONU to have a fraction of a wavelength's capacity. Only one transmitter needed in the OLT, no matter how many ONUs are connected.	More complicated than WDM. Requires ONUs to be synchronized.
CDM	No fixed limit on number of users; provides security.	Inter-channel interference increases with number of users; Most importantly, physical components must be able to handle signal rate much higher than the user's data rate [5].

Table 2. Advantages and disadvantages of media-access technologies in PON.

It should also be mentioned here that WDM and TDM approaches may be combined when a subset of ONUs share a common wavelength. For example, if the PON combines home and business users, business users may share one wavelength, and home users may share the other wavelength. This solution still remains scalable as new users can be added to each group without adding new hardware to the OLT.

3. Model Description

Based on the discussion above, it seems that an Ethernet and TDM combination has the best of all qualities. An ePON is a PON that carries Ethernet traffic. Because Ethernet is broadcasting by nature, it fits perfectly with the ePON architecture in the downstream direction (from network to user): packets are broadcast by the OLT and extracted by their destination ONU based on their media-access control (MAC) address. In the upstream direction (from user to network), each ONU will use a separate TDM channel.

In this study, we consider a model with *N* ONUs. Every ONU is being assigned a timeslot. All *N* timeslots together compose a frame. A frame typically would have a small overhead used for synchronizing the ONUs to the OLT's clock, but we consider it to be negligibly small for the purposes of our analysis. In all numerical examples presented in this study, the default value for N was chosen to be 16.

From the access side, traffic may arrive to an ONU from a single user or from a gateway of a local-area network (LAN), that is, traffic may be aggregated from a number of users. Packets should be buffered in the ONU until the correct timeslot for this ONU arrives. Then, packets will be transmitted upstream. Transmission speed of the PON and the user access link may not necessarily be the same. In our model, we consider R_D Mbps to be the data rate of the access link from a user to an ONU, and R_U Mbps to be the bit rate of the upstream slotted link from the ONU to the OLT (see Fig. 3), with default values of R_D and R_U being 100 Mbps and 1000 Mbps respectively, in our numerical examples.

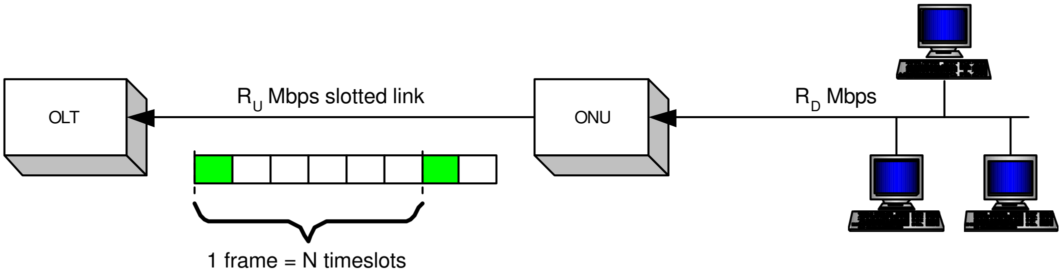


Fig. 3. System model.

The questions we shall try to answer in this study are: what is the average delay the packets will experience in the ONU buffer, how big this buffer should be, and what link utilization we can achieve.

To perform trace-driven simulations, we used Bellcore traces [3]. The arrival timestamps were converted into bytestamps (one unit corresponds to a time needed to transmit one byte). Even though the original traces were obtained on a 10-Mbps link, using bytestamps instead of timestamps allowed us to use the traces as if they were collected on a 100-Mbps link. To simulate network behavior with higher loads, we scaled the traffic up, i.e., we proportionally decreased every inter-packet gap to achieve the desired load. While doing so, we kept the minimum inter-packet gap to be 8 bytes (for preamble), as specified in the IEEE 802.3 standard. It is important to notice that the original traffic was only scaled up to simulate higher load. Scaling the traffic down does not preserve the property of self-similarity – same or similar degree of burstiness observed at different timescales (refer to [6] for an extended bibliography on the subject). Spreading packets far apart will reduce the burstiness observed on very fine scales. Scaling traffic up, on the other hand, preserves the burstiness, as the degree of burstiness observed at larger scale now will manifest itself at smaller scale. Also, burstiness will disappear at a very high load. This is due to the fact that most of the inter-packet gaps are reduced to a minimum of 8 bytes. This is in no way representative of the real network traffic and we did not simulate the ONU load of more then 62.5%. This particular value was chosen for the following reason: maximum bandwidth available to an ONU is R_U / N, which, based on default values, equals 1000 Mbps / 16 = 62.5 Mbps.

When offered load per ONU exceeds 62.5 Mbps (or 62.5% of input rate of 100 Mbps), the system becomes unstable. In reality, such a system should drop the packets, thus reducing the effective (carried) load.

4. Analysis of Packet Delay and Queue Size

In this section, we discuss how delay and queue size depend on the network load. We first consider a simple FIFO queue. Only the packets that arrive before the timeslot may be sent in the timeslot, i.e., the system uses "gated" service. If the next packet to be sent is larger than the remaining timeslot, then this packet and all packets that arrived after it will wait for the next timeslot.

Before we present our results, let us consider what are the constituents of the delay experienced by a packet. Packets arrive to the ONU at random times. Every packet has to wait for the next timeslot to be transmitted upstream. This delay is termed ***TDM delay***. TDM delay is the time interval between packet arrival and the beginning of the next timeslot. In [7], this delay is called “slot synchronization delay”.

Due to the bursty nature of network traffic, even at light or moderate network load, some timeslots may fill completely and still more packets may be waiting in the queue. Those packets will have to wait for later timeslots to be transmitted. This additional delay is called ***Burst delay***. Burst delay may span multiple frames (recall that a frame consist of N timeslots where N is the number of ONUs).

In our simulation, we define packet delay to be the time interval between the end of reception of the last byte and the beginning of the transmission of the first byte. Thus, packet transmission time is excluded from our calculations.

Ethernet traffic can be considered to be an ON/OFF process where an ON period corresponds to packets being transmitted, and an OFF period corresponds to inter-packet gaps. Then bursts of traffic, as seen by some buffer, can be characterized as a combination of bursts of ON intervals and OFF intervals. Burst of ON intervals is a burst of packet sizes, when the network suddenly sees group of packets of larger size. Burst of OFF intervals is a burst of inter-packet gaps, when the network sees a group of inter-packet gaps of very small size.

Of course, in real traffic, both mechanisms affect the overall traffic shape, and are not separable. However, in our first simulation result, we will attempt to observe network behavior when traffic is only subjected to packet size bursts and not to inter-packet gap bursts.

We will use the following traces:

Trace A: This is the original Bellcore trace [3], but scaled up to simulate the necessary network load. It still preserves the property of self-similarity, i.e., similar degree of burstiness can be observed at different timescales (see Fig. 4, trace A).

Trace B: This trace uses packet sizes from trace A, but has packet timestamps modified to obtain inter-packet gaps of equal size. Thus, effectively, this trace gets rid of inter-packet gap bursts. The burstiness observed in this trace is only due to bursts of packet sizes (see Fig. 4, trace B).

Trace C: This trace also uses packet sizes from trace A. It has packet timestamps modified to make inter-packet gap proportional to the size of the packet that follows it, i.e., a larger packet follows a larger gap. Thus the ON bursts and OFF bursts are exactly in anti-phase. If we look at this traffic at a scale of few packet sizes, we will see a constant traffic rate (expressed in bytes per unit of time), i.e., ON and OFF bursts cancel each other (see Fig. 4, trace C).

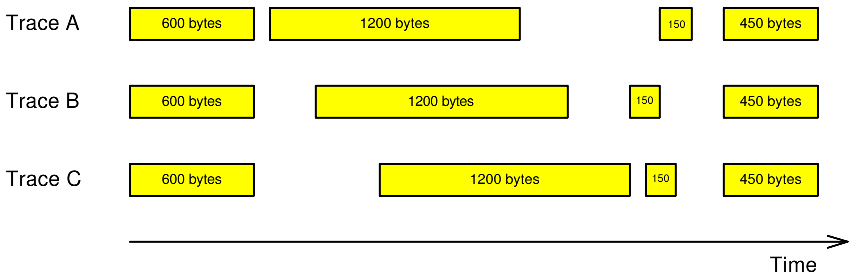


Fig. 4. Illustration of traces used in simulations.

We need to emphasize here that traces B and C do not bear any resemblance to the real network traffic. However, comparing packet delays in simulations using traces A, B, and C will let us visualize how much ON bursts and OFF bursts contribute to the delay.

The following are our simulation parameters:

- Each trace contained 1 million packets.
- Frame time = 2 ms. This is the time between the arrivals of successive timeslots for each ONU. Thus, the expected TDM delay is 1 ms.
- Timeslot size = 15625 bytes. This value is defined by the frame time, number of ONUs, and the line rate.

$$Timeslot\,(bytes) = \frac{Frame(s)}{N} \times Line\_Rate = \frac{2 \times 10^{-3}\,s}{16} \times 10^{9}\,\frac{bit}{s} \times \frac{1}{8}\frac{byte}{bit} = 15625(bytes)$$

- No packets were dropped, i.e., infinite buffer at each ONU.
- Load was varied from 11% to 63% with 1% increments. A load of 11% is the original network load. Higher loads were obtained by scaling the original load up.

Fig. 5 shows the results of our first simulation.

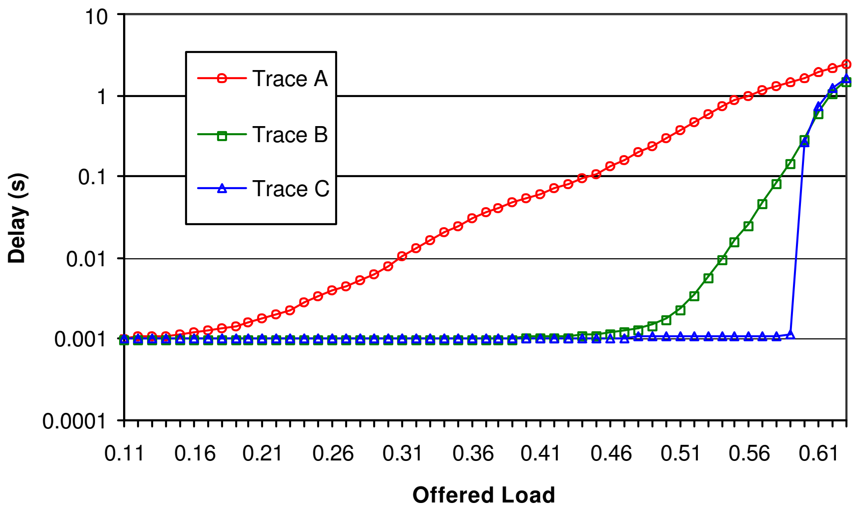


Fig. 5. Average packet delay.

Here we can see that trace C introduces the shortest delay. In fact, only TDM delay is present up to about 59% load. The reason for such a nice behavior is that every timeslot is getting approximately the same number of bytes. Then, when one timeslot finally overflows, all of them overflow, and we have avalanche-like increase of packet delay. Trace B shows mostly TDM delay up to a load of 40%. At this load, the number of timeslots that overflow start increasing exponentially. Trace A shows exponential increase almost from the beginning. It is mostly due to inter-packet gap bursts, i.e., packets are arriving close to each other (in a packet train). The fact that we have some timeslots overflowing means that there were some bursts of traffic that delivered more than 15625 bytes (timeslot size) in 2 milliseconds (frame time). This means that while the overall network load was only 11% or 11 Mbps, there were periods when the burst rate achieved at least 15625 (bytes) x 8 bit/byte / 2 ms = 62.5 Mbps.

Fig. 6 represents the average queue size sampled just before every timeslot arrival. This behavior is very similar to that of the average packet delay. The queue size for trace C grows linearly up to timeslot size (15625 bytes). On loads above 59%, every timeslot sees the queue of size larger than the timeslot can accommodate. Trace B shows that bursts of packet sizes can be tolerated up to a load of 40%. However, it is the inter-packet gap bursts that make buffering less efficient. Even at very low load, they introduce a fair amount of packet trains such that larger buffers are needed. With the increase of load, the needed buffer space increases exponentially (trace A). Similar queue behavior was observed in [8] in a model employing multiple sources of power-tail Markov-Modulated Poisson Processes.

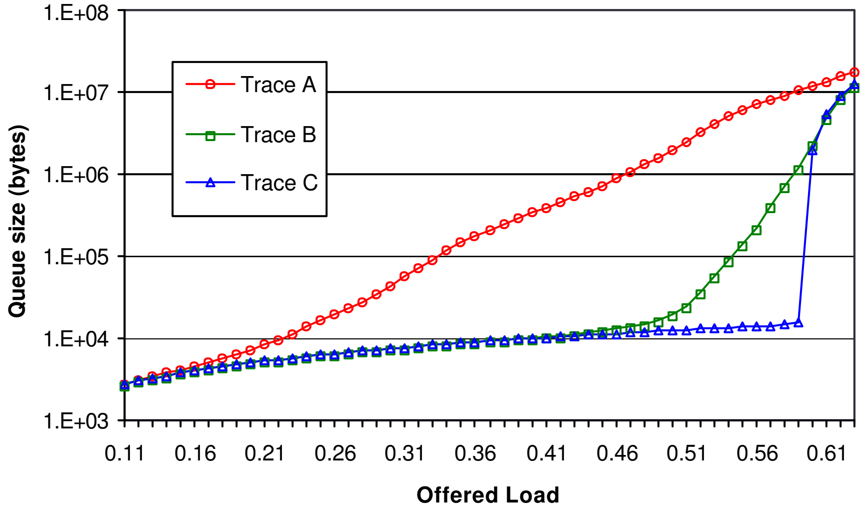


Fig. 6. Average queue size.

The above results show that the Burst delay dominates the TDM delay in the overall delay experienced by a packet. As such, it would not make much sense to reduce the TDM delay, say by reducing the timeslot size. It is also true that increasing the timeslot size will not introduce a lot of delay.

We also illustrated here that packet loss could not be prevented. It only can be mitigated at the expense of exponential increase of buffer space and packet delay. This property of self-similar traffic provides a startling contrast to models employing Poisson arrival process. Under those models, it was possible to increase the buffer and timeslot size just enough, so that traffic averaged over the frame time would appear smooth and would entirely fit in the buffer, so that no packet loss will be observed.

In a real network, the traffic bursts have a heavy-tail distribution [9, 10]. Tail of distribution function for such distributions decreases sub-exponentially, unlike Poisson where decrease is exponential. This leads to the fact that the probability of extremely large bursts is greater than in Poisson. This also means that no traffic smoothing is possible in real networks.

The next section will evaluate the effective egress bandwidth and will examine the possibility of finding optimal parameters that minimize the delay while maximizing the available bandwidth.

5. Bandwidth Utilization

Previously, we have shown that the maximum egress bandwidth available to an ONU is R_U / N (which equals 62.5 Mbps with our default parameters). In this model, we assume that Ethernet packets cannot be fragmented, i.e., if the next packet to be transmitted is larger than the remainder of the timeslot, the packet will wait for the next timeslot. This also means that the timeslot will be transmitted with an unused remainder at the end.

Then, the question is how large the mean remainder is?

Let T – timeslot size

R – random variable representing unused remainder

X – random variable representing packet sizes

A, B – range for packet sizes: $A \le size \le B$

In Ethernet A = 64 bytes, B = 1518 bytes

By definition, the expected remainder is

$$E(R) = \sum_{r=1}^{B-1} r \times P(R = r) \tag{1}$$

Obviously, the remainder can only be in the range from 1 to B-1. If the remainder is more than B-1 bytes long, then we are guaranteed that the next packet will fit into the current timeslot, thus reducing the remainder. Here, we assume that we always have packets waiting, i.e., load is heavy.

Assuming that we have placed k packets in the timeslot, what is the probability of getting a remainder of size r? Taking $S_k = X_1 + X_2 + \ldots + X_k$, this probability is

$$\begin{aligned} P(R = r \mid K = k) &= P(S_k = T - r \cap S_{k+1} > T \mid K = k) \\ &= P(S_k = T - r \cap X_{k+1} > r \mid K = k) \\ &= P(S_k = T - r \mid K = k) \times P(X_{k+1} > r \mid K = k) \end{aligned} \tag{2}$$

Since X is independent and identically distributed,

$$P(X_{k+1} > r \mid K = k) = P(X > r) \tag{3}$$

To get probability $R = r$, we sum (2) for all k, i.e.,

$$\begin{aligned} P(R = r) &= \sum_{k=1}^{\infty} P(R = r \mid K = k) \times P(K = k) \\ &= P(X > r) \times \sum_{k=1}^{\infty} P(S_k = T - r \mid K = k) \times P(K = k) \end{aligned} \tag{4}$$

We sum it for all k because we do not care how many packets fit in a timeslot. All we care is that, after we have added some number of packets, we still have the unused remainder of size *r*. Strictly speaking, we do not need to sum for all k. Timeslot of a specific size *T* can only accommodate *m* packets, where $\left\lfloor \frac{T}{B} \right\rfloor \le m \le \left\lfloor \frac{T}{A} \right\rfloor$

Now, the summation in (4) denotes the probability that the sum of several packet sizes equals to *T* - *r* without any references to number of packets used in the summation. In other words, this is the probability that any number of packet sizes sums to *T* - *r*. Thus, we have

$$P(R = r) = P(X > r) \times P(S = T - r) \tag{5}$$

We can view *S* as a renewal process with inter-renewal times *X*. Thus, we expect to have one renewal every *E(X)* bytes. The probability that some renewal will occur exactly at epoch *T* - *r* is, therefore, 1/*E(X)*, i.e.,

$$P(S = T - r) = \frac{1}{E(X)} \tag{6}$$

Substituting (5) and (6) into (1), we get

$$E(R) = \sum_{r=1}^{B-1} r \times \frac{P(X > r)}{E(X)} = \frac{1}{E(X)} \sum_{r=1}^{B-1} r \times [1 - F_X(r)] \tag{7}$$

And, for the probability density function, we have

$$f_R(r) = \begin{cases} \dfrac{1 - F_X(r)}{E(X)}, & 0 \le r \le B - 1 \\ 0, & otherwise \end{cases} \tag{8}$$

The amazing result here is that *E(R)* does not depend on the timeslot size. It only depends on the distribution of packet sizes. This agrees very well with our simulations.

As an example, if we assume uniform distribution for *X, A*=64, and *B*=1518, we get

$$E(R)=\frac{B^3-A^3+3A^2-B-2A}{3(A+B)(B-A+1)}=506.52...$$

It follows that the maximum utilization achieved by an ONU is

$$U=\frac{T-E(R)}{T} \tag{9}$$

Obviously, increasing the timeslot size should result in increased utilization. Fig. 7 (plot a) shows the utilization as the function of timeslot size.

But what about the dela y? As was mentioned before, the per-packet delay consists of two components: TDM delay and Burst delay. The TDM delay is proportional to the frame size and, as such, increases linearly with the timeslot size. The Burst delay on the other hand, decreases exponentially as the timeslot increases. The reason for this is that, with smaller timeslots, the utilization decreases (see Equation (9)), but mostly due to fact that shorter bursts now can cause timeslot overflow. Fig. 7 (plot b) depicts a typical combination of TDM and Burst delay as a function of timeslot size.

Next question we ask is how do we choose the optimal timeslot size such that utilization is maximized and average delay is minimized. The power function described in [4] can be employed as a convenient measure of optimality. The power function is defined as

$$P(T)=\frac{U(T)}{D(T)}, \tag{10}$$

where *U(T)* is a utilization as a function of timeslot size
and *D(T)* is a delay as a function of a timeslot size.

The optimal timeslot size would be the one where the power function is maximal. Below we present our reasoning in expecting the power function to have a maximum. TDM delay asymptotically behaves as

$$D_{TDM}(T)\sim C_1\times T \tag{11}$$

where C_1 is a positive constant.

Clearly, we expect TDM delay to be half of the frame size where frame size itself is a multiple of timeslot size. Burst delay $D_{BURST}(T)$ is expected to decay exponentially as a function of timeslot size. Thus, we have

$$D_{BURST}(T) \sim C_2 \times e^{-C_3 T} \tag{12}$$

where C_2 and C_3 are positive constants.

Total packet delay is equal to the sum of TDM and Burst delays:

$$D(T) = D_{TDM}(T) + D_{BURST}(T) \tag{13}$$

The resulting delay as a function of timeslot size is shown in Fig. 7, plot (b). Utilization is calculated (according to Equation 9) and thus behaves as the plot (a) in Fig. 7.

Plot (c) in Fig. 7 shows the expected behavior of the power function (see Equation 10). The functions in Fig. 7 were obtained analytically. They do not represent simulation results, but rather they give an idea of the relative shapes of utilization, delay, and power functions.

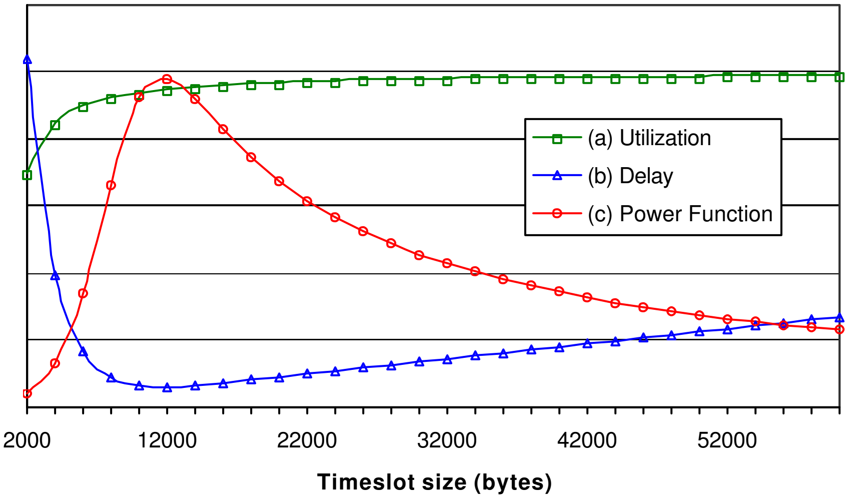


Fig. 7. Optimal timeslot size.

The only parameter in calculating the power function that depends on offered load is Burst delay. The reasonable question to ask is how our optimal operating timeslot size changes when the offered load changes. Fig. 8 presents a family of normalized power functions calculated for various network loads.

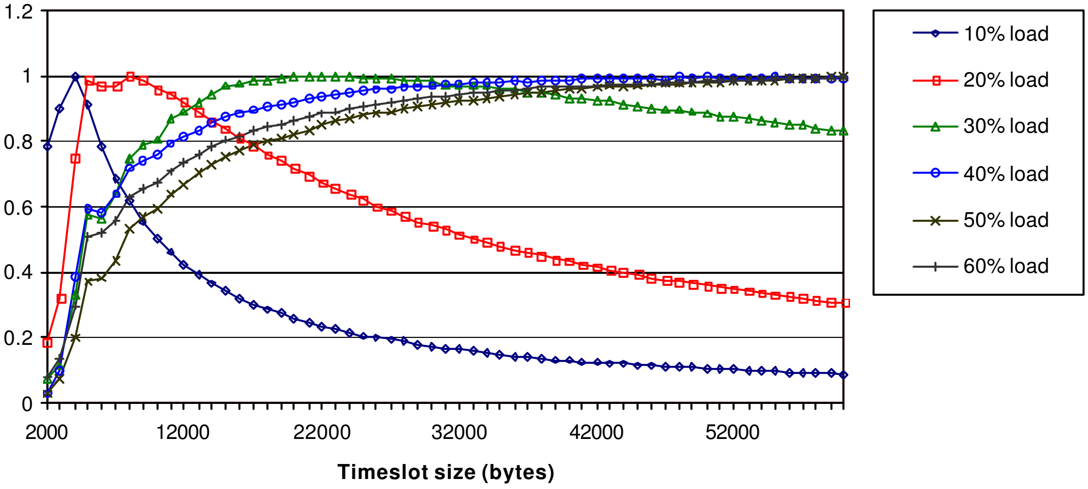


Fig. 8. Normalized power function for different loads.

This is an interesting discovery. Not only does the delay increase exponentially with increased load, but the maximum of the power function also shifts with exponential scale. This means that, if we attempt to readjust the timeslot size in real time to keep the optimal operational point, not only the Burst delay will be exponential, but TDM delay will also increase exponentially. This may have a negative impact on the performance, as the TDM delay affects not only the packets that arrived inside the bursts, but also the packets that arrived between the bursts.

Fig. 9 presents the dependency of the best timeslot size versus the offered load as measured by the power function. This is again to illustrate the exponential dependency. The simulation was performed with timeslots varying from 2000 to 60000 bytes with 1000-byte increments. The figure also shows the extrapolated function to see the timeslot increase above 60000 bytes.

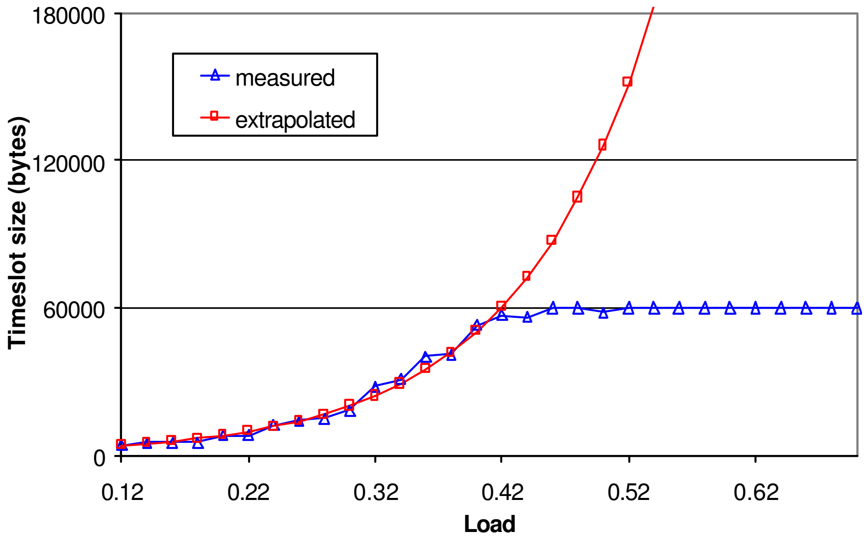


Fig. 9. Dependency of timeslot vs. offered load.

In this section we showed that the ONU could not completely utilize the slotted link available to it. There is an unused remainder at the end of the timeslots. The mean value of the remainder is independent of the timeslot size and only depends on the distribution of packet sizes. This explains why we have a knee of the delay plot for trace C (Fig. 5) around 59% and not around 62.5%. This is because link utilization is approximately 59/62.5 = 94.4%.

We also demonstrated that adjusting the timeslot size could not be a solution to optimizing the utilization-to-delay ratio. The timeslot size would need to be adjusted exponentially with respect to changed load. That would increase the TDM delay for all the packets sent by the ONU. More important, it would also increase the frame size and cause additional TDM delay for packets sent by other ONUs.

There is one more reason why the timeslot size should not be changed, namely for QoS issues. While the Burst delay can be avoided for high priority packets by using clever scheduling schemes, the TDM delay is a fundamental delay that affects all packets. Altering the timeslot size would mean that we are unable to provide any guarantees to the delay value or variation.

In the next section we will consider a scheduling approach in our attempt to reduce the size of the unused remainder, thus increasing the utilization.

6. Scheduling

In our delay and buffer size simulations, we used FIFO queues. A smarter approach may be to attempt to reorder *some* packets waiting in the buffer. If in the previously discussed method, a packet that is currently at the head of the queue does not fit in a partially occupied timeslot, this packet and all following packets will wait for the next timeslot.

However, if some later-arriving packet in the queue is small enough to fit into the current timeslot, then why wait? Fig. 10 illustrates these two approaches. Here three timeslots are needed without packet reordering, but only two timeslots will suffice with reordering.

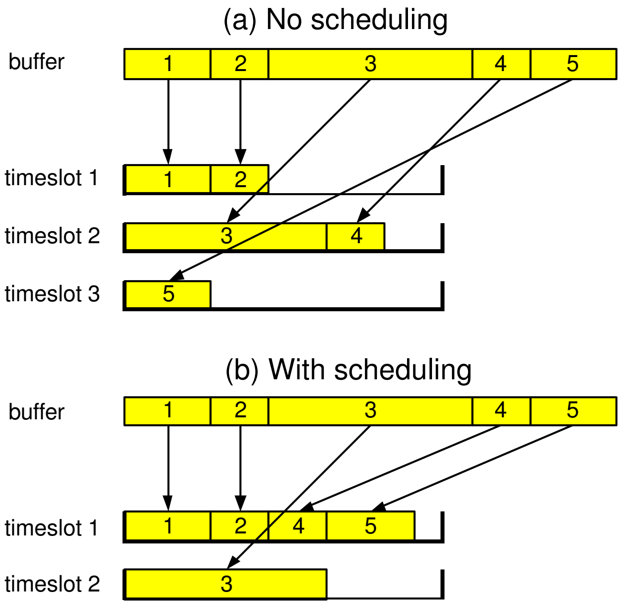


Fig. 10. Illustration of scheduling.

This is a variation of the bin-packing problem. Different flavors of the algorithm may be used: first fit, best fit, prediction, etc. Fig. 11 presents the comparison of link utilizations for "no reordering" scheme (FIFO) and reordering using first fit. These results were obtained by performing simulations with trace C at very high load (approx. 73%). This was done in order to have more timeslots saturated.

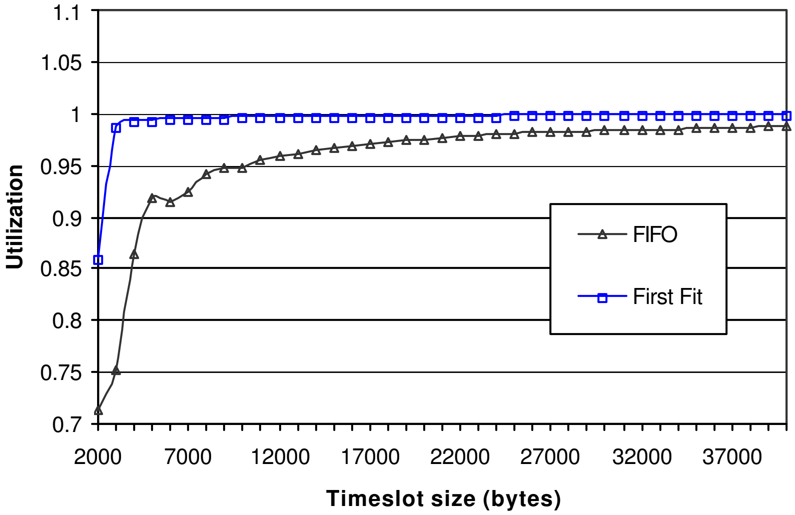


Fig. 11. Maximum link utilization (FIFO vs. First Fit).

This reordering can be easily implemented in hardware as well as in software. While we have not obtained a mathematical equation for *E(R)* in this case, it is easy to see that *E(R)* will depend not only on the packet size distribution, but also on the network load. Indeed, the higher the load, the more packets will be waiting in the queue, and the higher is the possibility of finding one packet that fits in the remainder of the timeslot.

However, as it turns out, first-fit scheduling is not such a good approach. To understand the problem, we need to look at the effects of packets reordering from the perspective of the TCP/IP payload carried by Ethernet packets. Even though TCP will restore the proper sequence of packets, an excessive reordering may have the following consequences:

1) According to the fast retransmission protocol, the TCP receiver will send an immediate ACK for any out-of-order packet, whereas for in-order packets, it may generate a cumulative acknowledgement (typically for every other packet) [11]. This will lead to more unnecessary packets being placed in the network.

2) Second, and more important, packet reordering in the ONU may result in a situation where n later packets are being transmitted before an earlier packet. This would generate n ACKs (n-1 duplicate ACKs) for the earlier packet. If n exceeds a predefined threshold, it will trigger packet retransmission and reduction of the TCP's congestion window size (the *cwnd* parameter). Currently, the threshold value in most TCP/IP protocol stacks is set to three (refer to the Fast Retransmission protocol in [11] or elsewhere).

Even if special care is taken at the ONU to limit out-of-order packets to only one or two, the network core may contribute additional reordering. While true reordering typically generates less than three duplicate ACKs and is ignored by the TCP sender, together with reordering introduced by the ONU, the number of duplicate ACKs may often exceed three, thus forcing the sender to retransmit a packet. As a result, the overall throughput of user's data may decrease.

So, what is the solution? As we mentioned earlier, we assume that the traffic entering the ONU is an aggregate of multiple flows. In the case of business users, it would be the aggregated flows from multiple workstations. In the case of a residential network, we still may expect multiple connections at the same time. This is because, as a converged access network, a PON will carry not only data, but also voice-over-IP (VOIP) and video traffic. Also, home appliances are becoming network plug-and-play devices. The conclusion is that, if we have multiple connections, we can reorder packets that belong to different connections, and never reorder them if they belong to the same connection.

The outline of the algorithm is given in Fig. 12.

```
Let     $Q$ be the queue of packets $q_1, q_2,.. q_n$ waiting in the ONU
        $C(q_i)$ – connection Id of packet $q_i$
        $P$ – set containing Ids of packets that were postponed
        $R$ – slot remainder

Repeat for every timeslot
{
        $i = 1$
        $P \in \varnothing$                         (Clear the set P)
        $R = |timeslot|$
        While $i \le n$ and $R \ge \min$
        {
                If $q_i \le R$ then                 (packet fits into timeslot)
                {
                        if $C(q_i) \notin P$        (i.e. packets from this connection
                                                    were not postponed yet)
                        {
                                send $q_i$.
                                $R = R - | q_i |$
                        }
                }
                else                                (packet does not fit into timeslot)
                        $P = P \cup C(q_i)$         (add connection Id to P)
                $i = i + 1$
        }
}
```

Fig. 12. Algorithm for connection-based packet reordering.

The algorithm in Fig. 12 preserves the order of packets within a connection by keeping track (in set P, see Fig. 12) of all the connection identifiers of packets that were postponed. Obviously, the finer the granularity of connection identifiers, the more reordering possibilities the ONU will have, but more memory would need to be allocated for the set P (which probably should be implemented as a hash table). So, if connection ID is identified only by the source address, then in case of a single user with multiple connections, the ONU will not be able to reorder any packets.

Looking at the destination address instead of the source address may improve the situation for ONUs with a single user, but this has a potential drawback in the situation when multiple users send packets to the same address. In this situation, even though packets are from different senders, the ONU will not reorder them.

A reasonable solution may be to look simultaneously at a *source-destination* pair, plus also to include the source and destination port numbers. Then, the ONU will have maximum flexibility. More studies need to be done to determine the statistical properties of the connections to estimate the advantages of fine granularity connection identifiers.

However, an important point is that the improvement achieved by this scheduling will be between the FIFO case and First Fit case. Clearly, the above algorithm will reorder some packets, which will make its utilization better than in FIFO. It is also true that some packets that belong to the same connection (and that First Fit will reorder) will not be reordered in the given algorithm; thus its performance will be lower than that of First Fit.

But unless we use the extremely small timeslot size (less than 5000 bytes), the difference in link utilization between the FIFO case and First Fit algorithms is only in the range between 1.5% and 4.5 % (refer to Fig. 11). The important question then is whether it makes sense to invest in additional hardware and software cost to implement something that can improve utilization by maybe 4% maximum. It would not make too much economical sense to implement this algorithm if these 4% of improvement should bear the whole cost of implementation. But we should remember that PON is viewed as a technology for full-service access networks. As such it should be able to provide QoS support. It may be either DiffServ, RSVP, or MPLS flow switching. In any case, the ONU should have the ability to reorder packets based on the TOS field in the IP header, Forwarding Equivalence Class (FEC), or some other priority identification. Then adding the algorithm to improve utilization may come at the very low additional cost.

7. Conclusion

In this study we discussed and evaluated design issues that must be dealt with in a PON access network. In Section 4, we investigated the packet delay. We found that the Burst delay considerably exceeded the TDM delay even at very light loads. At higher loads, this difference became even more dramatic. Similar situation was shown for the queue size: large bursts were present at very low average load. These observations led us to a conclusion that packet loss could not be prevented. Having larger buffers will slightly reduce the congestion, but will increase the Burst delay, as more packets will be accumulated during bursts.

Then we investigated ways to increase the bandwidth utilization. In Section 5 we showed that the ONU does not use all the bandwidth available to it. There will be an unused remainder at the end of a timeslot. Interestingly, we found the expected value of this remainder to be independent of the timeslot size (except the timeslots comparable in size with packet sizes). Then, we analyzed the possibility of adjusting the timeslot to find the optimal operating value that optimizes the delay and utilization. We found this approach to be not feasible, as it requires the timeslot adjustment to have exponential magnitude with respect to effective load. That would make the variations of TDM delay to have exponential amplitude.

In Section 6, we considered an alternative approach to improve the utilization: packet scheduling. We showed that the First Fit algorithm may slightly improve the utilization, but will have negative impact on the TCP/IP connection behavior. We then suggested a connection-oriented first-fit algorithm. That algorithm was found to be too computationally expensive weighting against its benefits. However, it may be implemented as part of QoS scheduling.

This study has some shortcomings. First, even though the Bellcore traces are of high quality, the services and usage of the Internet since the time the traces were collected have changed. Second, we have not yet verified the results presented here on a hardware prototype. And finally, we showed that packet loss is unavoidable, but have not yet simulated ONUs with finite buffers and various packet drop policies. This work is still in progress.

Bibliography

[1] G. Pesavento and M. Kelsey, "PONs for the broadband local loop," Lightwave, PennWell, vol. 16, no. 10, pp. 68 – 74, September 1999.

[2] B. Lung, "PON architecture 'futureproofs' FTTH", Lightwave, PennWell, vol. 16, no. 10, pp. 104 – 107, September 1999.

[3] W. Leland, M. Taqqu, W. Willinger, and D. V. Wilson, "On the self-Similar Nature of Ethernet Traffic (extended version)," *IEEE/ACM Transactions on Networking,* vol. 2, no. 1, (February 1994), pp. 1-15.

[4] L. Kleinrock, “On Flow Control in Computer Networks,” *Proc. of ICC’78*, (Toronto, Ontario, June 1978), vol. 2, pp. 27.2.1 - 27.2.5.

[5] B. Mukherjee, *Optical Communication Networks*, McGraw-Hill, New York, 1997.

[6] W. Willinger, M. Taqqu, and A. Erramilli, “A Bibliographical Guide to self-similar traffic and performance modeling for modern high speed networks,” *in Stochastic Networks: Theory and Applications in Telecommunication Networks* (F. P Kelly, S. Zachary, and I. Ziedins, eds.), vol. 4 of the Royal Statistical Society Lecture Notes Series, Oxford University Press, Oxford, 1996.

[7] J. Hammond and P. O’Reilly *Performance Analysis of Local Computer Networks*, Addison-Wesley Publishing Co., Reading, MA, 1986.

[8] P. M. Fiorini, “On Modeling Concurrent Heavy-Tailed Network Traffic Sources and its impact upon QOS,” *Proc. of ICC’99*, (Vancouver, June 1999), vol. 2, pp. 716 – 720.

[9] W. Willinger, V. Paxson, and M. Taqqu, “Self-Similarity and Heavy Tails: Structural Modeling of Network Traffic,” *A Practical Guide to Heavy Tails: Statistical Techniques and Applications*, (R. Adler, R. Feldman, M. S. Taqqu, eds.) Birkhauser, Boston, 1998.

[10] K. Park and W. Willinger, “Self-Similar Traffic: An overview,” (Preprint.) To appear in *Self-Similar Network Traffic and Performance Evaluation*, Wiley-Interscience, 2000.

[11] W. R. Stevens, *TCP/IP Illustrated, Volume 1*, Addison-Wesley Publishing Co., Reading, MA, 1994.