An Analysis of TCP-tolerant Real-time Multimedia Distribution in Heterogeneous Networks

Agnieszka Chodorek[1] and Robert R. Chodorek[2]
[1] Department of Telecommunications and Photonics
Kielce University of Technology
al. Tysiaclecia Państwa Polskiego 7, 25-314 Kielce, Poland
a.chodorek@tu.kielce.pl
[2] Department of Telecommunications
The AGH University of Science and Technology
al. Mickiewicza 30, 30-059 Kraków, Poland
chodorek@kt.agh.edu.pl

1. Introduction

The modern Internet is a heterogeneous network, which utilizes different network technologies. These technologies are optimized to specific environments. They differ in range of transmission, transmission medium (wired, wireless), achievable throughput, costs, etc. Thanks to usage of different network technologies, access to the Internet is possible almost all over the world – from large metropolises to sands of Sahara. However, usage of heterogeneous networks creates many problems, which are unresolved up until now.

Transmission in heterogeneous networks denotes, that a packet is sending via networks, which differ in quality of service. It involves congestions at point of contact of different technologies, what leads to packet losses. Network heterogeneity involves also difficulties in preserving real-time characteristics of transmission (delay, jitter).

Links in heterogeneous networks are shared by transmissions carried out by different applications (bulk data transfer, WWW, multimedia applications, etc.). Modern applications are different in the type of generated traffic (elastic, inelastic) and mechanisms of protocols and architectures used during transmission. As an effect, we observe strong interactions between individual transmissions, which can result in degradation of one or more competing flows. Solutions, which are intended for counteracting these degradation are TCP-friendly protocols [1][5][6][10] and TCP-tolerant mechanisms [2]. The last one will be analyzed in the chapter.

One of possible TCP-tolerant mechanisms is burst control, proposed in the paper [2]. This low-complexity mechanism is based on burst fragmentation and assures avoidance of TCP congestion collapse when competing with real-time transmission, but does not allow for transmission rate reduction when congestions appear.

The aim of the chapter is to analyze TCP-tolerant real-time multimedia distribution in heterogeneous networks. The simulational analysis was carried out in Berkeley's ns-2 environment. The particularly heavy emphasis was placed on quality of service of competing streams, measured by bandwidth, loss and delay parameters.

The rest of the chapter is organized as follows. Section 2 presents problems of coexistence of elastic and inelastic traffic in heterogeneous networks. Section 3 provides an overview of burst control mechanism. Section 4 describes simulational experiment, while section 5 presents an analysis of burst controlled real-time multimedia transmission. Section 6 presents an analysis of buffer occupancy in the case of burst controlled real-time multimedia transmission. Section 7 summarizes our experiences.

2. Coexistence of elastic and inelastic traffic in heterogeneous networks

In IP networks, there are two kinds of traffic, which are widely different in required Quality of Service (QoS) parameters. The criterion of identity is sensitivity to transmission delay and bandwidth. The first, elastic traffic is, generally, insensitive to transmission delay and bandwidth. It is generated by such applications, as ftp, WWW, electronic mail, peer-to-peer file transfer. These applications typically use the TCP transport protocol.
The second, inelastic traffic is, generally, sensitive to both, transmission delay and bandwidth. This kind of traffic, also known as streaming traffic, is generated by such applications, as Voice over IP (VoIP), Video on Demand (VoD), Internet Protocol Television (IPTV), Internet radio, Internet television, videoconferencing tools. These applications typically use the RTP or the UDP transport protocols.

2.1. The TCP transport protocol

Transmission Control Protocol (TCP) is the primary transport protocol of the Internet. The "Encyclopedia of Internet Technologies and Applications" defines the TCP as "*the connection-oriented, multi-purpose transport protocol, designed for reliable data transfer*" [5]. The TCP has several reliability-oriented mechanisms. The most important are:

- error control, intended for reliability assurance of TCP connections,
- flow control, intended for prevention of receiver buffers from overflowing,
- congestion control, intended for counteracting and avoiding congestions.

From sharing traffic point of view, the most important TCP's mechanism is the congestion control. This mechanism allows the TCP connection to react for variable network circumstances, and, as a result, to share the bottleneck link nearly equally. If we look into the problem of the QoS of TCP connections, we will find out, that the equality of throughputs of TCP flows in shred link determines fair bandwidth allocation between competing TCP flows.

2.2. The UDP transport protocol

The User Datagram Protocol (UDP) was designed for early IP networks, which were unicast in nature. It was designed as a very simple transport protocol, completing the TCP/IP protocol suite. However, in contrast to TCP whose mechanisms were optimized from the point of view of unicast transmission, the simplicity of the UDP protocol allows it to be used for multicasting as well.
Ascetic UDP's mechanisms include CRC-based error detection and multiplexing of transport connections (using socket interface). 16-bit checksum secures the UDP datagram (both UDP header and user's data) and the flow state information (IP addresses of end systems, port numbers used by end systems and the protocol identifier). No flow control, congestion control or error control is provided.
Although UDP isn't a "modern protocol" – it was defined a quarter of century ago – the protocol is still in common use. Just like TCP, at the present time UDP is an element of all popular operating systems. However, it is usually used for non-professional or semi-professional applications or (most frequently) serves as an underlying protocol for session-layer or application-layer control protocols or other transport protocols.

2.3. The RTP transport protocol

The Real-time Transport Protocol (RTP) [8] was designed for real-time transmission of multimedia information. Its practical application covers, among other things, interactive audio/video services and real-time distribution of audio and (or) video data to a large

number of recipients. In contrast to UDP, RTP was designed as a multicast (multipoint-to-multipoint) protocol, and unicast (point-to-point) transmission is treated as particular case of general multicasting.
Nowadays, version 2 of the RTP protocol is in common use. This version was introduced by RFC 1889 in 1996 and extended by RFC 3550 in 2003 [8]. Extensions made the protocol better for multimedia transmission to large multicast groups.
Typically, RTP protocol occupies the upper sublayer of the 4th (transport) layer of the OSI model, while the lower sublayer is occupied by UDP. Running RTP on top of UDP allows utilization of the multiplexing and checksum services of the lower protocol.
Real-time multimedia transmission needs fixed (constant or variable) bandwidth and cannot be window controlled. To meet these requirements RTP does not provide any typical window-based mechanism such as flow control or congestion control (instead, RTP-level translator-based congestion control is supported). The real-time transmission doesn't require reliable transmission (however, small BERs are preferred), so the RTP protocol doesn't ensure reliability. However, it detects out-of-order delivery and packet loss (both based on the sequence numbers included in RTP header), as well as packet damage (using checksum service of underlying UDP protocol). Because the receiver is able to repair out-of-order errors, unordered packets are delivered to application in sequence.
RTP protocol carries real-time data and, usually, also conveys additional information allowing the receiver to identify the type of delivered content (e.g. audio, video) and to identify the method of encoding (PCM, ADPCM, MPEG-4, etc.). The timestamp header field, which reflects the sampling instant of the first octet in the carried data, allows the protocol to support synchronisation of audio and video data and reconstruction of the timing. The RTP header also includes a field which identifies synchronization source (SSRC) of the data carried as the payload. If the synchronisation source is a mixer, the packet header will contain the list of identifiers of the contributing sources (CSRC).
RTP itself does not provide any control functions. For the purpose of control, the RTP Control Protocol (RTCP) was developed. RTCP is an integral part of the RTP specification (both RFC 1889 and RFC 3550). The RTCP functionality includes monitoring the quality of service, conveying information about the participants in a RTP session, supporting scalability of large multicast sessions and, optionally, conveying additional control information for the purpose of session management.
Each RTP transmission is associated with an RTCP transmission. In teleconferencing systems, where several data streams are sent independently, each contributing stream is augmented with RTCP flow, allowing individual control of each stream.

2.4. Elastic and inelastic traffic in shared link

In heterogeneous networks, the inelastic, multimedia traffic and the elastic, TCP traffic, are transmitted via the same link. The inelastic, multimedia traffic is self-limited (or rate-limited), i.e. has strictly defined, constant or variable, bit rate. As a result, the real-time multimedia stream has hard bandwidth requirements, but, from the other side, it never occupies more bandwidth than it results from the bit rate of the traffic source. The elastic, TCP traffic is unlimited (in practice, limited by flow and congestion control) and its bit rate strongly depends on network circumstances (bandwidth of bottleneck link, number of competing flows, etc.). As an effect, the TCP flow hasn't hard bandwidth requirements, and, in favorable circumstances, it can occupy whole bandwidth of the bottleneck link.

Although real-time transmission usually doesn't occupy the whole link capacity, in

some circumstances, competing TCP flow can collapse and achieve significantly lower throughput than the network can deliver. This unexpected collapse of TCP connection is identified as TCP-intolerance of real-time transport protocol [2][9]. Moreover, it is possible to observe a satellite phenomenon – real-time intolerance, where real-time stream also achieves significantly lower throughput than it results from a simple difference of link throughput and TCP throughput [3].

Similar term, TCP-unfriendliness, denotes such behavior of inelastic traffic, where one or more inelastic (typically: multimedia) streams are not able to equally share the bottleneck link with one or more TCP flows. In contrast, TCP-friendly streams are able to fairly (equally) share link with the TCP. The well-known definition of TCP-friendliness goes: "*We say a flow is TCP-friendly if its arrival rate does not exceed the arrival of a conformant TCP connection in the same circumstances*" [6]. Because TCP-unfriendliness is caused by lack of TCP-like congestion control mechanism [6], it can be eliminated using protocols, which apply this method of congestion control. Such protocols are called TCP-friendly protocols [1][5][6]. Three main properties of TCP-friendly protocols are [2][9]:

- avoidance of TCP congestion collapse,
- fairness toward competing flows,
- congestion control integrated with the transport protocol.

TCP-friendly protocols change bit rate of the flow and do not change (neither physically nor virtually) the bit rate of the source of real-time traffic [2]. Previous research shows, that if fair bandwidth allocation does not correspond with real-time requirements, the last two advantages of TCP-friendly protocol may cause degradation of real-time characteristics of the streaming traffic [4]. In result, it is impossible to assure QoS parameters of the real-time stream (throughput, latency, jitter, error rate) at acceptable level when TCP-friendly protocols are used. However, QoS parameters of the TCP flow (throughput, error rate) and link utilisation stays at high level.

In contrast to the TCP-friendliness, which is intended to assure fairness, TCP-tolerance is intended to preserve (as good as possible) primitive character of the traffic in the best-effort service. TCP-tolerance can by defined as such behaviour of transport protocol, which allows TCP to utilize bandwidth unoccupied by real-time multimedia transmission [2]. In other words, it is an ability of real-time transport protocol to reasonably share network resources with TCP flows. TCP-tolerance can be understood as the first attribute of TCP-friendly protocols (avoidance of TCP congestion collapse).

Unlike the TCP-friendliness, TCP-tolerance does not implement congestion control. However, it can be used with specialized congestion control mechanisms, optimized for multimedia transmission (as adaptive coding, translators, receiver-based layered multicast or receiver-based multicast stream replication).

3. Burst control for TCP-tolerant applications

Burst control spreads out burst over time, while the source's bit rate stays unchanged. It results in debursting, similar as obtained as an effect of traffic smoothing (or traffic shaping). However, in contradiction to traffic smoothing, burst control doesn't limit the rate of transmission of data.

If encoder generates video frames at 25 Hz (25 frames per second), the period ΔT between two successive frames will be constant ($\Delta T = 40$ ms). Due to usage of real-time transport protocol, at the output of the sender, we observe burst of packets. Let's assume, that the i-th burst consists of B_i packets, where B_i is the smallest integer $\geq \frac{f_i}{p}$ (where f_i is i-th frame size and p is a payload length).

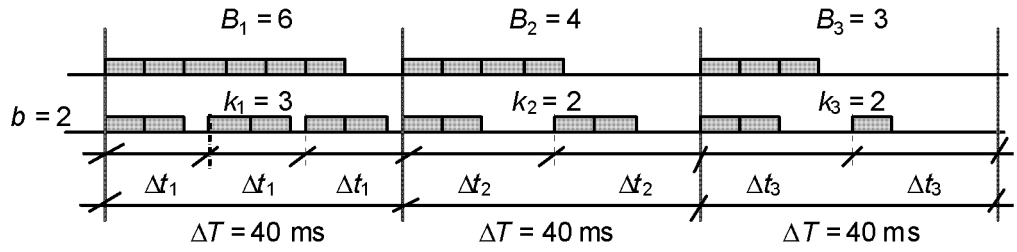

Figure 1: Burst control.

Burst control mechanism allows the sender to transmit multimedia stream at original bit rate (constant or variable), with fully controlled burstiness. The mechanism subdivides the period ΔT on k non-overlapping blocks (time slots) of size Δt. In each time slot, the sender sends burst of b packet (Fig. 1), except the last, where burst can be smaller or equal to b. Because burst is spreading out over ΔT, the source's bit rate (typically evaluated over ΔT period) stays unchanged.

The number of time slots k_i and the duration of time slot Δt_i depend on video source and the method of encoding. In particular:

$$\Delta t_i = \Delta T \cdot \frac{1}{C\left(\frac{f_i}{p \cdot b}\right)} \tag{1}$$

where $C(x)$ is a round toward infinity (the smallest integer greater than or equal to x) and p is a data packet's payload length.

Parameters p and b depends on applications. The p is set according to separate specifications of used real-time protocols (e.g. according to RTP profiles). The value of b should be estimated as trade-off between application requirements and TCP-tolerance. Tests carried by the authors show, that b set to 1 usually assures the best TCP-tolerance [2].

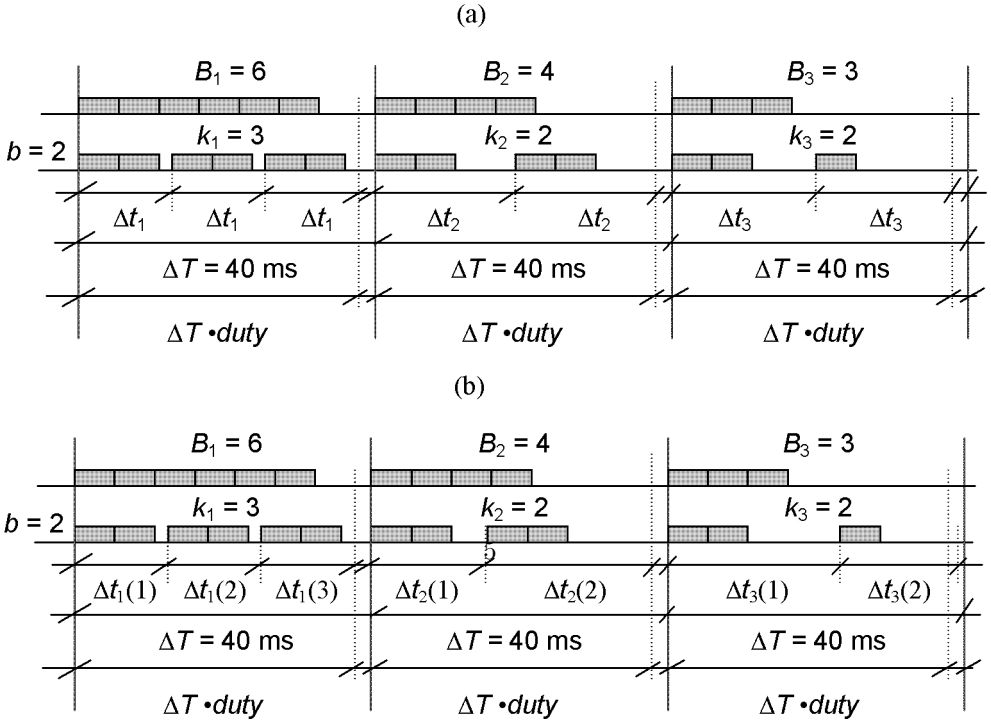

Figure 2: Burst control with duty factor: a) without randomization, b) with randomization of duration of time slots.

Burst fragmentation doesn't assure avoidance of synchronization of bursts generated from different sources. To avoid synchronization, two complementary mechanisms have been introduced:

- duty factor,
- randomization of duration of time slots.

Duty factor *duty* reduces the period ΔT to $\Delta T \cdot duty$ (Fig. 2a). In result, the duration of time slot Δt_i can be expressed by:

$$\Delta t_i = \Delta T \cdot \frac{1}{C\left(\frac{f_i}{p \cdot b}\right)} \cdot duty \tag{2}$$

This method is especially useful in the case of synchronized CBR sources, which are located in the same node (e.g. layered multicast, stream replication multicast). To avoid synchronization of bursts (and preserve synchronization of streams), streams (layers) should be transmitted using different values of *duty*. The duty factor can be set from 0 to 1, where 1 denotes usage of nominal time ΔT and 0 denotes $\Delta t_i = 0$ (thus, $b_i = B_i$). In practice, values of *duty* close to 1 are preferred [2].
In some cases, duty factor insufficiently reduces dangerous of synchronization. Thus, the other method – randomization of duration of time slots (Fig. 2b) – should be applied together with duty factor method. Time slots can be randomized, for example, by addition of small random value (positive or negative) or by addition of small percentage (e.g. 1%) of original value Δt_i computed from equation (2). In the chapter, randomized time slots Δt_i' have been computed according the equation:

$$\Delta t_i^{'} = \Delta t_i \cdot (1 + U\,(-0.01,\ 0.01)) \tag{3}$$

where $U\,(-0.01,\ 0.01)$ is uniformly distributed random number between –0.01 and 0.01.

The duty factor *duty* allows for easy randomization of duration of time slots. It gives the margin necessary to reduce the effects of cumulation of positive random components from equation (3), which can appear because of relatively small population of random variables (see Δt_3 in Fig. 2b).

4. Experiments

Burst control mechanism was implemented in Berkeley's *ns*-2 network simulator [7]. Experiments have been carried out using typical single-bottleneck topology, which consists of two routers and a set of senders and receivers. In all experiments throughput of bottleneck link was large enough to ensure live video transmission.

The transmission scheme has been simulated in a 4 different topologies (Fig. 3), to explore the performance issues as well as scalability. Senders, as well as receivers, were connected to the nearest router at 100 Mbps and 1 μs delay. Senders' buffers are theoretically unlimited (5000 packets). Routers were connected with a bottleneck link at 10 Mbps and τ_p propagation delay (default value of τ_p is 5 ms). In routers, Drop Tail queue was used and queue size was set to 5000 packets (infinite buffer size) or to 13 packets (finite buffers; queue assure buffering of 10 ms TCP traffic).

Topology $T1$ consists of a single video source (SV) and single receiver (R1). Topology $T2$ consists of a video source (SV), a TCP source (ST), video receiver (RV), and TCP receiver (RT). Topologies were used for analysis of mechanism's performance with ($T2$)

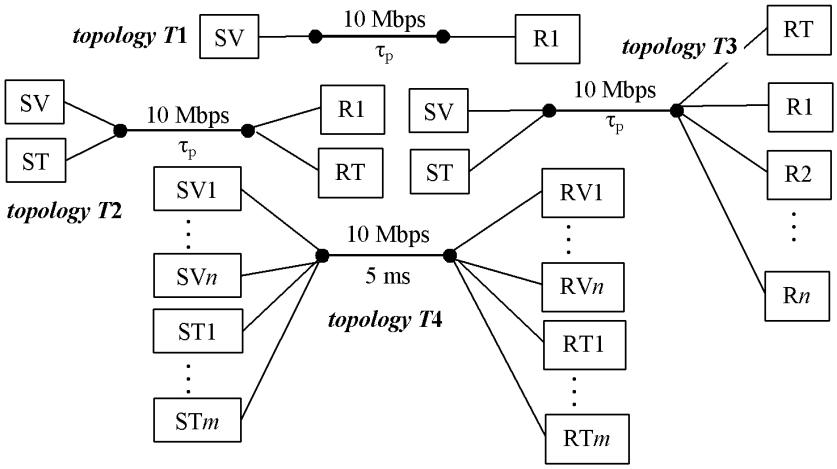

Figure 3: Topologies used in experiments.

and without ($T1$) background traffic. Topology $T3$ extends topology $T2$ with multiple video receivers. It was used for exploration of the scalability of the multicast transmission with respect to session size.
Topology $T4$ extends topology $T2$ with multiple sessions. The number of senders/receivers was varied from 1 to 10 to investigate the intra-session behaviour of burst controlled multimedia.

Both, elastic (TCP) and inelastic (video) traffic was modeled using build-in models. As the source of elastic traffic (ST), FTP over TCP (SACK version) was used. TCP packets have a maximum segment size (MSS) of 1000 bytes. As a model of video (inelastic traffic), constant bit rate (CBR) streams were used. CBR stream was transmitted using RTP [8]. Maximum payload size of RTP packets was set to 188 bytes (small packets) or 1000 bytes (large packets).

Each experiment consists of six phases:

- 3 Mbps CBR over RTP, without background TCP traffic (topology $T1$), infinite buffers,
- 3 Mbps CBR over RTP, with background TCP traffic (topology $T2$), infinite buffers,
- 3 Mbps CBR over RTP, with background TCP traffic (topology $T2$), finite buffers,
- 6 Mbps CBR over RTP, without background TCP traffic (topology $T1$), infinite buffers,
- 6 Mbps CBR over RTP, with background TCP traffic (topology $T2$), infinite buffers,
- 6 Mbps CBR over RTP, with background TCP traffic (topology $T2$), finite buffers.

The simulation was run for 5 (simulated) minutes. As measures of quality of transmission, bandwidth, loss and delay parameters were used. In particular, QoS of the real-time stream was measured by throughput, goodput, end-to-end delay, delay variation and packet error rate. QoS parameters of the TCP flow includes throughput, goodput and error rate.

5. An analysis of burst controlled real-time multimedia transmission

In the section, intra- and inter-session behavior of burst controlled multimedia will be discussed. In intra-session experiments we explore, how the RTP payload length and propagation delay impact the QoS measures (obtained for both, elastic and inelastic traffic). In inter-session experiments QoS measures were observed as a function of a number of competing flows and as a function of bit rate of single multimedia stream.

5.1. QoS measures as a function of RTP payload length

In experiment, we vary RTP payload length from 100 B to 64 000 B, while TCP's maximum segment size was set to 1000 B. The propagation delay at the bottleneck link was set to default value 5 ms. Because the buffer size is expressed in packets, RTP payload length has strong impact in TCP-unfriendly behavior of RTP protocol [3].

In the experiment we observe the general ability of burst controlled RTP to transmit video traffic in real-time.

For all analyzed conditions, TCP-tolerant system behaves stable for any value of the RTP payload length. Achieved goodput have arisen directly from the source's bit rate and doesn't depend on RTP payload length (Fig. 4a). Larger RTP throughput in the case of smaller values of payload length (Fig. 4b) is caused by larger influence of overheads. Overheads have also influence on end-to-end delay.

In contrast, TCP-intolerant system (without burst control mechanism – Fig. 4c, d) behaves unstable for small and medium values of RTP payload length. This instability is characterized by low goodput/throughput, which results from typical for TCP-intolerance (and TCP-unfriendliness) large packet losses. For example, in the burst controlled system, packet losses (RTP+TCP) never exceeds 3%, while in the TCP-intolerant system we observe losses larger than 80%. It's worth remark, that applying of burst control also decreases the delay variation. The graph of delay variation vs. RTP payload length shows that delay variation of the TCP-tolerant system is usually at least 10 times smaller than delay variation of the system without burst control (even if observed in the TCP-tolerant regions of characteristics of the system without burst control). It is caused by less asynchronous character of burst-controlled traffic.

However, the suggestion that burst control is unnecessary – it will be enough to increase RTP payload to achieve TCP-tolerance – is only theoretical. In Fig. 4c,d, 3 Mbps CBR stream achieves stability for RTP payloads equal to or larger than 1500 B, while 6 Mbps CBR stream achieves stability for RTP payloads equal to or larger than 5000 B. In practice, RTP payloads are limited by network MTUs and practical limitation (taken from multiple field trials) of RTP payload (as well as TCP's MSS) is about 1400 B in the case of long-distance transmission and no more than 1500 B for site-local transmission.

Characteristics of TCP throughput vs. RTP payload length (Fig. 4e,f) shows that burst control gives good TCP-tolerance for any value of the RTP payload length. In the case of finite buffers (13 packets) TCP's throughput/goodput is always larger than in the case of the system without burst control. Even in the worst case (6 Mbps CBR, RTP payloads equal to 100..150 B) it is possible to achieve TCP transmission (0.54..1.2 Mbps), while in the system without burst control TCP transmission totally collapses (6..9 kbps).

In the second experiment we analyze multicast session scalability. Video session size was varied according to topology $T3$. In the three experiments ($n = 5$, $n = 10$ and $n = 30$) we obtained the same graphs as in the case of single receiver (topology $T2$).

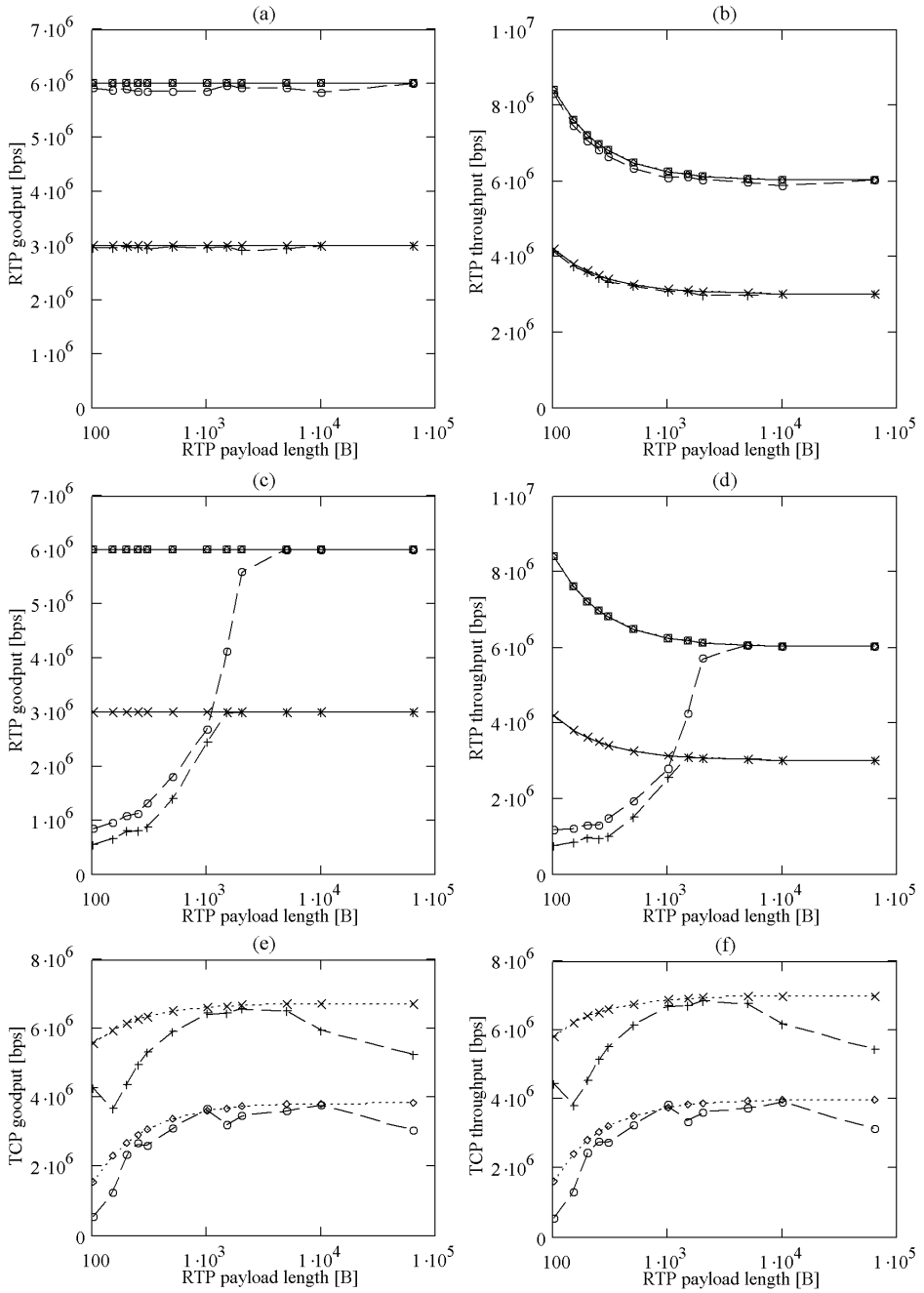

Figure 4: Goodput and throughput as a function of RTP payload length: a,b,c,d) RTP protocol, e,f) TCP protocol, a,b,e,f) with burst control, c,d) without burst control.
Legend: 3 Mbps CBR over RTP transmission: without background TCP traffic, infinite buffers (solid line, no additional symbol), with background TCP traffic, infinite buffers (dotted line, $\times$), with background TCP traffic, finite buffers (dashed line, $+$); 6 Mbps CBR over RTP transmission: without background TCP traffic, infinite buffers (solid line, $*$), with background TCP traffic, infinite buffers (dotted line, $\Diamond$), with background TCP traffic, finite buffers (dashed line, o).

5.2. QoS measures as a function of propagation delay

We repeated experiments of the previous subsections but vary the propagation delay of bottleneck link from 1 ms to 1 s, while RTP payload length was set to 1000 B (and is equal to TCP's MSS).

Although propagation delay have, generally, small influence on RTP transmission, longer delays at the bottleneck link increases delays in the TCP's congestion control loop. TCP becomes less aggressive, what can be important in the case of investigations of RTP/TCP coexistence in the same link.

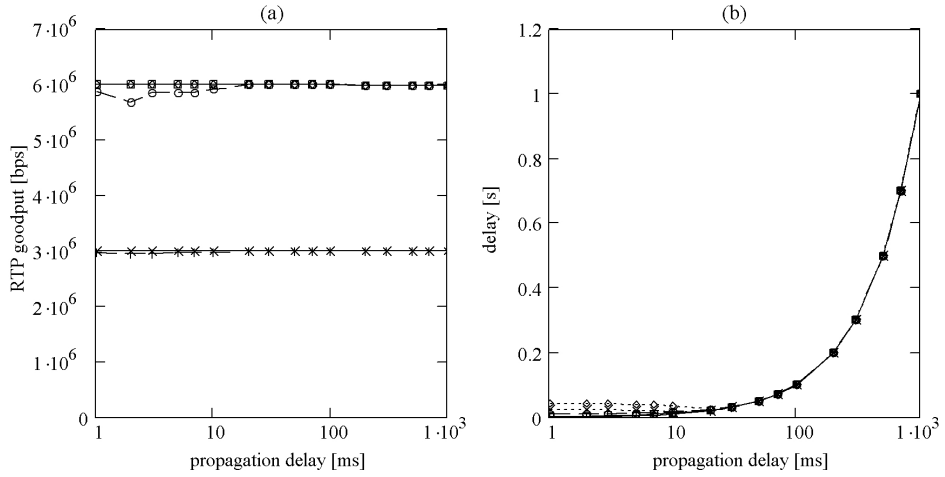

Figure 5: QoS measures of RTP transmission: a) RTP goodput vs. propagation delay, b) mean end-to-end delay vs. propagation delay. Legend: as in Fig. 4.

Results shows, that TCP-tolerant system behaves stable for any value of the propagation delay (Fig. 5a). As previously, achieved goodput/throughput of RTP arises directly from the source's bit rate and doesn't depend on the propagation delay. Small packet losses (less than 5%) are observed only for small and medium delays and for delays greater than or equal to 20 ms packet losses are close to 0 (0.0004% and less). In contrast, if we test TCP-intolerant system, RTP packet losses are stable and keep at constant level (about 50% for 3 Mbps CBR traffic and about 75% for 6 Mbps CBR traffic – typical behavior of TCP-intolerant/TCP-unfriendly system).

Propagation delay has strong dependency on time-based QoS measures. End-to-end delay is directly proportional to propagation delay (Fig. 5b) and there is no visible difference between characteristics of TCP-tolerant and TCP-intolerant system. However, for propagation delays less than 20 ms, the delay variance ranges from about $2{\cdot}10^{-7}$ to about $7{\cdot}10^{-6}$ and is about 10 times smaller than in the case of TCP-intolerant transmission. For propagation delays greater than or equal to 20 ms, delay variance of TCP-tolerant system is always smaller than $2{\cdot}10^{-7}$ (10..100 times smaller than observed for TCP-intolerant multimedia).

TCP's QoS characteristics have shape typical for TCP transmission (Fig. 6). In the case of TCP-tolerant system transmission was stable for all tested propagation delays,

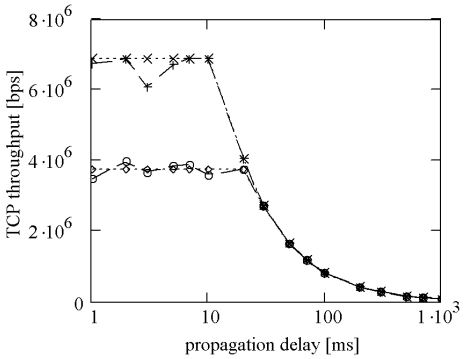

Figure 6: TCP throughput vs. propagation delay. Legend: as in Fig. 4.

while in the case of TCP-tolerant system stability was achieved only for very large delays (greater than or equal to 100 ms). However, TCP's performance in this region of characteristics is, typically, very low.

As previously, in the second experiment we analyze multicast session scalability. In the network of topology $T3$ we randomly chose propagation delays of links between receivers and their nearest router. In the three experiments ($n = 5$, $n = 10$ and $n = 30$) we observe, that characteristics of QoS measures vs. propagation delay are strictly the same as in the case of single receiver (topology $T2$).

5.3. Inter-session experiments

In the first experiment, i TCP flows (i =1,2,...,10) compete for 10 Mbps bandwidth with j CBR video streams (j = 1,2,...,10), 1 Mbps each. It's worth remarking, that if j =10, the network isn't well-dimensioned for live video transmission.
Results shows, that for all tested i and j, burst control is able to preserve real-time character of the multimedia stream (Fig. 7b). Moreover, burst control always manifest strong TCP-tolerant behaviour – TCP is able to perform reasonably fair and stable transmission (Fig. 7a). In the case of multimedia transmission without burst control (Fig. 7c,d), we observe degradation of both, TCP and RTP, flows.

In the second experiment, one ftp flow competes for 10 Mbps bandwidth with one CBR video stream. The target bit rate of video source varies from 1 Mbps to 10 Mbps. Results depicted in Fig. 8, show that TCP-tolerant transmission scheme is able to preserve real-time character of multimedia transmission and, simultaneously, is able to avoid collapse of TCP transmission (Fig. 8a). Such a situation is practically impossible if the real-time system doesn't implement TCP-tolerance (Fig. 8b). These properties of burst-controlled multimedia are observed even if bit rate of video source is close to the capacity of the bottleneck link.

It's worth remarking, that obtained results are better than results of typical TCP-friendly transmission (Fig. 9). TCP-friendliness achieved by TFRC protocol [10] will not influence on TFRC real-time characteristics only if there is possibility to fair bandwidth allocation (bit rate of video source is equal or less than $\frac{1}{2}$ bottleneck link capacity).

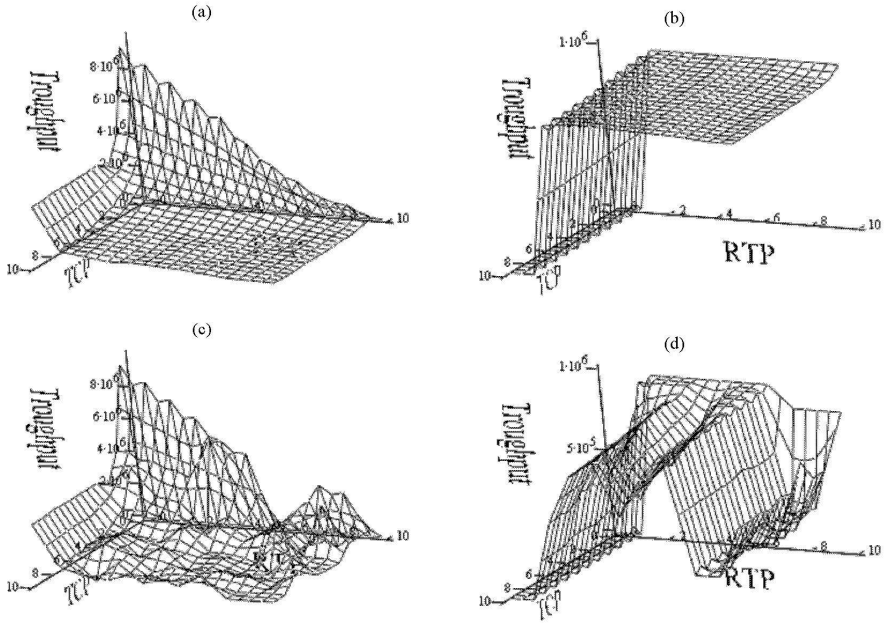

Figure 7: QoS measures as a function of a number of competing flows: a) TCP throughput in system with burst control, b) RTP throughput in system with burst control, c) TCP throughput in system without burst control, d) RTP throughput in system without burst control.

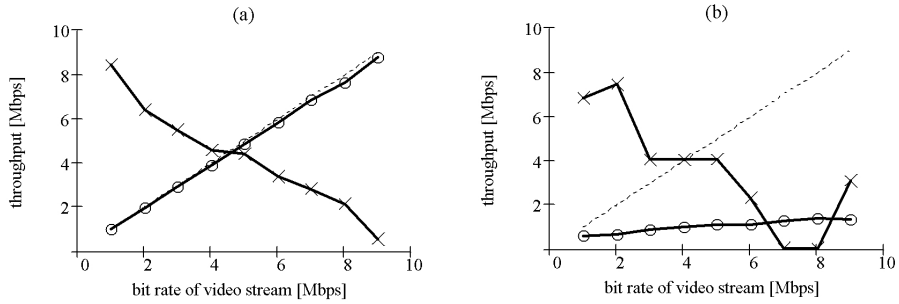

Figure 8: Throughput of RTP (o's) and TCP (x's) as a function of bit rate of video stream: a) system with burst control, b) system without burst control. Dotted line – throughput equal to bit rate of video stream.

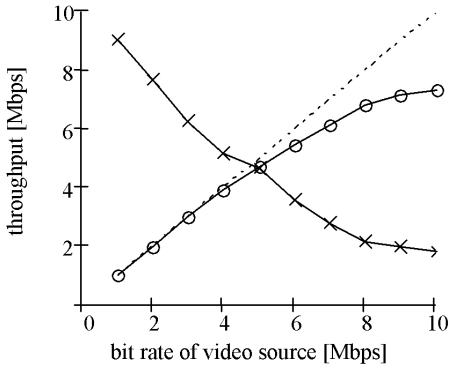

Figure 9: Throughput of TFRC (o's) and TCP (x's) as a function of bit rate of video stream [4]. Dotted line – throughput equal to bit rate of video stream.

6. Problems of buffer occupancy in multimedia content distribution

In the section we discuss a problem of buffer occupancy during the burst controlled multimedia transmission. In experiments we explore, how the RTP payload length, and propagation delay and bitrate of video stream impact the mean and variance of buffer occupancy.

6.1. Mean and variance of buffer occupancy as a function of RTP payload length

Like in the Section 5.1, we vary RTP payload length from 100 B to 64 000 B, while TCP's maximum segment size was set to 1000 B. The propagation delay at the bottleneck link was set to default value 5 ms.

In the case of CBR video over RTP without background TCP traffic and infinite buffers on bottleneck link, mean buffer occupancy is smaller for the system, which implements burst control (Fig. 10a, b). It is especially visible for small values of RTP payload length, where buffer occupancy is hundreds times smaller for burst controlled RTP than for transmission without burst control. In the case of large values of RTP payload length (equal to or larger than 10 000 B), difference between system with and without burst control is rather small. However, RTP payloads are limited by network MTUs and practical limitation (taken from multiple field trials) of RTP payload (as well as TCP's MSS) is about 1400 B in the case of long-distance transmission and no more than 1500 B for site-local transmission.

In the case of CBR video over RTP with background TCP traffic and both, infinite (Fig. 10c,d) and finite buffers on bottleneck link, mean buffer occupancy of the system with burst control is close to the mean buffer occupancy of the system without burst control.

For all six experiments, variance of buffer occupancy always was smaller for burst controlled multimedia transmission, if the RTP payload length does not exceeds 10 000 B (Fig. 11a,b). This is caused by less asynchronous character of burst-controlled traffic.

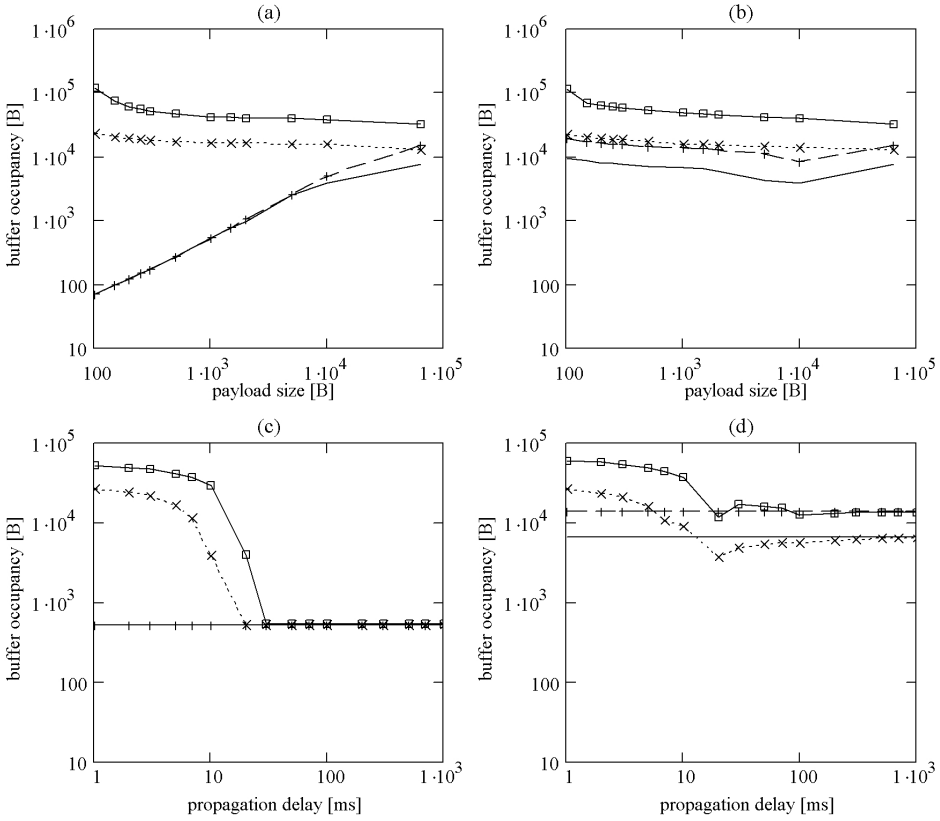

Figure 10: Mean buffer occupancy vs.: a,b) RTP payload length, c,d) propagation delay; a,c) with burst control, b,d) without burst control.
Legend: 3 Mbps CBR over RTP transmission: without background TCP traffic (solid line, no additional symbol), with background TCP traffic (dotted line, $\times$); 6 Mbps CBR over RTP transmission: without background TCP traffic (dashed line, $+$), with background TCP traffic (solid line, $*$).

6.2. Mean and variance of buffer occupancy as a function of propagation delay

Like in the Section 5.2, we vary the propagation delay of bottleneck link from 1 ms to 1 s, while RTP payload length was set to 1000 B (and is equal to TCP's MSS).
Let's remind, that although propagation delay have rather small influence on RTP transmission, longer delays at the bottleneck link increases delays in the TCP's congestion control loop. Thus, in the case of longer delays TCP becomes less aggressive, what is observed in Fig. 10 and Fig. 11 as a stabilization of characteristics at the level determined by transmission without background traffic and with infinite buffers on bottleneck link. In the case of all experiments, burst controlled transmission leads to:

- comparable mean buffer occupancy for smaller values of propagation delay (Fig. 10c,d),
- smaller mean buffer occupancy for larger values of propagation delay (Fig. 10c,d),
- significantly smaller variation (Fig. 11c,d),

when compared with the system which doesn't implement burst control.

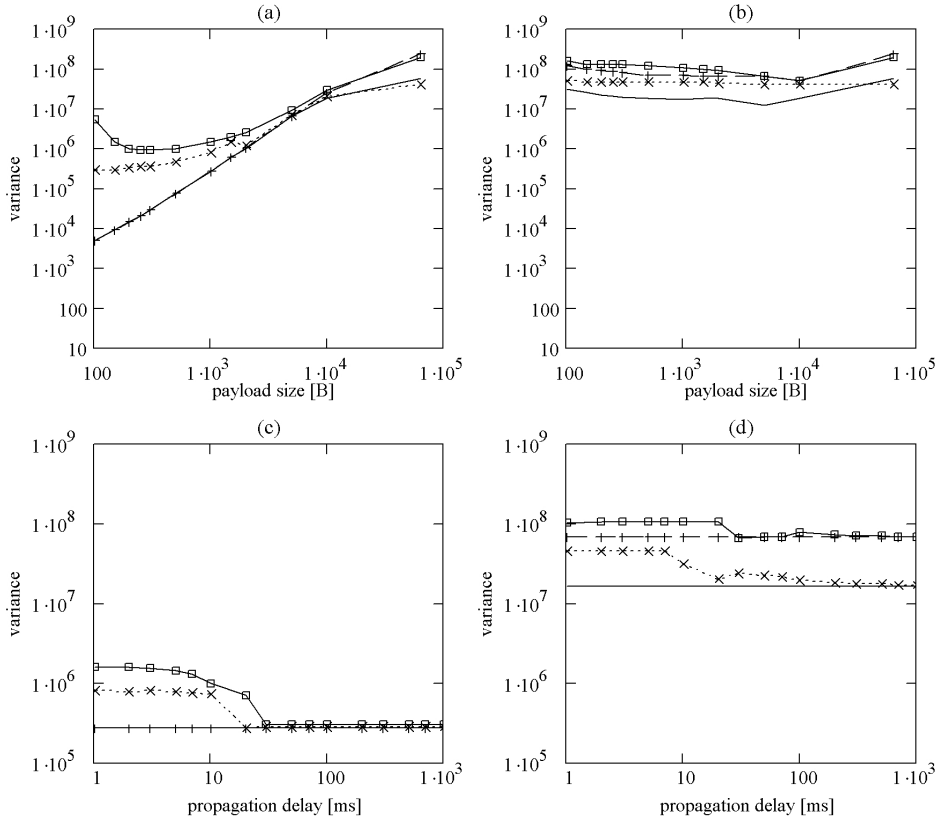

Figure 11: Variance of buffer occupancy vs.: a,b) RTP payload length, c,d) propagation delay; a,c) with burst control, b,d) without burst control. Legend: as in Fig. 10.

6.3. Mean and variance of buffer occupancy as a function of bit rate of video stream

In the last experiment, one ftp flow competes for 10 Mbps bandwidth with one CBR video stream. The target bit rate of video source varies from 1 Mbps to 10 Mbps. Results depicted in Fig. 12 show, that TCP-tolerant transmission scheme gives comparable mean buffer occupancy as the system without burst control. However, if the network is well-dimensioned for live video transmission (in Fig. 12 – bit rate of video stream should be less or equal to 9 Mbps), the burst controlled multimedia transmission is characterized by significantly lower variation of buffer occupancy.

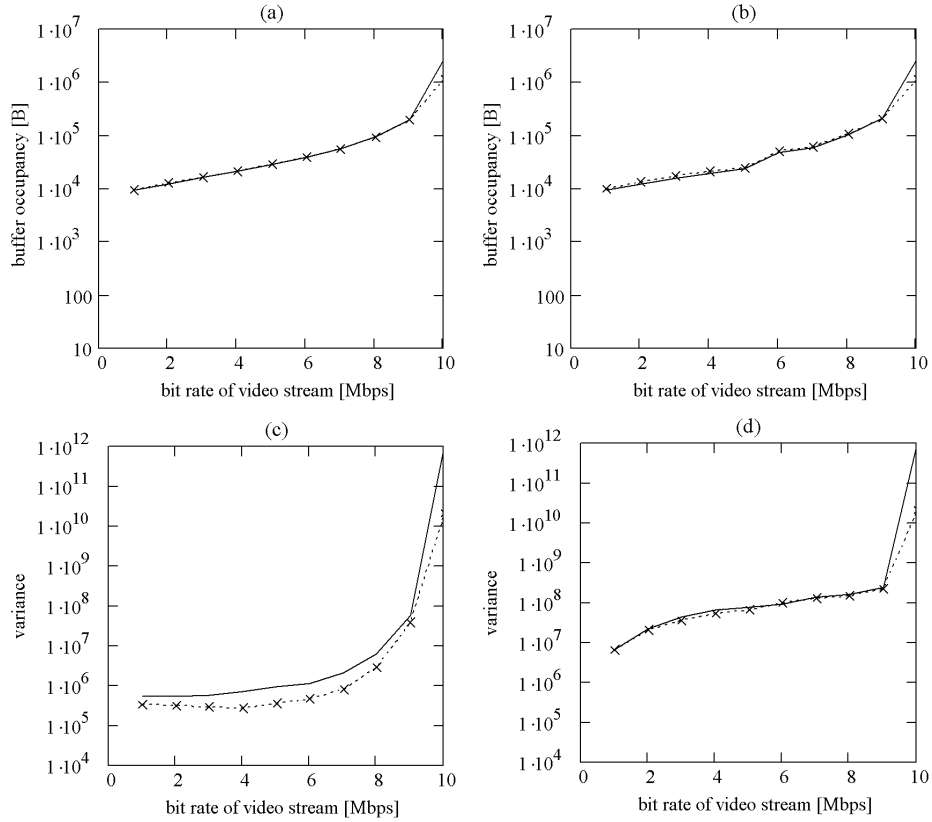

Figure 12: Statistical characteristics of buffer occupancy as a function of bit rate of video stream: a,b) mean buffer occupancy, c,d) variance; a,c) with burst control, b,d) without burst control. Legend: RTP payload length equal to 1000 B (solid line, no additional symbol), RTP payload length equal to 188 B (dotted line, $\times$).

7. Conclusions

TCP-tolerance can by defined as such behavior of transport protocol, which allows TCP to utilize bandwidth unoccupied by real-time multimedia transmission. TCP-tolerant transmission scheme is able to preserve real-time character of multimedia transmission

and, simultaneously, is able to avoid collapse of TCP transmission. One of possible TCP-tolerant mechanisms is burst control, which spreads out burst over time, while the source's bit rate stays unchanged.

An analysis of bust-controlled TCP-tolerant transmission, presented in the chapter, shows, that applying burst control to real-time transport protocols results in growth in the quality of service of competing streams (if compared with the best-effort), measured by bandwidth, loss and delay parameters. Moreover, the TCP-tolerance smoothes buffer occupancy, what is observed as significantly lower variation of buffer occupancy when compared with TCP-intolerant system. Last, but not least advantage of burst control is very good scalability of multicast connections.

8. Acknowledgements

This work is partially supported by Polish Government under Grant No. N517 012 32/2108 (years 2007-2009).

9. Literature

[1] Braden B., Clark D., Crowcroft J., Davie B., Deering S., Estrin D., Floyd S., Jacobson V., Minshall G., Partridge C., Peterson L., Ramakrishnan K., Shenker S., Wroclawski J., Zhang. L.: "Recommendations on Queue Management and Congestion Avoidance in the Internet". RFC 2309. April 1998.

[2] Chodorek A., Chodorek R. R.: "Burst control". Proc. of SympoTIC'04 October 24-26, 2004 – Bratislava, Slovakia. (IEEE Press Cat. No. 04EX877)

[3] Chodorek A.: "Coexistence of elastic and inelastic traffic for different buffer size in congested node". (chapter in book in Polish), WNT, Warsaw 2004.

[4] Chodorek A.: "Streaming video with TFRC – simulation approach". Proc. of SympoTIC'04 October 24-26, 2004 – Bratislava, Slovakia. (IEEE Press Cat. No. 04EX877)

[5] Chodorek A.: "TCP and TCP-friendly protocols". Chapter of the book: Freire M., Pereira M. (eds.) "Encyclopedia of Internet Technologies and Applications", Information Science Reference, 2008.

[6] Floyd S., Fall K.: "Promoting the Use of End-to-End Congestion Control in the Internet". IEEE/ACM Transactions on Networking, August 1999.

[7] http://nsnam.isi.edu/nsnam/index.php/Main_Page

[8] Schulzrinne H., Casner S., Frederick R., Jacobson V.: RTP: "A Transport Protocol for Real-Time Applications". RFC 3550. July 2003.

[9] Chodorek A., Chodorek R. R.: "An analysis of TCP-tolerant real-time multimedia distribution", in Proc. HETNETs 2008, Karlskrona, Sweden, February 2008.

[10] Floyd S., Handley M., Padhye J.: "TCP Friendly Rate Control (TFRC): protocol specification" Internet-Draft draft-ietf-dccprfc3448bis-06.txt, March 2008.