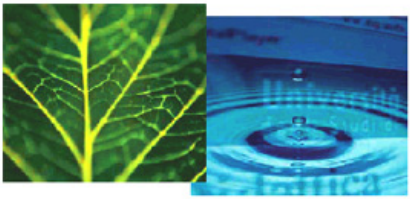
PhD Dissertation



International Doctorate School in Information and Communication Technologies

DIT - University of Trento

CROSS-LAYER PERFORMANCE OPTIMIZATION IN WIRELESS LOCAL AREA NETWORKS

Dzmitry Kliazovich

Advisor:

Prof. Fabrizio Granelli

University of Trento

Co-advisor:

Prof. Mario Gerla

University of California, Los Angeles

December 2006

Abstract

Cross-Layer Performance Optimization in Wireless Local Area Networks

by

Dzmitry Kliazovich

Doctor of Philosophy

Department of Information and Telecommunication Technology

University of Trento, Italy

The TCP/IP protocol suite provides the set of protocols which is the de facto standard for communications in Internet today. Designed in early days of ARPANET for traditional wired networks TCP/IP shows poor performance in wireless domain. This performance degradation is mainly due to limitations of wireless medium in terms of bandwidth, latency, information loss, and mobility.

Traditionally, the proposals for TCP/IP performance improvement target the compensation of undesirable characteristics of wireless medium or the introduction of network-aware algorithms within different layers of the protocol stack. Most of these proposed solutions optimize a single layer at a time.

In this research, we investigate the application of joint inter-layer optimization labeled as "Cross-Layer Design" for TCP/IP performance improvement in wireless networks.

Following the extensive study of performance bounds and limitations of wireless networks we designed two cross-layer optimization schemes for joined optimization of ARQ schemes operating at different layers of the protocol stack as well as for cross-layer optimization of congestion control algorithms.

The adverse effects of wireless links require an implementation of ARQ mechanisms at the link layer. On the other hand, higher layer protocols like TCP include ARQ to ensure reliability of communications. As a result, on some of the links the functionalities of ARQ implemented at different layers overlap leading to the unnecessary demand for bandwidth resources. The proposed Cross-Layer ARQ reduces the overhead associated with ARQ process implemented at higher layer based on the link layer ARQ feedback. More exact, it generates TCP acknowledgement based on the successful packet delivery indication received from the link layer. This releases the network capacity resources required for the transmission of TCP acknowledgement making them available to network nodes for a payload data transmission. This mechanism has been demonstrated to provide the performance and corresponding capacity improvements between 20% and 100% across the range of scenarios evaluated through simulations as well as IEEE 802.11 testbed experiments. Among the factors contributing to the performance enhancement of the proposed Cross-Layer ARQ are: medium busy time reduction, reduced sensibility to link errors, reduced Round Trip Time (RTT), and improved congestion control.

The core of TCP congestion control functionality relies on packet loss as indication of network congestion. However, in wireless networks a large portion of dropped packets is caused by link errors. With the aim of performance improvement the proposed approach, called Cross-Layer Congestion Control, exploits the network capacity information in term of available bandwidth and delay measured at the link layer instead of transport layer. The method requires the introduction of an additional module

within the protocol stack of the mobile node, able to adjust the outgoing data stream based on capacity measurements.

Today, the number of wireless users is several times higher than the number of internet users. On the other hand, most of the internet users operate using fixed connections. We believe this difference is mainly motivated by poor performance of data communications over the variety of wireless mediums. Moreover, the traditional way of performance improvement which is restricted to a particular protocol layer seems to be not sufficient. Thus, novel solutions which break layering principles allowing interdependence and joined protocol design will dominate in wireless network optimization. We believe the solutions presented in this research thesis can become the first step in the tremendously growing field of cross-layer design.

To my parents, Sergey and Natalya,

and sister Elena

Contents

List of Tables

List of Figures

Acknowledgements

It has been more than three years since I joined the doctorate school at Trento, and when looking back at the years spent here I realize that this dissertation contains only a small piece of what I have gained during this time. Most of all I am grateful to the people I met who influenced me during these years and opened new horizons for me.

First of all, I would like to express the outmost gratitude to my supervisor, Prof. Fabrizio Granelli. His invaluable support by all means, guidance, and sponsorship allowed me to fully concentrate on the research. Without his guidance and patience my PhD research would not be accomplished.

I am grateful to my co-advisor Prof. Mario Gerla and his group for hosting and supervising me in Network Research lab at the Computer Science department at the University of California, Los Angeles. I just hope to inherit at least some of Mario's skills. It was a great and extremely productive time.

I learned a lot from Prof. Nelson Fonseca and Prof. Michael Devetsikiotis. Their experience in the field of wireless networks and cross-layer design helped me to enrich my own knowledge in this very sphere. They opened for me insights of the research community, its organization, and its past and future trends.

I owe gratitude to Prof. Imrich Chlamtac whom I had met even before the time CREATE-NET was created. He opened the world of optical networking for me. However, besides team-work I am also grateful for giving me some tips and guidelines (sometimes indirectly) on business principles lying behind scientific research.

I would like to thank Daniele Miorandi and Hagen Woesner for encouraging our collaboration, for teaching me so many things in the field of networking during our numerous brainstorming discussions.

I must thank Ermanno Cavalli, Simone Redana, and Nicola Riato for taking me for an intern at Siemens where I had a chance to investigate 3G Long-Term Evolution (LTE) networking destined to come into cellular communications market.

Last, but not least, I want to express my sincere gratitude to my family for they love, support, guidance, and for bringing me up to what I am now.

The secret of a good sermon is to have a good beginning and a good ending, then having the two as close together as possible.

- George Burns

Chapter 1

1. Introduction and Preview

1.1. Motivation

There are two most outstanding trends in networking evidenced during the last decade: 1) tremendous growth of Internet, and 2) evolution of network access towards wireless technologies.

In 1990s, Internet growth was mainly motivated by the extremely successful bubble of World Wide Web (WWW). At that moment, Internet was mainly associated with a huge data storage resource containing millions of sites and billions of web pages.

However nowadays we are at the edge when it becomes deeply integrated into our physical environment. Currently it brings dramatic changes into news and publishing worlds, creates new approaches for search technologies, advertising, peer-to-peer networks and new approaches for personal and community communication.

The second revolutionary trend brings wireless devices to the access edge of the network. The wireless era that originally

started for voice communications shifts towards data communications providing an access any time, anywhere.

Nowadays, the number of wireless subscribers (mainly in cellular 2G, 2.5G and 3G networks) present in the market is two times more than the number of Internet users (see Figure 1.1). However, most of the Internet users today use wired connections leaving a primary focus of wireless cellular users on voice communication.

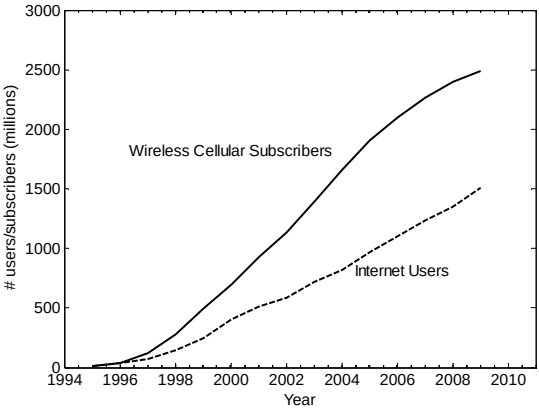


Figure 1.1: Growing trends in wireless access and Internet users. Sources: Ericsson, Inc. and Internet World Stats (www.internetworldstats.com)

One of the reasons stopping convergence lies in the lack of proper services designed specifically for wireless mobile users as well as an absence of proper terminal equipment. The later is currently being aggressively pushed by the main vendors such as Intel, Motorola and Nokia [1]. While the former, adaptation of services, is limited due to poor data transfer performance in wireless environment.

The TCP/IP protocol suite provides the set of protocols which is the de facto standard for communications in Internet today. Designed in early days of ARPANET for traditional wired networks TCP/IP shows poor performance in wireless network environment [2-4]. Among the main reasons of this performance

degradation are limitations of wireless medium in terms of bandwidth, latency, information loss, and mobility [5].

1.2. Challenges

Various approaches have been proposed to optimize TCP/IP performance in wireless networks. Most of them target at either a compensation of undesirable characteristics of wireless medium making it look like a wired one, or an introduction of direct modifications of higher layer protocols like TCP making them network-aware. In fact, just a few of the proposed solutions were adapted in final products mainly due to the low tradeoff between performance improvement and implementation complexity.

However, during the last several years an alternative approach for the design is gaining interest, namely Cross-Layer design [6, 7]. It overcomes layering principles employed in network architectures and protocol stacks allowing joint interlayer optimization. As a result, cross-layer design allows better performance optimization in such challenging scenarios as wireless and heterogeneous networks.

In this research thesis we first identify the performance bounds and limitations present in wireless networks and then design the cross-layer solutions targeting at two major problems causing performance degradation:

- *Wireless channel losses*: The loss probability experienced by packet transmission is in general higher on the wireless medium rather than on wired links: while Bit Error Rate (BER) varies from 10^{-8} to 10^{-6} for wired channels, it varies from 10^{-3} up to 10^{-1} for wireless channels [8]. Such error rates are unacceptable for the TCP [3], designed for wired networks.

In order to counteract such variation of BER, Automatic Repeat reQuest (ARQ) is employed at the link layer of the protocol stack. It requires the receiver positively acknowledge every successfully received data frame. On the other hand, TCP employs another ARQ at the transport layer to ensure reliability. Due to the overlap of ARQ functionalities at different layers several acknowledgements are generated for a single data block transmitted

in wireless channels. These acknowledgements correspond to the overhead reducing available bandwidth resources.

The optimization of the acknowledgement scheme will obviously bring performance improvement through the reduction of the medium-busy time and would require interaction between the transport and link layers – thus requiring proper cross-layering schemes.

- *Congestion control*: The congestion control implemented in TCP is probably the most important algorithm defining overall network performance. It controls the outgoing traffic rate with the purpose to keep it at a maximum network utilization level however avoiding network overload and further congestion collapse. Ideally, the outgoing rate should be equal to the bandwidth available on the path between the sender and the receiver.

However, in TCP/IP reference model no layer has complete and real-time information about available network resources over the multi-hop path where the communication is performed. For that reason, TCP congestion control functionality relies on packet loss as an indication of network congestion. However, in wireless networks a large portion of dropped packets is the result of link errors decreasing TCP throughput and error recovery performance.

Network performance can be optimized making TCP sender aware of the reason a certain loss occurred or to provide it with the knowledge of the available network resources. This would require interaction between physical, link, and transport layers of the protocol stack.

1.3. Contributions and Thesis Structure

In this thesis, a number of problems are introduced and appropriate solutions are designed using cross-layering approach. The performed evaluation and comparison with existing solutions limited in optimization to a particular layer show tremendous advantages enabled by cross-layer design. The major contribution of this thesis can be summarized in two parts:

- **Cross-Layer ARQ:** this scheme enhances the protocol stack of the wireless sender and receiver with cross-layer ARQ agents which enable collaboration between the link and transport layers. The ARQ agent generates TCP acknowledgement at the sender side locally as soon as the link layer confirms successful packet delivery. As a result, Cross-Layer ARQ approach avoids the transmission of TCP ACK packets over the wireless channel. The saved time can be used by other nodes for data packet delivery which increases overall network capacity.
- **Cross-Layer Congestion Control**: This scheme enhances TCP congestion control over wireless networks by providing the transport layer with end-to-end link capacity measurements performed at the link and physical layers. The method requires an addition of the software module into the protocol stack of the mobile node which is able to adjust outgoing data stream based on the obtained capacity measurements.

The rest of the thesis is organized as follows: Chapter 2 examines performance issues that arise in wireless networks defining performance bounds and limitations. Then, it presents an overview and comparison of the existing optimization solutions considering large variety of possible network deployment scenarios.

Chapter 3 addresses ARQ issues over wireless networks and introduces Cross-Layer ARQ solution designed for performance enhancements obtained from the introduced feedback between the link and the transport layers of the protocol stack.

In Chapter 4, we describe a cross-layer optimization of TCP congestion control over wireless networks.

Finally, in Chapter 5, we present a summary of the research work drawing conclusions and outlining directions for future research in the field.

Study the past if you would define the future.

- Confucius

Those who cannot remember the past are condemned to repeat it.

- George Santayana, The Life of Reason, 1905

Chapter 2

2. State of the Art

2.1. Introduction

Wireless networks are becoming increasingly popular in telecommunications, especially for the provisioning of mobile access to wired network services. As a consequence, efforts have been devoted to the provisioning of reliable data delivery for a wide variety of applications over different wireless infrastructures. In wireless network, regardless of the location, users can access services available to wired-network users.

In this scenario, the IEEE 802.11 standards represent a significant milestone in the provisioning of network connectivity for mobile users. However, the 802.11 medium access control strategy and physical variability of the transmission medium leads to limitations in terms of bandwidth, latency, information loss, and mobility. Moreover, the deployment of the Transmission Control Protocol (TCP) over IEEE 802.11 networks is constrained by the low reliability of the channel, node mobility and long Round Trip Times (RTTs).

This chapter aims at providing a comprehensive analysis of the performance limitations and potential enhancements to 802.11 networks. Proposals to overcome such limitations are compared and their suitability for specific deployment scenarios is presented.

The structure of this chapter is as following: Section 2.2 provides an overview of the IEEE 802.11 standards and its extensions. Section 2.3 surveys the performance issues related to throughput and delay in 802.11 networks. Section 2.4 introduces existing proposals to overcome those problems. Sections 2.5 and 2.6 provide comparisons of the different solutions. Finally, Section 2.7 draws some conclusions.

2.2. The 802.11 Standards

The IEEE 802.11 Wireless Local Area Network (WLAN) standard was first adopted in 1997 and revised in 1999 [9]. It aims at “providing wireless connectivity to automatic machinery, equipment or stations that require rapid deployment, which may be portable or hand-held, or which may be mounted on moving vehicles within a local area" [9]. The IEEE 802.11 specification provides “wireless standards that specify an “over-the-air” interface between a wireless client and a base station or access point, as well as among wireless clients” [10].

Figure 2.1 presents an overview of the IEEE 802.11 protocol stack. The standards specify the Medium Access Control (MAC) sublayer, the MAC management protocol and services as well as different physical layers (PHY). A key issue is transparency. Above the MAC layer, the 802.11 appears as any other 802.x LAN and offers similar services. The protocols are specified for communicating stations with and without the support of a specific infrastructure (Infrastructure Mode and Ad Hoc Mode, respectively). Furthermore, the standards describe the procedures for preserving privacy of user information.

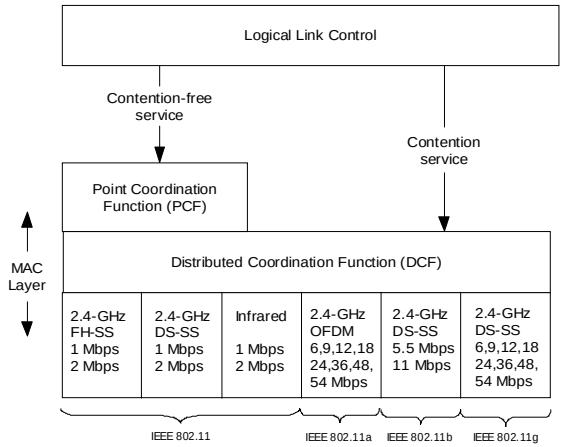


Figure 2.1: Overview of the IEEE 802.11 Protocol Stack

The MAC protocol provides two medium access methods: Distributed Coordination Function (DCF) and Point Coordination Function (PCF). The DCF is a contention protocol based on the Carrier Sense Multiple Access Protocol (CSMA), known as CSMA/CA, with CA standing for collision avoidance. It also uses small RTS/CTS (Request-To-Send / Clear-To-Send) packets to reserve the medium in order to avoid collisions due to problems involving a hidden terminal. The exchange of such control messages allows all the terminals within the receiving range of both the source and the destination terminals to defer transmission in order to allow successful delivery of a data frame.

Figure 2.2 illustrates the RTS/CTS mechanism. The Network Allocation Vector, NAV, represents the duration for which the stations have to leave the medium idle to allow successful delivery of both a data frame and the corresponding acknowledgement. When the traffic of a station is backlogged, Request-To-Send messages are sent to notify the other stations that it wants to transmit a packet to a specific receiver. The receiver then notifies the sender that it can transmit by sending a Clear-To-Send message. No other stations transmit any packet in a period corresponding to the transmission time of the packet to be sent.

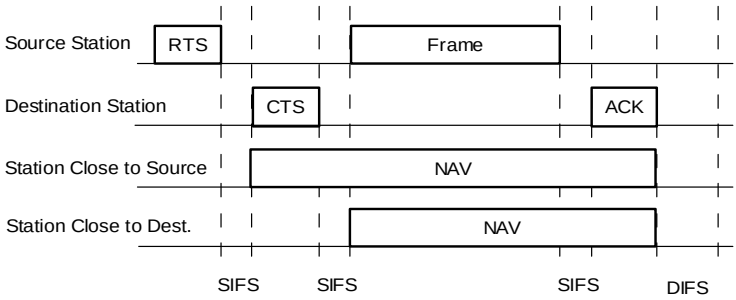


Figure 2.2: 802.11 RTS/CTS exchange

The small size of RTS/CTS control frames makes the probability of collision during their transmission lower than that of the collision of data frames, since these frames are usually larger than those involving RTS/CTS frames.

The joint usage of DCF and PCF presents obvious advantages in BSS infrastructure when the Base Station (BS) coordinates access to the wireless medium. However, experimental evidence shows that the performance of the PCF tends to be poor under certain conditions, such as during the simultaneous transmission of multimedia streams and best effort traffic [12].

The hierarchy of the IEEE 802.11 standards for wireless local area networks is presented in Figure 2.3. Three physical layers are defined by these standards:

- Infra Red (IR), which supports bitrates of 1 or 2 Mbps;
- Frequency Hopping Spread Spectrum (FHSS), operating at 2.4 GHz, which supports bitrates of 1 or 2 Mbps;
- Direct Sequence Spread Spectrum (DSSS), operating at 2.4 GHz, which supports bitrates 1 or 2 Mbps.

In order to improve the wireless channel capacity, physical layer extensions to the original IEEE 802.11 standard have been proposed.

The IEEE 802.11a extension adopts Orthogonal Frequency Division Multiplexing (OFDM) and works in the 5 GHz band to provide PHY data rates ranging from 6 Mbps up to 54 Mbps.

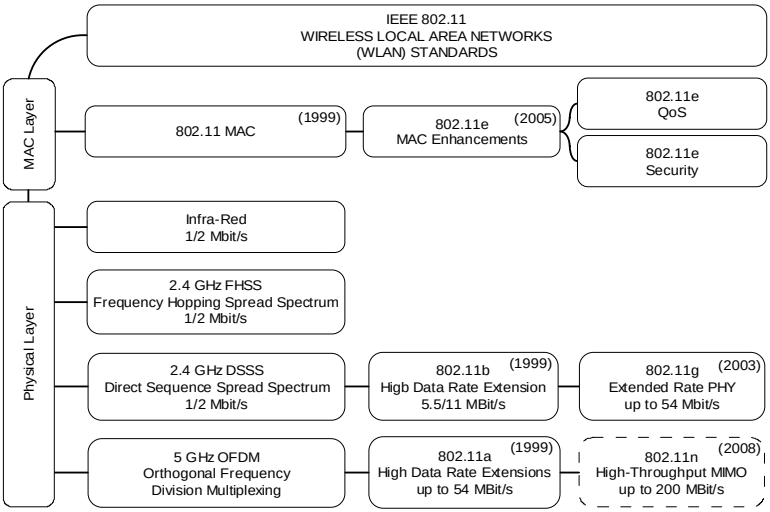


Figure 2.3: IEEE Wireless Local Area Network (WLAN) standards

The IEEE 802.11b extension, a High Data Rate Extension, is the most frequently used nowadays. It defines requirements for the extension of the DSSS at 2.4 GHz to achieve data rates of 5.5 Mbps and 11 Mbps. An important feature of this extension is a rate shift mechanism which makes it possible for high data rate networks to slow the rate down to 1 or 2 Mbps.

The IEEE 802.11g extension is similar to the 802.11a extension and specifies a physical layer for wireless LANs in both 2.4 GHz and 5 GHz bands, with a maximum rate of 54 Mbps. Such a rate is achieved by using OFDM. This provides backward compatibility with the 802.11b extension, but is not compatible with the 802.11a extension. This backward compatibility in the 802.11g extension can be considered as a disadvantage, since an Access Point (AP) running at the high data rate of 802.11g will switch down to the 802.11b rate upon the logging of any 802.11b device, thus reducing the transmission rate of all other devices in a cell [13].

The IEEE 802.11e protocol is an extension of the original MAC protocol aimed at providing Quality of Service (QoS) support for a variety of multimedia services over 802.11a, 802.11b and 802.11g physical layer specifications. In this extension, classes for service differentiation are defined. It introduces certain enhancements of the basic functions of the MAC operation: Enhanced DCF (EDCA) and the Hybrid Coordination Function (HCF), which operate in Contention and Contention Free periods, respectively.

The IEEE 802.11n task group was created at the end of 2003. The purpose of this group is to develop an extension of the IEEE 802.11 standards that produces a throughput greater than 100Mbps. For this extension, throughput will be measured in the region between the 802.11 MAC and higher layers rather than at the physical layer. This measurement procedure should reveal the real throughput available for applications. A discussion of the differences between the actual data rate available for an application and that reported by standard specifications is presented in the next section. The draft of the standard was released in January 2006. However, according to the estimated schedule of the 802.11n task group the standard release will not be finished until 2008.

For additional details about IEEE 802.11 extensions, the reader is referred to the website of the IEEE 802.11 Working Group [11].

2.3. Performance Bounds and Limitations

Although the IEEE 802.11 standards provide mobile broadband access to the Internet, it suffers significant performance limitations. This section provides an overview of these drawbacks, both from the theoretical as well as from the implementation point-of-view.

The performance of the IEEE 802.11 standards depends on both throughput and delay considerations when the CSMA/CA (with the RTS/CTS mechanism) is employed. Actually, the main goal of the proposed mechanisms is the provision of both high

throughput on the wireless channel and low delay in packet delivery.

The most relevant issues are discussed next:

- *Bandwidth:* The IEEE 802.11 standards specify the rates available for data transmission at the physical layer. The total link capacity is shared by all nodes which can operate within transmission range, including hidden terminals. Since collisions dramatically decrease the throughput, it is desirable to have knowledge of the total available bandwidth. Thus, various predictive algorithms have been proposed for that [14]. Moreover, the IEEE 802.11 standards have certain theoretical limitations, due to the MAC policy, and these cannot be eliminated by simply raising the channel capacity. In [15], these limitations are identified as the Throughput Upper Limit (TUL) and the Delay Lower Limit (DLL).
- *Latency:* Latency in a wireless medium is greater than in a wired one. The factors that influence latency are propagation delay, overhead added by both physical layer and link layer protocols and the retransmission policy implemented at the link layer.
- *Channel losses:* Wireless channels suffer from fading caused by interference with other sources. While the Bit Error Rate (BER) varies from 10^{-6} to 10^{-8} in wired channels, it varies from 10^{-3} to 10^{-1} in wireless channels [16]. The typical scheme used to recover from losses is the link layer ARQ (Automatic Response reQuest).
- *Mobility:* The most common network setting is the infrastructure BSS connected to a fixed network via a Base Station (BS). Handoff (switching between these BSs) requires that all the information associated with user activities be transferred to the next BS to prevent the termination of service provided to the mobile user.
- *TCP:* The problems that arise in the usage of TCP over wireless networks are due to their low reliability, as well as time-variant characteristics such as fading, shadowing,

node mobility, hand-offs, limited available bandwidth and large RTTs. TCP protocol, originally designed for wired networks which are characterized by stable links with packet losses mainly caused by congestion, performs poorly in such environments [2, 3, 91, 94].

The IEEE 802.11 standards specify different rates for data transmission, ranging from 2 Mbps to 54 Mbps. However, a relatively large portion of the channel capacity is wasted due to the high overhead required for the transmission of data frames on the wireless channel. Each message coming from the application layer is required to be encapsulated into lower layer Protocol Data Unit (PDU) in order to be transmitted on the physical layer. Figure 2.4 provides a graphical representation of this encapsulation when TCP is used.

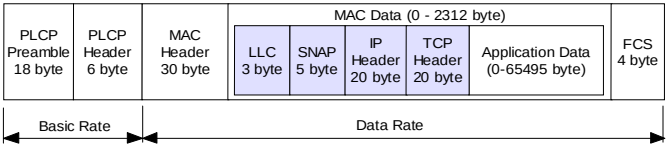


Figure 2.4: Packet encapsulation over 802.11 networks for TCP protocol

Legend:

FCS:	Frame Check Sequence
MSS:	Maximum Segment Size
MTU:	Maximum Transmission Unit
SNAP:	SubNetwork Access Protocol
LLC:	Logical Link Control
PLCP:	Physical Layer Convergence Protocol

Most overhead due to packet encapsulation is related to the PLCP preamble, which is necessary for the synchronization of the wireless receiver. This preamble, as well as the PLCP header, is transmitted at basic rate - regardless of the actual link speed. This makes it possible to operate at different speeds, since the information about the rate of the remaining portion of the PPDU is included into PLCP header. The PLCP preamble and header always

take 192 microseconds (for basic rate of 1 Mbps), regardless of the actual bit rate of the channel.

Table 2.1 displays the maximum throughput obtained under the hypotheses of non-occurrence of collisions, no fragmentation and no sending of RTS/CTS frames [17] for a frame size of 1500 bytes, which is the maximum transfer unit (MTU) commonly allowed in Ethernet networks. A high percentage of the wireless link capacity is clearly wasted in transmitting supplementary information, which reduces the bandwidth available for data transmission to a level well below the reported capacity. For the widely used IEEE 802.11b as well as for IEEE 802.11a extensions the throughput is reduced to less than a half of the reported capacity (for maximum data rates). This value may be further decreased by exponential backoff and RTS/CTS mechanisms.

Table 2.1: Achievable throughput of 802.11a, b

Physical Layer	Link speed (Mb/s)	TCP Throughput (Mb/s)	Efficiency (%)
802.11b	1	0.75	74.9
	2	1.41	70.7
	5.5	3.38	61.5
	11	5.32	48.4
802.11a	12	9.2	76.6
	24	16.2	67.5
	54	26.57	49.2

2.4. Available Enhancement Schemes

Various approaches have been proposed to optimize the performance of IEEE 802.11 wireless networks. These can be broadly categorized into three groups:

- *Link Layer solutions:* The principle of this approach is to solve problems locally, with the transport layer not being made aware of the characteristics of the individual links. Such protocols attempt to hide losses in the wireless link to make it appear to be a highly reliable one. Link layer solutions require no changes in existing transport layer protocols.

- *Transport Layer solutions:* The theory underlying this approach is the modification of the transport protocol in order to achieve high throughput on wireless links. Since some packets may be lost, the modified transport protocol should implement congestion control as a reaction to packet losses, moreover, other schemes should be implemented to consider the peculiarities of the wireless environment.
- *Cross-Layer solutions:* Cross-layer solutions break the principles of layering by allowing interdependence and joint development of protocols involving several layers of the protocol stack.

A graphical representation of this classification is presented in Figure 2.5.

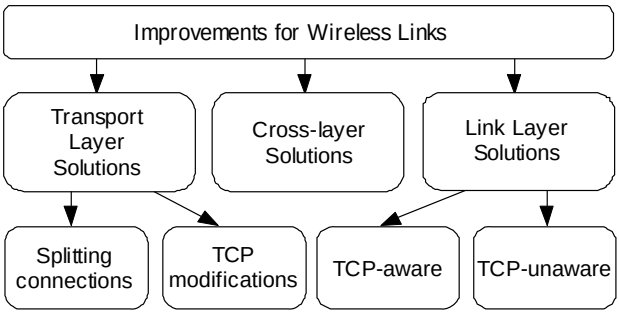


Figure 2.5: Graphical classification of possible improvements for 802.11 wireless networks

2.4.1 Link Layer Solutions

TCP was originally designed and optimized for wired networks. The performance problems related to its use in wireless domain are related to the nature and characteristics of the wireless medium. Hence, attempts to make wireless links resemble wired ones for high-level protocols are reflected in various approaches. The proposed solutions for the link layer can be classified into two groups on the basis of the awareness of the transport layer protocol. TCP-unaware protocols optimize the link layer by hid-

ing the differences between wired and wireless mediums so that the transport layer can operate as if it is installed in a wired network. This method does not violate the modularity of the protocol stack, however, since the necessary adaptations improve the reliability independent of higher-layer protocols. Nonetheless, this lack of awareness can affect performance under certain specific conditions.

For instance, a link layer retransmission technique may trigger a considerable number of TCP time outs, greatly decreasing the throughput of TCP. TCP-aware link layer solutions attempt to prevent unnecessary changes in the behavior of the transport protocol.

TCP-aware Link Layer Protocols

The TCP-aware link layer protocol presents certain advantages since knowledge of the protocol operating at the transport level allows fine tuning of the performance. For instance, TCP-unaware approach may trigger retransmissions at both link and transport layers simultaneously.

Snoop protocol [18] is used to handle connections in which most of the data are transferred from the Fixed Host (FH) to the Mobile Host (MH). The mobile node runs a snoop protocol while snoop agents are located at the base station, which is the most common place for bridging the wired and wireless parts of a network. Snoop agents are implemented in the routing module of the protocol stack of the base station in order to allow inspection of the packet headers. These snoop agents maintain caches of TCP packets that have not yet been acknowledged by the MH. The retransmissions are performed locally over the wireless link and are based on the reception of duplicate ACKs from the mobile node. Upon successful retransmission, duplicate acknowledgements are dropped at the Base Station (BS) to avoid the execution of the TCP fast retransmission mechanism.

A negative acknowledgement scheme has been added to improve the error recovery mechanism in case most of the data are sent from the MH to the wired network. In this way, the BS keeps

track of all packets lost in any window and generates negative acknowledgements to the MH.

These negative acknowledgements are typically based on the Selective Acknowledgement (SACK) option of the TCP [19].

The main disadvantages of Snoop protocols [20] are:

- Required changes in the BS protocol stack, demanding base station resources;
- No consideration of packet loss and delay during handoff;
- Failure in Snoop operation for encrypted traffic when there is no access to the packet header. The only possibility for handling encrypted traffic is to have part of the TCP header unencrypted, which is not feasible according to existing IP SEC standards specifications [21].

WTCP [22] performs local retransmission on the wireless link between the base station and the mobile node without the need for modification of TCP for the fixed nodes. WTCP running on a base station buffers all unacknowledged packets coming from the fixed sender, maintaining its own flow control over the wireless link. When an arriving segment is the next expected one, it is stored in the WTCP buffer along with information about its arrival time. The sequence number of the next expected packet is then increased by the number of bytes included into received segment. When a duplicate packet is received, it is dropped, since it has either been delivered to the wireless host or has already been buffered.

WTCP maintains the state of the information about the wireless part of the connection, such as the transmission window, sequence number of the last acknowledgement received from mobile hosts, and sequence number of the last segment sent to a mobile host.

Inaccurate RTT estimation at the sender can lead to unnecessary retransmissions triggered by timeouts, since the sender writes the timestamp upon the packet creation and the receiver echoes it back without any modification. If the segment is retransmitted, this RTT estimation may be affected. To avoid this, in WTCP the

timestamp is incremented by the fraction of time the packet spent in the BS buffer. With this mechanism, WTCP conceals the existence of wireless link errors and the RTT difference from the FH sender.

WTCP does not change the basic end-to-end TCP semantics, since acknowledgements to the fixed node are generated following successful packet delivery to the mobile node only.

TCP-unaware Link Layer Protocols

In early studies, enhanced link layer performance was achieved by the implementation of error correction techniques such as forward error correction (FEC) or the implementation of various Automatic Repeat ReQuest (ARQ) schemes for the retransmission of lost packets at the link layer. The combined implementation of these two techniques is considered in the AIRMAIL protocol.

*AIRMAIL (*Asymmetric Reliable Mobile Access In Link Layer) [23] is a protocol designed for both indoor and outdoor wireless networks. The combined usage of FEC and local retransmissions provided by ARQ aims at enhancing end-to-end throughput and latency by correcting errors in an unreliable wireless channel. The asymmetry in the design of the protocol reduces the processing load at the mobile node, since mobile terminals involve limited power and fewer computational resources than do base stations. The key idea in the asymmetric protocol design approach consists of empowering the base station with a certain degree of intelligence. The mobile terminal is required to combine several acknowledgements into a single acknowledgement to conserve power. The BS is required to send periodic status messages, making the acknowledgement from the mobile terminal event-driven. FEC implemented in AIRMAIL incorporates three levels of channel coding with adaptive interaction. The coding overhead is adaptively changed so that bandwidth expansion from forward error correction is minimized. However, sending large packets over the wireless link to save power reduces error correction possibilities for the TCP [24].

TULIP (Transport Unaware Link Improvement Protocol) [24] is designed for half-duplex radio links. The TULIP provides a reliable link for higher layer protocols. It is service-aware, i.e. it provides reliable service for TCP data traffic, and unreliable service for UDP traffic. On the receiver side, this protocol buffers packets providing in-order delivery and thus preventing the receipt of duplicate acknowledgements (by the TCP). Another important feature implemented in the TULIP is the local link layer retransmission of lost packets, which effectively prevents retransmissions over the entire path.

Delayed Duplicate Acknowledgements (DDA). DDA involve an attempt to approximate the behavior of Snoop protocols. The BS implements a local link layer retransmission scheme for packets lost on wireless links. Such retransmissions are triggered by the link layer acknowledgements, rather than by the TCP-duplicated acknowledgements of the Snoop protocol scheme. As specified in [25], each TCP data packet, as well as ACKs, is encapsulated into a single link layer data packet, with its successful reception acknowledged by the link layer ACK. This ACK contains the sequence number of the link layer packet received, which is independent of the TCP sequence number. Although it maintains sequence numbers at the link layer, DDA does not attempt in-order delivery of the TCP data packets. It differs from what happens with the Snoop protocol, for which duplicate acknowledgements are not dropped immediately but rather delayed. At the same time, the packet lost previously is retransmitted locally at the link layer. If the retransmission is not successful and the time for which duplicate acknowledgements are delayed has expired, they will be released in the direction of the TCP sender to trigger retransmission at the TCP level. DDA provides results equivalent to those produced by the Snoop protocol, although the technique does not perform well on slow wireless links [25].

DAWL (Delayed-ACK Scheme on Wireless Link) [26, 27] is a technique designed for the enhancement of link layer performance. It modifies the standard IEEE 802.11 MAC stop-&-wait ARQ by implementing the technique of native TCP delayed acknowledgement. The main idea is the lack of a need for immediate acknowledgement of packet receipt. DAWL assumes the existence of data in transit in the opposite direction. The possibility of encapsulating acknowledgements into data packets at the link level leads to the reduction of the traffic load in the wireless link. Moreover, DAWL implements a negative acknowledgement scheme for fast retransmission of lost packets.

Like the delayed ACK option of TCP, DAWL provides certain advantages in the presence of bi-directional traffic over wireless links. DAWL implementation requires a set of timers, which must be carefully tuned in order to optimize throughput. Although this approach provides advantages for operation on a single-hop wireless link, it is also problematic due to the:

- Difficulty in tuning timer values in the multi-hop environment;
- Increased delay in packet delivery;
- Increased buffer requirements for wireless nodes, and
- Poor performance for links with high error rates.

2.4.2 Transport Layer Solutions

As mentioned above, TCP was originally designed for wired networks, where packet losses are caused mostly by network congestion, rather than errors resulting from noisy channels, handoffs and node mobility. However, congestion window reduction is the only reaction of TCP to a packet loss of any kind.

Despite the inadequacies of TCP implementation for wireless environments, changes can be avoided by adopting link layer solutions.

A reasonable number of solutions designed to achieve better performance through the modification of the TCP itself is available. These are logically divided into two groups according to the

technique they introduce: connection splitting approach and TCP modifications.

Connection Splitting Solutions

In this scheme, the end-to-end TCP connection is divided into fixed and wireless parts, so that more degrees of freedom are available for the optimization of TCP over both wired and wireless links.

The disadvantages of connection splitting mainly involve the attempt to perform transparent from the wired node splitting of TCP. This leads to greater complexity in base station procedures, which is the most common and suitable place for splitting; the greater complexity involves not only handoff handling but also, prevention of end-to-end semantics of the TCP connection and, also greater software overhead caused by the TCP part of the stack involved at the intermediate point.

I-TCP (Indirect-TCP) [28] was one of the first proposals for using such a connection-splitting approach. I-TCP is based on the indirect protocol model proposed in [29]. In this approach, (on the transport layer) the end-to-end connection is split into two separate connections: one between the FH (with regular TCP) and the Mobility Support Router (MSR), commonly the base station which serves the MH, and the other between the MSR and the MH.

Two separate connections make it possible to optimize transmission over the wireless link, concealing the loss recovery process on the wireless link from the fixed sender by implementing a modified version of TCP. The flow (control) and congestion control mechanisms used by I-TCP allow faster reactions to wireless link problems such as communication interruption and mobility.

Whenever an MH moves to another cell (or BSS), all the information associated with the entire connection is handed over to the new MSR. The fixed host is completely unaware of such indirection, although it maintains the end-to-end TCP connection alive while the mobile node moves from one cell to another.

This I-TCP approach, however, presents certain drawbacks:

- End-to-end TCP connection semantics cannot be preserved;
- Additional overhead during hand-offs related to connection state transfer;
- Not applicable to encrypted traffic.

METP (Mobile End Transport Protocol) is a special transport protocol designed to use the connection-splitting approach on wireless links. The authors [30] propose to eliminate TCP and IP layers from the protocol stack of the wireless node in order to reduce the associated communication and processing overheads. The splitting point (BS) acts as a proxy for the TCP connection, providing for conversion of the packets received from the fixed network. Assuming that the wireless link is the only wireless hop within the end-to-end connection, the METP approach shifts IP datagram reception from the MH to the BS, which means that the packet passes through the IP and the transport layer at the base station. After the reception of a datagram, the BS then delivers the data to the MH by using the METP protocol, which involves a reduced packet header containing only minimal information (link source and destination addresses, port- and connection-related information).

METP provides reliable data delivery across the wireless link by introducing a special local retransmission scheme to the link layer ARQ. It can also keep the overall TCP connection alive while dealing with handoffs. For this reason, all information, including states and sending and receiving windows, has to be handed over to the new BS. The authors report a throughput enhancement of up to 37% over TCP Reno and of 23% over other approaches [30]. However, this approach also has drawbacks:

- End-to-end semantics are not preserved;
- Great increase in complexity of the BS due to increased packet processing through the BS protocol stack since it must be handled twice, once when it is received (by using the TCP/IP stack) at the fixed host and again when it is

transmitted to the wireless part of the network by the METP;

- Additional overhead related to the transfer of large amount of information during handoffs.

TCP Modifications

TCP Modification involves a group of solutions which promote small changes to TCP behavior, such as the mechanics of acknowledgement generation used by TCP. TCP modification does not require modification of the BS avoiding overhead in packet delivery and BS complexity increase. The major proposals in this framework are summarized below.

Selective acknowledgements (SACK) is an option [19] for the efficient handling of multiple losses within a single window. The SACK acknowledgement algorithm enables the receiver to inform the TCP sender when packets are received out of order. The sender can then retransmit only those packets which have not reached the receiver. This technique is designed as an improvement of the standard cumulative ACK schemes in which retransmission is triggered by the reception of duplicate ACKs.

TCP using SACK provides a technique which performs better than standard TCP for multiple losses in a single window [2]. However, the window size must be large enough to take advantage of the SACK characteristics. The main drawback of this selective acknowledgement is the modification of acknowledgement procedures required at both sender and receiver.

TCP Santa Cruz [31] uses the optional field of the TCP header for the implementation of new congestion control and error recovery strategies. The congestion control algorithm is based on relative delays, both that between packets transmitted by the sender and that other between packets received at the receiver. This information is calculated by the TCP Santa Cruz using timestamps added to the packet at both ends, a technique originally presented in TCP Vegas [32]. An increase in the amount of information available about the TCP flow provides for more accu-

rate RTT estimations at the sender side, while the ACKs lost on their way back do not influence the forward throughput. TCP Santa Cruz leads to an improvement in performance in relation to both TCP Reno and TCP Vegas. The main drawback is the increased complexity at the sender side.

Explicit Bad State Notification (EBSN) [33] notifies the sender whenever a BS is unsuccessful in delivering a packet over the wireless network. To do this, it sends an EBSN message explicitly. The TCP sender restarts its timer to avoid execution of the slow start algorithm. EBSN requires minor modifications in the TCP sender code.

Explicit Loss Notification (ELN) [34] makes a sender aware of errors unrelated to congestion which have occurred on the wireless link. The base station monitors TCP packets in both directions. When a duplicate acknowledgement is received from the TCP receiver, the BS can encapsulate the ELN message by setting the ELN bit in the TCP acknowledgement header and forwarding it back to the sender. The sender can then choose a type of reaction based on the type of loss. ELN does not, however, provide local retransmission, so no caching is necessary. However, the required checking of all TCP headers represents an increase in complexity and additional processing overhead associated with each packet.

2.4.3 Cross-Layer Design

All the approaches described above optimize a single parameter at a time, but when several different variables are to be considered they should be taken into account at the same time in order to achieve a truly optimal solution for the adaptation of a TCP originally developed for a wired environment to a wireless scenario.

Such joint optimization can be included in the wide range of recently-proposed solutions for optimizing wireless network design that are labeled “Cross-Layer Design” [7, 35, 36, 92]. This approach breaks the ISO/OSI layering principles by allowing in-

terdependence and joint design of protocols throughout different layers.

ILC-TCP (Interlayer Collaboration Protocol) [37], was designed to improve the performance of the TCP in wireless environments, involving long and frequent disconnections. The main modification is introduction of a State Manager (SM) in parallel with the protocol stack for gathering information about TCP, IP and link/physical layers. If necessary, this information can be furnished upon the request of the TCP layer. Each layer periodically reports its state to the SM. In case the conditions are not appropriate for TCP flow SM signals the TCP sender to stop sending packets. When conditions have improved, TCP can proceed with regular data delivery.

This approach tries to optimize performance in a scenario in which mobile hosts act as TCP senders. It is an end-to-end approach which requires no changes in the fixed TCP receiver.

The authors [37] report an improvement of up to 25% in throughput in relation to standard TCP when disconnections and varied mobility patterns are present. However, in the absence of connectivity problems, ILC-TCP offers no improvement in TCP operation.

ATCP. In this approach, feedback between the network and the transport layers is allowed as well as between the application and transport layers [38]. On the application level, information about priority is specified by the user and interpreted by the transport layer so that priorities can be established.

2.5. Comparisons

In this section, the available proposals are compared since these proposals act at different layers, there is no proposal that outperforms the others in all possible scenarios. The best way to compare them is to underline their differences through a comparison of their characteristics. A brief summary of the existing solutions and their advantages and disadvantages is presented in Table 2.2.

Table 2.2: Brief summary of existing solutions (advantages and disadvantages)

Name	Advantages	Disadvantages
Link Layer Solutions		
Snoop	Designed for BSS infrastructure; Performs local retransmissions; Does not change TCP.	Modifies BS stack; Handoffs are not considered; Encrypted traffic is not supported.
WTCP	Performs local retransmissions; More accurate RTT estimation.	Greatly increases BS complexity; Mandatory maintenance state information for TCP connections; Costly management of handoffs.
AIR-MAIL	Combines MH acknowledgements; Event-driven MS acknowledgement; Power saving; Adaptive FEC implementation.	High error rates cause TCP timeouts;
TULIP	Provides in-order delivery; Local retransmissions; Useful on half-duplex links.	Can cause TCP timeout.
DDA	Provides local retransmissions at the link layer; Dupacks are delayed before dropping; Can operate with encrypted traffic.	Difficulty in choosing delay value (d); Poor performance on slow links; Can cause TCP timeout.
DAWL	Delayed ACKs at the link layer; Performs well on short links with low error rate in presence of bi-directional traffic.	Difficulty in tuning timers in multi-hop environment; Increased delay in ACK packet delivery; Increased buffer requirements.
Transport Layer Solutions		
I-TCP	Useful in infrastructure BSS; Connection splitting for faster reaction to loss occurred on wireless link.	End-to-end semantics not prevented; Increased overhead from BS stack; Increased BS complexity; Overhead for state transfer during handoffs; Not applicable in asymmetric networks.
METP	Elimination of TCP and IP layers, thus reducing header transmission overhead. Simplified headers on wireless link. Designed for infrastructure BSS.	End-to-end semantics not prevented; Increased BS complexity. Handoff handling costly.
SACK	Selective TCP ACK scheme; Good performance when window size is satisfactorily large.	Modifies TCP acknowledgement scheme at the receiver.
TCP-SC	Modification of congestion control and error recovery mechanisms. Improved RTT calculation.	TCP modification; Increased TCP sender complexity;
EBSN	Notification of TCP sender about problems on the wireless link to prevent slow starts.	Minor modifications of TCP. BS overhead.
ELN	Notification of TCP sender about errors occurring on wireless links.	Increased complexity of TCP. BS overhead.

Name	Advantages	Disadvantages
Cross-Layer Solutions		
ILC-TCP	Useful in wireless environments with frequent and long disconnections.	Additional layer added to sender protocol stack.
ATCP	Prioritization of applications from upper layers. Optimization of operation on basis of link state and RTO estimation from lower layers.	Modification of TCP, as well as other layers, for feedback.

Certain features, however, are shared by all of the approaches belonging to the same group. These characteristics and their limitations are described for each of the layers.

Link Layer solutions. The main advantage of link layer solutions is the maintenance of end-to-end semantics, without modification of higher protocol layers. This makes it possible to leave untouched the existing implementations of the protocol stack in the various operating systems and limit the introduction of modifications to the link layer.

Most of the approaches which operate at this level rely on some intermediate point within the end-to-end connection for the introduction of performance improvements. For example, the Snoop protocol performs local retransmissions from the BS cash, and WTCP, although operating in a similar way introduces more accurate RTT estimations, thus preventing a reduction in TCP throughput. Both Snoop and WTCP must, however, have an access to the header of TCP packets in order to function, which reduces their value if traffic is encrypted. DDA solves this problem by introducing a local retransmission scheme based solely on information transferred at the link layer; by delaying duplicate acknowledgements, it prevents the TCP source from duplication of efforts in retransmissions, since packets are produced locally at the link layer. The TCP-unaware protocols AIRMAIL and TULIP also rely on link layer retransmissions, but both also employ techniques for enhancement, the FEC for AIRMAIL and in-order delivery for TULIP.

The maintenance of information related to the connection at intermediate nodes brings an increased complexity of IR, especially when transport layer per flow support is required (Snoop

and WTCP), as well as a reduction in handoff performance when a large amount of information needs to be transferred to another IR to prevent the termination of an end-to-end connection.

DAWL tries to simplify the system by introducing modifications only in the ARQ scheme at the link layer and does not consider local retransmission at the IR. This design is advantageous in case of an IR crash. When this happens, all the information stored on the IR is lost, in the other schemes, and this would likely cause the termination of the end-to-end connection.

Transport Layer solutions propose modifications to TCP in order to improve the performance on wireless links. The modifications in the transport layer can be adopted for within the entire connection (SACK) or, separated by sender or receiver (EBSN and ELN). Moreover, the modification can focus directly on the wireless link, as in connection-splitting solutions (I-TCP, METP).

The main requirement for the modification of the TCP for running on wireless links is the allowance of the separation of losses due to congestion from those related to the nature of the wireless link (increased error rate, handoffs, etc.). The connection splitting solutions (I-TCP and METP) do not preserve the end-to-end semantics of the TCP while localizing the problem of the wireless link. At the same time, they introduce an increased complexity to the IR, as in the case of Link layer solutions.

The other approaches within this group preserve the end-to-end TCP semantics. SACK modifies the retransmission scheme of TCP in order to reduce unnecessary retransmissions for non-continuous losses within a single TCP window, while TCP-SC modifies the TCP sender and receiver to improve the congestion control algorithm on the basis of relative delay information. Neither SACK nor TCP-SC requires IR support and the increased complexity of the sender does not accumulate at a single point, but is rather distributed among the several nodes of the network.

The explicit notification schemes EBSN and ELN require support from an IR in order to provide information either about the state of the wireless link (EBSN) or about the type of loss to the

TCP sender (ELN). The crash of an IR does not have a significant impact on the functionality of these solutions since there is no connection-related information stored at this point.

Cross-Layer solutions. All the solutions mentioned try to optimize the performance of IEEE 802.11 networks within a single link, without the support of the IR.

ILC-TCP provides the framework for TCP to obtain information about long and frequent disconnections from the lower layers of the protocol stack. More extensive feedback is introduced by ATCP, in which the TCP obtains information not only from the lower layers, but also from the application layer, depending on the level of priority of the applications running above it. ATCP introduces a modified version of TCP mechanisms considering information gathered from lower layers, such as link state and RTO estimation.

Table 2.3 provides a more detailed comparison of the existing protocols in relation to the following parameters.

Table 2.3: Comparison of existing solutions

Scheme	E2E	TCP modification	Retransmit at	IR Crash
Link Layer solutions				
Snoop	Present	Absent	BS (Transport)	Present
WTCP	Present	Absent	BS (Transport)	Present
AIRMAIL	Present	Absent	BS (Link)	Present
TULIP	Present	Absent	BS (Link)	Present
DDA	Present	Absent	BS (Link)	Present
DAWL	Present	Absent	Sender	None
Transport Layer solutions				
I-TCP	Absent	Present	BS (Transport)	Present
METP	Absent	Present	BS (Transport)	Present
SACK	Present	Present	Sender	None
TCP-SC	Present	Present	Sender	None
EBSN	Present	Present	Sender (Link)	None
ELN	Present	Present	Sender (Transport)	None
Cross-Layer solutions				
ILC-TCP	Present	Present	Sender	None
ATCP	Present	Present	Sender	None

- *Protocol Layer*: Solutions at the link layer try to localize a problem, and the optimization they perform is transparent from the transport layer. Solutions presented at the transport layer, which are aware of the existence of wireless link, try to optimize TCP performance for conditions typical for the wireless link. Cross-layer solutions provide joint optimization at both levels.
- *End-to-end (E2E) semantics*: This parameter identifies whether or not a solution preserves the end-to-end semantics of the TCP. Preservation means that the reception of an acknowledgement by the TCP sender guarantees successful data delivery to the TCP receiver throughout the entire end-to-end connection.
- *TCP modification*: This parameter indicates whether a solution requires modifications to the transport layer of the protocol stack. Since there are numerous implementations of TCP in various operating systems from different vendors, a modification of TCP may require a huge effort. For this reason, solutions which include TCP modifications may find implementation in very limited number of cases, even if the improvement achieved is usually high.
- *Intermediate Router (IR) support*: The overwhelming majority of solutions use an intermediate point within the end-to-end connection for performance optimization, such as splitting the connection at that point, or using it to notify about network conditions (ELN and EBSN). In BSS infrastructure, the base station commonly plays the role of IR. The rest of the parameters are connected with the existence of an IR along the transmission path.
- *Retransmit at*: The most commonly used technique for performance enhancement is the local retransmission of lost packets only on the wireless link, rather than throughout the entire end-to-end connection. The "Retransmit at" parameter indicates the point where retransmissions are performed, as well as the protocol layer in which a solution handles retransmission.

- *IR crash impact*: When a solution relies on the IR, the crash of that intermediate point can lead to the termination of the TCP connection. In some cases, there is no possibility of maintaining the data flow when state information is lost.

Table 2.3 underlines a wide range of existing solutions, the differences in functionality implemented and their impact on network design. Thus, in such a way, there is no single solution which will perform well in all scenarios.

2.6. Deployment scenario

Focusing on the different scenarios for deployment of the schemes presented makes it possible to define different architectures involving IEEE 802.11 wireless technology:

Single-hop wireless connections: This scenario involves transmission between two mobile stations equipped with 802.11 wireless network cards, and by far is the simplest scenario.

IEEE 802.11 networks rely mostly on the link layer ARQ to provide reliable delivery of packets to transport protocols. If the link layer abandons packet transmission after all possible retransmissions, that packet will be transmitted by the TCP. This does not, however, increase an overhead greatly, since there is only a single hop between the sender and the receiver. DAWL scheme takes advantage of improvements derived from the use of an ARQ scheme for relatively low channel error rates in the presence of bi-directional traffic.

The E2E transport layer solutions, such as SACK and TCP-SC, improve the mechanism of TCP acknowledgement and congestion control, as well as error recovery, and these will be reflected in performance improvements.

Multi-hop scenario: In this scenario, transmission occurs via multiple hops; there is no stationary infrastructure installed in multi-hop networks. Note that the MAC protocol specified by the IEEE 802.11 standard does not perform well in such an environ-

ment. Due to problems such as hidden nodes, exposed nodes and the unfairness of the exponential back off algorithm, this protocol "can not perform well in multi-hop networks" [39]. The analysis of existing solutions shows that the design of most proposals does not consider their operation in this scenario. In the best possible situation, only the final hop of a multi-hop connection is taken into account. Transport layer solutions (I-TCP, METP, EBSN, and ELN) require support from IR operating within a connection. In a multi-hop scenario, there is no centralized point for splitting (like BS), which makes the implementation of such schemes difficult. As a result, the usage of cross-layer schemes is more realistic in multi-hop network scenarios.

Wireless-cum-wired scenario: This is the most widely diffused scenario where a network is only partially wireless. The sender is located on the wired part of the network, which communicates with the mobile host through the gateway (the BS, in BSS infrastructure). Almost all solutions consider this scenario in their design. Moreover, some of them (such as Snoop, and WTCP) are especially designed to enhance performance in this case. All solutions in the transport and cross-layer approaches can be implemented in a wireless-cum-wired scenario.

2.7. Conclusions

Wireless networks are becoming increasingly popular due to the growing use of mobile access to network services. As a consequence, significant efforts have been devoted to provide reliable data delivery for a wide variety of applications over a variety of wireless infrastructures.

In this scenario, the IEEE 802.11 standard and its extensions have gained a worldwide diffusion, providing reasonable performance with reduced infrastructure and deployment costs. However, performance bounds and limitations of 802.11 WLANs exist. This chapter has provided an overview of the various solutions available for coping with these limitations.

From the analysis of existing improvements to the IEEE 802.11 standards, it is clear that there is no single best solution for all deployment scenarios.

Link layer solutions work on the wireless link without affecting higher-level protocols, but they increase the complexity of the base station and require the modification of the MAC protocol on the wireless link (usually implemented in the hardware).

Transport layer solutions aim at adapting the transport protocol to the characteristics of the wireless network, thus implying modification of the transport protocol in the protocol stack at both the sender and the receiver ends.

An alternate novel approach is represented by cross-layer solutions, which establish interdependence and collaboration between protocols in different layers of the stack. However, in spite of the theoretical advantages described above, a relatively low number of proposals aimed at TCP optimization is available in this category. Moreover, despite the arguments for a wider optimization across multiple layers [40], the overwhelming majority of the available cross-layer solutions are limited to the cooperation between the physical and the link layers [41].

We observe the cross-layering is currently one of the most promising techniques for optimization towards high performance in such a challenging environment as wireless networks.

The acknowledgment of our weakness
is the first step in repairing our loss.

- Thomas Kempis

Chapter 3

3. Cross-Layer ARQ

3.1. Introduction

Wireless communications clearly represent a fast-growing sector in data networks [42]. Mainly, wireless technologies provide mobile access to networks and services – omitting the requirement for a cable (and fixed) infrastructure, thus enabling fast and cost-effective network organization, deployment and maintenance.

Wireless technologies are envisaged to be widely deployed in the last mile – connecting end-user to the core of the network, while leaving transport of data in the core to cable or optical architectures. Indeed, last mile is the most critical issue in today's network architectures. The characteristics of the last mile links often determine the performance of the overall network representing the actual capacity bottleneck on the entire path from the data source to the destination and influencing the characteristics of traffic patterns flowing through the network.

In particular, wireless networks suffer from several performance limitations, in some cases related to excessive burden deriving from the layering paradigm employed for the TCP/IP protocol stack design. Indeed, TCP/IP was originally designed for wired links which general characteristics include high bandwidth, low delay, low probability of packet loss (high reliability), static rout-

ing, and no mobility. On the contrary, in the wireless domain, performance and resource availability are limited by the limited availability of transmission spectrum, the employed modulation type and the available transmission power. Loss probability experienced by packet transmission is in general higher on the wireless medium rather than on wired links: while Bit Error Rate (BER) varies from 10^{-8} to 10^{-6} for wired channels, it varies from 10^{-3} up to 10^{-1} for wireless channels [16]. Such error rates are unacceptable for the Transmission Control Protocol (TCP) [43], designed for wired networks, which delivers over 85% of Internet traffic [44, 45]. The reason for that is in the additive increase multiplicative decrease (AIMD) congestion control which treats all losses as congestion losses and thus underestimates the actual capacity provided by the network.

In order to counteract such variation of BER, Forward Error Correction (FEC) can be employed at the link layer. However, FEC is not the proper solution to provide reliable transmission on wireless networks. The main drawback is the waste of transmission resources deriving from its employment in absence of errors, therefore suggesting the usage of feedback information from the receiver in order to extrapolate information on the channel status. Thus, a traditional and widely implemented approach to increase reliability of wireless links is based on the usage of an Automatic Repeat reQuest (ARQ) protocol at the link layer.

ARQ provides a dynamic way to decrease error rate present on the wireless links by increasing delivery delay and consuming additional bandwidth resources. The most commonly used ARQ scheme in wireless networks is “stop & wait”: the sender is not allowed to send the next packet in the queue until the receiver positively acknowledges the successful delivery of the previous one. The advantage derives from the fact that only corrupted packets are retransmitted, introducing a level of overhead adapted to the conditions of the link. Scalability and low computational cost of implementation resulted in the employment of the ARQ principles in most of the wireless networks.

Table 3.1 presents a summary of the characteristics of the wireless network standards, aimed at underlining the common features and similarities among them. The reader should refer to the references for more details on cellular networks [46], IEEE 802.11 local area networks [11] and WiMax metropolitan area network [47] standards.

Table 3.1: Characteristics of leading wireless technologies

Technology	Nominal Range	Frequency Band	Channel Bandwidth	Physical Rate	TCP/IP Throughput	Mobility	ARQ
Wireless Wide Area Networks (WWAN)							
GSM (2G)	3–35 km	900 MHz, 1800 MHz (TDMA) 800 MHz, 1900 MHz (CDMA)	200 KHz (TDMA) 1.23 MHz (CDMA)	9.6 – 57.6 kbps	4 – 38 Kbps	Seamless global roaming	yes
(E)GPRS (2.5G)			200 KHz (TDMA)	56 - 115 kbps	32 – 84 Kbps		yes
EDGE				384 Kbps (48 – 60 Kbps per timeslot)	300 Kbps		yes
3G		1900 – 2025 MHz, 2110 – 2200 MHz	5MHz	large range 144 Kbps, medium range 384 Kbps, small range 2 Mbps	120 Kbps, 310 Kbps, 1.6 Mbps		yes
3G LTE		3G spectrum	1.25, 2.5, 5, 10, 20 MHz	DL: up to 100 Mbps, UL: up to 50 Mbps	Up to 80 Mbps		yes
Wireless Local Area Network (WLAN)							
IEEE 802.11	40 – 100 meters	2.4 GHz	22 MHz	1/2 Mbps	0.7/1.4 Mbps	Nomadic subnet roaming	yes
802.11b (Wi-Fi)				11 Mbps	5 Mbps		yes
802.11a		5 GHz		54 Mbps	25 Mbps		yes
802.11g		2.4 GHz		54 Mbps	25 Mbps		yes
802.11n		5 GHz	20/40 MHz	250+ Mbps	100+ Mbps		yes
Wireless Metropolitan Area Network (WMAN)							
IEEE 802.16 (WiMax)	Up to 50 km	11 – 66 GHz	20, 25, 28 MHz	32 – 134 Mbps		Fixed	no
IEEE 802.16a		2 – 11 GHz	1.75 – 20 MHz	4 - 75 Mbps	3.22 – 56 Mbps	Fixed	yes
IEEE 802.16e	1 – 4.5 km	2 – 6 GHz	5 MHz	15 Mbps		Pedestrian mobility – Regional roaming	yes

From the analysis of the table it is clear that, while employing different approaches at the physical layer (in terms of modulation, data rate, transmission bandwidth), most of the presented technologies provide reliable communications by employing different ARQ schemes at the link level.

However, the link layer is not the only layer which acknowledges packet delivery: TCP reliability is obtained through the utilization of a positive acknowledgement scheme which specifies TCP receiver to acknowledge data successfully received from the sender. TCP header reserves special fields for enabling it to carry acknowledgement information. As a result, the TCP receiver can produce a TCP acknowledgment (TCP-ACK) as standalone packet or, in case of bi-directional data exchange, encapsulate it into outgoing TCP segments.

Whenever a TCP segment is transmitted over the wireless link, the sender first receives an acknowledgement at the link layer. Then, TCP entity at the receiver generates an acknowledgement at the transport layer. This acknowledgement represents an ordinary payload for the link layer, which should be acknowledged by the link layer of the sender node.

Summarizing, in most of the available wireless network architectures (see Table 3.1), a single TCP data packet transmission is acknowledged three times: one at the transport level and two times at the link layer, and (for each acknowledgement) physical and link layer overhead is added. This results in a relevant performance reduction. For example, in case a wireless medium supports multiple rates (like in WiFi or WiMax), physical layer preamble and header are always transmitted at the lowest bitrate – for backward compatibility as well as due to communication range limitations – therefore penalizing performance more at higher bitrates.

Optimization of the acknowledgement scheme will obviously bring performance improvement through the reduction of the medium-busy time and would require interaction between the transport and link layers – thus requiring proper cross-layering schemes.

In this chapter we target cross-layering as a possible solution, presenting a novel cross-layer approach, called Link Layer ARQ Exploitation TCP (LLE-TCP), where the main performance advantages are achieved through the optimization of interlayer Automatic Repeat reQuest (ARQ) scheme functionality.

3.2. The Proposed Scheme (LLE-TCP)

The Link Layer ARQ Exploitation TCP (LLE-TCP) approach is proposed to exploit the information of the link layer ARQ scheme for a more efficient acknowledgement of TCP packet delivery. The core of the approach is based on the idea that, when a TCP packet is successfully delivered at the link level, the TCP ACK for the transport layer should not be sent through the channel, but it could be automatically generated locally at the sender side. In order to support this functionality, no changes are needed to the TCP protocol, but a new software entity needs to be introduced: the ARQ agent.

Before describing the LLE-TCP approach in detail, we would like to briefly introduce the reference scenarios for its deployment. Here, we differentiate between three types of wireless networks which can be classified by the employed topological structure and connectivity into: single hop wireless network (ad hoc network specified by IEEE 802.11, see Figure 3.1a), infrastructure network (IEEE 802.11, IEEE 802.16 or cellular network, see Figure 3.1c), and ad hoc multi-hop network (see Figure 3.1b).

Within the detailed description of the proposed approach we mostly refer to the single-hop network scenario (Figure 3.1a) for sake of simplicity of presentation, while implementation details of LLE-TCP in infrastructure and multi-hop networks are presented in dedicated sections (3.3.3 and 3.3.4, respectively). Furthermore, within the description of the proposed approach we often mention considerations of its possible implementations. In this case, we are mostly based on Linux OS open protocol model, assuming that other operating systems such as MS Windows have relevant conceptual similarities.

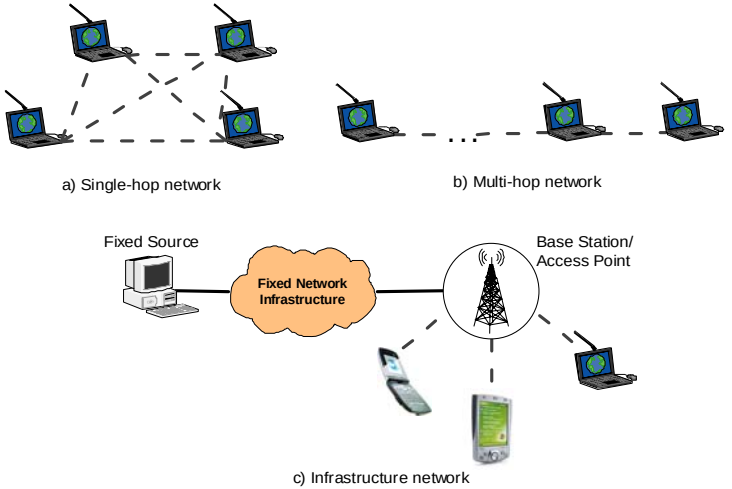


Figure 3.1: Network scenarios considered for LLE-TCP implementation

3.2.1 Cross-Layer ARQ Agent

As outlined in the introductory paragraphs of this section, the ARQ agent is a software module designed for the exploitation by TCP of the link layer ARQ scheme in wireless network environments. It operates within the protocol stack between TCP and MAC layers and it has interfaces to both TCP and MAC layers.

Figure 3.2 shows the logical position of the Cross-Layer ARQ agent in the protocol stack of the wireless node. The logical attachment of the ARQ agent to the link layer brings scalability to the proposed solution. Nowadays, the link layer of wireless networks is commonly implemented inside the firmware of wireless cards. Insertion of LLE-TCP functionality into the firmware releases resources of the main CPU, thus simplifying integration issues. Moreover, being attached to the link layer, the ARQ agent can be implemented inside the wireless card driver. Nevertheless, both implementations enable LLE-TCP operation only in the desired scenarios, i.e. in wireless networks employing ARQ at the link layer.

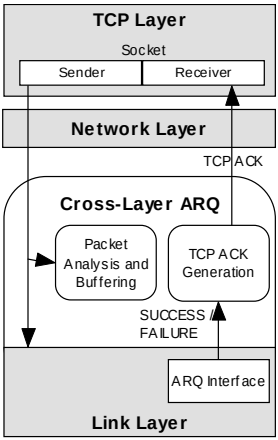


Figure 3.2: ARQ agent position within protocol stack

For the purpose of a better explanation of the core idea of the proposed approach, a standalone TCP data packet transmission using the designed scheme is presented in Figure 3.3. The following sub-sections give more detail about the proposed approach in terms of interactions of the ARQ agent with the link and transport layers and techniques used for local TCP ACK generation.

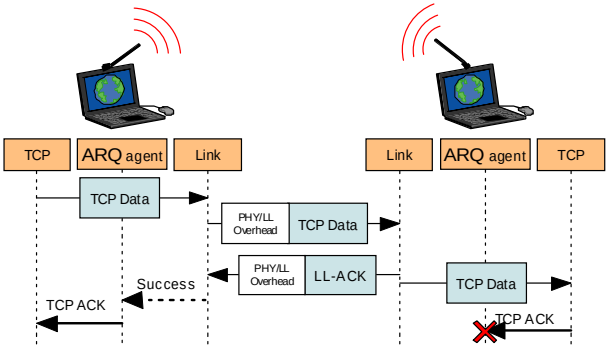


Figure 3.3: Packet delivery diagram of the cross-layer ARQ software module

3.2.2 Agent Interaction with the Link Layer

The link layer provides the ARQ agent with information related to packet delivery performed over the wireless link. After the agent releases a packet to the link layer, it remains waiting for SUCCESS or FAILURE response.

- SUCCESS indicates successful packet delivery. This event is generated upon reception of the ACK frame at the link level as the indication that the data packet is successfully received by the destination node.

- FAILURE indicates that, after all possible retransmission attempts, the link layer is not able to deliver the packet.

The link layer interface with the ARQ agent is unidirectional: it only provides the mentioned events and it does not receive any data from the agent.

3.2.3 Agent Interaction with the Transport layer

In the description of ARQ Agent interaction with transport layer, it is necessary to distinguish between the sender and the receiver sides.

On the sender side, the ARQ agent sniffs the packets generated by higher layers of the local protocol stack assuming to have an access to the network and transport layer headers. Whenever a TCP data packet is detected, the agent stores flow-related information such as flow sequence number carried by the packet.

Then, assuming to have a successful packet delivery at the link layer, the ARQ agent starts preparation of a TCP ACK which acknowledges the packet just transmitted to the transport layer. Such an acknowledgement is generated as a standalone TCP packet containing the ACK bit set to ‘1’ in its header and no data payload. Upon the SUCCESS event coming from the link layer, the prepared TCP ACK is released to the transport layer.

On the other hand, the FAILURE event indicates the packet is dropped at the link layer after exceeding all possible retransmission attempts. Following such indication, the ARQ agent requests retransmission from the TCP sender by generating three duplicate acknowledgements – which trigger fast retransmit procedure as

specified in [48]. This retransmission is requested only for the first packet lost in sequence, while for subsequent packet losses the ARQ agent simply ignores FAILURE messages.

Note that the presented ARQ agent does not change or overload any of the TCP flow control mechanisms. However, the elimination of the TCP ACK transmission over the wireless channel and corresponding impact on the delay component reduces the Round Trip Time (RTT) of a connection. As a consequence, this brings to an additional TCP performance gain due to faster window evolution and faster reaction to packet losses performed by the Additive Increase Multiplicative Decrease (AIMD) TCP flow control mechanism [49].

On the receiver side, the ARQ agent silently drops all standalone non-duplicate TCP ACK packets which are artificially generated at the sender side.

3.2.4 TCP Connection Phases

Figure 3.4 presents the modifications performed by LLE-TCP over the standard TCP functionality within the three main phases of TCP connection: connection establishing, data transmission, and connection termination.

Connection establishment phase, also called “three-way handshake”, accomplishes important tasks such as sequence number synchronization and negotiation of the size of the contention window. For that reason, the ARQ agent performs ACK suppression only for the third handshake in connection establishment and full ACK suppression for data exchange and connection termination phases.

In case of bidirectional data exchange, TCP encapsulates ACK into outgoing data packets. The receiver node does not suppress ACKs from packets of such type. However, suppression is performed at the sender side by the ARQ agent, which keeps track of the number of the last acknowledged TCP segment. In case the incoming packet acknowledges a segment number lower or equal to the already acknowledged one, the ACK flag is cleared.

In order to ensure proper functionality of the fast retransmit [48] introduced in TCP Reno, duplicate ACKs are never suppressed neither by ARQ agent at the receiver node nor at the sender node.

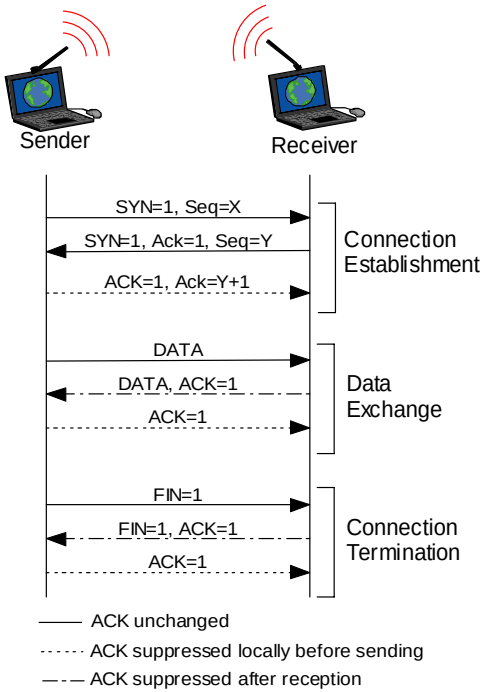


Figure 3.4: Acknowledgement suppression performed by LLE-TCP

3.3. Infrastructure Network Scenario

Nowadays, wireless networks are mostly used as a last-mile solution for data communications which can be found in almost every office, airport, coffee shop, etc. In this scenario, mobile nodes are connected to the core network using a set of mobile routers serving as bridges between the fixed network infrastructure and mobile devices. Multiple studies show that TCP performance is poor in such environments [18, 94], the main reason of which lies in

completely different characteristics between fixed and wireless parts along the end-to-end data connection [5].

In an infrastructure network scenario, ARQ agents are inserted into protocol stacks of the base station as well as of the mobile nodes (see Figure 3.5). The fixed host located in the wired part of the network runs standard TCP – thus avoiding any modification. The functionality of the ARQ agents is left unchanged compared with the single-hop scenario, with the only difference in the fact that TCP ACKs generated by ARQ agent at the base station are not delivered locally but routed to the fixed host. As a result, the transmission of TCP ACKs is avoided over the wireless part of the network (which is commonly a bottleneck of the connection).

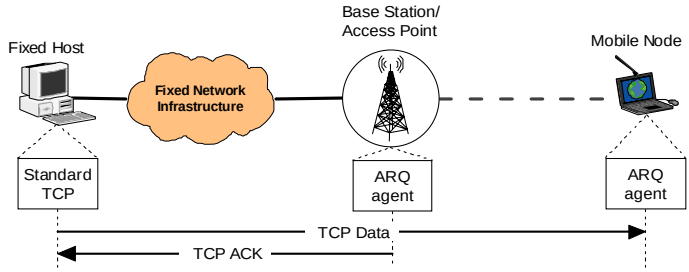


Figure 3.5: LLE-TCP in an infrastructure network scenario

Another relevant improvement of LLE-TCP lies in the possibility of enabling congestion control by the base station. Network congestion occurs when the amount of data sent into the network exceeds the available capacity, and AIMD window evolution of the best-effort TCP is a relevant cause of the congestion. It continuously increases the outgoing rate until the packets begin to be dropped due to router buffer overflows. A detailed study of network congestion, which motivated the employment of end-to-end congestion control, is performed by Sally Floyd et al. in [50].

A TCP connection between a fixed host and a mobile node traverses fixed and wireless sections of the network. Moreover, the limited capacity of the wireless link makes it the bottleneck in most of the cases. In this situation, TCP sender at the fixed host will always try to increase outgoing data rate – causing multiple

buffer overflows at the base station. For every packet drop, TCP window is decreased at least to its half and the dropped packet, which was already transported over the fixed part of the connection, is retransmitted on the complete end-to-end path through the network.

The problem of congestion can be solved by enhancing the functionality of LLE-TCP with the possibility of accessing the receiver advertise window (*rwnd*) (i.e. to exploit the TCP header field on every ACK for specifying the amount of an empty buffer space left at the base station).

Originally, *rwnd* [51] was proposed in order to notify the sender node about the amount of free space left in the receiver's buffer: at any given time, the sender should not send more data than currently allowed by the minimum between the congestion window and the advertised *rwnd*. This prevents overflows in case incoming data rate is greater than packet processing rate at the receiver node.

However, wireless last-mile networks have typically low capacity, which positions the communication link to be a bottleneck (rather than the node's computational or storage resources). Indeed, the maximum size of TCP congestion window in wireless networks is usually bounded by several tens of packets [52]. This makes the mobile node capable of processing such a limited amount of data. Nevertheless, in order to provide full support of *rwnd* functionality, the ARQ agent at the mobile receiver is specified "not to block" any outgoing packets which specify buffer exhaustion.

The main drawback of LLE-TCP implementation in infrastructure network scenario is the increased complexity of the base station. However, the base station does not require TCP layer implementation in a conventional sense. It just requires an access to TCP header fields such as connection port numbers, flow sequence number, ACK flag, and ACK sequence number. These fields can be read with an offset from the beginning of TCP header and are used for TCP ACK generation. Furthermore, for TCP ACK generation ARQ agent simply copies the data extracted

from TCP data packet into the corresponding fields of previously allocated TCP ACK packet template. Overall, the operations performed by ARQ agent at the transport layer are limited to simple offset read/write procedures.

In case the base station does not support LLE-TCP (an aspect which could be negotiated within the network registration procedure), the mobile station can use its standard TCP implementation transparently by simply switching off LLE-TCP software module.

3.4. Multi-hop Network Scenario

Networks comprised of wireless devices capable to operate without a fixed infrastructure constitute a set of wireless networks where data transmissions can cover several hops. Multi-hop wireless communication standards have not been developed yet, as reflected in many research proposals on the topic [53].

In this scenario, LLE-TCP operation is considered at the last hop of a data connection. ARQ agents are attached to the last hop router (LHR) as well as to the receiver node (see Figure 3.6). Each node in multi-hop ad hoc network is assumed to be capable of packet forwarding. Before proceeding with packet forwarding, each node first checks if the next forwarding node is the actual destination of the packet. This can be done by the analysis of routing tables for the correspondence of MAC and IP addresses (ARP tables), since most of the available multi-hop routing schemes contain addresses for at least one-hop neighborhood [54]. In case the forwarding node is the last hop router, it activates the ARQ module attached to its protocol stack.

LLE-TCP operation in the multi-hop scenario is similar to its functionality in the infrastructure network, where in this case the LHR serves as a base station. LHR generates end-to-end TCP ACKs back to the sender node relying on the link layer acknowledgements sent over the last wireless hop.

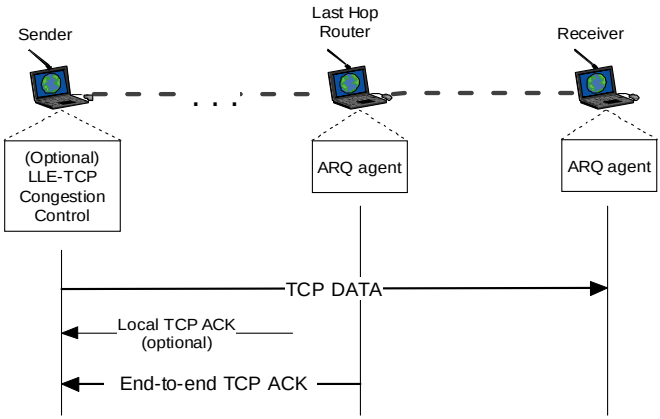


Figure 3.6: LLE-TCP in multi-hop network scenario

However, the congestion control methodology, which is optional for the LLE-TCP approach, is different from the infrastructure network scenario. In a multi-hop network, congestion control is controlled directly at the sender side by using an optional LLE-TCP congestion control module (LLE-TCP_CC). LLE-TCP_CC basically follows the functionality of the ARQ agent described in Section 3.2. It generates local TCP ACKs for the packets which are successfully transmitted over the first hop of a multi-hop connection. However, such ACKs cannot acknowledge successful delivery up to the destination. For that reason, LLE-TCP_CC stores all the outgoing TCP packets in the buffer until their end-to-end delivery is acknowledged by the ARQ agent of the LHR. In order to control the amount of incoming data from TCP, the *rwnd* field of locally generated TCP ACKs is set to one packet in case the LLE-TCP_CC's buffer is not full. Otherwise, *rwnd* is set to be equal to zero (freezing TCP transmissions) which can be resumed by sending single duplicate ACK for the last transmitted data packet with a positive value for the advertised window.

In summary, LLE-TCP_CC overrides the flow control mechanism performed by TCP without introducing any direct modifications to the transport layer. As a result, TCP becomes a controlled

source of packets, providing a basis for the control on the amount of data sent onto the network.

The implementation details of the optional congestion control module are presented in Chapter 3.

3.5. Experimental Results

LLE-TCP is a general solution designed for any network implementing multiple positive ARQ schemes at different layers. However, as reference scenario for performance evaluation of the proposed approach, IEEE 802.11 wireless LAN is chosen as the most wide-spread and well-analyzed environment nowadays. Evaluation results are obtained through simulations in ns-2 network simulator [55] and then validated by experiments on an IEEE 802.11b testbed.

Following the design objectives, the experiments are conducted in the following network scenarios: single-hop, infrastructure and multi-hop networks.

TCP Reno is chosen for comparison as the most common reference implementation of the TCP protocol, currently operating within the protocol stack of the majority of operating systems. The throughput of a single TCP connection is evaluated against the size of the TCP/IP datagram as well as the link error rate, which are chosen as the main factors influencing LLE-TCP performance. The average throughput is measured during the data exchange phase in order to avoid the influence on the results deriving from routing protocols, address resolution protocol and slow start window evolution phase.

3.5.1 Simulation Setup

In order to perform the experiments, the corresponding software modules of the ns-2 network simulator (version 2.28) are enhanced for supporting LLE-TCP functionality. IEEE 802.11b is chosen as the physical layer standard in order to provide full agreement with the results obtained on the testbed. The parameters used in simulation of the wireless links are reported in Table 3.2.

Table 3.2: Simulation Parameters

Parameter Name	Value
Slot	20 μs
SIFS	10 μs
DIFS	50 μs
PLCP preamble + header	192 μs
Data Rate	11 Mbps
Basic Data Rate	1 Mbps
Propagation Model	two-ray ground

3.5.2 WLAN Testbed Setup

The testbed is built with two laptop computers running OS Linux (Fedora Core 3 with kernel version 2.6.9) equipped with IEEE 802.11b Orinoco Silver cards. In order to support the desired functionality, LLE-TCP modules are inserted into Orinoco_cs drivers version 0.13d used by wireless cards.

Iperf (version 1.7.0) [56] performance measurement tool is used for throughput measurements. Experimental results are averaged on 10 runs of 5 minutes each.

Due to limited number of the available resources, testbed experiments are performed only in the single-hop scenario, where two stationary computers located within transmission range are connected using the “ad-hoc” connection mode.

The size of the TCP data packet remains constant within single flow duration, while NO_DELAY socket option is turned on in order to avoid data concatenation performed by Nagle algorithm [57].

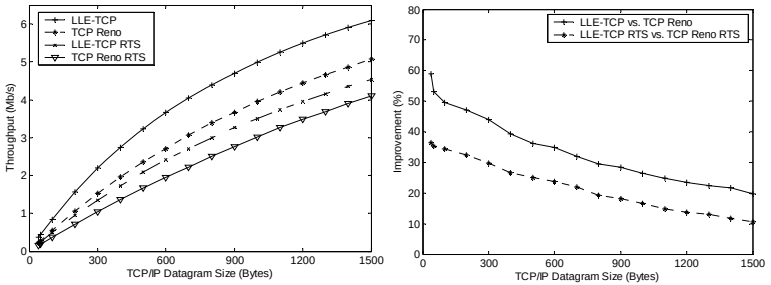
3.5.3 Single-hop Network

The single-hop scenario is built with two nodes located within transmission range: one of them continuously transmits data, while the other one serves as a passive receiver.

Figure 3.7a presents throughput comparison between the proposed LLE-TCP and TCP Reno implementations obtained with and without RTS/CTS exchange employed at the link layer.

The maximum LLE-TCP throughput corresponds to the largest datagram size of 1500 bytes (most common MTU in Ethernet) and is equal to 6.09 Mbps for LLE-TCP. With the same configuration, TCP Reno implementation achieves only 5.08 Mbps. In case of RTS/CTS exchange, LLE-TCP achieves 4.71 Mbps while TCP Reno provides only 4.1 Mbps.

The improvement level achieved by LLE-TCP is inversely proportional to the TCP/IP datagram size (see Figure 3.7b), which derives from the 40 bytes fixed size of the TCP ACKs (including TCP and IP headers). As a result, the level of the improvement is proportional to TCP ACK/datagram size ratio. Being fixed at around 20% for the maximum datagram size, it raises up to 50-70% for the smaller packet sizes. However, the general rule is that as the datagram size tends to the size of TCP ACK the improvement level tends to 100% (in case a TCP ACK is generated for every TCP data packet).

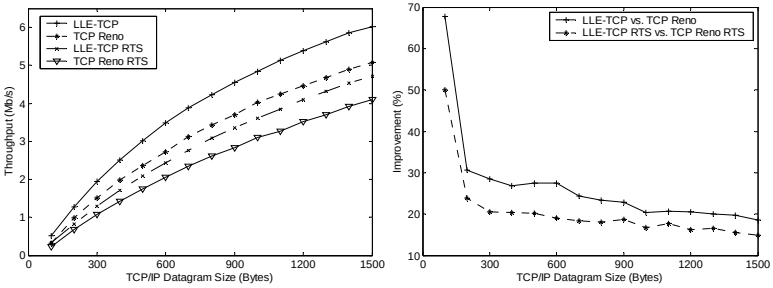


a) b)

Figure 3.7: a) Throughput and b) performance improvement achieved by LLE-TCP against TCP Reno (simulation results).

Results obtained from the testbed experiments (see Figure 3.8a) closely approximate those obtained from simulations. The average difference between simulation and testbed results is around 3%. The improvement level presented in Figure 3.8b has obvious quantitative difference with the results obtained from simulations. This comes from the fact that testbed implementation does not support TCP ACK generation within data packet transmission time. For that reason, the time for TCP ACK generation

and its processing delay by the protocol stack of the sender node correspond to the difference measured in the presented results.



a) b)

Figure 3.8: a) Throughput and b) performance improvement achieved by LLE-TCP against TCP Reno (testbed experiments)

Since LLE-TCP improvement derives from the number of TCP ACKs generated by the receiver, the performance of the proposed approach is analyzed against TCP Reno with one ACK per data packet (One-Ack) as well as Delayed-ACK (DelAck) acknowledgement strategies. Figure 3.9 presents simulation results in the presence of link errors with the TCP/IP datagram sizes fixed to 1 Kbytes.

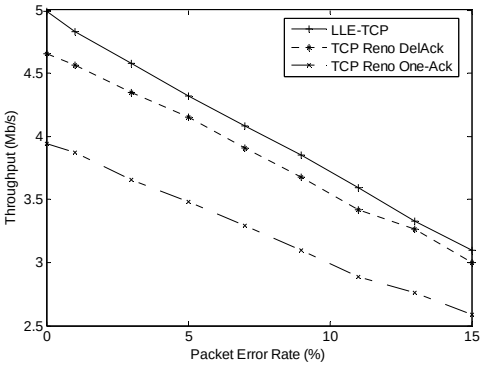


Figure 3.9: LLE-TCP throughput comparison against TCP Reno with different acknowledgement strategies under packet loss conditions

Average improvement achieved by LLE-TCP is equal to 23% for One-ACK and 7% for DelAck scenarios.

Summarizing, we experienced that the improvement of LLE-TCP scheme is highly dependant on TCP/IP datagram size: it ranges from 20% to 100% giving more advantages for applications producing smaller packets. This represents a positive feature of LLE-TCP, since multiple statistical studies show that 50% of Internet traffic includes packet sizes less than 100 bytes [44, 93].

Another observation is that an increase of channel error rate does not greatly modify LLE-TCP improvement level. This derives from the fact that complete elimination of TCP ACK transmission over the wireless link avoids error propagation in case of locally generated acknowledgements. However, data packets are still subject to transmission errors.

3.5.4 Multi-hop Network Scenario

The simulated multi-hop network consists of a variable number of static nodes arranged in a string topology (see Figure 3.6), where communication is allowed only between neighboring nodes due to transmission range limitations. RTS/CTS exchange is turned on in order to avoid the hidden node problem.

Figure 3.10 presents simulation results obtained with a variable number of nodes involved in the multi-hop connection with a TCP/IP datagram size equal to 1000 bytes. LLE-TCP ACK suppression is performed only at the last host hop, i.e. between the last hop router and the receiver nodes. As a result, throughput achieved by LLE-TCP is close to TCP Reno throughput for a large number (>7) hops. However, for a smaller number of hops, it brings an improvement ranging from 10% to 20%.

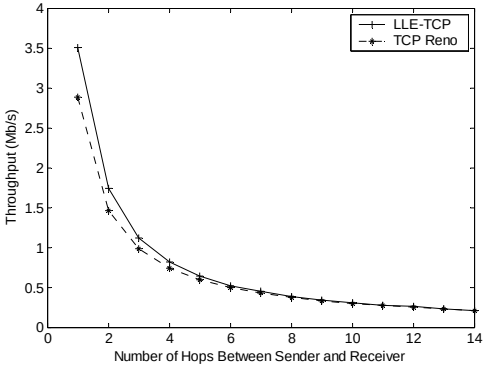


Figure 3.10: LLE-TCP throughput over a multi-hop connection

Figure 3.11 presents throughput achieved by a three-hop connection against the TCP/IP datagram size in static environment while Figure 3.12 evaluates LLE-TCP performance in presence of nodes mobility. In order to simulate mobile scenario, 30 nodes are randomly placed within a 100x100 meters area. The communication distance between neighboring nodes is limited to 22.5 meters. The source and the destination nodes are initially placed within a three-hop distance. Ad-hoc On-demand Distance Vector Protocol AODV [58] is chosen as a routing protocol, while RTS/CTS exchange is turned on for the entire experiment lasting for 1000 seconds.

Figure 3.12 shows that the proposed LLE-TCP approach is also well-suited for mobile scenarios. Indeed, while the forward path of the data flow (from TCP sender to the receiver) is unchanged, LLE-TCP moves TCP ACK generation point to the last-hop router reducing the reverse path length by one hop. This results in better connectivity of the topology leading to fewer route errors and consequent route discovery messages.

The second reason driving performance improvement is in the reduced RTT, which allows the TCP sender faster reaction to changes in the topology or to packet drops due to congestion.

Figure 3.13 presents the results for different acknowledgement strategies in an error-prone scenario. In multi-hop scenario we ex-

tended the set of receiver strategies by adding Dynamic Delayed Ack (DynDelAck) [59] which is transport layer modification designed for performance improvement over static multi-hop wireless links. DynDelAck is an extension of DelAck approach with a variable delay coefficient which depends on the sequence number of the incoming TCP packet. In particular, while DelAck outputs at least one ACK every two received packets, DynDelAck extends this limit up to one ACK for every four packets for persistent connections.

The results presented in Figure 3.13 show LLE-TCP always outperforms TCP Reno when both approaches follow the same receiver strategy. In particular, for low Packet Error Rates (PER) (less than 4%), LLE-TCP brings an improvement of 15.5%, 9.6%, and 4% over TCP Reno for DynDelAck, DelAck, and OneAck receiver strategies, respectively. The reader should note that the reported value of PER corresponds to a singe link between any two neighboring nodes.

For higher PERs, TCP Reno completely degrades its throughput down to zero depending on the delay ack coefficient used in receiver: TCP Reno with DynDelAck and DelAck receivers produces no throughput starting from PER=6%, while TCP Reno with OneAck starting from PER=9%. LLE-TCP reduces the number of hops required for TCP ACK delivery. As a result, LLE-TCP flows remain active even for higher (by 1-2%) PERs.

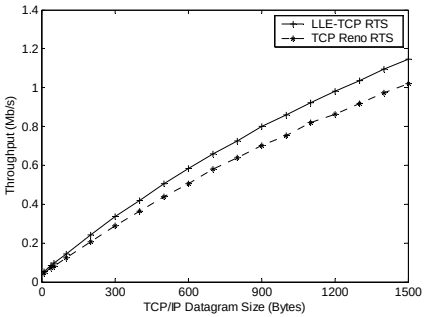


Figure 3.11: LLE-TCP throughput performance over a three-hop connection against TCP/IP datagram size

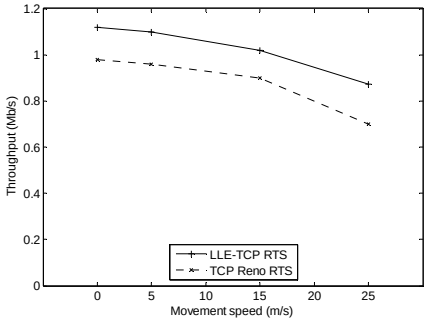


Figure 3.12: LLE-TCP throughput performance in mobile environment

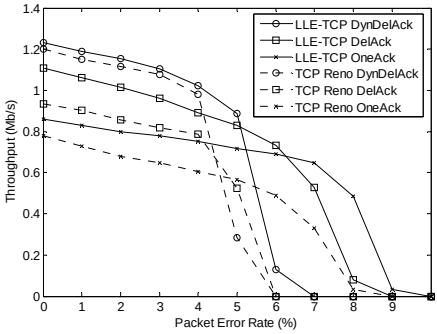


Figure 3.13: LLE-TCP throughput over a three-hop connection under the packet loss conditions

3.5.5 Infrastructure Network Scenario

In the infrastructure network scenario, the TCP source located in the fixed network communicates through the access point with a mobile node located in the mobile part of the network. The configuration of wireless link follow IEEE 802.11b standard, while the parameters of wired link (100 Mbps, 2 ms) mimic the case a coffee shop user connecting to an Internet server located within the city or a state.

The performance results obtained with different packet sizes (Figure 3.14) and in presence of link errors (Figure 3.15) closely approximate the corresponding results obtained in a single-hop

scenario. The main reason for that is that the TCP ACKs generated at the base station are transmitted only over the wired part of the network and not on the wireless link. Moreover, congestion control performed by the base station prevents buffer overflows and congestion related losses.

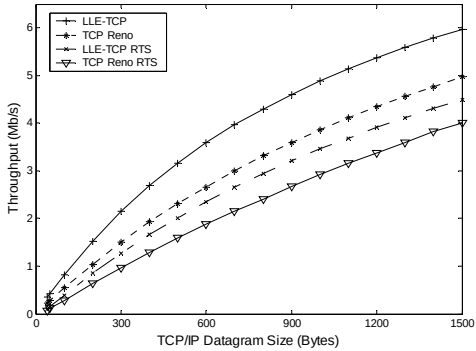


Figure 3.14: LLE-TCP throughput performance in infrastructure network scenario

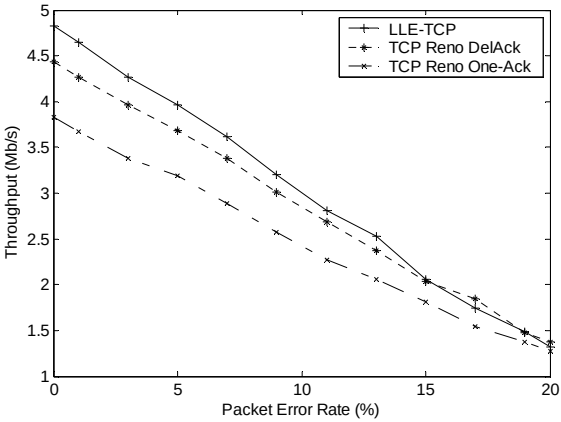


Figure 3.15: LLE-TCP throughput performance under the packet loss conditions

In order to evaluate the behavior of the proposed approach in multi-flow communications, we configured the fixed sender node to initiate multiple connections to the different mobile nodes in the infrastructure network scenario. Mainly, this scenario is targeted to an evaluation of two parameters: fairness and coexistence.

Fairness. In this experiment, acknowledgement suppression is performed for all the flows initiated at the fixed sender following the proposed approach. Results presented in Figure 3.16 show a good level of fairness (95%) calculated according to [60] in terms of the achieved throughput for a scenario when LLE-TCP is the only protocol running in the network. The average individual flow throughput is equal to 0.47 Mbps.

Coexistence. Another important factor that must be considered within the design of a new protocol is coexistence with protocols which are already implemented in the network. The importance of coexistence factor is underlined within the evaluation of TCP Vegas [32], where it is underlined that TCP Vegas can operate properly only in case it is the only protocol running in the network. Otherwise, TCP Reno grabs all the available bandwidth, dramatically reducing the throughput of TCP Vegas down to zero.

Results for the scenario, where only one out of ten flows follows LLE-TCP approach while other flows communicate using standard TCP (Figure 3.16), show good agreement with the design considerations. Standard TCP flows (from 1 to 9) maintain relatively stable throughput with the average of 0.30 Mbps and fairness index of 93%, while the throughput of the 10-th LLE-TCP flow is significantly higher and equal to 0.54 Mbps. Such performance advantage mainly comes from the improvements due to ACK suppression and reduced RTT of the connection.

As an outcome of the experiments, we observe the stability of LLE-TCP in multi-flow communication scenarios as well as good coexistence with the standard TCP. This underlines the possibility for an incremental deployment, i.e. there is no need to replace already existing products (drivers or wireless cards) which do not

support the LLE-TCP functionality while bringing the approach to the market.

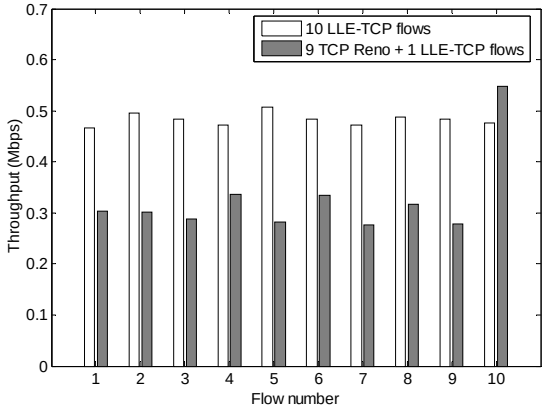


Figure 3.16: LLE-TCP fairness and coexistence with TCP Reno

The entire set of experiments presented above show the benefits for the node using LLE-TCP protocol. In order to present the benefits for the community of users, we simulated the scenario where 10 users connect to the server located in the fixed Internet via a base station. The users are equally divided into two sets: each of the first 5 users (set one) after downloading 2Mbytes stop their communications, while the other 5 users (set two) keep downloading for the entire simulation duration which is equal to 100 seconds. In the first case, the users from the set one use LLE-TCP approach, while in the second case standard TCP Reno protocol is used. The users of the set two are always communicating using TCP Reno.

The main metric showing the benefits for the community is the amount of data downloaded by all the nodes within the entire simulation time. However, taking into account set 1 always downloads the same amount of data equal to 5*2Mb, we present data delivery results only for the set two nodes (see Figure 3.17).

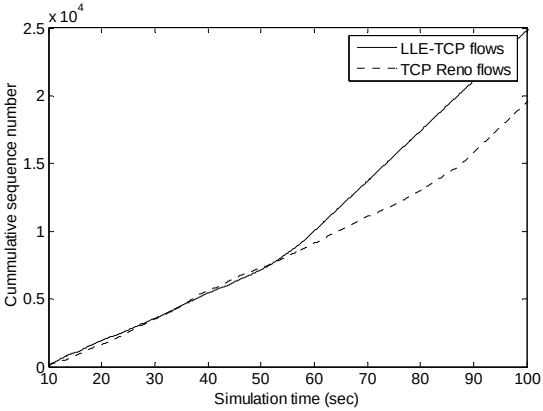


Figure 3.17: LLE-TCP cumulative performance

In case LLE-TCP is employed, set 1 ends downloading after 55 seconds of simulation time, while in the other case, with TCP Reno, it requires 87 seconds. As a result, in the first case set 2 succeeds on a global reception of 25 Mbytes packets, while in the second case only 19.5 Mbytes are received.

This is another argument in favor of an incremental deployment of LLE-TCP, which will ensure performance benefits not only for the users which already updated their products with the proposed approach but also for those who did not.

3.6. Conclusions

This chapter presents a novel yet generic approach for performance enhancement of TCP over wireless networks. Performance improvement comes from cross-layer optimization of ARQ schemes employed at different layers of the protocol stack.

The proposed solution avoids TCP ACK transmission over the wireless link through local generation of ACKs at the sender node (in a single-hop scenario) or at the base station (in the infrastructure scenario).

Cross-layer ARQ is not limited to wireless networks scenarios. If fact it can be implemented inside any protocol stack employing multiple ARQ schemes at different layers. However, the main

benefits are achieved over wireless links due to heavy overheads added at the link and physical layers.

The evaluation of the proposed approach is performed via simulations as well as testbed experiments in IEEE 802.11 WLAN scenario. Results presented for single-hop, multi-hop and infrastructure network scenarios demonstrate an improvement in the throughput in the range of 20-100% – depending on the packet size used by the connection and the number of wireless hops along the end-to-end data path.

Summarizing, main improvements of LLE-TCP are:

Throughput and medium busy time. The reduction of medium busy time for the transmission of TCP ACK packets over the wireless link clearly brings to a corresponding relevant improvement in terms of TCP throughput. Improvement level highly depends on the size of TCP segment as well as on the channel error rate.

Channel error rate. A passive reduction of channel error rate comes from the absence of TCP ACK packets on the reverse channel (they are generated locally at the sender side, and therefore channel errors have no impact on them).

Round Trip Time (RTT) reduction. TCP ACK suppression reduces RTT by the time required for TCP ACK transmission over the wireless link. Since TCP performance is reversely proportional to the RTT of a connection and the evolution of the congestion window is based on the receiver feedback, faster feedback allows faster reaction to packet losses.

Congestion control. An LLE-TCP congestion control module, optionally inserted into the sender's protocol stack, enables to shift the point of congestion control from TCP to the last hop router. Such technique allows the implementation of custom control mechanisms in multi-hop wireless networks. Reference [61] presents an example of such congestion control based on link layer capacity estimation of the end-to-end data path.

Fairness & Coexistence. Together with the performance advantages, LLE-TCP scheme ensures a good level of fairness as

well as proper coexistence with standard TCP Reno protocol. Moreover, it provides benefits not only for the users employing LLE-TCP, but also for those who do not.

As a result, an incremental deployment which can be performed over an existing and already operating network provides a direct way for improving the overall data transfer performance of the network.

If everything seems under control,
you're just not going fast enough.

- Mario Andretti

Chapter 4

4. Cross-Layer Congestion Control

4.1. Introduction

One of the main limitations present in the IEEE 802.11 standard is the restriction of the ad hoc network to the case when all the stations are located in range of each other. However, ongoing research overcomes such limitations, allowing data delivery over an end-to-end path which can consist of several wireless hops. In this chapter we address the general case of data transport over an ad hoc multi-hop wireless network.

Moreover, here we consider the problem of network congestion as the main reason of potential performance degradation, while aspects related to the nature of wireless links, such as limited bandwidth, increased latency, channel losses, mobility, etc., are neglected.

Congestion occurs when the amount of data sent to the network exceeds the available capacity. Transmitted data start to be dropped when available buffer resources, which are physically limited, are exhausted.

Such situation decreases network reliability in the sense of service provisioning for data communications. Transport level protocols improve reliability by implementation of different error

recovery schemes. However, they could lead to excessive data retransmissions, reducing an important parameter such as network utilization, while at the same time increasing latency in data delivery.

The core reason for network congestion is the amount of traffic emitted to the network. For such reason, the proposed solution for congestion avoidance is to control (and possibly optimize) the amount of traffic sent onto the network, considering limited availability of network resources.

The rest of the chapter is organized as follows: Section 4.2 presents an insight related to the nature of congestion, while Section 4.3 outlines the state-of-the-art in the field of TCP adaptation to wireless links. The proposed approach is detailed in Section 4.4, and experimental evaluation of its performance is presented in Section 4.5. Finally, Section 4.6 draws some conclusions and outlines of future work on the topic.

4.2. Nature of Congestion

In TCP/IP reference model no layer has complete and real-time information about available network resources over the multi-hop path where the communication is performed. For that reason, traffic sources can avoid network congestion based on network feedback, which is obtained as a reaction to a certain amount of data being sent on the network.

TCP congestion control is performed on an end-to-end basis. The receiver provides an acknowledgement (ACK) feedback back to the sender. Relying on the information provided by ACKs, the sender can detect which packets are lost.

The congestion control algorithm employed by TCP is window-based [62]. The size of the congestion window *cwnd* corresponds to the amount of data the sender is allowed to output to the network without acknowledgement. Congestion window evolution is the key mechanism for TCP congestion control. TCP uses additive increase and multiplicative decrease strategy for its window adjustment according to network conditions. The main phases of TCP window evolution are presented in Figure 4.1.

The connection is initiated with window size equal to one packet (1 MSS – Maximum Segment Size). Then, *cwnd* is increased exponentially for every non-duplicate ACK reception until the Slow Start Threshold (*ssthresh*) is reached. Prior the connection establishment, *ssthresh* is set to an initial value, which depends on the implementation of the protocol stack, and then adjusted on the basis of the estimate of the network capacity. This technique is called slow start phase.

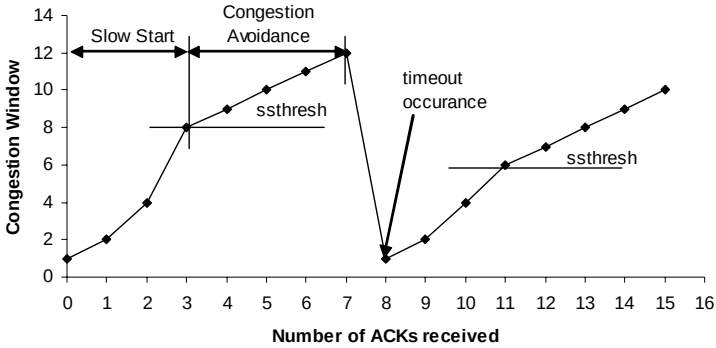


Figure 4.1: TCP congestion window (cwnd) evolution

When the *ssthreshold* is reached, TCP enters congestion avoidance phase. The window is increased linearly by one packet for each received ACK. The window growth in this phase is limited to a maximum window size, negotiated between sender and receiver during connection establishment and then updated on fly during the communication process (the receiver's advertised window).

There are two ways for TCP sender to detect data loss occurred on the communication link: reception of duplicate ACKs (dupacks) and timeout occurrence. In the first case, a dupack is generated by the receiver upon reception of an out-of-order packet, detected through the analysis of sequence numbers of incoming packets. Duplicate ACK reception triggers congestion window reduction to the half of its current size. In the second

case, upon the timeout occurrence, TCP reduces the size of *cwnd* to its initial value (equal to 1) entering slow start phase.

In summary, TCP increases congestion window until the amount of traffic present in the network exceeds the available bandwidth capacity. As soon as the packet gets lost due to congestion TCP window is reduced to its half.

Moreover, TCP was originally designed for wired networks where packet losses occur mostly due to congestion; for that reason, congestion avoidance/recovery is the only reaction of TCP to losses [48]. As a consequence of high error rate in wireless networks [16], TCP reaction to frequent packet losses severely limits the congestion window and thus underestimates the capacity of the networks, leading to non-optimal performance.

The performance of TCP directly depends on the window size, which optimal value should be set to *bandwidth*delay* product of the entire path of the data flow [63]. The excess of this threshold does not bring any additional performance enhancement, but only leads to increased buffer requirements at network routers.

At this point, it is necessary to underline the problems which arise in TCP communications over IEEE 802.11 networks.

The authors of [39] produced an evaluation of TCP performance in wireless multi-hop network, underlining two major problems: instability and unfairness. The observed instability in the performance is present even in the simplest scenario with only one data flow over a multi-hop connection. The main reason for that is touching TCP timeouts, which forces TCP to follow the slow start period greatly degrading the performance through congestion window reduction. The second phenomenon observed in [39] is unfairness, which happens between two TCP data flows that occupy the same number of hops on the same path: the flow started later in time will gain full bandwidth advantages, degrading the performance of previous flow down to zero. The unfairness phenomenon mentioned above is mainly caused by improper tuning of TCP and link layer retransmission timeouts, multiple collisions present on the link layer as well as well-known 802.11

MAC unfairness when the last succeeding station is always favored due to the employed exponential backoff scheme.

A new metric, called expected throughput, is introduced in [64] and used as a bound for comparison of the achieved performance for existing TCP implementations in wireless multi-hop scenarios. The simulations show that the performance of TCP, being far from the desired level, is unacceptable in several applications. Similar results are also presented in [65].

The problem of network congestion exists almost from the early beginning of the Internet [57]. Nowadays, many researchers underline that the problem of congestion is even more critical now rather than in 1980s. A relevant theoretical work on the topic presented in literature is [50], where Sally Floyd and Kevin Fall promote the usage of end-to-end congestion control algorithms for the design of future protocols, in order to avoid network collapse due to congestion. Similar results are obtained by the authors of [66] using game-theoretic approach.

4.3. Available Solutions

During the past years, a relatively strong effort of the research community was devoted to TCP adaptation to the wireless multi-hop network scenario, with the main focus on performance optimization, aimed at enabling uninterrupted network service provisioning.

The majority of the available solutions which modify congestion control algorithm of TCP can be logically classified into three following categories: 1) modifications of TCP based only on the information available at the sender node; 2) solutions which enhance the previous category by allowing network feedback; and 3) approaches which obtain bandwidth-delay information by introducing different capacity measurement techniques at the transport layer.

4.3.1 TCP modifications

One of the first approaches to perform precise RTT (Round Trip Time) estimation is TCP Vegas [32, 67]. First, TCP Vegas reads

and stores inside TCP header the system clock every time a segment is transmitted. Receiver echoes it back without performing any modification. Then, upon ACK arrival, the sender uses the stored timestamp for RTT calculation. The congestion avoidance algorithm is based on the analysis of the actual throughput achieved by the flow through its comparison with expected throughput. The expected throughput is calculated using measured RTT value and current size of the congestion window. In case the actual throughput is much less than the expected value, TCP Vegas decreases the size of the congestion window assuming that it is sending more data than the available network bandwidth.

Another approach presented by authors of [68] introduces congestion window adjustment based on an end-to-end bandwidth estimation technique. The key idea is in continuous measurements of the rate of returning ACKs at the TCP sender side. After congestion occurrence, the source running TCP Westwood attempts to set a slow start threshold and a congestion window on the basis of the effective bandwidth estimation.

A set of protocols for congestion avoidance named Adaptive Transport Layer (ATL) is presented in [69]. ATL consists of two adaptive transport layer protocols: ATL-TCP for reliable communications and ATL-UDP which is used for multimedia data delivery. The main idea of this approach is to allow more freedom in congestion window evolution. The dynamic adjustment algorithm assumes to obtain the information on bandwidth and RTT from the link layer. However, this work does not mention details on how such information is obtained by the link layer.

In summary, the solutions within this category attempt to conquer the roots of the problem. The main problem associated with poor performance of transport protocols over wireless networks lies in the fact that they are designed for wired networks – without taking into account limitations of the wireless scenario. In order to solve this problem, the presented solutions redefine transport protocols, replacing them with versions which are designed

considering the different characteristics of the wired and wireless scenarios.

Obviously, the solutions within this category provide reasonable performance improvement if compared with traditional wired implementations of transport layer protocols. However, the main disadvantage – which prevents wide deployment of the proposed approaches – lies exactly in the requirement for modification of a standardized and widely deployed transport protocol such as TCP. Thus, a huge effort from joined cooperation of industry and standardization committees is required to bring the proposed modifications to the end-user.

4.3.2 Explicit Feedback Solutions

Data communication over wireless networks is far from being identified by a simple two-hop scenario. The path of data delivery in most cases consists of several hops with different capacity characteristics. These characteristics include not only communicational parameters such as available bandwidth and delay, but also parameters associated with nodes on the data path – their memory and computational resources.

Solutions of this category allow network to be aware of pending data transmission between any pair of nodes. The main idea is to allow intermediate nodes on the data path to dynamically inform the sender about the amount of resources available from the network. Relying on such feedback, the sender can adjust the amount of data sent on the network in order to avoid congestion occurrence.

Random Early Detection (RED) [70] is a scheme which is proposed to deal with network congestion through explicit signaling to the source about growing probability of congestion occurrence. RED is designed to be implemented in intermediate routers, where congestion notification is based on queue monitoring. RED defines two buffer occupancy thresholds: low threshold and high threshold. When the average size of the queue length exceeds the low threshold, the packets start to be dropped with linear probability, proportional to the average queue length. In case

this length exceeds the high threshold, all incoming packets are dropped.

Congestion is detected by RED through the analysis of the actual queue length, comparing it to the predefined low and high threshold values. While operating with a queue size between these two thresholds, RED informs the sender node about growing probability of congestion occurrence by marking the incoming packets using a specially defined bit in the packet header. Being notified of the network congestion, TCP sender can perform congestion avoidance through congestion window reduction.

Packet drops performed by RED are designed to the purpose of compliance in operation with TCP sources which do not support RED framework.

In Explicit Congestion Notification (ECN), presented in [71], IP packets can deliver ECN notification using ECN bit introduced in the packet header. TCP header is modified as well, in order to support following indication of ECN-enabled support of TCP implementation as well as for ECN communication between TCP sender and receiver.

Another approach which reduces the mismatch between TCP window and available bandwidth-delay product, called Explicit Window Adaptation (EWA), is presented in [72]. Similar to RED congestion detection, it is based on the analysis of the available buffer size left at the edge routers. EWA attempts to adjust the size of the congestion window explicitly on the data path through modification of receiver's advertised window field of the packet. This scheme generalizes window advertising technique allowing the specification of available buffer space not only by the receiver but also by intermediate routers.

Summarizing, solutions presented in this category implement different explicit feedback techniques. All of them rely on the available buffer space at intermediate nodes, which forces them to depend more on the size of allocated memory rather than on the available capacity of the communication links. Such a tradeoff leads to an increased time for response to the congestion.

The main reason for congestion occurrence is the production of more traffic than the available resources for its delivery over the network. In case of a connection covering multiple hops of the network, data transmission is performed on a hop-by-hop basis: data is stored in the buffer, waiting, while node obtains access to the physical medium. In case the amount of incoming data overcomes the node's forwarding capacity, input buffer size would grow with a rate which is approximately equal to the difference between incoming and outgoing data rates. When the buffer is full, the node starts to drop incoming packets and congestion is detected.

For that reason, solutions based on the analysis of the queue length react to congestion later if compared with solutions which analyze the difference between capacities of the incoming and outgoing communication links.

4.3.3 Transport Layer Capacity Measurement

The ideal case for the source node is to have real-time capacity information of the entire data communication path for the entire duration of the connection. In order to approach to this ideal case, many proposals target an enhancement of transport layer protocols with capacity probing techniques.

Total capacity of the link is composed of bandwidth and delay components.

The delay associated with an end-to-end connection can be easily obtained by TCP through the calculation of the time difference between packet transmission and reception of the acknowledgement generated by the receiver for such packet. The obtained value corresponds to the cumulative delay experienced in forward and backward directions of the connection path. However, it is shown that acknowledgement-based RTT estimation could perform poor under specific circumstances [73]. More accurate RTT estimation is performed when sender includes timestamps into outgoing packets, which are echoed back without any modification at the receiver. The recommendation for high performance

extension and finer RTT estimation for TCP protocol is presented in [73].

The estimation of the bandwidth component is not that simple. Different bandwidth estimation solutions available in literature can be logically divided into passive measurements and active probing according to the algorithm they employ [74]. Passive solutions build their measurements based on the trace history of an existing data transmission. Such solutions are limited to the network paths that have recently been used for communication. On the other hand, active probing provides faster and more accurate estimations, while having the potentiality for exploration of the whole network.

The packet pair mechanism is a reliable technique for bandwidth measurement. The key idea of the approach is based on the measurement of inter-arrival time between two back-to-head packets transmitted through the entire connection path. A simple calculation based on inter-arrival delay and packet size returns the bandwidth of the link. A detailed description of packet pair measurement technique is presented in [75, 76].

The main drawback of packet pair technique is the dramatic reduction of the estimation accuracy in the presence of cross-traffic. CapProbe [77, 78] is a technique which improves packet pair measurements by filtering only those pairs of the packets which have minimal end-to-end delays. This method excludes those packet pairs which are influenced by cross-traffic in intermediate queues.

Summarizing, capacity measurement techniques presented within this section have obvious drawbacks which prevent their successful deployment:

- They simply do not work under certain circumstances, such as under presence of cross-traffic which is both intensive and non-reactive [77];

- Probing network bandwidth requires an insertion of additional traffic, which reduces the already limited network resources;

- The bandwidth information becomes available to sender after the time required for a roundtrip propagation of the probe sequence over the network path;

- Additional computation resources are required from the sender for statistical processing of the measured data.

4.4. Cross-Layer Congestion Control in Multi-hop Wireless Networks

This section presents Cross-layer Congestion Control for TCP (C^3TCP) for congestion control over wireless local area networks where data delivery is performed over multiple wireless hops. To the purpose of explanation of the proposed congestion control ideas, we consider a string network topology as the simplest topology which approximates the multi-hop scenario [79]. A four-node example of the string topology is presented in Figure 4.2.

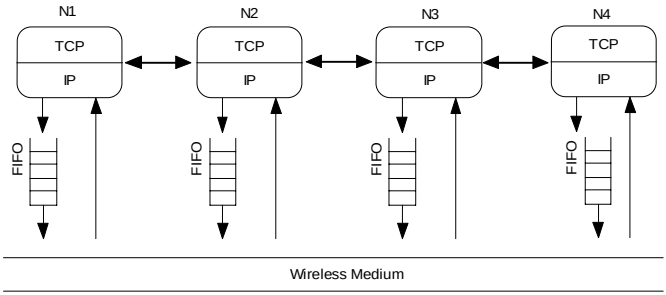


Figure 4.2: String topology for multi-hop wireless network

This topology allows direct communications only between neighboring nodes which is due to the limitations in transmission range. It is assumed that every node has an appropriate output FIFO buffer, where the link layer packets are queued while the wireless medium is busy. Input queuing is omitted for presentation simplicity without any influence on the results.

The second assumption consists in the availability of a routing path to every node of the multi-hop network: details related to route discovery are left out of the scope.

4.4.1 Bandwidth Measurement

Basic medium access mechanism specified by IEEE 802.11 standard is the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) with binary exponential backoff. The node is not allowed to transmit until the medium becomes free (i.e. no pending transmissions of other nodes). As a result, the whole bandwidth is divided among nodes which share the same (wireless) medium.

An optional part of the standard specifies RTS/CTS (Request-To-Send/Clear-To-Send) exchange, which takes place prior transmission at the link layer of a data frame and its acknowledgement. This scheme is designed as a solution to the hidden terminal problem. Consequently, it considerably reduces data losses caused by collisions.

From the considerations described above, one can conclude that finally the bandwidth is shared among the nodes which are located in the range of the sender as well as the receiver nodes.

The available bandwidth B for transmission of a certain amount of data can be obtained knowing the size of data D and time T taken for transmission of such data over a specific link:

$$B = \frac{D}{T} \tag{4.1}$$

The detailed framework for a single data packet transmission is presented in Figure 4.3. Having data to send at time T_{in}, the source node initiates the medium access procedure: it senses that medium is already occupied by another transmission and falls into exponential backoff with the initial size of the backoff window; during the next time of sensing the medium appears to be free, which means that the source node is allowed to initiate the transmission with RTS frame for medium reservation. Then, after Short Inter-Frame Space (SIFS) the destination node replies with CTS updating the Network Allocation Vector (NAV) of the nodes which are located within the range of the receiver.

At time T_{out}, the sender initiates data frame transmission onto the physical medium. Upon the successful reception of the data frame, the destination replies with positive acknowledgement.

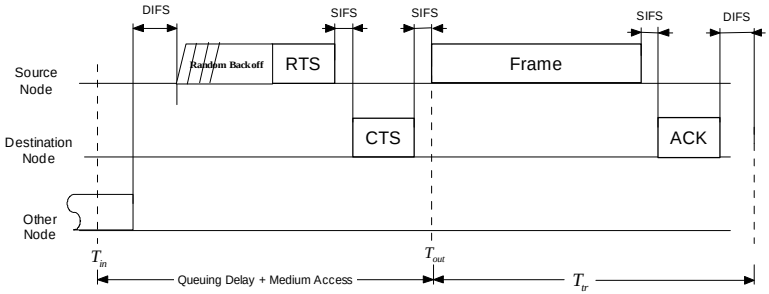


Figure 4.3: IEEE 802.11 basic medium access mechanism and data delivery procedure

Taking into account the described data delivery framework, the time required for the transmission of a single data packet using CSMA/CA can be obtained as follows:

$$T = T_{out} - T_{in} + T_{tr} \tag{4.2}$$

where the difference T_{out}-T_{in} includes data queuing delay corresponding to the time the node was waiting for all other nodes to finalize their pending transmissions as well as channel to channel access delay using random backoff and optional RTS/CTS exchange. A similar way of analysis was first presented by Giuseppe Bianchi in his theoretical model for the performance analysis of IEEE 802.11 DCF [80].

In order to calculate the time T_{tr} required for data frame transmission and its corresponding acknowledgement, it is necessary to take into account the framework employed by the physical layer.

Data encapsulation performed at physical and link layers is presented in Figure 4.4. Physical Layer Convergence Protocol (PLCP) preamble is transmitted prior any communication between nodes, for the synchronization of their physical circuits. PLCP Header contains signaling information about the subse-

quent MAC layer frame, such as length and bit rate. PLCP preamble and header are always transmitted at the basic rate, regardless of the maximum bit rate available on the medium.

The MAC header, being transmitted at the data rate specified in the PLCP header, contains information about the delivered data on the link layer. FCS field finalizes frame containing CRC information related to both MAC header and frame body.

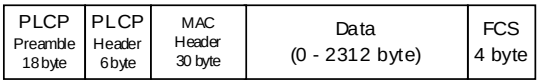


Figure 4.4: Physical and link layer encapsulation

According to physical and link layer specifications, the time required for data packet delivery (including the data frame and the corresponding ACK) is calculated as follows:

$$T_{tr} = T_{data} + SIFS + T_{ACK} + DIFS \tag{4.3}$$

$$T_{data} = \frac{PLCP.Preamble + PLCP.Header}{Basic.Rate} + \frac{MAC.Header + FCS}{Data.Rate} + \frac{Data}{Data.Rate} \tag{4.4}$$

$$T_{ACK} = \frac{PLCP.Preamble + PLCP.Header}{Basic.Rate} + \frac{ACK.Header + FCS}{Data.Rate} \tag{4.5}$$

where DIFS is the Distributed Coordination Function Inter-Frame Space.

The time required for a single data frame transmission T_{data} includes the term which corresponds to physical preamble and header (always fixed size and transmitted at the basic rate). For example, for basic rate of 1Mbps, the time required for the transmission of physical preamble and header is equal to 192μs. This

means that the value of the first term can be calculated once and then reused for subsequent calculations.

The second term corresponds to the time required for MAC header and CRC information, which could be theoretically transmitted at any rate supported by the standard. However, since most of the rates specified by the standard are obtained from the maximum one through simple operations, it is possible to pre-calculate them, for example, by building a short table with values calculated for the entire set of possible data rates for a given physical layer extension of the standard (a, b or g).

The algorithm for bandwidth measurement is the following:

1. Store the timestamp T_{in} for every data arrived to the link layer for further transmission. The data can arrive for forwarding from other nodes or it can be generated locally by upper layers of the protocol stack.
2. Prior actual data frame transmission, at time T_{out} calculate the time taken for packet delivery including queuing and packet transmission time T_{tr} using Equation (3).
3. Calculate the bandwidth experienced by the packet using Equation (1).

The bandwidth calculated by the presented algorithm includes following components: queuing delay, the time required to gain medium access and the delay associated with physical and link layer header transmission. It means that entire overhead which occurs before any actual data transmission over the physical medium is considered to be a factor which reduces the available bandwidth on the link.

The overhead added at physical and link layers is directly related to the utilization level. For example, utilization of wired local area networks is relatively high (97%) [81] while the utilization of wireless IEEE 802.11 networks is on low level [82].

4.4.2 Delay Estimation

Transport layer protocols provide end-to-end data delivery without having any knowledge on the network structure, like number of hops, parameters of communication link and so on, assuming

to have a single data pipe between end nodes. In order to reach the maximum performance, transport protocols ideally should fill the data pipe with its bandwidth-delay product.

Considering the four-node example presented in Figure 4.2, let us assume that node *N1* wishes to communicate with node *N4*. The data generated by the transport layer of node *N1* is placed in the output queue on the link layer. Then, after node *N1* gains medium access, the data packet can be transmitted to node *N2*. Node *N2*, upon the reception of the data packet which should be forwarded to the next node of the string topology, performs its medium access procedure – during which the packet can be required to wait in the output queue of node *N2*. Similar procedures are performed by the node *N3* as well, before the data reach the destination node *N4*.

Finally, in our multi-hop scenario, queuing delay is present on all the nodes except the destination node. Such queuing delay does not correspond to the length of data pipe between end nodes *N1* and *N4*. On the contrary, the length of this pipe consists only of the time required for actual data transmission at the physical layer through all the links along the communication path.

Most solutions for optimization of the TCP performance through congestion window adjustment rely on RTT as a delay measurement parameter. Such a way of delay measurements approximates forward and backward links within a single data pipe.

However, TCP assigns different communicational purposes to links in forward and backward directions. Thus, forward direction is used for transfer of application payload while the backward direction serves for the functionality of the TCP acknowledgement scheme.

In the proposed delay estimation technique we differentiate between forward and backward delays. Forward delay contains the length of the data pipe between sender and receiver nodes while backward delay measures the time required for the delivery of TCP ACK packets.

1) *Forward delay*. According to the considerations described above, the single-hop forward delay experienced by a data burst

includes channel access delay, time required for data delivery on the physical layer including related physical and link layer overhead, but excluding link layer queuing delay.

Forward delay is calculated using (2) setting T_{in} to be equal to the time the packet leaves the queue preparing for actual transmission on the link layer. Such estimation avoids the insertion of link layer queuing delay into the T_{out}-T_{in} component.

Most of the transport layer measurement techniques include also queuing delay along the entire path experienced by the data packet at the link and IP layers: the end-to-end data pipe is considered artificially longer than it actually is. In other words, a simple insertion of an additional traffic increases the bandwidth-delay product through an increased measured delay. However, the bandwidth-delay product should be decreased in this situation through reduction of the available bandwidth component while leaving forward delay unaffected.

Considerations described above bring advantages for link layer per-hop delay estimation if compared with end-to-end transport layer measurements.

2) *Backward delay*. The reliability of TCP is achieved through implementation of a positive acknowledgement scheme. The receiver acknowledges successful data delivery with TCP ACK packets going back towards the sender.

On contrary with forward delay measurement technique described above, TCP ACK delay does include both transmission and queuing delays, i.e. it is equal to the difference between the time the TCP ACK packet was generated by the receiver node and its reception by the TCP sender. Backward path delay on each single link is calculated using (2).

The proposed method for estimation of the delay in forward and backward directions allows the TCP sender to adjust the amount of outstanding data to the bandwidth-delay product in the forward path, while considering the backward path as a simple delay line for TCP acknowledgement reception.

4.4.3 "Options" Support for IEEE 802.11

The previous two sections describe techniques for available bandwidth and delivery delay measurements at the link layer. Such measurements are performed by wireless stations and correspond to the neighboring links which in fact constitute a single shared medium the nodes belong to.

Wireless multi-hop networks perform data communication over several short-range links. An evaluation of the capacity experienced by a certain data packet can be obtained by the analysis of capacity of the individual links: the end-to-end bandwidth $B_{end\text{-}to\text{-}end}$ over an n-hop path is equal to the bandwidth of the bottleneck on the path, while the end-to-end delay $D_{end\text{-}to\text{-}end}$ is obtained by the superposition of delays D_i introduced by individual links.

$$B_{end-to-end} = \min(B_1, B_2, ..., B_n) \tag{4.6}$$

$$D_{end-to-end} = \sum_{i=1}^{n} D_i \tag{4.7}$$

The obtained values for end-to-end bandwidth and delay should be forwarded to the source producing traffic to let it implement congestion control based on performed measurements. In order to support such functionality, we propose to extend IEEE 802.11 MAC protocol by allowing the specification of optional fields inside the MAC header.

Optimization of IEEE 802.11 link layer is a hot topic. Huge amount of optimization solutions are proposed by the research community which introduce an enhanced signaling for optimization of the link layer performance. In most cases, enhanced signaling requires the modification of the standardized MAC protocol or the specification of an additional protocol. Indeed, research work continues to go on after the specification of a particular protocol has taken place. Many novel approaches and optimization solutions appear which can not be easily applied to the existing specification. The idea to include a universal way for inserting additional information has touched most important protocol speci-

fications nowadays. Thus, IPv4 [83], IPv6 [84], TCP [43] specifications contain the support of optional fields.

The proposed modifications are aimed at enabling optional support within the IEEE 802.11 MAC header (Figure 4.5).

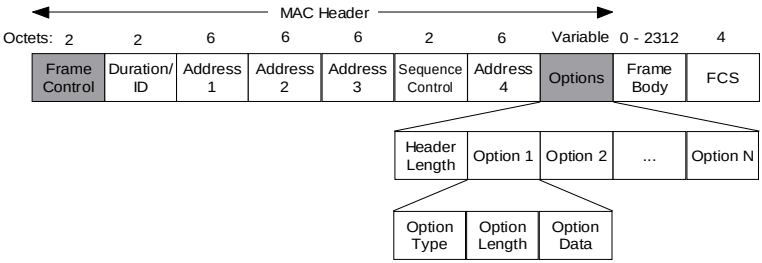


Figure 4.5: “Options”-enabled IEEE 802.11 data frame

“Options” is a variable length field which extends standard MAC header. It consists of “Header Length” field which specifies the entire length of the MAC header, including the list of options. The length of the header is required to perform separation of the data encapsulated into the frame from the MAC header.

Each option consists of option type, length in octets and data. Length is required to handle the case when a node does not support the corresponding option. The knowledge of the option’s length makes skipping the current option easier, jumping to the next one for processing.

Current wireless devices do not support optional fields within the MAC layer header. In order to provide backward support within the existing IEEE 802.11 standard specification a new type of packet is introduced, since an options-enabled data frame should be of a different type with respect to a normal data frame. For that purpose, one of the reserved types in the Frame Control field of the MAC header of data frames can be used. For example, the type equal to ‘10’ can be used for data transmission, while the subtype ‘1000’ indicates options-enabled data frame.

Backward compatibility with standard IEEE 802.11 devices also requires the specification of communication of options-enabled nodes with those which implement standard MAC. In

case options-enabled node wishes to communicate with another node, it should first try to establish communication using the new options-enabled data frame. If the destination node does not support the new type of data frame, it will just simply drop the received frame. The sender node will detect the frame loss through the lack of positive acknowledgement from the receiver after timeout occurrence, which is equal to SIFS + ACK transmission time. The second time the sender node should use standard data frames to communicate with such node.

It could happen that the first communication using options-enabled frames could be unsuccessful because of information corruption during data delivery in the channel. For that reason, the sender node should periodically attempt to communicate using new types of frames.

The presented backward compatibility technique is easy to implement, however it is not optimal in the sense that the sender node needs to probe the destination. These probes could be unsuccessful in case the destination does not support MAC-layer options, producing additional overhead which reduces the bandwidth utilization and increases the packet delivery delay. In order to avoid such additional overhead, the information about options-enabled capability could be encapsulated into route discovery protocol allowing nodes to have knowledge about which type of frame to use in advance.

Options-enabled data frames should not be used in case the node does not transmit any options inside the header.

4.4.4 The Proposed Approach: C^{3}TCP

Figure 4.6 presents an example of TCP communication over a 3-hop wireless network. Sender node *N1* initiates the transmission by sending a TCP data packet to node *N4* over the string topology. Upon reception of the TCP data packet, the link layer of node *N1* performs bandwidth and delay measurements for link *L1*. Then, it includes the measured information into the corresponding optional fields inside the MAC header. Node *N2*, after the reception of the data frame from node *N1*, performs the same

measurements for link L2. Then, it takes the minimal value for the measured bandwidth of links L1 and L2 and updates the bandwidth option in the MAC header. Delay experienced by the data on the link L2 is summed with the delay on the link L1.

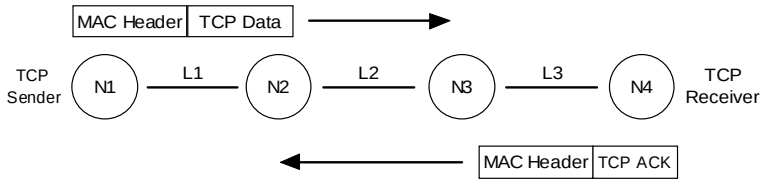


Figure 4.6: C^{3}TCP usage scenario: TCP connection over 3-hop wireless network

When the TCP data packet reaches the destination node *N4*, its MAC header contains bandwidth and delay experienced by the packet on the path through links *L1 – L2 – L3*. TCP receiver in *N4* replies with TCP ACK back to the sender indicating successful data packet reception. This TCP ACK is also encapsulated by link and physical layer headers, including the bandwidth-delay information obtained by *N4* during TCP data packet reception. Such information is simply echoed back using the appropriate optional fields.

Upon the reception of the TCP ACK packet, sender node *N1* will have the bandwidth and the delay for both transmitted TCP data and TCP ACK packets. Based on the obtained information, the sender can adjust the outgoing traffic using calculated bandwidth-delay product. The bandwidth is taken only from TCP data packet propagating in forward direction, while the delay is obtained as a sum of propagation delays of TCP data and TCP ACK packets.

The goal of the presented approach is to avoid any changes at the transport layer of the protocol stack. For that reason, an additional module, called Congestion Control Module (CCM), is inserted below the transport layer. This module cooperates with the link layer providing congestion control information for the transport layer, using a cross-layer collaboration technique. The implementation of cross-layer signaling we leave out of the scope,

however a detailed description of existing approaches is provided by authors of [85].

General architecture of C^{3}TCP and position of CCM within the protocol stack are specified in Figure 4.7. CCM has different functionalities depending whether it is implemented at the sender or at the receiver node.

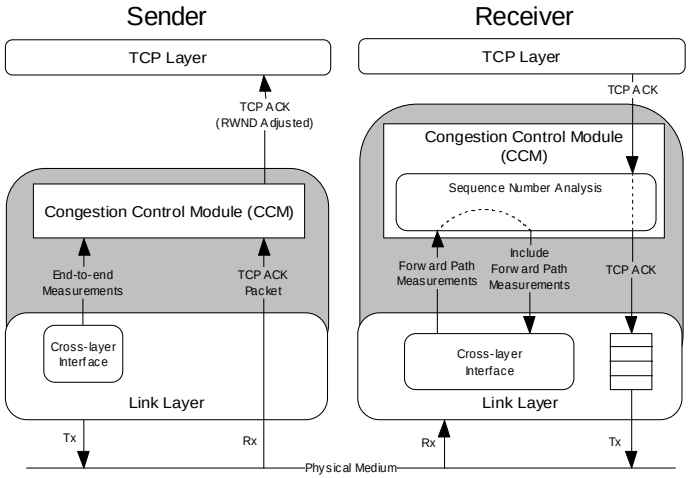


Figure 4.7: Congestion Control Module (CCM) architecture and its position within the protocol stack of C^{3}TCP-enabled nodes

At the receiver's side, upon the reception of a packet, CCM requests from the link layer the bandwidth and the delay which have been delivered with the TCP data packet. Having access to TCP headers, CCM traces the outgoing TCP ACKs. In case the produced TCP ACK acknowledges the received TCP data packet, CCM forwards the request to the link layer in order to include the stored bandwidth-delay information into MAC-layer header of the outgoing TCP ACK.

Modern implementations of TCP support cumulative or selective acknowledgements, which lead to the generation of one TCP ACK packet per several TCP data packets received. In this case,

CCM will include forward path measurement obtained from the last data packet acknowledged by the outgoing TCP ACK.

At the sender's side, CCM requests end-to-end measurements from the link layer upon TCP ACK packet reception. Then, it calculates the desired size of the congestion window based on the on the forward path bandwidth and RTT values on the forward path.

TCP specification includes receiver advertised window function, which main idea is to allow the receiver to specify (in the TCP ACK header) the desired congestion window size. In current implementations of TCP, this parameter includes unoccupied buffer space left on the receiver.

CCM uses the receiver advertised window (*rwnd*) field of TCP ACK packet for the delivery of the calculated congestion window. In detail, it leaves the lower *cwnd* value between the calculated one and the one reported by the receiver. Producing congestion control through the correction of receiver's advertised window is not a novel approach, on the contrary it is a common technique for the congestion window adjustment – presented in many works. For example, the authors of [72] use *rwnd* field of TCP header to inform the sender about network congestion from intermediate routers based on the free buffer space left on the edge device.

Rwnd signaling adjusts TCP window limiting its upper bound of evolution. In order gain full control on the size of the window, CCM should be enabled with acknowledgement generation for the local TCP layer in order to control the behavior of TCP congestion control algorithm.

4.4.5 Multi-node Multi-flow Scenario

In previous paragraphs, congestion control was mostly focused on the simple single-flow example. However, wireless multi-hop networks are aimed at supporting more complex scenarios, where different nodes initiate transmission of multiple data flows.

Figure 4.8 presents a three-flow example of multi-hop communications. Flow *F1* shares a part of the data path with flow *F2*. Flow *F3* will also influence flows *F1* and *F2*, since the destina-

tion node *N6* shares the same medium with nodes *N2*, *N3* and *N4*. It is assumed that all nodes use RTS/CTS framework in order to solve the hidden node problem.

Obviously, the bottleneck shared by all three connections is located at node *N3*. For that reason, the measured bandwidth between nodes *N7* and *N6* will contain the portion of bandwidth used by flow *F3*.

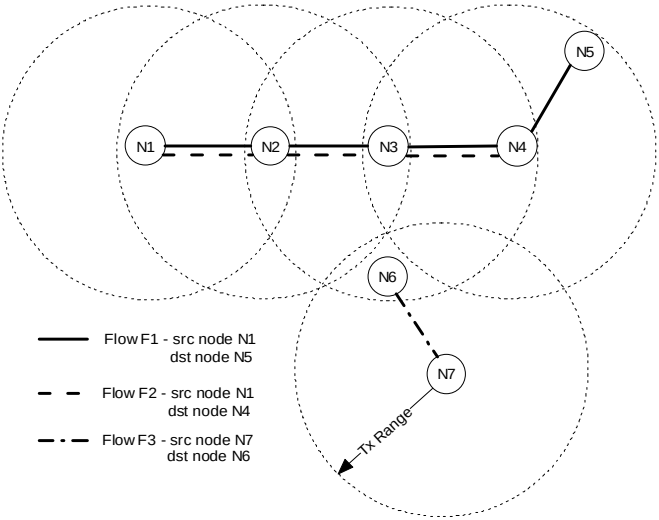


Figure 4.8: Multi-flow communications between different nodes of a multi-hop network

Flows *F1* and *F2* are the flows produced by the same sender node *N1*. This makes the measured bandwidth equal to the portion of bandwidth jointly occupied by both of them. For that reason, it is not possible to adjust contention windows without performing per-flow differentiation. However, per-flow differentiation can be supported by the C^3TCP framework.

The general structure of a possible flow-differentiation module is presented in Figure 4.9. The purpose of the Packet Classifier is to differentiate incoming packets according to the data flow they belong to. Then, the transmission scheduler allocates for every flow an appropriate portion of the available bandwidth to the

node. The implementation of scheduling algorithm could be simple or contain fairness and Quality of Service (QoS) support.

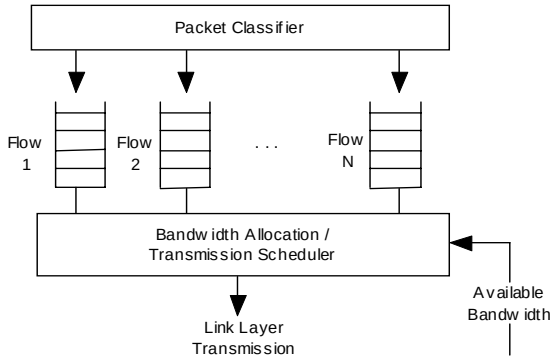


Figure 4.9: Flow differentiation for congestion control in the C^{3}TCP framework

4.4.6 Routing

A multi-hop IEEE 802.11-based wireless network is a self-organizing network, where nodes should perform route discovery in order to know the path or at least the next hop of the data communication path.

IEEE 802.11 standard does not specify any routing protocol, relying on a simple scenario where all the nodes are located within transmission range of each other. In order to adapt wireless networks to multi-hop scenarios, a growing amount of proposals gave birth to many routing protocols optimized for the wireless environment.

The existing routing protocols are categorized into two major groups: on-demand and proactive. On-demand protocols perform route discovery in the moment of the need for communication, for example AODV [58, 86] and DSR [87]. Proactive approaches like DSDV [88], on the contrary, try to avoid the delay overhead added by initial route discovery keeping the table with routes' description, updated at regular intervals.

However, the main aspect of routing is the metric which is chosen for route selection. Traditional routing protocols rely on shortest path routing, which brings performance optimization in wired networks through improvement in data delivery delay as well as bandwidth utilization. Wireless networks have an additional set of parameters which should be taken into account to make a proper choice. Such parameters include energy constrains, error rate, reliability of links, mobility and available throughput level

The importance of the last metric is underlined in [89], where the authors introduce an alternative metric for route selection which is called Medium Time Metric (MTM). MTM assigns a weight to each route that is proportional to the time taken for packet delivery over that particular route.

As an extension of MTM metric, we propose to differentiate different routes according to their bandwidth-delay product. Dynamical update of the weight of the routes can be produced based on the values measured with existing data flows of the nodes. Such a way of updating does not produce an additional overhead, in opposite to the case of routing protocol update leading to an improved efficiency in the utilization of network capacity.

4.5. Performance Evaluation

The performance of the proposed solution is analyzed using the ns-2 network simulator [55]. C^3TCP evaluation is performed using two scenarios. First scenario evaluates step-by-step C^3TCP operation showing good agreement with the design objectives, while the more complex second scenario better approximates the reality of ad hoc multi-hop network communications.

4.5.1 Scenario 1: String Topology

The first simulation scenario consists of four nodes involved in a single TCP connection and two nodes which produce cross-traffic UDP packets (Figure 4.10). Node *N1* is attached to a TCP agent, while TCP sink is located at the node *N4*. TCP packets are routed through the intermediate nodes *N2* and *N3* up to the destination

node *N4*. Due to transmission range limitations, cross-traffic stream shares the same medium with TCP flow only at the link between nodes *N3 – N4*. Each station's transmitting range is limited to 22.5 meters, and the distance between each pair of nodes in the simulation scenario is equal to 20 meters.

Congestion control module (CCM) is attached at the link layer of end nodes of the TCP connection. At the sender size (node *N1*), CCM dynamically adjusts TCP congestion window specifying its desired size by the RWND field of TCP header.

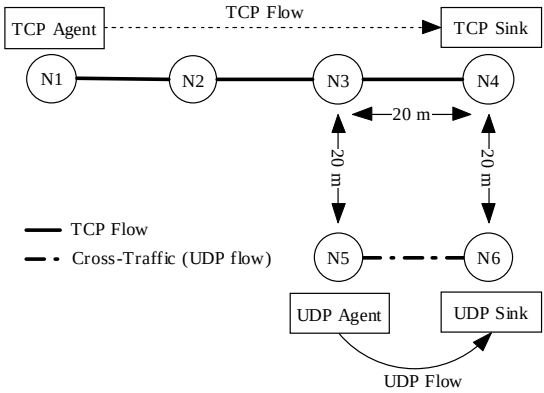


Figure 4.10: C^{3}TCP evaluation: Scenario 1

Simulation parameters are set to satisfy the IEEE 802.11b specification of the standard [9] at both physical and link layers.

TCP flow is started after 3.0 seconds of the simulation, being initially the only traffic in the network. Cross-traffic produced by node *N5* is present only in the interval between 15.0 and 30.0 seconds of simulation.

In order to evaluate the accuracy of bandwidth measurements provided by the presented technique, the difference between calculated bandwidth and the one obtained at the link level is presented in Figure 4.11. The dashed curve corresponds to the available bandwidth on the link, which is calculated without taking into account exponential backoff performed by the nodes as well as in absence of collisions. The results show good approximation

achieved by measurements in both cases: with and without cross-traffic.

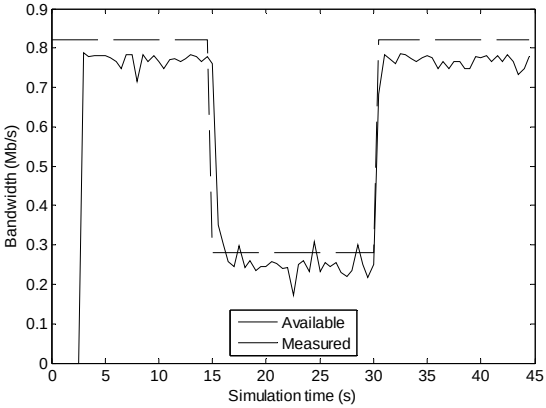


Figure 4.11: Accuracy of C^{3}TCP bandwidth measurement

Another important parameter for TCP performance is the RTT. Figure 4.12 presents a comparison between RTT measured at the link level against transport layer measurements. RTT measurements at the transport layer are performed using timestamp options specified in [73].

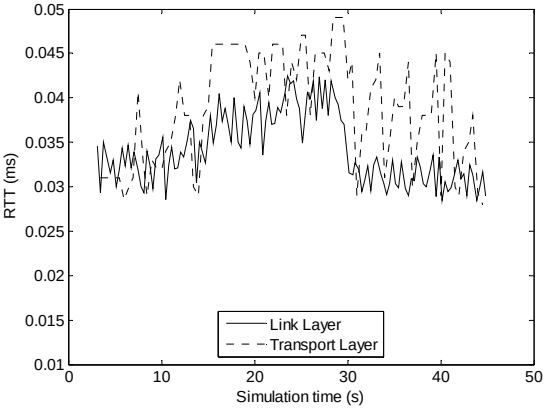


Figure 4.12: C^{3}TCP RTT measurements against standard TCP measurements

Transport layer is not aware about the medium it operates on. For that reason, it is not possible for standard TCP to differentiate between queuing and transmission delay, as both of them are included into the obtained RTT value.

In case TCP continuously outputs two data packets, the last will stay in buffer waiting for the transmission of the first one. This results in an artificial increase of the RTT delay by a time interval equal to the transmission delay of the first packet over the wireless link. Such situation leads to overestimate the length of available data pipe, which results in an unnecessary increase of the TCP congestion window.

In opposite to transport measurements, link layer measurements avoid including queuing delay into the measured round trip delay value. However, delay variation is still present due to the difference in medium access time as a consequence of collisions and random backoff. The jitter of the link layer curve which corresponds to single TCP flow measurements mostly reflect variations due to exponential backoff. In the presence of cross-traffic (from 15.0 to 30.0 sec), the average of RTT measured values is increased mostly due to collisions.

Results presented in Figure 4.12 underline the benefits obtained from the C^3TCP link layer measurements if compared with those performed at the transport layer, providing good confirmation of the theoretical advantages of link layer measurements.

The major metric for evaluating the performance of TCP flows is the obtained end-to-end throughput. The throughput comparison between TCP version with cross-layer congestion control (C^3TCP) and standard TCP implementation [43] is presented in Figure 4.13.

Results underline stability of throughput in the case of C^3TCP during all the phases of the experiment. The classical implementation of congestion control, on the opposite, always tries to enlarge the window, periodically incurring into congestion.

In more details, in case TCP flow is the only traffic present in network, C^3TCP throughput is comparable with one obtained by standard TCP implementation, with less jitter. However, when

cross-traffic is present (from 15.0 to 30.0 sec), standard TCP flow periodically drops throughput to 0, while C^{3}TCP always keeps the throughput level close to the available bandwidth – showing good utilization of the link capacity.

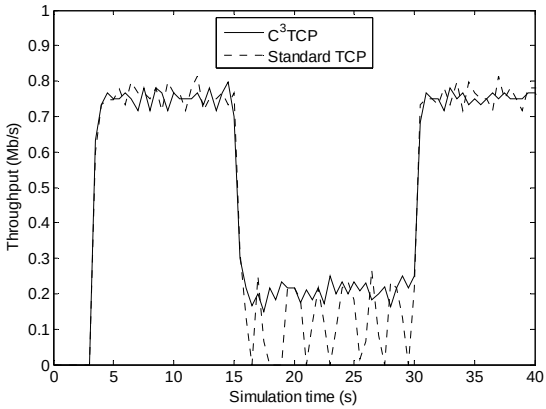


Figure 4.13: Throughput of C^{3}TCP against standard TCP implementation

The bottleneck on the TCP communication path in the evaluation scenario presented in Figure 4.10 is the link between nodes *N2* and *N3*. Node *N5* produces cross-traffic, generating RTS for obtaining medium access which is heard by node *N3* but not by node *N2*. As a result, node *N2* does not receive any response while trying to communicate with the node *N3*. The communication between nodes *N1* and *N2* is still possible, since none of them is aware of the cross-traffic and cross-traffic flow does not collide with transmissions of such nodes. The only assumption which is made is that signals transmitted by the nodes do not collide outside effective transmission range (22.5 meters). This comes from the implementation details of IEEE 802.11 MAC inside ns-2 simulator.

The TCP source attached to the node *N1* delivers data packets to node *N2* utilizing full bandwidth of the link between nodes *N1* and *N2*. Then, node *N2* can forward the received packets to node *N3* only after node *N5* finalizes its pending transmission. The dif-

ference between incoming and outgoing data rates observed by node *N2* results in multiple buffer overflows, which finally cause TCP throughput reduction.

Packets dropped from the buffer of intermediate routers are partially transported to the destination. It is demonstrated that multiple drops of such packets could lead to networks collapse [50]. In order to evaluate buffer usage as well as for better understanding the advantages of C^{3}TCP, buffer utilization at the bottleneck node *N2* is measured. The evaluated buffer is limited in size – allowing an allocation of up to 20 packets, which is about 5 times greater than the entire link capacity (using 1K TCP data packets). The obtained results are presented in Figure 4.14 for standard TCP and in Figure 4.15 for C^{3}TCP.

Buffer usage of standard TCP flow is relatively low (less than 5 packets) when the incoming and outgoing data rates present on node *N2* are comparable. However, in presence of cross-traffic, standard TCP source produces much more packets, overloading the bottleneck link. Such situation leads to buffer overflow and consequently multiple packet drops in the interval between 15.0 and 30.0 sec.

On the contrary, knowledge of the capacity of the communication path greatly reduces buffer usage for C^{3}TCP if compared with standard TCP implementation. Thus, buffer in node *N2* does not exceed the value of 10 packets in presence of cross-traffic and it is fixed in the interval between 0 and 3 packets when C^{3}TCP manages the only flow in the network. As a consequence, C^{3}TCP scheme does not produce more packets than the network can transport, saving communication resources and avoiding multiple packet drops along the communication path.

Standard TCP was chosen for the comparison as the most wide-spread TCP implementation, in order to underline the obtained performance improvement. However, an additional comparison with other approaches which aim at optimizing TCP congestion control framework is provided in the following.

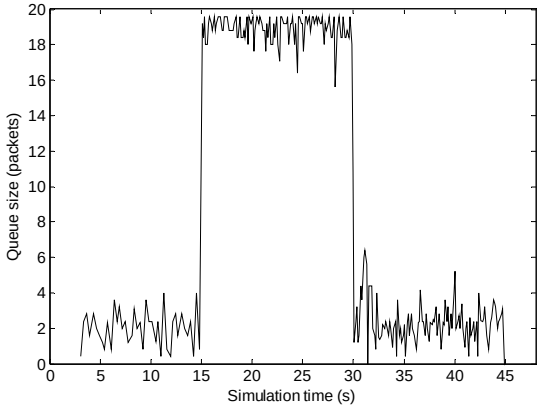


Figure 4.14: Buffer utilization at node N2 for standard TCP

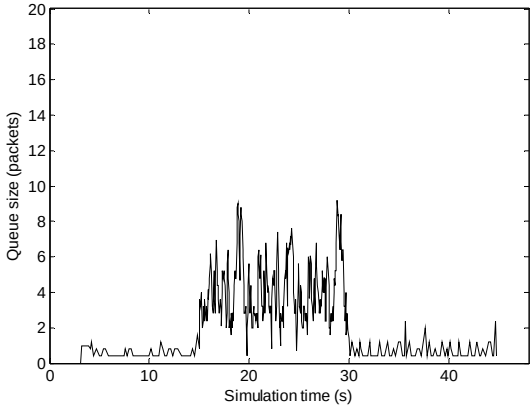


Figure 4.15: Buffer utilization at node N2 for C^3TCP

TCP Vegas [67] and TCP Westwood [68] are chosen among those congestion control solutions which most closely approximate C^3TCP from the theoretical point of view. The model of TCP Vegas is taken from standard ns-2 distribution, while TCP Westwood model is obtained from [90].

Throughput comparison results are presented in Figures 4.16 and 4.17 for TCP Vegas and TCP Westwood respectively.

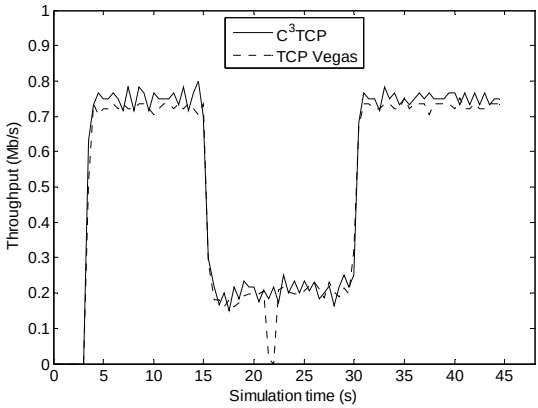


Figure 4.16: Throughput of C^{3}TCP against TCP-Vegas implementation

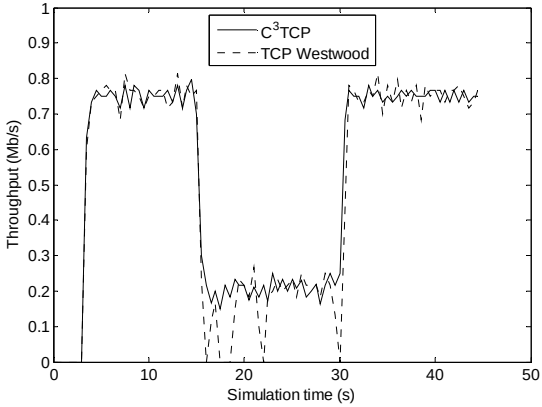


Figure 4.17: Throughput of C^{3}TCP against TCP-Westwood implementation

The results show that all the evaluated approaches achieve relatively close (with difference less than 2%) throughput in the scenario when TCP flow is not affected by cross traffic (from 3.0 to 15.0 and from 30.0 to 45.0 seconds of simulation).

However, when cross-traffic is present (between 15.0 and 30.0 seconds of simulation), both TCP Vegas and TCP Westwood pe-

riodically drop their throughput down to zero due to overestimation of the link capacity. For clarity of presentation, zoomed portions of the graphs are presented in Figures 4.18 and 4.19.

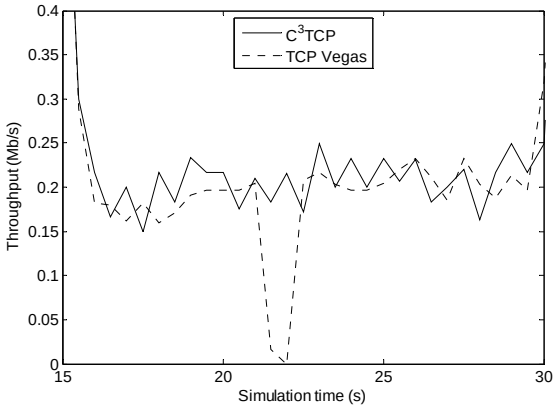


Figure 4.18: Throughput of C^{3}TCP against TCP-Vegas implementation (zoomed)

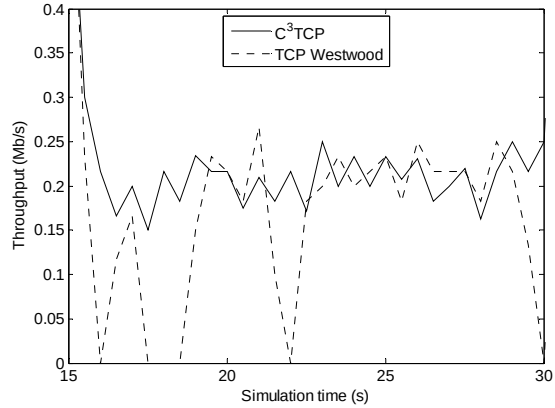


Figure 4.19: Throughput of C^{3}TCP against TCP-Westwood implementation (zoomed)

In the considered scenario, TCP Vegas performs better than TCP Westwood. The main reason for that is that TCP Vegas performs comparison between the estimated capacity and the actually

achieved throughput. Based of such comparison TCP Vegas does not increase congestion windows if this does not lead to a throughput increase. As a result, the more conservative TCP Vegas performs better than TCP Westwood.

The throughput of C^{3}TCP is stable for the entire simulation interval, showing good approximation of the available link capacity. The performed comparison underlines the advantages of the network capacity measurement at the link level rather than at the transport level.

For the purpose of quantitative comparison, results show an improvement achieved by C^{3}TCP of around 27%, 18% and 7% against standard TCP, TCP Westwood, and TCP Vegas, respectively.

4.5.2 Scenario 2: Grid Topology

The results presented in Scenario 1 show good agreement with the design of C^{3}TCP. However the assumed simplicity can not guarantee similar behavior in general case of operation in ad hoc multi-hop networks.

In order to approach the desired case Scenario 2 specifies flat-grid topology consisting of 20 nodes as it is shown in Figure 4.20. The size of the cell in the grid of 20 meters allows communication only between neighboring nodes connected by dashed line.

TCP agent attached to the node 10 and TCP Sink attached to the node 14 create “long” four-hop TCP flow, while the “short” two-hop flow is initiated between nodes 5 and 17. As a result, the first two hops of physical medium utilized by the “long” flow are shared with the “short” TCP flow.

UDP agent and sink attached to the nodes 8 and 4 respectively create constant bit rate cross-traffic flow which influences “long” TCP flow but not the “short” one.

Simulations were run for 100 seconds. Uninterrupted TCP traffic flows were started at the beginning of simulation, while the cross-traffic UDP flow was present only in interval between 30 and 70 seconds of simulation time.

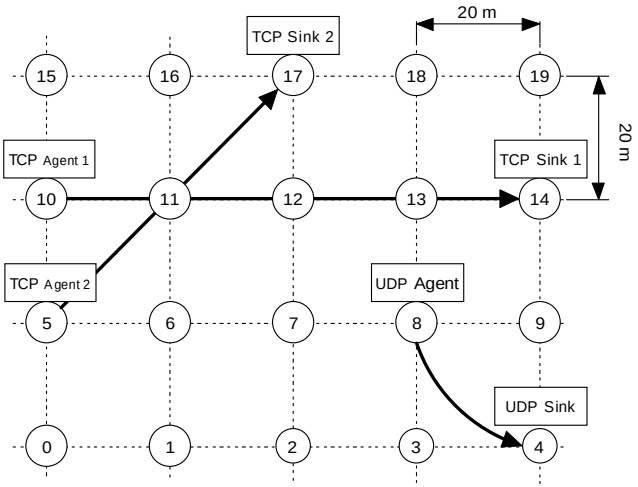


Figure 4.20: C^{3}TCP evaluation: Scenario 2

Obtained simulation results are presented in Figure 4.21 for standard TCP, TCP Westwood, TCP Vegas and proposed C^{3}TCP implementations. Within entire simulation time standard TCP implementation (see Figure 4.21a) continuously tries to increase congestion window on relatively low capacity link. As a result, due to multiple congestion-related packet losses and TCP timeout expirations the throughput of both flows shows high fluctuations. The "short" TCP flow gets slightly higher average of 0.4 Mbps if compared with 0.33 Mbps obtained by the "long" flow.

Similar to standard TCP behavior but with lower level of fluctuations is observed with TCP Westwood (see Figure 4.21b). Both flows are continuously trying to get full available bandwidth one over another. Periodic throughput reduction present on the graph comes from the fact of bandwidth overestimation which is caused mainly by the employed type of TCP Westwood ACK filter. However the large-scale sharing of bandwidth by TCP Westwood flows is relatively fair with an average throughput of 0.4 Mbps per flow.

Much more stable behavior is observed with TCP Vegas flows (see Figure 4.21c). The presented results show relatively large un-

fairness in sharing the available bandwidth. The “short” flow always show better throughput if compared with the “long” one. Additionally with the present of constant-bitrate UDP traffic the throughput of the “long” flow is dramatically decreased down to zero.

Figure 4.21d shows the results obtained with the proposed C^{3}TCP scheme. While not being disturbed by the cross-traffic both flows share the available bandwidth equally keeping their throughput average on 0.45 Mbps. In the interval between 30.0 and 70.0 seconds of simulation time the cross-traffic UDP flow is starting to take a part of the bandwidth from the “long” flow. As a result, the throughput of the “short” flow is increased with the portion of the bandwidth temporarily released by the “long” flow.

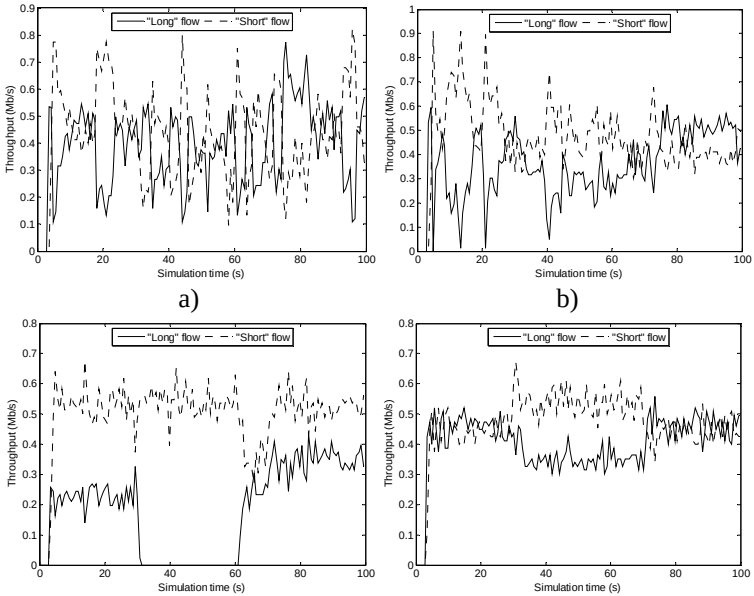


Figure 4.21: Simulation throughput results for data flows of a) Standard TCP b) TCP-Westwood c) TCP Vegas, and d) C^{3}TCP implementations

4.6. Conclusion

This chapter presents the problem of performance degradation of transport layer protocols due to congestion in wireless multi-hop local area networks. Following the analysis of available solutions to this problem, a cross-layer congestion avoidance scheme (C^3TCP) is presented, able to obtain higher performance by gathering capacity information such as bandwidth and delay at the link layer. The method requires the introduction of an additional module within the protocol stack of the mobile node, able to adjust the outgoing data stream based on capacity measurements. As an additional contribution it is proposed to provide optional field support to existing IEEE 802.11 protocol, in order to support the presented congestion control solution as well as many other similar approaches.

Achieved results underline good agreement with design considerations and high utilization of the available resources. Ongoing work is oriented to a comprehensive evaluation of the presented congestion control technique and a possible proposal to IEEE 802.11 working group to include the support of optional fields into next releases of the standard.

If we knew what it was we were doing, it would not be called research, would it?

Albert Einstein

Chapter 5

5. Conclusions and Future Work

In this research, we investigate the application of "Cross-Layer Design" for TCP/IP performance improvement in wireless networks.

The TCP/IP protocol suite, which is the de facto standard for communications in Internet today, is originally designed for traditional wired networks. As a result, TCP/IP shows poor performance in wireless network environment due to the limitations of wireless medium terms of bandwidth, latency, information loss, and mobility.

Traditionally, the proposals for TCP/IP performance improvement optimize a single layer at a time. However, in this thesis we show that Cross-Layer design allows better performance optimization and more flexibility in the design. It overcomes layering principles employed in network architectures and protocol stacks allowing joint interlayer optimization.

This thesis identifies two fundamental challenges which degrade the TCP/IP performance in wireless networks:

- *Wireless channel losses*: The loss probability experienced by packet transmission is several orders higher on the wireless me-

dium rather than on wired links. Such error rates are unacceptable for the TCP/IP designed for wired networks.

In order to counteract such variation of bit errors, Automatic Repeat reQuest (ARQ) is employed at the link layer of the protocol stack. It requires the receiver positively acknowledge the successfully received data frame. On the other hand, TCP employs another ARQ at the transport layer to ensure reliability. Due to the overlap of ARQ functionalities at different layers several acknowledgement acknowledgements are generated for a single data block transmitted over wireless channel. These acknowledgements correspond to the overhead reducing available bandwidth resources.

The optimization of the acknowledgement scheme brings performance improvement through the reduction of the medium-busy time and would require interaction between the transport and link layers – thus requiring proper cross-layering schemes.

- *Congestion control*: The congestion control implemented in TCP is probably the most important algorithm defining overall network performance. It controls the outgoing traffic rate with the purpose of keeping it at a maximum network utilization level however avoiding network overload and further congestion collapse. Ideally, the outgoing rate should be equal to bandwidth available on the path between the sender and the receiver.

Following the analysis of challenges posted above this dissertation provides corresponding solutions using cross-layer design approach:

- **Cross-Layer ARQ:** this scheme enhances the protocol stacks of the wireless sender and the receiver with cross-layer ARQ agents which enable collaboration between the link and the transport layers. The ARQ agent generates TCP acknowledgement at the sender side locally as soon as the link layer confirm successful packet delivery. As a result, Cross-Layer ARQ approach avoids the transmission of TCP ACK packet over the wireless channel. The saved time can be utilized by the nodes for data packet delivery which increases overall network capacity.

- **Cross-Layer Congestion Control**: This scheme enhances TCP congestion control over wireless networks by providing the transport layer with end-to-end link capacity measurements gathered at the link and physical layers. The method requires an additional module with the protocol stack of the mobile node which is able to adjust outgoing data stream based on the obtained capacity measurements.

We believe the solutions presented in this research thesis can become the first step in the tremendously growing field of cross-layer design.

On the other hand, cross-layer design is relatively new to the research community, and currently it evolves "spontaneously". However, there is a tradeoff between benefits of performance enhancements and future design complexity.

As a step forward we are currently working on the definitions of the approaches able to analyze and provide guidelines for the design of cross-layer solutions, and, even more important, to decide whether cross-layering represents an effective solution or not.

In [96], we initiated a quantitative approach for calculating the sensitivity of system performance with the respect to parameters across different layers for VoIP over Wireless LAN. The developed metamodel provides system optimization from the service provider perspective maximizing the cost benefit outcome function.

Bibliography

[1] "Mercenaries of the Wireless Terminal War", *Market Research*, December 2001.

[2] K. Fall and S. Floyd, "Simulation-based Comparisons of Tahoe, Reno, and SACK TCP," *Computer Communication Review*, vol. 26, no. 3, pp. 5 – 21, 1996.

[3] D. Vardalis and V. Tsaoussidis, "On the Efficiency and Fairness of Protocol Recovery Strategies in Networks with Wireless Components," in *Proc. of International Conference on Internet Computing*, CSREA Press, Las Vegas, June 2001.

[4] S. Goel and D. Sanghi, "Improving TCP Performance over Wireless Links," in *proceedings of IEEE TENCON*, December 1998.

[5] F. Granelli, D. Kliazovich and Nelson L. S. da Fonseca, "Performance Limitations of IEEE 802.11 Networks and Potential Enhancements," Yang Xiao and Yi Pan (ed), "*Wireless LANs and Bluetooth", Nova Science Publishers*, Hardbound, 2005.

[6] S. Shakkottai, T. S. Rappaport, and P. C. Karlsson, "Cross-layer design for wireless networks," *IEEE Communications Magazine*, vol. 41, no. 10, pp. 74 – 80, October 2003.

[7] V. Srivastava and M. Motani, "Cross-layer design: a survey and the road ahead," IEEE Communications Magazine, vol. 43, no. 12, pp. 112 – 119, December 2005.

[8] W. C. Y. Lee, Mobile Communication Design Fundamentals, 2nd Edition, John Wiley and Sons, 1993.

[9] IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, ISO/IEC 8802-11::1999(E), August 1999.

[10] 802.11 Working Group for Wireless Local Area Networks, *http://standards.ieee.org/wireless/overview.html#802.11*.

[11] IEEE 802.11 Working Group Web Site, *http://grouper.ieee.org/groups/802/11/*.

[12] M.A. Visser and M. El Zarki, “Voice and data transmission over an 802.11 wireless network Personal, Indoor and Mobile Radio Communications,” in *Proc. of PIMRC*, September 1995.
[13] D. Robb, “Although Newer, 802.11g Not Necessarily Better, Tutorial,” *Wi-Fi Plannet*, http://www.wi-fiplanet.com/.
[14] L. Cheng and I. Marsic, “Lightweight Models for Prediction of Wireless Link Dynamics in Wireless/Mobile Local Area Networks,” in *Proc. of the IEEE 2002 Sarnoff Symposium on Advances in Wired and Wireless Communications*, NJ, USA, March 2002.
[15] Y. Xiao and J. Rosdahl, “Throughput and Delay Limits of IEEE 802.11,” *IEEE Communications Letters*, vol. 6, no. 8, pp. 355-357, August 2002.
[16] K. Pentikousis, “TCP in Wired-Cum-Wireless,” *IEEE Communications Surveys*, vol. 3, no. 4, pp. 2 – 14, 2000.
[17] The Norwegian academic and research data network, *http://www.uninett.no/wlan/throughput.html*
[18] H. Belakrishnan, S. Seshan, E. Amir, and R. Katz, “*Improving TCP/IP Performance over Wireless Networks*,” in *Proc. of the 1st ACM International Conference on Mobile Computing and Networking (MOBICOM)*, Berkeley, CA, November 1995.
[19] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, “TCP Selective Acknowledgment Options,” *RFC 2018*, April 1996.
[20] H. Balakrishnan, V. Padmanabhan, S. Seshan, and R. Katz, “A Comparison of Mechanisms for Improving TCP Performance over Wireless Links,” *IEEE/ACM Transactions on Networking*, vol. 5, no. 6, pp. 756 – 769, December 1997.
[21] S. Kent and R. Atkinson, “Security Architecture for the Internet Protocol,” *RFC 2401*, 1998.
[22] K. Ratnam and I. Matta, “WTCP: An efficient Mechanism for Improving TCP Performance over Wireless Links,” in *Proc. of the Third IEEE Symposium on Computers and Communications*, Athens, Greece, June 1998.

[23] E. Ayangolu, S. Paul, T. LaPorta, K. Sabnani, and R. Gitlin, "AIRMAIL: A Link Layer Protocol for Wireless Networks," *Wireless Networks*, vol. 1, no. 1, pp. 47 – 60, 1995.

[24] C. Parsa and J.J. Garcia-Luna-Aceves, "Improving TCP Performance over Wireless Networks at the Link Layer," in *Proc. of Wireless Communications and Networking Conference (WCNC)*, September 1999.

[25] N. H. Vaidya, M. Mehta, C. Perkins, and G. Montenegro, "Delayed duplicate acknowledgements: A TCP-Unaware approach to improve performance of TCP over wireless," *Wireless Communications and Mobile Computing*, vol. 2, no. 1, pp. 59 – 70, 2002.

[26] D. Kliazovich and F. Granelli, "DAWL: A Delayed-ACK Scheme for MAC-Level Performance Enhancement of Wireless LANs," *ACM/Kluwer Journal on Mobile Networking and Applications (MONET)*, vol. 10, no.5, pp. 607 - 615, 2005.

[27] D. Kliazovich and F. Granelli, "A Delayed-ACK Scheme for MAC-Level Performance Enhancement of Wireless LANs," *11th International Conference on Telecommunications (ICT'2004)*, Fortaleza, Brasil, August 2004.

[28] A. Bakre and B. R. Badrinath, "I-TCP: Indirect TCP for mobile hosts," *in Proc. of the 15-th International Conference Distributed Computing Systems*, Vancouver, BC, Canada , June 1995.

[29] A. Bakre, "Design and Implementation of Indirect Protocols for Mobile Wireless Environments," *Rutgers University*, New Brunswick, NJ, USA, October 1996.

[30] K. Wang and S.K. Tripathi, "Mobile-end transport protocol: an alternative to TCP/IP over wireless links," *in Proc. Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies*, April 1998.

[31] C. Parsa and J. J. Garcia-Luna-Aceves, "Improving TCP Congestion Control over Internets with Heterogeneous Transmission Media," in *Proc. of IEEE ICNP*, Toronto, Canada, October 1999.

[32] L. S. Brakmo, S. W. O'Malley, and L. Peterson, "TCP Vegas: New Techniques for Congestion Detection and Avoidance," in *Proc. of ACM SIGCOMM*, October 1994.
[33] B.S. Bakshi, P. Krishna, N.H. Vaidya, and D.K. Pradhan, "Improving performance of TCP over wireless networks," in *Proc. of the 17th International Conference on Distributed Computing Systems*, Baltimore, Maryland, May 1997.
[34] H. Balakrishnan and R. Katz, "Explicit Loss Notification and wireless web performance," in *Proc. IEEE Globecom Internet Mini-Conference*, Sydney, Australia, November 1998.
[35] Z. H. Haas, "Design Methodologies for Adaptive and Multimedia Networks," *Guest Editorial, IEEE Communications Magazine*, vol. 39, no. 11, pp. 106-107, November 2001.
[36] Q. Wang and M.A. Abu-Rgheff, "Cross-layer signalling for next-generation wireless systems," in *Proc. of IEEE Wireless Communications and Networking Conference (WCNC)*, New Orleans, Louisiana, March 2003.
[37] M. Chinta and S. Helal, "ILC-TCP: An Interlayer Collaboration Protocol for TCP Performance Improvement in Mobile and Wireless Environments," in *Proc. of the third IEEE Wireless Communications and Networking Conference (WCNC)*, New Orleans, Louisiana, March 2003.
[38] V.T. Raisinghani, A.K. Singh, and S. Iyer, "Improving TCP performance over mobile wireless environments using cross layer feedback," in *Proc. of the IEEE International Conference on Personal Wireless Communications*, December 2002.
[39] S. Xu and T. Saadawi, "Does the IEEE 802.11 MAC Protocol Work Well in Multihop Wireless Ad Hoc Networks?" *IEEE Communication Magazine*, vol. 39, no. 6, pp.130-137, June 2001.
[40] M. Conti, G. Maselli, G. Turi, and S. Giordano, "Cross-layering in mobile ad hoc network design," *IEEE Computer*, vol. 37, no. 2, pp. 48 – 51, February 2004.
[41] M. van Der Schaar and Sai Shankar N, "Cross-layer wireless multimedia transmission: challenges, principles, and new

paradigms," *IEEE Wireless Communications*, vol. 12, no. 4, pp. 50 – 58, August 2005.
[42] T. S. Rappaport, A. Annamalai, R. M. Buehrer, and W. H. Tranter, "Wireless communications: past events and a future perspective," *IEEE Communications Magazine*, vol. 40, no. 5, pp. 148 – 161, May 2002.
[43] J. B. Postel, "Transmission Control Protocol," *RFC 793*, September 1981.
[44] G. J. Miller, K. Thompson, and R. Wilder, "Wide-area Internet traffic patterns and characteristics," *IEEE Network*, vol. 11, no. 6, pp. 10 – 23, November/December 1997.
[45] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, and S.C. Diot, "Packet-level traffic measurements from the Sprint IP backbone," *IEEE Network*, vol. 17, no. 6, pp. 6 – 16, November-December 2003.
[46] M. Zeng, A. Annamalai, and V.K. Bhargava, "Recent advances in cellular wireless communications," *IEEE Communications Magazine*, vol. 37, no. 9, pp. 128 – 138, September 1999.
[47] A. Ghosh, D. R. Wolter, J. G. Andrews, and R. Chen, "Broadband wireless access with WiMax/802.16: current performance benchmarks and future potential," *IEEE Communications Magazine*, vol. 43, no. 2, pp. 129 – 136, February 2005.
[48] W. Stevens, "TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms," *RFC 2001*, January 1997.
[49] F. Kelly, "Mathematical Modelling of the Internet," B. Engquist and W. Schmid (ed.), "Mathematics Unlimited - 2001 and Beyond", Springer-Verlag, Berlin, pp. 685-702, 2001.
[50] S. Floyd and K. Fall, "Promoting the use of end-to-end congestion control in the Internet," *IEEE/ACM Transactions on Networking*, vol. 7, no. 4, pp. 458 - 472, August 1999.
[51] D. Clark, "Window and Acknowledgement Strategy in TCP," *RFC 813*, July 1982.

[52] Z. Fu; H. Luo; P. Zerfos, L. Zhang and M. Gerla, "The impact of multihop wireless channel on TCP performance", *IEEE Transactions on Mobile Computing*, vol. 4, no. 2, pp. 209 – 221, March-April 2005.

[53] I.F. Akyildiz and W. Wang, "A survey on wireless mesh networks," *IEEE Communications Magazine*, vol. 43, no. 9, pp. 23 – 30, September 2005.

[54] M. Abolhasan, T. Wysocki, and E. Dutkiewicz, "A review of routing protocols for mobile ad hoc networks," *Ad Hoc Networks*, vol. 2, no. 1, pp. 1 – 22, January 2004.

[55] NS-2 simulator tool home page, *http://www.isi.edu/nsnam/ns/*, 2000.

[56] Iperf performance measurement tool, *http://dast.nlanr.net/Projects/Iperf*

[57] J. Nagle, "Congestion control in IP/TCP internetworks," *RFC 896*, 1984.

[58] C. Perkins, E. Belding-Royer, and S. Das. "Ad Hoc On Demand Distance Vector (AODV) Routing," *RFC 3561*, July 2003.

[59] E. Altman and T. Jimenez, "Novel delayed ack techniques for improving TCP performance in multihop wireless networks," in *Proc. of Personal Wireless Communications (PWC)*, Venice, Italy, September 2003.

[60] M. Shreedhar and G. Varghese, "Efficient fair queuing using deficit round robin," in *Proc. of ACM SIGCOMM*, 1995.

[61] D. Kliazovich and F. Granelli, "Cross-Layer Congestion Control in Ad Hoc Wireless Networks," *Ad Hoc Networks*, vol. 4, no. 6, pp. 678 – 708, 2005.

[62] V. Jacobson, "Congestion Avoidance and Control," *Computer Communication Review*, vol. 18, no. 4, pp. 314-329, August 1988.

[63] R. Wang, G. Pau, K. Yamada, M.Y. Sanadidi, M. Gerla, "TCP startup performance in large bandwidth delay networks," in *Proc. of the twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, March 2004.

[64] G. Holland, N. Vaidya, "Analysis of TCP performance over mobile ad hoc networks," *Wireless Networks*, vol. 8, pp. 275-288, March 2002.
[65] Z. Fu, X. Meng and S. Lu, "How bad TCP can perform in mobile ad hoc networks," in *Proc. of Seventh International Symposium on Computers and Communications (ISCC)*, July 2002.
[66] R. Garg, A. Kamra, and V. Khurana, "A Game-Theoretic approach towards congestion control in communication networks," *ACM SIGCOMM Computer Communication Review*, vol. 32, no. 3, pp. 47 – 61, July 2002.
[67] L. Brakmo and L. Peterson, "TCP Vegas: End to End Congestion Avoidance on a Global Internet," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 13, no. 8, pp. 1465 – 1480, October 1995.
[68] C. Casetti, M. Gerla, S. Mascolo, M. Sansadidi, and R. Wang, "TCP Westwood: End-to-End Congestion Control for Wired/Wireless Networks," *Wireless Networks*, vol. 8, no. 5, pp. 467-479, 2002.
[69] O. Akan and I. Akyildiz, "ATL: an adaptive transport layer suite for next-generation wireless Internet," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 22, no. 5, pp. 802 – 817, June 2004.
[70] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, pp. 397-413, August 1993.
[71] K. Ramakrishnan, S. Floyd, D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP," *RFC 3168*, September 2001.
[72] L. Kalampoukas, A. Varma, and K. Ramakrishnan, "Explicit Window Adaptation: A Method to Improve TCP Performance," in *Proc. of IEEE INFOCOM*, April 1998.
[73] V. Jacobson, R. Braden, D. Borman, "TCP Extensions for High Performance," *RFC 1323*, May 1992.
[74] H. Ningning and P. Steenkiste, "Evaluation and characterization of available bandwidth probing techniques," *IEEE Jour-*

nal on Selected Areas in Communications (JSAC), vol. 21, no. 6, pp. 879 – 894, Auguest 2003.

[75] K. Lai and M. Baker, "Nettimer: A tool for measuring bottleneck link bandwidth," in *Proc. of the 3rd USENIX Symposium Internet Technologies and Systems*, March 2001.

[76] C. Dovrolis, P. Ramanathan, and D. Moore, "What do packet dispersion techniques measure?" in *Proc. of IEEE INFOCOM*, April 2001.

[77] R. Kapoor, Ling-Jyh Chen, Li Lao, M. Gerla, and M. Y. Sanadidi, "CapProbe: A Simple and Accurate Capacity Estimation Technique," in *Proc. of ACM SIGCOMM*, Portland, USA, 2004.

[78] R. Kapoor, Ling-Jyh Chen, A. Nandan, M. Gerla, and M. Y. Sanadidi, "CapProbe: A Simple Technique to Measure Path Capacity," in *Proc. of ACM SIGMETRICS*, New York, USA, 2004.

[79] M. Gerla, R. Bagrodia, Z. Lixia, K. Tang, and W. Lan, "TCP over wireless multi-hop protocols: simulation and experiments," in *Proc. of IEEE International Conference on Communications (ICC)*, June 1999.

[80] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 18, no. 3, pp. 535 - 547, March 2000.

[81] D. Boggs, J. Mogul, and C Kent, "Measured Capacity of an Ethernet: Myths and Reality," *Computer Communication Reviews*, vol. 18, no. 4, pp. 222 – 234, August 1988.

[82] D. Kliazovich and F. Granelli, "Performance Improvements in Data Transfer over 802.11 WLANs," in *Proc. of the 10th IEEE Workshop on Computer-Aided Modeling, Analysis, and Design of Communication Links and Networks (CAMAD)*, Dallas, U.S.A., November 2004.

[83] J. Postel, "Internet Protocol: DARPA Internet Program Protocol Specification," *RFC 791*, September 1981.

[84] S. Deering, R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," *RFC 2460*, December 1998.

[85] Q. Wang and M.A. Abu-Rgheff, "Cross-layer signalling for next-generation wireless systems," in *Proc. of IEEE Wireless Communications and Networking Conference (WCNC)*, New Orleans, Louisiana, March 2003.

[86] C. Perkins and E. Royer, "Ad hoc On-Demand Distance Vector Routing," in *Proc. of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, New Orleans, LA, February 1999.

[87] D. Johnson, D. Maltz, and J. Broch, "DSR: The Dynamic Source Routing Protocol for Multi-Hop Wireless Ad Hoc Networks," Ad Hoc Networking, edited by C. Perkins, Chapter 5, pp. 139-172, Addison-Wesley, 2001.

[88] C. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers," in *Proc. of the Conference on Communications Architectures, Protocols and Applications (SIGCOMM)*, 1994.

[89] B. Awerbuch, D. Holmer, and H. Rubens, "The Medium Time Metric: High Throughput Route Selection in Multirate Ad Hoc Wireless Networks," *Kluwer Mobile Networks and Applications (MONET) Special Issue on "Internet Wireless Access: 802.11 and Beyond"*, 2005.

[90] TCP Westwood Home Page, *http://www.cs.ucla.edu/NRL/hpi/tcpw/*.

[91] S. Dawkins, G. Montenegro, M. Kojo, V. Magret, and N. Vaidya, "End-to-end Performance Implications of Links with Errors," *RFC 3155*, August 2001.

[92] V. T. Raisinghani and S. Iyer, "Cross-layer design optimizations in wireless protocol stacks," *Computer Communications*, v. 27, n. 8, pp. 720 – 724, 2004.

[93] N. Brownlee and K. Claffy, "Understanding Internet traffic streams: Dragonflies and tortoises," *IEEE Communication Magazine*, vol. 40, no. 10, pp. 110-117, October 2002.

[94] J. Broch, D. Maltz, D. Johnson, Y. Hu, and J. Jetcheva, "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols," in *Proc. of the Fourth Annual*

International Conference on Mobile Computing and Networking (MobiCom), Dallas, TX, October 1998.

[95] A. Al Hanbali, E. Altman, and P. Nain, "A survey of TCP over Mobile Ad Hoc Networks", *Technical report RR-5182*, May 2004.

[96] F. Granelli, D. Kliazovich, J. Hui, and M. Devetsikiotis, "Performance Optimization of Single-Cell Voice over WiFi Communications Using Quantitative Cross-Layering Analysis," submitted for publication to the *20th Iternational Teletraffic Congress (ITC'20)*, Ottawa, Canada, June 2007.

Biography

Dzmitry Kliazovich received his Masters degree in Telecommunication science from Belarusian State University of Informatics and Radioelectronics in 2002. He is currently working towards the Ph.D. degree in University of Trento, Italy. In 2005/2006 he was a visiting researcher at the Computer Science Department of the University of California at Los Angeles. He is an author of more than 20 research papers published in international books, journals and conference proceedings. His main research interest lies in field of wireless networking with a focus on performance optimization and cross-layer design.

Publications published during PhD study and work toward this research thesis:

Book Chapters

F. Granelli, D. Kliazovich and Nelson L. S. da Fonseca, *"Performance Limitations of IEEE 802.11 Networks and Potential Enhancements,"* Yang Xiao and Yi Pan (ed), ``Wireless LANs and Bluetooth", Nova Science Publishers, Hardbound, 2005.

Journals

D. Kliazovich, F. Granelli, and D. Miorandi, *"Logarithmic Window Increase for TCP Westwood+ Improvement in High Speed, Long Distance Networks,"* submitted for publication to the Computer Networks, 2006.

D. Kliazovich, F. Granelli, and M. Gerla, "*Performance Improvement in Wireless Networks using Cross-layer ARQ,"* submitted for publication to the Computer Networks, 2006.

D. Kliazovich and F. Granelli, "*Packet Concatenation at the IP Level for Performance Enhancement in Wireless Local Area Networks,"* to appear in Wireless Networks, 2007.

D. Kliazovich and F. Granelli, "*Cross-Layer Congestion Control in Ad Hoc Wireless Networks,"* Ad Hoc Networks, vol. 4, no. 6, pp. 678-708, 2006.

F. Granelli and D. Kliazovich, "*Cross-Layering for Performance Improvement in Multi-Hop Wireless Networks,"* Journal of Interconnection Networks (JOIN), vol. 7, no. 1, pp. 51-61, 2006.

D. Kliazovich and F. Granelli, "*DAWL: A Delayed-ACK Scheme for MAC-Level Performance Enhancement of Wireless LANs,"* Mobile Networks and Applications, vol. 10, no.5, pp. 607 - 615, 2005.

Conferences and Workshops

F. Granelli, D. Kliazovich, J. Hui, and M. Devetsikiotis, *"Performance Optimization of Single-Cell Voice over WiFi Communications Using Quantitative Cross-Layering Analysis,"* submitted for publication to 20th Iternational Teletraffic Congress (ITC'20), Ottawa, Canada, June 2007.

D. M. Batista, Nelson L. S. da Fonseca, F. Granelli, D. Kliazovich, *"Self-Adjusting Grid Networks,"* IEEE International Conference on Communications (ICC), Glasgow, Scotland, UK, June 2007.

F. Granelli, G. Boato, and D. Kliazovich, *"MORA: a Movement-Based Routing Algorithm for Vehicle Ad Hoc Networks,"* IEEE Workshop on Automotive Networking and Appications (AutoNet 2006), San Francisco, U.S.A., December 2006.

D. Kliazovich, F. Granelli, and H. Woesner, *"Bidirectional Optical Ring Network Having Enhanced Load Balancing and Protection,"* Optical Network Design and Modelling Conference (ONDM), Copenhagen, Denmark, May 2006.

D. Kliazovich, F. Granelli, G. Pau, and M. Gerla, *"APOHN: Subnetwork Layering to Improve TCP Performance over Heterogeneous Paths,"* IEEE Next Generation Internet Design and Engineering (NGI), Valencia, Spain, April 2006.

F. Granelli and D. Kliazovich, *"Cross-layering for Performance Improvement in Multi-Hop Wireless Networks,"* (invited paper) International Symposium on Parallel Architectures, Algorithms, and Networks (I-SPAN), Las Vegas, Nevada, USA, December 2005.

D. Kliazovich, F. Granelli, and D. Miorandi, *"TCP Westwood+ Enhancements for High-Speed Long-Distance Networks,"* IEEE International Conference on Communications (ICC'06), Istambul, Turkey, June 2006.

D. Kliazovich, F. Granelli, H. Woesner, and I. Chlamtac, *"Bidirectional Light-Trails for Synchronous Communications in WDM Networks,"* IEEE Global Communications Conference, GLOBECOM'05, St. Louis, U.S.A, December 2005.

D. Kliazovich and F. Granelli, *"Cross-Layer Congestion Control in Multi-hop Wireless Local Area Networks,"* IEEE International Conference on Wireless Internet (WICON'05), Budapest, Hungary, July 2005.

D. Kliazovich and F. Granelli, *"On Packet Concatenation with QoS support for Wireless Local Area Networks,"* IEEE International Conference on Communications (ICC'05), Seoul, Korea, May 2005.

D. Kliazovich and F. Granelli, *"A Cross-layer scheme for the TCP Performance Improvement in Wireless LANs,"* IEEE Global Communications Conference, GLOBECOM'04, Dallas, U.S.A, December 2004.

D. Kliazovich and F. Granelli, *"Performance Improvements in Data Transfer over 802.11 WLANs,"* 10-th IEEE Workshop on Computer-Aided Modeling, Analysis, and Design of Communication Links and Networks, CAMAD'04, Dallas, U.S.A., November 2004.

D. Kliazovich and F. Granelli, *"A Delayed-ACK Scheme for MAC-Level Performance Enhancement of Wireless LANs,"* 11-th International Conference on Telecommunications, ICT'04, Fortaleza, Brasil, August 2004.

Market Studies

D. Kliazovich and M. Gerla, *"NLOS, WiMAX, and Broadband Wireless: Competitive Market Analysis,"* Report, November 2005.