Bjørnar Libæk

Congestion Control for Delay Sensitive Unicast Multimedia Applications in Datagram Networks

Thesis for the degree of Philosophiae Doctor

Trondheim, Desember 2013

Norwegian University of Science and Technology
Faculty of Information Technology, Mathematics and Electrical Engineering
Department of Telematics

NTNU – Trondheim
Norwegian University of Science and Technology

NTNU
Norwegian University of Science and Technology

Thesis for the degree of Philosophiae Doctor

Faculty of Information Technology, Mathematics and Electrical Engineering
Department of Telematics

ISBN 978-82-471-4779-5 (printed ver.)
ISBN 978-82-471-4780-1 (electronic ver.)
ISSN 1503-8181

Doctoral theses at NTNU, 2013:321

Printed by NTNU-trykk

Abstract

This thesis emphasizes the need for congestion control mechanisms for multimedia flows requiring both low jitter and low end-to-end delays. Such mechanisms must not only satisfy the user in terms of high Quality of Experience (QoE), but also ensure that the underlying network is stable. It is important that the network does not end up in a thrashing state, where resources are wasted on work that does not contribute to the user's QoE.

From historically being a transport level mechanism, congestion control functionality for multimedia applications is now located in several layers of the protocol stack. Inelastic applications without the ability to adapt its sending rate requires network level support in order to avoid thrashing, in form of a Quality of Service (QoS) architecture. On the other hand, elastic applications can use end-to-end congestion control in order to reach the same goal. Congestion control functionality for such rate adaptive sources can be divided into a transport level component, and an application level component. The former ensures that the flows do not exceed the estimated available network capacity, maintaining fairness and network stability. The latter ensures that the QoE is maximized given the restrictions of the underlying transport level congestion control. The thesis argues that the two components should be implemented separately in order avoid a myriad of protocols with interfering regulation schemes.

The possibility of using the recently standardized Pre-Congestion Notification (PCN) architecture in dynamic mobile ad-hoc network (MANET) environments is studied. Due to the limited bandwidth and varying conditions, QoS mechanisms, especially admission control, are important when sending inelastic multimedia flows over a MANET. PCN admission control is originally designed for wired DiffServ networks. The thesis shows however that with some modifications, PCN is an alternative. Most importantly, the use of probing on ingress-egress paths which have been idle over a longer period significantly reduces the amount of signaling. In addition, the probing provides fresh information about the congestion status on a path, which is imperative when the admission decision is being made.

Within the area of transport level congestion control, the use of packet pairs for estimating the available bandwidth is studied. The packet pair algorithm was originally designed for estimating the bottleneck link capacity, while later, it has

been modified to also take into account the cross traffic. By using existing video packets as packet pairs, an estimate of the available bandwidth is achieved without inserting any extra packets in the network. Unfortunately, the estimate often suffers from inaccuracies. Therefore, a congestion control algorithm was proposed, named Packet Pair Assisted Congestion Control (PPACC), which uses the packet pair estimates merely as guidelines instead of directly controlling the sending rate. PPACC is compared to TCP-Friendly Rate Control (TFRC), and simulations shows that the proposed protocol outperforms TFRC, with significantly lower packet loss, delay and jitter. PPACC is not designed to be Transmission Control Protocol (TCP)-friendly, so it is not a candidate for large scale use over the open internet. However, it its use is more attractive in isolated environments such as within a DiffServ Assured Forwarding (AF) class. This way, it may be possible to serve flows with strict requirements on delay and jitter without adding the complexity of per flow admission control.

The thesis also contributes within the area of application level congestion control. It is pointed out that it is generally a bad idea to combat congestion loss with fixed forward error correction (FEC). FEC is often seen as an attractive alternative to retransmission when there are strict delay and jitter requirements. Unfortunately, the inserted redundancy represents a load increase which may lead to more severe congestion and result in an even larger effective packet loss observed by the application. It is however shown that by assigning a lower priority to FEC packets, this effect could be reduced.

The above findings lead to the studies on shadow probing. When a rate-adaptive source using scalable (layered) video adds an enhancement layer, the increase in sending rate may be significant. Consequently, congestion loss may follow. The thesis proposes to use two shadow probing techniques in order to reduce the impact of such loss. A probing period is introduced before the actual addition of the new layer. First, the shadow layer can be filled with redundancy from the already established layers, and thereby lowering the probability of quality degradation of the flow. The second technique instead aims at protecting the established layers of the competing flows, by assigning a lower priority to the probe layer (e.g. FEC packets).

The probing scheme was integrated with an application-level rate control (ARC) operating between the video application and the transport level congestion control (TFRC was used in this study). The ARC's task is to maximize QoE while obeying to the rate limit obtained from the transport layer. It is emphasized in the thesis that high variations in quality has a negative impact on QoE. For this reason, a strong focus was kept at avoiding being too aggressive when adding enhancement layers, even when the underlying transport layer reported available capacity. Simulations results shows that the proposed shadow probing techniques indeed improve the performance of layered video flows with high delay and jitter requirements.

Overall, the thesis provides insight into the broad research area of congestion control. Several issues are highlighted, and solutions are proposed. Throughout the thesis, the focus is on maximizing the quality of multimedia flows with strict delay and jitter requirements, and doing this while taking into account the stability of the underlying network.

Preface

This thesis is submitted to the Norwegian University of Science and Technology (NTNU) for partial fulfillment of the requirements for the degree of Philosophiae Doctor. The work for this dissertation started in August 2003, and has been supervised by Professor Øivind Kure at NTNU, Department of Telematics.

The work has been conducted at Centre for Quantifiable Quality of Service in Communication Systems (Q2S), Centre of Excellence in Trondheim, at the University Graduate Center (UNIK) at Kjeller and at the Norwegian Defense Research Establishment (FFI) at Kjeller.

Most of the work was financed by Q2S through a PhD scholarship. Q2S was funded by the Norwegian research council, NTNU and Uninett. In addition, FFI has financed one of the publications.

Acknowledgements

I would like to express my gratitude to my supervisor Øivind Kure for sharing his rich knowledge, his excellent abilities to provide to the point feedback, his good motivation skills and his availability through all these years.

I would also like to thank all of my earlier colleagues and staff at Q2S for contributing to a great working environment both professionally and socially. Also, thanks to co-authors Mariann Hauge and Lars Landmark for being excellent discussion partners, and thanks to FFI for financing parts of the work and for providing the needed flexibility in order for me to finalize the thesis and prepare the defense.

Finally, my deepest gratitude goes to you Toril, who has motivated me to finish the thesis despite all my hours at the office. I enormously look forward to future weekends and holidays with you, Oskar and Jesper.

Contents

List of Figures

List of Abbrevations

MPEG	Moving Picture Experts Group	30
NAL	Network Abstraction Layer	15
P2P	peer-to-peer	5
PCN	Pre-Congestion Notification	11
PDF	probability density function	27
PGM	Probe Gap Model	27
PHB	Per-Hop-Behavior	12
PLR	packet loss ratio	44
PPACC	Packet Pair Assisted Congestion Control	15
PSNR	peak signal-to-noise ratio	46
QCIF	Quarter CIF	
QoE	Quality of Experience	5
QoS	Quality of Service	5
RED	random early detection	21
RTP	Real-time Transport Protocol	5
RTT	round-trip time	24
SLA	service-level agreement	7
SNR	signal-to-noise ratio	16
SVC	Scalable Video Coding	15
TCP	Transmission Control Protocol	4
TFRC	TCP-Friendly Rate Control	12
TFRCC	TCP-Friendly Rate Control with compensation	47
TOS	Type Of Service	21
TRC	transport-level rate control	8
UDP	User Datagram Protocol	5
UEP	unequal error protection	51
VBR	variable bit-rate	12
VCL	Video Coding Layer	30
VoIP	Voice over IP	5
WRED	weighted random early detection	21

Part I

Introduction

Chapter 1

Introduction

1.1 Problem statement

A growing amount of multimedia applications in todays computer networks are unable to use Transmission Control Protocol (TCP) congestion control due to strict delay and jitter requirements. Network congestion may occur in any type of capacity limited network, such as the best effort internet, within a bandwidth limited service class in a DiffServ network, or in a network consisting of wireless links. When the congestion level increases, the quality degradation for these multimedia applications ultimately reaches a point where the users are unable to consume the content.

In order to avoid getting into this thrashing state, it is important that the total sending rate of these applications are governed by a control scheme. This can either be elastic applications with a rate adaptation scheme, or network level mechanisms such as flow admission control to govern the number of inelastic flows in the network.

When designing new solutions, all of the following aspects should be taken into account:

- The requirements of the user in terms of perceived quality
- The stability of the underlying networks
- Fair coexistence with competing traffic (e.g. TCP)

Additionally, in order to facilitate standardization, new solutions should to a large extent be independent of the application and of the underlying network technology, allowing their use in a broad range of environments.

1.2 Motivation and scope

Internet congestion control dates back to the mid-eighties, where an increasing bandwidth demand accompanied with a growing level of network heterogeneity lead to the *congestion collapse*. The network capacity was not large enough to handle the traffic, causing bottleneck queues to form and eventually forcing the routers to drop packets. At that time, congestion control functionality was not in use by the end systems, as Transmission Control Protocol (TCP)'s flow control was designed merely to prevent overload of the receiving system. The only reaction to the lost packets was retransmissions, resulting in an even higher load on the network, and a significantly reduced effective throughput.

The need for congestion control in the end systems was soon recognized, and TCP was therefore augmented with Van Jacobson's congestion control mechanisms [1], including *slow start* and *exponential retransmission back-off*. For more than two decades, TCP's congestion control has been one of the key elements making the internet a success, governing a dominant part of the total traffic volume. In this period, even though the bandwidth demand has increased significantly, TCP has been able to prevent a new collapse.

As the internet evolved and gained popularity among home users, multimedia applications gradually became one of the largest sources of traffic in the internet. While multimedia consume was based on pre-downloading of material using File Transfer Protocol (FTP) or peer-to-peer (P2P) file sharing, TCP remained the main transport protocol in use. The next step was the introduction of web streaming services (e.g. youtube). These services are mostly run over TCP, in spite of TCP's sawtooth-like pattern, with sudden drops in sending rate as response to detected congestion. This can be done because one-way streaming is associated with relatively loose delay requirements, and the jitter introduced by TCP can be compensated for using a playout-buffer (also known as jitter buffer) at the receiver side.

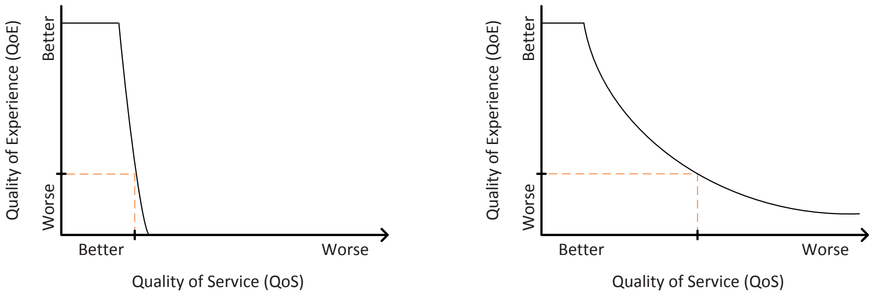
In the resent years however, applications with stronger real-time requirements are getting common, such as IP Television (IPTV), Voice over IP (VoIP), video conferencing and cloud gaming. These applications cannot rely on large playout buffers at the receiver, because of their strict delay and jitter requirements.

As a consequence, applications with such real-time requirements have used Real-time Transport Protocol (RTP)/User Datagram Protocol (UDP) instead, which are protocols without congestion control functionality. This is viable as long as there is excessive capacity in the network. However, relying on over-dimensioning is problematic, as history has shown that the capacity demand has a tendency to increase rapidly with the capacity offered. In addition, with the introduction of smart-phones and tablets, much of the multimedia consume has moved from the wired to the wireless domain. Here, over-dimensioning is even more challenging due the lack of frequency resources.

Consequently, multimedia traffic with real-time requirements will be transmitted over capacity-limited networks in the foreseeable future. These challenges have to a large extent been acknowledged by the research community as well as the industry.

A congestion collapse occurs when sources react to congestion by increasing the sending rate. Real-time multimedia applications rarely use retransmissions, but there are examples of applications that increases the robustness in the coding by adding redundancy. Thus, congestion collapse is a valid concern also for multimedia applications.

The congestion collapse phenomenon can be seen a special case of a more general situation where a network operates in a *thrashing* state, where the amount of *useful work* achieved is small. This situation is also a concern for multimedia flows which does not respond to congestion by increasing the sending rate. If too many flows are active simultaneously, congestion will occur, and the flows start to experience Quality of Service (QoS) disturbances such as increased jitter, delay and packet loss. The user's Quality of Experience (QoE), the subjective perception of the quality, determines whether the media is consumable or not. If it is, the network did useful work when transmitting it. When the QoS disturbance



(a) Inelastic multimedia **(b)** Elastic multimedia (rate-adaptive)

Figure 1.1. The relationship between objective quality (QoS) and subjective quality (QoE).

reaches a certain threshold, the QoE goes from consumable to non-consumable. Thus, by definition, thrashing is a fact when the network operates beyond this point.

The congestion control's task is to ensure that the QoS threshold is not exceeded. However, setting a reasonable threshold is not straight forward, as a mapping between QoS and QoE is needed. Figure 1.1 illustrates the relationship between subjective and objective quality, and shows example mappings between the QoS and QoE thresholds. The shape of this curve is application and user dependent, but subjective testing has revealed that is often exponentially decaying [2, 3], as shown in the figure. The application's degree of *elasticity* determines the shape of the curve. On the extreme end is inelastic multimedia, which cannot tolerate any disturbance beyond the threshold without becoming non-consumable as illustrated in figure 1.1a. On the other extreme are fully elastic applications such as TCP file transfer etc., which adapt the sending rate to the congestion level in the network, having a close to linear QoS and QoE relationship (not shown in the figure). In between there is the category of rate-adaptive multimedia applications (figure 1.1b). These are elastic in the sense that they have the ability to lower the sending rate upon experiencing congestion. Different from the fully elastic applications, elastic multimedia have requirements on bandwidth, delay and jitter. Congestion, and thereby QoS disturbances, can be avoided by reducing the sending rate. However, this cannot be done beneath the minimum bandwidth requirement.

In this thesis, congestion control solutions for both inelastic and elastic multimedia, as defined above, are discussed. They are categorized into the following three areas:

- Network level congestion control (inelastic multimedia)

- Transport level congestion control (elastic multimedia)
- Application level congestion control (elastic multimedia)

Network level congestion control refers to both proactive and reactive control mechanisms within the network. A flow level admission control limits the total number of flows that are admitted into the network proactively. A flow termination mechanism reactively removes flows from the network when congestion is experienced. Together, these are tools for providing congestion control, and avoiding thrashing situations by limiting the number of active flows in the network. Clearly, for inelastic multimedia flows, which are unable to adjust the sending rate, network level congestion control is the only alternative. This is also necessary for elastic multimedia flows due to their minimum bandwidth requirement, as these flows can only reduce their sending rate down to a certain level. Admission control for elastic multimedia is however left out of scope in this thesis.

Admission control may be designed as an end-to-end (transport level) mechanism, or more commonly as a part of a QoS network architecture. This thesis concentrates on the latter. QoS architectures provides the opportunity to treat flows differently based on their importance and QoS requirements. Contracts between the network service provider and the customer specify how the customer's traffic is treated by the network. Such contracts, or service-level agreements (SLAs), include detailed technical descriptions, expressing what the customer should expect in terms of service availability, bandwidth, delay, packet loss etc. In order to provide such service guarantees to the customer, a set of QoS mechanisms is used in the network, where the admission control mechanism is the most important one.

Admission control has been a topic of research for a long time. However, with the growing use of multimedia over dynamic wireless networks come new challenges to be solved. This thesis discusses network level congestion control for inelastic multimedia in mobile ad-hoc networks (MANETs).

Solutions for elastic multimedia applications are instead discussed within the context of transport- and application level congestion control. For such applications, with the possibility to adjust the sending rate on-the-fly, end-to-end mechanisms can be applied in order to adapt the sending rate to the changing network conditions. Here, the sources themselves are responsible for avoiding the thrashing situation, rather than the network. By means of feedback from the peer indicating the current level of congestion, the source adapts its sending rate. The challenge for such algorithms is that they need to take into account the requirements of the multimedia applications as well as preventing congestion. In addition, they need to co-exist with other congestion control mechanisms such as TCP, so care must be taken in order to prevent either one of them grabbing a too large share of the network capacity. Note that rate adaptive sources may also be used within a service class provided by a QoS architecture, particularly when that service class is not governed by per-flow admission control.

It is challenging to design generic mechanisms that meet the requirements of the applications and at the same time being network-friendly. There are so many different applications with different needs, and it is impossible to predict the needs of tomorrow's applications. Often the congestion control is designed for the application at hand, and optimized for a certain underlying network. Such cross layer design may provide near optimal performance in the target environment. However, if the application is replaced, or the underlying network changes, the congestion control will most likely have to be redesigned in order deliver acceptable performance. Also, if each application includes its own congestion control mechanism, fairness is challenged when they compete for the same bandwidth resources. As an example, this effect is seen even with different implementations of TCP, where small differences in behavior may cause a flow to gain significantly more than its fair share of the available bandwidth.

In general, the mechanism consists of two parts: The "lower" part estimates the network conditions and governs the maximum allowed sending rate, while the "upper" part is concerned with how the application best can utilize the available bandwidth in order to maximize the user perceived quality. The problem with cross layer solutions is that if only one of the two premises are changed (i.e. network or application), and a redesign is required, both the upper and the lower part needs to be rewritten. This leads to cost-ineffective application development. To avoid this, a key point of this thesis is to view the congestion control for rate adaptive sources as two more or less independent parts; the *application level congestion control*, and the *transport level congestion control*, realized by an application-level rate control (ARC) and a transport-level rate control (TRC) respectively. The ARC takes care of the upper part and is developed with each application, while the TRC deals with the lower part and is provided as a service from the operating system. A well-defined interface between them ensures improved inter-changeability in a long term perspective.

This thesis discusses the three mentioned approaches to congestion control for unicast multimedia flows with strict delay requirements. The three areas are reflected in the structure of the thesis as described in section 1.3. Admission control for inelastic (non-adaptive) multimedia sources is discussed, with focus on dynamic wireless environments. For elastic (rate-adaptive) multimedia, existing ARC and TRC solutions are evaluated, and new solutions are proposed. In addition key elements in the interface between ARC and TRC protocols are identified.

1.3 Structure of this thesis

This is a compilation thesis, which includes four peer-reviewed accepted publications. Part I puts the collection of publications into a broader context, while part II contains the actual publications, referred to as paper A, B, C and D.

The three areas of congestion control identified in section 1.2 are reflected in the structure of the thesis:

- Network level congestion control
- Transport level congestion control
- Application level congestion control

Figure 1.2 illustrates the structure of the thesis, and how the included papers relate to these areas. In Chapter 2, the reader is provided with a selection of background information related to each of three areas. In Chapter 3, the contributions of the included papers within the three areas are summarized; for each paper, there are three subsections describing the motivation, related work and contribution respectively. See section 1.4.1 for details about the four publications.

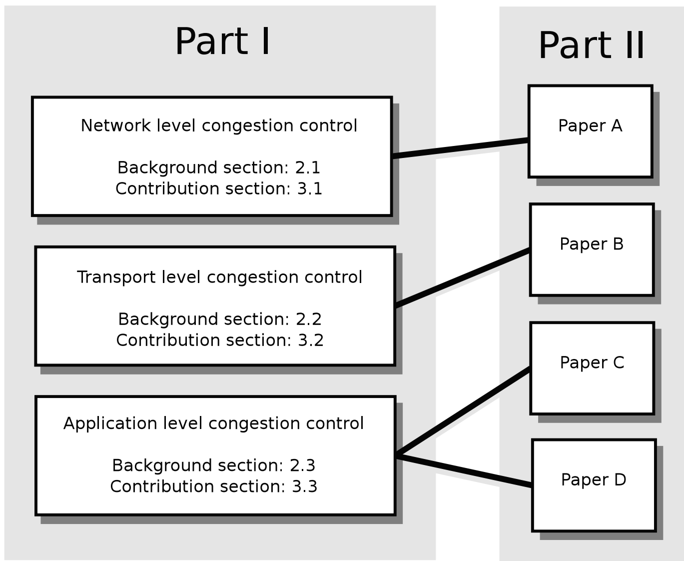


Figure 1.2. Relationship between the included papers in part II, and their respective background and contribution sections in part I

1.4 List of publications

Below is list of the publications included in the part II of the thesis, followed by a list of additional work with contributions from the thesis author which are not discussed in the thesis.

1.4.1 Included in thesis

Paper A	Bjørnar Libæk and Mariann Hauge and Lars Landmark and Øivind Kure"*Admission Control and Flow Termination in Mobile Ad-hoc Networks with Pre-congestion Notification*", Military Communications Conference (MILCOM). Orlando, Florida, October 29 - November 1 2012.
Paper B	Bjørnar Libæk and Øivind Kure, "*Congestion Control for Scalable VBR Video with Packet Pair Assistance*", Proceedings of 17th International Conference on Computer Communications and Networks (ICCCN), St. Thomas, US Virgin Islands, August 3-7 2008.
Paper C	Bjørnar Libæk and Øivind Kure, "*Protecting scalable video flows from congestion loss*", Proceedings of the Fifth International Conference on Networking and Services (ICNS), Valencia, Spain, April 20-25 2009.
Paper D	Bjørnar Libæk and Øivind Kure, "*Generic Application level rate control for scalable video using shadow probing*", Proceedings of the Fourth International Conference on Systems and Networks Communications (ICSNC), Porto, Portugal, September 20-25, 2009

1.4.1.1 About the ordering of the papers

As seen in the list above, the papers are not listed in chronological order. Instead, the list reflects the order used when discussing the three areas of congestion control in chapters 2 and 3. This is strictly based on pedagogical concerns in order to enhance the readability of thesis.

1.4.2 Additional work

Paper E — Odd Inge Hillestad, Bjørnar Libæk, Andrew Perkis, "*Performance Evaluation of Multimedia Services Over IP Networks*", Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on, 2005, 1464-1467

Paper F — Stian Johansen, Anna Kim, Bjørnar Libæk and Andrew Perkis, "On the Tradeoff between Complexity and Performance of error protection schemes for embedded codes over paralell packet erasur channels", In proceedings of the Norwegian Signal Processing Symposium (NORSIG-05), Stavanger, Norway, September 2005

Paper G — Bjørnar Libæk, Anne Nevin, Stian Johansen, Odd Inge Hillestad, Victor Nicola, Yuming Jiang, Peder Emstad, "*Congestion Control for Video and Audio*" a contribution to the EuroNGI WP2.1 state-of-the-art deliverable, June 2006

1.5 Contribution summary

This gives a summary of the contribution of the included papers.

1.5.1 Network level congestion control

Due to the limited bandwidth in wireless networks, QoS mechanisms and especially admission control is essential when transmitting inelastic traffic over a MANET. The recently standardized Pre-Congestion Notification (PCN) architecture provides admission control and flow termination to wired networks. However, there are good reasons why PCN should be considered also in wireless networks; less resources spent on mapping between different QoS regimes on the border between the wired and wireless domains, the availability of a built-in flow termination mechanism, and the independence from routing and link layer protocols.

Paper A shows through a simulation study that The Internet Engineering Task Force (IETF)'s PCN architecture is applicable in MANETs with some important modifications. The study shows that using the unmodified PCN architecture as described in the relevant RFCs, results in a poor performance. After surveying the challenges related to using PCN in MANETs, the paper proposes extensions/modifications to PCN which improve the performance significantly. The most important measure is to use probing on ingress-egress paths which have been idle for a longer period in order to get fresh information of network status and in order to reduce the total amount of signaling. In addition, the paper also describes needed modifications to the PCN metering & marking behavior when operating on a shared channel. The simulation results show that the proposed mechanisms indeed improves the network's ability to avoid ending up in a thrashing situation.

1.5.2 Transport level congestion control

The work done in paper B revealed the weaknesses of TCP-Friendly Rate Control (TFRC) when the application sends with a variable and/or adaptive sending rate. It was shown that TFRC requires relatively large buffers in the routers to avoid high packet loss. In addition, it was shown that TFRC caused significant jitter. The resulted from queuing delay in the send buffer at the source due to the mismatch between TFRC's sending rate and the applications instantaneous sending rate. Consequently, TFRC is not suited for variable bit-rate (VBR) video applications with strict delay requirements.

An alternative TRC algorithm was proposed, based on packet pair estimation of available bandwidth. The proposed packet pair estimation technique is non-intrusive, as packet pairs are formed using the video packets when possible. Different from related work, which use the estimated rate directly to control the sending rate, the proposed TRC only use the estimate as a guideline in order to control the magnitude of increase/decrease of the sending rate. Through simulations, the packet pair estimation technique was shown to perform significantly better than TFRC in terms of packet loss, delay and jitter. This indicates that it is possible to serve traffic with high delay and jitter requirements with the DiffServ Assured Forwarding (AF) Per-Hop-Behavior (PHB) if a proper TRC is used.

1.5.3 Application level congestion control

This thesis' contribution within the area of application level congestion control is made by papers C and D. It is recommended to separate the ARC protocol from the TRC below. Different aspects of an ARC for layered video streaming have been studied:

Due to the strict delay requirements, retransmission in order to protect against packet loss is not an option. A frequently used alternative is to use forward error correction (FEC). Paper C points at the challenges of using fixed FEC schemes to protect against congestion loss, as often being proposed in the literature. Fixed FEC means that a fixed level of redundancy is added to the flow for its complete lifetime. The added load caused by the redundancy contributes to congestion, and the paper shows both analytically and with simulations, that the addition of FEC is likely to cause increased packet loss, as observed by the application, in high load situations. For this reason, Paper C proposes to use FEC only in certain well-chosen periods, as opposed to use FEC continuously. This reduces the total amount of redundant packets in the network, and contributes to a lower load in the system.

From a QoE view, it is favorable with a constant lower video quality rather than an alternating quality with a higher mean value. The changes in quality, and especially quality reductions, disturb the viewer and take the attention away from the content. For this reason, when streaming layered video, enhancement layers should not be added and dropped too often. Paper D suggests that an ARC should take this into account when deciding whether to add new enhancement layers, and introduces the Layer Decrease Frequency (LDF) metric, which is used when evaluating ARC algorithms. The LDF is defined as the number of enhancement layer reductions per second.

Based on the two arguments stated above, an ARC algorithm is proposed in paper D, which both aims at keeping a low LDF and also avoid fixed FEC by using the proposed shadow probing techniques in order to minimize experienced packet loss. The LDF is controlled by a parameter that enforces a minimum time period between enhancement layer increments. The proposed ARC is designed to operate above a TRC which provides the allowed sending rate, and TFRC is used as an example. The idea behind shadow probing is to protect the existing/established enhancement layers during bandwidth probing experiments by either adding FEC to the established layers, or to set a lower priority on the probing layer if a priority mechanism is available in the network. Simulations results show that the proposed shadow probing techniques may significantly improve the perceived quality of multimedia flows with high delay and jitter requirements.

1.6 Research methodology

Aside from literature studies and mathematical analysis, network simulation has had a major role when obtaining the research results in the included papers. For this reason, the following sections describe how simulation was used, and provides added detail about the simulation experiments which were forced omitted in the papers due to length restrictions.

1.6.1 Discrete event simulation (DES)

In all the papers included in this thesis, Discrete Event Simulation (DES) is the primary tool used for assessing the performance of the various protocols and mechanisms. A DES-based network simulator provides cost effective means for testing and evaluating protocols and mechanisms as opposed to setting up laboratory experiments. In simulation experiments, full control of the environment is achieved, while in real-world experiments, there are often unknown and uncontrollable sources of error. An important advantage of simulation contra experimentation is the possibility to reproduce exact copies of a simulation experiment. This facilitates effective debugging and troubleshooting.

However, a number of assumptions must usually be made when creating simulation models, and it is important to have these assumptions in mind when interpreting the results and drawing conclusions. The output result is only valid for the given input parameters and only as long as the given assumptions hold. Wrong conclusions may be drawn if assumptions are ignored. In this sense, more realistic results may be obtained using laboratory experiments.

1.6.2 Collecting statistics

Most of the simulation experiments presented in the included papers are based on running multiple simulation runs for each data point in order to produce confidence intervals. Each run is done with different seeds in the random generators. As an example, if the objective is to estimate the mean of simulation output variable X with unknown distribution, each run then produces one sample X_i. The mean $\overline{X}$ is estimated as

$$\overline{X} = \sum_{i=1}^{m} \frac{X_i}{m}$$

where m is the number of simulation runs. The sample variance of X can be estimated with the unbiased estimator

$$S^2 = \sum_{i=1}^{m} \frac{(X_i - \overline{X})^2}{m-1}$$

From this, the variance of the sample mean estimator, which is used to produce the confidence interval, can be estimated:

$$S^2_{\overline{X}} = \frac{S^2}{m}$$

Finally, relying on the central limit theorem which states that the sample mean approaches normality when m is large enough, the confidence interval are found by assuming normal$(\overline{X}, S^2_{\overline{X}})$ distribution. Note that this method may be used independently of the underlying distribution of X.

1.6.3 Network simulation

In a DES based network simulator, the model typically consists of network nodes with internal components such as protocol entities and traffic generators, and links between them with limited capacity. Some nodes are modeled as routers while others are modeled as end hosts. The source nodes typically contain traffic generators, injecting packets into the network. Routers are most often modeled with finite buffers, both dropping and delaying packets. Statistics may in principle be collected anywhere in the model.

Several commercial and non-commercial network simulators are available. Examples include NS2, NS3, Opnet, Omnet++, QualNet, J-Sim and GloMoSim.

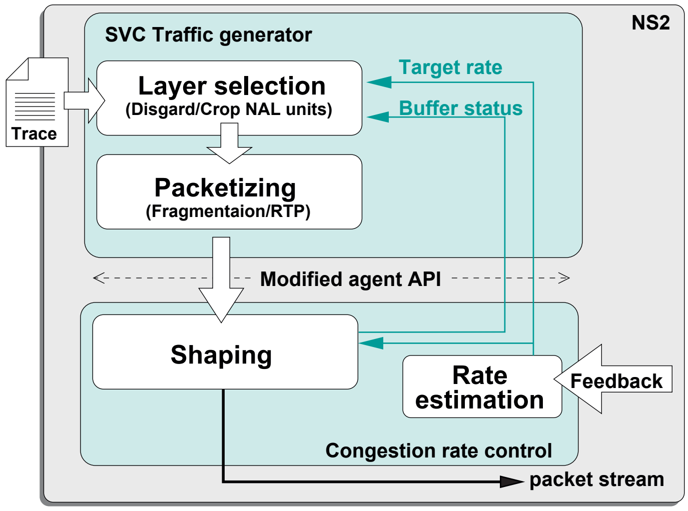


Figure 1.3. SVC Traffic generator in NS2

In the simulation experiments presented in paper A, Omnet++ was used, while NS2 was used in papers B,C and D. Omnet++ was chosen in paper A due to the detailed model of the physical and link layer of 802.11 networks, and a good collection of MANET routing protocols. The primary reason for choosing NS2 in the other three papers was the availability of the TFRC module, developed by the authors of TFRC themselves. Refer to section 2.2.1 for a description of TFRC. As described below, some modifications to the TFRC module had to be done.

1.6.4 The SVC traffic generator in NS2

In order to simulate realistic multimedia traffic in papers B,C and D, a trace based traffic generator was written for NS2. The generator read its input from a trace file, which was created from a Scalable Video Coding (SVC) encoded test video clip.

The generator can be configured to either use TFRC or Packet Pair Assisted Congestion Control (PPACC) as transport protocol. Figure 1.3 gives an overview of the SVC source node. The generator reads Network Abstraction Layer (NAL) units from the trace file. The trace file contained one line for each NAL unit, specifying the size of the packet and which enhancement layer the packet belonged to. At the beginning of each Group Of Pictures (GOP), the layer selector chooses when to add or drop layers in each of the three dimensions based on knowledge of sending rates of near future GOPs and the target rate from the transport entity. A pre-configured trajectory is followed when adding or dropping enhancement layers. Each step in this trajectory is a specific combination of the three dimensions,

Step	spatial layer	temporal layer	FGS layer
1	0	1	0
2	0	1	0
3	0	2	0
4	1	2	0
5	1	3	0
6	1	3	1

Table 1.1. Trajectory for layer selection

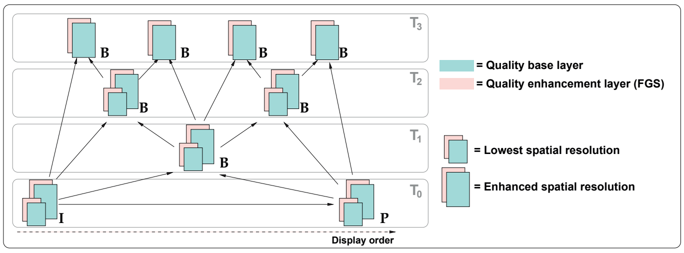


Figure 1.4. SVC Enhancement layer configuration. Two spatial, four temporal and two quality enhancement layers. Arrows denote dependencies. The spatial base layer has QCIF resolution (176x144) while the spatial enhancement layer has CIF (352x288). The lowest temporal resolution is 3 fps, which is doubled in each of the temporal enhancement layers, thus the highest temporal resolution is 24fps. The quality enhancement layer is FGS, which means that a packet in this layer can be truncated at an arbitrary position (in the current version of the standard, FGS has been replaced with Medium Grained Scalability (MGS) where packet truncation is removed, thus reducing the number of available rate points compared to FGS).

such that the rate of step i is larger than the rate of step i-1. The trajectory used is given in table 1.1. Note that the cropping functionality for the Fine Granular Scalability (FGS) layer is not used. Further, the filtered NAL units are packetized before being passed to the transport module. This involves fragmentation if the NAL unit is larger than the MTU.

The test clip [4] was encoded with the JSVM [5] reference software for SVC. The original video was in Common Intermediate Format (CIF) resolution (352x288) and had a frame rate of 24 frames per second. The encoded video had 8 frames in each GOP and two spatial, four temporal and 2 quality (signal-to-noise ratio (SNR)) enhancement layers. See figure 1.4 for details of the GOP structure. The resulting sending rates of the different layer combinations are shown in figure 1.5. It shows that there is a significant variation from GOP to GOP within each layer, as well as large differences in the sending rates between the layers.

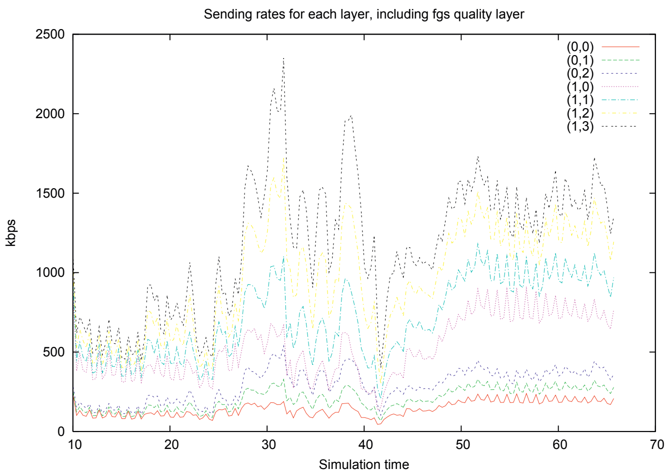


Figure 1.5. Sending rates for each combination of spatial and temporal layer in the test video clip, when always including the FGS enhancement layer. The rate is the average within a GOP. Explanation of the legend: (x,y) means spatial layer id x and temporal layer id y.

1.6.5 Trace-driven simulation

As described in the previous section, some of the simulation experiments presented in the included papers were based on trace-driven simulation. A video trace obtained from a real video test clip was fed to the simulator as input data.

The advantage of trace-driven network simulation is that the traffic pattern produced by the traffic generators is authentic. Stochastic traffic generators may approximate the lower moments of a traffic distribution, but often fail when it comes to higher order moments which may indeed be important for the output results. When using trace-driven traffic generators, one can safely assume that such effects are not present.

However, a disadvantage with trace-driven simulation is the lack of flexibility with respect to tuning the arrival process. A stochastic generator is easily tuned by changing the parameters of the underlying random distribution. Producing a new trace usually requires significantly more effort. Another disadvantage associated with trace-driven VBR video in particular, is that the traffic distribution is strongly correlated with the video content, and several trace files based on different video content should be evaluated before drawing conclusions.

As described in section 1.6.2, several simulation runs with different seeds in the pseudo-random generators is used with the purpose of creating confidence intervals. When using stochastic traffic generators, this ensures that the result does not depend on a specific limited part of the random number stream. Likewise, for trace-driven simulation, *trace shifting* can be used in order to achieve the same effect. When starting the simulation, a random starting point within the trace file is drawn using a random number generator. When the end of the trace file is reached, the generator continues from the beginning of the file. This was technique was used by the SVC traffic generator presented in the previous section.

Chapter 2

Background

2.1 Network level congestion control

An increasing amount of multimedia streaming applications can adapt the sending rate to the network conditions. Still however, a major part does not have this ability. Instead, when it is important to serve such inelastic flows, an alternative is to use QoS mechanisms within the network in order to provide QoS guarantees and to avoid congestion and thrashing situations. Different approaches can be taken; mechanisms such as traffic engineering and QoS routing can be used to lead the multimedia traffic onto paths with available resources. Congestion pricing can be used as an incentive for lowering the sending rate, or higher level multimedia gateways can be inserted into the network doing caching, transcoding, intelligent dropping etc.

Alternatively, a QoS architecture can be used, where flows request admission to the network, and resources are reserved in order to provide service guarantees. The two main QoS architectures specified for the internet is IntServ and DiffServ. In IntServ, resources are reserved for each individual flow in all nodes along the path between the source and destination. This way, hard guarantees can be made on the flow level, but the architecture does not scale well with the number of flows. In DiffServ on the other hand, the flows are aggregated into traffic aggregates in the edge of the network, and resource reservations in the core of the network is only done at aggregate level. This coarse grained reservation scheme improves the scaling properties of the DiffServ architecture compared to IntServ. On the downside, when a flow is admitted by the admission control at the ingress, only statistical guarantees are provided, meaning that the flow's QoS requirements will be met only by a certain probability specified in the service-level agreement (SLA) between the customer and the network provider.

DiffServ is by far the most common QoS architecture in the internet today. As the research in all the included papers leans on the DiffServ architecture, it will be described in more details in the following section.

2.1.1 DiffServ

The DiffServ architecture has been standardized by the IETF as a collection of RFCs. The main philosophy behind DiffServ is to move the bulk functionality to the edges of the network, while the network core is kept as fast and simple as possible. This gives a scalable architecture with a core being able to serve a large number of flows at high speeds and at the same time being able to differentiate according to the applications' QoS requirements.

In order to do this, DiffServ is based on traffic aggregation. Individual traffic flows are aggregated at the edge of the network into traffic classes, based on traffic type. Ingress routers do classification on incoming packets by either looking at the DiffServ Codepoint (DSCP) field in the IP header, or by doing packet inspection. After determining the traffic class, the packet is possibly remarked with a new DSCP value, in order to be mapped onto a traffic aggregate. Before forwarding the packet, the ingress may also apply other mechanisms such as policing or shaping. The core routers differentiate the traffic based on the DSCP value, which is mapped to a certain Per-Hop-Behavior (PHB). The PHBs define packet forwarding properties, and are designed to reflect the QoS requirements of different types of applications.

Through a number of RFCs, the IETF has defined a recommended set of PHB that is widely adopted:

- Default PHB, for best-effort traffic.
- Expedited Forwarding PHB, for real-time delay- and jitter sensitive traffic.
- Assured Forwarding PHB, for providing probabilistic delivery guarantees to subscribing users.
- Class selector PHB, for backward compatibility with the previous definition of the IPv4 Type Of Service (TOS) field.

These PHBs have been assigned dedicated DSCP values. In addition, it is possible for a network provider to define its own classes using available unassigned DSCP values.

To realize a PHB, core routers use a combination of packet scheduling and buffer management. Assuming that each class has its own queue, a scheduler selects the next queue to transmit when the outgoing link becomes idle. The scheduler may operate according to different schemes such as strict priority, round robin or weighted round robin, ensuring that the correct amount of transmission resources are allocated to each service class. Buffer management on the other hand, or active queue management (AQM), is used in order to differentiate within a single class. The DSCP may also contain information about the relative priority (drop precedence) of each packet, and the AQM may then have different dropping/marking probabilities for each priority level. An example is weighted random early detection (WRED) which is random early detection (RED) [6] with one buffer threshold for each priority.

As seen in the list above, multimedia streaming with strict delay requirements is intended to be sent using the Expedited Forwarding (EF) PHB. The EF PHB provides a "low loss, low latency, low jitter, assured bandwidth, end-to-end service through DiffServ domains" [7]. The EF PHB is implemented by having a separate queue for EF traffic, and a priority scheduler ensuring a guaranteed bandwidth for outgoing EF traffic from the node. The arrival rate of EF traffic to the node should not be larger than the guaranteed output rate. Otherwise, the (typically) short queue may start to drop packets if there is traffic from other classes being served by the outgoing link. To avoid this, admission control is needed at the ingress nodes, ensuring that the EF rate is below the EF capacity limit.

Several proposals for admission control have been made for DiffServ. One of them is the Bandwidth Broker (BB) scheme [8]. Here, a centralized entity in the network acts as a broker for all the links in the domain. When an ingress node receives an admission request from the outside, it queries the BB which has full knowledge of all admitted flows in the domain. Apart of the single-point-of-failure issue, the BB scheme scales poorly due to the vast amounts of signaling to and from the central node.

For these reasons, distributed admission control for DiffServ domains has been a popular topic in the research community. Instead of communicating with a central node, the ingress and egress nodes collaborate to assess the status of the path between them. The IETF has recently published a number of RFCs and Internet drafts describing an architecture for distributed admission control and flow termination in DiffServ domains, named Pre-Congestion Notification (PCN). The following section gives an overview of the PCN architecture.

2.1.2 Pre-Congestion Notification (PCN)

The intention of the PCN architecture [9] is to provide admission control and flow termination for a PCN traffic class in a DiffServ domain. Routers recognize the PCN traffic class by tagging the PCN packets with a specific DSCP value, and is typically treated with the EF PHB as described in the previous section. The PCN architecture is founded on a specific coding of the two last bits of the 8-bit TOS (IPv4) / Traffic class (IPv6) field of the IP header, also known as the Explicit Congestion Notification (ECN) bits. The basic idea is that each router in the domain monitors the traffic on all of its outgoing links. Each link is configured with two rates; the admissible rate and the supportable rate. When the rate of PCN traffic on a link exceeds these rates, the two PCN-bits in PCN packets are marked appropriately. By monitoring the marked packet stream, the egress router is able to assess the level of congestion on a specific ingress-egress-aggregate (IEA), and feeds this information back to the respective ingress as illustrated in figure 2.1. Based on this feedback, the ingress makes decisions about whether new flows can be admitted to that IEA, and whether existing flows should be terminated.

When using the three-state encoding which is published as a proposed IETF standard in [10], the routers are able to mark the packet as either not-marked(NM), threshold-marked (ThM), or excess-traffic-marked (ETM). How this is actually done depends on the marking scheme. Such a marking scheme is also proposed as an IETF standard in [11]. Here, two marking behaviors are described; threshold marking and excess marking. A threshold marker marks all packets if the PCN traffic rate exceeds the marker's configured rate. The excess marker on the other hand, marks only the proportion of packets which actually exceeds the configured rate.

The behavior of the edge nodes (ingress and egress), including the signaling between them, is not intended to be standardized. However, two alternative edge behavior descriptions have been published as experimental RFCs. [12] describes Controlled Load (CL) edge behavior where it is assumed that three-state marking

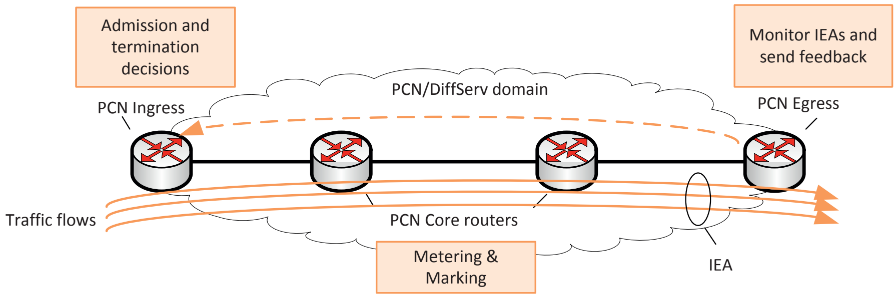


Figure 2.1. A PCN domain

is available in the network, while [13] describes a corresponding scheme when only two-state marking is available. In addition, numerous variants of edge behavior is surveyed in [14]. Refer to paper A for a detailed description of the CL edge behavior.

2.1.3 Relationship between PCN and Explicit Congestion Notification (ECN)

As described in the above section, PCN redefines the use of the ECN-bits in the IP header. ECN [15] was primarily designed in order to improve the performance of TCP flows by letting the routers mark packets when observing imminent congestion, as an alternative to dropping packets. The TCP source reacts equally to an observed marked packet as an observed dropped packet, while avoiding costly retransmissions.

Thus, ECN is an end-to-end mechanism designed for elastic flows which are able to adjust its own sending rate, while PCN is designed for inelastic flows needing network support to avoid congestion and thrashing situations.

It is important to note that IETF has made large efforts into making the PCN coding backwards compatible with the ECN coding.

2.2 Transport level congestion control

This section discusses different types of congestion control solutions, or transport-level rate controls (TRCs), at the transport level. By definition, according to the layering philosophy of the internet, these solutions are application independent. A TRC is mainly responsible for two tasks: 1) estimate the available bandwidth on the path to the destination, 2) police the outgoing packet stream in order to ensure that the sending rate is not above the estimated available bandwidth.

As described in the introduction, TCP has been a major contributor with respect to avoiding congestion in today's packet networks. Aside from preventing congestion, TCP was primarily designed for applications needing a piece of data transferred reliably between two systems as fast as possible. For jitter-sensitive multimedia applications however, which typically produces data at regular intervals, TCP causes problems for two reasons:

- Reliability is ensured using retransmissions, detected by the source when observing missing acknowledgements from the receiver. Retransmitted packets obviously have significantly larger delays than other packets and therefore contribute to jitter.

- A congestion window is used in order to govern the instantaneous sending rate. The window is increased and decreased according to an additive increase multiplicative decrease (AIMD) scheme. In the bandwidth probing phase (congestion avoidance phase), the rate is linearly increased with roughly one full size packet every round-trip time (RTT). When observing congestion, indicated by a missing acknowledgement, the window size is set to half. This behavior produces the well known sawtooth pattern of the TCP sending rate. When the window size is small, the outgoing sending rate may be significantly smaller than the application's data generation rate. Consequently, data must be queued at the sender side, resulting in varying queuing delays corresponding to the variations in the TCP sending rate. This also results in jitter. Even if the application is able to adapt its sending rate, the sudden decrease in TCP's sending rate may cause a major difference between the application's and TCP's sending rate over an interval long enough to build up a queue.

If the application tolerates delayed playback, a jitter buffer is normally used in order to compensate for the delay variations. There are many examples of video streaming applications using TCP. However, interactive multimedia applications such as video conferencing with strict delay guarantees cannot rely on such buffering without degrading the perceived quality. Other examples are IP TV applications, where the channel switching delay should be kept as small as possible, and cloud gaming, where delays must be kept small in order to ensure a gaming experience comparable to regular desktop gaming. These applications need a transport protocol offering significantly less jitter than TCP, without being a major contributor to end-to-end delay.

As a supplement to TCP, The Internet Engineering Task Force (IETF) has proposed TFRC as a transport level congestion control mechanism for unicast streaming multimedia flows requiring a smooth sending rate. TFRC is defined as a profile in the Datagram Congestion Control Protocol (DCCP) [16] framework, as Congestion Control ID 3 [17]. However, when sending layered variable bit-rate (VBR) video, TFRC has some weaknesses. Therefore, paper B proposes an

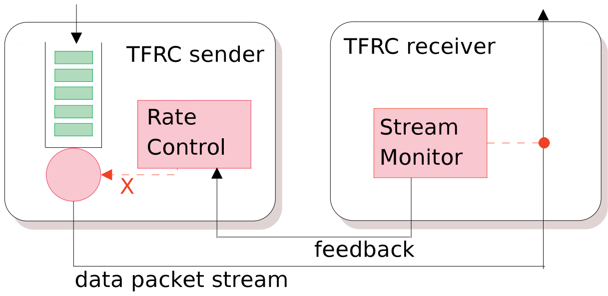


Figure 2.2. TFRC protocol overview

alternative TRC algorithm based on packet pair probing in order to estimate the available bandwidth.

In the following, section 2.2.1 describes TFRC and explains the issues related to layered video. Further, the packet pair probing technique is described in 2.2.2.

2.2.1 TCP-Friendly Rate Control (TFRC)

The IETF has adopted the TFRC protocol as one of the congestion control profiles in Datagram Congestion Control Protocol (DCCP). TFRC is a congestion control mechanism designed for multimedia applications that cannot tolerate TCP's sudden drop in the sending rate. Historically, such flows have been forced to use UDP as transport protocol. Unfortunately, since UDP does not include congestion control, TCP flows are likely to be starved when competing with bandwidth intensive UDP flows. TFRC is intended as an alternative to TCP and UDP, being reasonably fair to competing TCP flows while at the same time having a relatively stable sending rate. The protocol is described in detail in [18], but a brief overview is given here[1]

TFRC is built around the TCP-throughput equation [19], which calculates the approximate average long-term throughput of a TCP flow given an estimate of the loss event rate p and the RTT R:

$$T = \frac{s}{R\sqrt{\frac{2p}{3}} + t_{RTO}(3\sqrt{\frac{3p}{8}})p(1+32p^2)} \tag{2.1}$$

The sender maintains an estimate of the allowed sending rate X, which is used to control the output sending rate. X is updated approximately once each RTT, when a feedback control packet arrives from the receiver. The feedback packet contains the loss event rate p, time-stamp to estimate the RTT and the receiver's estimated incoming rate R_{rcv}. Upon receipt of a feedback packet, the

[1] Note that description of the TFRC algorithm is somewhat simplified here, for the purpose of readability. Refer to the standard specification for a correct description of the behavior

sender is able to calculate a new value for T using equation 2.1 and the newly arrived information. During the normal operation of TFRC, in the congestion avoidance phase, X is usually set equal to $max(T, 2 * R_{rcv})$.

Initially, TFRC operates in a slow start phase where the throughput equation is not involved in calculation of X. Instead, a doubling of the allowed sending rate X is performed once each RTT, as long the rate does not exceed $2 * R_{rcv}$. This gives a behavior similar to TCP's slow start. Different from TCP, TFRC will only use slow start initially, while TCP re-enters slow start after a retransmission-timeout.

TFRC has mainly been designed to serve constant bit-rate (CBR) type of applications, with fixed packet size and small variations in bandwidth demand on the round-trip time (RTT) time scale. Also, it is to a large extent assumed that the application always has data available. When these assumptions hold, TFRC is able to operate in a relatively stable manner. However, the stability is challenged when the video application is rate-adaptive and/or uses VBR coding, as pointed out in paper B. In these cases, there will often be periods where the application sending rate is lower than TFRC's allowed sending rate X. After such a period, when the application suddenly needs to increase the sending rate, TFRC has already set $X = 2 * R_{rcv}$. It is not unusual for a VBR encoded video that the low and peak sending differs with considerably more than a factor of 2, not to speak of the case where a new video enhancement layer is added. This causes two problems:

- Since the TFRC packet scheduler then operates at rate X lower than the application sending rate, growth in TFRC's send buffer is unavoidable. This may lead to large jitter which is harmful to the video quality.
- The sudden increase in the actual sending rate may cause congestion, and ultimately forcing other flows to drop layers. The consequence is an unstable behavior.

To meet the challenges related to TFRC and VBR video, there are two alternatives. Either, an alternative control method may be used, or a concealment layer between the application and TFRC may hide the misbehavior. An attempt of the former is described in paper B, using packet pair estimation as explained in section 2.2.2. The latter is discussed in papers C and D, where application level rate control is studied.

2.2.2 Packet pair estimation

2.2.2.1 Bottleneck capacity

In [1], Van Jacobson suggested that the bottleneck capacity of a network path could be estimated by measuring the time spacing between two packets arriving at the receiver[2] if they where sent back-to-back by the sender. Figure 2.3 illustrates

[2] Jacobson actually measured the spacing between the acknowledgement packets arriving at the sender side, assuming that it was the same as at the receiver side.

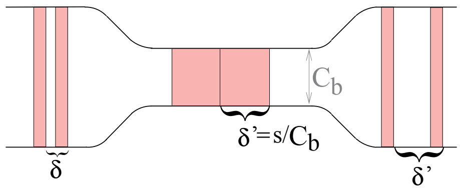


Figure 2.3. The probe gap model

the idea behind the technique, where the path is represented by a funnel where the vertical dimension represents the bandwidth and the horizontal dimension represents time. The colored areas show the position of the packets along the path. At the left, two packets are sent with a small input spacing $\delta < s/C_b$ on a high capacity link, where s is the size of the first (rightmost) packet and C_b is the bottleneck link capacity. It is assumed that this spacing is maintained when the two packets arrive at the bottleneck link in the middle, where the packets are spread out in time. The resulting gap between the packets is then $\delta' = s/C_b$. By using the measured output gap δ' and the known packet size, the receiver is now able to estimate the bottleneck capacity as $\hat{C}_b = s/\delta'$. This model is often referred to as the *Probe Gap Model (PGM)*.

Unfortunately, the estimator is unbiased only if there is no cross traffic interfering with the two probe packets anywhere on the path. Otherwise, cross traffic packets may either increase or decrease the gap depending on whether they interfere before or after the bottleneck link respectively. This was later addressed by Paxson [20], who proposed to identify modes in the probability density function (PDF) of the output gap distribution and stated that the largest mode is with a high probability the output gap from the bottleneck link.

2.2.2.2 Available bandwidth

The probe gap model can also be extended in order to estimate the available bandwidth on the bottleneck link. This involves estimating the bandwidth usage of the cross traffic. As the estimation of available bandwidth using packet pairs is a central topic in paper B, the detailed description is given in the related work section of that paper (section 3.2.2).

2.3 Application level congestion control

Historically, congestion control has been a transport level mechanism, reflected by its location in the OSI model. The primary objective has been to avoid congestion in the network, and to ensure fairness among flows. Belonging to the transport level, congestion control functionality has traditionally been implemented in the operating system, with a relatively simple interface to the applications. For multimedia applications however, a closer coupling between the application and the congestion controller is necessary, and the congestion control mechanism has therefore moved up the protocol stack. In this process of optimizing the application quality, properties such as fairness and network stability have a tendency to receive less attention.

When individual applications implement their own congestion control, they may work well in isolated environments where all flows are of the same type. However, if flows with different congestion control algorithms compete for the same bandwidth resources, they may interfere with each other's regulation schemes. This may cause instability with high oscillations in the sending rates, and it may cause unfairness with some flows claiming more than their fair share of bandwidth.

To avoid these issues, different applications can use the same transport-level rate control (TRC), thereby ensuring that the underlying control algorithm is the same for all flows. Over the open internet, TFRC is an example of a protocol intended for such use, carefully designed in order to operate together with TCP. However, as pointed out in section 2.2, TFRC is not well suited for layered video, as the sudden increase in application sending rate causes jitter, packet loss and oscillating behavior.

In general, as long as the TRC is not targeting a specific application, there is also need for an application-level rate control (ARC). An ARC has a closer tie to the application, and ensures that rate adaption is done on the premises of the application, and not only the network. The goal of the ARC is to optimize the QoE, which does not necessarily translate into optimizing QoS. For instance, a high average bandwidth is not always beneficial from a QoE perspective, if the cost is high variations in quality.

The tie to the application may vary in strength. The ARC can be an integrated part of the application, it can be designed as a standalone protocol serving one specific application, or it may be a standalone protocol designed for a range of applications. En example of the latter is described in paper D, where the proposed ARC is designed for applications using layered VBR video (i.e. it is codec-independent).

Further, the ARC may be designed to work together with a TRC, or alternatively, all the necessary "transport level mechanisms" can be incorporated into the ARC. However, as argued above, it is recommended to have a common TRC for a range of applications. Consequently, a split between the TRC and ARC is reasonable. Figure 2.4 illustrates the protocol stack where the application, ARC and TRC are shown as three separate protocols. This organization is the basis for the research presented in papers C and D. Clean interfaces between the protocols facilitates modularity (e.g ability to replace the TRC without redesigning/rewriting the application). Note that integrating the ARC with the application is possible while still gaining from having a separate TRC.

There are two key elements needed in order to achieve application level congestion control:

- First of all, the application needs to be elastic, being able to adjust the video quality. This must be done on-the-fly, making it possible to adapt the application data rate according to the varying network conditions. For inelastic applications, network level congestion control as described in section 2.1 is the alternative.

- Second, a decision-engine implementing a control method which takes decisions about how, how much and when the video quality should be adjusted. These decisions are typically based on feedback from the peer and/or the underlying TRC, and the intention is to optimize performance with respect to some video quality metric. When using a non-reliable transport protocol, the ARC may also apply error handling techniques such as retransmission or forward error correction (FEC).

Contributions of this thesis within the area of application level congestion control are made with respect to the last item, and will be discussed in 3.3. The described solutions rely on a video codec being able to adapt the sending rate. More specifically, layered video is assumed. For this reason, rate adaptive video in general, and SVC in specific, are briefly described in the following sections.

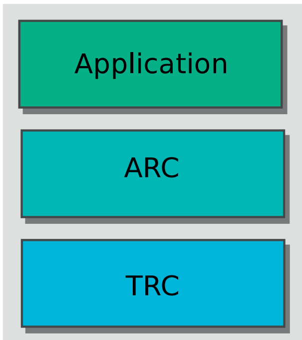


Figure 2.4. ARC is located between the TRC and the application

2.3.1 Rate adaptive video

There are three main ways of adapting the sending rate from a video source. The first is to have a rate adaptive video encoder, which adjusts its compression parameters to satisfy the provided target output rate. Another possibility is to encode the video in several versions, each with a different bitrate (and quality). The video source may then switch between these streams (i.e. bitstream switching) when experiencing changes in the network conditions. The third way is to use a scalable (layered) video codec, meaning that the stream output from the encoder can be divided into substreams with different quality and bitrate. A base layer may then be complemented with one or more enhancement layers, each contributing to an increase in the video quality. The latter is the approach taken in the Scalable Video Coding (SVC) amendment to the H.264/MPEG-4 Advanced Video Coding (AVC) standard, which is likely to be the most used scalable video codec in the foreseeable future. As this is the codec studied in papers B, C and D, a brief overview follows.

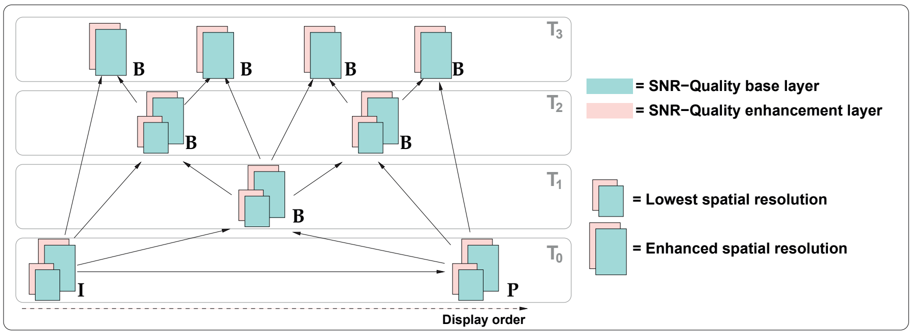


Figure 2.5. Example SVC GOP structure. Each square represents a NAL-Unit. T_0 is the temporal base layer, while each temporal enhancement layer (T_1-T_3) doubles the frame rate. At each temporal level, there is the possibility to add a SNR-quality enhancement layer and/or a spatial enhancement layer.

2.3.2 Scalable Video Coding

The H.264/AVC video coding standard is developed and maintained by Joint Video Team (JVT), a joint effort between ITU-T and ISO/IEC. The first version of the standard was finalized in 2003, providing a video codec with improved coding efficiency compared to its predecessors. This results in better utilization of both network and storage resources, as well as better error resilience in lossy environments. The codec is conceptually divided into two layers; the Video Coding Layer (VCL) and the Network Abstraction Layer (NAL). The VCL contains the compression functionality, while the NAL does the packetizing/mapping of the VCL output onto the underlying transport/storage technology (e.g. RTP). The output from the NAL is NAL-Units, containing a NAL-header describing the VCL-content.

In November 2007, version 8 of the standard was published, containing the SVC amendment. This included the possibility to add enhancement layers to an H.264/AVC compatible base layer, each providing a quality increment in the spatial, temporal or the SNR domain. As in the earlier Moving Picture Experts Group (MPEG) standards, a video frame can be either *intra coded* (I), *predicted* (P) or *bidirectionally predicted* (B). Figure 2.5 illustrates the concepts of scalability in the three dimensions, using an example GOP configuration. The motivation behind the SVC amendment was two-fold: flexibility in terms of heterogeneous receivers, and the ability to adapt the sending rate on-the-fly by simply discarding NAL-Units belonging to specific enhancement layers. Each NAL-Unit was also extended with an SVC header, containing information about the layer association of the contained VCL data.

Chapter 3

Contribution

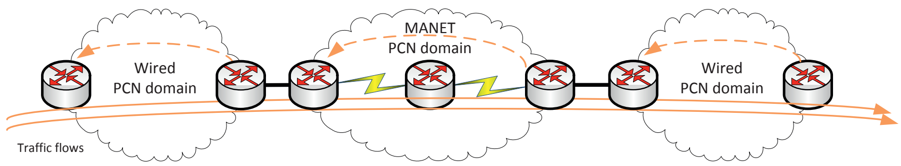


Figure 3.1. A MANET connected to PCN/DiffServ domains.

3.1 Network level congestion control

3.1.1 Motivation for paper A

As discussed in section 2.1, a QoS architecture with an admission control mechanism can be used in order to avoid congestion and thrashing situations in networks with inelastic or elastic multimedia traffic. In wired networks, over-dimensioning is often considered as an alternative in order to minimize the probability of congestion. In today's networks however, there is an increasing use of wireless technologies. Here, over-dimensioning is not an option due to the lack of frequency resources and limitations on channel capacity. In addition, in wireless environments such as in mobile ad-hoc networks (MANETs), there are often high dynamics caused by mobility and interference. Consequently, the channel capacity is varying with time, and the probability of congestion and the need for admission control is substantial. The dynamic environment with variations in channel conditions and likelihood of re-routing events also increases the need for termination of already admitted flows.

As discussed in the related work section (3.1.2), several admission control solutions for MANETs have been proposed in the literature, targeting the range of MANET-specific challenges not found in wired networks. There are however a number of reasons why Pre-Congestion Notification (PCN) (described in section 2.1.2) should be considered as an alternative to those solutions. Recall that PCN, and also DiffServ, are primarily designed for wired networks.

First of all, PCN is likely to be used in the wired networks to which the MANET is directly connected, as illustrated in figure 3.1. When using the same standardized QoS framework in all parts of the network, both capital and operational costs may be reduced if assuming that both the wired and the wireless networks belong to the same service provider. Less effort is needed in order to map between two different QoS technologies, and less resources are spent on education of the operators.

A second advantage of using PCN is that it comes with a built-in flow termination mechanism, in contrast to a large part of its alternatives. In MANETs, where significant reduction in capacity is likely due to shadowing effects, mobility, re-routing and background noise, the conditions during which an admission decision

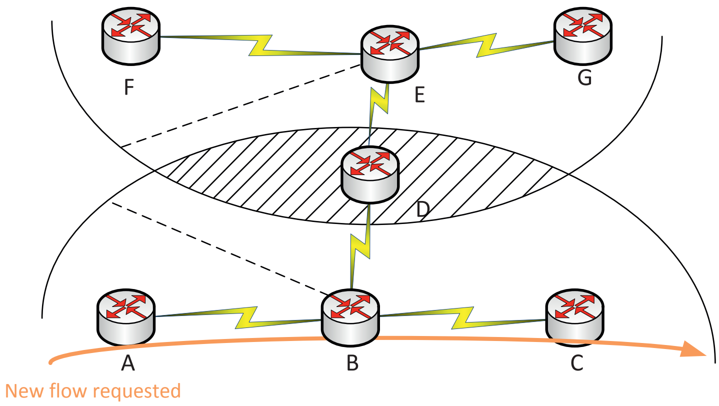


Figure 3.2. The hidden path issue. The curves mark the sensing range of node B and E respectively. Node D is in a critical area because it is impacted by both traffic admitted on the path from A to C and traffic on the path between F and G.

was taken may change drastically during the admitted flow's lifetime. The ability to intelligently select admitted flows for termination is therefore important.

Another property of PCN motivating its use in MANETs is the independence from link layer and routing protocols. This is important in a standardization context.

There are numerous challenges with distributed admission control in MANETs. The most critical ones are listed below:

- *Channel variations.* There are multiple reasons why the capacity on a link between nodes int a MANET is time varying. The mobility causes changes in the path-loss as well as changes in the shadowing effects caused by obstacles. In addition, there may be changes in external noise around the receiver. In turn, these channel variations may cause link breaks and re-routing of admitted flows onto new paths. Thus, in addition to the varying capacity, there is also a large degree of varying load. Consequently, over-admission may occur on a link even if no new flows where accepted into the network. There are two alternative ways to deal with this issue. The first is to have a conservative admission limit. This will result in good performance for the admitted flows, but high rejection rates and low utilization of the network resources. The other alternative is to rely on a flow termination mechanism to remove flows when congestion occurs. This will lower the rejection rates and increase the utilization of the network resources, but lower the performance of the admitted flows.
- *Shared channel.* In fixed networks, a router has full control on its outgoing

links, but in wireless networks, the channel is shared with all the neighbor nodes within the sensing range. It then becomes a collective responsibility to ensure that the capacity is not exceeded on the shared channel around any node. When assessing whether a node can admit a new flow, it not only needs to evaluate the resource usage on the channel around itself, but also around all of its one hop neighbors. This is valid for all nodes along the path of the new flow. Consider the example in figure 3.2. A new flow is requested from A to C through B. The channel load as observed by node B is low, and there is room for the new flow. However, the new flow will also affect node D, which is a 1-hop neighbor of B and is located in a critical area; The load around D may be higher due to traffic on the *hidden paths* through node E. Consequently, for the correct decision to be taken, it is necessary to take into account the traffic load around nodes in the critical area. Probing or other signaling along the requesting flow's path only is not sufficient, as extra signaling between neighbors is needed.

- *Empty ingress-egress aggregates (IEAs).* In wireless networks, where flow sizes are typically large relative to the network capacity, only a small number of flows may be admitted in the system compared to fixed high capacity networks. This means the likelihood of having source-destination paths ("IEA" is PCN terminology) without any admitted flows is significant. As many admission control schemes (including PCN) is based on monitoring existing traffic flows in order to collect information about congestion status, the basis of decisions is missing when paths have no traffic. Often the solution is to simply admit new flows in these situations, which is likely to cause over-admission.

- *No distinct core/border separation.* In fixed DiffServ networks, scalability issues are avoided by having a fast and simple core, while the edge routers perform more complex per-flow tasks such as classification, admission control, policing etc. Distributed admission control schemes (e.g. PCN) normally involves signaling between the border routers. In a MANET, the distinction between edge and core nodes is not present in the same way. Instead, the nodes take different roles with respect to each flow. E.g, a node may operate as an ingress router for one flow, while having the role as an intermediate node for another flow. Consequently, all nodes are potential edge nodes. This impacts scalability for distributed admission control schemes since all node pair must exchange signaling.

The work in paper A is motivated by need for admission control solutions for MANETs dealing with these challenges, combined with the identified advantages of using the PCN mentioned above. The paper reviews the possibility of using PCN in MANETs, as well as suggesting improvements and adjustments in order to enhance the performance.

3.1.2 Related work of paper A

As mentioned above, no publications on the use of PCN in MANETs exist at the time of writing, except paper A. There is however a significant number of work done on admission control in MANETs in general, with dominating focus on 802.11 networks. The survey paper [21] by Hanzo et al. gives an extensive summary of 28 proposed admission control schemes for 802.11-based MANETs. Here, the different schemes are categorized into two main groups: *routing coupled* and *routing decoupled*. An admission control scheme coupled with routing means that admission decisions depend on information gathered by a QoS routing protocol. Such schemes typically utilize the route discovery mechanisms of reactive MANET protocols in order to find routes with sufficient resources. On the other hand, routing decoupled schemes do not have this dependency on routing protocols, but must collect the needed information on its own before making decisions. The use of PCN falls into the latter category, which can be further grouped into *stateless* and *stateful* approaches. Stateful schemes stores state information about individual flows at intermediate routers on the path between the source and destination (i.e. ingress and egress routers). This facilitates a fine grained control of resources at each node, but the added complexity and the amount of signaling required to maintain the state cause scalability problems. Stateless schemes avoid these issues by only maintaining state information at the end points, which conforms well with the PCN architecture. Recall that in PCN, intermediate nodes only do metering and marking on traffic aggregates, not on individual flows.

To summarize, PCN falls into the category of stateless, routing decoupled admission control schemes. Hanzo et al. [21] include three protocols in this category. These protocols, DACME [22], PMAC [23] and SWAN [24], are briefly described below.

DACME is based on periodic probing on all paths with admitted traffic using a packet train (10 packets back-to-back) each 3 seconds, in order to assess the available capacity on a route. As acknowledged by the authors, such probing is problematic in dynamic environments due to estimation bias and variance. On the one hand, numerous probe packets are needed to avoid variance, while on the other hand, such intrusive probing may degrade the performance of the network. As DACME uses probing, the empty IEA issue is avoided since empty paths are probed prior to making a decision. Channel variations are handled by periodically probing paths with admitted traffic. If the estimated QoS on a path decreases below the required value, flows may be rejected. DACME does not address the hidden path issue directly, as decisions are taken purely on the basis of the probe results. How a new flow will affect neighbor nodes along the path is not taken into account.

PMAC uses passive packet loss and delay measurements on data traffic to determine the quality of routes. Flows are initially admitted on paths without traffic, and the destination monitors the flow and reports the quality back to the source, which uses this to make admission decisions on later arriving flows.

Due to the passive nature of the quality estimation (as opposed to probing), PMAC is significantly less intrusive than DACME. However, if there is no traffic between a source-destination pair, PMAC knows nothing about the quality of the path, and is forced to admit new flows. Thus, PMAC suffers from the empty IEA issue. Using the quality estimate, PMAC is also able to terminate flows with poor quality, in order to adapt to varying channel conditions. Similar to DACME, PMAC does not address the hidden path issue, as resource usage at neighbor nodes is unknown.

The third scheme in the category of stateless routing-decoupled schemes, SWAN, divide the traffic into two traffic classes: *real-time* and *best effort*. The real-time traffic is governed by an admission control scheme which also uses probing. Probe packets are sent over the path, and intermediate nodes estimate their available capacity and updates the probe packets with the bottleneck capacity. In addition, intermediate routers mark packets if congestion is detected using the ECN bits. When being informed about marked flows, the source attempts to re-admit the flow. If this attempt fails, the flow is terminated. In SWAN, the best effort traffic is shaped using a leaky bucket with a rate governed by an additive increase multiplicative decrease (AIMD) scheme. The probing ensures that the empty IEA issue is avoided. Like the two other protocols, SWAN does not address the hidden path issue as the intermediate nodes only estimate their local resource usage. Of the three protocols, SWAN has most resemblance with PCN.

3.1.3 Contribution of paper A

The overall contribution of A is to study the application of the PCN architecture in MANETs. After presenting a simulation study, the paper concludes that with some key modifications / extensions, the IETF's PCN architecture is applicable in MANETs, and that the proposed mechanisms indeed improves the networks ability to avoid thrashing situations. The study shows that using the unmodified PCN architecture as described in the relevant RFCs, results in a poor performance. After surveying the challenges related to using distributed admission control (with PCN specifically) in MANETs, the paper proposes extensions/modifications to PCN which target the MANET challenges and improve the performance significantly.

The most important modification is to introduce probing. As long as there is traffic on an IEA, the egress sends periodic reports back to the ingress identical to the original CL edge behavior described in [12]. However, if an IEA has been idle for a longer interval, the egress stops sending feedback reports on that IEA. When the ingress observes this, it is forced to initiate a probing session if new admission requests arrives for the given egress. This results in a scheme which combines the periodic reports of original CL with probing, both reducing the amount of signaling, and preventing the empty IEA problem, which are identified in the paper as the two most important issues. With original CL, the periodic signaling is sent on all IEAs always. With the proposed probing scheme, the

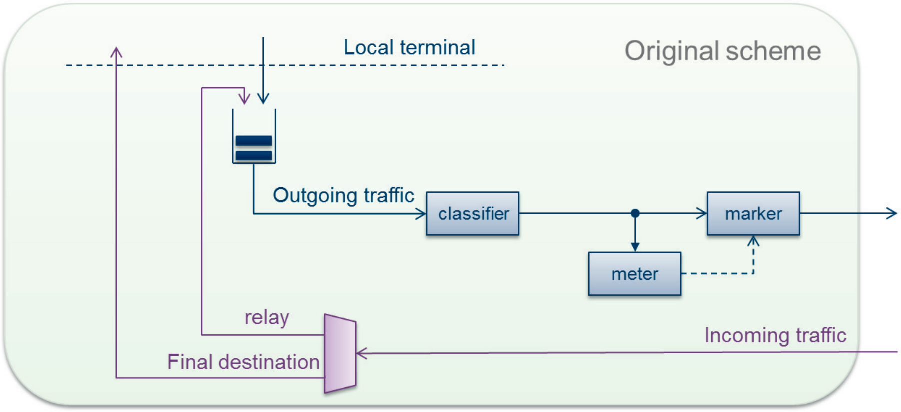


Figure 3.3. Original PCN metering and marking. Only outgoing packets are subject to metering and marking. Classifier ensures that only PCN traffic is fed to the meter and marker.

amount of signaling instead increases with the number of admitted flows (upper limited by the number of IEAs), which in most cases will represent a significant reduction. The problem with empty IEAs is avoided, as the probing session reveals the current status of the path prior to taking the decision. The proposed scheme uses only a single probe packet, which gives a relatively small delay increase and a low degree of intrusiveness. Probe packets are subject to normal metering and marking through the core, and the marking state is fed back to the ingress by the egress. It may however be favorable to use more than one probe packet for two reasons; to increase the robustness of probing in environments where packet loss is caused by random noise, or to mimic the requesting flow's characteristics. These topics are not investigated in the paper.

Paper A also proposes modifications to the PCN metering & marking algorithms described in [11] (threshold and excess markers). The original algorithms intended for wired networks meter only outgoing packets on a link (see figure 3.3). However, since it is now a shared channel that is being monitored, all packets on the channel must be metered, which includes both incoming and outgoing packets (see figure 3.4).

This metering and marking scheme does not take the into account the hidden path issue as explained in section 3.1.1, as the congestion status on the channel around neighbors along the path is unknown. Thus, the admittance of a flow may cause over-admission on nodes that are one-hop neighbors to the nodes along the requesting flow's path. This is a weakness of all the three protocols described in the related work section 3.1.2 as well (DACME, PMAC and SWAN). In the paper, it is suggested the possibility to feed 802.11's CTS (clear to send) signaling messages to the meter in order to assess the resource usage around 1

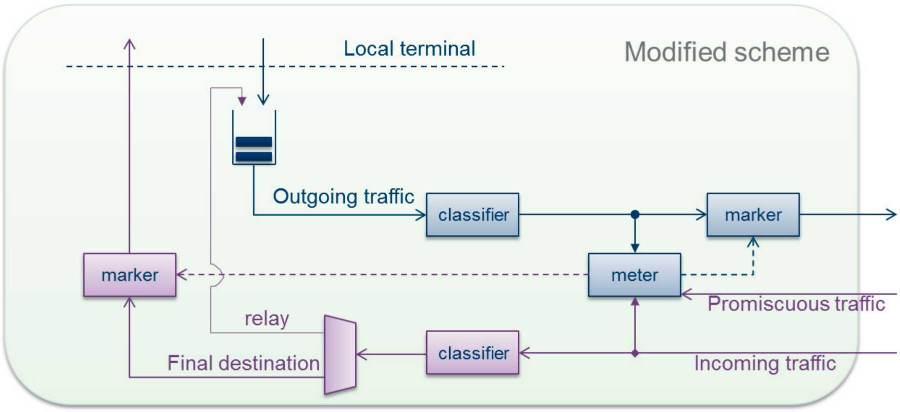


Figure 3.4. Modified PCN metering and marking. Metering is now done on all packets received on air. *Promiscuous traffic* refers to packets captured in promiscuous mode, not intended for this node. Additionally, marking is done when a packet reaches its final destination.

hop neighbors. This was not explored further, as it would cause an unwanted dependency on the link layer. An alternative would be to introduce more signaling between neighbor nodes.

Channel variations caused by mobility or background noise may cause link breaks and re-routing, potentially leading to over-admission on alternative paths. To some extent this can be handled by the flow termination mechanism. However, frequent terminations increase the degree of thrashing in the network, as the resources already spent by a terminated flow can to some extent be considered wasted, as discussed in section 1.2. Instead, it is important to have a more conservative admission threshold. In the paper, the admission limit was set to 0.7 Mbps on an 11 Mbps channel in a network with relatively low degree of mobility.

As also stated in the paper, there are still topics to be investigated further with respect to applying PCN mechanism in the MANET environment:

- The metering & marking algorithms should also take hidden nodes into account.
- Sensitivity analysis on the respective PCN parameters,
- Load control of best effort traffic is also needed, due to the shared channel environment.
- Comparative studies with admission control schemes dedicated for MANETs, in order to assess the difference in performance with PCN.

3.2 Transport level congestion control

3.2.1 Motivation for paper B

As explained in section 2.2, TCP-Friendly Rate Control (TFRC) is intended for elastic multimedia applications with need of a sending rate with less variations than what is offered by TCP, while still being TCP-friendly. As was also pointed out, TFRC has some weaknesses when sending layered/VBR video, mainly because TFRC's rate shaping causes jitter, and also because the regulation algorithm is dependent on inducing packet loss to function properly.

Paper B studies the use of the packet pair technique as an alternative regulation algorithm for scalable VBR streaming applications. As documented in [25], packet pair probing may be rather intrusive on cross traffic if an estimate of high accuracy is required. A high sampling intensity (frequent generation of packet pairs) is needed for two reasons: 1) More samples give a more accurate estimate. 2) Changes in the cross traffic are reflected faster. However, as the frequent probing may contribute to a significant load increase, actively inserting probes into the network with the sole intention of bandwidth estimation may be too costly.

It is however possible to piggyback packet pair estimation on existing packet streams, without inserting any extra packets into the network. Such *passive probing* is actually well suited for VBR video streaming due to the repetitive occurrence of key frames. Typically, at the start of each GOP, several packets are generated in order to accommodate the large amounts of data. These packets may very well be transmitted formed as packet pairs.

An important property of the TFRC protocol is TCP-friendliness, facilitating its use in environments where the bandwidth resources are shared with TCP-flows. There are however scenarios where this property is not needed, and the focus of the transport-level rate control (TRC) can be drawn towards application performance rather than being fair to TCP. One example is when multimedia flows are isolated from other traffic by the means of a dedicated QoS class. An example is a DiffServ Assured Forwarding (AF) class with bandwidth guarantees. Congestion within the class is handled by the individual flow's end-points.

3.2.2 Related work of paper B

3.2.2.1 Available bandwidth estimation

In section 2.2.2, the packet pair technique based on the Probe Gap Model (PGM) for estimating the bottleneck capacity was explained. In congestion control however, the goal is not to estimate the bottleneck link capacity as much as the *available bandwidth*. Informally, the available bandwidth can be defined as the bottleneck capacity subtracted the rate of the cross traffic:

$$A \equiv C_b - R_c \tag{3.1}$$

Several techniques have been proposed to estimate this quantity, and can be divided into two main categories: Estimation based on PGM, and rate based probing. Rate based probing relies on self-induced congestion and involves sending with a number of different rates and identifying the point at which the received rate is lower than the original sending rate. This is not suitable for video congestion control, as continuously running such probing sessions during a flow's lifetime would be too intrusive on the cross traffic.

The PGM techniques have in common that they estimate the available bandwidth by measuring the output gap, and try to infer how much of the gap increase is caused by the bottleneck capacity, and how much is caused by intervening cross traffic packets. If assuming a priori knowledge about the bottleneck capacity, this becomes trivial. Examples of PGM tools are *IGI* [26], *Spruce* [27] and *pathChirp* [28]. The main difference between these tools is the way they deal with estimation bias and variance. The estimators are usually unbiased only when all of the following conditions hold:

- there is only one bottleneck link
- this link is both the narrow[1] and the tight[2] link
- both packets in a pair arrives to the link in the same busy-period.

If these assumptions are not met, which is likely in today's networks, the estimate becomes biased. Also, as shown in both paper B and confirmed in the independent performance evaluation in [25], the sample variance for a single pair (i.e. Spruce) is very high when exposed to variable rate cross traffic. To compensate for this variance, numerous samples are required. This either leads to long waiting period before the estimate is available, or high impact on the cross traffic if sending many pairs within a small time period.

Due to the variable and unpredictable nature of packet switched networks, it is impossible to totally eliminate these inaccuracies. For this reason, a congestion control mechanism cannot solely rely on packet pair estimates. However, they may provide valuable information to increase-decrease algorithms, facilitating more intelligent decision-taking when adjusting the sending rate.

3.2.2.2 Packet pair piggybacking

In [29], the authors suggested to send all of these N packets back-to-back in a *packet train*, using the following estimator for the available bandwidth:

$$\hat{A}_1 = \frac{1}{\tau}\sum_{i=1}^{N-1} s_i \tag{3.2}$$

[1] the link on the path with the smallest capacity

[2] the link on the path with the smallest available bandwidth

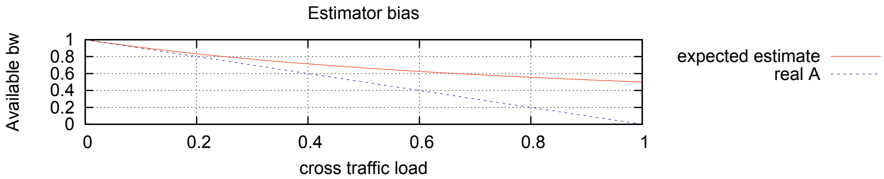


Figure 3.5. k

where s_i is the packet size of the individual packets in the train, and τ is the time interval between the first and the last probe packet measured at the receiver. Unfortunately, this estimator is strongly biased. As shown in appendix A, the expectation of the estimator is

$$E\left[\hat{A}_1\right] = \frac{C_b}{1 + \frac{R_c}{R_a}} \tag{3.3}$$

where R_a is the train's arrival rate to the bottleneck link. This clearly shows that the estimator is biased, as $E[\hat{A}_1] \neq C_b - R_c$. If assuming that the train is sent out with $R_a = C_b$ instead of sending the packets back-to-back, figure 3.5 illustrates how the estimator over-estimates the available bandwidth as the cross traffic load increases.

If substituting R_a with C_b in equation 3.3 and solving for $C_b - R_c$, a new estimator compensating for the bias of $\hat{A}_1$ is:

$$\hat{A}_2 = 2\hat{C}_b - \frac{\left(\hat{C}_b\right)^2}{\hat{A}_1} \tag{3.4}$$

3.2.3 Contribution of paper B

Paper B proposes an alternative TRC solution based on packet pair estimation of available bandwidth, named Packet Pair Assisted Congestion Control (PPACC). Similar to the proposed packet train technique described in [29], PPACC is non-intrusive on cross traffic, as packet pairs are formed using the video packets when possible. Different from related work, which use the estimated rate directly to control the sending rate, PPACC only use the estimate as a guideline in order to control the magnitude of increase/decrease of the sending rate.

As explained above, the packet train estimation technique is biased when using the $\hat{A}_1$ estimator. In PPACC, the $\hat{A}_2$ estimator is used instead, and the estimation is performed on packet pairs rather than trains (i.e. $N = 2$)[3]. A packet pair is formed whenever two packets are ready for transmission at the

[3]Note that $\hat{A}_2$ is mathematically equivalent with the Spruce estimator if setting $N = 2$, $\Delta_{in} = s/C_b$ and $\Delta_{out} = \tau$

same time. A header-bit is set if a packet is the first in a pair, and the second packet is held back in order to satisfy $\delta = s/C_b$.

Most importantly however, the estimates are merely used to weight the amount of rate adjustment instead of directly sending with the estimated rate as suggested in [29]. The reader is referred to paper B for a more detailed description of the proposed congestion control mechanism.

In the paper, the performance of PPACC was compared with TFRC using a simulator model developed in NS2. The packet pair estimation technique was shown to perform significantly better than TFRC in terms of packet loss, delay and jitter. This indicates that it is possible to serve jitter- and delay sensitive traffic with a DiffServ AF PHB if a proper TRC is used.

The simulations presented in paper B confirms the weaknesses of TFRC when the application sends with a variable and/or adaptive sending rate. It was shown that TFRC requires relatively large buffers in the routers to avoid high packet loss. In addition, it was shown that TFRC caused significant jitter. This resulted from queuing delay in the send buffer at the source due to the mismatch between TFRC's sending rate and the applications instantaneous sending rate. Consequently, TFRC is not suited for VBR video applications with strict delay requirements.

3.3 Application level congestion control

3.3.1 Motivation for paper C

Congestion control algorithms normally probe for available bandwidth by gradually increasing the sending rate until packet loss is observed. Retransmissions are then used to recover the lost packets. For multimedia applications with strict delay requirements, this is not an option, as retransmissions are likely to arrive too late. An important goal for an ARC is then to minimize packet loss caused by congestion, by being conservative when probing for bandwidth. However, it is impossible to completely avoid congestion loss, so precautions must be taken in order to reduce the quality degradation when this happens. In this respect there are two main classes of techniques; error concealment and error recovery. The former spans receiver-side techniques where the missing information is approximated based on available information in neighboring video frames. Error concealment techniques are codec-dependent and are out of the scope of this thesis.

Error recovery on the other hand, can be applied independently of the codec in use, and includes techniques where an exact copy of the lost information is reproduced. Here, redundancy is added to the packet stream, using some variant of forward error correction (FEC). There exists a number of different FEC codes, but the class of systematic block codes has properties that are favorable when applied to packet streams sent over the internet. Here, the input to the FEC encoder is k source packets (symbols) and the output is $n-k$ coded packets containing redundant information from the original packets. The total of n packets are transmitted, and the FEC decoder is able to reproduce any dropped source packet as long as at least k out of the n packet arrives at the receiver.

By adding *fixed FEC*, i.e. constant values for n and k throughout the flow's lifetime, the sending rate is increased by a factor $\frac{n}{k}$. For a given channel packet loss probability, the effective packet loss probability will decrease when this factor grows. However, if a dominant part of the traffic volume on a congested link is protected by FEC, the total load on that link may be significantly higher than in the unprotected case, resulting in a higher channel packet loss probability. For this is reason, using FEC as a measure for protection against congestion loss must be done with great care. Paper C emphasizes this by investigating the effect of adding fixed FEC both analytically and with simulations.

3.3.2 Related work of paper C

There are many proposals for variants of FEC applied to video streaming in the literature, where [30–33] are examples. They all have in common that they assume that the rate-increase caused by the redundancy does not affect the channel packet loss ratio. Some of them ([30–32]) even suggest to increase the FEC overhead when the packet loss probability increases. Obviously, if the packet loss is caused by congestion, increasing the load on the congested link will eventually lead to a congestion collapse.

The same issue partially applies for solutions based on *joint source-channel coding* [34]. Here, the channel rate and the channel packet loss probability are given, and the problem of finding the optimal division between source rate (entropy) and channel rate (redundancy) is addressed, often formulated as optimization problems. An assumption frequently made, is that variations in the channel loss probability are caused by external processes (cross traffic or wireless interference), rather than by congestion induced by the protected flows themselves. It is not taken into account that reducing the total sending rate may in fact result in a lower packet loss rate and a better quality for the end user. The chance of congestion collapse is however eliminated, because of the limitation on the maximum sending rate.

In [35], Cidon et al. study the packet loss distribution in single server queuing systems, and emphasize the danger of assuming independence in the packet loss process. It is demonstrated that this often leads to optimistic results when estimating the error correcting capabilities of FEC schemes. Also, the authors strongly argue against the use of FEC because of the increased load imposed by the overhead. The arguments are however based on results from ATM examples, not necessarily applicable to today's IP networks.

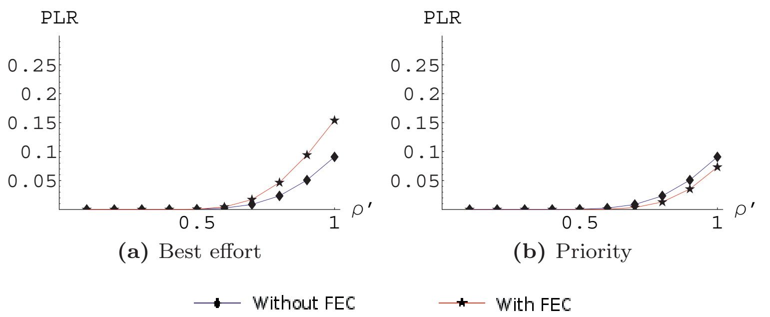


Figure 3.6. Theoretical EPLR with and without FEC in a single server queue (M/M/1) with room for 10 packets, and when all flows in the system use FEC.(The x-axis show the traffic load before adding FEC.)

3.3.3 Contribution of paper C

Paper C studies the effect of adding FEC on the packet loss rate observed by the application (i.e. after FEC decoding). It is assumed that the flows protected with FEC represent a large part of the total number of flows in the network. It is further demonstrated both analytically and with simulations, that in best effort networks, the use of FEC is a bad choice when the intention is to avoid packet loss caused by congestion. In fact, the increased load caused by the redundancy is likely to cause higher Effective Packet Loss Ratio (EPLR) compared to only transmitting the source data. It was found that only when having very small buffers in the intermediate routers, a FEC gain was observed. An explanation is that when having small buffers, packet loss occurs also at lower load. These losses are typically not caused by congestion, but by buffers not being large enough to handle the bursty VBR video flows.

When using systematic FEC, source packets are more valuable than the coded packets. A coded packet is only of value if k or more packets arrive at the destination, while received source packets are always valuable as they can be decoded alone. Based on this observation, paper C also studied the use of FEC when the network has priority support, assigning a lower priority to coded FEC packets. This contradicts with the usual way of assigning priority to scalable video, where there is often a static mapping between priority levels and enhancement layers. The analytical model was modified in order to account for the difference in packet loss ratio (PLR) for the two types of packets, and the results showed that assigning a lower priority to coded FEC packets has a positive effect on the performance. An example is shown in figure 3.6 (copied from the paper). Figure 3.6a clearly illustrates the danger of using FEC in congested best effort networks, as the effective PLR increases as a result of the added load. However, when assigning lower priority to the coded packets, figure 3.6b shows that there is in fact a gain from adding redundancy, even at loads close to 1. This was also confirmed by the simulation results.

To some degree, the findings in paper C confirms the concerns expressed by Cidon et al. [35] for using FEC. However, if the packet loss is not caused by congestion, if only a smaller portion of the traffic is protected by FEC, or if the coded packets are assigned to a lower priority, there may indeed be a performance gain.

3.3.4 Motivation for paper D

As argued in section 2.3, there are good reasons for separating the application-level rate control (ARC) and transport-level rate control (TRC). The TRC is mainly responsible for avoiding congestion in the network, while the ARC's responsibility is to maximize the user perceived quality, given the constraints set by the TRC. When streaming scalable video, an optimal perceived quality is not necessarily obtained by maximizing the number of enhancement layers. Another important factor (in addition to spacial, temporal and SNR quality) is the frequency at which enhancement layers are dropped. A constant medium quality is often preferred over an alternating quality with a higher average. Consequently, care should be taken by the ARC when adding a new enhancement layer, as the increased sending rate may cause congestion and ultimately forcing the ARC to drop one or several layers shortly after the increase.

As already indicated, it is important for the ARC to have a good strategy for adding enhancement layers when available capacity is reported by the TRC. The challenge for the rate controller is to know how much and how often to increase the sending rate. If the rate increment is too large, there is a high risk that the congestion threshold is reached, resulting in delay and packet loss and consequently forcing the layer to be dropped. In turn, this contributes to high variations in quality.

Some variant of *bandwidth probing* is commonly used; a careful progression of the sending rate until experiencing a congestion event. This strategy is however problematic when sending layered video. The time period lasting while waiting for the allowed sending rate to reach the level of the next layer can be several round-trip times. During this period the sending rate is kept constant, and the available bandwidth estimate which relies on congestion events gets out-dated.

To summarize, there is a need for an ARC which delivers good perceived quality for layered video applications while avoiding the issues associated with traditional bandwidth probing.

3.3.5 Related work of paper D

Paper D proposes a complete ARC solution which contributions can be categorized into three areas:

- The quality metric to be optimized by the control algorithm
- The control algorithm
- How to reduce the effect of packet loss

In the following sections, related work within the three areas are summarized.

3.3.5.1 Video quality metrics

A video congestion controller's prime concern is to optimize the video quality while ensuring that no more than the flow's fair share of network resources is used. Unfortunately, the term *video quality* is ambiguous in this context. In the end, it is the individual user's perceived quality that is to be maximized. When designing and evaluating a congestion control algorithm however, extensive use of subjective testing is usually too costly. Instead, objective quality assessment is most often performed, either by using easily measurable QoS parameters such as packet loss ratio (PLR), delay and jitter, or by using objective video quality metrics. The latter attempts to quantify the degradation of the resulting video with as strong correlation to subjective quality as possible. Subjective quality is typically expressed using Mean Opinion Score (MOS).

The most popular objective quality metric is peak signal-to-noise ratio (PSNR), measuring the ratio of the maximum signal power to the power of the interfering noise. The noise may either be information loss caused by compression, or by a lossy transmission channel (including congestion loss). Each frame of the original video is compared to the corresponding decoded frame.

When evaluating different congestion control solutions, it is a common approach to compare the average PSNR over a sequence of frames, and conclude that the solution with the highest average is the preferred one. However, as documented in the subjective tests presented in [36] as well as in [37], viewers tend to prefer constant quality near the average instead of an alternating quality. The quality change per-se represents an annoyance factor, stealing the viewer's attention away from the content. For this reason it is important to also consider measures of variability when evaluating competing solutions. One option is to calculate the variance of PSNR. If assuming that the reference signal is the full set of encoded enhancement layers, distortion to this signal is either caused by enhancement layers being dropped, or by packet loss/delay. From a QoE view, it is not important whether variance in PSNR is caused by one or the other. However, when evaluating and comparing different protocols, it is often preferable to be able to distinguish the two in order to explain the underlying processes. Also, calculating PSNR requires the actual video to be available both before and after transmission. This is usually manageable in a laboratory experiment setting, but more difficult when conducting simulation experiments. For these reasons, alternative measures of quality variations for scalable video is needed.

3.3.5.2 Control methods

Receiver buffer control A commonly used technique is to monitor the receiver's playout buffer. The primary function of this buffer is to hide jitter from the application, but it is also being used for rate control. The basic idea is that when packets are dropped or delayed in the network, the temporal filling of the playout buffer decreases. This ultimately causes buffer under-runs unless correct actions are taken. The typical approach is that if the filling is reduced below a threshold, the rate is decreased and the buffer will gradually start to grow. Ssesanga et al. [38] applied these ideas to scalable video congestion control. Here, the playout buffer is monitored, and when congestion causes the threshold to be reached, the FGS enhancement layer is cropped/reduced. When the filling is back to the target level, the enhancement layer is gradually increased. Unfortunately, their approach is only applicable to CBR video as the control method relies on bit-wise filling of the buffer, and not only temporal filling. Generally, the main challenge with playout buffer control is to know how much of the enhancement information to add in the increase phase, because there is no underlying estimate of the available bandwidth.

Virtual network buffer The virtual network control method was introduced by Xie and Zeng in [39], targeting rate control based on bitstream switching. The virtual network buffer is located between the video server and client application, and is an abstraction of the underlying network's complex collection of links and queues. The model accounts for both delay and packet loss, and provides constraints for the sending rate in order to prevent late arriving frames at the client. In [40], Zhu et al. combines the ideas from [39] with a modified version of TFRC called TCP-Friendly Rate Control with compensation (TFRCC). TFRCC is identical to TFRC in the way it calculates the TCP-friendly sending rate. However, the fundamental difference is that it allows the application source to send with a higher rate if the TCP-friendly-rate violates the constraints given by the virtual buffer model. A compensation technique is applied in order to ensure long-term TCP-friendliness. Thus, this is a typical cross-layer solution where the application and transport layer take a joint decision about the sending rate. FGS encoded video is assumed, thus the impact of sudden large increase in the source rate caused by adding an enhancement layer is not addressed.

Periodic probing In order to handle large increases in sending rate, which is typical for layered video, one approach is to perform probing experiments; A new layer is added only after a successful probing session. In [41], Liu and Hwang presented the Bandwidth Inference Congestion control (BIC) protocol. It is designed for multicast video streaming, where the video layers are sent in separate multicast groups. The sender periodically probes for spare capacity by including the packets from layer i+1 into the multicast group for layer i, effectively increasing the sending rate corresponding to the rate of layer i+1. To infer whether the available bandwidth can accommodate the new layer or not, the BIC receiver uses delay-trend estimation derived from *pathload* [42, 43], before possibly join the group containing layer i+1. Wang and Hsiao [44] proposed a similar scheme based on BIC, with a modified control algorithm claiming increased TCP-friendliness.

As intended, BIC achieves a lower Layer Decrease Frequency (LDF) compared to additive increase multiplicative decrease (AIMD) based algorithms. However, the intra-probe interval in both of the above proposals is not determined based on user perception, but on network statistics. Thus, there is still a chance that layers may be added and dropped too frequently with respect to perceived quality, especially when the background and/or foreground flows are highly variable. Another issue is intrusiveness. Long term intrusiveness is avoided by adjusting the interval between probes. During a probe however, the sending rate may be drastically increased, increasing the probability of packet loss both for the probing flow itself and for the cross traffic flows.

3.3.5.3 Reducing the effect of packet loss

An important objective for the ARC is to maintain a low Effective Packet Loss Ratio (EPLR), i.e. the packet loss observed at the application layer. Packet loss may be harmful to observed quality, especially when key frames are missing, which other neighboring frames depend on. In the literature, proposed solutions often fall into two categories; Either, redundancy is added to the packet stream, or alternatively, packets regarded as more important receive prioritized treatment in the underlying network:

As discussed in paper C, applying FEC to the video stream is a frequently used alternative to packet retransmission. However, as argued in the paper, if the FEC overhead occupies a too large amount of the capacity on the congested link, the effective packet loss ratio may be higher than if the redundancy was omitted. To avoid this situation, it is desirable that the total amount of FEC overhead is reduced without sacrificing the ability to recover lost packets.

The key to this is to predict when the need for FEC is higher, and apply stronger protection in periods where packet loss is anticipated. In [45], Seferoglu et al. presents a technique where FEC is added to a TFRC flow in periods where it is expected that the competing TCP flows will cause congestion. When the TCP flows perform their bandwidth probing (in the congestion avoidance phase), the gradually increasing sending rate eventually causes network queues to fill up. This can be observed by the TFRC flow as a gradually increasing delay. By estimating the delay and its derivative and observing the correlation with packet loss, a pattern can be identified. This can further be used in order to predict when congestion loss will occur, and a larger part of the available bandwidth can be allocated to redundancy. Simulation results are presented indicating that the proposed method outperforms fixed overhead FEC.

Alternatively, packets which are considered more important than others get elevated priority in the underlying network (e.g. DiffServ drop precedence or 802.11e Access Category). Examples are given in [46–50]. These solutions all have in common that SVC enhancement layers are mapped on to the underlying priority mechanism. A general problem is that when using too many priority levels, the packet schedulers' ability to differentiate between them, in terms of packet loss and delay, is reduced. This is documented in [51] by Moseng et al, also emphasizing the fact that if the traffic is variable, the separation between the classes gets even worse .

3.3.6 Contribution of paper D

In paper D, an ARC for scalable video targeting the issues presented in the previous section, is presented. It is designed to be located between the TRC and applications as explained in the introduction of section 3.3, and is termed a "generic ARC" reflecting the independence of the upper and lower parts of the protocol stack. A central part of the solution is to conduct probing experiments prior to adding new enhancement layers to the traffic flow. During a probing period, a *probe layer* is added, and the sending rate is increased in order to mimic the addition of the next enhancement layer. Only if the outcome is successful, the new layer is added. The ARC also includes two *shadow probing* techniques in order to reduce the effect of packet loss. Details about the proposed ARC and how it contributes in the three areas are given below:

3.3.6.1 Video quality metric

As stated in section 3.3.5, it is important to have in mind that frequent additions and removals of enhancement layers reduce the viewing experience. Paper D introduces the Layer Decrease Frequency (LDF) metric, which simply counts the number of enhancement layers dropped per second, and is used when evaluating the ARC performance. In the proposed ARC, the LDF is controlled by explicitly setting the minimum time between probes. This means that even if the TRC reports an allowed sending rate capable of accommodating a new enhancement layer, the ARC will hold back the layer increase until the configurable amount of time has past.

If a large value is chosen, the ARC will be more conservative with respect to adding new enhancement layers. This will reduce the chance of congestion, lowering the probability of being forced to drop enhancement layers shortly after the layer increase. One cost of this is a reduced utilization of the network resources. Fairness will also be challenged, as it will take a longer time to reach a stationary average sending rate. Thus, adjusting the minimum time between probes is a way of trading off utilization and fairness with perceived quality.

3.3.6.2 Control method

As stated above the ARC depends on an underlying TRC that provides an estimate of the available bandwidth. The paper assumes that TFRC is used, taking responsibility of the long term TCP-friendliness. Further, in order to handle layered video with significant differences in sending rate among the layers, the ARC performs probing experiments before possibly adding an enhancement layer. At the beginning of each Group Of Pictures (GOP), the ARC compares an estimate of the *near future peak sending rate* with the allowed sending rate reported by the TRC. If the future sending rate is larger than the TRC rate, the ARC will drop enhancement layers until the constraints of the TRC is fulfilled. However, if the TRC rate is large enough to accommodate a new enhancement layer, the ARC *may* initiate a probing session depending on the time interval since the previous probing event (successful or unsuccessful). This controls the LDF as described in the previous section, and is an improvement compared to BIC's probing interval which is dependent on network statistics rather than user perception.

As mentioned above, the ARC compares the TRC rate with an estimate of the application's *near future peak sending rate*. This is done in order to support VBR video. In the simulator, *near future* was defined to be approximately 3 seconds (10 GOPs). The accuracy of this estimate depends on the tolerable delay of the application. For an interactive video conference, the estimate will rather be a prediction based on previous GOPs. For a non-interactive application, a better accuracy is obtained by buffering GOPs on the sender side.

Another challenge when performing active probing experiments is intrusiveness on cross traffic. If the probing rate is higher than the available bandwidth, both the probing flow itself and the competing flows may experience quality degradation during the probing session. This is also an issue with BIC, as argued in section 3.3.5.2. The proposed ARC attempts to reduce the intrusiveness both proactively by having the minimum time between probes parameter, and reactively by applying techniques to reduce the effect of packet loss. The latter is described in the next section.

3.3.6.3 Reducing the effect of packet loss

Paper D introduces two shadow probing techniques applied by the ARC in order to reduce the effect of the packet loss caused by probing sessions. One is based on the use of FEC, protecting only the probing flow itself, while the other one is based on utilizing the underlying network's ability to prioritize traffic. The two techniques are described below.

FEC shadow protection As argued earlier, predicting when FEC is needed as opposed to having fixed FEC is preferable. With the proposed ARC, the concept of prediction is maintained, but instead of predicting the bandwidth increase of competing cross traffic flows, FEC is added by the ARC during the probing sessions, as this is when it is likely that the sending rate overshoots the available bandwidth. The sending rate during the probing session, the *probe rate*, should be equal to the sending rate when including the enhancement layer(s) intended for addition. This difference between the probe rate and the current sending rate is then to be filled up with redundancy. Consequently, the level of redundancy, or the *FEC code rate*, is given, which is different from most other use of adaptive FEC where the task usually is to optimize the code rate. It is assumed that a systematic block code is used (e.g. Reed-Solomon). The block length k is a configurable parameter reflecting the trade-off between delay and ability to correct burst loss. As the code rate k/n is given, it is trivial for the ARC to decide the value of the other parameter, n. Refer to the paper for a detailed explanation of how this is done.

Simulation results show that the effective packet loss, both overall and during probing, is significantly reduced when filling the probe layer with FEC packets.

Priority shadow protection Shadow probing with FEC only protects the probing flow itself during probe periods. For protection against packet loss caused by other flows, a method similar to the one in [45] may be used, adjusted for the right type of cross traffic. Alternatively, if priority mechanisms are available in the network (e.g. DiffServ drop precedence), a lower priority may be assigned to the packets belonging to the probe layer.

As discussed in section 3.3.5.3, the idea of using DiffServ drop precedence levels to provide stronger protection for more important video layers is not new. However, the novelty in the scheme proposed in D is in the way layers are assigned importance. Traditionally, the base layer is seen as most important, and each succeeding enhancement layer is considered less and less important. For multi dimensional scalable video such as H.264/SVC however, where the number of layer combinations can be quite high, it may be challenging to do the mapping between importance and the four precedence levels (available for DiffServ). Further, when using as much as four priority levels, the separation between them may become too weak, as pointed out in section 3.3.5. For these two reason, the paper proposes to only use two priority levels; high priority for the established layers, and low priority for the probe layer. This way, the established layers are being better protected against congestion for two reasons: First, since the probe layer has a lower priority, the increased sending rate of a probing flow is less likely to cause performance degradations for the competing flow's established layers. Second, having only two priority levels gives a maximal separation between them in terms of likelihood of congestion events. It may be possible to use a third level by assigning the highest priority to the base layer, but this is subject to further study. In general, a FEC scheme base on unequal error protection (UEP) should be used in order to provide different protection to the different layers, using the priority mechanism for the sole purpose of avoiding loss caused by probing. Simulation results show that shadow protection using the priority mechanism clearly gives the best performance. However, if only a subset of the network path has priority support, using a combination[4] of the two shadow probing techniques may be the preferred choice. I.e., marking all FEC packets with the lowest priority.

[4]When using priority shadow protection, the content of the probe layer can either be the actual new layer, or it can be the FEC shadow layer.

Chapter 4

Conclusions

This thesis has emphasized the need for congestion control mechanisms for multimedia flows requiring both low jitter and low end-to-end delays. Such mechanisms must not only satisfy the user in terms of high QoE, but also ensure that the underlying network is stable. It is important that the network does not end up in a thrashing state, where resources are wasted on work that does not contribute to the user's QoE.

From historically being a transport level mechanism, congestion control functionality for multimedia applications is now located in several layers of the protocol stack. Inelastic applications without the ability to adapt its sending rate requires network level support in order to avoid thrashing, in form of a QoS architecture. On the other hand, elastic applications can use end-to-end congestion control in order to reach the same goal. Congestion control functionality for such rate adaptive sources can be divided into a transport level component, and an application level component. The former ensures that the flows do not exceed the estimated available network capacity, maintaining fairness and network stability. The latter ensures that the QoE is maximized given the restrictions of the underlying transport level congestion control. The thesis argues that the two components should be implemented separately in order avoid a myriad of protocols with interfering regulation schemes.

The possibility of using the recently standardized Pre-Congestion Notification (PCN) architecture in dynamic MANET environments has been studied. Due to the limited bandwidth and varying conditions, QoS mechanisms, especially admission control, are important when sending inelastic multimedia flows over a MANET. PCN admission control is originally designed for wired DiffServ networks. The thesis has however shown that with some modifications, PCN is an alternative. Most importantly, the use of probing on ingress-egress paths which have been idle over a longer period significantly reduces the amount of signaling. In addition, the probing provides fresh information about the congestion status on a path, which is imperative when the admission decision is being made. Future work includes studying the impact of the *hidden path* issue, where the admission decisions are taken without taking into account the channel load experienced by hidden nodes along the path. Additionally, the possibility to also use the PCN mechanisms to control the best effort traffic should be investigated.

Within the area of transport level congestion control, the use of packet pairs for estimating the available bandwidth has been studied. The packet pair algorithm was originally designed for estimating the bottleneck link capacity, while later, it has been modified to also take into account the cross traffic. By using existing video packets as packet pairs, an estimate of the available bandwidth is achieved without inserting any extra packets in the network. Unfortunately, the estimate often suffers from inaccuracies. Therefore, a congestion control algorithm was proposed, named Packet Pair Assisted Congestion Control (PPACC), which uses the packet pair estimates merely as guidelines instead of directly controlling the sending rate. PPACC was compared to TCP-Friendly Rate Control (TFRC), and simulations showed that the proposed protocol outperforms TFRC, with significantly lower packet loss, delay and jitter. PPACC is not designed to be TCP-friendly, so it is not a candidate for large scale use over the open internet. However, it its use is more attractive in isolated environments such as within a DiffServ Assured Forwarding (AF) class. This way, it may be possible to serve flows with strict requirements on delay and jitter without adding the complexity of per flow admission control.

The thesis has also contributed within the area of application level congestion control. It has been pointed out that it is generally a bad idea to combat congestion loss with fixed forward error correction (FEC). FEC is often seen as an attractive alternative to retransmission when there are strict delay and jitter requirements. Unfortunately, the inserted redundancy represents a load increase which may lead to more severe congestion and result in an even larger effective packet loss observed by the application. It was however shown that by assigning a lower priority to FEC packets, this effect could be reduced.

The above findings lead to the studies on shadow probing. When a rate-adaptive source using scalable (layered) video adds an enhancement layer, the increase in sending rate may be significant. Consequently, congestion loss may follow. The thesis proposes to use two shadow probing techniques in order to reduce the impact of such loss. A probing period is introduced before the actual addition of the new layer. First, the shadow layer can be filled with redundancy from the already established layers, and thereby lowering the probability of quality degradation of the flow. The second technique instead aims at protecting the established layers of the competing flows, by assigning a lower priority to the probe layer (e.g. FEC packets).

The probing scheme was integrated with an application-level rate control (ARC) operating between the video application and the transport level congestion control (TFRC was used in this study). The ARC's task is to maximize QoE while obeying to the rate limit obtained from the transport layer. It is emphasized in the thesis that high variations in quality has a negative impact on QoE. For this reason, a strong focus was kept at avoiding being too aggressive when adding enhancement layers, even when the underlying transport layer reported available capacity.

Simulations results showed that the proposed shadow probing techniques indeed improve the performance of layered video flows with high delay and jitter requirements. Further work includes studying the proposed ARC in combination with alternative transport layer protocols, such as PPACC. In additions, larger scale simulation experiments, with more realistic traffic pattern, should be conducted.

Overall, the thesis has provided insight into the broad research area of congestion control. Several issues have been highlighted, and solutions have been proposed. Throughout the thesis, the focus has been on maximizing the quality of multimedia flows with strict delay and jitter requirements, and doing this while taking into account the stability of the underlying network.

References

[1] V. Jacobson, "Congestion avoidance and control," in *ACM SIGCOMM '88*, Aug 1988, pp. 314–329.

[2] S. Khorsandroo, R. Noor, and S. Khorsandroo, "A generic quantitative relationship between quality of experience and packet loss in video streaming services," in *Ubiquitous and Future Networks (ICUFN), 2012 Fourth International Conference on*, 2012, pp. 352–356.

[3] M. Fiedler, T. Hossfeld, and P. Tran-Gia, "A generic quantitative relationship between quality of experience and quality of service," *Network, IEEE*, vol. 24, no. 2, pp. 36–41, 2010.

[4] Digital Cinema Initiatives (DCI) and The American Society of Cinematographers (ASC), "StEM mini-movie access procedure (wedding scene)," November 2004. [Online]. Available: http://www.dcimovies.com

[5] Joint Video Team (JVT), "JSVM reference software version 5.2."

[6] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Netw.*, vol. 1, no. 4, pp. 397–413, Aug. 1993. [Online]. Available: http://dx.doi.org/10.1109/90.251892

[7] V. Jacobson, K. Nichols, and K. Poduri, "An Expedited Forwarding PHB," RFC 2598 (Proposed Standard), Internet Engineering Task Force, Jun. 1999, obsoleted by RFC 3246. [Online]. Available: http://www.ietf.org/rfc/rfc2598.txt

[8] K. Nichols, V. Jacobson, and L. Zhang, "A Two-bit Differentiated Services Architecture for the Internet," RFC 2638 (Informational), Internet Engineering Task Force, Jul. 1999. [Online]. Available: http://www.ietf.org/rfc/rfc2638.txt

[9] P. Eardley, "Pre-Congestion Notification (PCN) Architecture," RFC 5559 (Informational), Internet Engineering Task Force, Jun. 2009. [Online]. Available: http://www.ietf.org/rfc/rfc5559.txt

[10] B. Briscoe, T. Moncaster, and M. Menth, "Encoding Three Pre-Congestion Notification (PCN) States in the IP Header Using a Single Diffserv Codepoint (DSCP)," RFC 6660 (Proposed Standard), Internet Engineering Task Force, Jul. 2012. [Online]. Available: http://www.ietf.org/rfc/rfc6660.txt

[11] P. Eardley, "Metering and Marking Behaviour of PCN-Nodes," RFC 5670 (Proposed Standard), Internet Engineering Task Force, Nov. 2009. [Online]. Available: http://www.ietf.org/rfc/rfc5670.txt

[12] A. Charny, F. Huang, G. Karagiannis, M. Menth, and T. Taylor, "Pre-Congestion Notification (PCN) Boundary-Node Behavior for the Controlled Load (CL) Mode of Operation," RFC 6661 (Experimental), Internet Engineering Task Force, Jul. 2012. [Online]. Available: http://www.ietf.org/rfc/rfc6661.txt

[13] A. Charny, J. Zhang, G. Karagiannis, M. Menth, and T. Taylor, "Pre-Congestion Notification (PCN) Boundary-Node Behavior for the Single Marking (SM) Mode of Operation," RFC 6662 (Experimental), Internet Engineering Task Force, Jul. 2012. [Online]. Available: http://www.ietf.org/rfc/rfc6662.txt

[14] M. Menth, F. Lehrieder, B. Briscoe, P. Eardley, T. Moncaster, J. Babiarz, A. Charny, X. Zhang, T. Taylor, K.-H. Chan, D. Satoh, R. Geib, and G. Karagiannis, "A survey of pcn-based admission control and flow termination," *Communications Surveys & Tutorials, IEEE*, vol. 12, no. 3, pp. 357 –375, July 2010.

[15] K. Ramakrishnan, S. Floyd, and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP," RFC 3168 (Proposed Standard), Internet Engineering Task Force, Sep. 2001, updated by RFCs 4301, 6040. [Online]. Available: http://www.ietf.org/rfc/rfc3168.txt

[16] E. Kohler, M. Handley, and S. Floyd, "Datagram Congestion Control Protocol (DCCP)," RFC 4340 (Proposed Standard), Internet Engineering Task Force, Mar. 2006, updated by RFCs 5595, 5596, 6335. [Online]. Available: http://www.ietf.org/rfc/rfc4340.txt

[17] S. Floyd, E. Kohler, and J. Padhye, "Profile for Datagram Congestion Control Protocol (DCCP) Congestion Control ID 3: TCP-Friendly Rate Control (TFRC)," RFC 4342 (Proposed Standard), Internet Engineering Task Force, Mar. 2006, updated by RFCs 5348, 6323. [Online]. Available: http://www.ietf.org/rfc/rfc4342.txt

[18] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "TCP friendly rate control (TFRC): Protocol Specification. IETF RFC 5348," Sep 2008.

[19] J. Padhye, V. Firoiu, D. F. Towsley, and J. F. Kurose, "Modeling tcp reno performance: A simple model and its empirical validation," *IEEE/ACM Transactions on Networking*, vol. 8, no. 2, pp. 133–145, April 2000.

[20] V. E. Paxson, "Measurements and Analysis of End-to-End Internet Dynamics," PhD Dissertation, University of California, Lawrence Berkeley National Laboratory, April 1997. [Online]. Available: citeseer.ist.psu.edu/paxson97measurements.html

[21] L. Hanzo II and R. Tafazolli, "Admission control schemes for 802.11-based multi-hop mobile ad hoc networks: a survey," *Communications Surveys & Tutorials, IEEE*, vol. 11, no. 4, pp. 78 –108, October 2009.

[22] C. T. Calafate, J. Oliver, J.-C. Cano, P. Manzoni, and M. P. Malumbres, "A distributed admission control system for manet environments supporting multipath routing protocols," *Microprocess. Microsyst.*, vol. 31, no. 4, pp. 236–251, Jun. 2007. [Online]. Available: http://dx.doi.org/10.1016/j.micpro.2007.01.002

[23] C. McCann, G. Elmasry, B. Russel, and B. Welsh, "A measurement-based approach for multilevel admission of heterogeneous traffic in wireless ad-hoc networks," in *Military Communications Conference, 2004. MILCOM 2004. 2004 IEEE*, vol. 3, 2004, pp. 1562–1565 Vol. 3.

[24] G.-S. Ahn, A. Campbell, A. Veres, and L.-H. Sun, "Supporting service differentiation for real-time and best-effort traffic in stateless wireless ad hoc networks (swan)," *Mobile Computing, IEEE Transactions on*, vol. 1, no. 3, pp. 192–207, 2002.

[25] A. Shriram and J. Kaur, "Empirical evaluation of techniques for measuring available bandwidth," in *INFOCOM 2007. 26th IEEE International Conference on Computer Communications*, may 2007, pp. 2162 –2170.

[26] N. Hu and P. Steenkiste, "Evaluation and characterization of available bandwidth probing techniques," *Selected Areas in Communications, IEEE Journal on*, vol. 21, no. 6, pp. 879–894, 2003.

[27] J. Strauss, D. Katabi, and F. Kaashoek, "A measurement study of available bandwidth estimation tools," in *IMC '03: Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement*. New York, NY, USA: ACM, 2003, pp. 39–44.

[28] V. J. Ribeiro, R. H. Riedi, R. G. B. J. Navratil, and L. Cottrell, "pathChirp: Efficient Available Bandwidth Estimation for Network Paths," 2003.

[29] D. T. Nguyen and J. Ostermann, "Streaming and congestion control using scalable video coding based on H.264/AVC," *Journal of Zhejiang University SCIENCE A*, vol. 7(5), pp. 749–754, 2006.

[30] S. ryong Kang and D. Loguinov, "Impact of fec overhead on scalable video streaming," in *NOSSDAV '05: Proceedings of the international workshop on Network and operating systems support for digital audio and video.* New York, NY, USA: ACM, 2005, pp. 123–128.

[31] T. Tsugawa, N. Fujita, T. Hama, H. Shimonishi, and T. Murase, "TCP-AFEC: An adaptive FEC code control for end-to-end bandwidth guarantee," in *Proc. Packet Video 2007*, 2007, pp. 294–301.

[32] H. Wu, M. Claypool, and R. Kinicki, "A model for MPEG with forward error correction and TCP-friendly bandwidth," in *NOSSDAV '03: Proceedings of the 13th international workshop on Network and operating systems support for digital audio and video.* New York, NY, USA: ACM Press, 2003, pp. 122–130.

[33] J.-X. Wang, A. Su, Y.-C. Chen, and J.-N. Hwang, "Effective congestion and error control for scalable video coding extension of the h.264/avc," in *Proc. IEEE International Symposium on Circuits and Systems ISCAS 2008*, 2008, pp. 1834–1837.

[34] K. Stuhlmuller, N. Farber, M. Link, and B. Girod, "Analysis of video transmission over lossy channels," *Selected Areas in Communications, IEEE Journal on*, vol. 18, no. 6, pp. 1012 –1032, june 2000.

[35] I. Cidon, A. Khamisy, and M. Sidi, "Analysis of packet loss processes in high-speed networks," *Information Theory, IEEE Transactions on*, vol. 39, no. 1, pp. 98–108, Jan 1993.

[36] D. S. Hands and K. Cheng, "Subjective responses to constant and variable quality video," in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, ser. Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, vol. 6806, Mar. 2008.

[37] B. Girod, "Psychovisual aspects of image communication," *Signal Processing*, vol. 28, no. 3, pp. 239 – 251, 1992, multidimensional Signal Processing. [Online]. Available: http://www.sciencedirect.com/science/article/B6V18-48XCY63-C4/2/fd30816b4d7da317d94551bbf379ce2d

[38] P. Ssesanga and M. Mandal, "Efficient congestion control for streaming scalable video over unreliable IP networks," in *Image Processing, 2004. ICIP '04. 2004 International Conference on*, vol. 3, 2004, pp. 2059–2062 Vol. 3.

[39] B. Xie and W. Zeng, "Rate-distortion optimized dynamic bitstream switching for scalable video streaming," in *Multimedia and Expo, 2004. ICME '04. 2004 IEEE International Conference on*, vol. 2, 2004, pp. 1327–1330 Vol.2.

[40] P. Zhu, W. Zeng, and C. Li, “Joint design of source rate control and qos-aware congestion control for video streaming over the internet,” *Multimedia, IEEE Transactions on*, vol. 9, no. 2, pp. 366 –376, feb. 2007.

[41] Q. Liu and J.-N. Hwang, “A scalable video transmission system using bandwidth inference in congestion control,” in *Circuits and Systems, 2004. ISCAS '04. Proceedings of the 2004 International Symposium on*, vol. 2, may 2004, pp. II – 17–20 Vol.2.

[42] ——, “End-to-end available bandwidth estimation and time measurement adjustment for multimedia QOS,” in *Multimedia and Expo, 2003. ICME '03. Proceedings. 2003 International Conference on*, vol. 3, 2003, pp. III–373–6 vol.3.

[43] M. Jain and C. Dovrolis, “End-to-end available bandwidth: measurement methodology, dynamics, and relation with TCP throughput,” *Networking, IEEE/ACM Transactions on*, vol. 11, no. 4, pp. 537–549, 2003.

[44] S.-S. Wang and H.-F. Hsiao, “Tcp-friendly congestion control for layered video streaming using end-to-end bandwidth inference,” oct. 2008, pp. 462 –467.

[45] H. Seferoglu, U. Kozat, M. Civanlar, and J. Kempf, “Congestion state-based dynamic fec algorithm for media friendly transport layer,” may 2009, pp. 1 –10.

[46] A. Khalek, C. Caramanis, and R. Heath, “A cross-layer design for perceptual optimization of h.264/svc with unequal error protection,” *Selected Areas in Communications, IEEE Journal on*, vol. 30, no. 7, pp. 1157 –1171, august 2012.

[47] L. Chen and G. Liu, “A delivery system for scalable video streaming using the scalability extension of h.264/avc over diffserv networks,” in *Intelligent Information Hiding and Multimedia Signal Processing, 2008. IIHMSP '08 International Conference on*, aug. 2008, pp. 665 –669.

[48] X. Li, T. Ren, and J. Xu, “A cross-layer design for transmission of scalable h.264 video over ieee 802.11e networks,” in *Computational Problem-Solving (ICCP), 2010 International Conference on*, dec. 2010, pp. 306 –309.

[49] M. Li, Z. Chen, and Y.-P. Tan, “Joint packet prioritization and qos mapping for svc over wlans,” in *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, march 2010, pp. 2298 –2301.

[50] Q. Liu, S. Liu, and C. W. Chen, “A novel prioritized spatial multiplexing for mimo wireless system with application to h.264 svc video,” in *Multimedia and Expo (ICME), 2010 IEEE International Conference on*, july 2010, pp. 968 –973.

[51] T. K. Moseng and O. Kure, "Diffserv in ad hoc networks," in *Proceedings of the Third international EURO-NGI network of excellence conference on Wireless systems and mobility in next generation internet*. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 113–125. [Online]. Available: http://dl.acm.org/citation.cfm?id=1961386.1961399

Part II

Included Papers

Paper A

Admission Control and Flow Termination in Mobile Ad-hoc Networks with Pre-congestion Notification

Bjørnar Libæk and Mariann Hauge and Lars Landmark and Øivind Kure

Military Communications Conference (MILCOM). Orlando, Florida, October 29 - November 1 2012.

Abstract

MANETs (mobile ad-hoc networks) are already starting to get deployed in tactical settings. In such networks, where capacity is limited compared to wired networks, QoS mechanisms are needed in order to serve high priority traffic with acceptable quality under both light and heavy load conditions. A crucial part of a QoS architecture is the admission control mechanism, making decisions on whether new flows may be admitted into the network based on the current traffic situation.

The IETF is currently standardizing an architecture for admission control and flow termination for wired DiffServ domains, named Pre-congestion Notification (PCN). In this paper the possibility of applying the PCN mechanisms to wireless MANETs is studied. MANET specific challenges are discussed, as well as necessary modifications to the PCN mechanisms. Simulation results are presented, identifying the introduction of probing as a key measure in order to reduce the amount of signaling and to base admission decisions on fresh network status information.

1. Introduction

In military tactical MANETs where the network capacity is limited, it is essential that the network load is kept below a critical level. Beyond this level, packet loss and delay cause inelastic flows to have unmet requirements and a utility close to zero. Thus, if still serving these flows, network resources are wasted, and the system operates in a non work-conserving state. To ensure a predictable service with a satisfying quality of experience (QoE) for the users, admission control (AC) is needed in order to limit the number of active flows in the network. The admission control mechanism enforces an upper limit on the network load when the system load (flow arrival rate) is high. Generally, the cost of this is a lower utilization of the network capacity.

Other QoS (quality of service) mechanism such as priority queuing and prioritized access to the shared medium (e.g. 802.11e [1]) can be used to prioritize important traffic over less important traffic. However, these mechanism's abilities to differentiate traffic of different priority levels are close to non-existing when the network is heavily congested. Thus, for such QoS mechanisms to work, admission control is mandatory.

Another important QoS mechanism, closely related to admission control, is flow termination (FT). Unless the admission control is configured extremely conservatively, there will always be a risk of over-admission in the network. This may be caused by events such as re-routing of a batch of admitted flows onto a new path with less available capacity. In such situations, the network should have the ability to enforce termination of some flows in order to improve the QoS for the remaining flows. As described later, over-admission is for several reasons more difficult to avoid in MANETs, so a flow termination mechanism is certainly needed.

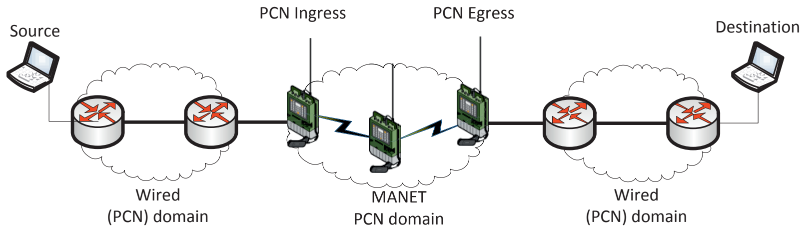


Figure 1. Example network scenario

The pcn working group under IETF's transport area is currently standardizing an architecture for admission control and flow termination targeting fixed DiffServ [2] networks, known as the Pre-congestion Notification (PCN) architecture [3]. Here, AC and FT decisions are taken by the edge[1] nodes, based on signaling from the core nodes. Signaling is done via packet marking using the two last bits in the IP TOS field. By monitoring packet marking on incoming flows, the egress node can infer whether or not the traffic rate on any link on the path exceeds any of the two configurable rates; the *admissible rate* (AR) and the *supportable rate* (SR). As these two bits are also used by ECN [4] (Explicit Congestion Notification) for end-to-end congestion signaling, large efforts have been made in order to allow the use of PCN without losing the ECN capability.

This paper studies the use of PCN mechanisms in MANETs. Even though a number of admission control techniques targeting MANETs have been proposed in the literature [5], there are significant advantages of having a common AC/FT architecture in both the mobile/wireless and the fixed parts of the network. As DiffServ becomes increasingly deployed in both military and civilian fixed networks, and with IETF's efforts to standardizing PCN admission control and flow termination in DiffServ domains, it is likely that the MANETs will be connected to PCN domains in the future as illustrated in figure 1. A common framework for admission control and flow termination simplifies the technical solutions as well as lowers the management costs.

There are however significant challenges that need to be addressed when using the PCN in MANETs. The PCN mechanisms have been designed under the assumption that links are stable, non-shared, high capacity relative to flow size, etc. In addition, the DiffServ concept of dividing the work load between edge and core nodes is to some degree invalid in a MANET, where all nodes may take the role as egress, ingress or core depending on each flow's path.

The main contribution of this paper is to evaluate the use of PCN in MANETs, discussing the effects of the MANET-specific challenges. In order to tackle these challenges, the paper also proposes extensions to PCN as currently described in the various RFCs and internet drafts. Most importantly, a PCN scheme involving probing is proposed. It is simulated and compared to two other PCN schemes

[1] In this paper, *edge nodes* refers to the ingress and egress nodes of the DiffServ/PCN domain

designed for wired networks. Note that the proposed modifications are not visible from outside of the MANET, as the encoding and interpretation of the PCN bits remain unchanged. The signaling between the source and the PCN ingress is out of scope of this paper. As MANETs, and specifically tactical networks, can be constructed using a wide range of link layer and routing protocols, it is important that the introduction of PCN is done with protocol independence in mind. To be able to conduct simulation experiments however, OLSR [6] routing and IEEE 802.11 MAC were used, but the PCN mechanisms do not depend on their specific features.

Section 2 describes the PCN architecture in more detail as well as discusses challenges and related work with respect to the use of PCN in MANETs. In section 3, modifications to some of the PCN mechanisms are described, while section 4 describes the three alternative PCN schemes which are evaluated and compared in the simulation study presented in sections 5 and 6. Finally the paper is concluded in section 7.

2. Background

2.1. Pre-Congestion Notification

PCN is designed to protect inelastic flows in a DiffServ domain, offering both admission control and flow termination functionality. In a PCN domain, every link is associated with two configured rates; the *admissible rate* (AR) and the *supportable rate* (SR). If the rate of *PCN traffic* over a link exceeds the admissible rate for that link, new flows requesting admission over that link should be blocked. Further if the rate of PCN-traffic exceeds the supportable rate, one or several admitted PCN flows using that link should be terminated, forcing the PCN-rate below the SR. By metering the PCN traffic on their outgoing links, the intermediate (core) routers are able to know whether the PCN rate is below or above the two thresholds and signal this *pre-congestion status* to the egress using packet marking in the ECN field (last two bits in the IPv4 TOS/ IPv6 Traffic Class). The egress monitors the incoming packet stream, and sends feedback to the ingress node which makes the admission control and flow termination decisions.

The metering and marking is done using two token buckets for each outgoing link; a *threshold marker* and an *excess marker*, configured with the AR and SR rate respectively. See [7] for the detailed description of these algorithms. In the rest of this paper, it is assumed that the two bits can carry three encoding states: The NM (Not marked) code point, which is the initial value for all packets, the ThM (threshold marked) code point and the ETM (excess traffic marked) code point. If the PCN rate is below AR, no change is done to outgoing PCN packets. If the rate is above the admissible rate, but below the supportable rate, the threshold meter marks all PCN packets with the ThM code point. Finally, if the rate is above supportable rate, the excess marker marks a fraction of the

PCN packets corresponding to the SR-overload with the ETM code point. The latter allows the edge nodes to assess the correct amount of flows to terminate. Consult [8] for an example of how three state encoding can be achieved while still preserving the end-to-end ECN capability.

The PCN standardization is currently focused on ingress-egress-aggregate (IEA) based AC, as opposed to monitoring packet marking on individual flows. When using the IEA based approach, the aggregate of all flows between an ingress-egress pair is monitored, and AC decisions are made based on the total packet marking on that aggregate. Each ingress keeps a state variable K for each egress, taking values from the set $\{block, admit\}$. The ingress sets the value of K based on feedback reports from each egress, containing information about the packet marking on the aggregate. New flow requests will be rejected at the ingress if K equals *block* for that specific destination.

A new DiffServ traffic class is defined in order to carry PCN traffic. The PCN traffic class receives prioritized forwarding through the PCN domain using e.g. the EF (expedited forwarding) PHB (per-hop behavior) in each node. PCN traffic is recognized by a (set of) standardized DSCP (DiffServ code point) value(s).

The QoS signaling needed between the host application and the ingress in order to request admission to the PCN domain is not standardized by PCN. It is however suggested to use the IntServ over DiffServ [9] approach, where RSVP (resource reservation protocol) is used end-to-end, and each PCN domain is an RSVP hop.

2.2. Admission Control in MANETs

This section first presents a number of challenges that arises when applying the PCN mechanism to a MANET. Some of the challenges are valid for admission control in general, while others are PCN specific. After this, related work is discussed.

2.2.1. Challenges

Channel variations In wired networks, the admission limit (e.g. AR in PCN) on a link is set relative to the link's fixed capacity. In MANETs however, the channel capacity as seen from layer 2 is influenced by several time varying factors such as distance, background noise, shadowing etc. These variations are amplified when mobility increases. As discussed in the introduction, a secondary effect of channel variations is the increased likelihood of re-routing events, which in turn may cause over-admission and flow termination. Consequently, if the ambition is to provide a predictable service for the admitted flows with a low probability of being terminated, a conservative admission limit must be chosen in order to account for these variations. Unfortunately, this may cause under-utilization when channel conditions are good.

Shared channel When using PCN in fixed networks, each router has full control of the outgoing link, and there is only one meter per link. In MANETs however, the unidirectional links are replaced by a shared channel, where multiple nodes compete for the same resources. Here, we loosely define a node's channel to being shared with all other neighbors within the interference range surrounding that node. The admissible rate now limits the total PCN-rate sent by all nodes within this area. It then becomes a collective responsibility to ensure that the admissible rate is not exceeded around any node. To achieve this when using PCN, the token bucket metering function in each node must be modified in order to meter as much of the traffic on the channel as possible, not only the traffic sent by the node itself.

Ideally, when making an AC decision in wireless networks, not only the resource status of the nodes on the requesting flow's path should be taken into account, but also the resource status of all neighbors along the path. In the rest of this paper, we assume that such information is not available, as a node only meters the transmissions it actually captures on the air[2]. The consequence of this is that the acceptance of a flow on one path may cause over-admission in the neighbor nodes, which in turn may result in termination of flows going through those nodes.

Large flows relative to capacity In the PCN design, it is assumed that the impact of admitting one flow is insignificant, and decisions are normally based on PCN marking of existing flows. In MANETs, the rate of a flow can be significant compared to the total capacity of the network, so there is a higher probability that the acceptance of one single flow causes over-admission. One possibility could be to compare the flow's flow specification with an estimate of the available bandwidth. This is however outside current thinking in PCN, and it is also difficult to estimate the available bandwidth in a MANET. Another approach which is more in-line with the PCN architecture, is to use probing. The probing session should then try to mimic the new flow, and the metering and marking in the core nodes do not need to differentiate between packets from admitted flows and probe packets.

Empty IEAs Another consequence of the large relative flow size is that the number of admitted flows will be low compared to the number of ingress-egress pairs. This increases the probability of having *empty IEAs*, i.e. ingress-egress pairs without any admitted flows. This is a challenge because the egress gets no information about the congestion status on the path as there are no packet streams to monitor. Unless probing is used on such IEAs, the admission control is forced to admit new flows, with a higher risk of over-admission.

[2]It is possible to extend the metering algorithm with the ability to meter based the link layer's hidden node counter measures such as RTS/CTS in 802.11, but this is not explored further in this paper in order to maintain a link layer independence.

All-to-all signaling A fundamental property of MANETs is that all nodes are equal. In wired DiffServ networks, nodes are configured as either edge or core. In MANETs, all nodes have both roles. A consequence of this is that if there are N nodes, there are N(N-1) ingress-egress pairs, all requiring some signaling. For this reason, it is crucial that the amount of signaling on each IEA is kept as low as possible.

Signaling sent over forward channel As the signaling is also sent over the shared channel, feedback signaling competes for the same resources as the PCN traffic in the forward direction. This has two unwanted consequences: The probability of reaching the ingress with a negative feedback message (i.e. telling the ingress that there is congestion on the path) is reduced, which in turn may cause the ingress to accept new flows on congested paths. To avoid this, extra robustness on feedback signaling may be required in MANETS. The other consequence is that the signaling steals capacity from the data traffic, and may in fact increase when marked packets are observed (e.g. when using report suppression as described in [10]). This self-induced congestion effect has a negative impact on the QoS of the admitted flows, and is only avoided by taking into account the amount of signaling when setting the AR and SR limits.

Security issues in tactical networks Security issues arise when signaling is required to traverse security domain borders. This is neither MANET nor PCN specific, but a general challenge when introducing admission control on tactical networks. When the source in a high level security domain needs to communicate with a network node at a lower security level, this needs to be explicitly handled at the security border. In PCN, this is the case for the signaling between the source and the egress (e.g. using RSVP). The way this is handled is not discussed further in this paper.

As only edge-to-edge PCN is assumed in this paper, manipulation and interpretation of the two PCN bits are only done within the PCN domain. Thus, there is no issue with these bits being used as a covert channel through the security border, as opposed to end-to-end ECN.

2.2.2. Related work

At the time of writing, all PCN standardization work as well as all PCN related publications focus on wired networks. Also, there are no publications on the use of PCN in neither mobile nor fixed infrastructure wireless environments. The topic of admission control in MANETs has however gained significant attention in the research community, with numerous publications. An extensive survey is presented in [5]. It is important to emphasize that the main incentive for applying PCN to MANETs is not to achieve better performance than existing admission control schemes specifically designed for MANETs, but to exploit the fact that PCN is becoming a standardized architecture which is likely to be implemented in wired networks connected to the MANET. Nevertheless, comparing the PCN approach with other MANET admission control solutions is absolutely an important task, but it is outside the scope of this paper. Also note that admission control of best effort traffic is also required in a full QoS solution, but how this is done is not discussed further in this paper.

Another related topic is the use of ECN in wireless networks, in order for TCP to distinguish packet loss caused by congestion from loss due to background noise. This topic is however not directly related to admission control.

3. MANET extensions to PCN

In order to adjust the PCN mechanisms to the challenging MANET environment, some fundamental changes need to be made.

3.1. Metering & Marking on shared channel

As explained in section 2.2.1, PCN metering needs to be modified to work on a shared channel. In addition to metering outgoing packets as done with fixed links, the nodes must now also meter PCN-packets sent on the channel by their neighbor nodes. This requires promiscuous mode operation by all nodes. Note that hidden nodes are not taken into account with this approach, as discussed in section 2.2.1.

3.2. AC decision with blocked neighbors

When using IEA based AC, the ingress knows the status on the path to each egress. For different reasons (e.g the empty IEA issue), the feedback from some egress nodes may be missing/outdated, resulting in an ingress unaware of congestion on the path to those egress nodes. Ideally, if having recently been notified about congestion on the path to one egress, the ingress should then also block new request to other egress nodes which paths also traverse the congested link. Even when having access to full topology information (link state routing) however, the incoming PCN-packet stream at the egress carries no information about the location of the congested link(s), so such assumptions cannot be made. In MANETs, there is one exception; If at least one one-hop neighbor of an ingress is reporting congestion, that ingress should block all admission requests regardless of destination.

4. Alternative schemes

In the rest of this paper, three different PCN schemes for MANETs are compared. Each of the three schemes includes both an AC method and an FT method. All three AC methods are ingress-egress-aggregate (IEA) based, and all use the

MANET extension described in 3. The first one, OBAC+MPTIF, minimizes the amount of signaling by only sending feedback when observing marked packets. The second one, original CL, uses periodic signaling between all ingress-egress pairs in order to increase the robustness of the signaling. Finally, the proposed scheme modifies the original CL to reduce the amount of signaling on empty IEAs by introducing probing.

4.1. OBAC and MPT-IF

This scheme is based on the combination of Observation Based Admission Control (OBAC) and Marked Packet based Termination for Individual Flows (MPT-IF), both described in [11]. When using OBAC, when the egress observes an ThM-marked packet, it sets K to *block*. Only when no marked packets have been observed for the last D^{min}_{block} seconds, the state variable is reset to *admit*. The way the egress signals the value of K back to the ingress is not described in [11]. In the remainder of this paper, it is assumed that OBAC only sends feedback when K is toggled. The advantage of this is a low signaling overhead, but the poor robustness increases the probability of over-admission due to missing negative[3] feedback messages. When using OBAC, feedback is only sent if there is traffic on the IEA, and is triggered by marked packets. An unfortunate consequence of this is that in a system with empty IEAs, no information about the state of the path is present in the ingress at the decision time, leaving the ingress with no choice but to accept new flows. This is another reason for over-admission when using OBAC.

With MPT-IF, the egress monitors incoming flows individually (i.e. not IEA based), and keeps a counter for each flow. The counter is decremented for each incoming ET-marked packet for that flow, and if the value of the counter is less than zero, the egress sends a feedback message to the ingress telling it to terminate the flow. The initial value of the counter controls the responsiveness to congestion. This signaling is more robust than for OBAC, since a feedback message is triggered for every ET-marked packet arriving after the counter reached zero.

4.2. CL

The IETF is close to standardizing the Controlled Load (CL) mode of operation for PCN [10], which describes both admission control and flow termination mechanisms as well as the signaling between the egress and the decision point. In CL, the egress monitors each incoming IEA, and once each measurement interval calculates the NM-rate, ThM-rate and ETM-rate as the rate of NM-marked, ThM-marked and ETM-marked packets respectively. Based on these three rates, it also calculates the CLE (Congestion Level Estimate) as the ratio of marked to

[3] A negative feedback message forces the ingress to block new flows

unmarked PCN traffic on the IEA. At the end of the measurement interval, a feedback report is sent to the ingress node (assuming that the decision point is collocated with the ingress), containing the three measured rates, the CLE and an optional list of all ETM-marked flows. If CLE is less than the configurable CLE-limit, the admission state K is set to *admit*, otherwise it is set to *block*. In addition K is also set to *block* if the ingress detects missing feedback reports, using a timer. Flow termination is also done by the ingress based on the same report, if the report contained a non-zero value for the ETM-rate. The ingress then finds the amount of PCN traffic to terminate to be the difference between the pcn-sent-rate and the sustainable-aggregate-rate for this IEA. The pcn-sent-rate is the offered rate measured by the ingress, and the sustainable-aggregate-rate is the sum of the NM-rate and the ThM-rate (i.e. all non ETM-marked traffic on the IEA). Finally, the ingress selects a set of flows to terminate so that the sum of their rates satisfies the calculated termination rate. Preferably, this set of flows is selected from the list of ETM-marked flows if included in the report from the egress. Policies could also be applied in order to first terminate low priority flows.

CL also includes an optional report suppression scheme which is used to limit the overhead related to the signaling between the egress and the ingress. When using report suppression, the feedback interval is effectively increased when the congestion level is low. Refer to [10] for the details of the algorithm.

Due to the periodic feedback, CL prevents over-admission due to missing feedback reports as described for OBAC. However, the cost for this is significant amounts of signaling overhead because of the all-to-all signaling issue described in 2.2.1. A negative effect of using report suppression is that the amount of signaling increases with the congestion level, as more packets get marked. CL does also suffer from the empty IEA problem. An empty IEA will continuously report zero marking rates, so new flows will be accepted even when there may be congestion on the path.

4.3. CL with probing

To reduce the amount of signaling overhead introduced by CL, and to avoid the empty IEA problem, the possibility to combine CL with probing is investigated. The idea is to deactivate periodic feedback when no traffic has been sent on an IEA for a certain amount of time. The set of values that can be assigned to K is now extended to $\{block, admit, invalid\}$, and if the ingress has not received any feedback from the egress for a configurable *invalid period*, the state is set to *invalid*. If admission requests are received by the ingress in this state, explicit probing is performed before responding to the request. The probe is sent in the PCN class, being subject to marking as other PCN traffic. When arriving at the egress, a probe response message is returned, containing a flag indicating whether the probe was marked or not. The probe response is also subject to marking on the return path to the ingress. Only if the ingress receives an unmarked

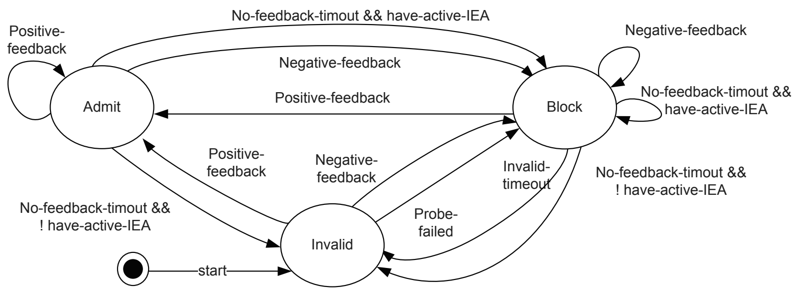


Figure 2. CL-probe Ingress FSM: Events: *Positive feedback*: Received feedback report with CLE $\leq$ CLE-limit. *Negative feedback*: Received feedback report with CLE $>$ CLE-limit. *No-feedback-timeout*: No feedback received within timeout period. *Invalid-timeout*: No feedback within invalid period. *Probe-failed*: Probe response either marked, contained non-zero flag or not received. *have-active-IEA*: (Condition) The IEA is non-empty

probe response within a time limit, with the flag set to zero, will the new flow be admitted. In this paper, it is assumed that the probing only consist of one single probe packet. This works with threshold marking, as all packets are marked if the admissible rate is exceeded. However, in order to increase the probability of reaching the egress over lossy links, or in order to prevent the large flow issues described in section 2.2.1, multiple probe packets could be used. Figure 2 shows the state machine of the ingress node when using CL with probing. Flow termination is done in the same way as for the original CL.

When using probing in a system where the source is ready to send data as soon as the AC decision is available, the delay introduced by the probing causes a lower utilization. However, if end-to-end signaling is done using RSVP as described in section 2.1, the internal probing delay in the MANET is unlikely to add any extra delay to the total setup time. As revealed by the simulations, the effect of the probing delay is however ignorable.

CL with probing has significantly less amounts of signaling, as periodic feedback is only sent on active IEAs.

5. Simulator overview

The simulator model is based on the Inetmanet [12] extension to the Omnet++ [13] network simulation framework (version 4.1). The various modules used are described in the following section.

class	AIFSN	Init CWmin	Init CWmax	TXOP
AC 2	4	127	255	3.008 ms
AC 3	1	63	127	1.504 ms

Table 1. IEEE 802.11*e* parameters. (AC0 and AC1 is not used)

5.1. Protocol stack

5.1.1. Physical and link layer

The 802.11 implementation in Inetmanet was used, configured as 802.11ge running at 11 Mbps. Routing and PCN signaling were sent as AC3 (highest priority) while PCN data packets were sent as AC2. Table 1 summarizes the 802.11e QoS parameters. RTS/CTS threshold was set to 500B. PCN metering and marking functionality was added to the link layer.

5.1.2. Network layer (MANET routing)

At the network layer, OLSR was used. The Inetmanet implementation of OLSR is a port of the UM_OLSR [14] which is also used in NS2. No changes to the default values of OLSR parameters were made, except from enabling link layer notifications (LLN) as described in [6]. The reason for presenting only simulations with LLN enabled is that LLN significantly improved performance when admission control was disabled. Thus, the relative gain of having admission control as shown in the presented results, will increase even further if turning off LLN. In addition to the MANET routing, the network layer also contains the IP module, which does the actual forwarding of IP packets based on the routing table produced by OLSR.

5.1.3. Transport layer

All PCN traffic in this study use UDP (User Datagram Protocol) transport, as the PCN class is only intended for flows that are not adapting their sending rate to network conditions such as TCP (Transmission Control Protocol).

5.1.4. PCN agent

The edge (ingress and egress) PCN functionality (both AC and FT) was included in the *PCN agent*, located at the transport layer. The PCN agent is configured to use either one of the four alternative schemes (No PCN, OBAC+MPTIF, CL or CL with probing), and also takes care of the signaling with the traffic source generator. An implementation specific ingress feature was added in order to not accept any admission requests to destination (egress) if the ingress had no valid route to the egress. This feature was also used when PCN was turned off. For OBAC, D_{block}^{min} was set to 10 seconds, and for MPTIF the initial value of the egress counter (see section 4.1) was set to 10 packets. For CL, the report interval was set to 2 seconds, and increased to 10 seconds during report suppression. The CLE-limit was set to zero.

5.1.5. Application layer (traffic generators)

The foreground traffic generator, generating PCN flows, is based on Inetmanet's UDPBasicBurst2 which implements an ON/OFF source emitting a 64kbps CBR (constant bit-rate) packet stream in ON-state (800B packet each 0.1 seconds). The module was modified to request admission from the AC module before going in ON-state. If the admission request is rejected, it goes back to OFF-state. The time in both states was randomly drawn from a negative exponential distribution with 40s mean. No background traffic is simulated in this study.

5.2. Simulation scenario

40 nodes with a 500m transmission range (equal to sensing range) were placed randomly in an area of 1700x1700 meters. This allows for 5 radio hops along the diagonal. The interference range is approximately 3 times the distance between the receiving and transmitting nodes. S sources were randomly (with replacement) distributed over the 40 nodes, where $S \epsilon \{15, 30, 45, 60, 90, 150\}$ was used to control the system load. The nodes moved according to the Random Way-point mobility model, with a speed drawn from the uniform(2 m/s,5 m/s) distribution, and a wait time of 5s. Refer to Inetmanet documentation / source code for further details.

The simulation length was 2000s, and statistics collection started at 100s to avoid transient effects in the start of the run. The routing protocol is started at 0s, while traffic generators were started at 50s in order to allow the routing to stabilize.

5.3. Configuration of PCN metering and marking

The AR and SR rates were set to 0.7 Mbps and 1.5 Mbps respectively. As pointed out earlier, a conservative admission limit is necessary in MANETs for several reasons. The values were chosen based on observing the packet loss and delay in simulations where PCN was turned off. The relatively large difference between the two values was chosen in order to account for over-admission caused by re-routing. A threshold meter and an excess meter were used, configured with the admissible rate and supportable rate respectively. The bucket size of the threshold meter was set to 520MB, while the bucket threshold was set at 500MB. The small difference between these two parameters means that marking starts early when the PCN rate exceeds the admissible rate. As an example, let R_f denote the flow rate (64kbps); If the PCN rate equals $AR + 1/2R_f$, the PCN rate becomes 732kbps, it takes approximately 600ms before ThM marking starts. The size of the threshold alone determines the persistence of the marking when the PCN rate decreases below the AR. When using 500MB, and the rate is equal to $AR - 1/2R_f$, it takes approximately 15 seconds before marking stops. Thus, the threshold meter is configured rather conservatively. The bucket size of the excess meter was set to 300MB. If the PCN rate exceeds the SR rate with $1/2R_f$, it takes about 9 seconds before ETM marking starts. A large value was chosen here in order to avoid too much termination, at the cost of a longer period with reduced QoS for admitted flows.

Note that the objective of this simulation study is to compare the different schemes' ability to control the network load with the given set of parameters. Sensitivity analysis of these parameters is outside the scope of this paper.

6. Performance evaluation

6.1. Evaluation criteria

This section describes the different simulation statistics, which are used when comparing and evaluating the different schemes. Note that statistics directly showing the amount of under- or over-admission is not included due the complexity caused by the dynamic environment. Over admission is however indirectly reflected by the quality experienced by the admitted flows (e.g. packet loss and termination probability).

6.1.1. L7 throughput

The mean delivery rate of PCN packets measured at the application layer in the destination nodes, where a PCN packet is a data packet belonging to an admitted flow. The L7 throughput gives an indication of the utilization of the network, but does not reflect the QoS experienced by individual flows.

6.1.2. Packet Loss Ratio (PLR)

Measured at layer 7, the PLR is the total number of received PCN packets at the destination nodes, divided by the total number of sent PCN packets. Note that this is the total packet loss ratio as seen at the application layer, and does not reflect how the packet was lost (tail-drop, no-route, max-retry etc). The PLR gives a good indication of the QoS of admitted flows.

6.1.3. Flow Admission Ratio (FAR)

The total number of admitted flows divided by the total number of admission requests.

6.1.4. Flow Termination Ratio (FTR)

The total number of flows terminated by the FT mechanism, divided by the total number of admitted flows. A high termination ratio is a clear sign of over-admission.

6.1.5. L7 Goodput

The L7 goodput is defined as the number of *useful* information bits delivered to the application per second. For inelastic multimedia flows, which can only tolerate a certain amount of packet loss and delay, the received information may be useless if they experience packet loss and delay above the tolerance thresholds. As opposed to the L7 throughput, which includes all information received, the L7 goodput only includes the amount of data received by successful flows. In the following, a flow is considered successful if it experiences a packet loss ratio less than 10%, 90% of the flow's data packets have a delay less than 200ms and it is not terminated by the FT mechanism. Note that this success criteria is application specific.

6.1.6. Mean hop-count

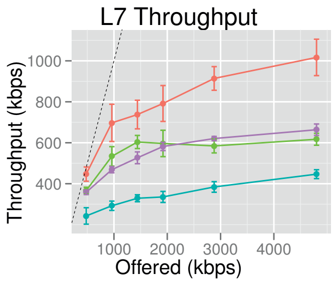
The mean number of hops traveled by successfully received PCN packets. PCN packets dropped somewhere along the route is not included in this statistic. When comparing the different schemes, the hop-count indicates their willingness to accept flows over longer paths, thus it may be seen as a measure of fairness.

6.2. Simulation results

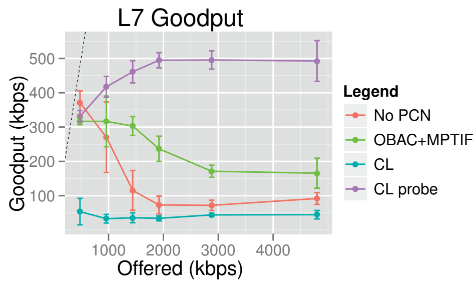
In this section, simulation results of the three PCN schemes are compared. In addition, the scenario was simulated with both AC and FT turned off. Consequently, the plots contain four graphs; one for each of the three PCN schemes, and one for the case where AC and FT were turned off (labeled "No PCN"). The x-axis in the plots represents the system load, which is comparable for all schemes. All simulated points are surrounded by the 95% confidence interval, calculated using independent replications, where each point is simulated a number of times with different random number seeds.

6.2.1. No PCN

As expected, the highest L7 throughput (figure 3a) is achieved when PCN is disabled. Also, as seen in figure 4a, packet loss reach extreme levels when the offered load increases. The delay statistics is not included in this paper, but simulations also showed an extreme growth of the end-to-end delay when PCN was turned off. Another interesting observation is that the admission ratio (figure 5a) decreases significantly as the load increases. Recall that the only reason for

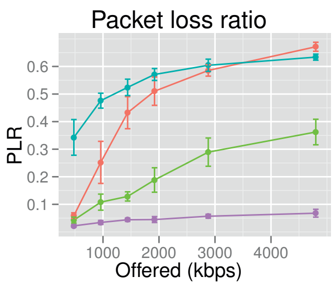


(a) L7 throughput

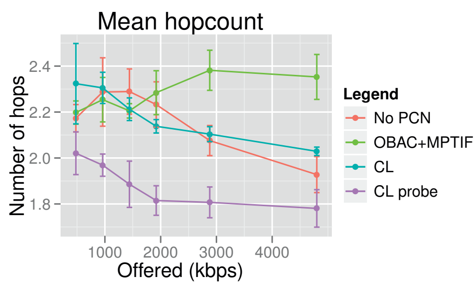


(b) L7 goodput

Figure 3



(a) Packet loss ratio (L7)



(b) Mean hop-count

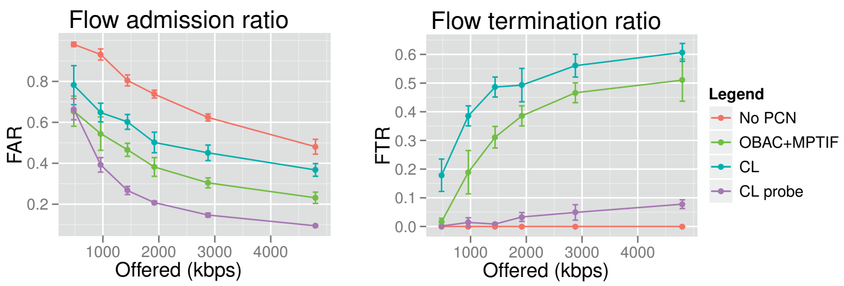
Figure 4

rejecting flows in this case is missing routes. This means that the increased traffic causes the routing protocol to fail, even though routing packets are prioritized at the link layer. In fact, the major reason for packet loss is *missing route to destination*, which motivates the use of admission control.

The L7 goodput is shown in figure 3b, and clearly illustrates the importance of admission control. Even though the L7 throughput increases with the system load, the amount of useful information delivered to the application rapidly decreases when not using PCN. Consequently, the network operates in a non work-conserving state, as significant resources are wasted when serving useless traffic.

6.2.2. OBAC+MPTIF

OBAC+MPTIF has a L7 throughput near the configured admissible rate. However, looking at the packet loss and termination ratio for this scheme reveals that there are significant amounts of over-admission. This is also reflected in the L7 goodput which is much lower than the throughput. It does however show



(a) Flow admission ratio

(b) Flow termination ratio

Figure 5

an improvement compared to the No PCN case. As described in section 4.1, OBAC suffers from over-admission due to the empty IEA issue as well as missing negative feedback reports, and this is confirmed by the simulation results.

6.2.3. Original CL

As expected, the high amount of signaling overhead introduced by CL is very harmful for the network. CL only achieves about 2/3 of the L7 throughput compared to OBAC, and has almost twice as much packet loss. Original CL has more packet loss than the no PCN case, and the L7 goodput is in fact lower. Hence, the system performs better without PCN compared to using original CL.

The positive effect gained by the added signaling robustness compared to OBAC is hidden by the negative effect of having high amounts of signaling stealing resources from the PCN traffic. It is not shown in any of the graphs, but the amount of PCN signaling in the original CL case is approximately equal to the amount of PCN traffic.

As original CL also suffers from the empty IEA problem, over-admission and flow termination is also the case here, as shown in figure 5b.

6.2.4. CL with probing

As expected, CL with probing behaves quite differently compared to the other two alternatives. Its performance is not perfect, but it outperforms the alternative schemes on nearly all points. A good quality is provided to most of the flows, as the packet loss and delay (not shown) is near constant and significantly lower than both OBAC+MPTIF and original CL. Together with the low termination ratio, this shows that there is little over-admission in the network, mainly because the empty IEA problem is eliminated when using probing. This is also reflected in figure 5a, showing that CL with probing is the most restrictive solution in terms of admission ratio. Consequently, a high L7 goodput is achieved compared to the other schemes. Some over-admission is still occurring however, which is most likely explained by the hidden node issues described in section 2.2.1.

Another observation is that CL with probing has the lowest mean hop-count. This indicates as expected that probing with only one single packet decreases the possibility of reaching destinations far away, as the packet loss probability increases with the number of hops. Increasing the number of probe packets may compensate for this effect as described in 4.3, but care should be taken to ensure that the probe packets influence as little as possible the QoS of the admitted flows.

6.3. Simulation summary

The simulation results confirm that the challenges described in 2.2.1 must be taken seriously. The empty IEA problem seems to be dominating, as both OBAC and original CL admits too much traffic which in turn results in significant amounts of flow termination. It is however unclear to which extent this is caused by the hidden node effect as described in section 2.2.1. Another important conclusion confirmed by the simulations is that CL with periodic signaling on all IEAs as described in [10] is not directly applicable in a MANET, even with significantly larger intervals between updates. The combination of low capacity with all-to-all signaling over a shared channel causes a significant resource reduction for the PCN traffic.

The simulations also revealed that the introduction of probing to eliminate the empty IEA problem was successful, as the CL with probing scheme has little over-admission and a high L7 goodput. A reduction in the performance due to delay introduced by waiting for the probe response is not visible in these simulations, as the other effects are dominating.

7. Conclusions

MANET-specific challenges related to admission control in general and PCN specifically has been discussed. A simulation model has been developed, and three different PCN schemes have been simulated and evaluated in a simulation study.

The study has showed that the PCN architecture can be applied to MANETs, without dependencies to a specific link layer or MANET routing protocol. The metering and marking algorithms as described in the current PCN internet drafts needs to be modified in order to be used on a shared channel. It is also critical that the amount of signaling between egress and ingress is drastically reduced compared to the CL specification. It is suggested that probing is used to accomplish this, as well as reducing the risk over-admission on empty IEAs. This is confirmed by simulations, which show that the proposed CL with probing scheme outperforms its alternatives.

In addition to sensitivity analysis on the respective PCN parameters, more work should be done on improving the metering algorithm of PCN traffic on a shared channel with hidden nodes, so that the AC decision is based on the pre-congestion status of all affected nodes, not only the nodes along the path of the new flow. Also, probing with more than one probe packet to increase the probability of accepting flows on longer paths should be considered.

Finally, a complete QoS architecture for MANETs must also include ways to control the load of best effort traffic. A topic for further research is to study the possibility to use the PCN mechanisms to accomplish this as well.

8. References

[1] "Ieee standard for information technology - telecommunications and information exchange between systems - local and metropolitan area networks - specific requirements - part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications," *IEEE Std 802.11-2007 (Revision of IEEE Std 802.11-1999)*, pp. 1 –1076, 12 2007.

[2] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An Architecture for Differentiated Service," RFC 2475 (Informational), Internet Engineering Task Force, Dec. 1998, updated by RFC 3260. [Online]. Available: http://www.ietf.org/rfc/rfc2475.txt

[3] P. Eardley, "Pre-Congestion Notification (PCN) Architecture," RFC 5559 (Informational), Internet Engineering Task Force, Jun. 2009. [Online]. Available: http://www.ietf.org/rfc/rfc5559.txt

[4] K. Ramakrishnan, S. Floyd, and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP," RFC 3168 (Proposed Standard), Internet Engineering Task Force, Sep. 2001, updated by RFCs 4301, 6040. [Online]. Available: http://www.ietf.org/rfc/rfc3168.txt

[5] L. Hanzo II and R. Tafazolli, "Admission control schemes for 802.11-based multi-hop mobile ad hoc networks: a survey," *Communications Surveys & Tutorials, IEEE*, vol. 11, no. 4, pp. 78 –108, October 2009.

[6] T. Clausen and P. Jacquet, "Optimized Link State Routing Protocol (OLSR)," RFC 3626 (Experimental), Internet Engineering Task Force, Oct. 2003. [Online]. Available: http://www.ietf.org/rfc/rfc3626.txt

[7] P. Eardley, "Metering and Marking Behaviour of PCN-Nodes," RFC 5670 (Proposed Standard), Internet Engineering Task Force, Nov. 2009. [Online]. Available: http://www.ietf.org/rfc/rfc5670.txt

[8] T. Moncaster, B. Briscoe, and M. Menth, "A PCN encoding using 2 DSCPs to provide 3 or more states," Working Draft, IETF Secretariat, Internet-Draft draft-ietf-pcn-3-state-encoding-02.txt, March 2012.

[9] Y. Bernet, P. Ford, R. Yavatkar, F. Baker, L. Zhang, M. Speer, R. Braden, B. Davie, J. Wroclawski, and E. Felstaine, "A Framework for Integrated Services Operation over Diffserv Networks," RFC 2998 (Informational), Internet Engineering Task Force, Nov. 2000. [Online]. Available: http://www.ietf.org/rfc/rfc2998.txt

[10] A. Charny, F. Huang, G. Karagiannis, M. Menth, and T. Taylor, "PCN Boundary Node Behaviour for the Controlled Load (CL) Mode of Operation," Working Draft, IETF Secretariat, Internet-Draft draft-ietf-pcn-cl-edge-behaviour-12.txt, Feb. 2012.

[11] M. Menth, F. Lehrieder, B. Briscoe, P. Eardley, T. Moncaster, J. Babiarz, A. Charny, X. Zhang, T. Taylor, K.-H. Chan, D. Satoh, R. Geib, and G. Karagiannis, "A survey of pcn-based admission control and flow termination," *Communications Surveys & Tutorials, IEEE*, vol. 12, no. 3, pp. 357 –375, July 2010.

[12] The INETMANET framework for OMNeT++ 4.x. [Online]. Available: https://github.com/inetmanet/inetmanet/wiki

[13] The OMNeT++ simulation framework. [Online]. Available: http://www.omnetpp.org/

[14] UM-OLSR. An implementation of the OLSR protocol for the NS2 network simulator. [Online]. Available: {http://masimum.inf.um.es/fjrm/?page_id=116}

Paper B

Congestion Control for Scalable VBR Video with Packet Pair Assistance

Bjørnar Libæk and Øivind Kure

Proceedings of 17th International Conference on Computer Communications and Networks (ICCCN), St. Thomas, US Virgin Islands, August 3-7 2008.

Abstract

The SVC (Scalable Video Coding) extension to the recent video compression standard H.264/MPEG-4 AVC paves the way for video congestion control, by allowing flexible on-the-fly adjustments of the sending rate. We study the interaction between TFRC (TCP-Friendly Rate Control) and application level rate control, and observe that TFRC requires relatively large router buffers to keep packet loss on acceptable levels. In turn, this leads to increased jitter which excludes delay sensitive applications. As a response, the Packet Pair Assisted Congestion Control is presented and compared with TFRC.

The two transport protocols are simulated in a DiffServ environment, and we find that proper congestion control may replace complex admission control mechanisms without sacrificing utilization or QoS guarantees. In fact, the results indicate that semi-elastic video can be sent in an AF (Assured Forwarding) class with small buffers, leaving the EF (Expedited Forwarding) class to traffic with more extreme delay requirements.

1. Introduction

With video conferencing replacing travel and emerging applications like HD IPTV, it is likely that video streaming will represent a sizable traffic source in the years to come. Indeed, the wired network capacities are increasing, but situations where aggregate variable rate video streams exceed the available capacity will continue to occur. Prime examples are wireless access networks and 4G mobile networks. It is therefore still of value to propose and analyze new congestion control schemes targeting VBR (Variable Bitrate) video.

In bandwidth limited environments, QoS (Quality of Service) support from the network is necessary to provide service guaranties to multimedia flows. In smaller networks, per-flow reservations with strict guarantees may be possible, but in the general case, DiffServ's [1] flow aggregation approach is more likely to provide a good balance between expenditures, scalability and strength of guarantees. Because SLAs (Service Level Agreement) are most often negotiated on the aggregate level, several multimedia flows have to share the reserved resources, and the need for congestion control is maintained.

DiffServ provides strict delay and jitter control through the EF (Expedited Forwarding) [2] class, but EF traffic will always be limited to a smaller fraction of the available bandwidth and is best suited for low volume real-time traffic such as VoIP (Voice over IP). Video however, is often associated with the Assured Forwarding class (AF) [3]. AF offers a rate guarantee, but when the load is high, large buffers may lead to excessive delay-jitter for variable rate multimedia traffic.

With new advances in scalable video coding such as the SVC (Scalable Video Coding) extension to H.264/AVC [4] and its ability to adapt the video stream to the available bandwidth, video congestion control is getting one large step closer towards realization. However, there are still open questions concerning how these capabilities can be incorporated into a congestion control protocol. A congestion control scheme for layered VBR video must include:

- *Transport level rate control*: The control method used at the transport level to estimate the allowed sending rate.
- *Application level rate control*: A mechanism to adapt the enhancement layers to the transport level estimate.

To facilitate interchangeability between transport protocols and video applications, it is advisable that transport and application level rate control are handled independently. This is also reflected in the literature, where contributions normally target only one of these issues. Prime examples are IETF's proposed standard TFRC (TCP Friendly Rate Control) [5] and increase-decrease algorithms [6–8], where the focus is on optimizing properties such as fairness, utilization and network stability. As will be shown in this paper, TFRC is sensitive with respect to router buffer size when sending VBR video, clearly exposing the jitter/loss trade-off. On the other hand, application centric approaches focus on how to make use of the allowed sending rate in order to optimize user perceived quality. Examples are [9] introducing the concept of optimum quality adaptation trajectories, and [10] where a real-time adaption algorithm is designed to minimize adaptation variability while maximizing network utilization.

There are also examples of hybrid solutions, where network and application considerations are handled in the same operation. In [11], rate control decisions and adding and dropping of layers are based on the filling of the play-out buffer, but unfortunately this technique only works with CBR (Constant Bitrate) video. Another interesting attempt to combine network and application goals where made in [12], where the authors proposed to use *packet pair probing* as a tool to estimate the available bandwidth for a video flow, and select enhancement layers based on these estimates. However, the packet pair estimator used in this paper was biased, not conforming to the Probe Gap Model (PGM) model.

In this paper, the *Packet Pair Assisted Congestion Control* (PPACC) is presented and compared with TFRC. PPACC is inspired by the work in [12], but packet pair congestion control is brought one step further by improved handling of estimation bias and variance.

We have added SVC application modules to the NS2 network simulator, and evaluate the two transport protocols using real video material. This way, the interaction between application and transport level rate control can be studied, all though the main contribution is a transport level mechanism.

The protocols are simulated in a DiffServ network, demonstrating the possibility to use AF for semi-elastic video traffic by combining small router buffers and PPACC. In addition, by using the drop precedence levels in an AF class, important packets within the video stream (e.g the *base layer*) can be protected by having lower drop probabilities in the routers.

The next section briefly explains the relevant technologies, before PPACC is explained in section 3. Section 4 describes the network simulator, and section 5 reports simulation results. Finally the paper is concluded in section 6.

2. Background

2.1. SVC extension to H.264/AVC

The H.264/MPEG-4 AVC standard was developed by JVT (Joint Video Team), a partnership between ITU-T's VCEG (Video Coding Experts Group) and ISO/IEC's MPEG (Moving Pictures Experts Group). The standard was approved in 2005, and the main goal was to create a video compression standard capable of delivering good quality at significantly lower bit rates than existing compression techniques, targeting a wide range of applications as well as network technologies. An overview of the H.264/AVC standard is given in [13].

In November 2007, the amendment for Scalable Video Coding (SVC) was added to the standard. This extension provides the ability to augment the H.264/AVC compatible base layer with enhancement layers in the spatial, the temporal and the quality domain. Scalability is simply done by discarding packets from the NAL (Network Abstraction Layer) unit stream produced by the encoder. See [14] for a short overview of SVC. Figure 1 illustrates how NAL units in a GOP (Group Of Pictures) can be organized to provide three dimensional scalability (this is the configuration used in the simulations presented in section 5).

2.2. TCP-Friendly Rate Control

TFRC is an equation based congestion control protocol for unicast multimedia flows, and is currently being standardized by the IETF (Internet Engineering Task Force) as one of the congestion control *profiles* in DCCP (Datagram Congestion Control Protocol) [15]. It provides a control mechanism with smoother rate changes than TCP's (Transmission Control Protocol) AIMD (Additive Increase Multiplicative Increase) scheme, while at the same time attempting to consume the same bandwidth resources as a TCP flow would do under equal conditions. It uses a slightly modified version of the TCP throughput equation derived by Padhye in [16]. The equation calculates the expected throughput of a TCP flow given the estimated RTT (Round Trip Time) and *loss event rate*, and the flow is shaped according to the expected value. The sending rate will therefore only be reduced when the RTT or the packet loss increases. Also, for several reasons described in [17], TFRC may have significant differences in throughput compared to TCP.

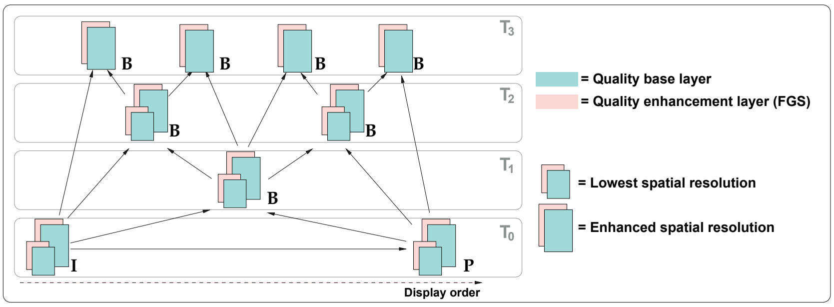


Figure 1. SVC Enhancement layer configuration. Two spatial, four temporal and two quality enhancement layers. Arrows denote dependencies. The spatial base layer has QCIF resolution (176x144) while the spatial enhancement layer has CIF (352x288). The lowest temporal resolution is 3 fps, which is doubled in each of the temporal enhancement layers, thus the highest temporal resolution is 24fps. The quality enhancement layer is FGS (Fine Granular Scalability), which means that a packet in this layer can be truncated at an arbitrary position (in the current version of the standard, FGS has been replaced with Medium Grained Scalability (MGS) where packet truncation is removed, thus reducing the number of available rate points compared to FGS).

2.3. Packet pair

Using the packet pair technique to estimate bottleneck capacity was first suggested by Van Jacobson [18] and later thoroughly studied by Paxson [19]. The basic idea is that if two packets are sent back-to-back from the sender, they will arrive at the receiver with a time spacing $\delta' = s/C_b$ where s is the packet size and C_b is the bottleneck capacity[1]. This holds as long there is no cross traffic intervention on any of the two packets. When cross traffic is introduced, such interventions lead to estimation errors, but as Paxson describes in his thesis, C_b can be accurately estimated by locating the largest mode in the distribution of the arrival interval.

In congestion control however, we are interested in estimating the *available bandwidth*, i.e. the available capacity on the bottleneck link after subtracting the bandwidth used by cross traffic flows. This topic has been well studied, but unfortunately, estimation bias and especially variance has proved to be major challenges. For these reasons, congestion control cannot rely on packet pair estimation only, but it is our belief that packet pairs can provide useful information to the congestion control mechanism.

When sending VBR video, transmission of packets back-to-back happens frequently during a session, because video frames are often fragmented into several IP packets. This holds particularly well for key pictures which typically

[1]The capacity of the slowest link on the path from the sender to the receiver.

occur at least once in each GOP. As will be described in section 3, PPACC takes advantage of this property.

2.4. DiffServ

To maintain video quality in bandwidth limited networks, QoS support is necessary. The DiffServ architecture provides statistical guaranties with respect to bandwidth, packet loss and delay in a scalable manner. In the following paragraphs, DiffServ's key concepts and their relations to video congestion control are outlined.

Service Level Agreements

Traffic is divided into classes, and all packets belonging to the same class are subject to equal treatment from the network. This *service level* is specified in Service Level Agreements (SLA) between the customer and the provider, and resources can be reserved either dynamically (on demand) or statically. Dynamic reservations require centralized per-flow admission control and substantial amounts of signaling, which scales poorly with the number of active flows. More likely, SLAs are static and negotiated on the aggregate level, not per-flow. An example is a campus network with a DiffServ enabled connection to the national research backbone. The ingress router in the research network only monitors the aggregate traffic, while the egress router in the campus network handles admission control for requesting flows.

Per-flow reservations

DiffServ's strength results from aggregating flows in the core network. However, per-flow reservations may still be handled in the access network while keeping the amounts of router state at acceptable levels. This requires flow granularity both on admission control and on output queuing in the ingress router, but more complex schemes like *IntServ over DiffServ* [20] are also possible. Knowing the reserved resources for a flow is beneficial for the congestion control entity, which could base the increase and decrease decisions on the reserved rate. One example is [21], where the authors modify TFRC such that the sending rate is never reduced below the reserved rate, which obviously improves performance of the protocol.

However, the problem with this scheme is to decide what rate to reserve. It should be significantly higher than the rate of the base layer, or else the performance gain will be marginal. On the other hand, the more bandwidth reserved for each flow, the less number of flows can be admitted by the admission control. This results in non-optimal utilization of the total capacity, simply because statistical multiplexing effects for VBR video can be substantial.

AF vs EF

The AF PHB (Per Hop Behavior) in DiffServ is targeted at semi-elastic flows with certain bandwidth requirements, e.g one-way streaming video. The SLA specifies the assured rate for a flow (or flow aggregate), and the ingress routers perform traffic conditioning operations such as metering, re-marking, shaping and dropping of out-of-profile packets. For interacting services such as video conferencing, delay requirements may prohibit the use of AF, leaving EF as the only option. In this paper however, simulations are presented that indicate that by using proper congestion control in the AF class, and having small buffers in the routers, queuing delay can be significantly improved without increasing packet loss probability. Unfortunately, this does not hold for TFRC, which require relatively large buffers in order to keep packet loss low.

Drop precedence

In the AF PHB, packets can be marked with up to three different drop precedence levels. Regardless of the congestion control in use, this functionality facilitates protection of the more important packets in the video stream (i.e. base layer packets) using AQM (Active Queue Management) in the routers. The relative importance of the enhancement layers depends to an extent on user preference, e.g, if high SNR (Signal-to-Noise Ratio) quality is preferred over high temporal resolution or vise versa. It is not the intention of this paper to suggest a specific mapping scheme, but it was necessary to choose a sample mapping to demonstrate the potential quality gain.

3. Packet Pair Assisted Congestion Control

3.1. Overview

In TFRC, the sending rate will increase until either a packet loss occurs or average RTT increases. Adjusting the size of the router buffers affects which of these events will occur most often, but for inelastic video applications, both events are unfortunate. PPACC is an attempt to avoid this trade-off, by using packet pair estimates as a tool to get additional information about the available bandwidth. To minimize the effects of the bias and variance issues mentioned in section 2.3, the dynamics of the estimates are used to decide the amount of rate increase or decrease; If estimated available bandwidth is strongly growing, a more aggressive rate increase can be allowed, but if the estimates are only moderately growing, or even decreasing, a more conservative rate claim is performed. An equivalent approach is applied when reducing the sending rate.

The packet pair estimator used is the same as in Spruce [22], which is based on the Probe Gap Model (PGM):

$$\hat{A} = 2\hat{C}_b - \frac{\delta' \left(\hat{C}_b\right)^2}{s} \tag{1}$$

where $\hat{C}_b$ is the estimated bottleneck link capacity, δ' is the output gap and s is the packet size. As in Spruce, to avoid the situation where the first probe packet leaves the bottleneck router before the second arrives, the input gap δ is set to s/C_b. The main difference between Spruce and PPACC, is the interval between the pairs. While Spruce uses Poisson intervals to gain from the PASTA property, PPACC is restricted by the application packet stream, and can only generate a pair whenever two packets arrive back-to-back. Note also the requirement of a priori knowledge about the bottleneck capacity. This information can be shared globally among all sessions sending to the same destination, with a validity typically much longer than a video session lifetime. Each packet pair sample can contribute to this *shared* estimate, as described in section 2.3.

3.2. Receiver operation

When the PPACC receiver recognizes a packet pair, it measures the output gap, applies equation (1) and keeps a smoothed version of the estimate using EWMA (Exponentially Weighted Moving Average). This value is fed back to the sender periodically. Additionally, in case of detected packet loss, explicit packet loss notifications are transmitted.

3.3. Sender operation

In addition to generating the pairs, the sender's primary task is to calculate the *target rate* based on the feedback packets from the receiver. The target rate is comparable with TFRC's allowed rate resulting from the TCP throughput equation.

Generating pairs

When two packets arrive back-to-back from the application layer, a pair is "generated" by tagging the first packet with a *first-in-a-pair* bit, and the second is held back for δ seconds. It is also made sure that the first packet is not sent too close to a previously sent packet. This reduces the risk of queuing of the first packet which potentially can cause the input gap to decrease before reaching the bottleneck. Also, because routers can be either *store and forward* or *virtual cut through*, both packets should be of equal size.

Congestion control state machine

An overview of the sender operation is given in the state diagram in figure 2, and consists of three states; INCREASE, DECREASE and FLAT. The FLAT state

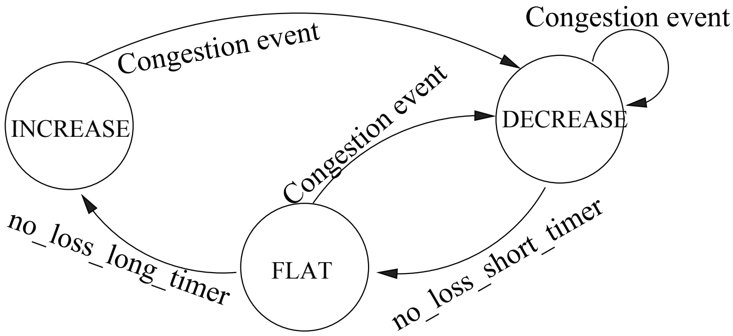


Figure 2. PPACC sender state machine. The state names refers to the dynamics of the target rate in the respective states. As long as no congestion is experienced, target rate is gradually increased. If a congestion event occurs, the DECREASE state is entered, and the no_loss_short_timer is started. In this state, as long as new congestion events arrive, the target rate is reduced. In the FLAT state, the target rate is kept unchanged for a random multiple of GOP periods, before returning to the INCREASE phase.

is introduced to avoid high frequent oscillations in the target rate, and to avoid global synchronization among the competing senders.

Adaptive bias removal

Before calculating the target rate, an adaptive bias removal technique is applied to the estimated available bandwidth. This is done by introducing the bias estimator $\hat{\beta}$, which is subtracted from $\hat{A}$. $\hat{\beta}$ is simply the EWMA of the difference between $\hat{A}$ and the actual sending rate R_{actual} at the time of each packet loss detection. This follows from the intuition that sending with the rate R_{actual} caused a packet drop, so R_{actual} is therefore close to the real available bandwidth.

Calculating the target rate

The target rate R_{target} is updated at the beginning of each GOP. As described in figure 3, as long as the current target rate is smaller than the estimated available bandwidth, the sender attempts to maintain $\frac{dR_{target}}{dt} = \frac{d\hat{A}}{dt}$. A more conservative approach is followed if the target rate is above the estimated available bandwidth. The constants ϵ and κ enforces minimum and maximum rate increase respectively.

The algorithm followed in the DECREASE state is equivalent, and will not be described here.

4. NS2 Simulator

In addition to the PPACC transport module, the following was implemented:

if (R_{target} ¡= $\hat{A}_{adj}$){
R_{target} += max(ϵ , min(κ , $\frac{d\hat{A}}{dt}$ * t_{GOP}))
}else{
if ($\frac{d\hat{A}}{dt}$ ¿= 0) R_{target} += ϵ
else ¡no change¿
}

Figure 3. INCREASE phase

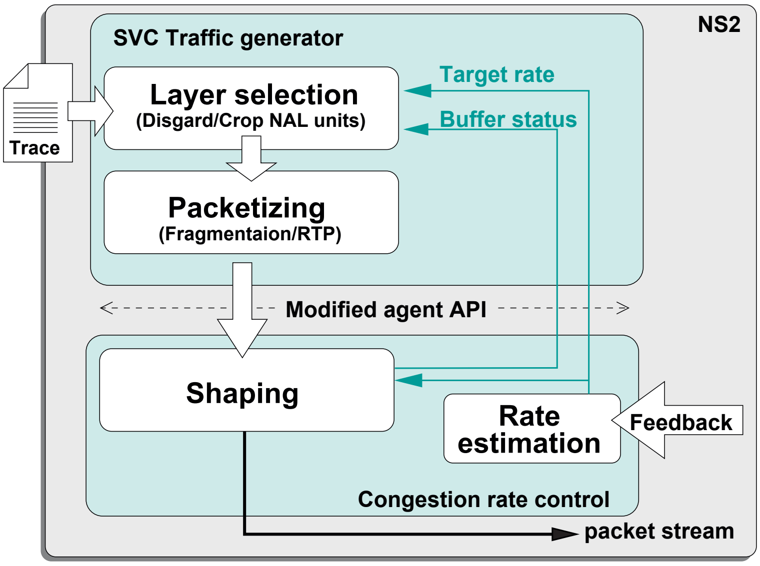


Figure 4. SVC Traffic generator in NS2

4.1. SVC traffic generator

A trace based SVC traffic generator has been implemented in NS2, which can be configured to either use TFRC or PPACC as transport protocol. Figure 4 gives an overview of the SVC source node.

The generator reads NAL units from a trace file produced by the encoder [23]. The layer selector chooses when to add or drop layers in each of the three dimensions, based on knowledge of future GOPs and the target rate from the transport entity. It first selects a spatial-temporal combination that not exceeds the target rate, and fills up the residual rate with the FGS quality enhancement layer. The exact selection algorithm is not described here, but is similar to the one in [12]. Further, the filtered NAL units are packetized before being passed to the transport module.

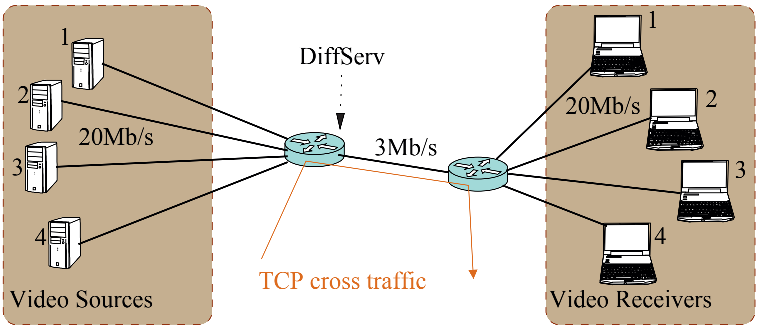


Figure 5. Simulation topology

4.2. SVC Receiver

This module performs all tasks needed on the receiver side. This includes packet loss detection, packet pair estimation (PPACC only), reassembly of NAL units, sending feedback packets and gathering statistics. Also, a simple play-out buffer was implemented, to be able to discard late arriving packets.

4.3. Modifications to NS2

To be able to send packets containing a payload, a send queue was added to the TFRC sender, and the data sending API in TFRC was changed. Also, it was necessary to modify the agent interface to make information of the target rate and the send queue available from the application level.

5. Simulation results

The main purpose of this experiment is to show the effect of router buffer size on the performance of the two transport protocols. This is done by comparing the two output parameters, one-way delay and packet loss ratio.

5.1. Simulation setup

The simulations were run using the topology in figure 5. The cross traffic on the congested link consists of two TCP flows, and are marked with the AF2x code point, while the four video flows are marked with the AF1x . WFQ (Weighted Fair Queuing) scheduling is used between the two classes, and 2Mbps is reserved for the AF1 class. The AF1x queue uses WRED (Weighted Random Early Detection), giving lower drop probabilities for the base layer packets, while the AF2x queue uses tail-drop. The only input parameter differing in these

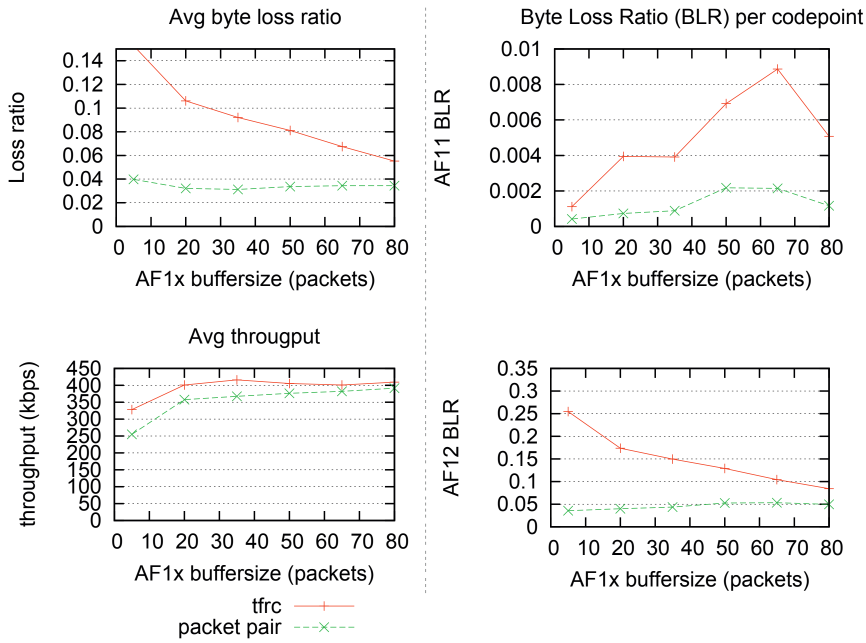


Figure 6. Left side: Average byte loss ratio and throughput for all video flows. Right side: Average byte loss ratio for different code points.

simulations is the buffer size in the bottleneck router. The largest value used was 80 packets, which roughly translates into 400ms maximum one-hop delay.

The stream was encoded with two spatial, four temporal and two quality levels as illustrated in figure 1. The test material was extracted from [24].

5.2. Packet loss

The results are shown in figure 6, and clearly illustrate the differences between the two protocols in terms of packet loss. For smaller buffers, when using TFRC, the router drops more than 10% of all video packets, and even more if looking only at the AF12 class (right side of figure 6). The simulations also show that DiffServ does a good job protecting the base layer for both protocols, all though PPACC performs slightly better also here.

5.3. Throughput

TFRC's aggressive bandwidth probing improves the overall throughput of the video flows, but this is also what causes the higher drop rates. It is discussable whether this relatively small increase in throughput can defend the large increase in packet loss. From a network point-of-view, it can't, because dropped packets have unnecessarily confiscated upstream network resources. However from the application point-of-view, it is not that simple; On the one hand, more received data means more information for the decoder. In turn, this generally means improved quality, at least for numeric assessments such as PSNR (Peak Signal-to-Noise Ratio). On the other hand, when the packet loss is high, the application loses control over which packets that successfully reach the decoder. Absence of some packets may render other received packets useless, due to temporal and spatial dependencies among the NAL units.

All though TFRC achieves a slightly higher throughput and possibly higher average PSNR, its aggressiveness will certainly increase the variance of PSNR compared to PPACC, which has a negative effect on the user perceived quality. This, in combination with the fact that aggressive behavior fills up router buffers and contributes to an unstable network, makes it difficult to defend the throughput increase.

5.4. Delay variations

As seen above, larger buffers in the routers will lead to a decrease in the packet loss ratios observed when using TFRC. However, larger buffers will have negative impact on the delay jitter for the video flows, which will have to be compensated with larger play-out buffer at the receiver. For one-way streaming, this leads to longer pre-buffering periods. For two-way multimedia applications however, where delays should typically not exceed 150-200ms [25], larger buffers is not a solution. In figure 7a, the distributional properties of one-way delay for video packets are illustrated using box-plots. Comparing PPACC and TFRC, three observations can be made:

- PPACC generally have less delay that TFRC.
- For TFRC, the delay distribution has a heavier tail,
- For PPACC, the tail of the delay distribution is significantly more correlated with the buffer size than for TFRC.

If queuing in the congested router was the only source of delay, one would expect a linear increase in the maximum observed delay when adding more buffer space. For PPACC, this seems to hold, but not for TFRC. The reason is TFRC's shaping of the packet stream using the calculated allowed sending rate. This rate is updated once each RTT, while the application adjusts its layer combination at the beginning of each GOP. In our configuration, and generally, a GOP is longer than RTT. This means that in periods for as long as a GOP, the application may send data at a higher rate than TFRC outputs, requiring a send buffer at the source. Indeed, our simulations show that this behavior has a significant effect on the delay distribution.

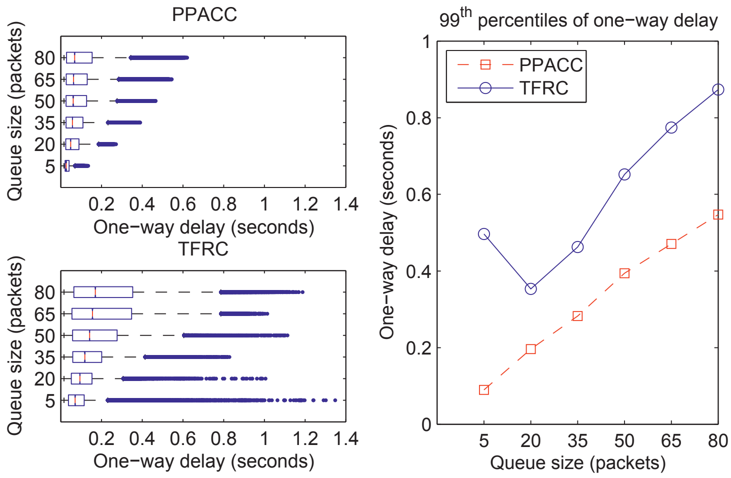


Figure 7. a) Box-plots summarizing the distribution of one-way delay of video packets. b) 99^{th} percentiles of delay distribution. I.e. 99 percent of all packets have equal or less one-way delay than indicated by these graphs

As pointed out above, applications may use play-out buffer to compensate for the delay jitter. Figure 7b plots the 99^{th} percentiles for the delay distributions as functions of router buffer size. That is, if you accept that 1% of the packets arrive too late, this gives a good indication on how large play-out buffers are needed. This emphasizes the advantage of avoiding TFRC's rate shaping, both because smaller play-out buffers are needed, and because stronger correlation between network buffers and delay distribution results in more predictable behavior. The latter simplifies the process of deciding the length of the buffers.

6. Conclusion

TFRC has been compared with PPACC, when sending scalable VBR video in a DiffServ environment. The results yield significant differences in delay jitter in TFRC's disfavor. There will always be a trade-off between packet loss and delay when configuring the router buffer size, but this study has shown that it is possible to design a transport protocol that can handle smaller buffer sizes and at the same time keep the packet loss low. A consequence of this is the ability to stream VBR video with stringent delay jitter requirements in an AF class, leaving the EF class to applications with even stronger QoS needs. However, for one-way streaming applications with looser delay requirements, TFRC and larger buffers may be the best choice.

The proposed packet pair technique is still immature in a congestion control context, but the simulation results show that in spite of the bias and variance issues, packet pairs provide valuable information when estimating the available bandwidth.

7. References

[1] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An Architecture for Differentiated Service," RFC 2475 (Informational), Internet Engineering Task Force, Dec. 1998, updated by RFC 3260. [Online]. Available: http://www.ietf.org/rfc/rfc2475.txt

[2] B. Davie, A. Charny, J. Bennet, K. Benson, J. L. Boudec, W. Courtney, S. Davari, V. Firoiu, and D. Stiliadis, "An Expedited Forwarding PHB (Per-Hop Behavior)," RFC 3246 (Proposed Standard), Mar. 2002.

[3] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski, "Assured Forwarding PHB Group," RFC 2597 (Proposed Standard), Jun. 1999, updated by RFC 3260.

[4] ITU-T and ISO/IEC, "Advanced Video Coding for Generic Audiovisual Services," ITU-T Recommendation H.264 - ISO/IEC 14496-10 (AVC), November 2007.

[5] M. Handley, S. Floyd, J. Padhye, and J. Widmer, "Tcp friendly rate control (tfrc): Protocol specification. IETF RFC 3448," Jan 2003.

[6] Y. Huang, G. Xu, and L. Huang, "Adaptive AIMD rate control for smooth multimedia streaming," in *Computer Communications and Networks, 2003. ICCCN 2003. Proceedings. The 12th International Conference on*, 2003, pp. 171–177.

[7] D. Loguinov and H. Radha, "End-to-end rate-based congestion control: convergence properties and scalability analysis," *Networking, IEEE/ACM Transactions on*, vol. 11, no. 4, pp. 564–577, 2003.

[8] N. Sastry and S. Lam, "CYRF: a theory of window-based unicast congestion control," *Networking, IEEE/ACM Transactions on*, vol. 13, no. 2, pp. 330–342, 2005.

[9] N. Cranley, L. Murphy, and P. Perry, "User-perceived quality-aware adaptive delivery of MPEG-4 content," in *NOSSDAV '03: Proceedings of the 13th international workshop on Network and operating systems support for digital audio and video.* New York, NY, USA: ACM Press, 2003, pp. 42–49.

[10] T. Kim and M. Ammar, "Optimal quality adaptation for scalable encoded video," *Selected Areas in Communications, IEEE Journal on*, vol. 23, no. 2, pp. 344–356, 2005.

[11] P. Ssesanga and M. Mandal, "Efficient congestion control for streaming scalable video over unreliable IP networks," in *Image Processing, 2004. ICIP '04. 2004 International Conference on*, vol. 3, 2004, pp. 2059–2062 Vol. 3.

[12] D. T. Nguyen and J. Ostermann, "Streaming and congestion control using scalable video coding based on H.264/AVC," *Journal of Zhejiang University SCIENCE A*, vol. 7(5), pp. 749–754, 2006.

[13] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the H.264/AVC Video Coding Standard," *Circuits and Cystems for Video Technology, IEEE Transactions on*, vol. 13, no. 7, pp. 560–576, July 2003.

[14] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable h.264/mpeg4-avc extension," in *Image Processing, 2006 IEEE International Conference on*, 2006, pp. 161–164.

[15] E. Kohler, M. Handley, and S. Floyd, "Datagram Congestion Control Protocol (DCCP), RFC 4340," March 2006.

[16] J. Padhye, V. Firoiu, D. F. Towsley, and J. F. Kurose, "Modeling tcp reno performance: A simple model and its empirical validation," *IEEE/ACM Transactions on Networking*, vol. 8, no. 2, pp. 133–145, April 2000.

[17] I. Rhee and L. Xu, "Limitations of equation-based congestion control," in *SIGCOMM '05: Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications*. New York, NY, USA: ACM Press, 2005, pp. 49–60.

[18] V. Jacobson, "Congestion avoidance and control," in *ACM SIGCOMM '88*, Aug 1988, pp. 314–329.

[19] V. E. Paxson, "Measurements and Analysis of End-to-End Internet Dynamics," PhD Dissertation, University of California, Lawrence Berkeley National Laboratory, April 1997. [Online]. Available: citeseer.ist.psu.edu/paxson97measurements.html

[20] Y. Bernet, P. Ford, R. Yavatkar, F. Baker, L. Zhang, M. Speer, R. Braden, B. Davie, J. Wroclawski, and E. Felstaine, "A Framework for Integrated Services Operation over Diffserv Networks," RFC 2998 (Informational), Internet Engineering Task Force, Nov. 2000. [Online]. Available: http://www.ietf.org/rfc/rfc2998.txt

[21] G. Jourjon, E. Lochin, L. Dairaine, P. Senac, T. Moors, and A. Seneviratne, "Implementation and performance analysis of a qos-aware tfrc mechanism," in *Networks, 2006. ICON '06. 14th IEEE International Conference on*, vol. 2, Sept. 2006, pp. 1–6.

[22] J. Strauss, D. Katabi, and F. Kaashoek, "A measurement study of available bandwidth estimation tools," in *IMC '03: Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement*. New York, NY, USA: ACM, 2003, pp. 39–44.

[23] Joint Video Team (JVT), "JSVM reference software version 5.2."

[24] Digital Cinema Initiatives (DCI) and The American Society of Cinematographers (ASC), "StEM mini-movie access procedure (wedding scene)," November 2004. [Online]. Available: http://www.dcimovies.com

[25] ITU-T Recommendation G.114, "General Characteristics of International Telephone Connections and International Telephone Circuits: One-Way Transmission Time," Feb. 1996.

Paper C

Protecting scalable video flows from congestion loss

Bjørnar Libæk and Øivind Kure

Proceedings of the Fifth International Conference on Networking and Services (ICNS), Valencia, Spain, April 20-25 2009.

Awarded as best paper

Abstract

To minimize packet loss caused by bandwidth probing, a video congestion control algorithm may use Forward Error Correction (FEC) to reproduce dropped packets. Unfortunately, if a dominant part of the competing flows adds redundancy, the extra overhead may cause increased packet loss ratios in the network, and in turn do more harm than good. In this paper however, we demonstrate both analytically and with simulations, that by using a standard DiffServ dropping mechanism and assigning a lower precedence level to FEC packets, the negative effect of the added overhead can be strongly pacified. The analysis also show that the FEC benefit is clearly maximized when router buffers are small, favoring multimedia flows with delay and jitter requirements.

1. Introduction

As video traffic is consuming an increasing amount of the total bandwidth resources on the internet, the need for video congestion control is widely acknowledged both in the research community and in the industry. Examples are IETF's TFRC [1] (TCP Friendly Rate Control) and the scalable extension [2] to the H.264/AVC [3] video codec respectively.

Most congestion control mechanisms use variants of *bandwidth probing*. The available bandwidth is estimated by gradually increasing the sending rate to the point where congestion events start to occur, after which the sending rates is decreased. For multimedia flows, these inherent congestion events (e.g packet loss or increased delay) are likely to harm the output quality.

Historically, to protect against packet loss caused by congestion, the most common approach is to retransmit the lost packets as done by TCP (Transmission Control Protocol). Unfortunately, this is unsuitable for delay sensitive video applications. An intuitive response is to use Forward Error Correction (FEC) to be able to reproduce lost data at the receiver side, lowering the *effective PLR* (packet loss ratio). A single flow will always gain from adding FEC if the *channel PLR* remains unchanged. However, the fundamental problem in the context of congestion control is that when a large portion of the total traffic volume is protected by FEC, the extra overhead is likely to cause a higher channel PLR. In some situations, typically at low offered load, the amount of recovered packets may be large enough to lower the effective PLR below the original channel PLR (i.e. the PLR before adding FEC). In other situations however, the effective PLR may in fact become higher than the original channel PLR. For a given redundancy level, as the load increases there often exist a *cut-off point* after which the effective PLR grows beyond the original PLR. Obviously, when the relative amount of traffic using FEC is large, the cut-off point will be reached earlier than otherwise. However, if protection against congestion loss is the goal, the cut-off point should be reached as late as possible, preferably never. To reduce the negative impact of the increased load, we study the effect of assigning a lower priority to the FEC packets. Both analytical studies and simulation results are presented indicating that this approach can in fact eliminate the cut-off point, which motivates the use of FEC in congestion control.

In section 3, the existence of the cut-off point is demonstrated analytically using the M/M/1/B queuing model. A modified model taking into account packet prioritization is presented, showing the positive effect of assigning a lower precedence level to FEC packets. In section 4, we present results from simulations using real H.264/AVC encoded video flows, confirming the analytical observations. Further, in section 5, a possible way of incorporating FEC in a congestion control algorithm for scalable video is suggested, before concluding in section 6.

2. Related work

In [4], Cidon et al. study the packet loss distribution in single server queuing systems, and emphasize the danger of assuming independence in the packet loss process. It is demonstrated that this often leads to optimistic results when estimating the error correcting capabilities of FEC schemes. Also, the authors strongly argue against the use of FEC because of the increased load imposed by the overhead. The arguments are however based on results from ATM examples, not necessarily applicable to today's IP networks.

In [5], Dán et al. investigate the impact of the packet size distribution (PSD) on the packet loss process and in turn on the performance of FEC, concluding that the exponential distribution is a good approximation of the PSDs in the internet in this respect.

A large number of contributions have been made to the literature, where FEC is applied to improve performance during congestion. Several more or less sophisticated mechanisms are proposed to adapt the redundancy level to the changing network conditions. What often seems to be ignored however, is the self-induced change in the packet loss process when adjusting the redundancy level. Examples are [6], [7] and [8].

3. Analytical motivation

To protect network streams from packet loss, Reed-Solomon (R-S) codes [9] are widely used. Here, A block of k data packets are represented by $n > k$ coded packets, and the receiver is capable of correcting up to $n - k$ packet drops. We assume *systematic FEC*, meaning that all data packets are represented uncoded in the packet stream, and therefore are individually "decodable".

We define the *effective packet loss ratio* in a block as $E_{plr} = L_d/k$, where L_d is the number of missing data packets in the block after the correction process. Further, define the long term average channel packet loss ratio to be a function of the offered traffic: $p_c(\rho)$, and the parameter $\alpha \in [0, 1]$ to express the relative amount of traffic protected by FEC. Let ρ' and ρ'' denote the offered load before and after adding FEC respectively. Observe that

$$\rho'' = \rho' + \alpha\rho' \frac{n-k}{k}$$

expresses the total load, where the second term on the right hand side is the *FEC overhead*. Using this framework, we see that FEC is only beneficial when

$$\mathbf{E}[E_{plr}|\rho''] < p_c(\rho') \tag{1}$$

i.e, the expected effective packet loss ratio when using FEC must be smaller than the original packet loss ratio. The previously mentioned cut-off point is found when left and right hand sides of the above expression are equal.

In the following, expressions for $\mathbf{E}[E_{plr}]$ and $p_c(\rho)$ are derived for two different bottleneck queuing models. First, an ordinary FIFO queue is studied using the well known M/M/1/B model [10]. Second, we extend the model to handle service differentiation using a simple *dual tail-drop* priority queue.

In both models, the expression for $\mathbf{E}[E_{plr}]$ [1] is found by conditioning on the number of dropped packets j in a block:

$$\mathbf{E}[E_{plr}] = \sum_{j=0}^{n} \frac{\mathbf{E}[L_d|j]}{k} P(j,n) \tag{2}$$

where $P(j,n)$ is the PMF (Probability Mass Function) for the packet loss distribution (i.e, the probability of j drops out of n consecutive packets). Due to the lack of a closed form expression for $P(j,n)$ in the M/M/1/B queue, we use the recursion method from [4] to derive it.

3.1. Tail-drop model

3.1.1. The M/M/1/B queuing model

The stationary probability of having i packets in the queue is

$$\Pi_1(i,\rho) = \Pi_1(0,\rho)\rho^i \quad , \quad i = 0,1,..,B$$

, where $\Pi_1(0,\rho) = 1/\sum_{i=0}^{B} \rho^i$ is the probability of an empty queue, ρ is the offered load and B is the buffer size. The channel packet loss ratio is simply the probability of a full queue:

$$p_c(\rho) = \Pi_1(B,\rho) \tag{3}$$

[1] We omit the condition on ρ (see equation 1), although note that both $P(j,n)$ (and $\mathbf{E}[E_{plr}]$ in the priority model) depends on it.

3.1.2. Effective packet loss ratio

Because of the error correcting capabilities of R-S, the expected number of uncorrected packets can be expressed as

$$\mathbf{E}[L_d|j] = \begin{cases} 0 & , j \leq n-k \\ j\frac{k}{n} & , j > n-k \end{cases}$$

I.e, all packets can be recovered if no more than $n-k$ are lost, but no lost packets are recoverable if more than $n-k$ packets are lost[2]. Inserting into (2) gives:

$$\mathbf{E}[E_{plr}] = \sum_{j=n-k+1}^{n} \frac{j}{n} P(j,n) \tag{4}$$

3.1.3. Packet loss distribution

As mentioned, the recursion method outlined in [4] is used to calculate $P(j,n)$. For convenience, we repeat the equations. The recursion is initialized at $n = 1$:

$$P_i(j,1) = \begin{cases} 1 & , j = 0 \\ 0 & , j \geq 1 \end{cases} \quad , i = 0, 1, .., B-1 \tag{5}$$

$$P_B(j,1) = \begin{cases} 1 & , j = 1 \\ 0 & , j = 0, \quad j \geq 2 \end{cases} \tag{6}$$

and continues for $n > 1$ with:

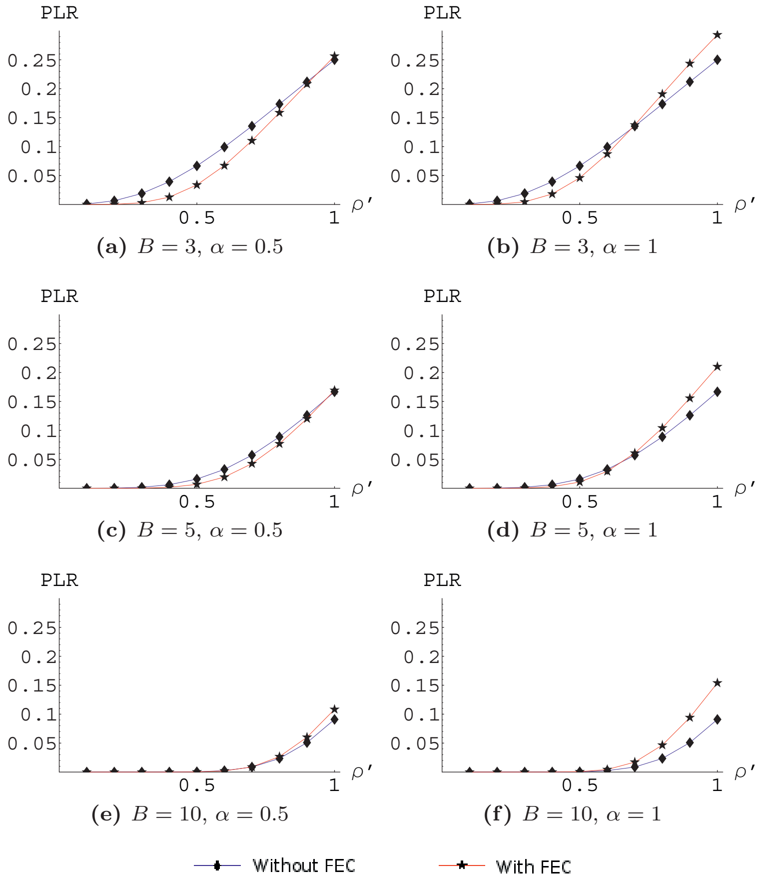
$$P_i(j,n) = \sum_{k=0}^{i+1} Q_{i+1}(k) P_{i+1-k}(j, n-1), \quad 0 \leq i \leq B-1 \tag{7}$$

$$P_B(j,n) = Q_B(k) P_{B-k}(j-1, n-1)$$

where $Q_i(k)$ is the probability of k packets departing during an inter-arrival period. The packet loss distribution is finally derived by conditioning on the system state:

$$P(j,n) = \sum_{i=0}^{B} \Pi_1(i, \rho'') P_i(j,n)$$

[2]Note that if more than $n-k$ packets are dropped, some of the data packets may still arrive at the receiver. Because we assume *systematic FEC*, all data packets are sent uncoded, and are therefore individually “decodable”.



(a) $B = 3$, $\alpha = 0.5$ (b) $B = 3$, $\alpha = 1$ (c) $B = 5$, $\alpha = 0.5$ (d) $B = 5$, $\alpha = 1$ (e) $B = 10$, $\alpha = 0.5$ (f) $B = 10$, $\alpha = 1$

Figure 1. Using the tail-drop model: $\mathbf{E}[E_{plr}|\rho'']$ and $p_c(\rho')$ as functions of ρ'. Examples are shown both for $\alpha = 1$ (no cross traffic) and $\alpha = 0.5$. The x-axis shows the traffic load before adding FEC.

3.1.4. Evaluation

In figure 1, $\mathbf{E}[E_{plr}|\rho'']$ and $p_c(\rho')$ are plotted as functions of ρ', using equation 4 and 3 respectively. The graphs clearly illustrate that the cut-off point exists, and that the FEC benefit is larger when cross traffic is added (figure 1 a, c and e). Another thing worth mentioning is that the benefit from FEC diminishes quite quickly as the buffer size increases. This motivates the use of FEC for delay sensitive applications where queuing delays are critical.

3.2. Priority model

This model is targeted at DiffServ networks, where all video flows can be aggregated into a single service class. From the assumption on systemic FEC follows the fact that data packets are more valuable than FEC packets. For this reason, a higher precedence level (i.e lower relative drop probability) can be assigned to the data packets.

In the following, we extend the analytical model presented in section 3.1 to handle a simple two-priority queue. Instead of having the same tail-drop threshold (i.e buffer size B) for all packets, a low-threshold L is used as a virtual buffer size for low priority packets. Low priority (i.e. FEC) packets are only accepted if the current number of packets in the buffer is less than this value, and dropped otherwise. This effectively reserves $B - L$ slots in the buffer for high priority packets.

To achieve this, modifications had to be done both to the queuing model, the $\mathbf{E}[L_d|j]$, as well as the packet loss distribution.

3.2.1. The $M/M/1/B_2$ queuing model

In the $M/M/1/B_2$ model, we assume that there are two Poisson sources, one for high priority and one for low priority packets. High and low priority packets arrive with intensity λ_h and λ_l respectively, and the total load of the aggregate Poisson process is $\rho = (\lambda_h + \lambda_l)/\mu$. Also, we define $\sigma = \lambda_h/(\lambda_l + \lambda_h)$ to be the relative amount of high priority packets.

The stationary probabilities are easily derived using the same approach as for the standard $M/M/1/B$ model [10]. The only difference is that the arrival intensity drops from $\lambda_l + \lambda_h$ to λ_h after state L:

$$\Pi_2(i, \rho) = \begin{cases} \Pi_2(0, \rho)\rho^i & , \quad i = 1, 2, .., L \\ \Pi_2(0, \rho)\rho^L(\sigma\rho)^{i-L} & , \quad i = L + 1, .., B \end{cases}$$

where $\Pi_2(0, \rho) = 1/[\sum_{i=0}^{L} \rho^i + \rho^L \sum_{i=L+1}^{B} (\sigma\rho)^{i-L}]$. The stationary drop probabilities for high and low priority packets are given by

$$\begin{aligned} p_h(\rho) &= \Pi_2(B, \rho) \\ p_l(\rho) &= \textstyle\sum_{i=L}^{B} \Pi_2(i, \rho) \end{aligned}$$

respectively, while the overall packet loss ratio for the system is given by the weighted average:

$$p_c(\rho) = \sigma p_h(\rho) + (1 - \sigma)p_l(\rho)$$

3.2.2. Effective packet loss ratio

Taking into account that the drop probability is different for data and FEC packets, the expected number of dropped data packets in a block given the total number of dropped packets j can now be expressed as

$$\mathbf{E}[L_d|j] = \begin{cases} 0 & , j \leq n-k \\ j\frac{k}{n}\frac{p_h(\rho)}{p_c(\rho)} & , j > n-k \end{cases}$$

Inserting into (2) gives

$$\mathbf{E}[E_{plr}] = \sum_{j=n-k+1}^{n} \frac{j}{n}\frac{p_h(\rho)}{p_c(\rho)} P(j,n) \tag{8}$$

, where $P(j,n)$ is an adaptation of Cidon's model to the $M/M/1/B_2$ model, explained in the next paragraph.

3.3. Packet loss distribution

In order to calculate the packet loss distribution in the $M/M/1/B_2$ model, Cidon's equations [4] 4 and 6 were modified, while 1,2,3 and 5 is unchanged. For $n = 1$:

$$P_i(j,1) = \begin{cases} 1 & , j = 0 \\ 0 & , j \geq 1 \end{cases} \quad , i = 0, 1, .., L-1 \tag{9}$$

$$P_i(j,1) = \begin{cases} \frac{k}{n} & , j = 0 \\ \frac{n-k}{n} & , j = 1 \\ 0 & , j \geq 2 \end{cases} \quad , i = L, L+1, .., B-1 \tag{10}$$

$$P_B(j,1) = \begin{cases} 1 & , j = 1 \\ 0 & , j = 0, \quad j \geq 2 \end{cases} \tag{11}$$

Equation 9 is the same as 5 for $i < L$. However, when $i \leq L$, the probability of dropping a packet depends on whether this is a high or low priority packet. For $n > 1$:

$$P_i(j,n) = \sum_{k=0}^{i+1} Q_{i+1}(k) P_{i+1-k}(j, n-1), \quad 0 \leq i \leq L-1$$

$$\begin{aligned} P_i(j,n) = {} & \frac{k}{n} \sum_{k=0}^{i+1} Q_{i+1}(k) P_{i+1-k}(j, n-1) + \\ & \frac{n-k}{n} \sum_{k=0}^{i} Q_i(k) P_{i-k}(j-1, n-1), \quad L \leq i \leq B-1 \end{aligned} \tag{12}$$

$$P_B(j,n) = Q_B(k) P_{B-k}(j-1, n-1) \tag{13}$$

Finally,

$$P(j,n) = \sum_{i=0}^{B} \Pi_2(i,\rho) P_i(j,n)$$

3.3.1. Evaluation

In figure 2, $\mathbf{E}[E_{plr}|\rho'']$ and $p_c(\rho')$ are plotted as functions of ρ', using equation 8 and 3 respectively. The low-threshold is set using $L = \lceil(1-\sigma)B\rceil$ where

$$\sigma = \frac{\rho'}{\rho''} = \frac{k}{\alpha(n-k)+k}$$

is the relative amount if high priority traffic.

As with the tail-drop model, we still see a larger gain from adding FEC when the bottleneck is shared with cross traffic. However, after the introduction of priority queuing, even when all flows use FEC, the cut-off point has been eliminated. Also, the requirement on small buffers is loosened. To summarize, these observations encourage further investigation of the combination of FEC and priority queuing.

4. Simulations

In this section, the indications from the analytical study are augmented with results from NS2 simulations where more realistic arrival patterns for video flows were used.

4.1. Scenario

The simple single bottleneck topology illustrated in figure 3 was used. The traffic consists of one foreground video flow, zero or more background video flows and a Poisson cross traffic flow. Figure 4 illustrates the components of the source and destination nodes for video flows. The video sources periodically read frames from a 60 second trace file containing a H.264/AVC encoded movie clip [11], shifted in time. A FEC module divides the packet stream into blocks of k packets each, and inserts $n-k$ packets as far away from each other as possible within the block.

In the simulations presented here, $k = 17$ and $n = 20$ were used. Clearly, the block length will affect the performance, but further sensitivity analysis on this is out of the scope of this paper. Generally, small block lengths lead to lower end-to-end delays, but also increase the probability of dropping FEC packets given that data packets are dropped from the same block.

The cross traffic source produces a packet stream with negative exponentially distributed (n.e.d.) packet sizes and inter-arrival periods. The distribution parameters were chosen to match the target sending rate $(1-\alpha)\rho' C_b$, where C_b is the bottleneck capacity.

The scenario is simulated with and without FEC, using different combinations of load, buffer size and the α parameter. Further, each of these combinations are simulated both with and without insertion of FEC packets. To produce graphs comparable to the ones presented in section 3, the packet loss rate from the

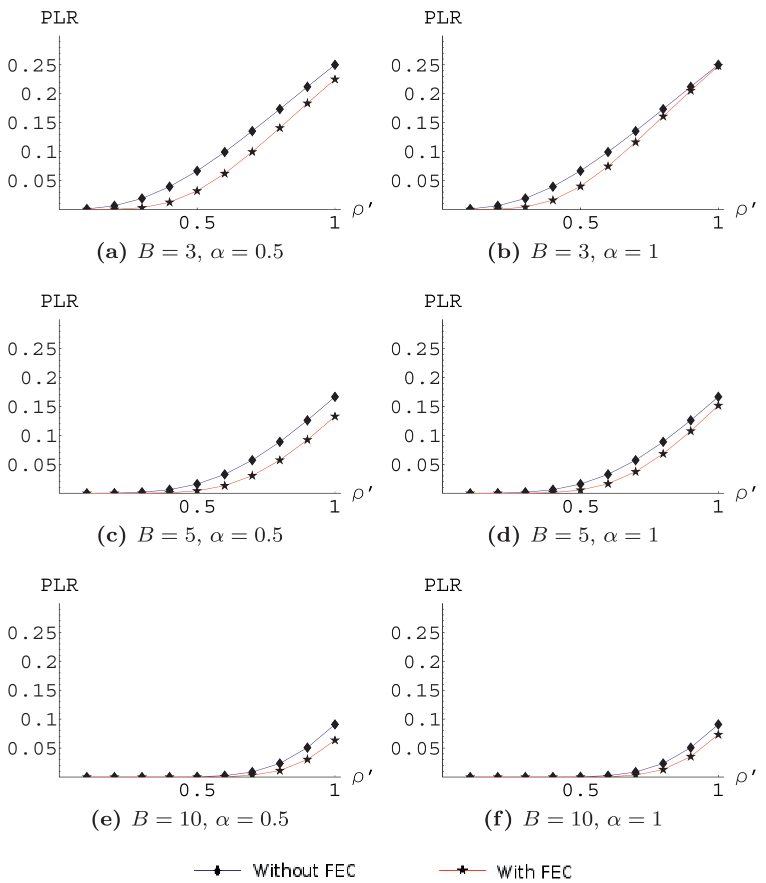


Figure 2. Using the priority model: $\mathbf{E}[E_{plr}|\rho'']$ and $p_c(\rho')$ as functions of ρ'. Examples are shown both for $\alpha = 1$ (no cross traffic) and $\alpha = 0.5$. The x-axis shows the traffic load before adding FEC.

simulation runs where FEC is turned off is plotted against the effective packet loss rate (i.e. the packet loss rate after FEC reproduction) from the runs where FEC is turned on.

In the following, two experiments based on the scenario above are presented. In the first one, a best effort network is used, while in the second experiment, priority queuing is introduced in the bottleneck router.

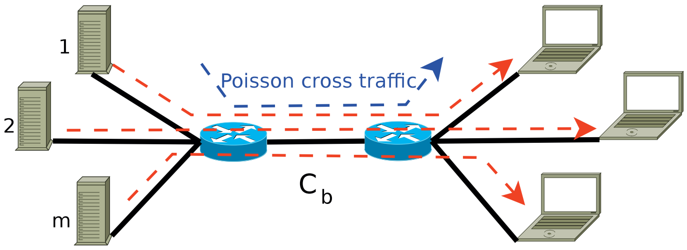


Figure 3. Simulation topology. The number of flows is changed to adjust the load.

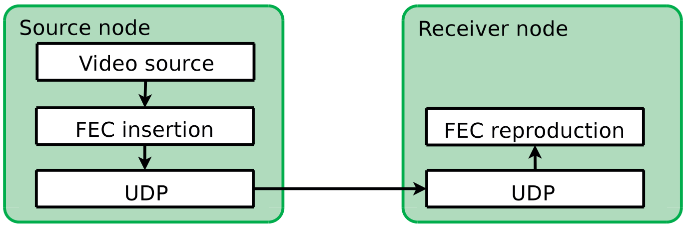


Figure 4. Video flow components

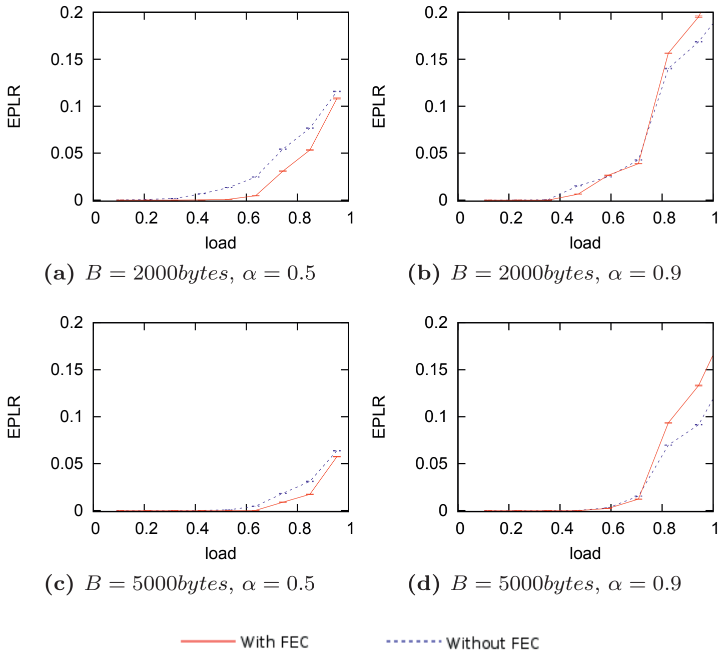
4.2. Experiment 1: Best effort

In this experiment, all packets (video data, FEC and cross traffic packets) are treated equally in the network. Figure 5 summarizes the simulation results, and to a large degree confirms the results obtained from the analytical study. When most of the capacity[3] is occupied by video flows (figures 5b and 5d), the FEC benefit is only marginal at lower loads, and the cut-off point is reached relatively early. As expected, this effect gets stronger for larger buffer sizes. When more cross traffic is added, the cut-off point is not present for any three of the buffer sizes, and the FEC benefit is significant.

4.3. Experiment 2: Priority

In these simulations, all FEC packets from the video flows were assigned a lower priority than video data and cross traffic packets. The dual tail-drop priority queue modeled in section 3.2 indicated a positive effect when introducing priority queuing. Besides being easy to model analytically, its availability in today's routers minimizes deployment issues. However, it may be challenging to configure the low threshold without knowing the relative distribution of high and low priority traffic.

[3]A small amount of Poisson cross traffic was added to have an element of randomness in the simulations.



(a) $B = 2000bytes$, $\alpha = 0.5$ (b) $B = 2000bytes$, $\alpha = 0.9$

(c) $B = 5000bytes$, $\alpha = 0.5$ (d) $B = 5000bytes$, $\alpha = 0.9$

Figure 5. Experiment 1 (Best effort)

For this reason, the priority experiment was carried out using both the dual tail-drop queue, and a zero configuration *strict priority FIFO* (SPF) queue. In this queue, all packets are accepted as long as the queue is not full. If a high priority packet arrives to a full queue, a low priority packet is "preempted" and the high priority packet is accepted. Contradictory to normal strict priority queuing (scheduling) where each priority has a separate FIFO queue, this approach preserves FIFO ordering and enforces a maximum delay on all packets that are not dropped. It is implemented by dropping packets at both tail and front, with O(1) complexity.

Figure 6 summarizes the results, confirming the main observation from the analytical study, namely that the introduction of priority queuing to a large degree eliminates the cut-off point. Comparing the two queuing algorithms, the SPF queue results in lower E_{plr} when cross traffic is high. At $\alpha = 0.9$ however, by using this particular configuration of the dual tail-drop we manage to avoid the cut-off, while by using the SPF queue we don't. Considering the fact that the SPF queue does not need configuration, it is clearly the best choice. However, it may be difficult to defend the upgrade of existing routers only to achieve a seemingly small performance gain compared to the dual-tail drop.

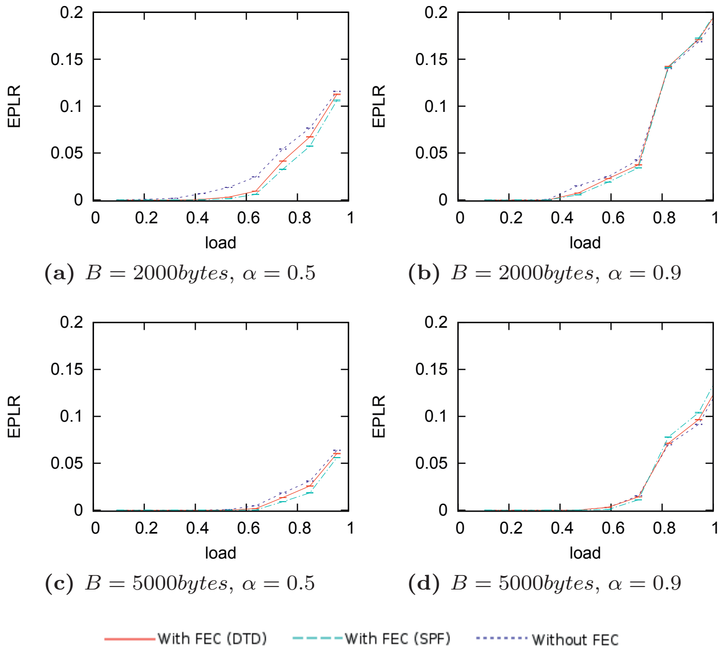


Figure 6. Experiment 2 (priority queuing): In the DTD case, the low threshold L is set as specified in section 3.3.1.

5. Applying FEC in video congestion control

As stated in the introduction, the main motivation behind this paper is to study to what degree FEC is able to protect delay sensitive video flows from packet loss caused by bandwidth probing. In this section, a possible approach is suggested, combining FEC and video congestion control.

We assume a scheme separating the functionality into an *application level rate control* (ARC) and a *transport level rate control* (TRC), each targeting different needs. The TRC estimates the available bandwidth by means of feedback from the opposite peer, and calculates the eligible sending rate. The ARC on the other hand, tries to optimize the video quality, by deciding when to add and drop enhancement layers (e.g H.264/AVC SVC [3] [2]) while conforming to the eligible rate reported by the TRC. Note that sending as much data as allowed does not necessarily translate into optimal video quality. Human perception typically disfavor frequent quality variations, so adding a new enhancement layer should only be done when the likelihood of having to drop it very soon is small. Also, the rate increase resulting from the new enhancement layer may cause increased packet loss and degraded perceived quality.

Define the start of a *probing period* as the time instance when the ARC observes that the reported rate from the TRC is large enough to accommodate a new enhancement layer. Instead of starting to send this layer right away, we propose to increase the sending rate within the bounds of the eligible rate, by adding FEC to the existing layers. The probing period is regarded as successful if the rate increase does not in turn cause the TRC to lower the rate within a configurable time interval. After a successful probe period, FEC is turned off and replaced by the new enhancement layer. Otherwise, if the TRC reports congestion, the probe period is interrupted and the ARC goes back to only sending the existing layers.

Judging from the simulation results, this approach will potentially decrease the effective packet loss both with and without priority support from the network. In situations where priority support is not available, there is a higher risk of reaching the cut-off point. However, because FEC is turned off when congestion is detected, this risk is reduced. Also, it is possible to indirectly control the α parameter by adjusting the length and frequency of probing periods, which will in turn affect the location of the cut-off point.

If priority support is not available, FEC will only protect the flow itself when entering a probe period. There is however a chance that the increased rate will cause damage to other flows, but a well designed TRC should be able ensure fairness among the flows in the long run. If priority support is available, low priority FEC packets from one flow is less likely to affect existing layers from other flows, loosening the fairness requirements for the TRC. For this reason, the choice of TRC is not obvious. In the best effort internet, where fairness is an important issue, IETF's TFRC (TCP Friendly Rate Control) [1] may be the best alternative. For delay-sensitive video however, more controlled environments such as DiffServ [12] networks is a likely requirement, providing precedence levels [13] and the ability to control the buffer size. As demonstrated in [14], TFRC suffers from high packet loss ratios when buffer sizes are small, so alternative congestion control approaches should be considered.

6. Conclusions and further work

To minimize packet loss caused by bandwidth probing, a video congestion control algorithm may use Forward Error Correction to reproduce dropped packets. Unfortunately, if a dominant part of the competing flows adds redundancy, the extra overhead may cause increased packet loss ratios in the network, and in turn do more harm than good. However, in this paper it has been demonstrated both analytically and with simulations, that by using a standard DiffServ dropping mechanism and assigning a lower precedence level to FEC packets, the negative effect of the added overhead can be strongly pacified. The analysis has also shown that the FEC benefit is maximized when router buffers are small. This correlates well with the delay and jitter requirements of multimedia flows.

The results motivate further studies of the combination of FEC and prioritization in the context of video congestion control. Open issues include choosing the block length k in a way that optimizes both effective packet loss and delay with respect to perceived quality. Also, the effect of the length of the probing periods should be studied, as it impacts both on user perception and overall performance of the congestion control algorithm. Likewise, studies on the sensitivity of the low threshold parameter in the dual tail-drop queue are needed. We intend to extend our network simulator with the algorithm sketched out in section 5, and to investigate these issues closer.

7. References

[1] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "TCP friendly rate control (TFRC): Protocol Specification. IETF RFC 5348," Sep 2008.

[2] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable h.264/mpeg4-avc extension," in *Image Processing, 2006 IEEE International Conference on*, 2006, pp. 161–164.

[3] ITU-T and ISO/IEC, "Advanced Video Coding for Generic Audiovisual Services," ITU-T Recommendation H.264 - ISO/IEC 14496-10 (AVC), November 2007.

[4] I. Cidon, A. Khamisy, and M. Sidi, "Analysis of packet loss processes in high-speed networks," *Information Theory, IEEE Transactions on*, vol. 39, no. 1, pp. 98–108, Jan 1993.

[5] G. Dán, V. Fodor, and G. Karlsson, "Packet size distribution: An aside?" in *Quality of Service in Multiservice IP Networks*, vol. Volume 3375/2005. Springer Berlin / Heidelberg, 2005, pp. 75–87. [Online]. Available: http://www.springerlink.com/content/wc5l41t90kn4kl9g/

[6] S. ryong Kang and D. Loguinov, "Impact of fec overhead on scalable video streaming," in *NOSSDAV '05: Proceedings of the international workshop on Network and operating systems support for digital audio and video.* New York, NY, USA: ACM, 2005, pp. 123–128.

[7] T. Tsugawa, N. Fujita, T. Hama, H. Shimonishi, and T. Murase, "TCP-AFEC: An adaptive FEC code control for end-to-end bandwidth guarantee," in *Proc. Packet Video 2007*, 2007, pp. 294–301.

[8] H. Wu, M. Claypool, and R. Kinicki, "A model for MPEG with forward error correction and TCP-friendly bandwidth," in *NOSSDAV '03: Proceedings of the 13th international workshop on Network and operating systems support for digital audio and video.* New York, NY, USA: ACM Press, 2003, pp. 122–130.

[9] I. S. Reed and G. Solomon, "Polynomial Codes Over Certain Finite Fields," *Journal of the Society for Industrial and Applied Mathematics*, vol. 8, no. 2, pp. 300–304, 1960.

[10] L. Kleinrock, *Queueing Systems, Volume 1: Theory.* John Wiley & Sons, 1975.

[11] Digital Cinema Initiatives (DCI) and The American Society of Cinematographers (ASC), "StEM mini-movie access procedure (wedding scene)," November 2004. [Online]. Available: http://www.dcimovies.com

[12] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An Architecture for Differentiated Service," RFC 2475 (Informational), Internet Engineering Task Force, Dec. 1998, updated by RFC 3260. [Online]. Available: http://www.ietf.org/rfc/rfc2475.txt

[13] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski, "Assured Forwarding PHB Group," RFC 2597 (Proposed Standard), Jun. 1999, updated by RFC 3260.

[14] B. Libæk and Øivind Kure, "Congestion Control for Scalable VBR Video with Packet Pair Assistance," in *Computer Communications and Networks, 2008. ICCCN 2008. Proceedings of 17th International Conference on*, Aug 2008.

Paper D

Generic application level rate control for scalable video using shadow probing

Bjørnar Libæk and Øivind Kure

Proceedings of the Fourth International Conference on Systems and Networks Communications (ICSNC), Porto, Portugal, September 20-25, 2009

Awarded as best paper

Abstract

A generic application level rate control (ARC) for scalable video is presented. The ARC is designed to operate between a streaming video application (server-side) and a transport level rate control (TRC). TFRC is used as a reference TRC when evaluating the proposed mechanisms. When using scalable video such as H.264/SVC, incrementation of quality layers may cause significant increase in the sending rate. The ARC utilizes a technique we name *shadow probing*, which protects the established video layers from packet loss in these situations. Also, highly variable estimates of available bandwidth reported by the TRC are hidden from the application, in order to reduce the frequency of adding and dropping layers. Simulation results are presented showing that the mechanisms are able to improve perceived quality for both interactive and streaming video in networks with and without QoS support.

1. Introduction

As video streaming is consuming an increasing amount of the total bandwidth resources on the internet, the need for video congestion control is widely acknowledged both in the research community and in the industry. One example is IETF's TFRC [1] (TCP Friendly Rate Control) which is a part of DCCP (Datagram Congestion Control Protocol) [2]. TFRC is a transport level congestion control mechanism designed for streaming media applications requiring a sending rate with less throughput variations than offered by TCP (Transmission Control Protocol). On the application level, the scalable extension [3] to the H.264/AVC [4] video codec is an important contribution, providing the ability to add and drop enhancement layers on-the-fly. This effectively makes it possible for the streaming application to adjust its sending rate based on the available network bandwidth, typically estimated at transport level by using e.g. TFRC.

Any mechanism attempting to estimate the available bandwidth based on network feedback is bound to struggle with estimation error, as well as variance in the cross traffic over multiple time scales. Unfortunately, TFRC is no exception. If the application were to add enhancement layers as soon as allowed by TFRC, the frequency of quality changes would become high. Since spatio-temporal variability may degrade user experience [5], it is often favorable to maintain a constant moderate quality instead of an alternating quality with a possibly higher average PSNR. For this reason, we emphasize the need for an *application level rate control* (ARC), operating between the streaming application and the *transport level rate control* (TRC). The ARC decides when to add and drop enhancement layers, both taking into account available network resources estimated by the TRC (e.g TFRC), and user level requirements. The TRC ensures that the sending rate does not exceed the flow's fair share. An alternative approach often taken in the literature, is a single congestion control mechanism optimized for a specific combination of networking technologies and video codecs. We believe however, that such cross layer solutions are less viable in the long run, as the technology advances both at network and application level. Instead, a generic ARC with well defined interfaces to the streaming application and the TRC is favorable in the internet.

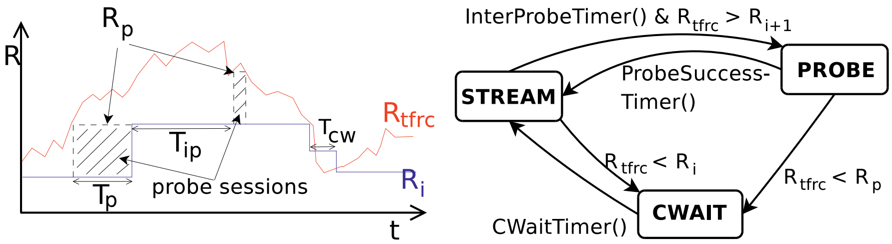
Most congestion control mechanisms use some variant of *bandwidth probing*. The available bandwidth is tested by gradually increasing the sending rate to the point where congestion events start to occur, consequently followed by a rate decrease. For multimedia flows, these inherent congestion events (e.g packet loss or increased delay) are likely to harm the perceived quality. Unfortunately, when sending scalable video over TFRC, the packet loss caused by bandwidth probing may be severe. Imagine that the application sends layers 1 through i for a longer period without experiencing congestion. During this time, TFRC's eligible rate R_{TFRC} is likely to grow well above the actual sending rate R_i, because of the absence of congestion events. Noticing all the available bandwidth, the ARC may decide to add layer $i+1$ and $i+2$ because $R_{i+2} \leq R_{TFRC}$. The significant jump in the sending rate increases the probability of exceeding the congestion threshold, which in turn causes packet loss and a steep reduction of TFRC's eligible rate. This will degrade the perceived quality both because of the information loss in layers $\leq i$, and because of the alternating quality per se.

In this paper, an ARC that meets the above challenges is outlined. The basic idea is to introduce a probing period preceding a potential incrementation of enhancement layers. During this period, a *shadow layer* is added, attempting to resemble the new layer without causing damage to the already established layers. To accomplish this, we propose two (combinable) techniques. The first is to fill the probe layer with FEC packets containing redundant information, while the other is to assign a lower drop precedence to the probe layer in networks supporting service differentiation.

First, the relevant parts of TFRC are described in more detail along with its limitations related to layered video. In section 3, the proposed ARC is described while simulation results are presented in 4.

2. TFRC and layered video

TFRC is an *equation based* congestion control mechanism targeting streaming media applications, based on the TCP throughput formula derived in [6]. This formula is used by the sender to calculate the long term average TCP throughput R_{tcp} given estimates of RTT (Round-Trip Time) and *loss event rate* taken from receiver feedback reports. Approximately once each RTT, R_{tcp} is further used as input to the algorithm where the *allowed transmit rate* R_{tfrc} is calculated. To ensure that the average sending rate does not exceed the allowed sending rate,



(a) Example **(b)** ARC State Machine

Figure 1. ARC operation

TFRC shapes[1] the outgoing packet stream. As a consequence, if the application produces data at a higher rate than R_{tfrc}, TFRC's send buffer will build up and possibly add both extra delay and packet loss.

TFRC's complete increase-decrease algorithm will not be described here, but it is necessary to explain how *data limited* periods are handled. These are defined as periods where the actual sending rate is below the allowed transmit rate, a likely situation if an ARC withholds enhancement layers due to considerations related to user perception. If no loss is detected during such a period, the TFRC sender sets the allowed transmit rate to the minimum of R_{tcp} and $2 * R_{rcv}$. The latter is the estimated incoming traffic rate at the receiver, also included in the feedback reports. Consequently, TFRC may admit a doubling of the sending rate after data limited periods.

Obviously, such significant and sudden jumps in the sending rate are likely to cause packet loss. The fact that the transport level rate estimate is "outdated" after longer periods of data limited periods is the main motivation factor for our proposed ARC. This is not restricted to TFRC, but is valid for all congestion control mechanisms relying on congestion events to estimate the available bandwidth.

3. ARC operation

The ARC decides when to add and drop enhancement layers, always making sure that the average sending rate is kept below the allowed transmit rate reported by TFRC. In addition, to be able to control the frequency of adding and dropping layers, two stabilizing periods are introduced as illustrated in figure 1a: The *inter probe interval* T_{ip} enforces a minimum time between two layer incrementations, while the *congestion wait interval* T_{cw} lets the ARC wait to observe the effect

[1]In the earlier protocol specifications, as well as the NS2 implementation, a leaky bucket (i.e peak rate) shaper was used. However, the latest RFC accepts the use of a credit based shaper to allow a higher degree of burstiness

of a layer drop before possibly removing additional layers. The simple state machine in figure 1b summarizes the operation. In the STREAM state, the established layers (layers 1 through i) are transmitted normally. The following sections describes how layers are added and dropped respectively:

3.1. Adding enhancement layers

If the ARC has been in the STREAM state for more than T_{ip} seconds, and then observes that the eligible rate may accommodate at least one new enhancement layer ($R_{tfrc} > R_{i+1}$), it initiates a *probing session* by entering the PROBE state. In this state, a *probe layer* is added, with a sending rate equal to the new layer. Note that TFRC is not able to distinguish between probe and real layers, so the eligible rate evolves as if the probe layer was real. Thus, by monitoring the eligible rate during the probing period, the ARC obtains a good indication on whether there is room for a new layer or not. If the eligible rate stays above the probing rate during the time interval T_p, the probe session is considered successful. Now, the ARC returns directly to the STREAM state after replacing the probe layer with the new layer. Otherwise, if TFRC responds with reporting an allowed transmit rate below the probe rate, the session is interrupted and the probe layer is removed. In the latter case, it is assumed that the probe layer alone caused the congestion. Thus, no excising layers are removed at this point, even though the eligible rate may have decreased below the original sending rate. As will be described in the next section, the ARC will instead wait T_{cw} seconds in the CWAIT state to see if the removal of the probe layer alone was enough.

3.2. Dropping enhancement layers

If R_{TFRC} is reported below the current average sending rate while being in the STREAM state, the ARC removes enough enhancement layers to conform to the new rate estimate. It then enters the CWAIT (congestion wait) state where it stays for T_{cw} seconds. The intention of this period is to improve stability by waiting to see if the reduction had the wanted effect. It is important that this period is not too long, because TFRC's send buffer will fill up if the ARC sending rate is larger than the eligible rate. On the other hand, the interval must be large enough to be able to receive the first feedback reflecting the rate decrease. Some higher percentile of observed RTTs is advisable, because significant queuing delay is likely in this situation..

3.3. Loss protection: Shading the probe layer

Until now, compared to pure TFRC, the described ARC operation merely improves on stability and the frequency of adding and dropping layers. However, after initiating a probing session, there is still a good chance that the amount of packet loss will increase. In the following, the primary focus is aimed at protective mechanisms that minimize the probe layer's negative effects on the excising traffic. Two different combinable solutions are proposed. When applied, the probe layer becomes a *shadow layer*.

3.3.1. FEC protection

The first proposed solution is to fill the probe layer with FEC packets containing redundant information from the excising layers. To accomplish this, we assume an (n, k) erasure-based FEC code (e.g Reed-Solomon), where the FEC encoder takes k *source symbols* (i.e the original data packets) as input and generates $n - k$ *redundant symbols* (i.e FEC packets). The resulting group of $n > k$ packets form a FEC block. The FEC decoder is able to reconstruct all k original packets as long as any k out of the n packets are received. If less than k packets make it to the destination, only the received data packets (if any) are usable.

We also assume that the encoder requires all the k source packets to be of equal length. However, this is typically not the case with scalable VBR video. To mitigate this conflict, we utilize the algorithm described in [7], where the source packets are padded with zeros in order to become equal to the size of largest source packet s_{max}, before being fed to the FEC encoder. The padding is not transmitted over the network, but all the FEC packets are of size s_{max}. The $n - k$ FEC packets from a block are interleaved with the data packets from the next block.

Usually, the FEC parameters are chosen in order to achieve a certain efficiency, trading off network bandwidth. In this case however, at the start of the probing period, both the probe rate R_p and k are given. Thus, the problem is reduced to find the smallest n satisfying:

$$\frac{S + (n - k)s_{max}}{S} \geq \frac{R_p}{R_i} \tag{1}$$

where R_i is the current rate and S is the total size of all the data packets. The numerator and denominator of the left hand side is the number of bytes before and after adding redundancy respectively. Solving for n, we get:

$$n = \lceil \frac{S(\frac{R_p}{R_i} - 1)}{s_{max}} + k \rceil \tag{2}$$

Using this technique, the sending rate is inflated while the probing flow's effective packet loss[2] during probing sessions is reduced. Existing traffic from competing flows will continue to be unprotected, but TFRC will still ensure a certain level of fairness in the long run. Remember, at this point it is not a question of whether to increase the rate or not, as this decision has already been made by the ARC based on TFRC's eligible rate.

[2] Application level packet loss after FEC recovery

It may be argued that FEC could be used all the time, not only during probe sessions. However, as shown in [8], if a majority of the flows adds FEC to their packet stream at the same time without lowering the video quality, there is a significant probability that the added overhead causes a growth in the channel packet loss ratio large enough to actually increase the effective packet loss at high traffic loads. It was also demonstrated that this probability diminishes as the relative amount of FEC overhead is reduced. These observations substantiate the idea of limiting FEC to probing periods only. This way, the relative amount of FEC overhead is kept at a lower level, both because the probing sessions are time limited and because not all flows are likely to be in the probing phase at the same time.

The block length k is a configuration parameter, which must be chosen wisely[3]. A large k will result in a constant addition to the delay seen from the application. On the other hand, a small k relative to the router buffer size will increase the risk of burst losses where both data and FEC packets from the same block are dropped. However, as indicated by the simulations presented later, the effective packet loss when using FEC is not highly sensitive with respect to the choice of k. Consequently, significant reduction in packet loss ratio may be achievable while still meeting the delay requirements of e.g. interactive applications.

3.3.2. DiffServ protection

A clear advantage of the FEC solution is that it can be used in best effort networks. However, only the probing flow's own layers are protected, while competing traffic may suffer from the sudden reduction in available bandwidth. The next technique addresses this shortcoming by taking advantage of priority mechanisms if available. For instance, the DiffServ Assured Forwarding (AF) architecture [9] provides three drop precedence levels within each AF class. Depending on their relative importance, packets from a single flow are then assigned different drop precedence levels encoded in the DSCP (DiffServ Code Point) field of the IP header. The DiffServ enabled routers will then drop higher priority packets with lower probability at the cost of lower priority packets.

The combination of layered video and network service differentiation has been frequently addressed in the literature. Almost without exceptions, contributions in this area are concerned with optimizing the mapping of video layers to the available priority levels. Usually, the goal is to maximize average PSNR. An obvious challenge is that only a few priority levels are available (DiffServ AF has three), while in modern multi-dimensional scalable video codecs such as H.264/SVC, the number of enhancement layers may be much higher. An optimal mapping scheme for one combination of parameters such as video codec, encoder configuration and video content, may certainly not continue to be optimal if one or more of these parameters are changed. Another problem is that all though

[3] In this work, k is configuration parameter. However, an adaptive approach may be possible

these static mapping schemes may be able to improve average PSNR, they cannot be used to control the variability.

Instead, we simply propose to assign high priority to the existing layers and low priority to the shadow layer. This simple scheme has two obvious advantages compared to traditional mapping. First, combined with the ARC described earlier, the quality variations are reduced because packets from the shadow layer are dropped before other packets, interrupting the probing periods earlier. Second, the priority mapping is independent of the relative importance of the different enhancement layers. Thus, the mapping can be used for any scalable coding scheme.

Most important in this context however, is that this priority mapping will protect the existing traffic from quality degradations caused by packet loss during probing sessions. Also, compared to the FEC technique, this method will protect *all* flows, not only the probing flow itself.

This scheme only utilizes two priority levels, but as in DiffServ AF, additional levels may be available. It may for instance be favorable to take advantage of a third level by assigning the highest priority to the base layer of the video. Especially, if rate variations within enhancement layers (due to the video content) is large compared to variations caused by adding and dropping layers, extra protection for the base layer should be considered. However, the priority mechanism's ability to differentiate decreases quickly with the number of priority levels in use, so it is not obvious that the third level should be utilized. In the simulations presented later, only two levels are used.

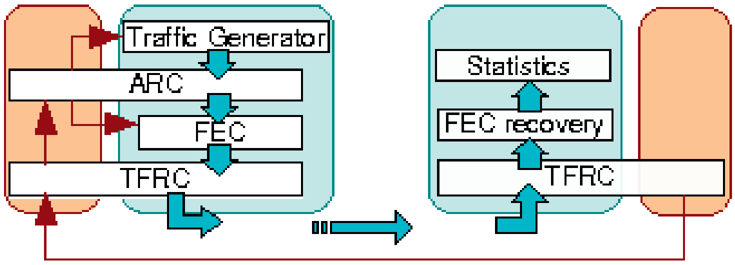
3.3.3. Combination

Even though the two schemes are independent, they can be combined. By simply assigning low priority to all FEC packets, all competing traffic is protected from the shadow layer. In addition, the probing flows have the possibility to recover lost packets from established layers. This is particularly useful when DiffServ capabilities are available on a subset of the routers on the path only.

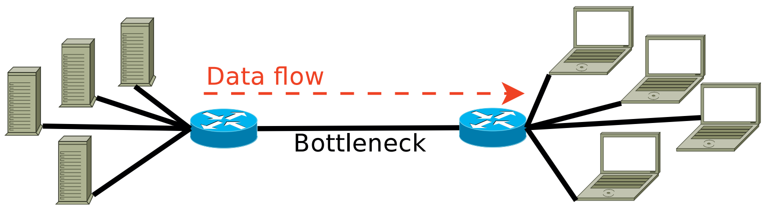
Note that the efficiency of FEC when sent with lower priority may not be as high as when sent with normal priority. Given that high priority data packets are dropped, it is likely that lower priority FEC packets from the same block are also dropped if the block length is small. For this reason, when using the combined technique, a larger k may be needed in order to lower the effective packet loss ratio enough to be able to defend the added delay and complexity associated with FEC.

4. Simulations

To evaluate the described mechanisms, an NS2 [10] simulation model was created. Figure 2a shows the components of the source and destination nodes, while the network layout is illustrated figure 2b. Four video flows share a bottleneck link



(a) End system components



(b) Network topology

Figure 2. Simulator overview

with a capacity of approximately 4.5Mbps. The scenario mimics a situation where the traffic volume is dominated by video flows, which are all governed by the same control method (ARC+TRC). Influence on and by competing TCP/UDP flows is out of the scope of this paper, so cross traffic is not included. In each simulation, all video sources start at the beginning, and are not stopped until the end of the run. The one-way propagation delay from source to receiver is 15ms.

4.1. Simulator components

4.1.1. TFRC

In order to use NS2's TFRC agent, some modifications to the API were needed. First, functionality was added in order for the application to be able to send actual packets to TFRC, not only specify the number of bytes to send. This also implied implementing a send buffer inside the TFRC agent. Second the application (ARC) was given access to both the current allowed transmit rate R_{tfrc}, as well as the status of the send buffer.

4.1.2. ARC

The ARC component includes all the functionality related to shadow probing as described in section 3. The module can be configured to operate with or without

DiffServ marking, and with or without FEC generation. When operating without FEC, the probe layer instead consists of the new video layer. When DiffServ marking is turned off, all packets are treated equally by the bottleneck router.

4.1.3. FEC

This module was implemented according to the assumptions taken in section 3.3.1. At the start of each probing interval, equation 2 is used to find the appropriate number of FEC packets. The FEC packets from block i is interleaved with the data packets in block $i+1$. The FEC decoder counts the number of received packets belonging to a block, and reproduces lost data packets if this number is k or more.

4.1.4. DiffServ

As described in section 3.3.2, packet are marked as either high or low priority. The bottleneck router uses a simple dual tail-drop queue where low priority packets are dropped if arriving to a queue with more than L packets. This simple dropping algorithm is available in most router implementations. The threshold L is set according to the rule: $L = \lceil \delta * B \rceil$ where B is the total buffer size and δ is the relative amount of buffer space reserved for high priority traffic.

4.1.5. Traffic generators

Two different traffic generators where used to resemble scalable video. Primarily, a Poisson based generator where used, but a trace based generator where used to validate the results. The Poisson generator produces a packet stream with negative exponentially distributed inter-arrival times and packet sizes. Mean packet size was set to 512 bytes, while the mean inter-arrival time was chosen to match the target sending rate for each video layer. The sending rate for layer i is $i * 100kbps$, $i = 1, 2, ...$, giving a relatively fine granularity. This generator is a strong simplification but provides a sending rate with a flexible mean and high variability. The latter is a challenge for TFRC. More realistic video sources typically have a smoother sending rate, but with more correlation at GOP and frame time scales.

The trace based generator reads video frames from a H.264/SVC encoded trace file, containing information such as frame size and layer association. The frames are packetized and sent to the ARC. To avoid synchronization, each source only reads a random number of GOPs (Group Of Pictures) from the file before skipping another random number of GOPs. This pattern is repeated during the simulation. When the end of the trace file is reached, the generator starts reading from the beginning of the file. Due to the lack of flexibility, this generator was only used in order to validate the results from the Poisson generator.

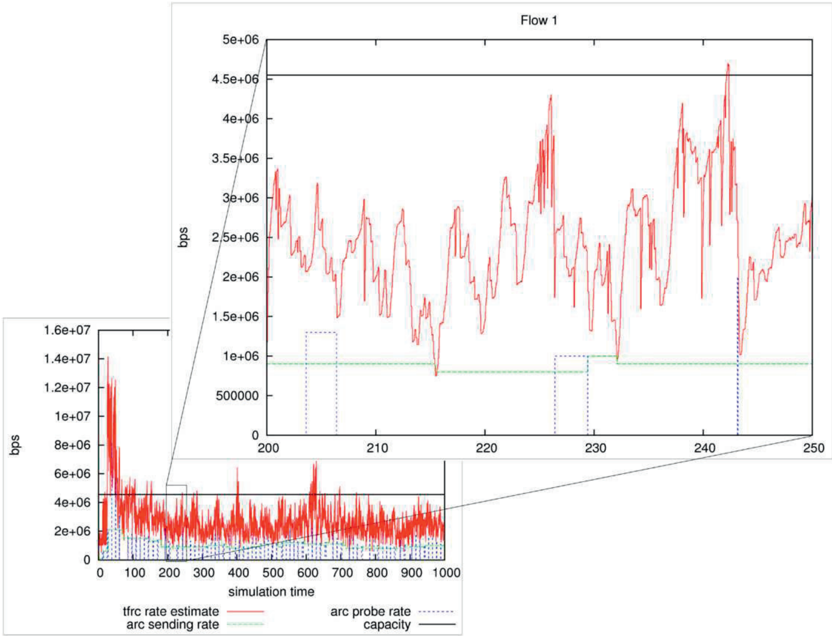
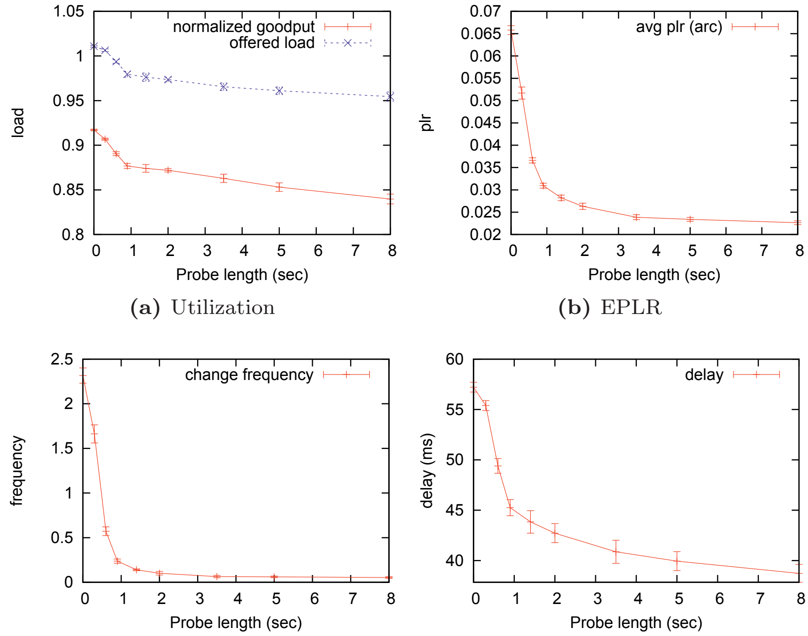


Figure 3. Simulation run example

4.2. Output parameters

The following output parameters were studied in the simulation experiments:
EPLR (Effective Packet Loss Ratio): The loss ratio observed at application level, i.e. after FEC recovery.
LDF (Layer Dropping Frequency): The number of quality reductions per second. If more than one layer is dropped in one reduction event, all of them are counted. The frequency of adding layers is of course strongly correlated, but we concider the LDF to be significantly more harmful.
One-way delay: The latency observed at application level. This includes delay in the FEC encoder (not modeled), queuing in TFRC's send buffer, network delay (propagation/transmission/queuing) and latency added by the FEC decoder. Playout-buffering at application level is not included.
Normalized goodput: The application level reception rate divided by bottleneck capacity. Dropped packets and packets belonging to the probe layer are not included.



(a) Utilization (b) EPLR

(c) LDF (d) One-way delay

Figure 4. Experiment 2

4.3. Experiment 1: rate dynamics

Figure 3 shows how one of the four video sources adapts to the allowed transmit rate reported by TFRC. At about 205s and 243s, there are examples of unsuccessful probing periods, while a new video layer is successfully added around 230s. The graph also shows how increase in the cross traffic forces the ARC to drop video layers at approximately 216s and 232s. In addition to verifying the ARC operation, this graph clearly demonstrates the highly variable TFRC rate, emphasizing the need for an ARC. Even though all four sources are active during the simulation, TFRC reports as much as all capacity being available at several occasions. In spite of this, the ARC succeeds in keeping the average sending rate close to one fourth of the capacity, while the frequency of adding and dropping layers is low.

4.4. Experiment 2: Probe length sensitivity

The intention of this experiment is to study how the choice of T_p (length of the probing interval) affects different performance parameters. Here, both protection mechanisms where turned off and the bottleneck buffer size was set to 32kB. For each value of T_p, 10 simulations with different seed in the traffic generators where run, each with a duration of 5000 simulated seconds. The sample means over the 10 simulations are plotted in figure 4, enclosed by the 99% confidence intervals. For the special case $T_p = 0$, the new enhancement layer is added without probing. As expected, the consequence is a high LDF (more than twice each second!), and also relatively high packet loss and delay. When increasing the probing period, significant improvements are achieved, at the expense of the goodput. As an example, when using a 3s probing period and giving away approximately 5.5% of the goodput, we get a 26% reduction in one-way delay, 62% reduction in packet loss ratio and as much as a 95% reduction in the LDF. The graphs show a decreasing sensitivity as the probe interval increases. This is explained by the fact that TFRC needs a certain time before the periodic feedback packets report a congestion level close to the real value (due to smoothing of the samples). After waiting this long, the ARC can with a high probability correctly accept the new layer, and there is little gain in waiting further. However, there is a large penalty for accepting a layer too early, as both packet loss and delay soon will force the ARC to drop the layer.

4.5. Experiment 3: Loss protection

In this experiment, the effects of the two protection mechanisms where studied. Three probe layer configurations where simulated, each with and without the use of network priority support, giving a total of six combinations of protection mechanisms. The three configurations are FEC with $k = 8$, FEC with $k = 40$ and no FEC. In the latter case, the probe layer was filled with the new video layer as in experiment 2. Further, each of the six combinations where simulated with seven different bottleneck queue sizes. The probe length was set to 3 seconds, and the δ parameter in the simulations where DiffServ was in use was set to 0.4. All points in the graphs are surrounded by the 99% confidence interval resulting from 40 simulation runs.

Figure 5 summarizes the results. The EPLR during probing sessions, as experienced by the probing flow, is averaged over all flows and plotted in 5a. Figure 5b plots the total EPLR averaged over all flows during the entire simulation period. In 5c, the one-way delay measured at application level is plotted, while figure 5d shows the standard deviation of TFRC's allowed transmit rate averaged over all flows. In the following, the most important observations are discussed:

4.5.1. Observation 1

In the best effort simulations (labeled "nopri"), the introduction of FEC in the probe layer results in significantly lower EPLR during the probing sessions. A large part of the dropped data packets are recovered. This also contributes to a lower total EPLR, especially for larger buffer sizes.

4.5.2. Observation 2

When applying priority protection, both total EPLR and EPLR during probing sessions are reduced even further, for all three configurations (newlayer, k=8 and k=40). In fact, for the larger buffer sizes, application level packet loss is almost eliminated without trading off utilization (figure 5f). When not using priority however, the offered load and thus the normalized goodput decreases with larger buffer size. The reason is that when the RTT grows, the unsuccessful probing periods will last longer, and consequently cause more packet loss for the competing flows. This will in turn force a larger number of them to drop layers. When using priority, most of the packet loss is limited to the probing flow itself, and the competing flows are unaffected.

4.5.3. Observation 3

When none of the protection mechanisms are applied, EPLR during probing is in fact increasing with buffer size (newlayer+nopri). This contradicts with traditional queuing theory where larger queues normally results in lower packet drop probabilities because larger bursts can be handled. This phenomenon is also present after adding FEC. However, when using priority protection, a more "normal" behavior is observed.

To explain this, we need to take into account both the ARC and TFRC behavior. When the buffer size is larger, the variance in delay (jitter) will cause more variance in the TFRC transmit rate as shown in figure 5d. In the situations where the TFRC rate is at low extremes, no probing sessions will be started, and thus will not contribute the EPLR during probing. At the high extremes however, probing sessions will be started, and the probing rate is now more likely to be above the congestion threshold. In the best effort situation, the increased buffer size is not large enough to compensate for the "extreme" probing rate. When using the priority queue however, the rate of the established layers are kept constant while the reserved buffer space for high priority packets increases. This demonstrates why priority on the shadow layer is an effective way to limit the damage of a too aggressive rate control.

4.5.4. Observation 4

Figure 5c confirms that the average one-way delay is mainly affected by the buffer size and the FEC block length. An important question is whether the FEC technique can be used by interactive applications with strict delay requirements, typically in the order of 100ms-150ms. Such applications must either be used in high capacity networks where congestion control unnecessary, or the network must have QoS mechanisms that offer delay guarantees. If the latter is the case, it is also likely that priority support is available. Then, based on the simulations results, the large reduction in EPLR when using both FEC and priority protection can indeed defend the added delay. For one-way streaming applications on the other hand, the jitter is usually a more critical factor than the average delay. When using FEC, the added delay is constant for a given k, and is therefore not contributing to jitter which is strongly dominated by the variance in queuing delay.

4.5.5. Observation 5

Figure 5a confirms that a larger k is needed when combining FEC and priority protection as discussed in 3.3.1. In fact, the effective packet loss is higher when using $k = 8$ compared to using the new video layer. The reason is that the FEC encoder adds correlation to the arrival process, while the new layer is added by simply increasing the intensity of the Poisson generator. When using a small k, the FEC packets are more likely to be dropped in the same loss event as the data packets in the same block, especially since they have lower priority. Remember that this is merely an issue when every router on the path has priority support. When this is not the case, FEC will provide extra protection if some of the best effort routers are congested. Note also that the large improvement in total EPLR when adding FEC is mostly not caused by FEC recovery, but by the fact that the probe periods fail faster because the correlation causes TFRC to observe more packet loss.

4.6. Experiment 4: Realistic arrival process

In this experiment, an H.264/SVC encoded video trace was used instead of the Poisson generator. Figure 6 presents a subset of the results showing similar behavior as in experiment 3. To a large extent, the observations still hold, with the most notable difference being that EPLR during probing starts to decrease when buffer size is above 16Kb in the non-priority cases. Packet loss is in general lower due to the fact that the layers are more course grained, lowering the offered traffic. Nevertheless, the two protection mechanisms both show improved results also here.

5. Conclusions and further work

When scalable video is streamed over IP networks, the estimated available bandwidth reported from the transport level is usually too variable to be used as the sole control mechanism when decisions regarding adding and dropping of enhancement layers are to be made. For this reason, the need for an application level rate control (ARC) mechanism for scalable video has been emphasized. An ARC has been proposed, designed to operate between a streaming video application (server-side) and a transport level rate control (TRC). The principles are not bound to specific video codecs or transport protocols, but TFRC was chosen as the reference TRC when evaluating the proposed mechanism. The ARC's main jobs are to filter out the high frequent changes in the estimated available bandwidth reported by the TRC, and to reduce the damage when the reports are too optimistic. This is done using a technique named *shadow probing*, which utilizes FEC and/or network differentiation to protect the established video layers when the sending rate is drastically increased. The simulation results have shown that the described mechanisms are able to improve perceived quality for both interactive and streaming video in networks with and without QoS support. Unavoidably, this is achieved at the price of a lower utilization.

Further work includes evaluating the mechanisms using alternative transport protocols and router drop algorithms, as well as looking at specific FEC schemes. More simulations with realistic video traces should be done in order to study the performance for correlated packet streams. Also, larger scale experiments should be conducted, with paths consisting of both best effort and priority enabled routers as well as TCP and UDP cross traffic.

6. References

[1] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "TCP friendly rate control (TFRC): Protocol Specification. IETF RFC 5348," Sep 2008.

[2] E. Kohler, M. Handley, and S. Floyd, "Datagram Congestion Control Protocol (DCCP), RFC 4340," March 2006.

[3] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable h.264/mpeg4-avc extension," in *Image Processing, 2006 IEEE International Conference on*, 2006, pp. 161–164.

[4] ITU-T and ISO/IEC, "Advanced Video Coding for Generic Audiovisual Services," ITU-T Recommendation H.264 - ISO/IEC 14496-10 (AVC), November 2007.

[5] D. S. Hands and K. Cheng, "Subjective responses to constant and variable quality video," in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, ser. Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, vol. 6806, Mar. 2008.

[6] J. Padhye, V. Firoiu, D. F. Towsley, and J. F. Kurose, "Modeling tcp reno performance: A simple model and its empirical validation," *IEEE/ACM Transactions on Networking*, vol. 8, no. 2, pp. 133–145, April 2000.

[7] M. Luby, L. Vicisano, J. Gemmell, L. Rizzo, M. Handley, and J. Crowcroft, "The Use of Forward Error Correction (FEC) in Reliable Multicast," RFC 3453 (Informational), Dec. 2002. [Online]. Available: http://www.ietf.org/rfc/rfc3453.txt

[8] B. Libæk and Øivind Kure, "Protecting scalable video flows from congestion loss," in *Networking and Services, 2009. ICNS 2009. Proceedings of 5th International Conference on*, Apr 2009.

[9] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski, "Assured Forwarding PHB Group," RFC 2597 (Proposed Standard), Jun. 1999, updated by RFC 3260.

[10] "The Network Simulator NS-2, v2.33," http://www.isi.edu/nsnam/ns/.

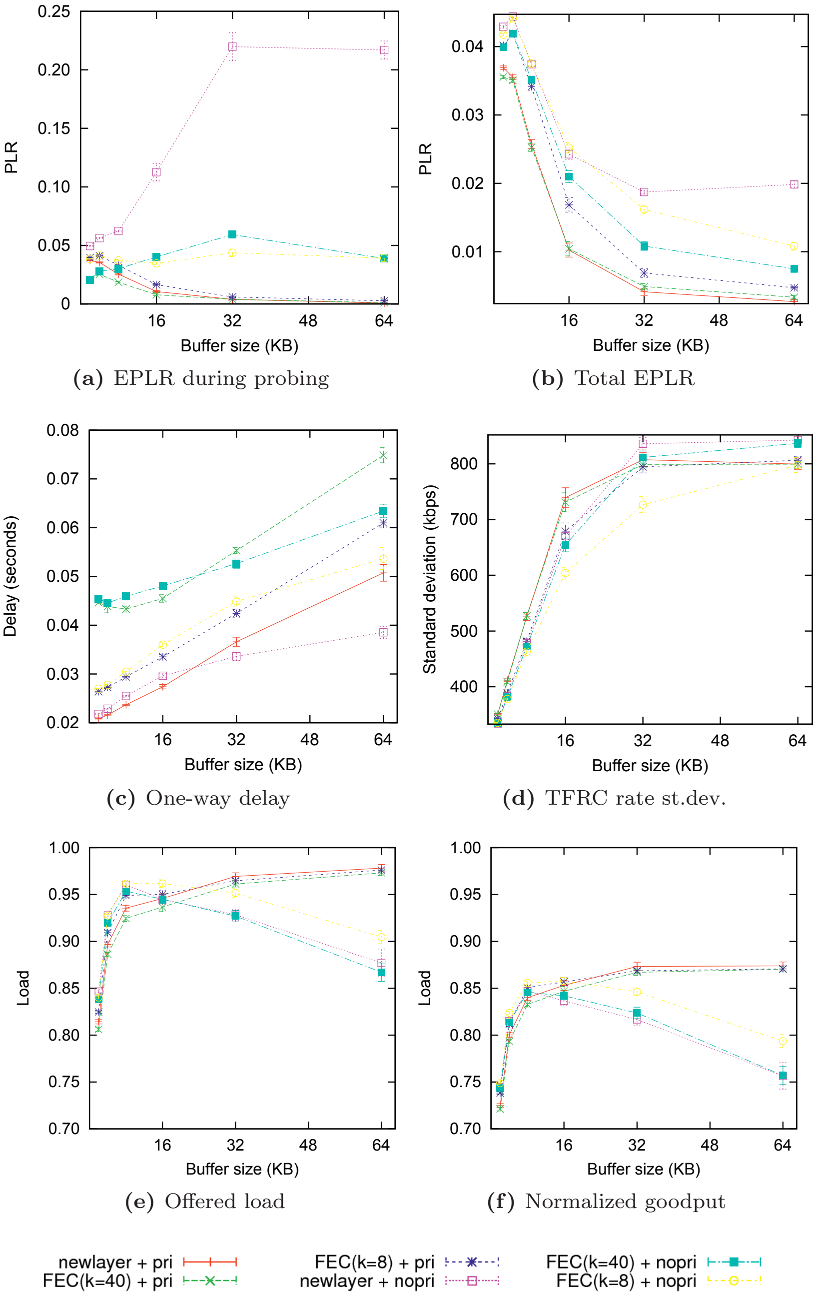
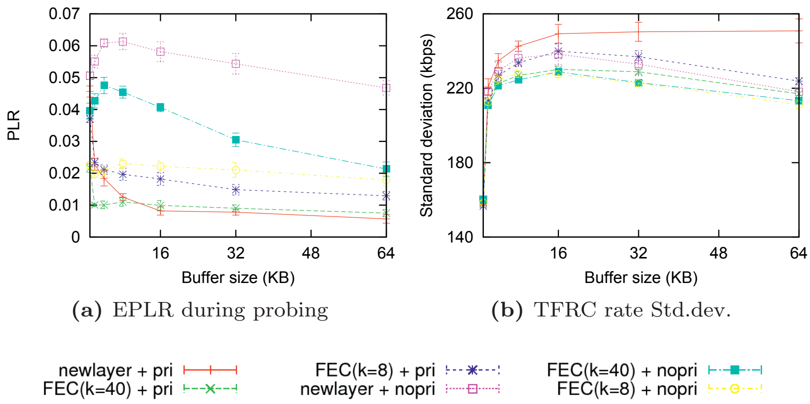


Figure 5. Experiment 3



(a) EPLR during probing **(b)** TFRC rate Std.dev.

Figure 6. Experiment 4

Appendix A

Expectation of the Nguyen estimator

Let

$$\hat{A}_1 = \frac{1}{\tau'} \sum_{i=1}^{N-1} s_i \tag{1}$$

be an estimator for the available bandwidth, given that N packets of size s in a train sent back-to-back and the interval between the arrival of the first and the last packet is τ' measured at the receiver side. Now consider the expectation of $\hat{A}_1$. It is assumed that none of the probe packets are lost, so the number of bytes received during $\tau^{'}$ is constant and equal to $(n-1) * s$. Based on the renewal-reward argument expressed later, we have that $E[1/\tau^{'}] \approx 1/E[\tau^{'}]$, and further that:

$$E\left[\hat{A}_1\right] = \frac{E\left[\sum_{i=1}^{n-1} s_i\right]}{E\left[\tau'\right]} = \frac{(n-1)\,s}{E\left[\tau'\right]} \tag{2}$$

The time interval $\tau^{'}$ consists of the time the bottleneck router spent on sending $n-1$ probe packets and M competing packets arriving at the bottleneck during the time interval τ, e.g simultaneously with our packet train. Then,

$$E\left[\tau^{'}\right] = (n-1)\,\frac{s}{C_b} + E\left[M\right]\frac{E\left[S_c\right]}{C_b} \tag{3}$$

where S_c is the packet size of the competing cross traffic, and

$$E\left[M\right] = \lambda_c \tau = \frac{R_c}{E\left[S_c\right]}\,(n-1)\,\frac{s}{R_a}$$

where $\lambda_c = \frac{R_c}{E[S_c]}$ is the long term arrival rate of the competing packets, R_a is the train's arrival rate at the bottleneck router and $\frac{s}{R_a}$ is the time between the arrival of to consecutive probe packets at the bottleneck router. Putting the pieces together and simplifying the expression, we have the following:

$$E\left[\hat{A}_1\right] = \frac{C_b}{1 + \frac{R_c}{R_a}} \tag{4}$$

In the first step in equation 2, it was assumed that $E[1/\tau^{'}] \approx 1/E[\tau^{'}]$. This can be justified by regarding the departure of the packet train from the router as a renewal-reward process. If we assume that the *idle periods* (i.e the time between each packet train) are long enough to let the queue be empty at least once, the output intervals $\tau_i^{'}$ will be independent and equally distributed (i.i.d.). In general, the necessary length of the idle periods should exceed an average busy period in the queue, but this depends on the distribution of the cross traffic. The result is a renewal process Z_t, which denotes the number of finished output intervals at time t.

Now, let B_i be the number of bytes (i.e. reward) received in time interval $\tau_i^{'}$ $(i = 1, 2, ...)$. Now, because $B_i = ns$, $i = 1, 2, ...$, the random variables B_i are also i.i.d, with $E[B_i] < \infty$. Thus, the total number of bytes received after a time t can be expressed as the renewal-reward process: $S_t = \sum_{i=1}^{Z_t} B_i$. From the results of renewal theory, it follows that

$$\lim_{t \to \infty} \frac{1}{t} S_t = \frac{1}{E[\tau^{'}]} E[B]$$

Because we have removed the idle periods from t, fortunately, this expression equals the expected arrival rate during the output intervals. This shows that $E[\hat{A}_1] = E[B]/E[\tau^{'}]$, which justifies equation 2.