Available online at www.sciencedirect.com

ScienceDirect

Computer Networks 51 (2007) 4396–4411

Computer Networks

www.elsevier.com/locate/comnet

Performance improvement in wireless networks using cross-layer ARQ

Dzmitry Kliazovich [a,*,1], Fabrizio Granelli [a], Mario Gerla [b]

[a] *DIT – University of Trento, Via Sommarive 14, I-38050 Trento, Italy*
[b] *UCLA, Computer Science Department, CA 90095, United States*

Received 7 February 2006; received in revised form 2 June 2007; accepted 11 June 2007
Available online 16 June 2007

Responsible Editor: E. Altman

Abstract

This paper presents a novel cross-layer approach (LLE-TCP) designed for performance enhancement of TCP over a large variety of wireless networks. LLE-TCP avoids TCP ACK packet transmission over the wireless channel. As a result, the saved time can be utilized by the nodes for data packet delivery. The proposed scheme enhances the protocol stacks of the wireless sender (or a base station) and the receiver with cross-layer ARQ agents which support ACK suppression. ARQ agent suppresses the outgoing ACKs at the receiver side and generates them locally at the sender or base station.

The performance evaluation of the proposed approach is performed via simulations as well as IEEE 802.11 testbed experiments for single-hop and infrastructure network scenarios. LLE-TCP demonstrates the performance improvement in the range of 20–100% depending on the transmitted TCP/IP datagram size.

Among the factors contributing to performance enhancement are: medium busy time reduction, reduced sensibility to link errors, reduced round trip time (RTT), and improved congestion control.

A good level of throughput fairness as well as a fair coexistence with state-of-the-art TCP modifications ensures proper functionality of the proposed approach, while performance advantages extended even on non-LLE-TCP users favor an incremental deployment of the technique in existing networks.
© 2007 Elsevier B.V. All rights reserved.

Keywords: Cross-layer ARQ; Performance enhancement; Acknowledgement suppression

1. Introduction

Wireless communications clearly represent a fast-growing sector in the framework of data networks [1]. 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.

[*] Corresponding author. Tel.: +39 348 294 0764; fax: +39 0461 88 2093.
E-mail addresses: klezovic@dit.unitn.it (D. Kliazovich), granelli@dit.unitn.it (F. Granelli), gerla@cs.ucla.edu (M. Gerla).
[1] This work was done while Dzmitry Kliazovich was a visiting researcher at the Computer Science Department of the University of California at Los Angeles.

1389-1286/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2007.06.002

As a drawback, the capacity offered by wireless links is relatively low as compared to wired networks. Such capacity limitations derive from the very physical nature of the wireless medium, characterized by limited and shared bandwidth, time-varying behavior, interference, etc.

Nevertheless, 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/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 routing, and no mobility. On the contrary, in the wireless domain, performance is constrained by available transmission spectrum, employed modulation and 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 [2]. Such error rates are unacceptable for the transmission control protocol (TCP) [3], designed for wired networks, which delivers over 85% of Internet traffic [4,37]. 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 the delivery delay. 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 1 presents a summary of the characteristics of the wireless network standards, aimed at underlining the common features and similarities among them. The data presented for technologies that are not yet standardized (IEEE 802.11n) are based on available drafts. The reader should refer to the references for more details on cellular networks [5], IEEE 802.11 local area networks [6] and WiMax metropolitan area network [7] standards.

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.

Table 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 (2.5 G)			200 kHz (TDMA)	56–115 kbps	32–84 kbps		Yes
EDGE				384 kbps (48–60 kbps per timeslot)	300 kbps		Yes
3 G		1900–2025 MHz, 2110–2200 MHz	5 MHz	large range 144 kbps, medium range 384 kbps, small range 2 Mbps	120 kbps, 310 kbps, 1.6 Mbps		Yes
Wireless local area network (WLAN)							
IEEE 802.11	40–100 m	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

Summarizing, in most of the available wireless network architectures (see Table 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.

This paper targets 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.

The structure of the paper is the following: Section 2 provides an overview of the state-of-the-art, while Section 3 introduces the proposed scheme, outlining the different implementation scenarios. Finally, Section 4 presents experimental results and Section 5 concludes the paper.

2. Related work and motivation

The literature on data transfer optimization in wireless scenarios is huge, for that reason this section overviews only closely related works, while for a more precise survey the reader should refer to [8,9].

Many proposals for TCP performance optimization over wireless networks target the root of the problem, i.e. TCP flow control and error recovery mechanisms, introducing different TCP modifications. A well-known modification in the considered scenario is TCP Westwood [17] with its variations [18,19]. An estimate of the available capacity on the end-to-end path is computed by using appropriate filtering of returning ACK flow. Then, upon loss detection, the outgoing rate is adjusted to fit the available bandwidth-delay product instead of performing blind window reduction. TCP Veno [20] implements delay-based congestion control by estimating the number of backlogged packets in the bottleneck buffer – a technique originally proposed in TCP Vegas [21]. In addition, it tries to distinguish between congestion- and loss-related losses based on the estimated bottleneck buffer size preventing window reduction for non-congestion related losses.

On the contrary, flow control solutions adjust outgoing rate without providing reliability of data delivery, for real-time multimedia traffic applications. Proposals in this category can be implemented as an additional transport layer protocol or flow control above transport layer on top of unreliable UDP or RTP protocols. Rate adaptation protocol (RAP) [27] and datagram congestion control protocol (DCCP) [28,29] are examples of rate control protocols, while TCP-friendly rate control (TFRC) [35] is an equation-based rate control proposal designed to achieve a good fairness with TCP flows coexisting on the network. Analytical rate control (ARC) [36] improves the way losses are treated in TFRC by distinguishing between congestion related and random losses making ARC appropriate for wireless networks.

The main drawback preventing deployment of the proposed TCP modifications and other rate control protocols is in the requirement for sender protocol stack modification which is difficult to perform in world-wide scale. This motivated multiple attempts to hide undesirable characteristics of the wireless links from the transport layer while keeping TCP sender's stack unchanged.

One of the first of such attempts to mask losses on the wireless link from a fixed sender is proposed in [10]: the snoop agent introduced at the base station performs local retransmissions triggered by sniffing duplicate ACKs coming from the wireless receiver.

In [11], the authors of explicit loss notification (ELN) method propose to notify the TCP sender about the reason of the loss, in order to distinguish between errors on the wireless link and congestion-related packet drops. Based on such feedback, the sender node follows congestion control procedure only in case of packet lost due to congestion, and simply retransmits the packet without window reduction otherwise.

Connection-splitting solutions, such as Indirect TCP (I-TCP) [12], M-TCP [13], or Mobile-End Transport Protocol (METP) [38], break end-to-end connection into fixed and wireless sections. Each section uses a transport protocol optimized for such specific environment. Connection splitting solutions and more generic group of performance

Table 2
Comparison of the proposed LLE-TCP approach with existing state-of-the-art solutions

Group	Solution	Optimization layer	E2E	Modifies fixed sender	Ease to unplug	Ease to deploy
TCP modifications	TCP Westwood TCP Veno TCP Vegas	Transport	√	√	×	×
Non-TCP congestion control	RAP DCCP TFRC ARC	Transport/above transport	√	√	×	×
Connection splitting	I-TCP	Transport	×	√		
	M-TCP		√	×	√	√
	METP		×	×		
PEPs	Snoop	Transport/link	√	×	√	√
	LLE-TCP (proposed)		√	×	√	√

enhancement proxies (PEPs) [14] achieve good level of TCP performance improvement. However, they violate TCP semantics and end-to-end principle of Internet protocol design (i.e. an acknowledgement received by the sender does not indicate successful data delivery up to the destination node).

After a detailed analysis of the existing solutions (briefly overviewed above), we identified the cross-layer approach as an appropriate principle for the design of the proposed LLE-TCP protocol. Moreover, we target the design objectives derived both from the link layer as well as split-connection approaches, which can be summarized as follows:

from link layer solutions:

1. Preserve end-to-end TCP semantics.
2. No direct modification of the widely deployed TCP implementation.
3. Flexible on/off style on-demand performance optimization, i.e. optimization is performed only in case the node operates in the wireless network and optimization is desirable.

from connection splitting approaches:

4. Focus performance optimization on the wireless section of the connection.
5. Low implementation complexity.

In order to satisfy the design objectives presented above, the proposed solution performs joint optimization at the transport and the link layers. This idea can be included in a wide range of recently-proposed solutions for optimizing wireless network design that are labeled as "Cross-Layer Design" [15,16] methodologies, that extend the layering principles by allowing interdependence and joint design of protocols crossing different layers.

Furthermore, the proposed approach can be classified as performance enhancement proxy (PEP) solution, designed to be implemented at the base station for TCP optimization in an heterogeneous network environment.

Table 2 presents a top-level comparison among existing approaches and the proposed LLE-TCP solution in terms of layer for optimization, preserving end-to-end (E2E) TCP semantics, the requirement for protocol stack modification at the fixed sender, the simplicity to deactivate optimization technique if no wireless network is present, as well as the possibility for wide deployment.

The reader should note that LLE-TCP is not constrained to any specific TCP implementation and thus can lead to additional performance gains if applied jointly with TCP or other rate control optimization techniques.

3. The proposed scheme (LLE-TCP)

In this paper, we propose *link layer ARQ exploitation TCP* (LLE-TCP) approach to exploit the information of the link layer ARQ scheme for a more efficient acknowledgement of TCP packet delivery.[2]

[2] The basic concept of LLE-TCP approach was initially presented at the Globecom'04 conference [34].

The basic idea behind LLE-TCP approach is to shift TCP ACK generation point from mobile receiver to the base station. This is achieved by introducing an ARQ agent within the protocol stack of the base station which generates TCP ACK packets based on incoming traffic analysis. As a result, wireless link resources in terms of bandwidth and transmission delay associated with TCP ACK transmission over wireless channel can be released.

Reference scenarios for LLE-TCP deployment are represented by single hop wireless network (ad hoc network specified by IEEE 802.11, see Fig. 1a) and infrastructure network (IEEE 802.11, IEEE 802.16 or cellular network, see Fig. 1b).

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 fixed network infrastructure and mobile devices. Multiple studies show that TCP performance is poor in such environments [10], the main reason of which lies in completely different characteristics between fixed (wired) and wireless parts along the end-to-end data connection [8].

Infrastructure network (Fig. 1b) is chosen as a reference scenario for LLE-TCP presentation, while implementation details in ad hoc single-hop network scenario (Fig. 1a) are presented later.

Furthermore, when implementation details are described, reference platform is the Linux OS open protocol model, with the assumption that other operating systems such as MS Windows have relevant conceptual similarities.

3.1. Cross-layer ARQ agent

LLE-TCP requires implementation of a software module, called ARQ agent, inside base station (BS) protocol stack above the link layer. The ARQ agent sniffs the ingress traffic from the fixed network assuming to have 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.

Fig. 2 shows LLE-TCP packet delivery diagram while ARQ agent flowchart is presented in Fig. 3.

Assuming to have a successful TCP data segment delivery, the ARQ agent starts preparation of a TCP ACK which acknowledges the packet just transmitted to the transport layer. Such acknowledgement is generated as a standalone TCP packet containing the ACK bit set to ‘1’ in its header and no data payload. This generation does not require an implementation of TCP layer in a conventional sense, as it consists of simple copy of appropriate TCP header fields (such as connection port numbers, flow sequence number, ACK flag, and ACK sequence number) into a TCP ACK packet template previously allocated in memory. Then, generated TCP ACK is not sent to the TCP sender immediately, but stored in the buffer.

As soon as TCP data packet is successfully delivered (indicated by the received link layer acknowledgement), the link layer notifies ARQ agent by sending “*Tx. Success*” event. Otherwise, “*Tx. Failure*” event is sent, indicating that, after all possible retransmission attempts, the link layer is not able to deliver the packet.

On “*Tx. Success*” event, the ARQ agent releases the prepared TCP ACK packet to the fixed host, while in case of failure it simply drops the generated TCP ACK releasing the associated memory resources.

On the receiver side, module referred as ARQ client in Fig. 2 silently drops all standalone non-duplicate TCP ACK packets which are artificially generated at the sender side.

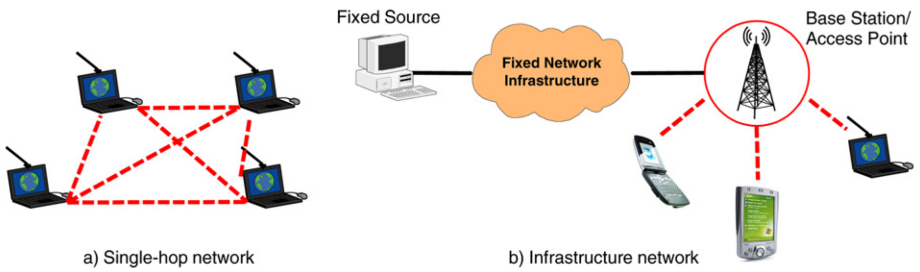
Fixed Source
Fixed Network Infrastructure
Base Station/ Access Point
a) Single-hop network
b) Infrastructure network

Fig. 1. Reference network scenarios considered for LLE-TCP implementation.

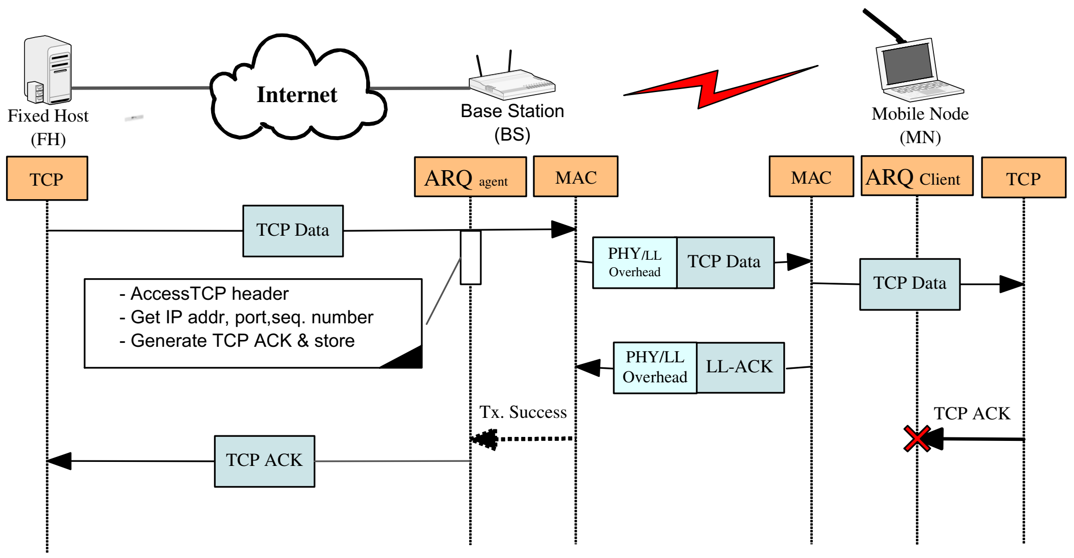

Fig. 2. LLE-TCP packet delivery.

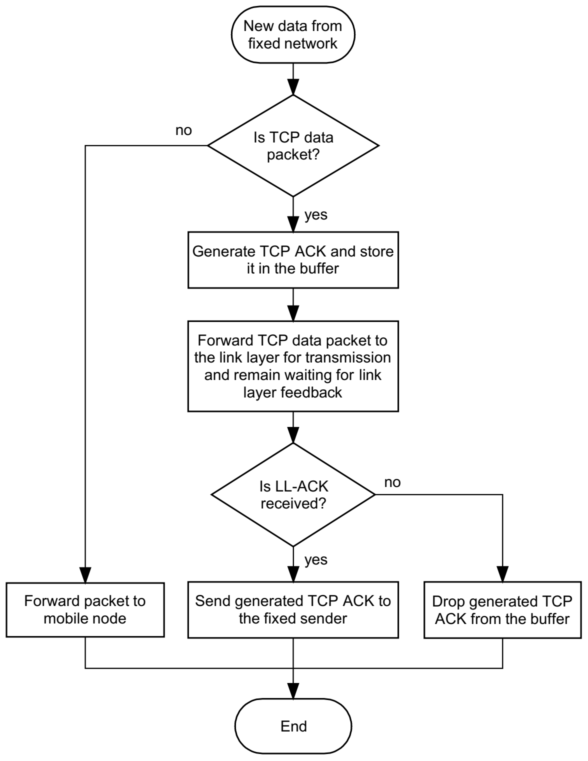
New data from fixed network
Is TCP data packet?
no
yes
Generate TCP ACK and store it in the buffer
Forward TCP data packet to the link layer for transmission and remain waiting for link layer feedback
Is LL-ACK received?
no
yes
Forward packet to mobile node
Send generated TCP ACK to the fixed sender
Drop generated TCP ACK from the buffer
End

Fig. 3. ARQ agent flowchart.

LLE-TCP approach does not change or override any of the TCP flow control mechanisms. However, suppression of TCP ACK transmission over the wireless channel and corresponding impact on the delay component reduces the round trip time (RTT) of the connection. As a consequence, this results in 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 [22].

Regarding backward compatibility and incremental deployment, in case the base station does not support LLE-TCP functionality (which can be negotiated within the network registration procedure) the mobile station should use standard TCP implementation by switching off ARQ client software module.

In single-hop ad hoc network scenario (Fig. 1a) ARQ agent is implemented at the wireless sender protocol stack, and TCP ACKs generated are not routed over the network (like in case of infrastructure network scenario) but locally delivered to the TCP layer.

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 within 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.

3.2. TCP connection phases

Fig. 4 presents the modifications performed by LLE-TCP over the standard TCP functionality within the three main phases of the 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.

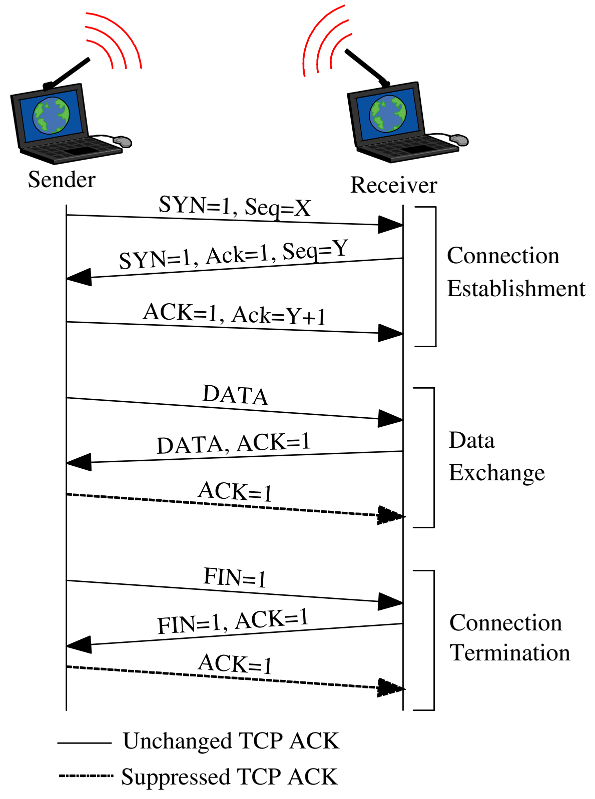

Fig. 4. Acknowledgement suppression performed by LLE-TCP.

In order to ensure proper functionality of the fast retransmit [23] introduced in TCP Reno, duplicate ACKs are never suppressed neither by ARQ agent at the receiver node nor at the sender node.

3.3. Congestion control

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 best-effort TCP is a relevant cause of the congestion. It continuously increases the outgoing rate until are start 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 Floyd et al. in [24].

A TCP connection between a fixed host and a mobile node traverses fixed and wireless sections of the network. Moreover, 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 due to congestion. For every packet drop, TCP window is decreased at least to its half and the dropped packet, which was already transported over the fixed network, is retransmitted on the whole end-to-end path.

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 was proposed in order to notify the sender node about the amount of free space left in the receiver's buffer [25]: 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 [26]. 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.

4. 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 [30] 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 and infrastructure wireless 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.

4.1. Simulation setup

In order to perform the experiments, the corresponding software modules of the ns-2 network simulator (version 2.28 from February 2005) 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. The chosen channel frequency of 2.472 GHz, transmission power of 31.6 mW, and receiver sensitivity threshold of 5.82×10^{-9} W correspond to a transmission range of 22.5 m. Results obtained during simulations are averaged over 10 runs with different seeds used for initialization of random generator while the duration of each simulation is set to 100 s. The rest of ns-2 configuration parameters are set to their default values unless explicitly specified.

Table 3
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

4.2. 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) [31] performance measurement tool is used for throughput measurements. Experimental results are averaged on 10 runs of 5 min 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 [32].

4.3. Single-hop network

The single-hop scenario is built by 2 nodes located within transmission range: one of them continuously transmits data, while the other one serves as a passive receiver.

Fig. 5a 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

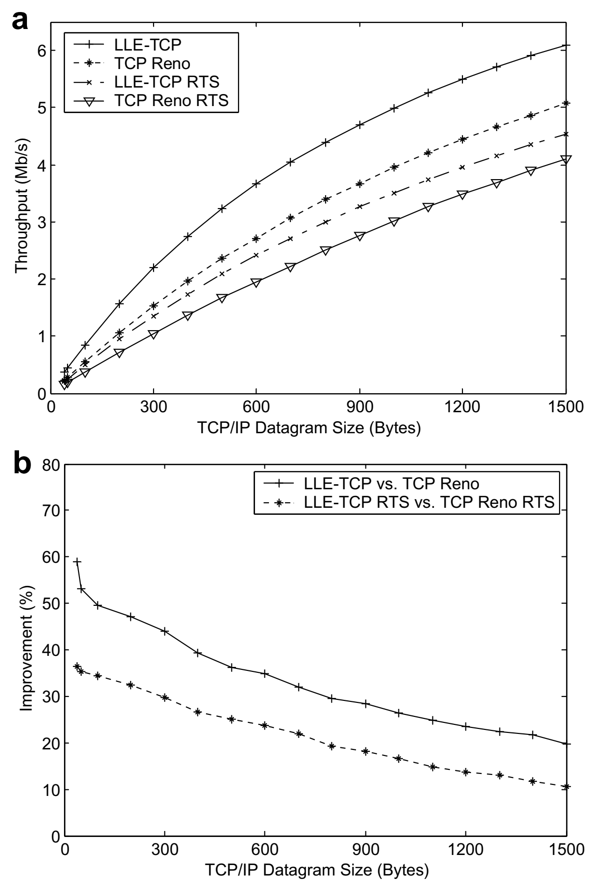

Fig. 5. (a) Throughput and (b) performance improvement achieved by LLE-TCP against TCP Reno (simulation results).

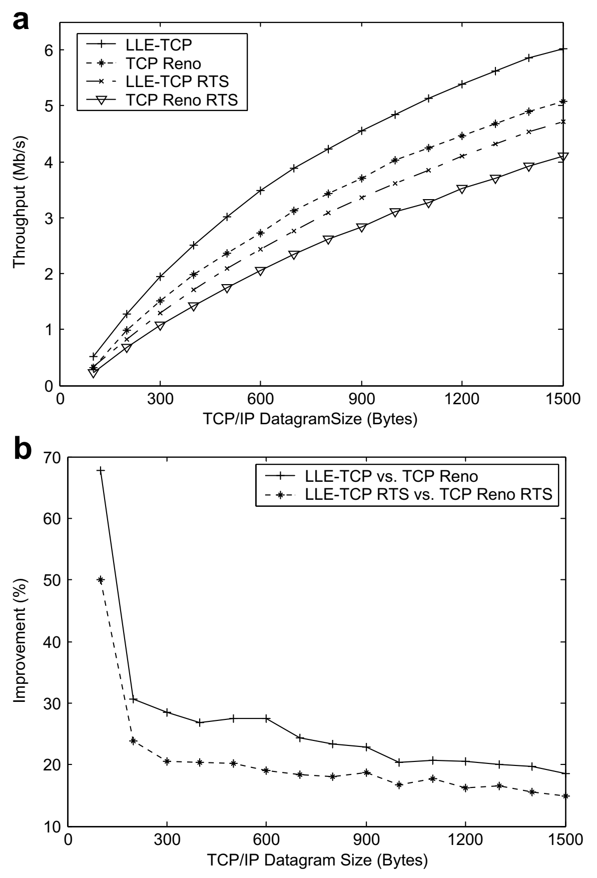

Fig. 6. (a) Throughput and (b) performance improvement achieved by LLE-TCP against TCP Reno (testbed experiments).

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 Fig. 5b), 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 rises 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).

Results obtained from the testbed experiments (see Fig. 6a) closely approximate those obtained from simulations. The average difference between simulation and testbed results is around 3%. The improvement level presented in Fig. 6b 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.

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. Fig. 7 presents simulation results in the presence of link errors with the TCP/IP datagram sizes fixed to 1 kbytes. Link error model is implemented on packet level and follows uniform distribution with a given packet error rate (PER).

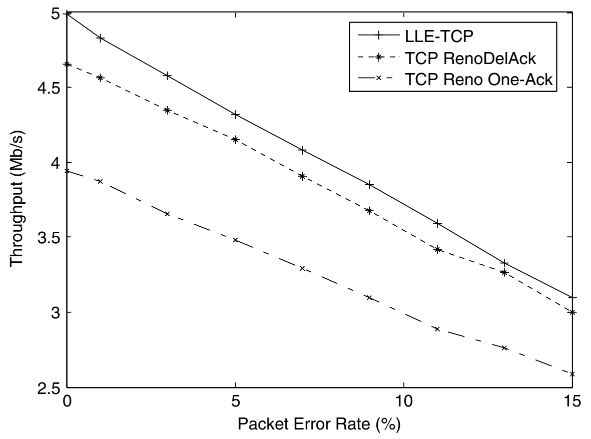

Fig. 7. LLE-TCP throughput comparison against TCP Reno with different acknowledgement strategies in different packet loss conditions.

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 [4].

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.

4.4. 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, 5 ms) mimic the case a coffee shop user connecting to an Internet server located within the city or a state. The access point ingress buffer policy is FIFO and buffer size is limited to 700 packets.

The fact that LLE-TCP proxy approach can be applied to different transport-layer TCP implementations motivated us to compare the results achieved by LLE-TCP implemented on top of the following transport layer protocols: TCP Reno, TCP NewReno [39], TCP Westwood [17], FAST TCP [41], TCP with selective acknowledgement (SACK) option [40], and TCP Illinois [42] – all considered suitable transport protocols for a heterogeneous network environment. The implementation code for the protocols not included into standard ns-2 distribution is obtained from packages provided by the authors. All the configuration parameters are set to their default values.

Throughput performance results with respect to different packet sizes are presented in Fig. 8. All the evaluated versions of TCP maintain a similar level of throughput due to simplicity of wireless-cum-wired topology as well as sufficient bottleneck buffer resources at the base station. In this scenario LLE-TCP performance ranges from 35% for large packets to 70% for small packets.

In error prone environment (Fig. 9) all TCP versions maintain similar level of throughput for small error rates ($PER < 0.20$). For higher error rates, TCP Reno and TCP NewReno degrade their performance down to zero for $PER = 0.32$, while LLE-TCP is able shift this point to $PER = 0.36$, leading to a throughput performance similar to that achieved by TCP SACK approach. However, the most relevant LLE-TCP performance improvement is demonstrated in case when TCP FAST with its fast window dynamics is not able to tolerate high error rates and drops the throughput down for $PER = 0.25$. In such scenario, the usage of LLE-TCP allows TCP FAST operation up to $PER = 0.37$.

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. This scenario is targeted to the evaluation of two parameters: fairness and coexistence.

Fairness is evaluated by computing flow throughput and Jain's fairness index [43] in the scenario with LLE-TCP enabled and comparing results with those obtained in the original scenario without LLE-TCP support.

Another important factor that must be considered for the design of a new protocol is coexistence with protocols already implemented in the network.

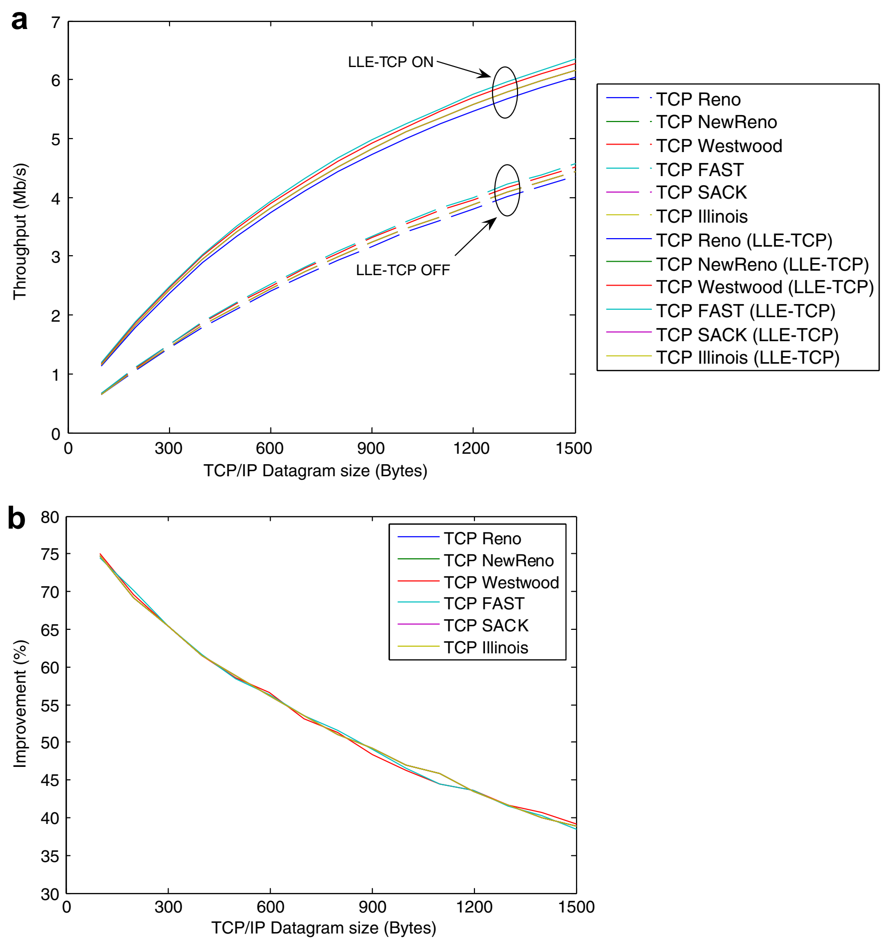

Fig. 8. (a) Throughput and (b) performance improvement achieved by LLE-TCP in infrastructure network scenario (with no wireless link errors).

The importance of coexistence factor is underlined in the evaluation of TCP Vegas [33], where it is shown that TCP Vegas can operate properly only in case it is the only transport protocol running in the network. Otherwise, congestion based TCP protocols like TCP Reno grab all the available bandwidth, dramatically reducing the throughput of TCP Vegas down to zero.

Fig. 10 illustrates results obtained with all 10 evaluated flows using the same TCP protocol version and no LLE-TCP enabled (Scenario 1) as well as with LLE-TCP enabled for 5 out of 10 evaluated flows (Scenario 2). Average and cumulative flow throughputs presented in Fig. 10 are extended with the computed Jain's fairness index and summarized in Table 4 for both scenarios.

The results demonstrate that LLE-TCP flows injected into a network running state-of-the-art TCP implementations achieve a good fairness level. The reason for that is in the design of LLE-TCP, which does not change flow control or error recovery of underlying TCP version – the whole optimization being done only by shifting TCP ACK generation point.

Slight unfairness of less than 1% (see Table 4) is an outcome of the reduced RTT for LLE-TCP flows. This phenomenon, known as RTT unfairness of TCP flows [44,45], is the consequence of receiver-driven congestion window evolution.

As an outcome of multi-flow the experiments, we observe the stability of LLE-TCP in multi-flow communication scenarios as well as good coexistence with all evaluated TCP versions. This underlines the possibility for an incremental deployment, i.e. there is no need to replace already existing products (drivers or wireless cards) which

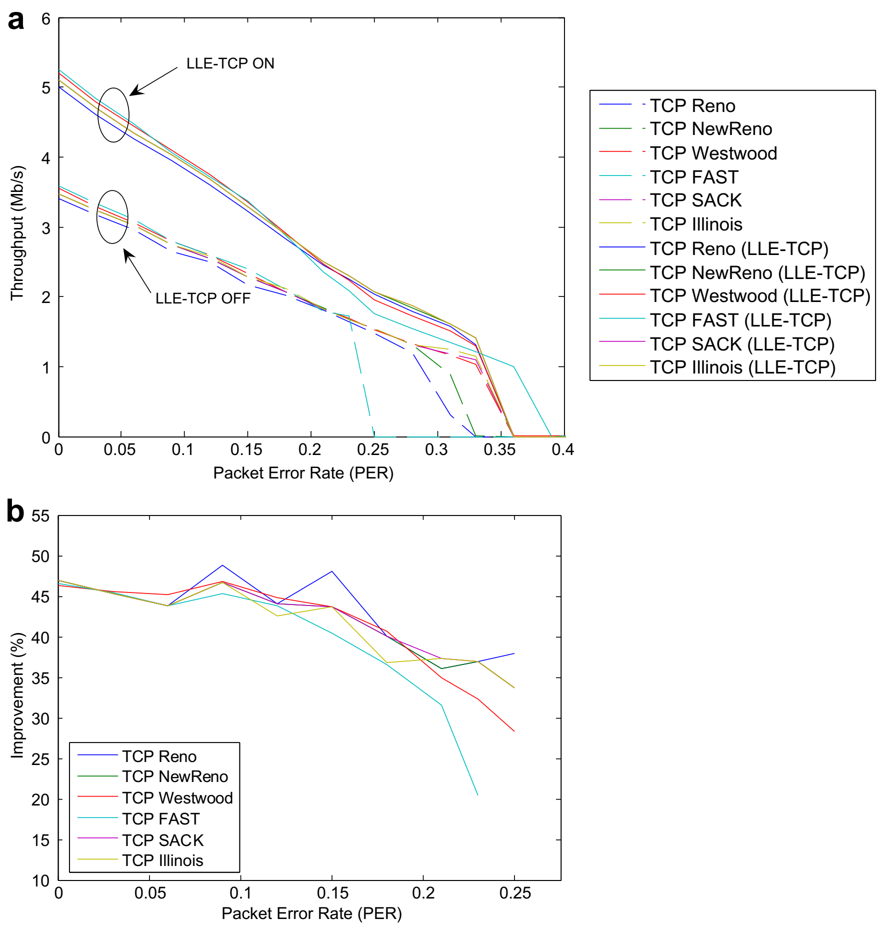

Fig. 9. (a) Throughput and (b) performance improvement achieved by LLE-TCP in error prone environment (TCP/IP datagram size is 1000 bytes).

do not support the LLE-TCP functionality while bringing the approach to the market.

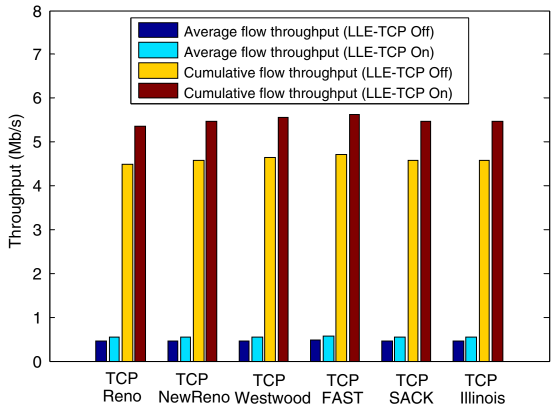

Fig. 10. LLE-TCP fairness and coexistence.

Table 4
LLE-TCP fairness

TCP protocol	Jain's fairness index (%)	
	Scenario 1	Scenario 2
TCP Reno	0.95	0.94
TCP Newreno	0.97	0.95
TCP Westwood	0.99	0.98
TCP FAST	0.97	0.97
TCP SACK	0.90	0.89
TCP Illinois	0.68	0.68

Good level of fairness and cumulative throughput values underline that the capacity increase coming from TCP ACK suppression over the wireless link is equally shared between contending flows, leading to relevant benefits for the network users.

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

5. Conclusions

This paper presented 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, LLE-TCP, 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).

While LLE-TCP is designed for any networks which employ multiple ARQ schemes at different layers of the protocol stack, main benefits are achieved over wireless links with 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 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.

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.

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.

References

[1] T.S. Rappaport, A. Annamalai, R.M. Buehrer, W.H. Tranter, Wireless communications: past events and a future perspective, IEEE Communications Magazine 40 (2002) 148–161.

[2] K. Pentikousis, TCP in wired-cum-wireless environments, IEEE Communications Surveys 3 (2000) 2–14.

[3] J.B. Postel, Transmission Control Protocol, RFC 793, September, 1981.

[4] G.J. Miller, K. Thompson, R. Wilder, Wide-area Internet traffic patterns and characteristics, IEEE Network 11 (6) (1997) 10–23.

[5] M. Zeng, A. Annamalai, V.K. Bhargava, Recent advances in cellular wireless communications, IEEE Communications Magazine 37 (1999) 128–138.

[6] IEEE 802.11 Wireless Local Area Networks. Available from: <http://grouper.ieee.org/groups/802/11/>.

[7] A. Ghosh, D.R. Wolter, J.G. Andrews, R. Chen, Broadband wireless access with WiMax/802.16: current performance benchmarks and future potential, IEEE Communications Magazine 43 (2005) 129–136.

[8] F. Granelli, D. Kliazovich, Nelson L.S. da Fonseca, Performance limitations of IEEE 802.11 networks and potential enhancements, Wireless LANs and Bluetooth, Nova Science Publishers, Hardbound, 2005.

[9] Ye Tian, Kai Xu, N. Ansari, TCP in wireless environments: problems and solutions, IEEE Communications Magazine 43 (3) (2005) S27–S32.

[10] H. Balakrishnan, S. Seshan, E. Amir, R. Katz, Improving TCP/IP performance over wireless networks, in: Proceedings of the 1st ACM International Conference on Mobile Computing and Networking (MOBICOM), Berkeley, CA, November, 1995.

[11] H. Balakrishnan, R. Katz, Explicit loss notification and wireless web performance, in: Proceedings of IEEE Globecom Internet Mini-Conference, Sydney, Australia, November, 1998.

[12] A. Bakre, B.R. Badrinath, I-TCP: indirect TCP for mobile hosts, in: Proceedings of the 15th International Conference on Distributed Computing Systems, June, 1995, pp. 196–143.

[13] K. Brown, S. Singh, M-TCP: TCP for mobile cellular networks, Computer Communication Review. 27 (5) (1997).

[14] J. Border, M. Kojo, J. Griner, G. Montenegro, Z. Shelby, Performance enhancing proxies intended to mitigate link-related degradations, IEFT, RFC 3135, June, 2001.

[15] Z.H. Haas, Design methodologies for adaptive and multimedia networks, Guest Editorial, IEEE Communications Magazine 39 (11) (2001) 106–107.

[16] Qi Wang, M.A. Abu-Rgheff, Cross-layer signaling for next-generation wireless systems, in: IEEE Wireless Communications and Networking Conference (WCNC 2003), vol. 2, March, 2003, pp. 1084–1089.

[17] S. Mascolo, C. Casetti, M. Gerla, M. Sanadidi, R. Wang, TCP Westwood: end-to-end bandwidth estimation of efficient transport over wired and wireless networks, in: Proceedings of ACM Mobicom, Rome, Italy, 2001.

[18] L.A. Grieco, S. Mascolo, Performance evaluation and comparison of Westwood+, New Reno, and Vegas tcp congestion control, ACM Computing and Communication Review 34 (2004) 25–38.

[19] D. Kliazovich, F. Granelli, D. Miorandi, TCP Westwood+ enhancements for high-speed long-distance networks, in: IEEE International Conference on Communications (ICC'06), Istambul, Turkey, June, 2006.

[20] C.P. Fu, S.C. Liew, TCP Veno: TCP enhancement for transmission over wireless access networks, IEEE JSAC 21 (2) (2004) 216–228.

[21] L. Brakmo, L. Peterson, TCP Vegas: end to end congestion avoidance on a global Internet, IEEE JSAC 13 (8) (1995) 1465–1480.

[22] F. Kelly, Mathematical modelling of the Internet, in: B. Engquist, W. Schmid (Eds.), Mathematics Unlimited – 2001 and Beyond, Springer, Berlin, 2001, pp. 685–702.

[23] W. Stevens, TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms, IETF, RFC 2001, January 1997.

[24] S. Floyd, K. Fall, Promoting the use of end-to-end congestion control in the Internet, IEEE/ACM Transactions on Networking 7 (1999) 458–472.

[25] D. Clark, Window and Acknowledgement Strategy in TCP, RFC 813, July 1982.

[26] Z. Fu, H. Luo, P. Zerfos, L. Zhang, M. Gerla, The impact of multihop wireless channel on TCP performance, IEEE Transactions on Mobile Computing 4 (2005) 209–221.

[27] R. Rejaie, M. Handley, D. Estrin, RAP: an end-to-end rate-based congestion control mechanism for realtime streams in the Internet, in: IEEE Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies, Proceedings (INFOCOM), vol. 3, March, 1999, pp. 1337–1345.

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

[29] E. Kohler, M. Handley, S. Floyd, Designing DCCP: congestion control without reliability, in: Proc. Of ACM SIGCOMM, 2006.

[30] NS-2 simulator tool home page, 2000. Available from: <http://www.isi.edu/nsnam/ns/>.

[31] Iperf performance measurement tool. Available from: <http://dast.nlanr.net/Projects/Iperf>.

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

[33] L. Brakmo, L. Peterson, TCP Vegas: end to end congestion avoidance on a global Internet, IEEE Journal on Selected Areas in Communication 13 (8) (1995) 1465–1480.

[34] D. Kliazovich, F. Granelli, A cross-layer scheme for the TCP performance improvement in wireless LANs, in: IEEE Global Communications Conference, GLOBECOM'04, Dallas, USA, December, 2004.

[35] M. Handley, S. Floyd, J. Padhye, J. Widmer, TCP friendly rate control (TFRC): Protocol Specification, IETF draft, RFC 3448, March, 2007.

[36] O.B. Akan, I.F. Akyildiz, ARC: the analytical rate control scheme for real-time traffic in wireless networks, IEEE/ACM Transactions on Networking 12 (4) (2004).

[37] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, S.C. Diot, Packet-level traffic measurements from the Sprint IP backbone, IEEE Network 17 (6) (2003) 6–16.

[38] Kuang-Yeh Wang, S.K. Tripathi, Mobile-end transport protocol: an alternative to TCP/IP over wireless links, in: IEEE Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), vol. 3, March–April, 1998, pp. 1046–1053.

[39] S. Floyd and T. Henderson, New Reno modification to TCP's fast recovery, RFC 2582, April, 1999.

[40] M. Mathis, J. Mahdavi, S. Floyd, A. Romanow, TCP selective acknowledgement options, RFC 2018, April, 1996.

[41] C. Jin, D.X. Wei, S.H. Low, TCP FAST: motivation, architecture, algorithms, performance, in: IEEE Infocom, March, 2004.

[42] S. Liu, T. Basar, R. Srikant, TCP-Illinois: a loss and delay-based congestion control algorithm for high-speed networks, in: International Conference on Performance Evaluation Methodologies and Tools (VALUETOOLS), Pisa, Italy, October, 2006.

[43] R. Jain, D. Chiu, W. Hawe, A quantitative measure of fairness and discrimination for resource allocation in shared computer systems, in: DEC Research Report TR-301, September, 1984.

[44] K. Tan, J. Song, Q. Zhang, M. Sridharan, A compound TCP approach for high-speed and long distance networks, in: Microsoft Press, MSR-TR-2005-86, July, 2005.

[45] G. Marfia, C.E. Palazzi, G. Pau, M. Gerla, M.Y. Sanadidi, M. Roccetti, TCP-Libra: exploring RTT Fairness for TCP, UCLA Computer Science Department Technical Report # UCLA-CSD TR-050037.

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. From September 2005 to February 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.

Fabrizio Granelli was born in Genoa in 1972. He received the "Laurea" (M.Sc.) degree in Electronic Engineering from the University of Genoa, Italy, in 1997, with a thesis on video coding, awarded with the TELECOM Italy prize, and the Ph.D. in Telecommunications from the same university, in 2001. Since 2000 he is carrying on his teaching activity as Assistant Professor in Telecommunications at the Dept. of Information and Communication Technology – University of Trento (Italy). In August 2004, he was visiting professor at the State University of Campinas (Brasil). He is author or co-author of more than 60 papers published in international journals, books and conferences. He is guest-editor of ACM Journal on Mobile Networks and Applications, special issues on "WLAN Optimization at the MAC and Network Levels" and "Ultra Wide Band for Sensor Networks". He is General Vice-Chair of the First International Conference on Wireless Internet (WICON'05) and General Chair of the 11th Workshop on Computer-Aided Modeling and Design of Communication Links and Networks (CAMAD'06).

His main research activities are in the field of networking and signal processing, with particular reference to network performance modeling, medium access control, wireless networks, next-generation IP, and video transmission over packet networks. He is Senior Member of IEEE.

Mario Gerla received a graduate degree in Engineering from the Politecnico di Milano in 1966, and the M.S. and Ph.D. degrees in Engineering from UCLA in 1970 and 1973. He became IEEE Fellow in 2002. After working for Network Analysis Corporation, New York, from 1973 to 1976, he joined the Faculty of the Computer Science Department at UCLA where he is now Professor. His research interests cover distributed computer communication systems and wireless networks. He has designed and implemented various network protocols (channel access, clustering, routing and transport) under DARPA and NSF grants. Currently, he is leading the ONR MINUTEMAN project at UCLA, with focus on robust, scalable network architectures for unmanned intelligent agents in defense and homeland security scenarios. He is also conducting research on scalable TCP transport for the Next Generation Internet (see www.cs.ucla.edu/NRL for recent publications).