On the Practical Feasibility of Fair TCP Communications in IEEE 802.11 Based Multihop Ad Hoc Wireless Networks

Inauguraldissertation

zur
Erlangung der Würde eines Doktors der Philosophie
vorgelegt der
Philosophisch-Naturwissenschaftlichen Fakultät
der Universität Basel
von

Evgeny Osipov

aus Krasnoyarsk, Russland

Basel, 2005

Genehmigt von der Philosophisch-Naturwissenschaftlichen Fakultät

auf Antrag von

Prof. Dr. Christian Tschudin, Universität Basel
(Dissertationsleiter)

Prof. Dr. Petri Mähönen, Universitätsprofessor an der Rheinisch-Westfälischen Technischen Hochschule Aachen
(Korreferent)

Basel, den 22. November 2005

Dekan

Prof. Dr. Hans-Jakob Wirz

Abstract

In the center of this dissertation is the question whether it is practically feasible to achieve deterministically good quality for traditional network services such as file transfer and Web browsing in multihop wireless mobile ad hoc networks (MANETs).

Despite the straight forward benefits of MANETs such as quick installation due to the absence of wireline infrastructure, and the virtue of dynamic re-configuration, these networks mainly exist in research labs so far. One of the stumbling blocks which prevents MANETs from wide deployment and popularization, is the poor and unstable performance of the TCP protocol which underlies file transfer and Web traffic. In particular, the problem considered in this thesis is the severe unfairness between multiple TCP sessions in a wireless context.

Overall, the thesis explores the operational range of MANETs in which the quality of network services is acceptable for an end user. The first part of our work reveals that this range is extremely narrow for the plain combination of TCP and wireless communication according to IEEE 802.11.

The second part of this work studies the interactions of TCP and IEEE 802.11 assuming static routing in the network. It gives a systematic view on fairness in MANETs. The max-min fairness model from the wireline Internet is adapted to the specifics of the wireless environment. The resulting solution presented in this thesis is an adaptive distributed capacity allocation scheme for multihop wireless networks. It leads to a dramatic improvement of TCP performance and a significant extension of the operation range.

The third part analyzes the effect of ad hoc routing on the quality of TCP sessions. The routing traffic itself is one of the reasons for unfair TCP communications.

Finally, the thesis addresses implementation issues of the suggested fairness model. It describes a distributed protocol for the dynamic control of the network load, which is implemented both for a network simulator and a real-world operating system.

Acknowledgments

I would like to express my professional gratitude to my scientific adviser Prof. Dr. Christian Tschudin. What I regard as the main driving forces of this dissertation are his ability to catch the ideas "on-the-fly" and the comfortable and productive atmosphere that he created in our Computer Networks group. All these factors made the work of planning and writing the thesis an enjoyable task.

I would like to acknowledge the participation in my professional life of all members of the Computer Network Architectures lab at Swedish Institute of Technology (SICS) and in particular its head Dr. Bengt Ahlgren. An important part of the dissertation was developed during my stay at the CNA lab as a visiting researcher.

This dissertation would hardly have been possible without the overall support from my wife Ekaterina. Thank you for your love, patience and acceptance of our nomadic life.

Wholeheartedly I express my thankfulness and love to my parents and sister who all these years supported, encouraged and cheered me up from Krasnoyarsk.

I would like to thank my friend Nikolay Sokrut and his family for giving a hand to my family when it was most needed. Many thanks to the families of Dmitry Khoptyar, Sergey Sergeyev, Dmitry Glebov, Alexey Choulepnikov and all my friends from Russia and all over the world, who believed in me and supported me during my PhD studies.

Contents

List of Figures

Introduction

Mobile ad hoc networks (MANETs) entered our life recently with the tremendously rapid spread of the Wi-Fi broadband technology. In MANETs there are no base stations. If any two nodes are located within the radio line-of-sight of each other, they are allowed to communicate directly. On larger distances the nodes use multihop routing to deliver their packets to destinations. The straightforward benefits of ad hoc networks are quick installation due to absence of wireline infrastructure, mobility, since nodes can communicate while in motion and natural capabilities of reconfiguration and redeployment. These advantages make MANETs ideal for many applications, from personal area networks to large sensor networks. However, the native properties of radio transmission and frequent topology changes due to node mobility create many challenging research problems.

The problem addressed by this dissertation is poor TCP performance in ad hoc networks. This is one of the stumbling blocks which prevents these networks from wide commercial deployment. In our research we consider mobile ad hoc networks built using the IEEE 802.11 technology. The thesis presents our cross-layer architectural solution to the problem of severe TCP unfairness in MANETs. In our approach we do not modify the respective standards for the MAC layer (IEEE 802.11) nor the transport layer (TCP).

Contribution of the thesis

The solution presented in this dissertation is an adaptive distributed capacity allocation scheme for multihop wireless networks. The capacity is allocated on a per-session basis at a specific point in time during the route establishment. This work makes contributions in three general areas: The analysis of TCP performance in multihop wireless networks, the analysis of the impact of ad hoc routing protocols on the quality of TCP connections and the distributed discovery of the network state in MANETs. Specifically, my contribution is

1. Adaptation of the fairness framework from the wireline Internet to the case of MANETs; Derivation of the adaptive rate limit for the outgoing traffic at ingress nodes which permits a fair distribution of the network bandwidth between competing end-to-end sessions;

2. Development of a methodology for the analysis of interactions between ad hoc routing protocols and TCP communications in MANETs and estimation of the routing load in the network;

3. Specification and implementation of a distributed algorithm for gathering the information about the presence of active end-to-end data communications in mobile ad hoc networks.

The major result of this dissertation is that the unfairness virtually vanishes when the suggested mechanisms are implemented in MANETs. The direct consequence of this work is guaranteed stable non-interruptive service for MANET applications including traditional FTP, Web, interactive SSH sessions and UDP-based sessions.

Organization of the dissertation

- Chapter 1 guides the reader through the key operations of wireless network technology in general and mobile ad hoc networks in particular. It introduces the IEEE 802.11 standard, the TCP/IP protocol stack and routing

solutions specific to MANETs. A reader familiar to the areas of networking and wireless networking may skip this overview and proceed with the "Problem statement" (Chapter 2).

- Chapter 2 is an introduction to the problem area addressed by this dissertation. We illustrate the problem of severe TCP unfairness in MANETs by an extensive set of simulations. We show that even under optimal network and routing configurations the performance of the combination of IEEE 802.11 and regular TCP is not at all acceptable for the end-user. We also review the existing approaches for mitigating bad TCP performances as reported in the literature. At the end of the chapter we present the motivation for this work.

- Chapter 3 is a short chapter where we outline our approach for solving the problem. It serves as a guideline to Chapters 4, 6 and 7 where we develop each part in details.

- Chapter 4 addresses our first contribution: We examine the properties of the radio transmission medium and TCP protocol. We formally describe a fairness framework for MANETs. To conform the data traffic to the formulated fairness framework we estimate a boundary load of multihop 802.11 based networks and derive the rate limits for the outgoing traffic at ingress nodes.

- Chapter 5 presents a full scale experimental evaluation of the *space-load* fairness framework described in the previous chapter. We assess the validity of the proposed fairness model and used assumptions. Our main result is that applying the derived rate limits, almost perfect fairness is achieved for all practically meaningful numbers of active TCP sessions, also leading to an overall network throughput increase.

- Chapter 6 covers the second contribution of the thesis: We study the behavior of our ingress throttling scheme in the presence of ad hoc routing protocols. We develop a methodology for the analysis of the impact of the routing traffic patterns on the quality of the ongoing TCP sessions. We show that the routing traffic itself can be a reason for TCP unfairness in MANETs. We also identify the operational scale of MANETs where both

routing and data traffic can co-exist without significant degradation of the quality of end-to-end sessions.

- Chapter 7 presents the third contribution of this thesis: We elaborate on the implementation issues of the space-load fairness framework. We present a path density protocol (PDP) which gathers the necessary information for computing the ingress rate limits for individual node pairs. We validate our protocol both in simulations and real-world experiments. With this protocol we address the overall contribution of this dissertation – the practical feasibility of fair TCP communications in MANETs.
- Chapter 8 summarizes the achievements of this dissertation and discusses general insights obtained during the development of the topic. We outline future research directions and open problems that appeared during the work on the thesis.

Publications related to the topic of the dissertation

1. [15] Ch.Tschudin and E. Osipov, "Estimating the Ad Hoc Horizon for TCP over IEEE 802.11 Networks", In *Proc. MedHoc'04*, Bodrum, Turkey, June 2004.
2. [56] E. Osipov, C. Jelger, Ch.Tschudin, "TCP capture avoidance in wireless networks based on path length and path density", Technical Report CS-2005-003, University of Basel, Switzerland, April 2005.
3. [53] E. Osipov, "Empirical Upper Bound on TCP Transmission Rate for Guaranteed Capture-Free Communications in multihop IEEE 802.11 Based Wireless Networks", Technical Report CS-2005-001, University of Basel, February 2005.
4. [54] E. Osipov and Ch.Tschudin, "A path density protocol for MANETs", In Proc. IEEE ICPS Workshop on Multihop Ad hoc Networks: from theory to reality (REALMAN05), July 2005. (A revised version is to appear in *Ad Hoc & Sensor Wireless Networks*, Old City Publishing, 2005)

Chapter 1

Mobile ad hoc networks (MANETs)

In this chapter we present the state of art of wireless local networking technology and recapitulate the key aspects of the TCP/IP protocol stack relevant to the topic of the dissertation.

1.1 Evolution of wireless ad hoc networks

The history of wireless computer networks began a long way before the appearance of the currently popular IEEE 802.11 technology. Already in 1973 DARPA, the Defense Advanced Research Project Agency, initiated research on the feasibility of using packet-switched radio communications for reliable data transmission [37]. In fact TCP, the standard protocol for reliable data transfer, which is used in the Internet today, was originally built for low-reliability wireless packet radio networks. Within a decade from its start the DARPA Packet Radio Network (PRNET) evolved to a robust and operational experimental network [36]. However, remaining under governmental and military control this network remained in the experimental status.

1.1.1 WiFi and IEEE 802.11

The key event which pushed the packet radio networks to the public arena was a decision taken by US Federal Communications Commission (FCC) in 1985 to make 900 MHz, 2.4 GHz and 5.8 GHz bands of radio spectrum available for communication purposes without the need for a governmental license [77]. However, it took another three years until the work on a common wireless standard in the unlicensed spectrum began in 1988. Amazingly enough, the first basic specification of the standard IEEE 802.11 (known under the commercial brand name "Wi-Fi") appeared only nine years later in 1997 and it was finalized under the name IEEE 802.11b another two years after that in 1999. It is at that time when the term WiFi appeared on the market. Appearance of a cheap (less than $100) wireless adapter from Lucent in 1999 gave a push to the overall spread of the Wi-Fi technology which we are witnessing nowadays. We describe the basic functionalities of the IEEE 802.11 standard in Section 1.3.

The IEEE 802.11 compliant network adapters support two operational modes: the infrastructure based, which is used in wireless local area networks (WLANs and Hot-spots), and the infrastructure-less (or ad hoc) mode, which is used in ad hoc networks with arbitrary topologies. In the following two subsections we describe the major difference between the two network architectures.

1.1.2 Infrastructure based wireless local area networks

A WLAN typically extends an existing wireline local area network with the help of access points (AP). The access point is a device with both wireline and wireless network interfaces. WLANs are built by attaching the wireline end of the access point to the edge of the wireline local area network. Clients communicate with the AP over the wireless network interface.

The network topology of the infrastructure based network is a *star* with the AP in the center and all wireless clients are located on the one hop distance from the AP. This implies that two wireless stations located in the range of assured data reception of each other should communicate via the access point.

In the infrastructure based network the wireless client operate like a wireline client would. This means that a station either transmits locally originated data or

receives the data for which the station is the final destination.

1.1.3 Ad hoc networks

The main idea of ad hoc networks is that any two stations located in the range of assured reception of each other can communicate directly and not via an access point. As for the stations located on larger distances they must utilize a multihop routing and use other nodes as relays for their traffic to the respective destination. The infrastructure-less mode is specified in the IEEE 802.11. The IETF working group MANET created in 1998 [81] has as its primary goal to specify a standard for such a routing protocol.

1.2 Overview of IEEE 802.11

In this section we discuss the aspects of the IEEE 802.11 technology relevant to MANETs in general and to the topic of the dissertation in particular. We present the major functions of the IEEE 802.11 standard at the physical and MAC layers. Since the work presented in this dissertation does not concern modifications or analysis of the physical and MAC layers functionalities, we present a compact essence of their operations.

1.2.1 Physical layer

IEEE 802.11 capable devices operate within the 2.4 GHz (802.11, 802.11b and 802.11g) or 5GHz (802.11a) frequency bands[1]. Operating in these bands 802.11-based products do not require any licensing. Spread-spectrum techniques used in these standards increase reliability and allow many unrelated products to share the spectrum without explicit cooperation and with minimal interference.

[1]The information in Sections 1.2.1 and 1.2.2 is obtained from corresponding IEEE 802.11 standards in [79].

A. Spread spectrum techniques

The two spread spectrum techniques are specified for the physical layer of 802.11 wireless standard: the frequency hopping spread spectrum (FHSS) and direct sequence spread spectrum (DSSS).

The FHSS technique is used only in the original 802.11 standard. In this case, the 2.4 GHz band is divided into 75 1-MHz sub-channels. The sender and receiver agree on a hopping pattern, and data is sent over a sequence of the sub-channels. Each communication within the 802.11 network occurs over a different hopping pattern, hence the chance of two senders transmit on the same channel simultaneously is minimized.

The DSSS technique is used in all other modifications of the 802.11 standard. It divides the operational frequency band into 14 22-MHz channels. Data is sent across one of these 22 MHz channels without hopping to other channels. The error correction due to noise and interferences is done by a so called "chipping" technique, where each transmitted bit is converted into a series of redundant bit patterns. Although this technique adds a redundancy it minimizes the need for retransmissions.

B. Available data rates

The transmission rates of the devices that use the FHSS technique (original 802.11 standard compliant) are 1 Mb/s and 2 Mb/s. These devices cannot transmit with a higher rate because of specifics of the FHSS.

In the subsequent 802.11b standard DSSS was standardized as the only technique for the physical layer. This together with more advanced signal coding schemes allowed to specify the physical layer support for two new speeds 5.5 Mb/s and 11 Mb/s.

Further advances in the coding techniques allowed to standardize 54 Mb/s transmission rate in the 802.11a standard. The shift to the 5GHz frequency band made corresponding 802.11a devices incompatible with earlier *b*-based devices.

This disadvantage was overcome in the 802.11g standard which operates in the same frequency band as 802.11 and 802.11b, however allows almost as fast data rates as in IEEE 802.11a.

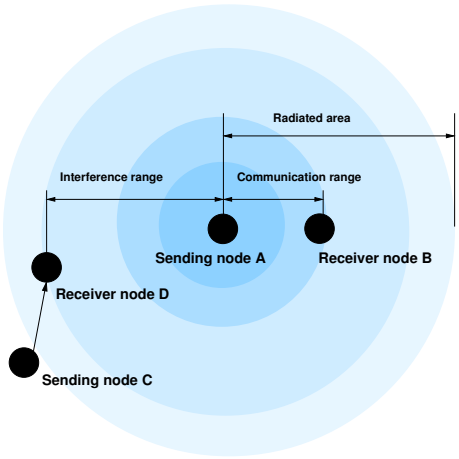


Figure 1.1: Radiated area, communication and interference ranges of a radio transmitting node.

C. Communication and interference ranges

Figure 1.1 schematically illustrates the concepts of radio transmission and communication regions. In the figure the circle area shaded with different color intensities illustrates the propagation of a signal transmitted from the sending node A. The radiated area of the node depends on transmission power of the node and the characteristics of the propagation environment, which can be an open space, urban or indoor. Let us consider the two regions of the radiated area, the communication and interference regions, in more details. We talk about communication region of node B with respect to node A and interference region of node A with respect to node D, which communicates with node C.

For both receiving nodes B and D the four primary factors which affect the size of their *communication* regions are:

1. Physical mechanisms of radio signal propagation,

2. Interference,

3. Receiver sensitivity,

4. The data rate of communication.

In addition to the natural attenuation of the radio signal with distance the primary physical mechanisms that affect radio communication range is *multi-path fading*. Multi-path fading occurs when multiple copies of the signal arrive at the receiving antenna at the same time but in different phase. Canceling out each other to a certain degree the resulting signal strength is reduced.

Interference with an intended signal occurs when the total strength of signals from other radio transmitters operating on the same radio frequency (transmission from node A in Figure 1.1) is higher than the strength of the intended signal (signal at node D from node C in the figure). The influence of interferences on radio transmission can vary from some reduction in throughput between two stations to complete blackout of the receiving station.

The receiver sensitivity and the data rate of communications are two factors which further diversify the communication range of wireless nodes. The receiver sensitivity of a wireless node indicates the level of signal strength that must be present to correctly receive data at a specified bit-error rate. The difference between the receiver sensitivity and the signal-to-noise ratio (SNR) is that it is a function of the data rate used by the transmitter. The theoretical receiver sensitivity can be calculated as in (1.1).

$$ReceiverSensitivity = N_t + N_s + 10log(BW) + SNR_{min}. \tag{1.1}$$

where N_t and N_s are respectively thermal and system noise, *BW* is the frequency bandwidth and SNR_{min} is the minimum SNR required for a given bit-error rate. The formula shows the well known fact that doubling the used data rate the receiver sensitivity decreases by 3 dB, which implies a shorter communication range for higher transmission speeds. Figure 1.2 shows the latest observation for the base 802.11 transmitting device with only two data rates 1 and 2 Mb/s.

It is important to note that there is no sharp transition between the communication zones at different transmission rates. First of all this is because the receiver sensitivity of wireless cards from different vendors are different. In addition to

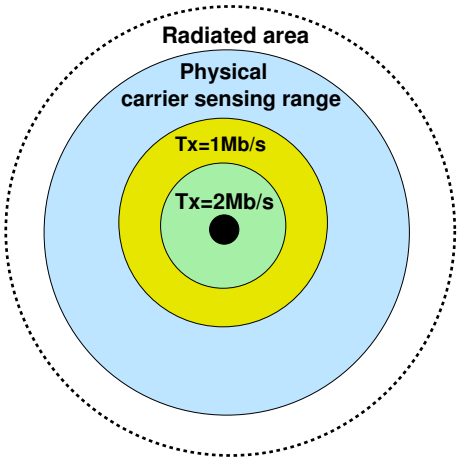


Figure 1.2: Communication ranges of a *base* IEEE 802.11 device.

this, the attenuation of the radio signal in the air depends on various factors. In [2] it is shown that even the cards from the same vendor show different communication ranges depending on the operating environment (either indoor, urban or open space) or even on weather conditions. Table 1.1 presents the estimated communication ranges for IEEE 802.11b wireless cards obtained by the authors during an extensive set of real time measurements.

	11 Mb/s	5.5 Mb/s	2 Mb/s	1 Mb/s
Data TX_range (m)	30	70	90 – 100	110 – 130

Table 1.1: Transmission ranges of IEEE 802.11b wireless cards at different data rates.

Analysis tools for MANETs

As we have seen in the previous section the communication ranges of the IEEE 802.11 based devices can be as large as several hundreds meters. Moreover they vary in size depending on the operating environment, vendor of the particular wireless equipment and even on weather conditions. This results in major difficulties for conducting repeatable real-world experiments involving mobility. Therefore, from the beginning of ad hoc networking simulators were probably the most important tool to investigate the performance of MANETs.

In our research we use a popular open source simulator ns-2 [82] developed originally at the University of California in Berkeley. Ns-2 is a discrete event simulator which provides a solid support for simulating the existing protocols of the TCP/IP stack over wireline networks. In its current version, thanks to the extensions done by MONARCH group [84], ns-2 has a stable support for accurately simulating the physical aspects of wireless MANETs. We do not describe all functionalities of ns-2 here and refer to [76] for a detailed description. Related to the topic of this dissertation we briefly describe the radio transmission model used in ns-2 and communication ranges assumed in the simulator.

There are three radio transmission models adopted in ns-2: A free space model, a two-ray ground model and a shadowing model. In our work we use the two-ray ground model. It is an extension to the free space model (Friis transmission equation [78]). It accounts for the multi-path fading effect and predicts the received signal power on large distances more accurately than the open space model. The received signal power at distance D from the transmitter is calculated by this model as:

$$P_r(D) = \frac{P_t G_t G_r (h_t h_r)^2}{D^4}. \tag{1.2}$$

In (1.2) P_t is the transmission power, G_t and G_r are antennae gains of the transmitter and receiver respectively and h_t and h_r are the corresponding heights of the antennae. The free space and two-ray ground models predict the received power as a deterministic function of distance between the communicating nodes. Both models represent the communication range as an ideal circle as shown in

Figure 1.2 for example. The reason for using the two-ray ground radio transmission model in our work is that it gives us an opportunity to work deterministically with the transmission ranges, which we need for the construction of the simulation scenarios. Using the parameters for the 914 MHz Lucent WaveLAN DSSS radio interface, the radius of the communication region of a node (base rate of 1Mb/s) assumed in ns-2 is 250 m and the radius of the interference zone is 550 m.

1.2.2 IEEE 802.11 MAC

The IEEE 802.11 standard specifies a carrier sense multiple access with collision avoidance (CSMA/CA) as the medium access protocol. Collision detection (CD), as is deployed in IEEE 802.3 Ethernet, cannot be used for the radio transmissions of IEEE 802.11. The reason for this is that when a node is transmitting, it cannot hear the transmission from any other node at the same time, since its own signal will dominate on any other signal arriving at the node. The IEEE 802.11 MAC protocol accounts for this specific and defines the medium access procedure as described in the following subsections.

A. Medium access

In IEEE 802.11 MAC, when a node has a packet pending for transmission, it first listens the medium to ensure that no other node is transmitting. If the channel is clear, the node transmits the packet. Otherwise, it chooses a random back-off interval which determines the amount of time the node must wait until it is allowed to transmit its packet. During periods in which the channel is clear, the transmitting node decrements its back-off counter. When the channel is busy it does not decrement its back-off counter. When the back-off counter reaches zero, the node transmits the packet. Since the probability that two nodes will choose the same back-off factor is small, collisions between packets are minimized.

B. Avoiding the hidden terminal problem

The hidden terminal problems is illustrated in Figure 1.3. As shown in the figure the hidden terminal problem appears when station A can communicate with

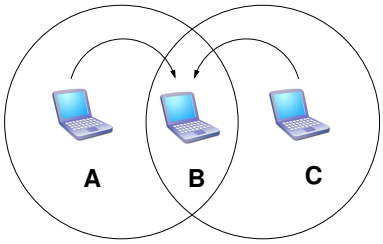


Figure 1.3: The hidden terminal problem.

station B, station B can communicate with station C, but station A cannot communicate with station C. Hence, sensing a clear channel nodes A and C can start transmission simultaneously to node B. As a result none of the packets will be received correctly by node B.

In order to minimize the effect from this problem, the IEEE 802.11 MAC standard foresees a so called *virtual* carrier sensing mechanism based on the exchange of short control messages between two communicating stations prior to data transmission.

Whenever a packet is to be transmitted, the transmitting node first sends out a short request-to-send (RTS) packet containing information on the length of the packet. If the receiving node hears the RTS, it responds with a short clear-to-send (CTS) packet. After this exchange, the transmitting node sends its packet. When the packet is received successfully the receiving node transmits an acknowledgment (ACK) packet. Note that the IEEE 802.11 standard for the MAC layer specifies this mechanism as optional. In the following subsection we show the impact of this mechanism on the available data rates.

C. Available data rates

As we have stated above, the IEEE 802.11 standards define multiple transmission rates: 1 Mb/s and 2 Mb/s for the base 802.11 standard and 1 Mb/s, 2 Mb/s, 5.5 Mb/s, 11 Mb/s for IEEE 802.11b. However, the real data throughput of 802.11 technology available to the users is lower. This is because of transmission overhead induced by different protocols on all layers of the TCP/IP stack as shown in

Figure 1.4 plus the overhead from the control traffic.

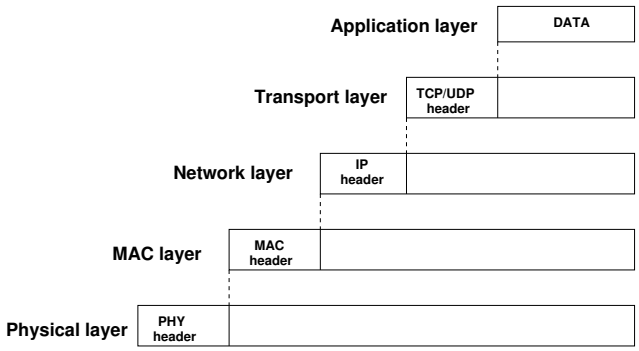


Figure 1.4: Transmission overheads.

In [2] the maximum throughput of IEEE 802.11 is studied for different transmission rates. Table 1.2 shows the actual data rate achievable by a user (i.e. application) for packet size of 1024 Bytes. Note that the throughput of 802.11 naturally decreases for packets of smaller sizes because of the increased control overhead.

	11 Mb/s	5.5 Mb/s	2 Mb/s	1 Mb/s
Data rate (no RTS/CTS) (Mb/s)	5.1	3.4	1.6	0.9
Data rate (RTS/CTS) (Mb/s)	4.3	3.0	1.5	0.8

Table 1.2: Actual data rate achievable for different transmission rates of IEEE 802.11b devices.

1.3 TCP/IP in MANETs

As we discussed in Section 1.1, the IEEE 802.11 wireless technology was introduced to the market as a useful extension to the wireline Internet when the later

gained an enormous popularity. The 802.11-based clients are legitimate Internet nodes, hence all wireless terminal devices such as portable notebooks, personal digital assistants etc. comply to the existing specification of the TCP/IP protocol stack. We refer to [80, 86] for detailed technical description of TCP and IP functionalities. In this section we discuss the aspects of the Transmission Control Protocol (TCP) which are relevant to the topic of dissertation.

1.3.1 TCP basic operations

TCP is the dominant transport layer protocol in the Internet for reliable end-to-end data transmission. There are several flavors of TCP which grew up form the original TCP Tahoe release [86]. According to recent studies [32] the most popular version is TCP New Reno [85]. Keeping the key functionalities of the original TCP Tahoe, New Reno differs by more advanced congestion control algorithm. We refer to the corresponding documentation for technical details.

TCP New Reno has four transmission phases: slow start and congestion avoidance, as in the regular TCP Tahoe, and fast recovery and fast retransmit. Like the original TCP, TCP New Reno maintains two variables:

- Congestion window (*CWND*) size,
- Slow start threshold (*sshtresh*).

As usual, a TCP connection between two nodes starts in slow start mode in which *CWND* is increased by one maximum segment size (*MSS*) for every received acknowledgment (*ACK*). In the slow start phase the *CWND* grows exponentially every round trip time. When *CWND* reaches the threshold value the TCP sender enters the congestion avoidance phase. In this phase TCP allows the sender to increase the transmission window linearly by one segment upon reception of a new ACK. The reception of an acknowledgment by the sender indicates that the last in-order packet is received successfully by the receiver. Unlike regular TCP Tahoe, which enters the slow start phase when packet loss occurs (indicated by arrival of duplicate ACKs), TCP New Reno invokes the fast retransmit and recovery phases. In these phases the sender reduces the congestion window to half size and linearly increases CWND as in congestion avoidance. TCP New Reno enters

the slow start phase only upon expiration of the exponential retransmission timer. These specifics of TCP New Reno result in a faster recovery from link congestion.

1.3.2 TCP performance in MANETs

Adopting the unmodified standard TCP to wireless networks immediately revealed serious performance problems [29, 21, 63]. During its evolution, TCP became a mature protocol fine-tuned to the specifics of wireline networks. The major assumption for TCP's congestion control mechanism is that packet losses are signals of network congestion. However, this assumption does not hold in wireless environment where high bit error rate, unstable channel characteristics and user mobility may contribute to packet losses. As a result of the erroneous interpretation of radio collision induced packet losses as a network congestion, TCP reduces its rate and its throughput decreases.

In ad hoc networks the combination of radio transmission medium and multihop transmissions places additional limitations on the TCP throughput. In the wireline Internet each hop is carried on a dedicated link. Thus, the transmission capacity of the links between hops are separated. This implies that a transmission originated on one link does not collide with the transmission ongoing on the link one or more hops away. In MANETs, however, we have a "super-shared" medium where multihop links belong to the same radio collision domain. This results in a very rapid drop of the throughput with the number of hops for a single TCP session as illustrated in Figure 1.5.

1.3.3 Proposed approaches for improving TCP performance in wireless networks

During the last decade several approaches for improving TCP performance in wireless networks were suggested. Overall they can be classified into two major categories: the split-connection approaches and the end-to-end approaches [21].

The main idea of split-connection approaches is to hide the mobility and wireless related problems from the TCP senders located in the wireline part of the Internet. The router on the border between the wireline and wireless networks behaves as a terminal for the two parts of the connection. Both sender and re-

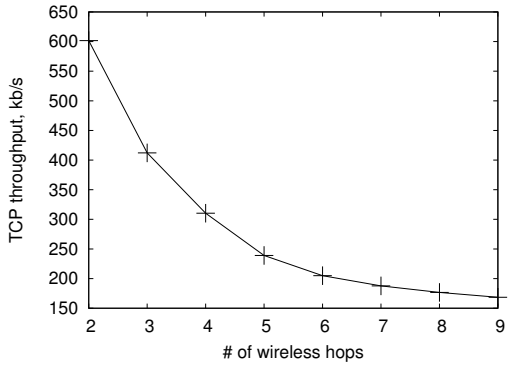


Figure 1.5: TCP throughput loss in multihop wireless networks. The results are obtained from simulations of a string topology with different numbers of wireless hops using *base* IEEE 802.11 devices.

ceiver nodes communicate with the splitting point independently. The intermediate router coordinates the transmissions between the end nodes by being the source of acknowledgments. The major representatives of this group are I-TCP [5], M-TCP [12] and WAP [88]. Breaking a TCP session in two parts, the split-connection approaches violate the end-to-end semantic of TCP. Since in the scope of this dissertation we focus on finding a solution which comply with the end-to-end paradigm of TCP we do not discuss these approaches further and refer to the corresponding references for more details.

The end-to-end approaches deal with adapting the TCP congestion control mechanism to the specifics of the wireless transmission medium. The first category of end-to-end approaches aims at creating means for TCP sender to distinguish between the packet losses caused by congestion, radio interference induced bit errors and unavailability of routes to the destination. The typical representatives of these approaches are ATCP [44], TCP-ELFN [29] and TCP-DCR [8]. The second type of proposals is receiver-oriented. These approaches try to steer the behavior of senders by means of smart acknowledgment generation techniques. The representatives of this type of schemes are a dynamic adaptive acknowledgment strategy [50], TCP-Eifel [45]. We discuss the end-to-end approaches in more

detail in the problem statement chapter while presenting the related work survey.

1.4 Ad hoc routing in MANETs

The overall idea of mobile ad hoc networks is to extend the limited coverage range of the radio transmitters. In MANETs each mobile node functions both as a host and as a relay node for traffic destined to other hosts. To accomplish this bi-functionality, every node in a network participates in an ad hoc routing protocol. When the idea of multihop wireless networking attracted a wide research community, many routing protocols for such networks appeared. The goal of the IETF working group MANET (Mobile Ad hoc NETworks) [81] is to unify and standardize the routing approaches for ad hoc networks. As a result of its work, two global routing approaches were identified: the *proactive* and *reactive* approaches.

The *proactive* approach is inspired by the routing experience in the wireline Internet. The routing topology is created prior to data transmissions from mobile nodes. The routing information is then dynamically updated according to changes of the network topology. In contrast, the *reactive* routing approach assumes no existing routing state in the network prior to data transmission from the particular station. Upon arrival of a first data packet the node enters a route discovery phase in which it announces the request for the particular destination address to the network. In reactive routing the routing information is maintained in the network only for the period of activity of the particular session. The major representatives of proactive routing is OLSR [16]. For reactive routing these are DSR [35] and AODV [57]. These protocols have been approved as “experimental standards”. We describe the major operations of these protocols in the following subsections.

Extending the fixed Internet, all routing protocols mentioned above are implemented on IP layer. Another approach, which currently is not considered in the scope of MANET working group, is that of link layer routing. In this section we also describe the Lightweight Underlay Network Routing protocol (LUNAR) as it is the base testing tool of the solution presented in this dissertation.

1.4.1 OLSR

The Optimized Link State Routing protocol (OLSR) is a proactive routing protocol for MANETs. Its proactive nature means that the protocol regularly exchanges topology information to all nodes of the network. OLSR is an IP layer routing protocol which makes no assumptions about the underlying link layer.

OLSR defines nodes with special functionality, the multi point relays (MPR). Each node selects a set of its neighbor nodes as MPRs. In OLSR, the MPR nodes are responsible for spreading the control traffic in the entire network.

Only MPR nodes have the task of maintaining link state information in the network. The link state information includes reachability to the nodes which have selected it as an MPR. The MPRs then compute the shortest path routes to any destination in the network.

A mobile wireless node selects those MPRs from the set of its one hop neighbors to which it has a bi-directional link. Therefore the problem of asymmetrical links is avoided in OLSR.

1.4.2 DSR

The Dynamic Source Routing protocol (DSR), as is clear from its name, uses the source routing paradigm. In this protocol each packet traversing the network carries the complete list of nodes on the path towards the destination.

The DSR protocol provides connectivity in ad hoc networks by two mechanisms: Route discovery and route maintenance. To perform a route discovery the source node broadcasts the route request message (RREQ) specifying a desired destination. Broadcasting of RREQs is done by a controlled flooding. During the propagation of RREQs the IP addresses of intermediate nodes are appended to the message. Upon reception of the RREQ message by the destination node it answers with the route reply (RREP) message which follows the discovered route in the reverse direction to the source node. In order to reduce the load created by re-broadcasting of RREQs each node in the network maintains a cache of source routes learned from overheard route reply messages.

If a link between two nodes goes down, the route maintenance mechanism is activated. When an intermediate node detects unavailability of nodes listed in

the source route of a packet, it generates a route error (RERR) message which is propagated back to the sending node. The sender reacts on the reception of the RERR message by either switching the traffic to another known route to the same destination or by initiating the route discovery phase again.

1.4.3 AODV

The Ad hoc On-demand Distance Vector routing protocol (AODV) utilizes the idea of reactive route discovery as in DSR but uses a hop-by-hop forwarding approach.

In order to differentiate between consequent RREQs for the same destination, AODV uses the concept of sequence numbers. When a source node wishes to find a route to a destination, the route request message is broadcasted to the one hop neighbors with the last known sequence number for that destination. The RREQs are re-broadcasted by the neighbors to the network until the message reaches either the destination node or an intermediate node which knows the path to the destination. When re-broadcasting RREQs the intermediate nodes create the route in the reverse direction to the source node. The forward route for the particular destination is created when an intermediate node receives the route reply message from the destination.

In order to maintain routes, AODV uses two different mechanisms. The first mechanism is based on periodic “HELLO” messages issued by every node. By means of these messages other nodes in the one hop neighborhood discover that this node is alive. The absence of HELLO messages from known neighbors within a certain time frame indicates “link” breakage to this node. In this case the node notifies any upstream node that has recently forwarded packets to a destination over the failed link by means of a route error message containing an infinite metric for that destination. The second mechanism for route maintenance is based on a cooperation between the link and routing layers. When a link layer cannot transmit a packet to the next hop it sends a Link Layer failure notification to the AODV process which initiates the route maintenance mechanism.

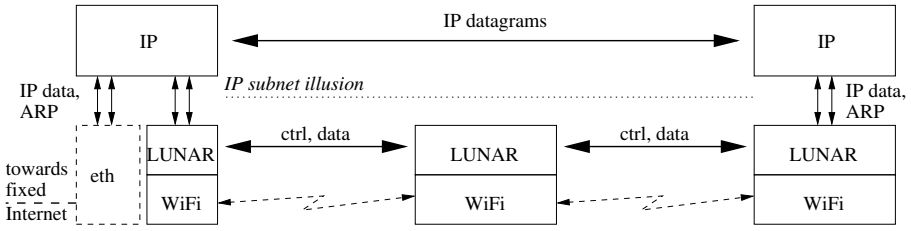


Figure 1.6: Position of LUNAR in the TCP/IP protocol stack.

1.4.4 LUNAR

LUNAR is situated at layer 2.5 of the TCP/IP stack ("just below IP") as shown in Figure 1.6. ARP requests are intercepted and turned into route requests which are transmitted across a wireless network using a simple mechanism of flooding and limited re-broadcasting (by default the range is limited to three hops). Traversing the network LUNAR's route request messages obtain a special forwarding label in each router along a path to a destination. When the destination is found the whole end-to-end path is mapped into a single label which is then converted to the format of an Ethernet address and passed to the ARP table of the source node. This creates an illusion of a subnet to the IP stack. LUNAR does not have mechanisms for route maintenance and reparation as other routing protocols, instead it rebuilds all routes for ongoing connections from scratch every three seconds. The value of three seconds was chosen with reference to the HELLO interval in AODV: it corresponds to two HELLO rounds which are needed by AODV to determine a change of a route. This operation positions LUNAR in the classification of the routing protocols somewhere in between the re- and proactive protocols: It starts to establish routes on demand as AODV or DSR, on the other hand it rediscovers the whole topology at fixed intervals. Understanding that complete rediscovery of the routes may lead to large bursts of broadcast traffic in networks with large number of nodes, the specification of the protocol explicitly states that LUNAR is designed for relatively small networks of dozen of nodes and a maximum route length of three hops.

1.5 Summary

In this chapter we described the state of art regarding the IEEE 802.11 wireless technology and mobile ad hoc networks (MANETs). The major concepts we emphasize from this overview are the complex structure of the communication regions of the IEEE 802.11 radio transmitters and the ubiquitous use of the TCP/IP protocol stack for communications over wireless ad hoc networks. We also highlighted the awareness of the research community about poor performance of the existing communication protocols when they are deployed in wireless networks without modifications. While there exists a number of attempts for mitigating the bad TCP performance in MANETs, the problem is still unsolved. In the next chapter we emphasize in particular the severe unfairness between multiple multihop TCP sessions, also known as TCP capture.

Chapter 2

Problem statement: The "Ad hoc horizon"

Ad hoc routing is the basic mechanism which enables connectivity in mobile ad hoc networks. It is natural then that the first performance measurements in MANETs were done on their scalability in terms of the number of participating network nodes and the length of the routes. Due to obvious difficulties of conducting large scale real-world experiments with radio transmitting devices these studies were done mainly using simulations.

In the literature we can find examples of simulation based analysis which explore networks of 50 nodes with actual paths of up to eight hops [11] or even networks of 10'000 nodes with routes over more than 100 hops [42]. These and similar optimistic scalability studies consider mainly CBR data sources. The justification of this choice is not fully stringent[1]. Having in mind conference, emergency or military scenarios as potential applications, it is difficult if not impossible to imagine how these scenarios can be served by CBR traffic only. Such

[1] Quote from [11] "As the goal of our simulation was to compare the performance of each routing protocol, we chose our traffic sources to be constant bit rate sources. ... We did not use TCP sources because TCP offers a conforming load to the network, meaning that it changes the times at which it sends packets based on its perception of the network's ability to carry packets. As a result, both the time at which each data packet is originated by its sender and the position of the node when sending the packet would differ between the protocols, preventing a direct comparison between them."

scalability investigations are in sharp contrast with the existing observations of the performance problems of the existing TCP/IP protocol stack while operating in MANETs [29, 43, 69, 58, 23, 67]. The developers working with real 802.11 equipment know that the corresponding findings bear little resemblance to reality. Experiences with the Ad hoc Protocol Evaluation testbed APE [47] indicate that paths longer than 3 hops are not reliable enough for even modest network usage.

We decided to put the concerns of the researchers about the poor quality of traditional network services such as web browsing or file transfer in the context of scalability studies of MANETs. In this chapter we conduct such investigations using simulations and define a scaling limit of current MANETs with respect to the quality of communications perceived by an end user. We call this limit the "ad hoc horizon" and define it as the region spanned by either the number of nodes or the number of hops beyond which TCP performance is not acceptable for ordinary end user tasks like web browsing. We estimate that this ad hoc horizon is currently located at 2 to 3 hops or 15 nodes for the combination of TCP and IEEE 802.11, thus confirm the assessment from practical experiments.

The purpose of this chapter is twofold. Firstly, we illustrate the severeness of TCP performance problems in MANETs. Secondly, analyzing the reasons for poor performance of TCP based communications and exploring the existing mitigating attempts, we define the scope of the problems which we address in this dissertation.

We develop our reasoning as follows. In Section 2.1 we describe our methodology on the analysis of the network performance for exposure of the "ad hoc horizon". In Section 2.2 we analyze the impact of different simulation settings and configuration of TCP and IEEE 802.11 MAC parameters on TCP performance. We show that the TCP unfairness problem persists even under optimal network and protocol configurations. In Section 2.3 we present a survey of the literature on the topic of poor TCP performance and mechanisms for its improvement in mobile ad hoc networks. Finally in Section 2.4 we summarize the material presented in this chapter and state our motivation for the research line presented in this dissertation.

2.1 TCP capture and "Ad hoc horizon"

The relevance of the ad hoc horizon relates to the huge deployment base of IEEE 802.11 devices and TCP protocol stacks: For the next years to come there will be no way around using them. Any proposal regarding modifying the MAC protocol or making TCP wireless-friendly will face considerable acceptance problems and will have interoperability issues with "legacy" equipment. The ad hoc horizon thus defines an area wherein the ad hoc routing protocols can safely operate with current devices and where they have a chance to deliver acceptable performances today. It is desirable that advances in research will extend this horizon, in which case this limit serves as a benchmark for any proposed solution to the TCP over 802.11 problem.

2.1.1 Methodology

This section describes our approach for isolating the performance envelope wherein a wireless ad hoc network delivers "useful" services. In a nutshell our methodology can be characterized as making optimal assumptions on the system that we want to test while focusing on the worst performances that can be observed in such settings. For example, we will introduce a family of static topologies and thus remove any bad effects due to mobility. On the other hand, we will not look at average performance figures but always select the poorest case that an unlucky end user might encounter.

A. The family of beam star scenarios

Simulation studies for ad hoc routing protocols, especially when it comes to comparisons and rankings, avoid choosing "special" scenarios and communication patterns. This helps to remove biased settings and avoids the tuning of a protocol for particular circumstances. In our case, however, choosing a special scenario has the status of a counter-proof as we are looking for worst case figures: If we can show bad performance in one scenario, it is possible that other scenarios exist that offer even worse conditions, which would only reinforce the findings.

In order to test our hypothesis of an ad hoc horizon we created a family of

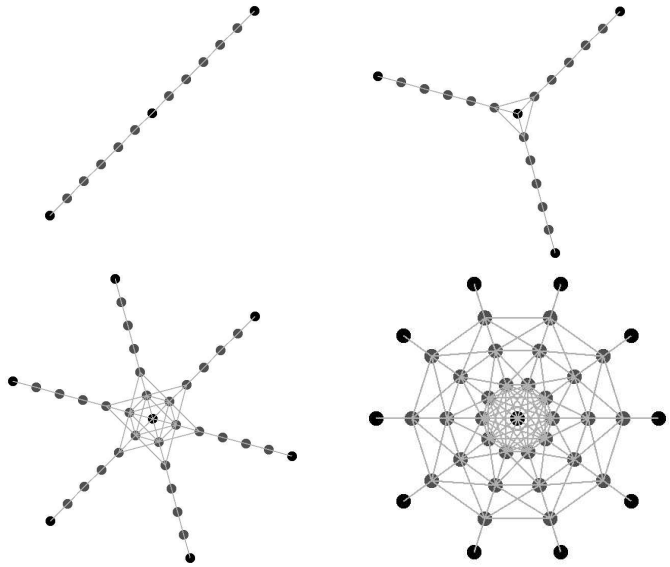


Figure 2.1: "Beam star" network topologies.

topologies called "beam star". A central node (which could be a gateway into the fixed Internet) serves as end or start point for all TCP sessions. From this central node, a varying number of beams of identical length emanate in a regular fashion. Figure 2.1 shows four instances of this topology family for various numbers of beams and beam lengths (2x6, 3x6, 6x6, 10x4) with potential connectivity. Note that the last figure is shown with a different scale: the internode distance along a beam is identical for all topologies.

The communication pattern consists of parallel FTP session. Each beam is used for an FTP transfer where the central node establishes sessions with each beam head. This results in a moderate number of sessions (one per beam) when compared to the total number of nodes, especially for long beam lengths. The total number of nodes is given by the formula $1 + beams \cdot beamlength$.

The family of beam star topologies permits us to study the effect of increasing path lengths and growing the number of nodes in a controlled manner. We expect both to have a negative impact on TCP performance. Thus, we can assess the combined limit for effective TCP data delivery in multihop ad hoc networks.

B. Worst case figures under optimal configurations

A "usefulness" metric from an end user's point of view is the time during which a TCP session does not make any progress, i.e. what the user perceives as a "frozen" connection. For each session we examine the sequence of TCP segments and record the time intervals during which TCP protocol waits for the reception of a new segment that can be delivered to the application. In order to account for short bottlenecks in the network and to mimic some amount of user tolerance, we only count no-progress intervals longer than 3 seconds. These larger than 3 seconds intervals are accumulated and put into relation to the total duration of the test run. The higher this ratio is, the more stammering was the session: A 100% *no-progress ratio* means a complete stall. For a given topology we compare the no-progress ratios of all TCP sessions and test runs and retain the highest ratio.

The other metric we examine is the *unfairness* among TCP sessions. In order to construct this metric we complement the value of the classic Jain fairness index:

$$u_1 = 1 - \frac{(\sum_{i=1}^{N} Thr_i)^2}{N \sum_{i=1}^{N} Thr_i^2}. \tag{2.1}$$

where Thr_i is the throughput of FTP session i and N is the number of active sessions. A value of 0 for the unfairness index u_1 means that all TCP sessions receive the same (although perhaps low) share of the network's total capacity, while an unfairness of 1 means that one session is monopolizing all bandwidth.

C. Measurement setup

In our simulation with ns-2 we used the settings presented in Table 2.1. The beamstar scenarios use an internode distance of 130 meters. The resulting potential connectivity patterns can be seen for the topologies in Figure 2.1.

IEEE 802.11 bandwidth	2 Mbps
Transmission range	250 m
Interference range	550 m
RTS/CTS handshake	On
Routing protocol	AODV-UU 0.8 [74] (AODV v13), local repair enabled
Application	FTP, random start within 10 sec
Simulation run	300 sec
Number of repetitions	6

Table 2.1: Simulation settings for the "Ad hoc horizon" experiment.

2.1.2 Exposing the "ad hoc horizon"

Using the approach described above, we present a series of simulation results that first identify the ad hoc horizon and then explore to which extent this finding depends on choices made.

A. TCP no-progress ratio

As a base case for demonstrating the ad hoc horizon we chose a setting that is optimal in terms of an ad hoc network: Nodes are stationary, we use ns-2's simplified 802.11 model with a single 2 Mbps transmission speed for unicast messages and the base 1 Mbps rate for broadcast traffic, the RTS/CTS handshake is enabled in all simulations, AODV is permitted to use link layer feedback and all TCP sessions run over the same number of hops.

In this setting we extract the worst no-progress ratio among all TCP sessions as defined in Section B. This ratio is then plotted against the number of beams and beam length, yielding a surface (see Figure 2.2). The diagram covers a broad range of different topologies. For example, the x-axis represents all "string topologies" from 1 to 10 hops, while the y-axis represents topologies where all nodes are

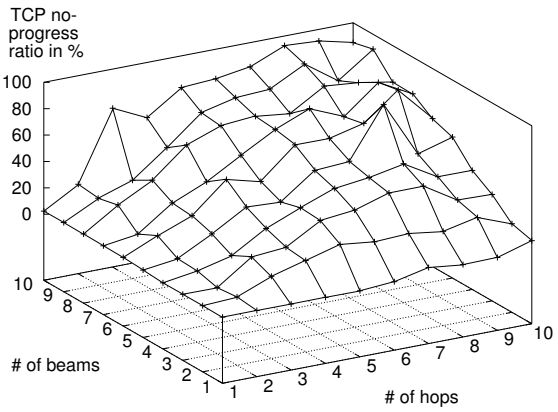


Figure 2.2: Worst accumulated TCP no-progress time for AODV, in dependency of the number of beams and their length.

within a 1-hop distance.

As a reading example for Figure 2.2, we note that with 2 beams an 3 hops (resulting in a network of 7 nodes) we have a stall-ratio of 30%, hence more than 30% of the time at least one session appeared to be "frozen". Not visible from the graph, but by inspection of the traces, this no-progress time contains stalls of up to 17 seconds.

The main observation that we point out is that there is a marked slope when moving from 2 to 3 hops. In the other dimension – the number of beams – we get similarly bad figures for two hops when we reach the number of six or seven beams. Together, this corresponds roughly to a region of 15 nodes or 3 hops, whatever dimension is explored first, beyond which the TCP no-progress ratio is 30% or more. We call the border of this region the "ad hoc horizon". It is a horizon in the sense that we don't see useful TCP services beyond this range. For an end user this means that at the edge of the ad hoc horizon he has to anticipate that his TCP session might be stalled for one third of the time.

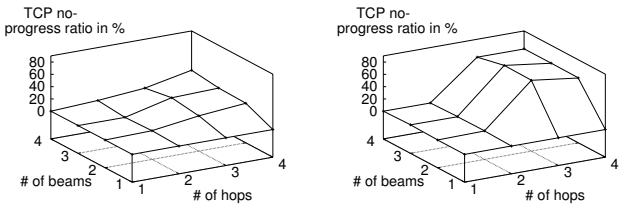


Figure 2.3: Comparing TCP no-progress time for AODV, with (left) and without (right) layer feedback.

B. Disabling link layer feedback

The use of link layer feedback is popular in simulation but is not easily available in reality. To get an assessment of the impact of this choice (and to explain the frustration of practitioners in the field), we ran the same simulation but disabled link layer feedback for AODV.

Figure 2.3 (right sub-figure) demonstrates the expected degradation in a drastic way: After 2 hops we literally hit a wall. Instead of a 30% stall ratio this number jumps to 60%. Note that the left figure (for a range of 1–4 beams and 1–4 hops) is identical to the corresponding region in Figure 2.2: In the case where AODV cannot use link layer feedback, we hit the 60% no-progress ratio already with 3 hops while with link layer feedback this plateau is reached only at about 6 beams and 6 hops.

C. TCP unfairness

Next we examined "TCP unfairness" (2.1) as another metric for capturing the potential dissatisfaction an end user might experience. In the ideal case, all TCP sessions are treated fairly. Users would probably accept reduced TCP throughput and even some stalls as long as they are spread over all users in a equal way.

In Figure 2.4 we plot TCP unfairness (with and without link layer feedback), i.e. how unequally the various TCP sessions are treated by the ad hoc network. As can be observed in the right sub-figure where link layer feedback is disabled,

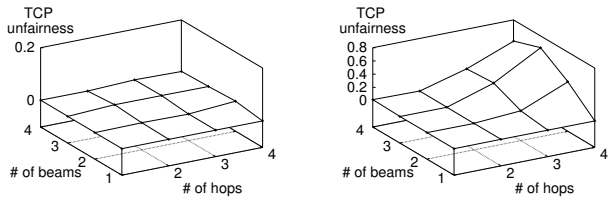


Figure 2.4: TCP unfairness for AODV, with (left) and without (right) link layer feedback.

we have again a slope starting at 3 or 2 hops, depending on the number of beams. This confirms – although less dramatically than with the no-progress time – that TCP's performance in an ad hoc network starts to seriously degrade after 2 hops. Note that the left figure has another scale and that with link layer feedback we have almost no unfairness in this region.

D. Using OLSR instead of AODV

In order to exclude that AODV's on-demand routing style is responsible for the observed TCP stalls and unfairness, we ran the simulations using the proactive routing protocol OLSR.

In Figure 2.5 we compare the TCP unfairness ratio when link layer feedback is disabled for both AODV (left figure) and OLSR (right figure). As can be seen, OLSR performs even worse and does not help to expand the ad hoc horizon.

2.1.3 Analysis and discussion

The ad hoc horizon identified in the previous section is mostly due to TCP's behavior, as we will discuss in this section. Although TCP unfairness over IEEE 802.11 is a well known problem (and we do not add new insights here), we examine more closely why TCP unfairness occurs in the beamstar scenarios. This will permit us to draw some conclusions on where modifications are required for the triple of TCP, ad hoc routing and the 802.11 MAC protocol.

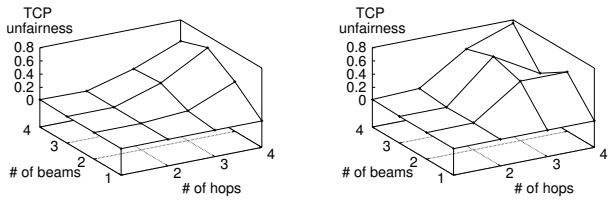


Figure 2.5: TCP unfairness without link layer feedback, comparing AODV (left) and OLSR (right).

A. Packet-level analysis of TCP behavior

A simple but nevertheless interesting scenario is the beam star topology with 2 beams and 3 hops. Figure 2.6 shows the communication pattern for two FTP sessions in that scenario where node N0 is the center node: For each of the beams an FTP session is started towards the beams' head nodes N3 and N6. The figure shows the first 100 seconds of a simulation run with dark areas indicating data traffic while the dashed area represents broadcast traffic due to AODV's control messages.

We see that initially both FTP flows can transfer some data from the center to their respective beam heads, but that repeated timeouts for the flow to the right leads to a rather large no-progress period of approximately 40 seconds! Later on, the flow to the right is able to become active again, but at the price of starving the left FTP flow.

Regarding the no-progress and the unfairness metrics plotted in Section 2.1.2 this means that although the fairness is not too bad for this simulation run, we have a serious no-progress problem.

For understanding why the right FTP session fails to become active again after a first timeout, we show in Figure 2.7 a detailed view of the packet exchanges around the time when the first loss of a TCP acknowledgment occurs, as they were extracted from the simulation traces.

In Figure 2.7 we have indicated "bow waves" with dashed lines. They rep-

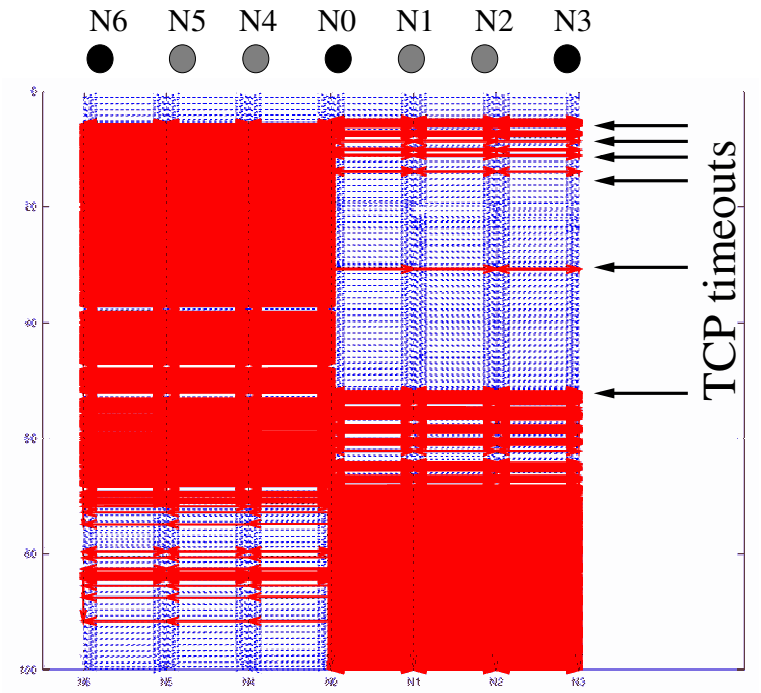


Figure 2.6: Alternating starving of the two beams' TCP sessions, leading to a low unfairness index but a high no-progress ratio.

resent the interference area of a speeding "forwarding train": As a packet is forwarded along a beam, it disturbs ripple like all communications around its transmission place. The bow wave labeled with (1) is due to the forwarding of a data packet from the center node to the leftmost beam head. The internode distance is set to 130 m which means that the interference range of 550 m covers 4 hops. During the time where the left beam's bow wave is covering most of the right beam, the right side cannot even do a single RTS: Event (a) shows such an unsuccessful attempt.

When the activity momentarily stops at the left side, the right side manages

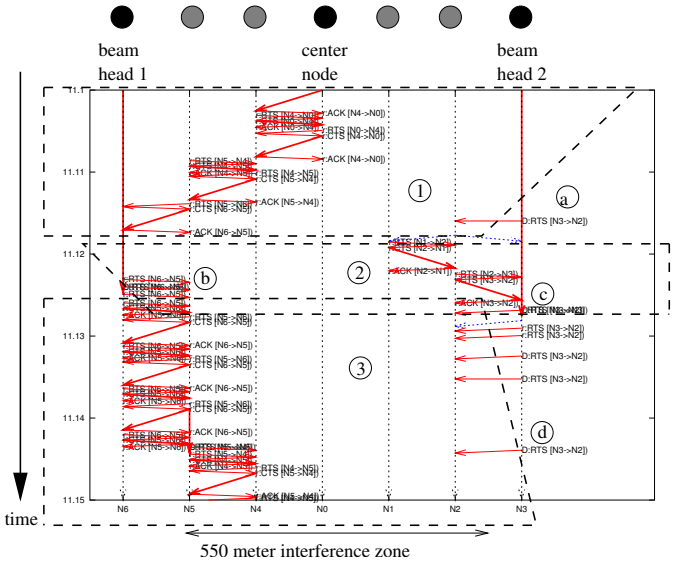


Figure 2.7: The seed of unfairness: RTS/CTS failure due to remote transmission activity.

to complete the forwarding of a data packet to the right most beam head. During this time, the bow wave labeled with (2) prevents the left side to proceed. Unfortunately for the right side, the left side regains control thanks to the persistence of the RTS/CTS scheme, where a new bow wave (3) starts immediately. This time the second-most left node is draining its packet queue towards the node N6: The previous bow wave also hindered the forwarding inside the left beam. Due to this massive blocking of the medium (which happens outside of the RTS/CTS coordination range for the waiting nodes on the right side), 3 RTS failures occur between event (c) and (d), eventually destroying the TCP ACK packet at the right most node that waited for expedition to the center.

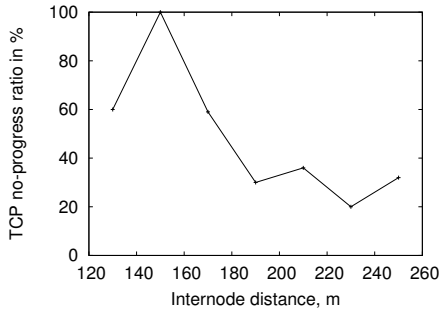


Figure 2.8: How changing the internode distance affects the no-progress metric for a 3-beam-2-hop scenario.

2.2 Searching for mitigating factors

Our main observations about the ad hoc horizon, presented so far, were based on rather specific network configurations and traffic patterns. We used fixed internode distances in the considered topologies, standard settings of the parameters of TCP and IEEE 802.11 MAC protocols and continuous data exchanges (full FTP streams) going on in parallel. In this section we present an additional set of experiments where by varying the simulation settings, network topology and protocol parameters we search for factors mitigating the poor TCP performance.

2.2.1 The effect of internode distance on TCP performance

After having seen the reach of the interference zone for a simple 2-beam-3-hop scenario, we wondered to which extent the distance between nodes would affect our ad hoc horizon result. We used the same 2-beam-3-hop scenario as above and increased the internode distance from the initial 130 m to the transmission limit of IEEE 802.11 in ns-2 that is, 250 m.

As can be seen in Figure 2.8, an internode distance of 150 m would produce even worse figures and thus yield a more marked ad hoc horizon. Whether this 150 m distance would also have been “optimal” for other beam stars has to be ex-

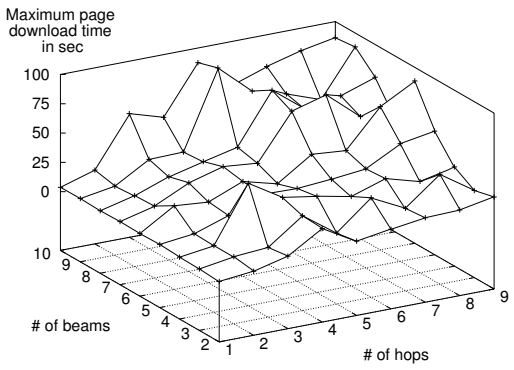


Figure 2.9: Download times for HTTP sessions competing with a single FTP session in the beamstar scenarios.

plored. Note that even when we had chosen an internode distance of 200 or more meters, the no-progress time ratio would still have landed in the unacceptable 20 to 30% range.

2.2.2 Using HTTP instead of FTP traffic

One possible concern with our beam-star approach is that it is not reflecting reality where, for example, we expect more intermittent HTTP traffic. The question is whether a lighter traffic load will produce less pessimistic results. We performed a simulation experiment where short HTTP requests have to compete against a single FTP session.

Unfortunately, HTTP traffic suffers in a way similar to the case of competing FTP sessions: Figure 2.9 shows the maximum download time that users in a beamstar scenario might encounter, again depending on the number of beams and hops (note that the y-axis starts with 2 beams, as one beam is now reserved for the FTP session). In this experiment we assumed that users sitting at the beam heads would fetch a 36K-web page from the center node every 15 sec on average.

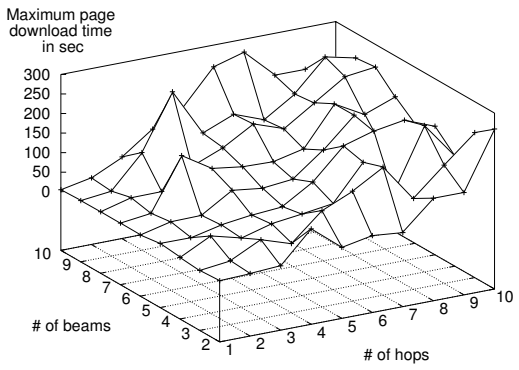


Figure 2.10: Download times for HTTP sessions competing with a single FTP session in the beamstar scenarios with a 2% packet drop rate.

Again, the terrain after 2 to 3 hops becomes rather bumpy and the potential fetch times jump from one second to 50 seconds and more.

Another concern regarding the synthetic beamstar scenario is that the derived results are due to the specifics of the scenario. Adding mobility and more fluctuation could potentially prevent that one stream captures all capacity. In other words, the hypothesis is that because of the more volatile environment, everybody will be suffering, but that this would hurt the big (FTP) data transfer more severely than the small (HTTP) requests. To test the validity of this argument, we performed the same simulation experiment as before but added random packet losses on every node to accommodate for the fluctuations due to mobility or changing reflections. Again, we use the beamstar scenario but add a packet drop probability of 2%, which results in reducing the throughput of a single FTP session over 3 hops to roughly half of the previous value. Unfortunately, although FTP packets are now more spaced, the HTTP sessions also suffer and the general picture is even worse than in the previous experiment (see Figure 2.10).

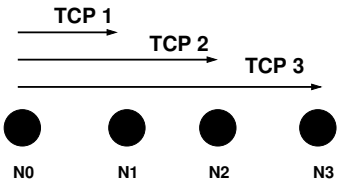


Figure 2.11: Simple test scenario for experiments and simulations.

2.2.3 Optimizing the configurable parameters for better TCP performance

So far we illustrated the problems of running traditional network services in MANETs using standard settings of TCP and IEEE 802.11 MAC protocols. The reported in the literature investigations on the reasons for the severe TCP unfairness in MANETs suggested a number of optimizations to the configurable TCP and MAC parameters aiming at improvement of the situation.

In this subsection we study the effect of optimizing TCP and IEEE 802.11 MAC parameters at the edge of the ad horizon on a simple topology with three hops, four nodes and three competing TCP flows as depicted in Figure 2.11. The three competing flows follow the paths of one, two and three hops respectively. The goal of this set of experiments is to see the dynamics of two major TCP performance parameters, namely throughput and unfairness.

In the experiments we will measure the combined (total) TCP throughput achieved by all three TCP flows, which we denote as Thr_{tot}. In order to evaluate the unfairness between TCP flows, we now use another form of this metric, which better reflects the individual dissatisfaction of end-users in the case of flows of different lengths. We define the unfairness index u_2 as the normalized distance (2.2) of the actual throughput of each flow from the corresponding optimal value.

$$u_2 = \frac{\sqrt{\sum_{i=1..n}(Thr_{opt_i} - Thr_i)^2}}{\sqrt{\sum Thr_{opt_i}^2}}. \tag{2.2}$$

	Simulations		Real-world test	
RTS/CTS	ON	OFF	ON	OFF
Thr_{tot}, (kb/s)	431	525	500	614
$Unfairness$	0.28	0.31	0.34	0.33

Table 2.2: The effect of RTS/CTS handshake on TCP performance.

In this formula Thr_{opt_i} is the ideal throughput of flow i obtained under fair share of the network capacity. In order to compute this value we divide the throughput of the corresponding flow obtained when running alone in the network by the number of competing flows. Thr_i is the actual throughput of the same flow achieved while competing with other flows. This index (2.2) takes the values between 0 and 1 and reflects the degree of global user dissatisfaction, hence the value of 0 corresponds to perfectly fair communications and 1 represents the opposite case. This time we also perform a series of experiments in a real-world testbed with the same configuration.

A. Disabling RTS/CTS exchange

We first evaluate the effect of RTS/CTS handshake on our two TCP performance metrics. In [38] it is shown that on all topologies disabling the RTS/CTS exchange improves TCP throughput. The results of the first experiment are presented in Table 2.2. We observe that disabling RTS/CTS exchange improves the total throughput by almost 100 kb/s. Although the unfairness is slightly larger in the second case, fully disabling RTS/CTS exchange is beneficial for TCP communications in MANETs.

The difference between the actual measured values for the corresponding metrics in the simulations and the real-world experiments can be attributed to the usage of simplified MAC and radio transmission models of ns-2. Since the evaluation of the accuracy of the used simulation models is not the goal of this dissertation we do not discuss these issues further. Note, however, that the dynamics of the two performance metrics is identical both in simulations and the real-world test.

(a) One common queue

CWND	Simulations		Real-world test	
	Default	Custom	Default	Custom
Thr_{tot}, (kb/s)	525	496	614	611
$Unfairness$	0.31	0.47	0.33	0.4

(b) Separated queues

CWND	Simulations		Real-world test	
	Default	Custom	Default	Custom
Thr_{tot}, (kb/s)	794	554	630	659
$Unfairness$	0.75	0.31	0.33	0.27

Table 2.3: The effect of CWND and interface queue optimization on TCP performance (no RTS/CTS exchange).

B. Optimizing CWND

An important parameter which influences TCP performance is the value of the maximum congestion window ($CWND$). The analysis of the related work (see Section 2.3 for corresponding references and description of the method) shows that adaptation of the TCP congestion window according to the number of wireless hops traversed by a flow allows to improve the performance characteristics of TCP in MANETs. This optimization is also called "window clumping".

It was shown in [23, 25] that TCP achieves highest throughput in a multihop wireless network when the maximum $CWND$ is scaled at sources as $h/4$, where h is the number of hops traversed by the flow. Further studies of the effect of $CWND$ clumping in [38] refine this bound and show that when the window size scales as $3 \cdot h/2$ TCP achieves optimal throughput on general topologies. We pick the latest result and configure the $CWND$ of each flow as $CWND(TCP1) = 2MSS$, $CWND(TCP2) = 3MSS$, $CWND(TCP3) = 4MSS$, where MSS is TCP maximum segment size. We let the three flows run again and the results of this experiment (simulation and real-world test) are reflected in Table 2.3 (a).

At first glance the outcome of this optimization step is surprising. Instead

of increasing the performance, both the total throughput and the unfairness index became actually worse. However the analysis of individual throughput of each flow (not shown in the table) reveals that flows following longer paths have an advantage over shorter flows. The reason for this is the default structure of the interface queue at the source node which is a single FIFO queue for packets of all three flows. Indeed, the queue becomes dominated by packets from longer flows since their CWND is larger, hence resulting in worse figures for the fairness of shorter flows.

C. Splitting the IFQ at sources

A simple optimization to the previous problem is to assign the incoming packets of different flows separate queues and to serve them in a round robin manner.

We account for the last observation and perform our third experiment configuring the interface queue at the source node to have three logical queues, one for each TCP flow. In the test bed we implemented this queue splitting by using the traffic controller (*tc*) that is a part of Linux. We performed the experiments with the default and optimal values of *CWND*, now with optimized structure of the interface queue. The results are presented in Table 2.3 (b).

Consider fields "Custom" containing the measurements for the case with customized $CWND$ in the corresponding tables. As we expected when splitting the interface queue, the resulting total throughput in the network increased both in simulations and real-world experiments. In addition to this we observed that the unfairness index value remains on the same level as in the case of disabled RTS/CTS handshake and the default window size. The unfairness index in simulations with the default $CWND$ / common interface queue and with customized $CWND$ / the split queue is 0.31. The corresponding values in the real-world test are 0.33 (common queue and default $CWND$) and 0.27 (split queue and customized $CWND$). The last observation confirms our assumption about the positive effect of combining the optimal TCP parameters and splitting the interface queue.

2.2.4 Lessons from the search for mitigating factors

The hypothesis, which we have tested in the experiments presented in this section, is whether variation in simulation settings, the use of lighter traffic patterns in the network and optimization of the configurable parameters of TCP and IEEE 802.11 MAC protocols are able to produce less pessimistic results regarding the ad hoc horizon.

Varying the internode distance in a network at the edge of the ad hoc horizon we observed that at some distances the TCP no-progress ratio become even worse. At the same time at those distances where this metric is relatively better, its values are still unacceptable for the end-users. Playing with the internode distances we highlighted the key role of long ranging radio interferences in formation of the ad hoc horizon. The use of a different application, i.e. web browsing instead of FTP transfer, reveals the same unfairness problem and does not lead to an extension of the ad hoc horizon. This observation highlights that the TCP capture problem is technology specific and not the application specific.

The importance of considering an optimization of the configurable TCP and IEEE 802.11 MAC parameters comes from the large deployment base of the existing technologies. The important result from the optimization of the *in-built* TCP and MAC parameters is that it indeed leads to certain improvements of TCP performance characteristics. However, in general, even with optimized protocol configurations, the problem of TCP unfairness and hence the sub-optimal TCP throughput remains.

Overall, in this section we found that TCP in combination with multihop forwarding over IEEE 802.11 based networks is not capable of providing reasonable fair sharing of a network's capacity even under optimal network and protocol configurations.

2.3 Literature survey

In this section we present a literature survey on the topic of TCP performance and mechanisms for its improvement in wireless networks. From the very beginning of the wireless era poor TCP performances over wireless links were known to the research community. The problems which were initially discovered in the

scenarios where only the last leg on the path of a TCP flow was over a wireless link naturally persist and even aggravate in a multihop wireless network.

We begin with the review of the papers indicating poor TCP performance in wireless environments in Section 2.3.1. After that we overview the approaches targeting the improvement of the situation on different layers of the TCP/IP protocol stack. In Section 2.3.2 we overview approaches dealing with adjustment of the MAC and link layers for better TCP performance. In Section 2.3.3 we discuss adjustments to the "legacy" TCP which aim at adaptation of the protocol to the specifics of the wireless environment. In Section 2.3.4 we present the papers which evaluate the degree of TCP performance improvements introduced by different modifications on TCP and MAC layers. Finally, we review the papers on interactions between TCP and ad hoc routing protocols in Section 2.3.5.

2.3.1 Discovering the poor TCP performance in wireless networks

Already in 1996 Balakrishnan et al [6] studied single hop wireless networks and identified the major reasons for poor TCP performance: The erroneous behavior of TCP congestion control in the case of bit error driven packet losses and consequent invocations of retransmission timers. In their paper the authors present a comparative study of the effect of transmission error correction at link layer on TCP performance, different flavors of TCP's congestion control mechanism and wireless-specific TCP connection splitting approaches. Their major conclusion is that assisting TCP with retransmission of a packet lost due to bit errors at the link layer significantly improves TCP performance. The usage of split-connection approaches (e.g. I-TCP [5], M-TCP [12], WAP [88] etc), however, by itself does not eliminate the problem of long TCP stalls and does not increase TCP throughput. The authors also show a positive effect of delayed acknowledgment schemes, which we describe below.

Similar findings were reported in [21, 24]. ElAarag in [21] also formulates a set of requirements for newly developing schemes and transport protocols for wireless networks: (1) The reliable transport in wireless networks should avoid erroneous triggering of the congestion control as a reaction to bit error caused packet losses; (2) The protocol should resolve the repeated timeouts problem; (3) It should efficiently handle frequent and long disconnections of the mobile hosts;

(4) The modifications should keep the end-to-end semantics of TCP; and (5) It should provide compatibility with the existing software of the wireline hosts.

Poor TCP performance and serious unfairness between competing TCP flows in mobile ad hoc networks is reported by Xu and Saadawi in [69]. The authors identified long ranging radio interferences as the major reason for TCP instability and bad fairness. They pointed out the impossibility to deal with the exposed terminal problem which is in essence caused by interferences. The authors also identify inefficiencies of the unlimited increment of TCP congestion window. We present the papers further developing this finding in Section 2.3.3 below.

The work in [58] also identifies the severe unfairness problem of TCP communications in MANETs. The authors show poor scalability of ad hoc networks in terms of the number of simultaneously active TCP connections. The paper highlights the need for improved congestion control, new scheduling and traffic management solutions.

The discovered inability of TCP to differentiate between packet losses, which happen due to the network congestion and the wireless link induced packet losses due to bit errors, became a basis for a large number of studies and modifications targeting an improvement of the situation. Extensive surveys of the existing approaches in this area are given in [46, 50, 27]. In our literature survey we describe the most representative proposals, which deal with adjustments of the MAC and link layers in order to reduce the impact of the wireless transmission medium on TCP performance and adaptation of TCP to the wireless environment.

2.3.2 Improving the performance of IEEE 802.11 MAC and link layer

The work in [65] suggests a distributed algorithm for exchanging the status of local transmission queues at the link layer and scheduling information in the one hop neighborhood of every wireless node. For this purpose the authors suggest modifications to the distributed coordination function of IEEE 802.11 MAC to implement broadcasting of such information. Further on this information is used to compute relative weights for all packets waiting for transmission in all stations in a manner that mimics the behavior of the weighted fair queuing scheduling in the wireline Internet.

The authors in [4] describe an approach of probabilistic hopping between

available radio channels of the IEEE 802.11 devices. The major idea is to announce the transmission schedule over different channels to all nodes in the neighborhood of one hop. It is shown that with such scheme the effect of interferences is minimized and the total throughput of the network is increased.

A cross-layer approach to improve TCP performance in wireless environment is described in [8]. The authors suggest to enrich the link layer with the functionality of retransmitting TCP data segments which were lost due to bit errors and to introduce additional delay to the congestion control of TCP when reacting on missing acknowledgments. The authors propose the Delayed Congestion Response TCP (TCP-DCR), which offers an improvement to TCP performance in single hop wireless networks. The application of TCP-DCR to multihop wireless networks is not considered in the paper.

A technique suggested for enhancing TCP fairness in multihop ad hoc networks is distributed neighborhood RED [68]. The idea behind *nRED* is to coordinate the dropping or marking of messages in wireless nodes in a distributed way by overhearing the transmission in the neighborhood of two hops from each node and by estimating the size of a virtual queue of the whole neighborhood. The dropping and marking process, as it is clear from the name of the approach, is adopted from the RED algorithm in the wireline Internet [83].

2.3.3 Adaptation of TCP to wireless environment

The work on adaptation of the TCP protocol to the specifics of the wireless and in particular multihop wireless environment is going along the two major threads: Packet loss discrimination and reducing the traffic load from a TCP session. The later type of approaches can be sub-classified as follows: (1) Optimization of TCP's congestion window (*CWND*) limit, also referred to as "window clumping"; (2) Adaptive acknowledgment strategies; and (3) Adaptive rate limitation in wireless nodes.

A. Error detection approaches

In [45] the authors suggest an enhancement to TCP's error recovery scheme. The Eifel algorithm that they suggest deals with the problem of spurious TCP timeouts.

The problem may arise in wireless networks with potentially large transmission delays: While all the data segments from a particular transmission window may successfully reach the destination, the acknowledgments may arrive to the source with a delay, which is longer than the current value of the TCP retransmission timer. Arriving to the source after the retransmitted packet was sent, the ACKs for the previously transmitted packets are wrongly interpreted as a signal of the loss of the retransmitted packet. This might result in retransmission of the entire window. This problem leads to reduction of TCP throughput and unnecessary invocation of the Go-Back-N procedure. The core of the Eifel algorithm is the introduction of a time stamp to the TCP data segments. The stamp for the corresponding packet received at the destination is inserted in the acknowledgment frame for this data segment. By keeping the history of the transmission times of the data segments from the particular window, the sender now can differentiate between ACKs for the original transmission and ACKs for the retransmitted packets. The algorithm can boost the end-to-end throughput by several tens of percent and eliminates the need for additional proxies between the end points of the connection.

Oliveira and Braun in [51] suggest the use of fuzzy logic theory to discriminate the packet losses due to network congestion and bit errors. Their approach relies on the observation of round trip times and thus does not require involvement of the intermediate nodes.

A combination of an error detection mechanism and an assured ACK delivery strategy is proposed in [72]. The author suggests a early packet notification scheme (ELPN), which allows the intermediate nodes to signal the events of packet losses on the path towards the corresponding source. The second suggested mechanism is best effort ACK delivery. With this mechanism the lost acknowledgments are retransmitted by either intermediate nodes or the TCP receiver. Altogether these two mechanisms lead to certain improvement of the end-to-end TCP throughput.

B. CWND clumping

The observations regarding the effect of the congestion window size on TCP performance were first reported by Fu et al. in [23]. The authors show that the *CWND* value for which TCP achieves the best throughput should scale as $h/4$ for a TCP

flow traversing h wireless hops. The rationale behind this finding is simple: The shared medium of multihop wireless links does not allow simultaneous sending of packets from several nodes in the same radio interference range. An unlimited *CWND* results in a growing backlog of the TCP data segments in the intermediate nodes on the path of the particular TCP connection. In its turn, the continuous sending of this backlog later on results in the increase of the forwarding delays of the packets, which now are backlogged some hops ahead and may lead to losses of acknowledgments on their way back to the source. The direct implication of this phenomenon is the degradation of the end-to-end TCP throughput. Kawadia in [38] refine this bound and show that when the window size scales as $3 \cdot h/2$, TCP achieves optimal throughput on general network topologies.

In [67] the authors describe the TCP capture problem, investigate the reasons for TCP unfairness in MANETs and illustrate the positive effect of *CWND* clumping on TCP performance in MANETs. Our observations on the reason for TCP capture confirm the findings of this paper: It is the effect of radio interferences which cause the loss of data segments. The problems caused by the radio interferences is impossible to detect at TCP level and is very hard to signal to the source.

C. Adaptive acknowledgments strategies

Amongst the particular problems, in [26] the author shows that the congestion control of TCP in the fast re-transmit phase is very sensitive for the additional delays induced by the wireless links. The paper highlights the need for smart acknowledgment strategies for the improvement of TCP performance.

In [1] the authors propose to limit the acknowledgment flow on wireless links to improve TCP throughput. In their scheme the rate of ACKs in the backward direction depends on the value of sequence numbers of the data segments arriving to the destination. The proposed scheme leads to around 50% improvement of TCP throughput.

Oliveira and Braun in [52] suggest another delayed acknowledgment technique based on the observation of the arrival rate of the data segments. Using this strategy, the receiver adjusts itself to the wireless channel condition by delaying more ACK packets when the channel is in good condition and less ACKs otherwise.

D. Adaptation of TCP transmission rate

A cross-layer approach based on the network layer feedback called ATCP is proposed in [44]. The information that the network feedback mechanism reports to the sources of TCP connections is (1) Failure of the route; (2) The event of a packet loss during the transmission on MAC layer; and (3) The true network congestion. In all these cases TCP is instructed to adapt its behavior as following. On the loss of the connectivity the TCP sender goes into a persist state, so that it does not unnecessarily (re-)transmit packets. As a reaction to bit error related packet loss, the ATCP layer retransmits the lost packet before invocation of the congestion control mechanism by TCP. Finally, the congestion control performs the traditional actions in the case of a TCP network congestion.

An architectural proposal called INSIGNIA for improving the quality of TCP communications in MANETs is suggested in [41]. The authors describe and implement a set of mechanisms at source nodes which adapt the transmission rate according to the current estimates of the network load on the path of the particular connection. The authors propose the use of an in-band signaling for the delivery of the network state information to TCP senders. Every INSIGNIA-enabled wireless node locally performs measurements of the current traffic load. When relaying an IP packet this value is written into the IP header. This is processed by the destination node and the information about the load in every hop on the path is signaled back to the source node, which in turn reduces the transmission rate to fit the estimated available capacity.

In [22] the authors introduce a new congestion control algorithm for TCP over multihop IEEE 802.11 based networks. The core of the suggested algorithm is an adaptation of the transmission rate of TCP at the sending node using current estimation of the end-to-end delay and the coefficient of variation of recently measured round trip times. In the evaluation of the proposed scheme the authors account both for throughput and fairness characteristics of TCP connections and show significant improvements of these characteristics in comparison to the related approaches.

Yang et al in [71] propose to limit the arrival rate from the interface queue to the transmission buffer on MAC layer depending on the observations of the departure rate in the past. Their proposal assumes that the rate limitation at a trans-

mitting node is implemented at the link layer, thus leaving the functionality of the MAC protocol unchanged. The authors compute the transmission rate limits using past observations of the packet emission rate from the node and apply heuristics for the rate control. The control mechanism is implemented in forwarding nodes as well as in sources of communications.

2.3.4 The significance of the modifications on different layers on TCP performance

An important insight on the overall impact of modifications on different layers of the TCP/IP protocol stack on TCP performance in MANETs is presented in [38, 39]. The authors perform an extensive set of experiments in the real-world testbed. By thorough statistical analysis the authors reveal the factors which place the most significant impact on TCP performance in ad hoc networks. According to their studies these factors are the window clumping and disabling the RTS/CTS handshake. We illustrated these findings also in this chapter (see Section 2.2.3).

2.3.5 Interactions between TCP and ad hoc routing

Surprisingly enough we found very few papers which investigate the interactions between TCP and ad hoc routing. The initial study of the effect of routing on the quality of TCP communications is presented by Dyer and Boppana in [20]. Their study concentrates on TCP throughput achievable using different ad hoc routing schemes. Doing such measurements the authors mainly evaluate the quality of the particular routing scheme with respect to the path recovery times. The authors do not explicitly consider particular TCP over MANET problems.

The work of Perkins and Hughes [58] presents an evaluation of the effect of DSR and AODV on TCP performance. The authors conclude that when using the source routing protocol – DSR, TCP achieves higher throughput than in the case of AODV.

In [49] Nahm et al describe problems of interoperation between TCP and on-demand ad hoc routing protocol. They observe that TCP causes overreaction of the routing protocol, which degrades the quality of the end-to-end connection. The authors show the cases where TCP, occupying the available bandwidth, pre-

vents propagation of the control routing messages. This leads to large re-connect intervals, which in turn reduce the TCP throughput. The authors propose to use the known mechanisms for reduction of TCP traffic load including an adaptive window increment schemes as described above.

The work in [60] from Uppsala University studies the interactions between TCP, UDP and routing protocols in MANETs. The authors show that the broadcast traffic generated by dynamic routing adds instability to ad hoc networking. In particular hidden terminals and channel capture effects cause instability of routes due to losses of control routing messages caused by the competing data traffic. This results in a very high packet loss rate for UDP traffic and long timeouts of TCP.

2.4 Summary and motivation

In this chapter we exposed the severe performance problem when using the existing TCP/IP protocol stacks over MANETs. With TCP capture, only few connections obtain access to the network's transmission capacity, severely degrading the performance of other flows.

2.4.1 Summary of our findings

While the problems highlighted in this chapter are known to the MANET research community, we presented them in the context of scalability studies of mobile ad hoc networks. The present study is intended to eliminate the too optimistic gap between scalability studies based on simulations and experiences from the real-world test beds. Our simulation based approach and the worst case interpretation strategy allowed us to quantify the scaling limits of the current TCP and IEEE 802.11 MAC protocols. Although making optimistic assumptions on the environment and configuration for a wireless ad hoc routing protocol, we found rather discouraging worst case figures for the combination of plain TCP, ad hoc routing and IEEE 802.11. They confirm experimental insights that operating multihop forwarding is only recommended in a very limited range. This "ad hoc horizon", as we call it, is in the low 2 to 3 hops range and extends to a dozen nodes.

In this chapter we also analyzed the reasons for the severe unfairness between competing TCP flows to appear. We found that poor TCP performance in multihop wireless networks is due to the interaction of IEEE 802.11 MAC and TCP congestion control mechanism. In this interoperation the complex structure of the radio transmission ranges plays the key role. We illustrated how TCP's congestion control "wrongly" interprets the losses of data segments due to interferences as network congestion, which for the most "unlucky" TCP flow leads to consequent expiration of the TCP re-transmission timer and the "collapse" of the congestion window. In the mean time the winning flows enjoy the spare network bandwidth by increasing their transmission rate, by this aggravating the situation for the affected flows. We highlighted the problem that in this case the congestion control of TCP (perfected and fine-tuned in TCP for the wireline Internet) is helpless in resolving the capture problem which happen on the border of communication zones for the affected nodes. Moreover the specifics of the radio interference range are such that the affected nodes themselves cannot signal the problem back to the sources. Altogether these factors allow the winning flows to ignore the presence of affected competitors.

2.4.2 Motivation

In order to expose the internal interaction problems of MANETs, we also conducted extensive studies of the factors for mitigating the poor TCP performance in MANETs. We illustrated that even under optimal network and protocol configurations unfairness between TCP flows exists. These observations clearly indicate the need for further full-scale improvements of MANET functionalities. In this respect the concept of the ad hoc horizon, defined in this chapter, offers a challenging benchmark. This became the major motivating factor for conducting the work described further in this dissertation.

Analyzing the literature we observe that the work directed at solving the TCP capture problem is being conducted considering interactions of particular MANET protocol sets, i.e. *TCP + IEEE 802.11 MAC*, *TCP + routing*, and little or no attempts to consider all protocols involved in the communication process in MANETs as a whole. In our research we want to explore the overall behavior of the *TCP + routing + IEEE 802.11 MAC* system.

Relating to the huge deployment base of the devices compliant with the existing TCP/IP and IEEE 802.11 technologies we foresee considerable interoperation and compatibility problems of the approaches targeting modification of the "legacy" products. Making our goal to explore the practical feasibility of MANETs to provide end users with communications of an acceptable quality now and with the *existing* resources, we spur the adoption of MANET technology in the society. This, in our opinion, should serve as a kick off for the appearance of new MANET specific applications and potentially to new and better transmission technologies and communication standards.

Chapter 3

Fair TCP throttling for 802.11 based MANETs

In the previous chapter we identified the interoperation problems between TCP and multihop transmissions over IEEE 802.11 MAC. On one hand it is the existing TCP congestion control mechanism which was optimized for the wireline networks and does not work properly in wireless environment. On the other hand it is the complex structure of the radio communication regions. In particular there is a region around a node where the transmitted data packets cannot be correctly received by other nodes, however the signal level is high enough to destruct the reception of other packets for the nodes located in this region. Generally speaking the two reasons are mutual. The problem is that the loss of packets due to bit-errors is a misleading information for the congestion control mechanism of TCP. This is well understood by the research community. Apparently the true reason for packet losses in wireless networks is not visible at the TCP layer. As we showed in the literature survey some approaches targeting the improvement of TCP performance attempt to discriminate the reasons for packet losses, others deal with modifications of TCP congestion control in order to make it more "wireless friendly".

In the core of our solution is a different approach on the analysis of TCP communications in MANETs. We do not attempt to improve the TCP congestion

control. We neither propose a scheme to discriminate the packet losses. Rather we create a smart session-oriented engine very close to the interface queue. This thin layer is aware about the consequences of the uncontrolled TCP transmissions over multihop wireless networks and adjusts the behavior of the outgoing traffic to maximize the benefits both for the locally originated session and all competing TCP flows in the network.

Overall the functionality of such engine is simple. In the simplest case the input information to this engine is the number of competing connections on the path of the particular multihop TCP connection (*path density*) and the knowledge of the saturation point of the radio medium, beyond which a geographical region with multiple multihop wireless connections becomes congested (i.e. *boundary load of the bottleneck region*). Based on this information, the engine shapes the outgoing traffic so that it does not disturb other flows. If all sources of the competing TCP connections cooperate and execute the same shaping algorithm then the total network capacity is fairly divided between the flows. In this chapter we outline the general principles of our solution. We present our approach to compute the transmission rate limits for the ingress nodes and identify the place in the protocol stack of a wireless node where the computed rate limit will be enforced. We also outline an extension to ad hoc routing protocols which allows to obtain the necessary information for the computation of the limit from the network at run time.

3.1 Problem decomposition

The major difficulties in the analysis of TCP performance in MANETs is the variety of the interacting protocols which at the end provide the end-to-end data delivery. Those are IEEE 802.11 MAC, an ad hoc routing protocol, and TCP itself. The referred difficulties come from the specifics of the IEEE 802.11 transmission medium. As we already discussed in Section 1.3.2, in contrast to the wireline Internet, several multihop links of MANETs can belong to the same radio collision domain. Therefore, if in the case of the Internet we can consider interactions on a certain layer of the TCP/IP stack abstracting from the operations of other layers, in MANETs it is essential to consider the cross-layer interactions. However, we

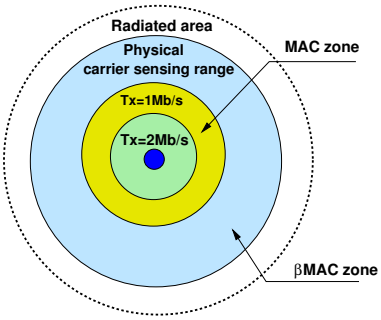


Figure 3.1: Communication ranges of a *base* IEEE 802.11 device.

see the task of simultaneous optimization of all involved protocols as extremely complex due to large variety of independent parameters which are needed to be taken into account. We approached the problem of TCP unfairness in MANETs in the following structured way. In general it can be characterized as an incremental increase of complexity of the considered problems.

3.1.1 Communication and interference ranges

In this dissertation we focus on the behavior of an entire end-to-end multihop TCP flow. When talking about a stream of packets between application layers at a source and destination we will use terms *flow* and *connection* interchangeably. Further on in the text we will refer to a set of nodes including the source, destination and nodes that forward packets of the TCP flow as a *source-destination association* or the flow's *path*.

Conventionally when talking about transmission ranges of the IEEE 802.11 enabled devices, a single transmitting node is considered. For the single node the radio transmission ranges are defined as schematically shown in Figure 3.1. We denote the area of the assured data reception up to the slowest transmission rate (1 Mb/s) as the *MAC zone* of a node. We call the area beyond the MAC zone but within the physical carrier sensing range the *βMAC zone* (beyond MAC). We also extend the definitions of the transmission ranges for the entire end-to-end

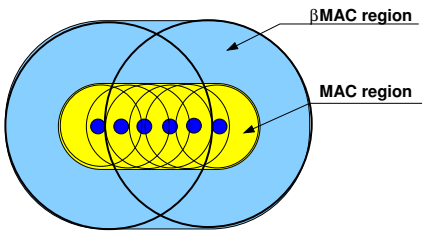


Figure 3.2: Communication ranges for the association (Only βMAC zones of the end nodes of the association are shown. Internode distance is 126 m).

multihop *association* as illustrated in Figure 3.2 and define MAC region, βMAC region as follows.

The **MAC region** is the space around the association in which other stations not included in the association are able to receive data packets issued by nodes of the association with the basic data rate of 1 Mb/s.

The **βMAC region** is the space around the association beyond the MAC region of the association but within the βMAC zone of every node included in the association. Nodes of other associations located in this area can not receive data packets from this association but reception of own packets is affected by interferences produced by nodes of this association.

The above defined communication regions imply two different ways of nodes placement for several associations with respect to each other, hence two distinct cases of a network formation:

1. The nodes of several associations partly or completely share the MAC regions of each other;

2. The associations are located solely in βMAC regions of each other.

In this dissertation we in details cover the first case, as in our opinion it corresponds to a usual operation of MANETs. The case where the competing associ-

ations are located outside the communication regions of each other we address in our Technical Report [53].

3.1.2 Considering static networks

Observing a spectrum of the existing problems with TCP in mobile ad hoc networks we may conclude that to a large extend it is not clear if current TCP would perform well even over relatively static multihop wireless ad hoc networks. We foresee that good performance of TCP over static networks is a mandatory prerequisite for good performance over networks with mobility. In this dissertation we consider relatively static networks with little or no mobility. Our goal is to explore a possibility of stable TCP communications over the combination of IEEE 802.11 MAC and routing protocols. We consider the MANETs' kinematic as a standing alone research area. We assume that there exist other solutions dealing with mobility-induced problems of TCP. In the previous chapter we listed some representatives of this kind in the literature survey.

3.1.3 Exploring TCP + IEEE 802.11 MAC tandem

In our approach we firstly study pure interactions of TCP and IEEE 802.11 MAC assuming static routing in the network. This allows us to illuminate a subset of problems caused by interoperations of the two protocols. For this subset we were able to find a solution, which allows stable fair communications of multiple TCP flows in multihop MANETs. This part of the work is outlined in Section 3.2 and fully developed in Chapters 4 and 5.

3.1.4 Enabling routing activity

Once obtained the stable operation of TCP + IEEE 802.11 MAC tandem we add another component of communications in MANET, the ad hoc routing protocol. In this part we analyze the effect of different routing traffic patterns on TCP performance during stable operations of MANETs. The results obtained on the first research step facilitated the process of the qualitative and quantitative assessment

of the IEEE 802.11 MAC + routing + TCP interactions. We outline our analysis methodology in Section 3.3 and present the detailed discussion in Chapter 6.

It is important to note that we develop each subset of the main TCP unfairness problem in the scope of the "ad hoc horizon" defined in the previous chapter. This allows us to qualitatively judge on the level of improvements or degradation of the network performance introduced by every considered method. By doing this we also understand the scaling limitations of the ad hoc mode of the IEEE 802.11 technology for multihop communications in terms of the number of simultaneously active data sessions and the network size.

3.1.5 Adding the reality: Implementation considerations

Finally, when we find that all major components of MANET communications can nicely interoperate, we develop the practical issues, which allow the implementation of the found solutions in real-world networks. The development of such an extension to ad hoc routing protocols allows practical implementation of our ingress throttling scheme designed on the step of TCP + IEEE 802.11 MAC analysis. With this part completed we round off the technical development of the general topic of the dissertation on practical feasibility of fair TCP communications in multihop MANETs. The outline of this work is given in Section 3.4 and its full description in Chapter 7.

3.2 An adaptive ingress throttling approach

Our investigation aims at understanding the behavior of multiple multihop TCP connections over IEEE 802.11 MAC in the case where routes are statically set before the communications begin and remain unchanged during the communication process. On the first stage of our research we formally introduce a fairness framework for MANETs. For this we adapted the fairness model from the wireline Internet to the specifics of multihop wireless networks. The major outcome of this stage is new definitions of bottleneck regions and the boundary load within them. We propose an algorithm of load distribution for the connections competing inside the bottleneck region.

On the second stage we derive an ingress rate limit which ensures that the sum of the loads produced by all TCP flows inside the bottleneck region does not exceed the boundary load. In its simplest form the rate limit is a function of:

- The number of hops for a particular TCP connection;
- The underlying physical layer transmission rates along a path of the particular connection;
- The number of competing connections on the path of the considered connection (path density).

These parameters are feasible to obtain using the facilities of ad hoc routing protocols as described in Chapter 7. We apply the derived rate limit to configure a scheduler at the interface queue of sources of TCP sessions and shape the outgoing traffic accordingly. As a result none of the TCP sessions is able to benefit from temporal weaknesses of the competitors by capturing the transmission capacity.

3.2.1 Outcome

The major result from this stage is that we were able to achieve a stable interoperation of TCP and IEEE 802.11 MAC protocols and dramatically improve the fairness properties of the end-to-end TCP sessions. Consequently we significantly extend the ad hoc horizon. Now we talk about tens of simultaneously active TCP sessions over up to 10 hops paths.

3.3 The impact of the routing traffic on the improved TCP performance

This part of our research concerns the evaluation of the influence of ad hoc routing protocols on the improved combination TCP + IEEE 802.11 MAC. We develop this thread as follows.

On the first stage we observe major properties of the routing traffic. We evaluate the effect of different routing traffic patterns on TCP communications assuming that the sources cannot adjust their behavior to routing activities in a controlled manner.

After that, on the second stage we extract the "fingerprints" of the effect of different traffic patterns produced by several ad hoc routing protocols on the quality of ongoing TCP sessions. Finally, using the optimized throughput property of our ingress throttling scheme we indirectly estimate the load produced by the particular routing protocol.

3.3.1 Outcome

Adding routing protocol activities to the improved TCP + IEEE 802.11 MAC system reveals that the routing traffic itself can be a reason for TCP unfairness in MANETs, we call this phenomenon the *routing capture* or simply *RT-capture*.

During the analysis of the effect of different routing schemes on the TCP performance we identified an operational scale of MANETs in terms of the number of simultaneously active TCP sessions and the number of network nodes. We call the limit of the admissible operation range the *"RT-horizon"* of MANETs. Within this limit the interactions between the routing and data traffics introduce minimal effect on the fairness and the progress quality of the end-to-end data sessions.

Our major conclusion about the effect of the routing traffic patterns on TCP communications is that periodic, non error-driven broadcast of even short control messages is harmful for data communications and leads to narrowing the operational region of MANETs.

3.4 A path density protocol for MANETs

Now when we identified the operational region of MANETs where all components of data communications can co-exist without degrading the quality of the end-to-end user communications, we address the issues of practical feasibility of the developed ingress throttling approach. We consider a special, but nevertheless practically important class of MANETs where all connections traverse a common

bottleneck geographical region. In order to make the ingress throttling scheme realistically implementable, the sources of TCP flows need to obtain the information about the number of competing connection along their routes at run time. We call this parameter of our model the *path density*. With this information the source nodes estimate the available share of network capacity and compute the rate limit for the locally generated TCP connections. Shaping the traffic according to the computed limit, almost perfect fairness is achieved and the total network throughput is increased.

We developed a distributed scheme for gathering the path density information: A *state-full* route reply driven path density protocol (PDP). The main idea of the statefull path density protocol is that on reception of a *route reply* message all involved nodes establish a state corresponding to the end-to-end session. A node delays the announcement of a connection to the neighborhood until it sees the first data packet from this connection, as only this event indicates that the connection is active. The announcement of the active connections is done every time a node sends or re-broadcasts a route request message.

3.4.1 Outcome

The specification of the protocol for gathering the path density information in MANETs allows us to conclude that the realization of the ingress throttling scheme is feasible in reality. Apart from the technical novelty of the developed protocol, we also suggested an efficient technique for the development and testing new distributed algorithms in MANETs.

In the core of our approach is the usage of a single code base both for a network simulator and real-world operating systems. This allowed us to extensively test the functionality and debug the problems of the scheme on the variety of network topologies. Once obtained a stable operation of the protocol we were able to instantly recompile and obtain the real-world version. We stress that this reality-oriented approach is important for the development of any distributed scheme for MANETs.

Chapter 4

Space-load fairness framework for MANETs

In this chapter we present our first contribution. We analyze the interactions among multiple multihop TCP connections running over the IEEE 802.11 based ad hoc networks. We consider the case of static routing and no mobility, otherwise we do not place additional assumptions. The competing TCP flows may use all available transmission rates of IEEE 802.11b at the physical layer, traverse different numbers of hops, and use different values of the maximum segment size. We approach the problem of TCP unfairness firstly by considering the fairness framework from the wireline Internet. We reflect the existing fairness model to the case of the IEEE 802.11 based MANETs. Secondly, we suggest a set of mechanisms by means of which the competing TCP flows conform to the specified fairness framework for MANETs.

We proceed in this chapter as follows. In Section 4.1 we present a general definition of the fairness framework for wireline Internet. Section 4.2 is the main section where where we formulate the *space-load* fairness framework for MANETs and present a set of mechanisms for its enforcement in real networks. We summarize the material of this chapter in Section 4.3.

4.1 Fairness framework for the wireline Internet

The concept of *fairness* in the wireline Internet nowadays is a stable and complete theory [61, 7, 10]. The core of the fairness theory includes network models, the practically implementable mechanisms for achieving the fairness and the formal machinery for the analysis of the fairness properties of these mechanisms. In this section we present the network model and definitions of fairness based on the extensive summary of the fairness framework in [9][1].

Overall the goal of the fairness framework for packet switched networks is to make sources to limit their sending rate by taking into consideration the state of the network in order to avoid a congestion collapse. The congestion collapse is a severe decrease in total network throughput when the offered load from the competing sessions increases.

A fairness can be in one of two classes: *max-min* fairness and *proportional*. The class of *proportional* fairness has various subclasses which we do not cover in this dissertation.

4.1.1 Network model

Consider a set of sources $s = 1, ..., S$ and links $l = 1, ..., L$. Let $A_{l,s}$ be the fraction of traffic of source s which traverses link l and let c_l be the capacity of link l. Thus a network is defined as the couple $(\vec{x}, A)$. A feasible allocation of rates $x_s \geq 0$ is defined by: $\sum_{s=1}^{S} A_{l,s} x_s \leq c_l$ for all l.

4.1.2 Definition of the bottleneck link

Based on the network model defined above, link l is said to be a bottleneck for source s if and only if

1. Link l is saturated: $c_l = \sum_i A_{l,i} x_i$
2. Source s on link l has the maximum rate among all sources using link l: $x_s \geq x_{s'}$ for all s' such that $A_{l,s'} > 0$.

[1] In the following subsections we adopt the notations and definitions from [9].

4.1.3 Definition of max-min fairness

A feasible allocation of rates $\vec{x}$ is "max-min fair" if and only if an increase of any rate within the domain of feasible allocations must be at the cost of a decrease of some already smaller rate. Theorem 1.2.1. in [9] states that a feasible allocation of rates $\vec{x}$ is max-min fair if and only if every source has a bottleneck link.

A. The algorithm of progressive filling

There is an algorithm which allows max-min fair allocation based on the results of the above theorem; it is called "progressive filling". In essence the algorithm works as follows. Having an initial transmission rate equals zero every source starts to increase it with the same *growth rate* among all sources. The process continues until one or more links become saturated. In this case the sources that use these links stop increasing their transmission rates. Other sources continue to increase the rate until new saturated links are found. In this fashion each source finds its bottleneck link on which its rate is equal to or larger than the rate for other sources. Due to the finite set of links and sources the "progressive filling" algorithm terminates.

4.1.4 Reflecting the network model for wireline networks to wireless MANETs

Before proceeding with the definition of a framework for fairness in MANETs let us first analyze the input components to the network model described above and relate these components to the case of MANETs.

A. From wireline "link" to wireless "L-region"

Apparently the major stumbling block in reflecting the above network model to the case of MANETs is a notion of the *link* and the associated terms *capacity* and *rates* of sources. In the case of wireline networks a *link* is a physical *wire* or *fiber*, which has a limited capacity c measured in bits per second. Sources $s = 1, ..., S$ generating the traffic at rate r_s, measured in bits per second, share link l if $\sum_i r_i \leq c_l$.

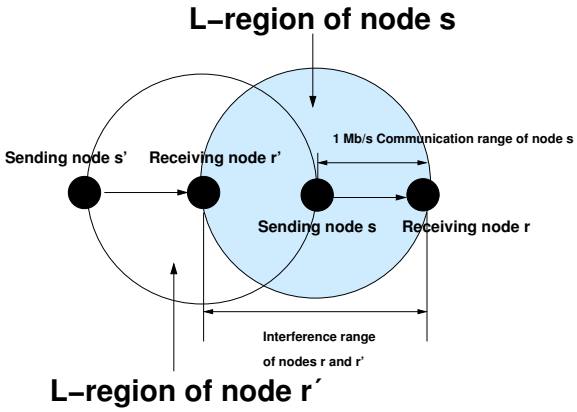


Figure 4.1: Illustration of the "L-region" concept.

In general for the IEEE 802.11 based networks the term "link" is misleading. Indeed, it is incorrect to consider an imaginary line between two communicating nodes as the link. As we indicated in Chapter 1 the radio signal from a given packet transmission is propagated in a geographical region of a certain size. We illustrated as well that inside this region there can be other stations transmitting their packets to in general different destination nodes. Therefore for our MANET fairness framework we need to redefine the term "link". We define the term *"L-region"* (link region) as follows.

Definition 4.1 (L-region): The *L-region* is an area equal to the size of the 1 Mb/s MAC zone of an IEEE 802.11 radio transmitter, which has the following properties:

1. The center of L-region is located in a node transmitting or receiving data of at least one end-to-end data flow;

2. Inside the L-region there exist nodes transmitting or receiving data of other end-to-end data flows different from those carried by the central node.

The above definition of the L-region is illustrated in Figure 4.1. With the

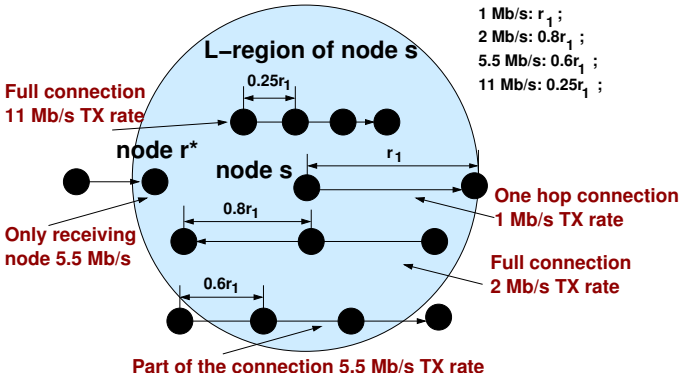


Figure 4.2: What should be considered as capacity of L-region and what should be assigned to a flow fully inside this region?

network settings depicted in the figure there are two L-regions around nodes s and r'. Note that the regions of the same size with the centers in nodes r and s' are not L-regions since they contain only nodes that carry traffic of one data flow.

B. From wireline "sources" to wireless "associations"

Defining the L-region as a geographical region with the properties of the wireline link we need to reconsider the concept of the data "source". On a wireline link a part of capacity is consumed by packet transmissions from a single entity – the session's "source". The consumed capacity is expressed in terms of the "rate".

Let us consider Figure 4.2 where we zoomed in a single L-region. Defined as an area equals the size of 1 Mb/s MAC zone of an IEEE 802.11 transmitting node, the scale of the L-region is large. If we assume the radius of 1 Mb/s MAC zone $r_1 = 250\ m$ (using ns-2 radio transmission model and the default settings) the area of the L-region is $S_{L-region} = \pi r_1^2 = 196250\ m^2$. Using the estimates of the radio transmission ranges from real-world measurements (see Table 1.1) where $r_1 = 130\ m$: $S_{L-region} = 53066\ m^2$. Despite the different actual values for the area of the L-region, the important point is that this region is very sparse. As it is

illustrated in the figure inside an L-region it is possible to place different numbers of nodes transmitting packets of different end-to-end data flows depending on the used transmission rate at the physical layer. In the upper right corner of the figure we listed the relative internode distances for different transmission rates.

The important point of the above observations is that in order to consider the sharing of the resources (which resource to share we cover in the following subsection) inside the L-region we need to consider not individual "sources" that consume a part of the L-region's resources but a set of nodes which happen to be inside the L-region.

Definition 4.2 (An association): We refer a set of nodes including the source, the destination and the nodes that forward packets of a TCP flow as to a *source-destination association*.

Now that we defined an analogy to the "source" we can state the condition of belongingness of an association to the L-region:

Belongingness of an association to the L-region condition: Node n of association a belongs to L-region λ if and only if it is able to communicate with the central node of the L-region with the base IEEE 802.11 transmission rate of 1 Mb/s.

The above stated belongingness condition implies the following rule of defining new L-regions:

Defining new L-regions: If node n of association a belongs to L-region λ with the center in node $k \neq n$, it is also a center of another L-region λ' intersecting with L-region λ. Inside L-region λ' there might exist associations such that $\{a'_i\} \in \lambda'$ and $\{a'_i\} \notin \lambda$.

The belongingness condition also implies a potential possibility of a two hops communication between all associations included inside the L-region. As we shall see later this is the essential condition for the practical feasibility of fair communications in MANETs.

C. From wireline "rate" and "capacity" to wireless "load share" and "boundary load"

Consequently our definition of the L-region as a substitution for the wireline link makes the terms "rate" of sources and "capacity" of the link in their conventional sense in the wireline domain meaningless in MANETs.

The notion of "rate" in the wireline domain relates to the notion of "capacity" of the link. On a given link the rate of traffic from the particular source is a fraction of the link capacity $x_s \leq c_l$. Thus the term "rate" makes sense only when the term "capacity" is well defined and its value is finite. In the case of L-region the later term is impossible to identify uniquely in conventional bits per second. Having multiple available transmission rates of 802.11 devices at the physical layer (e.g. 1 Mb/s, 2 Mb/s, 5.5 Mb/s, 11 Mb/s for IEEE 802.11b) it would be incorrect to use one of these rates for the definition of the capacity of the L-region. This is because in general nodes inside the L-region may use any of the available physical layer transmission rates.

What we can talk about is a *load* which an association or a part of the association *generates* or *consumes* inside the L-region. We refer this term as to *conserved* load or simply *C-load*. We comment on the duality of the load definition later and now proceed with the definition of the total C-load and the boundary C-load.

Definition 4.3 (Total C-load inside L-region): Total C-load of an L-region is the load generated or consumed by nodes of distinct associations included in the L-region.

In the light of the above observations on the non-determinism of defining the capacity and the rate inside the L-region we also have a non-determinism in expressing load in conventional bits per seconds. Therefore we define *C-load* of an L-region as a unit-less measure.

Definition 4.4 (Boundary C-load inside L-region): There is a *boundary C-load* inside the L-region beyond which the region becomes congested. The *boundary C-load* is also a unit-less measure. We normalize this parameter to one.

Note that such dual definition of the *conserved* load (i.e generated and consumed) is essential for the L-region. There can be a case where only the receiving node of an association is located inside the particular L-region. This is for example the case with node r^* in Figure 4.2. Assume this node does not generate any traffic. Obviously, if the L-region is congested the reception of data packet is impossible for r^* due to the hidden terminal effect of IEEE 802.11 MAC. Therefore there should be a spare part of the L-region's resource in order to allow r^* a possibility to correctly receive packets. Now with the resource to share inside the L-region defined as C-load, we define *C-load share* to be the analog of the wireline "rate".

Definition 4.5 (C-load share): *C-load share* is a fraction of the boundary C-load that the particular connection generates or consumes inside the L-region among other connections ongoing in the L-region. We denote this parameter ϕ.

4.1.5 Summary of the wireline fairness framework

In this section we reviewed the fairness framework in the wireline Internet. Specifically, we focused on fairness of sharing the capacity of the links in the case of multihop communications. We described a network model which is used to define objectives for *max-min* fairness.

Attempting to reflect the wireline network model to the case of MANETs we understood that the major concepts such as the source, the link, the rate of sources and the capacity of the links are not suitable for wireless MANETs. This is due to specifics of the IEEE 802.11 transmission medium. We presented new entities called the *association*, the *L-region*, the *C-load share* and the *boundary C-load*, which serve as substitutes for the corresponding terms in multihop wireline networks.

The major consequence of the newly defined terms is that for the consideration of fairness in MANETs we shift the focus from the *link-capacity* domain, specific to the wireline networks, to the MANET specific *space-load domain*. In the following section we describe our space-load fairness framework over L-regions.

4.2 Fairness framework for MANETs

In this section we define our framework of fairness in mobile ad hoc networks. We start by considering the fairness of resource sharing on the level of *L-regions*, which correspond to the fairness of links' capacity sharing in the wireline networks.

4.2.1 Space-load fairness over L-regions

In order to present the *space-load* fairness framework of MANETs we present our space-load network model of MANETs, define fairness objectives and present the mechanisms for achieving fair communications between competing associations.

A. MANET network model in the space-load domain

Let us now present the network model of MANETs in the defined *space-load* domain. Recall that now we have L-regions instead of links. As explained above instead of data rates allocation ($\vec{x}$) in the space-load domain we use $\vec{\phi}$ as possible allocations of *C-load shares* for the *associations* competing inside the L-region.

Space-load network model: We consider a set of associations $a = 1, ..., A$ and the set of existing L-regions $\lambda = 1, ..., \Lambda$. Let $\Gamma_{\lambda,a}$ be the indicator of the presence of association a inside L-region λ:

$$\Gamma_{\lambda,a} = \begin{cases} 1, & a \in \lambda \\ 0, & otherwise. \end{cases}$$

Thus the network is defined as a couple ($\vec{\phi}$, Γ). A feasible allocation of C-load shares $\phi_a > 0$ is defined by $\sum_{a=1}^{A} \Gamma_{\lambda,a}\phi_a \leq 1$ for all L-regions λ.

B. Definition 4.6 (Bottleneck L-region)

With the space-load MANET model defined above, we define a bottleneck L-region for association a if and only if

1. L-region λ is saturated: $\sum_i \Gamma_{\lambda,i}\phi_i = 1$

2. Association a in L-region λ has the maximum C-load share among all associations located in L-region λ: $\phi_a \geq \phi_{a'}$ for all a' such that $\Gamma_{\lambda,a'} = 1$.

C. Max-min fairness in space-load domain

A feasible allocation of C-load shares $\vec{\phi}$ is "max-min fair" if and only if an increase of any C-load share within the domain of feasible allocations must be at the cost of a decrease of some already smaller share.

A feasible allocation of C-load shares $\vec{\phi}$ is max-min fair if and only if every association belongs to a bottleneck L-region. The proof of this statement resembles the proof of the original Theorem 1.2.1 in [9].

4.2.2 The algorithm of C-load shares distribution

For the complete picture of the fairness framework in MANETs we need to describe an algorithm for *max-min fair* distribution of C-load shares between the associations inside L-regions and suggest a mechanism by means of which the associations conform to the assigned fair shares. In this subsection we address the first issue and leave the description of the second problem to Section 4.2.3.

In comparison to the simple "progressive filling" algorithm, which is steered by the sources of communications the proposed algorithm is a distributed assignment scheme which is executed by all nodes of all associations. Below we present a formal description of the algorithm. In order to simplify the description we assume the following:

1. During the execution of the algorithm the network and the set of associations are stable. This means that associations neither leave the initial L-region nor appear in the new L-region;

2. Initially all associations are not assigned the C-load shares. Further on we refer an association without the assigned load share as to a *fresh association* and an association with the assigned load share as to an *assigned association*;

3. All nodes in the network execute the same algorithm and cooperate;

4. The information is distributed between the MANET nodes by means of a message passing scheme. However for a general description of the algorithm we do not suggest any particular protocol and assume that all necessary information is accessible at a centralized control point.

Denote the number of associations inside L-region λ (*L-region density*) as $\rho_\lambda = \rho_\lambda^{fresh} + \rho_\lambda^{assigned}$, where ρ_λ^{fresh} and $\rho_\lambda^{assigned}$ are correspondingly the numbers of fresh and assigned associations inside L-region λ. The algorithm of max-min fair assignment of C-load shares is as follows:

1. All central nodes of L-regions suggest a C-load share to the visible fresh associations according to the following formula:

   (a) L-regions with only fresh associations: $\phi = \frac{1}{\rho_\lambda}$;

   (b) L-regions with assigned and fresh flows: $\phi = \frac{1-\sum_{i=1}^{\rho_\lambda^{assigned}} \phi_i}{\rho_\lambda^{fresh}}$;

2. Among all L-regions choose those L-regions which suggest the minimal C-load share. Assign the computed share to the associations which are included in these regions. Do not modify the shares of these associations after that.

3. Repeat steps 1 and 2 until all flows are assigned the C-load share (all L-regions contain only assigned associations).

The presented above algorithm terminates because the set of associations and consequently the set of L-regions are finite. Now we need to prove that the algorithm assigns the load shares to all fresh associations max-min fairly. In other words that at every iteration of the algorithm every fresh flow is suggested a bottleneck share in all L-regions which it traverses.

Property 4.1: In any L-region the newly assigned associations at iteration k have

shares less than or equal to the shares suggested by *the same* L-region for the assignment at iteration $k-1$:

$$\phi_{\lambda}^{new(k)} \leq \phi_{\lambda}^{suggested(k-1)}$$

This property is straightforward due to the choice of the minimum from the suggested shares in Step 2 of the load share distribution algorithm. Denote $N(k)$ the number of new assigned associations within an arbitrary L-region at iteration k of the algorithm. Then Property 4.1 also implies

$$\sum_{i=1}^{N(k)} \phi_{\lambda,i}^{new(k)} \leq N(k) \cdot \phi_{\lambda}^{suggested(k-1)} \tag{4.1}$$

Proposition 4.1: In all L-regions the shares suggested to the fresh associations at any iteration of the algorithm are equal to or larger than the shares of other associations inside the corresponding L-regions assigned at the previous iterations.

PROOF. We show that the proposition holds for an arbitrary chosen L-region at all iterations while it has fresh (not assigned) associations. In this region according to Property 4.1 the newly assigned associations have C-load shares less than or equal to that was suggested by this L-region at the previous iteration. In order to prove the proposition it is sufficient to show that the suggested to the fresh associations shares is a non-decreasing sequence. That is for L-region λ:

$$\phi_{\lambda}^{suggested(0)} \leq \phi_{\lambda}^{suggested(1)} \leq ... \leq \phi_{\lambda}^{suggested(k)} \leq \phi_{\lambda}^{suggested(k+1)} \leq ...$$

$$... \leq \phi_{\lambda}^{suggested(final)}, \tag{4.2}$$

where $\phi_{\lambda}^{suggested(i)}$ is the C-load share suggested by L-region λ at iteration i and $\phi_{\lambda}^{suggested(final)}$ is the suggestion for the last fresh association in λ after which the association becomes assigned either in this or in another L-region.

We proceed by induction. Let iteration 0 of the algorithm corresponds to its starting time where the suggested shares are initialized to 0 ($\phi_\lambda^{suggested(0)} = 0$). Hence all L-regions including our λ have only fresh flows. The suggested by L-region λ shares at the end of iteration 1 is $\phi_\lambda^{suggested(1)} = 1/\rho_\lambda$. Note that if any L-region has the same number of fresh and assigned associations during several iterations, the suggestion of the C-load shares from this L-region to the fresh flows is the same. Further on we assume that inside λ one or more new assigned associations appear at every iteration $2, 3, ..., final$ of the algorithm.

Obviously the condition (4.2) holds for iterations 0 and 1. Assume that the condition is true for iteration k with respect to iteration $k-1$ and show that the condition holds for iteration $k+1$.

At all iterations $k > 1$ the suggested shares are computed using formula in Step 1b of the algorithm:

$$\phi_\lambda^{suggested(k)} = \frac{1 - \sum_{i=1}^{\rho^{assigned(k)}} \phi_i}{\rho^{fresh(k)}}$$

As for iteration $k+1$, the suggested shares are

$$\phi_\lambda^{suggested(k+1)} = \frac{1 - \sum_{i=1}^{\rho^{assigned(k+1)}} \phi_i}{\rho^{fresh(k+1)}},$$

where $\rho^{assigned(k)}$, $\rho^{assigned(k+1)}$, $\rho^{fresh(k)}$ and $\rho^{fresh(k+1)}$ are the number of respectively assigned and fresh associations at the *beginning* of iteration k and $k+1$ correspondingly.

Assuming that the assigned shares are ordered relative to their appearance in the L-region at every iteration, note that

$$\sum_{i=1}^{\rho^{assigned(k+1)}} \phi_i = \sum_{i=1}^{\rho^{assigned(k)}} \phi_i + \sum_{\rho^{assigned(k)}+1}^{\rho^{assigned(k+1)}} \phi_i. \tag{4.3}$$

That is the sum of all assigned shares at the beginning of iteration $k+1$ is the existed sum of assigned shares at the beginning of iteration k plus the sum of the

shares assigned to the fresh associations at the end of iteration k. The number of members in the last summation $N(k) = \rho^{assigned(k+1)} - \rho^{assigned(k)} - 1$ is the number of new assigned associations in λ appeared at the beginning of iteration $k+1$. Note also that at iteration $k+1$ the set of existing fresh associations differs from the previously existed set of fresh associations on $N(k)$ associations. That is

$$\rho^{fresh(k+1)} = \rho^{fresh(k)} - N(k). \tag{4.4}$$

Now we need to show that $\phi_\lambda^{suggested(k+1)} - \phi_\lambda^{suggested(k)} \geq 0$:

$$\frac{1 - \sum_{i=1}^{\rho^{assigned(k+1)}} \phi_i}{\rho^{fresh(k+1)}} - \frac{1 - \sum_{i=1}^{\rho^{assigned(k)}} \phi_i}{\rho^{fresh(k)}} \Rightarrow$$

$$\frac{1 - \sum_{i=1}^{\rho^{assigned(k+1)}} \phi_i}{\rho^{fresh(k)} - N(k)} - \frac{1 - \sum_{i=1}^{\rho^{assigned(k)}} \phi_i}{\rho^{fresh(k)}} \Rightarrow$$

$$\frac{N(k)(1 - \sum_{i=1}^{\rho^{assigned(k)}} \phi_i) - \rho^{fresh(k)}(\sum_{i=1}^{\rho^{assigned(k+1)}} \phi_i - \sum_{i=1}^{\rho^{assigned(k)}} \phi_i)}{\rho^{fresh(k)} \cdot (\rho^{fresh(k)} - N(k))}.$$

Now the denominator of the above expression is always positive due to (4.4), hence we need to determine the sign of the numerator. There using (4.3) we have:

$$N(k)(1 - \sum_{i=1}^{\rho^{assigned(k)}} \phi_i) -$$

$$-\rho^{fresh(k)}(\sum_{i=1}^{\rho^{assigned(k)}} \phi_i + \sum_{\rho^{assigned(k)}+1}^{\rho^{assigned(k+1)}} \phi_i - \sum_{i=1}^{\rho^{assigned(k)}} \phi_i)$$

Using (4.1):

$$\sum_{\rho^{assigned(k)}+1}^{\rho^{assigned(k+1)}} \phi_i \leq N(k)\phi^{suggested(k)} = N(k)\frac{1-\sum_{i=1}^{\rho^{assigned(k)}}\phi_i}{\rho^{fresh(k)}}$$

That is

$$N(k)(1-\sum_{i=1}^{\rho^{assigned(k)}}\phi_i) - \rho^{fresh(k)}\sum_{\rho^{assigned(k)}+1}^{\rho^{assigned(k+1)}}\phi_i \geq$$

$$\geq N(k)(1-\sum_{i=1}^{\rho^{assigned(k)}}\phi_i) - \rho^{fresh(k)} \cdot N(k)\frac{1-\sum_{i=1}^{\rho^{assigned(k)}}\phi_i}{\rho^{fresh(k)}} = 0.$$

Which proves that $\phi_\lambda^{suggested(k)} \leq \phi_\lambda^{suggested(k+1)}$. Since the suggested to the fresh associations shares is a non-decreasing sequence with number of iterations, every L-region with both fresh and assigned associations always tries to be a bottleneck for the fresh associations. Hence, the algorithm assigns the C-load shares *max-min* fairly.

□

Let us illustrate the operation of the C-load share distribution algorithm by an example. Figure 4.3 shows a sample network topology with nine end-to-end flows, hence nine associations. For simplicity of the presentation we will consider only suggestions of the load shares submitted to the centralized control point by the nodes marked by a darker color and numbered $(n1, ..., n11)$ in the figure. The relative distances between the associations which are shown in the figure imply that the marked nodes can hear the transmission from associations as indicated in Table 4.1.

At the start time of the algorithm (iteration 0) all associations in the network are fresh with C-load shares initialized to zero. This is marked as $f(0)$ in the figure. Later on when the associations become assigned we mark this fact as $a(\cdot)$ where in the parentheses we indicate the assigned share.

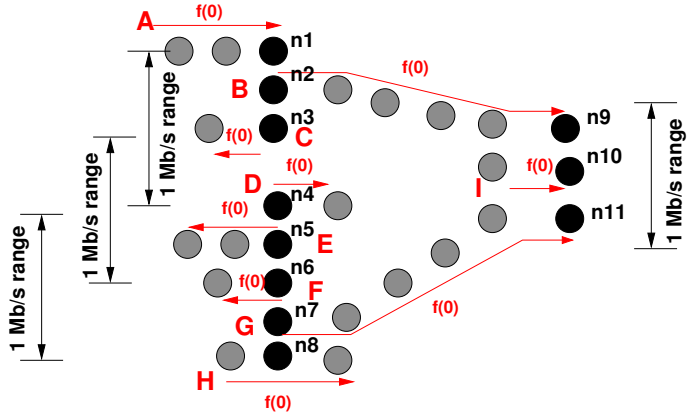


Figure 4.3: Example for illustration of the C-load share distribution algorithm in operation.

At the end of iteration 1 the shares suggested by the marked nodes to the initially fresh associations are shown in Table 4.2. The algorithm in Step 2 assigns shares $1/5$ to associations D, E, F, G, H. Thus, at the end of the first iteration one bottleneck with respect to D, E, F, G, H is determined and the algorithm proceed further with iteration two.

At the end of iteration 2 the shares suggested by the marked nodes to the fresh associations are shown in Table 4.3. The algorithm in Step 2 assigns shares

$n1$	$n2$	$n3$	$n4$	$n5$
$\{A, B, C\}$	$\{A, B, C\}$	$\{A, B, C, D\}$	$\{C, D, E, F\}$	$\{D, E, F, G\}$
$n6$	$n7$	$n8$	$n9$	$n10$
$\{D, E, F, G, H\}$	$\{E, F, G, H\}$	$\{F, G, H\}$	$\{B, G, I\}$	$\{B, G, I\}$
$n11$				
$\{B, G, I\}$				

Table 4.1: Association sensitivity ranges.

	A	B	C	D	E	F	G	H	I
$n1$	f(1/3)	f(1/3)	f(1/3)	-	-	-	-	-	-
$n2$	f(1/3)	f(1/3)	f(1/3)	-	-	-	-	-	-
$n3$	f(1/4)	f(1/4)	f(1/4)	f(1/4)	-	-	-	-	-
$n4$	-	-	f(1/4)	f(1/4)	f(1/4)	f(1/4)	-	-	-
$n5$	-	-	-	f(1/4)	f(1/4)	f(1/4)	f(1/4)	-	-
$n6$	-	-	-	f(1/5)	f(1/5)	f(1/5)	f(1/5)	f(1/5)	-
$n7$	-	-	-	-	f(1/4)	f(1/4)	f(1/4)	f(1/4)	-
$n8$	-	-	-	-	-	f(1/3)	f(1/3)	f(1/3)	-
$n9$	-	f(1/3)	-	-	-	-	f(1/3)	-	f(1/3)
$n10$	-	f(1/3)	-	-	-	-	f(1/3)	-	f(1/3)
$n11$	-	f(1/3)	-	-	-	-	f(1/3)	-	f(1/3)

Table 4.2: Suggested shares to the fresh associations at the end of iteration 1.

$4/15$ to associations A, B, C. Thus, at the end of the second iteration the second bottleneck with respect to associations A, B, C is determined and the algorithm proceed further with iteration three.

	A	B	C	D	E	F	G	H	I
$n1$	f(5/15)	f(5/15)	f(5/15)	-	-	-	-	-	-
$n2$	f(5/15)	f(5/15)	f(5/15)	-	-	-	-	-	-
$n3$	f(4/15)	f(4/15)	f(4/15)	a(3/15)	-	-	-	-	-
$n4$	-	-	f(6/15)	a(3/15)	a(3/15)	a(3/15)	-	-	-
$n5$	-	-	-	a(3/15)	a(3/15)	a(3/15)	a(3/15)	-	-
$n6$	-	-	-	a(3/15)	a(3/15)	a(3/15)	a(3/15)	a(3/15)	-
$n7$	-	-	-	-	a(3/15)	a(3/15)	a(3/15)	a(3/15)	-
$n8$	-	-	-	-	-	a(3/15)	a(3/15)	a(3/15)	-
$n9$	-	f(6/15)	-	-	-	-	a(3/15)	-	f(6/15)
$n10$	-	f(6/15)	-	-	-	-	a(3/15)	-	f(6/15)
$n11$	-	f(6/15)	-	-	-	-	a(3/15)	-	f(6/15)

Table 4.3: Suggested shares to the fresh associations at the end of iteration 2.

At the end of iteration 3 the share suggested by the marked nodes to the only left fresh association I is $8/15$ as it is indicated in Table 4.4. The algorithm terminates as there are no fresh associations left. Now all associations are assigned their bottleneck shares. Figure 4.4 shows the final assignment of C-load shares after the algorithm has terminated. We marked the bottleneck L-regions as 1,2 and 3 in the order of their detection by the algorithm.

4.2.3 Enforcement of fair C-load shares in MANETs

After having described the fairness framework for MANETs and suggested an algorithm for *max-min* fair assignment of C-load shares to the competing associ-

	A	B	C	D	E	F	G	H	I
$n1$	a(4/15)	a(4/15)	a(4/15)	-	-	-	-	-	-
$n2$	a(4/15)	a(4/15)	a(4/15)	-	-	-	-	-	-
$n3$	a(4/15)	a(4/15)	a(4/15)	a(3/15)	-	-	-	-	-
$n4$	-	-	a(4/15)	a(3/15)	a(3/15)	a(3/15)	-	-	-
$n5$	-	-	-	a(3/15)	a(3/15)	a(3/15)	a(3/15)	-	-
$n6$	-	-	-	a(3/15)	a(3/15)	a(3/15)	a(3/15)	a(3/15)	-
$n7$	-	-	-	-	a(3/15)	a(3/15)	a(3/15)	a(3/15)	-
$n8$	-	-	-	-	-	a(3/15)	a(3/15)	a(3/15)	-
$n9$	-	a(4/15)	-	-	-	-	a(3/15)	-	f(8/15)
$n10$	-	a(4/15)	-	-	-	-	a(3/15)	-	f(8/15)
$n11$	-	a(4/15)	-	-	-	-	a(3/15)	-	f(8/15)

Table 4.4: Suggested shares to the fresh associations at the end of iteration 3.

ations, we need to interpret the model parameters in the terms meaningful to the network nodes. In this subsection we present the mapping of the space-load model parameters into the rate-capacity domain.

A. TCP throughput as a reference to the boundary C-load

The interpretation of the boundary C-load by sources of TCP connections in terms of the transmission rate is somewhat straight forward. We need to find a condition under which every node of an association tends to generate maximal load inside a geographical region. If for a moment we consider the wireline Internet this condition has a direct analogy in terms of the *bandwidth-delay product* – the amount of traffic that the entire path can accommodate[2]. For the estimation of the bandwidth-delay product the major property of TCP protocol is used: A single TCP flow in a steady state is a perfect estimator of the available bandwidth in the network. We will use this property for the estimation of the boundary C-load.

Indeed running along over a multihop MANET a single TCP flow will generate maximal load. In the steady state every node of the particular association has a continuous backlog of packets. If we consider an arbitrary multihop association and potential L-regions with the centers located in the nodes of this association we can always identify the L-region where the TCP connection will constantly be active. Figure 4.5 shows a constant "air presence" of a single TCP session inside

[2]In the Internet the bandwidth-delay product is used to dimension the $CWND$ parameter at sources to prevent local congestion [61].

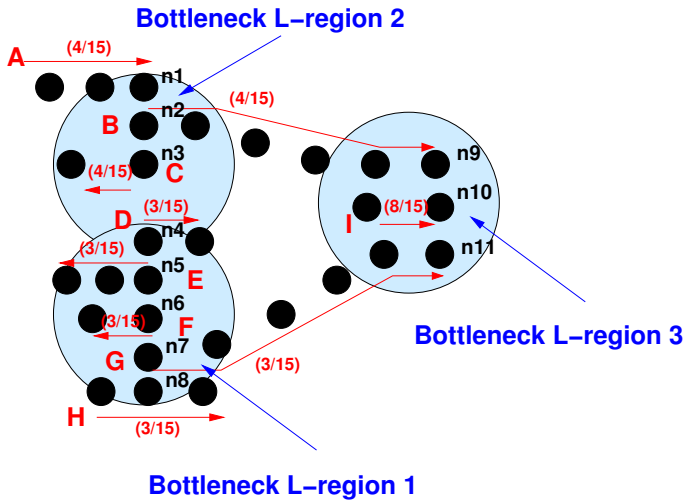


Figure 4.4: Example of C-load share distribution by the algorithm.

the L-region. In the figure we have a single three hops TCP session from node N1 to node N4. In the steady state the amount of data packets backlog at node N1 will be constant because of continuous arrival of acknowledgments. As it is visible from the figure in the potential L-region with the center in node N2 our TCP session constantly produces the load from nodes N1, N2 and N3. Therefore the knowledge of the maximal throughput achievable by the flow can serve as a good reference to the boundary C-load of the L-region. In Chapter 5 we present an experimental assessment of the above observations.

It is obvious that the load generated by the association inside the L-region can be regulated at the source node which in the first place generates the traffic along this association. Namely, delaying the transmission of the subsequent data packet for a certain amount of time at the source we would reduce the overall load produced inside the L-region. We discuss this issue below.

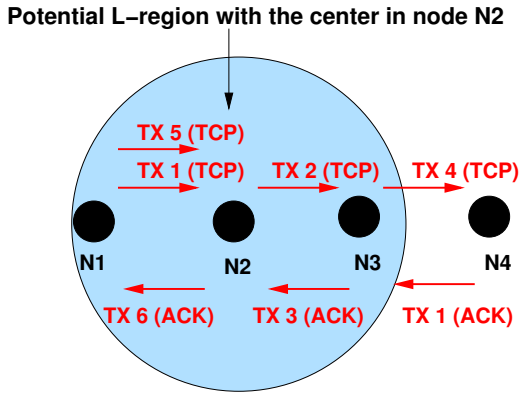


Figure 4.5: Constant presence of a single flow inside an L-region.

4.2.4 The ingress throttling formula

Taking the maximally achievable throughput by session i as a reference to the boundary C-load in order to conform traffic of this data session to the assigned fair share of the C-load ($\phi_{bottleneck}$) the output rate from the sources of TCP connection i should be reduced according to the following formula:

$$r_i^{ingress} \leq Thr_{max}(h, MSS, TX_{802.11}) \cdot \phi_i^{bottleneck}. \tag{4.5}$$

In the above formula $Thr_{max}(h, MSS, TX_{802.11})$ is the maximal throughput of a TCP flow that traverses h hops, transmits data segments of size MSS and $TX_{802.11}$ is the used transmission rate at the physical layer along the path of the flow. This parameter can be experimentally or formally estimated for all possible combinations of the input parameters. The parameter $\phi_i^{bottleneck}$ is the fair C-load share of TCP session i in its bottleneck assigned by the C-load distribution algorithm described above.

A. Treating UDP traffic inside the space-load fairness framework

In order to conform UDP traffic to the space-load fairness framework the ingress throttling formula (4.5) should be adjusted to account for the one-way nature of UDP communications. What we should do is to increase the derived rate limit by the fraction corresponding to the transmission of TCP acknowledgments. For a given estimate of $Thr_{max}(h, MSS, TX_{802.11})$ the ingress rate limit is then computed as

$$r_i^{ingress(UDP)} \leq \frac{MSS + ACK}{MSS} Thr_{max}(h, MSS, TX_{802.11}) \cdot \phi_i^{bottleneck}. \quad (4.6)$$

In the above formula ACK is the size of a TCP acknowledgment in bits. Now when we know the actual transmission rate limit for the competing TCP/UDP connections we need to enforce it at the corresponding ingress nodes. In the next subsection we define the place where this functionality will be implemented.

4.2.5 Implementation considerations

A. Rate throttling enforcement at the ingress nodes

Schematically the architecture of a wireless node is presented in Figure 4.6. For simplicity of presentation we assume that one source originates only one end-to-end connection. We also assume that the interface queue is either logically or physically divided into three queues: One for data packets originated at this node, the second one for all other data packets and the third for all packets of the routing protocol. Note that the last two queues are also the default structure of the interface queue in ns-2. All three queues are drop-tail in nature. Upon arrival from the routing layer all packets are classified according to their source IP addresses and type and placed into the corresponding queue.

The scheduler from the interface queue to the MAC layer consists of two stages. At the first stage we have a fixed delay non-work-conserving scheduler with tunable delay parameter Δ. When $\Delta = 0$ the first stage scheduler works as usual work-conserving scheduler and the whole scheduling system works as a scheduler with three priorities.

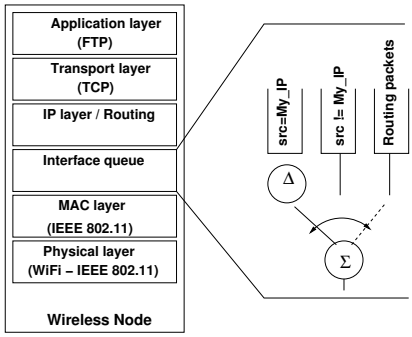


Figure 4.6: Structure of a IEEE 802.11 enabled (ns-2) node.

Scheduler Σ at the second stage is a non-preemptive work-conserving scheduler with highest priority to the queue with routing packets, then to the queue with locally generated packets and finally to the forwarding queue.

The transmission rate limit (4.5) is used to set parameter Δ of the scheduler for the queue with locally generated packets. We compute the delay parameter as

$$\Delta = \frac{MSS_i}{r_i^{ingress}(h, MSS, TX_{802.11})},$$

where MSS_i is the size of the maximum data segment size used by TCP session i.

B. Space-load architecture for MANETs

Assume that $Thr_{max}(h, MSS, TX_{802.11})$ – the estimates for the maximal achievable throughput are hardwired at source nodes and available on-demand. Now at the particular ingress node in order to conform the packet transmission to the *space-load* fairness framework we need (1) to choose the correct value of the maximal throughput according to the characteristics of the particular end-to-end connection and (2) to obtain the fair C-load share for this connection in the network.

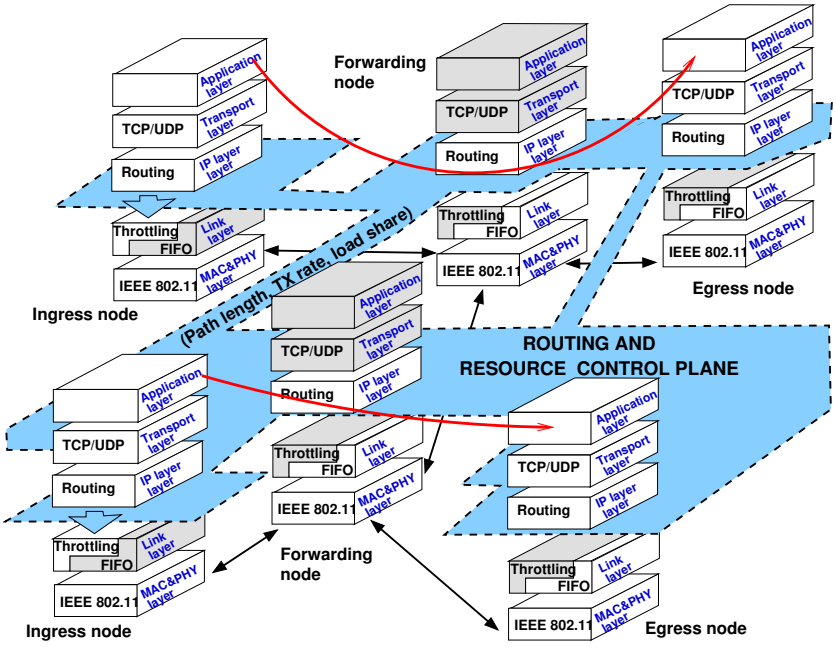


Figure 4.7: Architectural view.

Figure 4.7 offers an architectural view on the enforcement of the fairness framework in MANETs. The white colored boxes in the figure show the layers of the TCP/IP protocol stack actively participating in the communication process of the two existing end-to-end data sessions.

Note that out of the three input parameters for $Thr_{max}(h, MSS, TX_{802.11})$ the maximum segment size (MSS) can be detected locally for every incoming to the scheduler packet. The available IEEE 802.11 transmission rates on the path ($TX_{802.11}$), the value for the path length (h) as well as the assigned for the particular connection fair C-load share ($\phi^{bottleneck}$) should be provided by the network. In order to obtain these parameters we suggest to enrich the routing layer with the functionality of controlling the resource distribution in MANETs. The rationale behind this our decision is straightforward.

In the case that a reactive ad-hoc routing protocol is used to provide connectivity in multihop wireless network the path length parameter is easily obtainable from the route reply (RREP) message received by the source. The other two parameters (the physical layer transmission rate and the fair C-load share) are not directly obtainable from the currently existing routing protocols. However, the end-to-end nature of the reactive routing schemes and a broad propagation range of the control routing messages make the routing layer perfect for disseminating and gathering the network and connection state information. In Chapter 7 we discuss the feasibility of obtaining such information using LUNAR [64], which is a L2.5 reactive routing solution. LUNAR is well suited for this task as it establishes independent paths between peer nodes (while as DSR and AODV are aggregating routing information among routing paths). We will not discuss routing protocol modifications further in this chapter.

4.2.6 Special class of MANETs – the common bottleneck L-region

In this dissertation among others we will focus on a special but nevertheless practically important class of MANETs. We will consider a class of networks where all competing data flows belong to at least one common bottleneck L-region. This means that all flows traverse a common geographical region of the area equals to the 1 Mb/s MAC zone. In reality this might correspond to a scenario of a large open space hot-spot with the wireless gateway to the wireline Internet. In this case all flows tend to follow the paths towards this hot-spot. Another scenario is a set of parallel connections in a geographical corridor with non radio-transparent walls. This is a valid model of the multihop communications along a street in the city where all connections follow a path to the Internet gateways.

With the assumption of a common bottleneck L-region the algorithm of C-load shares distribution becomes a task of finding the maximum density of L-regions ρ_{max} along the paths of the competing connections. We will refer this entity as to the *"path density"* throughout the text. In this case the fair load shares $\phi_i^{bottleneck}$ for each connection can be computed at the sources as $1/\rho_{max}$. In Chapter 7 we present a protocol for gathering the path density for every connection in the network by this showing the practical feasibility of the suggested fairness framework.

4.3 Summary

In this chapter we presented the first contribution of the dissertation. We formally specified the fairness model for wireless multihop ad hoc networks. For this we adapted the existing fairness framework for the wireline Internet to the specifics of MANETs. Within the scope of the defined framework we suggested a methodology to derive adaptive limits on transmission rates at sources.

We suggested an algorithm for fair distribution of the load inside geographical regions (L-regions) where multiple multihop connections compete for the transmission capacity. Taking the ideal throughput of a single TCP session as a reference to the boundary load of the L-regions we computed the limit on ingress transmission rates, which ensures that the total load from multiple TCP connections inside the bottleneck L-region does not overflow the boundary load. Our solution does not require changes to the standard TCP nor IEEE 802.11 and is implementable by enhancing routing protocols and the use of traffic policing at the ingress nodes.

In the next chapter we present a full scale experimental evaluation of the suggested mechanisms and assess the validity of the proposed fairness framework and used assumptions.

Chapter 5

Validation of the space-load fairness framework and discussion

In the previous chapter we presented a max-min fairness framework for mobile multihop ad hoc networks. For this we adapted the corresponding framework from the wireline Internet to the specifics of the wireless networks. The resulting fairness model for MANETs is considered in the *space-load* domain natural for this type of networks. In this chapter we present the results of the experimental assessment of the space-load fairness framework, compare our model to the related approaches and discuss the implications of the proposed solution and future research directions.

We proceed in this chapter as follows. In Section 5.1 we present a set of experiments performed both in simulations and a real-world testbed. We demonstrate the validity of the fairness framework and used assumptions. The results from the experimental evaluation show dramatic improvements in fairness of TCP communications. In Section 5.2 we perform a detailed comparison of our fairness framework to the related fairness models. Finally in Section 5.3 we discuss the implications of our space-load fairness model and present open research issues.

5.1 Validation through simulations and real-world experiments

In the experiments presented below we apply the space-load fairness model to the IEEE 802.11b based networks. We test our theoretical findings on a wide range of topologies where we vary the length of the paths of the competing flows, the underlying transmission rate at the physical layer.

5.1.1 Experimental, simulation setups and used performance metrics

Here we describe the common settings for all simulations (with the network simulator ns-2.27 [82]) and the real-world experiments.

The real-world results were obtained from a setup of four DELL Latitude laptops with ZyXEL ZyAIR B-100 wireless interfaces. We use the Linux operating system with 2.6 kernel with embedded control on the interface queue by means of a traffic controller (*tc*).

In all setups we used TCP Newreno as the most popular variant of TCP. We performed real-world experiments with the maximum TCP data segment size (MSS) equals 600 B. In simulations we operated with a broader range of MSS sizes from 100 B to 1000 B. All results presented in this section are means over 30 runs of each experiment.

We use FTP file transfers in both real-world experiments and simulations. The routes for all flows are statically assigned prior to the data transmissions. As all TCP flows started we allow a warm-up period of 12 seconds to exclude initial traffic fluctuations from the measurements. The duration of all real-world experiments and simulations is 120 seconds.

In all experiments we assume that the information about the bottleneck C-load shares, the physical layer transmission rates on the path and the estimates of the ideal throughput for the flows with corresponding parameters is available at sources of TCP flows. In order to estimate $Thr_{max}(h, MSS, TX_{802.11})$ needed to compute the rate limit (4.5) and subsequently the delay parameter Δ of the scheduler at the interface queue, we run a flow with the corresponding parameters in isolation before the experiments and measure this value. Since the main purpose of the experiment is a functional assessment of our solution and not an

auto-configuration of the used parameters for our ingress throttling scheme, we configure the delay parameter of the scheduler in the interface queue prior to the start of the experiments. We address the feasibility of such auto-configuration in Chapter 7.

A. Performance metrics

In the experiments we assess the network performance using the following set of metrics:

1. Individual (per-flow) TCP throughput. Denote this metric as Thr_i where i is the index of the particular TCP connection;

2. Combined (total) TCP throughput of all existing in the network TCP flows. Denote this metric Thr_{tot};

3. Unfairness index u_2: It is the normalized distance (5.1) of the actual throughput of each flow from the corresponding optimal value.

$$u_2 = \frac{\sqrt{\sum_{i=1..n}(Thr_{opt_i} - Thr_i)^2}}{\sqrt{\sum Thr_{opt_i}^2}}. \tag{5.1}$$

   In this formula Thr_{opt_i} is the ideal throughput of flow i obtained under fair sharing of the network bandwidth. In order to compute this value we apply the assigned fair C-load share to the ideal throughput of the session running in isolation. Thr_i is the actual throughput of the same flow achieved while competing with other flows. This index takes the values between 0 and 1 and reflects the degree of global user dissatisfaction, hence the value of 0 corresponds to perfectly fair communications and 1 represents the opposite case. In addition the unfairness index u_2 also qualitatively characterizes the individual end-to-end throughput of each flow. The closer the index value to zero the better is each individual throughput.

4. The *unsmoothness* metric as it is defined in the next subsection for the qualitative assessment of the TCP session progress.

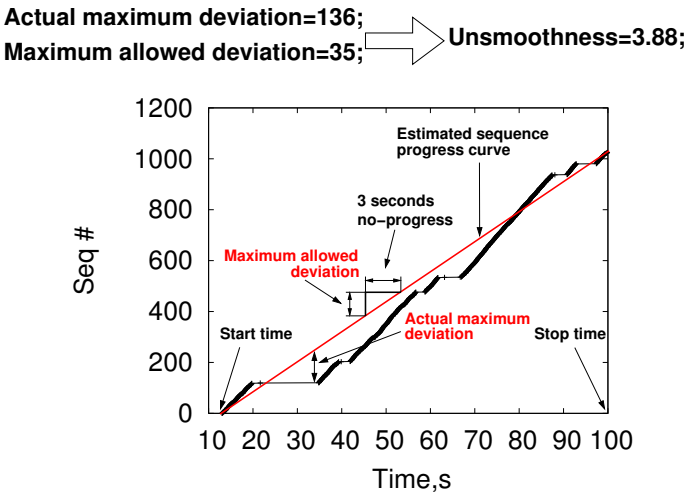


Figure 5.1: Computation of the unsmoothness metric

B. Definition of the "unsmoothness" metric

As we have shown in Chapter 2 considering only conventional performance metrics does not reveal the full picture of the quality of TCP communications in MANETs. In particular we showed that while fairness and throughput metrics can indicate good network performance one or more of the competing flows might suffer from long no-progress intervals. To assess the user level "usefulness" of the end-to-end communications we introduced the "TCP no-progress ratio" metric, which is the accumulated stall times for the particular TCP connection over the life time of the session.

Note that the "TCP no-progress ratio" metric does not capture the maximal duration of the stall interval for the particular session. This time in order to qualitatively assess the progress of each TCP session we construct a more conservative "unsmoothness" metric as is illustrated in Figure 5.1. The construction of the metric is done during the analysis of communication traces. After the end of each test run we parse the packet trace and record the progress of the received sequence numbers in time for each flow and the corresponding start times for every session.

The stop time for all sessions is assumed to be the same. With this information we estimate the ideal curve of the sequence number progress. It is shown by the straight line in Figure 5.1. After that we analyze the recorded process of sequence number arrivals for each flow and compare each received sequence number with the estimated "smooth" value for the corresponding time. The result of the comparison is the absolute deviation of the actual sequence number from the estimated value. Obviously, in most of the measurements we will have some deviation from the estimated curve even for a perfectly smooth flow due to an error of real numbers rounding. In order to allow some small deviations we compute the maximum allowed deviation. This value corresponds to the three seconds no-progress time which was chosen based on our empirical observations that longer no-progress times definitely indicate the presence of the TCP capture phenomenon. Finally, we compute the "unsmoothness" metric as in (5.2).

$$Unsmoothness = \frac{max(|Deviation_{actual}|)}{Deviation^{max}_{allowed}}. \tag{5.2}$$

The "unsmoothness" metric (5.2) is always larger than zero. We say that the quality of a TCP flow is acceptable for an end-user if $Unsmoothness \leq 1$.

5.1.2 Flows of variable path length – the multiple bottlenecks case

We begin the assessment process by considering networks with multiple bottleneck L-regions. We perform this experiment in the network simulator only. We use the topology depicted in Figure 5.2.

In the network we have four TCP connections with different path lengths. All flows use the same transmission rate at the physical layer – 2 Mb/s. In this network we can identify three bottleneck L-regions with three competing connections (TCP1, TCP2, and TCP3) and four bottleneck L-regions with two competing connections (TCP1 and TCP4). For simplicity of the presentation, only one bottleneck of each kind is marked in the figure. Applying the algorithm of C-load shares distribution: $\phi_{TCP1} = 1/3$, $\phi_{TCP2} = 1/3$, $\phi_{TCP3} = 1/3$ and $\phi_{TCP4} = 2/3$. Thus L-region 1 is the bottleneck for flows TCP1, TCP2 and TCP3 and L-region 2 is the bottleneck for flow TCP4, therefore the allocation of

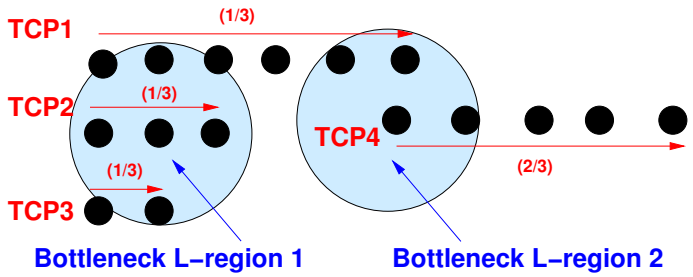


Figure 5.2: Network setting for the experiment with two disjoint bottlenecks.

C-load shares is max-min fair. We run two sets of simulations: In one the assigned load shares were ignored by the sources and each TCP could transmit as much traffic as possible; In the second set of experiments we throttled each TCP connection at the corresponding sources based on the estimates of the boundary C-load and the assigned shares. Figure 5.3 shows the results from this experiment.

The darkest bars show the estimated ideal throughput for each flow. We can immediately mark a severe TCP capture with respect to $TCP1$ in the case where all flows run over standard IEEE 802.11b network. Comparing the resulting TCP throughputs in this case with the estimated ideal value we see that the almost complete shut down of $TCP1$ is caused by the joint effect of transmissions from $TCP3$ and $TCP4$. Being the shortest of all flows, $TCP3$ transmits faster. Apparently the congestion control of $TCP2$ is able to capture its share of load in L-region 1. However $TCP1$ competes not only in L-region 1 but also in L-region 2. Inside L-region 2 $TCP4$ has a vary favorable situation – it competes only with $TCP1$ which is already weakened by the competition with $TCP2$ and $TCP3$. In this situation it is not a surprise that $TCP4$ makes the situation for $TCP1$ even worse.

Now, let us have a look at the bars corresponding to the case where all flows are aware about the competitors on the path and throttle the output rate of TCP segments according to the assigned shares. We see that in this case the communications are fair with respect to each flow. In this case each source generates as much load as the corresponding bottleneck L-region can handle. By doing this no

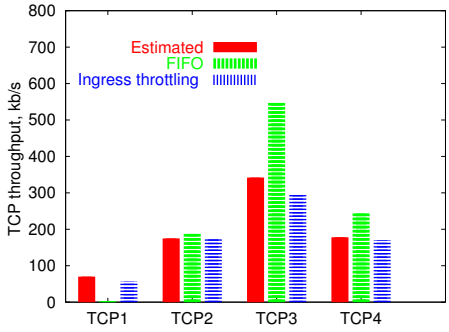


Figure 5.3: Network performance with two bottleneck L-regions.

one flow is able to capture the capacity.

A. Multiple MSS sizes

In this experiment we seek an assessment of our fairness framework in a network with more heterogeneous settings. This time we allow all TCP connections to use MSS of different sizes. We repeat the previous experiment and configure MSS for each TCP session as following: $MSS_{TCP1} = 200$ B, $MSS_{TCP2} = 500$ B, $MSS_{TCP3} = 1000$ B and $MSS_{TCP4} = 100$ B. Figure 5.4 shows the network performance in this experiment.

We again observe a complete shutdown of session $TCP1$ in the case where all TCP sessions transmit traffic in an uncontrolled manner. This time the shortest flow $TCP3$ transmit twice as much as its fair throughput. This is a natural behavior of this flow since it uses MSS of the largest size among all flows. Conforming all flows to our fairness framework the situation changes drastically. The throughput of the silent before that $TCP1$ increased more than 20 times. The original throughput of $TCP1$ in the uncontrolled network is 0.6 kb/s and with our ingress throttling is 23 kb/s, which is only 10 kb/s less than the estimated fair throughput.

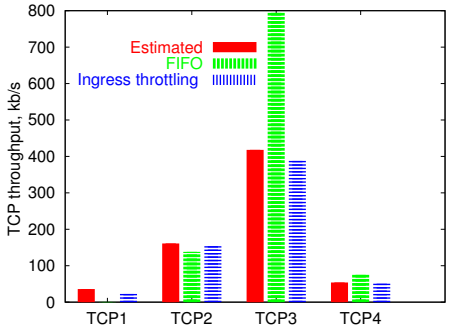


Figure 5.4: Network performance with two bottleneck L-regions and different sizes of MSS. $MSS_{TCP1} = 200$ B, $MSS_{TCP2} = 500$ B, $MSS_{TCP3} = 1000$ B and $MSS_{TCP4} = 100$ B.

5.1.3 The common bottleneck case

As we indicated in Section 4.2.6 it is important to consider a special class of MANETs where all competing connections share at least one common bottleneck L-region. In this section we extensively explore the performance of such networks when our ingress throttling scheme is implemented and analyze the properties of the suggested fairness framework for MANETs.

A. Comparative assessment of the fairness framework in real-world testbed and simulations

The purpose of this experiment is to assess our findings not only in simulations but also in real-world testbed. As a showcase we choose the experiment presented in Section 2.2. There we tested the effect of the optimization of the in-built parameters of TCP and IEEE 802.11 MAC on TCP throughput and unfairness in MANETs. Recall the topology that we used in that experiment – it is shown in Figure 5.5.

In Section 2.2 we found that the optimization of the *in-built* TCP and MAC parameters indeed leads to certain improvements of TCP performance charac-

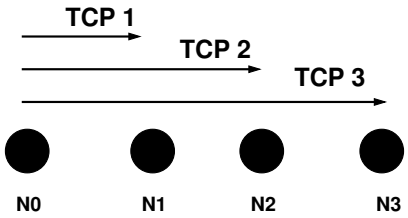


Figure 5.5: Network topology for comparative assessment of the fairness framework in simulations and real-world testbed.

	Simulations		Real-world test	
	Original performance	Rate throttling	Original performance	Rate throttling
Thr_{tot}, (kb/s)	554	605	659	667
$Unfairness$	0.31	0.09	0.27	0.04

Table 5.1: The effect of rate throttling in the case of flows with variable lengths.

teristics. However, in general, even with optimized protocol configurations, the problem of TCP unfairness and hence the sub-optimal TCP throughput remains. The columns "Original performance" in Table 5.1 show the best performance values achieved in simulations and real-world experiments when we apply "window clumping", disable RTS/CTS exchange and separate the interface queue to service the competing TCP connections independently (see Section 2.2 for more details of this experiment). Now we enable the rate throttling for every TCP connection. Moreover in this experiment we disable the "window clumping" mechanism. We discuss the validity and the benefits of this step below. In Table 5.1 we show the resulting network performance in this experiment.

The total TCP throughput is increased on 50 kb/s in simulations and 10 kb/s in real-world experiment. The unfairness reduced drastically more than 3 times in simulations and almost 7 times in the real-world test-bed. Although the exact values of the used metrics are different in simulations and the real-world experiments,

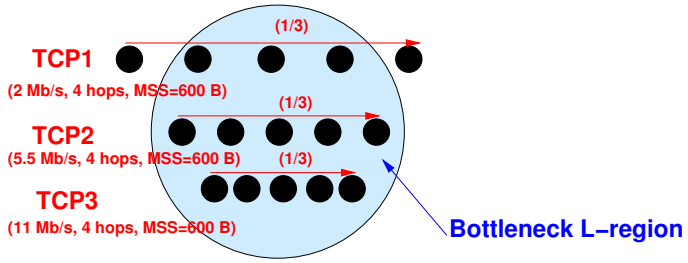


Figure 5.6: Network settings for the experiment with multiple transmission rates at the physical layer.

in this experiment we show a similar improvement of the network performance in the two used environments.

Recall that in this experiment we do not artificially restrict TCP's $CWND$ to a certain value. The major reason for this is the nature of the ingress throttling mechanism and the background motivation for using the "window clumping". The idea behind $CWND$ restriction is to do not allow the particular TCP session to send more traffic than the network can handle. Apparently this idea is embedded in our ingress throttling scheme. The source nodes in our scheme limit the transmission rate in order to do not overload the bottleneck L-regions. Therefore additional limitation of the data flow rate at the TCP layer is not needed. The benefit of this property of our scheme is clear. The "window clumping" is somewhat unnatural approach at the transport layer because TCP does not know the path length for the particular flow. Now we naturally implemented this mechanism at the middle layer, where this information is accessible from the routing protocol.

B. Flows with variable 802.11b transmission rates

In this experiment we assess a very important aspect of the validity of our theoretical findings in a network where the competing flows might use all available transmission rates of the IEEE 802.11b physical layer. We assume that the information about the transmission rate at the physical layer along the path of the competing connections is available at the sources. In Figure 5.6 the network topology for

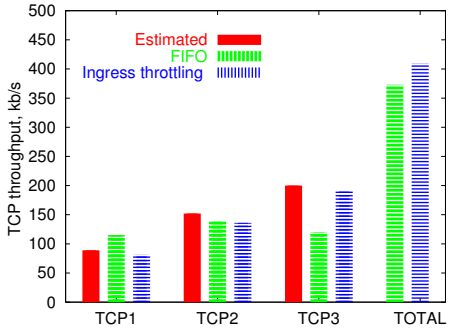


Figure 5.7: Network performance: flows with different 802.11 transmission rates.

this experiment is depicted. There are three competing four hops TCP connections which use correspondingly 2, 5.5 and 11 Mb/s transmission rate between the hops. Note that the choice of the internode distances for the three connections is in accordance to the transmission rate used by the particular TCP flow (see Figure 4.2 for the details). All connections use $MSS = 600$ B. We performed two series of the experiment: In the first case the sources do not perform shaping of the outgoing traffic and in the second case our ingress rate throttling is enabled. The performance results are shown in Figure 5.7.

As for the original network performance depicted in the figure by the middle bars in each group, we see that flow $TCP1$, which uses the slowest transmission rate, achieves higher throughput than the estimated value under fair allocation of C-load shares. This is a natural and well known situation known as performance anomaly of IEEE 802.11 MAC [28]. In this phenomenon the MAC protocol attempting to provide max-min allocation of the transmission rates at any node favors transmissions with slower rates by this degrading the performance of potentially faster transmissions.

Enabling our ingress throttling we restrict this flow to use only its share of the C-load in the L-region. By doing so the resulting per-flow throughputs are proportional to the underlying transmission rates. This proportionality is embedded in our interpretation of the boundary C-load and the computation of the throttling

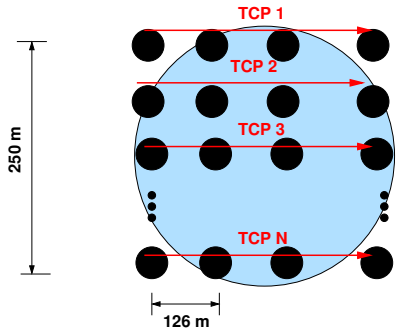


Figure 5.8: A set of three hop networks.

limit based on the individual properties of the competing connections. As a result we also observe an increase in total TCP throughput in the network in comparison to the original behavior of TCP over IEEE 802.11 MAC.

The proportionality of TCP throughputs observed in this experiment is one of the most important properties of our fairness framework. Firstly it is in accordance to the recent findings in the area of fairness in MANETs: In [59] the authors numerically analyzing max-min and proportional types of fairness in multihop wireless network conclude that proportionally fair rates allocation in MANETs is preferable over the max-min fair allocation. Moreover we can reference the work in [33]. There the authors prove a proposition that the proportional fairness is achieved when the wireless stations in the WLAN have the same fraction of the "effective air-time usage". They also show that the *bandwidth proportional* fairness in WLANs is equivalent to *air time max-min* fairness.

C. Checking the hypothesis of maximal TCP throughput of a single flow as a reference to the boundary C-load.

This experiment we perform on a set of TCP flows with equal potentials all crossing the common bottleneck L-region in the network depicted in Figure 5.8. In this

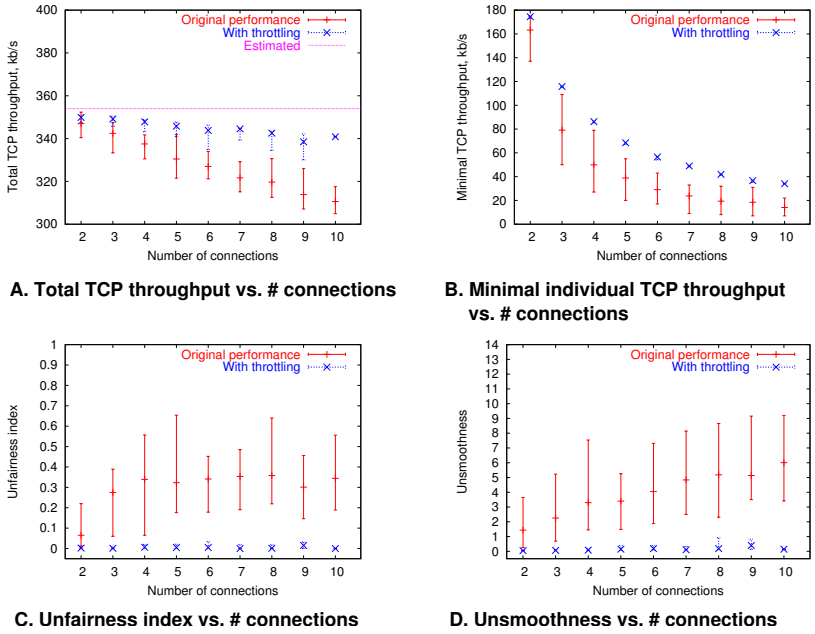


Figure 5.9: Network performance in the scenario with equal opportunities TCP flows.

network the majority of network nodes are located inside the single bottleneck L-region. This network allows us to test two things. Firstly, we can verify whether our hypothesis of taking the maximal throughput of a single TCP connection as a reference to the boundary C-load inside an L-region is valid. Indeed, if it is a wrong hypothesis then the total TCP throughput of all TCP connections running without shaping will be higher than this value. Secondly, we perform the first scalability test, checking the dynamics of the unfairness index with the increase of the number of competing connections. We vary the number of connections inside the L-region from 2 to 9.

Figure 5.9 shows the dynamics of the combined TCP throughput in the network Thr_{tot}, the minimal individual TCP throughput, the unfairness index (5.1),

and the "unsmoothness" metrics for different numbers of the competing connections inside the common L-region. The straight line marked as "Estimated" in Figure 5.9*a* corresponds to the TCP throughput of a single three hops TCP flow running alone in the network. All graphs show the mean values and the range between the minimal and maximal values of the corresponding metrics over 30 simulation runs. In each run we seeded the random number generator of ns-2 randomly.

As we observe from the figure, the total TCP throughput in the case where no shaping is done by the sources and TCP flows can transmit as much traffic as they can is always lower than the throughput of a single session (the straight line in Figure 5.9*a*). By this we confirm our hypothesis from Section 4.2.3 to consider TCP throughput of a single TCP flow in isolation as a reference to the boundary C-load of the L-region.

From the figure we observe that enabling the ingress throttling the resulting total throughput is equal to or larger than that in the case of plain combination of TCP and IEEE 802.11 MAC. Moreover, the maximal deviation from the estimated value in the case where our scheme is enabled is only 3% while in the case without throttling this value is 12%. The reason for that we cannot achieve a perfect match of the total throughput to the estimated value is that controlling the load inside the L-region we do not control the contention at the MAC layer. As a result packets transmitted simultaneously from different stations collide during transmission. Therefore each individual and subsequently the total throughput in the network decreases.

From sub-figures *b,c and d* we see a dramatic improvement of the quality of TCP communications in the case of enabled ingress throttling. Under our scheme we achieve an increase of the minimal individual throughputs, all flows are smooth and free from long interruptions and overall almost perfect fairness is achieved.

It is important to point out the stability of the performance metrics. When the ingress throttling is enabled the network consistently shows much better performance in all simulation runs. At the same time the service offered by the plain combination of TCP and IEEE 802.11 MAC is highly variable and to a large extend unpredictable.

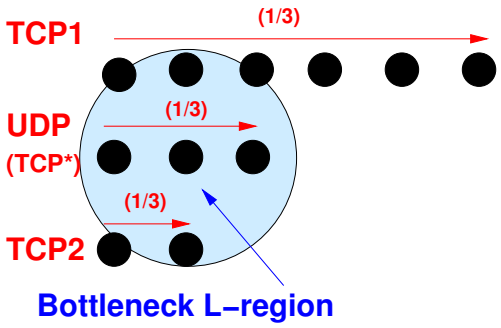


Figure 5.10: Network setting for the experiment with UDP traffic.

5.1.4 Treating UDP traffic inside the space-load fairness framework

In order to conform UDP flows to the *space-load* model, their transmission rate should be limited as we discussed in Section 4.2.4. In this section we present the experiments which illustrate the effect of UDP traffic throttling. For this purpose we used the topology depicted in Figure 5.10. There we have three data sessions: $TCP1$ is a 5 hops TCP flow, UDP is a two hops UDP flow and finally $TCP2$ is a one hop TCP connection. We perform three sets of simulation based experiments in this network as described below.

A. Experiment set 1: Reference case

In this experiment instead of the two hops UDP flow we have a TCP connection (TCP^*) of the same length. We run simulations where all three flows send traffic in an uncontrolled manner and then enable our ingress throttling scheme. This experiment should serve as a reference with respect to the achieved throughput by flows $TCP1$ and $TCP2$ under fair sharing of C-load in the bottleneck L-region. In this network each flow should not generate and/or consume more than 1/3 of the boundary C-load. Table 5.2 shows the estimates of $Thr_{max}(h, MSS, TX_{802.11})$ for our three flows (first column), the target throughput when all flows conform to their fair shares of C-load (second column), the achieved throughput when our

Flow id	Thr_{max} (kb/s)	$Thr_{max} \cdot 1/3$(kb/s)	Throughput w/o throttling (kb/s)	Throughput w throttling (kb/s)
$TCP1$	212	70	87	70
TCP^*	527	175	188	177
$TCP2$	1026	342	286	314

Table 5.2: The reference case for the experiments with UDP traffic.

ingress throttling is disabled and when our scheme is enabled (third and fourth columns respectively). For simulations we set $MSS = 600$ B and the transmission rate at the physical layer to 2 Mb/s.

As it is visible from the table the initial unfairness is not too large. However, applying our ingress throttling the throughput of $TCP2$ increases on 28 kb/s. The total TCP throughput in the network in both cases is 561 kb/s (not shown in the table). Now let us add UDP flow instead of flow TCP^*.

B. Experiment set 2: UDP flow with potentially fast transmission rate

In this set of experiments we initially configure UDP application to transmit with a high data rate. The transmission rate of UDP traffic at the application layer, hence the resulting throughput of the UDP flow running alone in the network is 480 kb/s. All flows use packets of equal size 600 B and the underlying transmission rate at the physical layer is 2 Mb/s. The purpose of this experiment is to explore the quality parameters of all three flows in the following three cases: (a) All flows send traffic in an uncontrolled manner; (b) TCP flows are throttled according to their fair share of the bottleneck C-load and the UDP transmission is uncontrolled; and finally (c) all flows including UDP limit their transmission rate according to their fair C-load share ($\phi = 1/3$).

To assess the quality of the UDP flow in addition to the end-to-end throughput we also will measure the packet loss rate computed by the following formula:

$$Loss = (1 - \frac{Num_{recvd}}{Num_{sent}}) \cdot 100\%,$$

Flow id	Throughput w/o throttling (kb/s)	Throughput w throttling of TCP only, (kb/s)	Throughput w throttling all flows, (kb/s)	Reference fair throughput w throttling, (kb/s)
$TCP1$	114	70	70	70
UDP	219 (55%)	347 (27%)	186 (0%)	186
$TCP2$	230	212	318	342

Table 5.3: UDP experiment set 2: Potentially fast UDP flow.

where Num_{send} is the number of packets sent by the ingress node and Num_{recvd} is correspondingly the number of packets received at the destination node. Table 5.3 shows the results of our three experiments. In the fields containing the throughput of the UDP flow we also indicate the packet loss rate in parentheses.

Let us analyze the results. Firstly, consider the case where the ingress throttling is disabled for all flows (first column in the table). We observe that the throughput of $TCP2$ is about 110 kb/s less than the estimated fair throughput (see last column in the table). At the same time $TCP1$ exceeds its fair throughput on more than 40 kb/s. This is a clear indication of unfairness between TCP flows. Let us now observe the performance of the UDP flow. In addition to the loss in throughput (260 kb/s in comparison to its potential transmission rate 480 kb/s), the flow experiences a very large packet loss rate (55%). This is certainly unacceptable for any real-time UDP based application. Overall, in the first experiment we monitor a degradation of the communications quality both for TCP and UDP traffic.

Let us now enable the ingress throttling for the two TCP flows, leaving the transmission from the UDP source uncontrolled. The second column in the table shows the results of this experiment. What we observe now is that by forcing TCP flows to transmit with not more than their fair rate, the performance of UDP flow increases (the throughput increases on about 130 kb/s and the loss rate decreases two times). However the higher transmission rate of the UDP flow results in the further decrease of $TCP2$ throughput – now it is 130 kb/s less than the estimated fair throughput. Here we see a clear need for the UDP source to reduce its transmission rate in order to prevent the starvation of the TCP flows.

For the last experiment we computed the transmission rate limit for the UDP flow according to formula (4.6). Taking the estimate of the maximally achievable TCP throughput $Thr_{max}(h, MSS, TX_{802.11})$ for $h = 2, MSS = 600B$, $TX_{802.11} = 2$ Mb/s (see the value in the first column of Table 5.2 for TCP^*), the resulting rate limit for the UDP flow in our test network is 186 kb/s.

There are two possible ways to enforce this rate limit for the UDP flow at its source. In the first case we can assume that the UDP application is adaptive, meaning that it has a functionality of adjusting the transmission rate according to the information provided by a signaling scheme. In this case we can limit the transmission rate already at the application layer. The second option is to leave the application "blind" and to enforce the rate limit at the interface queue in the same way as we do for the TCP traffic. The obvious drawback of this approach is that UDP application will continue to experience the high packet loss rate. For the experiments we choose the first approach, since current UDP applications already support one or another kind of signaling.

The results of the last experiment are shown in the third column of Table 5.3. We immediately observe the level of the performance improvement for $TCP2$. Now its throughput differs from the estimated fair throughput only on 20 kb/s. At the same time UDP's throughput is now conforming to its fair value and the packet loss rate is zero. Overall by enabling our ingress throttling both UDP and TCP traffics win.

C. Experiment set 3: UDP flow with low transmission rate

The goal of this experiment is to monitor the performance of the three considered data sessions in the case where UDP application generates traffic at a slower rate than the computed fair rate limit. With all other settings being unchanged we configure the output rate of the UDP traffic at the application layer to be 35 kb/s (compare with 186 kb/s of the estimated fair rate limit). Obviously with such settings when our ingress throttling is enabled the network will be underutilized. However, assume that the UDP application can signal the desired transmission rate to the control plane, which in turn converts this value to the fraction of the full speed connection and passes the result as a desired load share to the C-load distribution algorithm. In this case the algorithm will allocate larger shares in our

Flow id	Throughput w/o throttling (kb/s)	Throughput w throttling $\phi = 1/3$, (kb/s)	Throughput w throttling $\phi = 0.4$, (kb/s)	Reference fair throughput w throttling, (kb/s)
$TCP1$	152	70	84	84
UDP	35 (0%)	35 (0%)	35 (0%)	35
$TCP2$	310	330	397	410
TOTAL	497	435	516	529

Table 5.4: UDP experiment set 2: Initially slow UDP flow.

bottleneck L-region to $TCP1$ and $TCP2$. In our example this scheme would work as follows.

1. The UDP application supplies the share distribution algorithm with the desired rate 35 kb/s;

2. At the ingress node the algorithm realizes that it is roughly 1/5 of the full speed connection ($\frac{35}{186}$);

3. The algorithm assigns the load share 1/5 to the association carrying the traffic of this connection;

4. At the second iteration of the algorithm the associations carrying flows $TCP1$ and $TCP2$ obtain C-load shares 4/10 each.

With these C-load shares the estimated rate limits, hence fair throughputs for $TCP1$ is 84 kb/s and for $TCP2$ is 410 kb/s. Let us now see the actual values of the throughputs in the cases where (a) the ingress throttling is disabled; (b) when it is enabled with load shares $\phi = 1/3$ for $TCP1$ and $TCP2$; and (c) with load shares $\phi = 0.4$ for the two TCP flows. Table 5.4 shows the results of the three experiments. The important aspect to observe here is the increase of the throughput for $TCP2$ on almost 90 kb/s and the increase of the total network throughput (from 497 kb/s to 516 kb/s) in the case where the load distribution algorithm accounts for the slower transmission rate of the UDP flow.

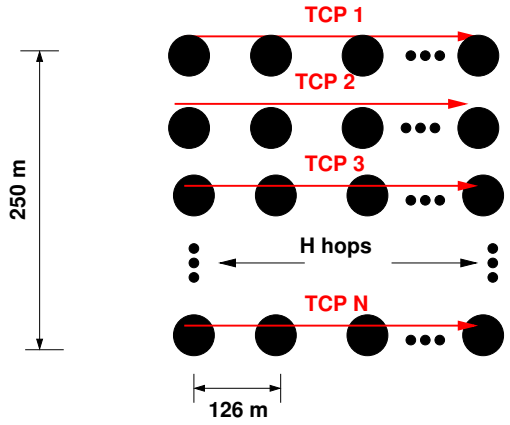
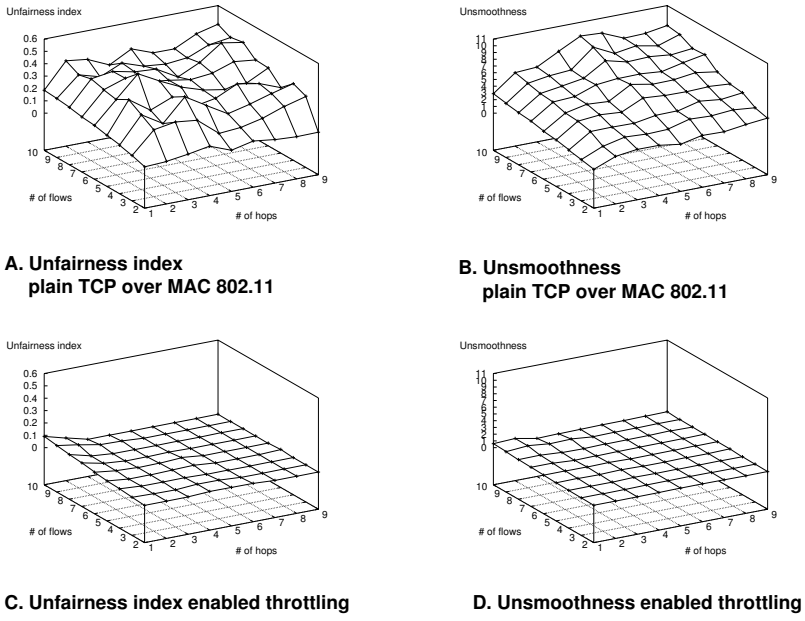


Figure 5.11: A family of topologies for detecting the ad hoc horizon.

5.1.5 Further scaling the network – the "ad hoc horizon" without and with ingress throttling

Finally we perform an experiment to see the effect of the fairness framework and the ingress throttling scheme on our "ad hoc horizon" metric. Recall that we have introduced this metric to capture the scaling limit of the combination of TCP and IEEE 802.11 MAC. In Chapter 2 we showed quite a pessimistic value of this metric. According to our estimation the current scale of the IEEE 802.11 based multihop ad hoc networks with respect to TCP communications is very narrow and is in the range of two to three simultaneously active connections with paths of up to two hops. Let us now see how this horizon has changed when we enable the ingress throttling scheme. In order to show the effect of our solution on a broader range of networks we performed a series of experiments for a family of topologies depicted in Figure 5.11.

This time we scaled the network in two dimensions: the lengths of connections and the numbers of competing flows. Since the unfairness metric is valid for



A. Unfairness index plain TCP over MAC 802.11

B. Unsmoothness plain TCP over MAC 802.11

C. Unfairness index enabled throttling

D. Unsmoothness enabled throttling

Figure 5.12: TCP unfairness index and *unsmoothness* metric for the family of "ad hoc horizon" topologies.

evaluation of two or more flows we varied the number of competing TCP flows from 2 to 9. The route lengths for each flow is scaled from 1 to 9 hops. The results are summarized in Figure 5.12.

We observe that the unfairness virtually vanishes when throttling according to our rate limit is implemented. Moreover the flat surface of the "unsmoothness" metric indicates that the progress of all competing flows is close to the ideal. This is probably the most illustrative result: As one can see the two hops / three connections ad hoc horizon as we had with the original combination of TCP and IEEE 802.11 MAC does not exist any more.

5.2 Discussion of the Huang-Besaou-Jing-Liew "space-air-time" fairness framework

We devote this section to the detailed description of a fairness framework for MANETs which is closely related to our *space-load* fairness model. This fairness framework was presented in two disjoint publications: In 2001 Huang and Besaou [30] presented their formal max-min fairness framework on *ACM MobiHoc*. In 2005 Jing and Liew [33] extended the framework presented in the first publication to proportional fairness.

In the first paper the proposed fairness framework assumes equal capabilities of the transmitting stations in terms of the physical layer transmission rates. The capacity in MANETs is considered as a function of space. The authors suggest a practically implementable algorithm of fair distribution of capacity shares between the contending data flows.

The second paper shifts the focus from the *space-capacity* domain to the *space-air-time* domain. Leaving the major functionalities of the initial framework unchanged the authors in [33] suggest another interpretation to the assigned shares. Now instead of the amount of traffic transmitted by each session in bits, the *air time* of the particular session in the contention region is considered. This new interpretation allowed the authors to generalize the initial fairness framework and to prove an important property of the proportionality of the resulting end-to-end throughput when the air-time shares for the sessions inside the contention region are max-min fairly allocated.

Jointly we will refer the fairness model presented in these two papers as to the *space-air-time* fairness framework. In this section we first describe the major functionalities of the *space-air-time* model and then give a detailed comparison of the common and different parts between this and our *space-load* fairness framework.

5.2.1 The "space-air-time" fairness framework

The formal machinery of the *space-air-time* fairness framework is based on the representation of MANETs communications in a form of the flow contention graph. Consider a sample network topology and a set of end-to-end data flows

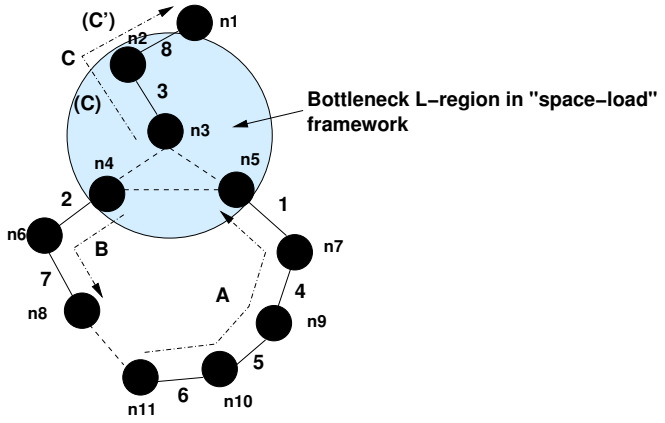


Figure 5.13: Comparison of the "space-air-time" and "space-load" fairness frameworks. Example 1.

depicted in Figure 5.13[1]. In this scenario we have a MANET with three end-to-end data sessions *A,B* and *C*. They are shown by the dash-dotted lines in the figure.

For the construction of the flow contention graph in the *space-air-time* fairness framework all end-to-end multihop flows are viewed as a set of independent one hop flows. In Figure 5.13 these one hop flows are shown by plain lines between nodes and are marked with numbers $1, ..., 8$. The presence of the contention between nodes which do not have transmission between each other but are located in the same area of the assured radio signal reception is indicated by the dashed lines. In the flow contention graph each one hop flow is represented by a vertex and the edge between two vertices indicates the presence of the contention.

Consider a subgraph with vertices $1, 4, 5$ from Figure 5.14*a*. The two flows in the *space-air-time* framework are counted as contending if they are visible at

[1]For the presentation of the *space-air-time* fairness framework we use the original network topology used by the authors of the framework. See Figure 2 in [30].

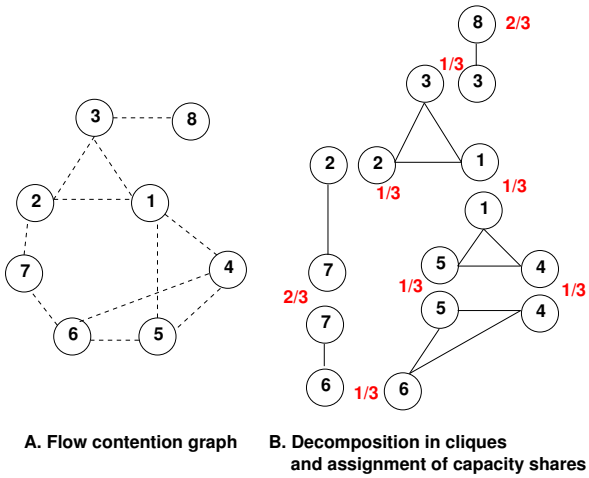


Figure 5.14: Flow contention graph and decomposition in cliques in the "space-air-time" framework for topology 1.

either end of the two flows. In our example flow 1 indeed contends with flow 5 (in the sense of the *space-air-time* framework) because the transmission from node $n7$ (the starting node of flow 1) can be heard at node $n9$ (the terminating node of flow 5). As an opposite example consider the contention subgraph which includes flow 8. Indeed, according to the *space-air-time* framework flow 8 does not contend with flows 1 and 2 because the transmission from node $n2$ cannot be heard at any node involved in the transmission process of flows 1 and 2.

Now, considering flow contention graph $G = (N, A)$, where N is the set of one hop flows and A is the set of the vertices representing the presence of contention between flows, a clique cl is defined as a subset of N such that for all distinct pairs of vertices $u, v \in cl$, the edge $[u, v] \in A$. In other words the clique is a *complete subgraph* of the flow contention graph $G = (N, A)$. The clique in the *space-air-time* framework is an analog to the *wireline link*.

The goal of the *space-air-time* fairness framework is to achieve *max-min* fair assignment of capacity shares between all cliques of the flow contention graph.

Figure 5.14*b* illustrates the creation of the flow contention graph and its decomposition in cliques for the sample topology presented in Figure 5.13.

A. Max-min fair share assignment in a global contention graph

Let $CL = \{cl_1, cl_2, ..., cl_n\}$ be the set of all possible cliques, $D = \{d_1, d_2, ..., d_n\}$ be the number of vertices (contending flows) in the corresponding clique (degree of a clique) and $C = \{c_1, c_2, ..., c_3\}$ be the capacities of the corresponding cliques initialized to 1. The authors of the *space-air-time* fairness framework suggest the following algorithm of fair capacity share assignment:

1. Sort CL in a nondecreasing order of CD^{-1}

$$\frac{c_i}{d_i} \leq \frac{c_j}{d_j} \leq ...,$$

2. Assign the share to all vertices in the clique with smallest CD^{-1} value as

$$\forall v \in cl_i : \phi_v = \frac{c_i}{d_i},$$

3. Remove all vertices of cl_i from all other cliques in CL and update the degrees and capacities of the cliques

$$\forall cl_k \in CL : \begin{cases} cl_k & \leftarrow & cl_k \backslash cl_i \cap cl_k \\ d_k & \leftarrow & d_k - \mid cl_i \cap cl_k \mid \\ c_k & \leftarrow & c_k - \frac{c_i}{d_i} \times \mid cl_i \cap cl_k \mid \end{cases}$$

4. if $max(d_i) = 1$ then

5. STOP

6. else

7. GOTO 1.

The assignment of the capacity shares to the one hop flows in the identified cliques in our sample contention graph are shown in Figure 5.14*b*.

As it is proved in [30] the allocation of the capacity shares to the one hop flows according to the presented above algorithm is max-min fair. The authors also suggest a distributed version of the share assignment algorithm and a set of mechanisms at the MAC layer by means of which the contending flows conform their offered load to the assigned air time shares.

We do not further discuss other details of this framework since the presented description is sufficient for the cross comparison of the *space-air-time* framework and our *space-load* framework. We proceed with the comparison in the next subsection.

5.2.2 Functional comparison of the "space-air-time" and our "space-load" fairness frameworks

The most obvious common point of our *space-load* fairness framework and the *space-air-time* framework is the understanding of the MANET's capacity (or in our interpretation of this term as *C-load*) as a function of space. In both models the boundary "capacity" is normalized to 1 due to the non-determinism of its definition. On the other hand the most obvious differences between the two models are:

1. Atomicity of the considered contended entities;
2. The degree of contention between them.

If in the *space-air-time* framework a multihop end-to-end data flow is viewed as a set of independent one hop flows, in our *space-load* framework we consider the contention between entire multihop flows (*associations*). We comment on the details of the first difference below and now discuss the implications of the difference in the degree of contention used in the two frameworks.

A. Degree of contention in the two frameworks

Let us illustrate this difference on the example of the network depicted in Figure 5.13 above. There we also marked the bottleneck L-region as it is identified by our

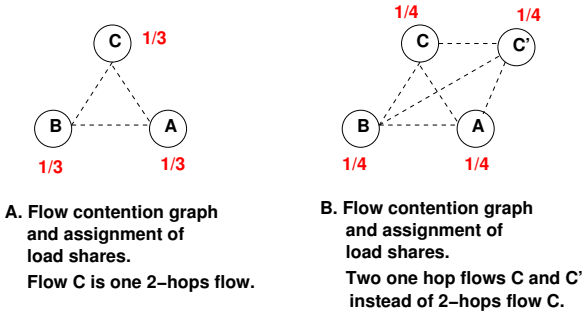


Figure 5.15: Flow contention graph and assignment of C-load shares in our "space-load" framework for topology 1.

algorithm of C-load shares distribution. Consider two possible scenarios reflected in the figure: (*a*) In the first scenario flow C traversing nodes $n3$, $n2$ and $n1$ is a two hop flow and (*b*) in the second scenario instead of this we have two independent one hop flows C and C'.

Let us compare the resulting assignment of the shares in the two frameworks (see Figure 5.14*b* for the capacity shares assignment in the *space-air-time* framework). Note that in the case of the *space-air-time* framework the flow contention graphs, hence the share assignment for the competing flows are the same for the two considered cases. However in our framework we assign the C-load shares differently in each case. In Figure 5.15 we show the detected by our algorithm of C-load shares assignment bottleneck L-region in the form of the flow contention graph for the two cases.

As one can see in case (*a*) when C is the two hops flow, our algorithm assigns shares $\{A = 1/3, B = 1/3, C = 1/3\}$. The same result is achieved by running the capacity share assignment algorithm of the *space-air-time* framework on the

contention graph for flows 1,2 and 3. Assuming the equal transmission rates at the physical layer the resulting throughput of flows B and C will be limited to 1/3 on their first transmission leg and of flow A to the same fraction on its last leg. Therefore at the end all three flows in the two compared frameworks will achieve the same 1/3 of their ideal throughput.

Now consider case (*b*) where instead of the two hop flow C we have two one hop flows C and C'. While nothing has changed in the contention graph of the *space-air-time* framework, in the L-region of our framework a new end-to-end flow appeared. The contention graph for the bottleneck L-region in this case is shown in Figure 5.15*b*. As we see, our algorithm assigns the load share $1/4$ to each end-to-end session.

The difference in the shares assignment is explained by our definition of the *L-region*. In the case of our framework, inside the particular L-region we also include the flows which might be outside the area of assured data reception of specific associations, however which contend for the transmission medium in the interference region of these associations. This can be illustrated by the example of the transmission and the reception processes at nodes $n2$ and $n5$ respectively. If we do not account for the connection C' (or one hop flow 8 in the case of the other fairness framework) in the computation of the capacity/load shares we might end up in the situation where continuous transmissions from node $n2$ will cause packet losses at node $n5$ due to the effect of long-ranging radio interferences. As we illustrated in Chapter 2 this is the key reason for the formation of the TCP capture problem.

B. Atomicity of the contending entities

The example presented above also leads us back to the discussion on the first major difference between the two fairness frameworks – the atomicity of the considered contending entities. Let us have a look at the capacity share distribution in the *space-air-time* framework in Figure 5.14*b*. What we have here is that the algorithm of this framework assigns different shares to the same end-to-end data session on its different transmission legs. Consider flow C as a two hop flow: The share assignment algorithm of the competing framework assigns share $1/3$ on the first hop and $2/3$ on the second hop. How relevant is such assignment from the

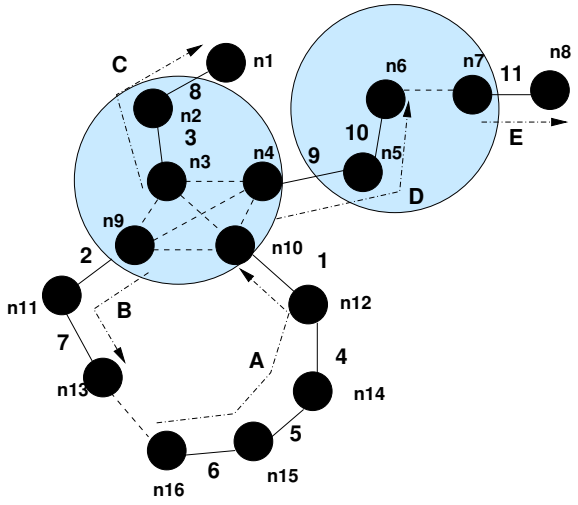


Figure 5.16: Cross-comparison of "space-air-time" and "space-load" fairness frameworks. Example 2.

flow's point of view? Obviously if there exist a mechanism by means of which the assigned shares are enforced in the network, flow C would be restricted to 1/3 of its throughput independently on the larger share on its second hop.

In our *space-load* framework we propagate the assigned C-load share over the entire path of the flow. This potentially would give us a flexibility of giving higher C-load shares to other flows in other bottleneck L-regions traversed by this flow. We illustrate this difference between the two frameworks on example as follows. Let us add another two hops flow D to the previous scenario. The resulting network is depicted in Figure 5.16 (let us ignore for a moment the one hop flow E and the corresponding L-region shown in the figure).

Figures 5.17 and 5.19*a* show the constructed flow contention graphs and the capacity/load share assignments in the *space-air-time* and our *space-load* frameworks correspondingly. To this end we see again that the two frameworks result

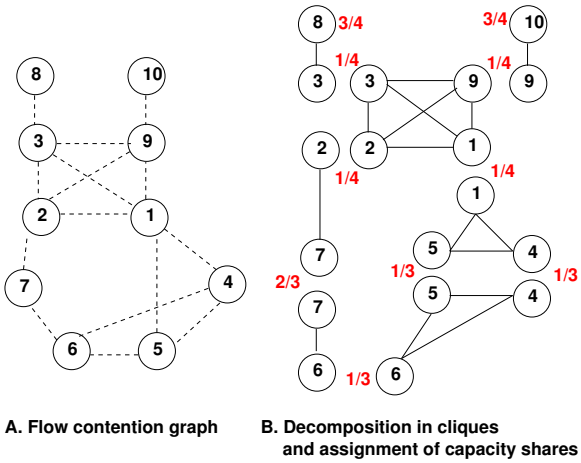


Figure 5.17: Flow contention graph and decomposition in cliques in the "space-air-time" framework for topology 2 without flow E.

in reduction of the throughput of the contending data sessions in the bottleneck regions (or cliques) to $1/4$ of what the flows would achieve running along in the network. Let us add yet another one hop flow E (one hop flow 11) and see the shares assignment in this case. The results of the two algorithms after their termination are shown in Figures 5.18 and 5.19*b*.

We immediately mark that considering the two hops end-to-end flow D as a set of two independent one hop flows (flows 9 and 10) the *space-air-time* suggest to reduce the throughput of one hop flow E to $1/2$. While in our framework the same flow obtains the share $3/4$.

By this illustrative example we showed a better spacial re-use of resources in our *space-load* fairness framework. However, we would like to be very clear about not judging the *correctness* of the later framework. Moreover, in the target environment for this framework this may be the only achievable fair allocation of the resources. We point out this aspect in Section 5.2.3 while discussing the implementation issues.

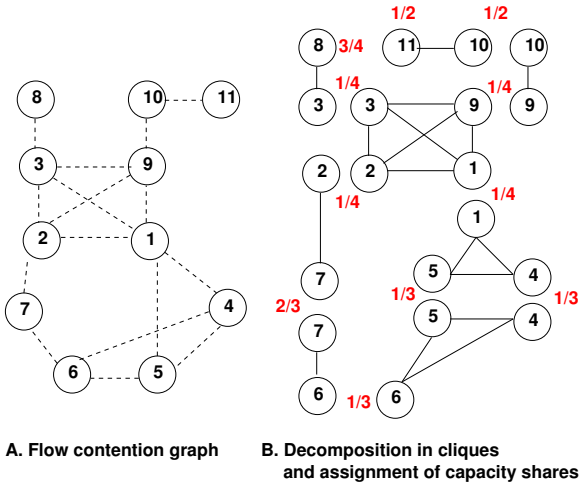


Figure 5.18: Flow contention graph and decomposition in cliques in the "space-air-time" framework for topology 2 with flow E.

C. Interpretation of the assigned shares

In comparison to the *space-air-time* framework we also interpret the assigned load shares differently. In the former case the authors require an on-line estimation of the used air-time from all contending flows at the MAC layer. This estimation happens periodically. By the end of each period the MAC layer compares the results of the estimation to the computed fair share for each flow and adjusts (either reduce or increase) the contention window of IEEE 802.11 MAC accordingly on the per-packet basis.

In the case of our framework we operate with the estimates of the maximally possible load that the particular end-to-end session can generate within the bottleneck L-region. This allows us to deterministically decide on the shaping delay Δ for every session at the ingress node. By doing this we of course foresee some loss of network utilization in the case where the particular flow does not fully use the estimated fair transmission rate. However, as a payback we obtain a deterministic

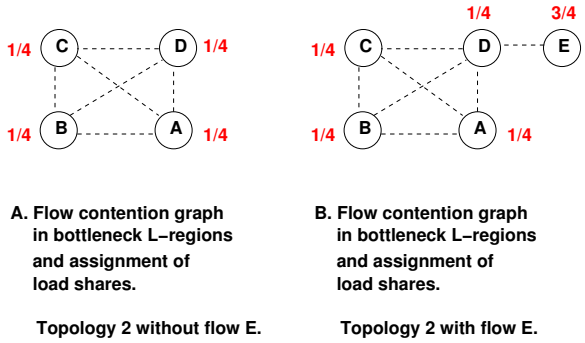


Figure 5.19: Flow contention graph and assignment of load shares in our "space-load" framework for topology 2 with and without flow E.

guarantee on the maximally achievable throughput and fairness for each flow.

5.2.3 Implementation comparison of the "space-air-time" and our "space-load" fairness frameworks

The important difference between the two considered fairness frameworks is the target implementation place inside the TCP/IP protocol stack. The *space-air-time* framework is intended for implementation at the MAC layer. The authors suggest to distinguish between the contending flows by overhearing the transmissions of the MAC layer control messages and the data packets. They also propose a special message exchange scheme for distributing the local knowledge about the contending flows between the neighbors. This implementation explains the choice of the authors to consider the entire multihop data session as a set of independent one hop flows. In general at the MAC layer there is no way to spread the computed air-time shares for the entire length of the data session.

As for our *space-load* fairness framework we initially abstracted from the limitations of the MAC protocol. We designed our framework for implementation at the middle layer, close to IP, where the information about the end-to-end nature of

the sessions is present. This allowed us to think of a routing protocol for MANETs as a network resource control plane which is able to distributively compute the fair C-load shares on the *per-association* basis and propagate this information to other flows ongoing beyond the reachability of the MAC protocol for the particular node. This favorable position of our framework within the TCP/IP protocol stack also eliminates the need for adding new functionalities to the standard IEEE 802.11 MAC. We perform all the shaping actions before the data packets enter the MAC layer and only at ingress nodes. The last property also implies that the intermediate nodes do not participate in the scheduling decision and only execute the shares assignment algorithm.

5.3 Discussion

Our ingress rate throttling solution is an adaptive distributed capacity allocation scheme for multi-hop wireless networks. The capacity is allocated on a per-session basis at a specific point in time during the route establishment. In this section we discuss the implications of the presented scheme and future research directions.

5.3.1 The effective ad hoc horizon

As we have observed in Section 5.1, enabling our ingress throttling scheme results in a significant extension of the scaling limit of the IEEE 802.11 based MANETs with respect to fairness of TCP communications. The flat surface of the ad hoc horizons (see Figure 5.12 *c,d*) indicates that fair TCP communications are possible for virtually any number of simultaneously active connections with very large routing paths. However a valid question is how useful this finding is from the user's point of view? What we essentially do in our solution is that we force every ingress node to limit its transmission rate depending on the number of competing sessions in the bottleneck L-region.

Obviously for an arbitrary TCP or UDP session there is a limit on the number of competitors in its bottleneck L-region after which the resulting rate throttling limit becomes so small that it would not satisfy even minimal user's expectations.

We call this limit the *effective* ad hoc horizon. We suggest that the number of simultaneously active sessions in the network should be limited in order to avoid such situation.

There are two possible ways to accomplish this requirement. One way is to place a "session watchdog" in all network nodes; Another way is to implement it at the ingress nodes. The problem with the first approach is that the network nodes would require a large amount of per-flow information. Namely, the nodes should know all parameters needed for the computation of the rate limit for each connection inside their L-region: the number of hops, the physical layer transmission rate, the used packet size. We foresee that this kind of scheme is hardly implementable in reality.

On the other hand, it is reasonable to assume that the users themselves would not commence the communication if they would know the expected throughput on the particular path. Hence it would be a natural solution to move the watchdog functionality to the ingress nodes. Based on the network state information (namely the suggested C-load share and the characteristics of the locally originated data session) the ingress nodes can compute an estimate of the resulting end-to-end throughput before the application even starts the data transmission. If the expected throughput is less than the meaningful minimal value then the control plane should advise the application to either postpone or abandon the communication. This, in the simplest case, can be done by not submitting to the ARP table the routing information for the particular destination.

Definition 5.1 (Effective ad hoc horizon): For the particular TCP session we call the combination of the network state (the number of active sessions in the L-region, the suggested C-load share) and the parameters of this data session (the number of hops on the path, the physical layer transmission rate, the used packet size) which results in the useful minimal throughput of this data session the *"effective ad hoc horizon"*.

In order to illustrate this concept let us construct the horizon for a data session which might use one of the available 802.11b transmission rates and traverse different number of hops. We will construct the horizon for a TCP session with $MSS = 600$ B. Firstly, let us define the meaningful minimal end-to-end through-

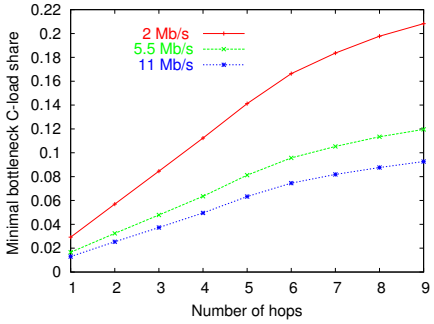


Figure 5.20: Effective ad hoc horizon: The minimal acceptable assigned fair share of C-load for a single session. MSS=600 B.

put for a TCP session to be $Thr_{min} = 30$ kb/s. We see the throughput below this value as unacceptably low. Secondly, we estimate the maximally achievable throughput of an isolated TCP session for different physical layer transmission rates and number of hops traversed by the flow ($Thr_{max}(h, MSS, TX_{802.11})$). Now dividing the estimated maximal throughput by Thr_{min} reveals the maximal number of the competing connections inside the L-region for our flow:

$$Num_{max} = \lfloor \frac{Thr_{max}(h, MSS, TX_{802.11})}{Thr_{min}} \rfloor. \tag{5.3}$$

Assuming that the boundary C-load inside the bottleneck L-region is evenly distributed between the competing connections (ϕ is computed using formula in Step 1(a) of the load distribution algorithm), the minimal acceptable assigned C-load share for this connection is

$$\phi_{min} = \frac{1}{Num_{max}} = 1/\lfloor \frac{Thr_{max}(h, MSS, TX_{802.11})}{Thr_{min}} \rfloor. \tag{5.4}$$

Figure 5.20 shows the values of ϕ_{min} for different number of hops and different physical layer transmission rates for a TCP session which uses $MSS = 600$

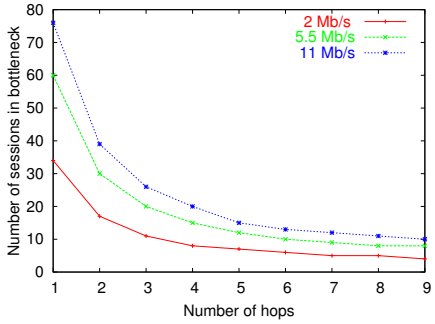


Figure 5.21: Effective ad hoc horizon: The maximal number of simultaneously active TCP sessions in the bottleneck L-region. MSS=600B.

B obtained through formula (5.4). The corresponding curves in the figure indicate the effective ad hoc horizon meaningful for the control plane of our ingress throttling scheme. Here is an example of how the control plane should interpret the graph. Assume the routing protocol establish a 4 hops path for a data session with $MSS = 600$ B and reports the transmission rate on the path 11Mb/s. If the load distribution algorithm assigns the C-load share for this flow larger than 0.4 the control plane decides to commence the data transmission otherwise the application would be suggested to try the path establishment later and the session is abandoned.

What can we say about the network scale at the edge of the effective ad hoc horizon? Let us display the horizon differently. Now we express the minimally acceptable load share in terms of maximal number of competing connections inside an L-region where this share was assigned. The graph in Figure 5.21 show the number of connections crossing h hops for each physical layer transmission rate beyond which the end-to-end throughput is below the minimally acceptable threshold.

The graph should be read as follows. For instance, assume that the maximally possible path in an ad hoc network is 4 hops. If we have at least one connection sending the data over the maximal number of hops with the IEEE 802.11 data rate

2 Mb/s, the number of competing connections in any one hop neighborhood along the path of this connection with arbitrary other characteristics should be less than 10. Otherwise the resulting throughput of the four hop flow will be below the acceptable threshold Thr_{min}.

This interpretation of the *effective* ad hoc horizon allows us to draw an important conclusion in the scope of the scalability studies of the IEEE 802.11 based MANETs. The effective ad hoc horizon should serve as a reference for the construction of network topologies and the choice of traffic settings in the experiments intended to show scalability of one or another protocol or solution.

Consider the following example. We create a scenario which potentially allow communications over nine hops and formation of the common bottleneck L-region. This corresponds, for example, to a stripe-shaped area with the length equal to 1200 m and the width equals 100 m. In this area it is possible to form a nine hop *association* with internode distance 130 m. Assume we create 20 TCP sessions in this region and at least one session follow a path of nine hops. Assume also that this session uses $MSS = 600$ B and the physical layer transmission rate is equal to 2 Mb/s for all nodes of its association. The maximally achievable throughput of a single nine hop TCP flow in isolation as we measured in our previous experiments is 144 kb/s. We can calculate the expected throughput for our nine hop session from formula (5.3). The resulting value is only 7.2 kb/s; This is too low throughput for even a minimal user's expectation. Our concept of the *effective* ad hoc horizon suggests that in this scenario we should not generate more than four TCP sessions to achieve the meaningful expected throughput of 30 kb/s.

5.3.2 Further research directions

Having outlined the major concepts of MANETs fairness framework and mechanisms of the ingress rate control, we highlight in this section some open questions and problems for further investigation.

A. Formal estimation of the maximally achievable TCP throughput in MANETs

In the first place we suggest that further developments of the proposed architecture should be directed on formal analysis of the maximally achievable throughput of single TCP session. A few papers attempt to formally model a single TCP flow over multi-hop wireless network. In [40] the authors formally characterize the throughput of a single multi-hop TCP connection in wireless network. The work describes an approach to model TCP transmissions over IEEE 802.11 network with enabled RTS/CTS exchange as an embedded Markov chain. Although simulations show good accuracy of the model, its extension for the case without RTS/CTS handshake and generalization to the total network load generated by the connection is yet to be developed.

B. Excess capacity and utilization

In our scheme the particular end-to-end data session will be guaranteed a share of the *available* network capacity for its entire duration considering its worst case communication, namely when it always has data to transmit e.g. long file transfer. If this is the case our scheme provides both perfect fairness and good network utilization. However, in reality we will also have a number of short-living communications and flows with a number of short transmission bursts e.g. web browsing. In this case the share of capacity will not be fully utilized. When several connections originate from the same source, the leftover capacity can be reused by the active flows, otherwise the network will be underutilized until the route for the inactive connection will be removed. Therefore it is essential to have an effective dynamic mechanism to update the ingress nodes with the path density information.

C. Wireless stub and transit networks

So far we considered a pure MANET scenario where we looked at TCP flows starting and terminating inside the wireless network. We need to extend our study to mesh networks where TCP flows (i) start in MANET and end in fixed network,

(ii) start in fixed network and end in MANET or (iii) use MANET clouds as transit networks.

The first case corresponds to the normal functionality of the described solution. In the last two cases our functionality will naturally be added to the ingress gateways of MANET. While in case (ii) the ingress node should place packets of distinct connections in a separate queue, in case (iii) queuing should be done on an ingress-egress aggregate basis.

5.4 Summary

In this chapter we presented a full scale experimental evaluation of the *space-load* fairness framework. We discussed the implications of our solution in the scope of scalability studies of MANETs and outlined future research directions. We also compared our solution to the related "space-air-time" framework and highlighted the functional and implementation differences between the two approaches.

We showed with simulations as well as with real-world experiments full compliance of the *ingress throttling* scheme to our *space-load* fairness model. The major result of this chapter is that by limiting the transmission rate at sources of TCP flows according to the derived rate limiting formula, TCP capture does not occur and almost perfect fairness is achieved for all involved flows.

The important implication of our *space-load* fairness framework is that we can make a quantitative prediction of the achievable TCP throughput for different experimental scenarios. The concept of the *effective* ad hoc horizon suggests a reference for the construction of network topologies and the choice of traffic settings in the experiments intended to show scalability of one or another protocol or solution.

We continue studies of TCP over the IEEE 802.11 based MANETs by considering the effect of the traffic from ad hoc routing protocols on the improved TCP performance. For this we devote our next chapter.

Chapter 6

Space-load fairness in presence of routing traffic

In Chapter 4 we proposed an approach for throttling the output transmission rate at ingress nodes of TCP connections. We showed how dramatically the fairness between competing TCP connections and the total TCP throughput in the network can be increased if the source nodes become aware about the presence of the competitors along the path and limit their output rate according to this information. Leaving the treatment of how to disseminate such information in MANETs to Chapter 7, in this chapter we continue to investigate the reasons for TCP unfairness in multihop wireless networks and add another important component of data communications in MANETs – the routing protocol.

Obviously the traffic generated by a routing protocol will affect the performance of the improved TCP + IEEE 802.11 MAC pair. We predict that the uncontrolled broadcast transmissions that is present in all popular routing schemes for MANETs imposes a limit on network size and number of sustained TCP sessions which we call the "routing ad hoc horizon".

We develop this thread as follows. In Section 6.2 we discuss the co-existence issues of our ingress throttling approach and the routing traffic. Our main observations follow in Section 6.3. There we develop an approach to analyze the impact of the load created by routing traffic on TCP sessions. The important result of this

section is that we are able to identify the *admissible operation range* of MANETs in terms of the network size and the number of simultaneously active TCP sessions where routing traffic does not become a reason for the unfairness of TCP communications. In Section 6.4 we summarize the material of this chapter.

6.1 Considered routing protocols

For the analysis of the effect of ad hoc routing on TCP performance we choose AODV as the most popular reactive routing scheme as well as LUNAR. For the experiments we use AODV-UU [74], the stable and RFC compliant implementation of AODV from Uppsala University. The stable implementation of LUNAR is available at [87].

Two variants of AODV protocol are considered. In the first variant the HELLO mechanism is enabled to maintain the connectivity between neighbors, further on we refer this variant of AODV as to AODV-HELLO. In the second AODV variant, further referred as to AODV-LL, the route maintenance is done by means of the explicit link layer feedback. In the former case the loss of the connectivity between nodes forwarding traffic of a specific connection is detected by three missing HELLO messages; in this case the route maintenance procedure is invoked and the problem is signaled back to corresponding sources. In AODV-LL a packet loss during the transmission is detected by the link layer; the problem is then immediately reported to the routing engine, which in turn invokes the route maintenance operations.

In the case of LUNAR we use the standard settings as described in [64] and the opportunistic LUNAR. In the standard LUNAR the entire path for each connection is rebuild every three seconds. In the opportunistic LUNAR we modified the route refreshing procedure as we describe later on in the text. Overall, the two variants of AODV and the two variants of LUNAR protocol produce four distinct patterns of routing traffic in the networks. As we shall see later in this chapter they affect TCP performance in MANETs differently. The methodology for the analysis of the impact of the routing traffic on TCP performance presented in this chapter is certainly applicable to other ad hoc routing scheme. However, we leave the analysis of other routing protocols to future work.

6.2 Ingress rate limit and routing traffic

The first issue that we address in this chapter is the co-existence of our ingress throttling scheme and the routing traffic. The ingress throttling scheme is a dynamic capacity allocation scheme which ensures that all competing flows do not use more than their fair share of the load inside the bottleneck L-region and altogether do not exceed the boundary load of the L-region. Adding routing protocol activities to the network, some part of the available capacity will obviously be consumed by the routing traffic. The reasonable question in this respect: How should we account for this additional traffic load in our throttling scheme? Before answering this question, we make observations on the structure and properties of the routing traffic.

6.2.1 Structure and properties of the routing traffic

The traffic generated by a reactive ad-hoc routing protocol is formed by two types of transmissions: broadcast and unicast. The broadcast transmissions happen during route establishment and route maintenance phases. As we already discussed in Section 1.4 all existing reactive routing schemes for ad-hoc networks utilize flooding as the dissemination mechanism for route request messages (RREQs). Some protocols, like AODV, in addition to the initial flooding, deploy periodic exchange of broadcast HELLO messages to observe the connectivity between neighbors. The unicast transmission is used mainly to transfer route reply messages (RREP) and error messages.

As it is observed in the literature, e.g. [31, 66], the main burden of the routing traffic comes from broadcast (re-)transmissions. The contribution of unicast routing messages to the routing load is minor in comparison to the former. Considering the process for creating and maintaining connectivity for a single end-to-end connection, the broadcast routing transmissions have the following straightforward properties:

1. Propagation of route request messages using flooding (re-broadcasting) is not coordinated by the source node.

2. The duration of a broadcast burst depends on the network size.

3. In general, routing message transmission events are independent from the actual data transmission pattern.

The first and the second properties come directly from the fact that a route request wave is initiated by transmission of a single packet from the source node. Then this packet is copied and independently re-transmitted by all nodes that participate in the routing protocol. As a result the propagation of a distinct RREQ message depends only on the number of nodes which re-broadcast its copy.

The third property can be illustrated by an example referring to the LUNAR routing scheme. In LUNAR the emission of route request messages happens every three seconds as long as the particular connection is active. This three seconds interval is independent from the actual transmission rate of the data traffic unless the interval between two subsequent packets is larger than three seconds. In this case the protocol assumes that the connection does not exist and terminates the activity. In AODV the broadcast transmissions can be initiated by route request messages as well as HELLO messages. The frequency of emission of HELLO messages is periodic with one second interval. Hence, broadcast messages depend also for AODV on the network state and not on the actual activity of the data sessions.

6.2.2 Routing load

In our ingress throttling scheme it would be desirable to account for the additional load created by the activities of a routing protocol for calculating the rate throttling formula (Equation 4.5). In our research we considered two different approaches for doing this task. However, we discarded them as unrealistic from the implementation's point of view. Below we present these unsuited approaches and highlight their problems.

A. Incorporating the routing load into the ingress throttling scheme

The first approach for accounting the competing routing load in the computation of the throttling formula is to estimate the rate of the routing traffic in the network and subtract this value from the ingress rate limit. This approach is difficult to

implement mainly due to the following two reasons. Firstly, as we will show in the next section, estimation of the routing load is a difficult task by itself. Secondly, our ingress throttling scheme assumes that the sources that generate the competing traffic cooperate with each other in the sense that all of them reduce the transmission rate. In the case of the routing traffic we do not have a specific source which would be able and in charge of controlling its rate. As we showed above with Properties 1 and 3, every node participating in a routing protocol may be the source of a control routing message. In this respect we see it as a very difficult task to create a mechanism, which in a distributed manner would control the transmission process of the routing messages from multiple nodes. Without such a mechanism, shaping the TCP flows taking into account the routing load does not make sense since the uncontrolled routing transmissions will violate the cooperation principle that underlies our scheme.

B. Adjusting the behavior of sources to uncontrolled routing transmissions

The second approach to account for the routing traffic in the rate limit computation is to take the uncontrolled transmissions of routing messages as given and to appropriately adjust the behavior of the data sessions at sources. In other words to consider the routing traffic as *unavoidable noise* and to adjust the ingress throttling scheme so to minimize the effect of routing on data communications. We consider this approach as unrealistic because it is difficult to determine the exact time of routing transmissions and their duration. The first difficulty comes from the independence of routing transmissions with respect to the actual data traffic pattern (see Property 3 above). As we will show in the next section we cannot determine the time with acceptable accuracy at which the routing transmissions will cause the worst effect on the data transmission. Secondly, the duration of the routing activity depends on the network size, namely, the number of nodes that potentially can send the competing broadcast traffic. In order to obtain this information from the network we need to introduce an additional protocol which gathers this information at run time. While potentially this is feasible, in the light of the first reason we decided to do not continue with the development of such a scheme.

6.2.3 Evaluation of the effect of routing protocols on unmodified incress throttling scheme

Accounting for the above observations, we decided to follow a different way. We will evaluate the effect of different routing traffic patterns on TCP communications assuming that the sources cannot adjust their behavior to the routing activities in a controlled manner. We assume that all TCP sources compute the ingress rate limit given by Equation (4.5) as if the routing traffic does not exist.

The excess routing traffic will certainly cause packet losses in addition to those caused by collisions between data packets themselves. Since different routing protocols produce unique traffic patterns, the number of such losses will be different for each routing protocol. We foresee that up to a certain level the routing-caused losses of the data packets will be tolerable for TCP connections. However, beyond that, these losses will put some TCP sessions in the slow start phase. As we discussed above the distributed nature of routing transmissions does not allow us to control the "rate" of the routing traffic in the network in the same way it is done by TCP. Therefore it would be wrong for TCP to react on routing-caused losses of the data segments as in the case of network congestion. As a result of such wrong invocation of the slow starts, the corresponding TCP flows become more sensitive to every additional packet loss. Therefore there is a possibility that TCP unfairness appears again.

We were able to extract the "fingerprints" of such effects for the routing protocols considered in this chapter. Our approach allows us to draw conclusions about a data communication "friendliness" of different routing traffic patterns and to determine the scale of MANETs where both the routing and data traffics can co-exist without significant degradation of the quality of the end-to-end data sessions.

Outline of our methodology

In order to evaluate the effect of different routing protocols on the quality of TCP flows we utilize two main properties of our ingress throttling solution:

1. The smooth progress of the shaped TCP flows.

2. The optimal utilization of the network when all competing TCP flows are

active.

We use the *unsmoothness* metric (see Section 5.1.1 B) to qualitatively assess the level of distortion to the data communications produced by a routing protocol. We then determine an admissible operation range of MANETs in terms of the number of simultaneously active TCP sessions and the number of network nodes where the interactions between the routing and data traffics do not seriously affect the fairness and the smoothness properties of the end-to-end data sessions. After that we estimate the load produced by the particular routing protocol depending on the number of nodes participating in the routing protocol.

6.3 RT-capture, RT-horizon and equivalent routing load

Performance evaluation of an ad hoc routing protocol is a complex process. In the context of our research we do not evaluate nor compare different routing schemes with respect to the efficiency of the path establishment process and the quality of the established routes. Rather, we seek a quantitative and qualitative assessment of the effect of the routing protocol activity on the data traffic. When we started to search for appropriate performance metrics we faced certain difficulties. In the literature related to the performance analysis of routing schemes [11, 20, 42, 19] we found very few *quantitative* performance metrics. The informational RFC 2051 [17] summarizes these *metrics* as follows:

1. Average number of data bits transmitted per data bit delivered, also referred as to "Packet delivery ratio". This metric is normally interpreted as a measure of the quality of data delivery within the network.

2. Average number of control routing bits transmitted per data bit delivered, sometimes referred as to "Normalized routing load". This metric measures the overhead produced by transmission of routing messages.

These two metrics can be used mainly for two purposes. Firstly, to quantify an average load produced by routing traffic and secondly, to assess the effect of routing on UDP based communications for which the packet loss rate is an illustrative performance characteristic. However, in the case of TCP communications,

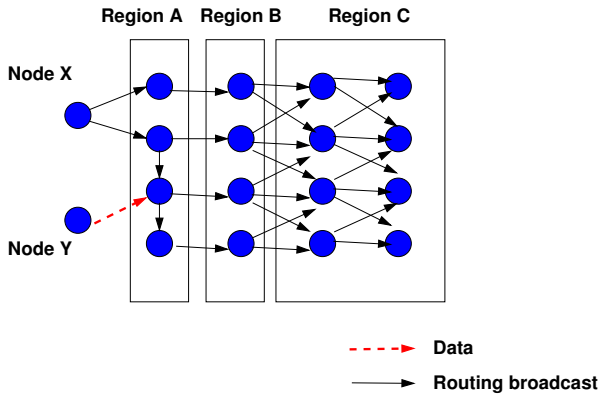


Figure 6.1: Illustrative example of inability of conventional metrics to capture the spatial effect of routing on data transmission. The multiple arrows from every node in the marked regions reflect the broadcast nature of a single packet transmission.

none of them is able to characterize the overall quality of the ongoing sessions. In the following subsection we present a reasoning for why these metrics do not allow to correctly quantify the impact of the routing traffic on TCP in MANETs.

6.3.1 Why current performance metrics are not informative

As we discussed in Chapter 2, individual throughput of different flows might be the same, although some of the flows may suffer from long no-progress intervals. Therefore, knowing the proportion of the number of delivered TCP data segments to the number of emitted data segments does not reveal the degree of service degradation for the particular flow.

Let us have a closer look at the "normalized routing load" metric. This metrics attempts to relate the transmission events from the routing protocol to the amount of delivered user's traffic. Consider a scenario depicted in Figure 6.1 and assume that the propagation of a routing message issued by Node X causes a loss of one data packet issued by Node Y. During the measurement period we observe 13

transmissions of routing messages in the entire network. The question is: How relevant is the information that 13 routing packets caused the loss of one data packet?

In fact only transmissions in Region B may cause the loss of our data packet if these transmission events coincide in time. If the transmission from Node Y happens at the same time as routing transmissions in Region A, Node Y will refrain from the transmission because it will sense the medium as busy. The nodes in Region C are located three hops away from Node Y, therefore transmissions from this region would not interfere with our data packet. Finally, only transmissions in Region B will cause the loss of the data packet because of the hidden terminal effect. Therefore the actual effect of routing traffic on the data transmission in our example is four routing packets per one lost data packet. The major conclusion from this simple observation is that the "normalized routing load" metric does not reflect the *spatial* effect from the routing activity on the quality of data sessions.

6.3.2 The "unsmoothness-unfairness" metric

In our opinion it is very hard if not impossible at all to capture the spatial effect of the routing traffic described above. In a general network for a given packet transmission (or a set of subsequent transmissions) we need to find a correspondence between the transmissions of routing messages in different geographical regions of the network and losses of the data packets. We conjecture that constructing such a metric based on direct observation of transmission events for data and routing packets is problematic. We found a way to quantify the effect of the routing on the quality of TCP sessions indirectly, which we describe below.

The major property of our ingress throttling scheme is that by shaping every competing flow in the network according to the throttling formula we eliminate the possibility for one flow to capture the capacity from another flow. The progress of a specific TCP session during the net traversal is smooth and free from interruptions. This property is illustrated in Figure 6.3. The figure shows the sequence number progress of corresponding receivers for four three-hops TCP connections in the topology depicted in Figure 6.2. Moreover, our rate throttling scheme is enabled and we assume static routes to corresponding destinations.

We can use this property of our scheme as an *ideal* behavior of TCP sessions.

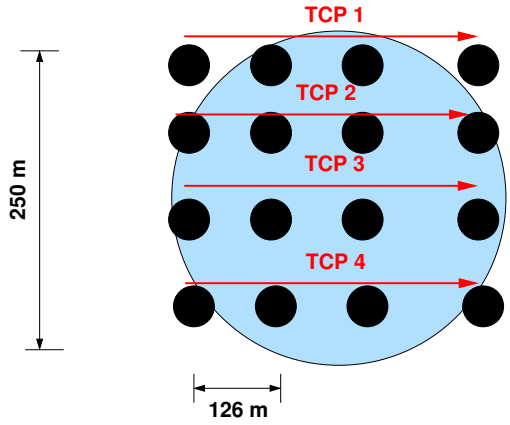


Figure 6.2: Network topology for illustration of the *smoothness* property of the ingress throttling scheme.

Certainly, adding routing traffic to the network, the resulting sequence number curves for each flow will deviate from the *ideal*. The degree of such deviation will reflect the impact of the routing activity in the network on the quality of TCP sessions. In order to capture the degree of such deviation we use the *unsmoothness* metric introduced in Chapter 4. Recall that the *unsmoothness* metric (5.2) is always larger than zero. We say that the quality of a TCP flow is acceptable if $Unsmoothness \leq 1$.

A. The combination of the *unsmoothness* and *unfairness* metrics

The analysis of TCP communications is a complex process. As we have shown in Chapter 2, considering only fairness or only throughput in isolation does not reveal the whole performance picture. Our major observation from that chapter was that while *unfairness* and throughput metrics might look fine, the progress of each TCP session might be stammering. In other words a TCP session might suffer

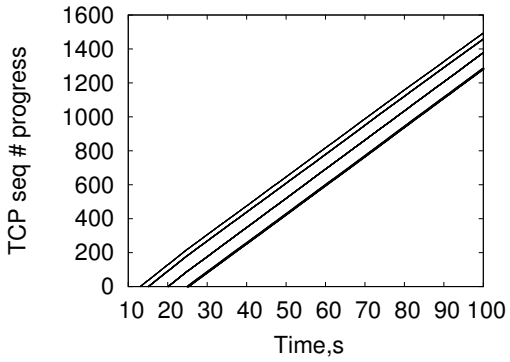


Figure 6.3: Smooth progress of TCP flows under ingress throttling.

from long no-progress intervals. Therefore it is essential to consider a combination of metrics for the analysis of TCP performance. In particular we are seeking for such a combination that reflects: (1) the quality of the progress of each session, (2) the end-to-end throughput achieved by each session and (3) the fairness of communications of multiple TCP sessions. As for the last characteristic we can use any of the "unfairness" metrics which we utilized so far, e.g. the complement of the Jain unfairness index (Equation 2.1) or the normalized distance from optimum (Equation 2.2).

To assess the first two TCP performance characteristics we will use the *unsmoothness* metric, which we designed exactly for the analysis of the quality of a TCP session progress. As for the TCP throughput, in this chapter we are not interested in the actual value of this metric; rather, we are looking for its qualitative measure that is able to say whether the throughput was sufficiently good or not. The *unsmoothness* metric provides us with this information. As we showed in Chapter 5, our ingress throttling scheme results in the increase of the individual *and* the total TCP throughputs in the network. Figure 6.3 also illustrates that in this case the arrival process of TCP data segments is smooth. Therefore, by analyzing the *unsmoothness* metric we make the needed qualitative assessment of the end-to-end TCP throughput: The closer to zero this metric is, the better is the individual and total throughput in the network. In the rest of this chapter we will

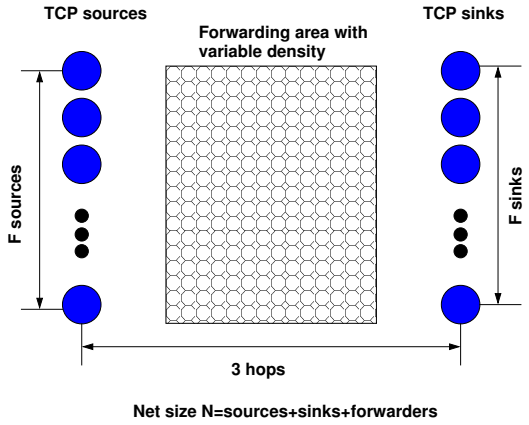


Figure 6.4: Topology for testing the effect of different routing traffic patterns on TCP communications. Perfect connectivity is assured.

analyze the TCP performance using the combined *"unsmoothness-unfairness"* metric.

6.3.3 The effect of different routing protocols on the ad-hoc horizon

At the center of our experiments is a prediction that beyond a certain network size the broadcast routing traffic will be able to cause serious TCP unfairness problem. With our ingress throttling scheme enabled, the only reason for the unfairness problem to appear again would be the destructive effect of the routing traffic which we cannot control. We will refer to this phenomenon as the "routing capture" (or "RT-capture") further on.

The main goal of our experiments is to determine the *admissible operation range* (AOR) of MANETs in terms of the number of simultaneously active flows and the number of nodes in the network beyond which the routing "noise" will annihilate all improvements that can be achieved by cross-layer optimization. We call the limit of the AOR the *"routing ad hoc horizon"* (or simply *RT-horizon*).

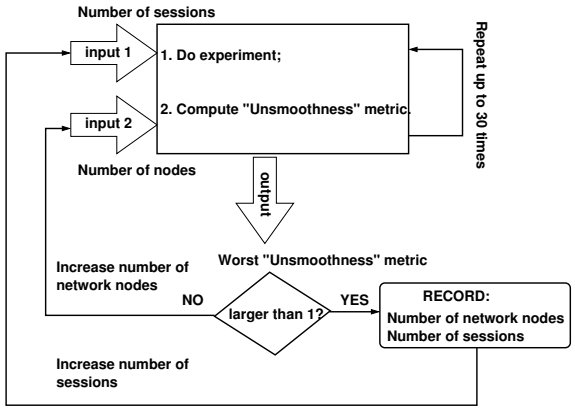


Figure 6.5: Structure of experiments for determining the RT ad hoc horizon.

A. Description of RT-horizon experiments

For the experiments we use the topology in Figure 6.4. In this topology we are able to vary two parameters: The number of competing TCP sessions and the number of nodes in the network. By increasing the number of simultaneously active sessions we increase the intensity of the routing traffic. By increasing the number of nodes in the forwarding region we increase the duration of the broadcast bursts, since more nodes are involved in the (re-) broadcasting process.

In order to determine the "RT-horizon" we performed a series of simulation based experiments, whose structure is graphically shown in Figure 6.5. In all simulations we used continuous FTP traffic from all sources. The maximum segment size is set to 600 B. The congestion window of TCP $CWND = 32$ MSS. The 802.11 data transmission rate at the physical layer is 2 Mb/s, RTS/CTS handshake is disabled. In all experiments our *ingress throttling* is enabled.

Every experiment is repeated 30 times with randomly chosen seed for the random numbers generator of ns-2 in each simulation. In each run we measured the *unsmoothness* and Jain unfairness index. If after 30 runs with the same number of simultaneously active flows and the number of nodes in the forwarding area the worst unsmoothness metric is less than 1 and unfairness index is less than 10%

we increase the number of the forwarding nodes and repeat the experiment. We continue to increase the number of nodes in the forwarding area until the largest unsmoothness becomes larger than 1 or the unfairness becomes larger than 10%. At the end of all experiments we obtain a list of pairs (number of connections / number of network nodes) beyond which either unsmoothness or unfairness metrics are unacceptable for an end user.

During the experiments with AODV-HELLO the best measured values of the *unsmoothness* metric is between one and two. For this variant of AODV we determine the *"weak RT-horizon"* as it is described below. We present the results from this experiment for three hop MANETs in the following paragraphs.

B. Analysis of the results

The results for the two variants of AODV protocol are shown in Figure 6.6*a* and for LUNAR in Figure 6.6*b*. The shaded areas in the figures show the admissible operation range (number of active sessions / number of nodes in the network) where the *unsmoothness-unfairness* properties of all competing TCP flows is acceptable in the sense discussed in the previous subsection. Outside the shaded area both *unsmoothness* and *unfairness* metrics become unacceptable for at least one end user.

The left slopes of the AOR show the minimal network configurations where the particular number of distinct connections is possible. For example, four distinct three hops flows are possible when we have one source node, four destinations and two forwarding nodes, that is in total 7 nodes in the network. The right border of the corresponding AORs represents larger network configurations for the particular number of TCP sessions. For example, in the case where AODV with the link layer feedback is used, four connections with the acceptable *unsmoothness* and *unfairness* metrics can exist in a network with four distinct sources, four destinations and 22 forwarding nodes, that is 30 network nodes in total.

AODV-LL

As it is visible from the graph in Figure 6.6 the least effect on the *ideal* TCP behavior is introduced by the routing traffic pattern of AODV with the link layer feedback. This is because of the error-driven invocations of broadcast transmis-

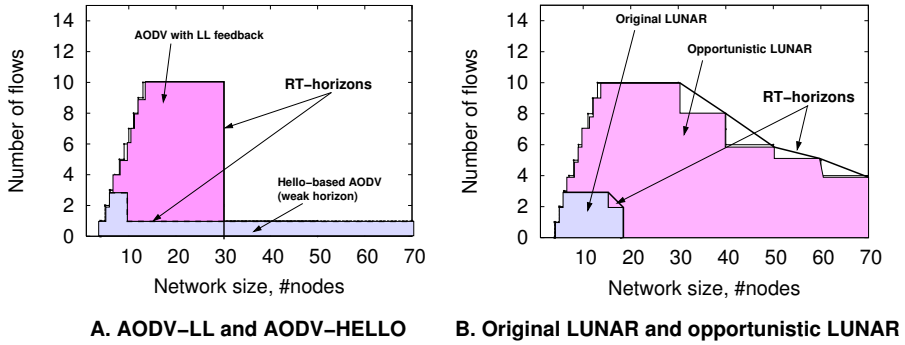


Figure 6.6: Admissible operation range and RT-horizons for MANETs. Ingress throttling is enabled.

sions. In AODV-LL the broadcast activities are initiated as a reaction to packet losses reported by the link layer to the AODV engine. In the case of a packet loss the protocol assumes that the connectivity to the corresponding neighbor was lost and initiates the route recovery procedure. Since our throttling scheme is deployed at sources, the collisions between data packets are not frequent since all flows are shaped to fit their fair share of the bottleneck's load. The only reason why data packets are lost in this case is the routing activity. As we observe in networks of up to 30 nodes, the broadcast traffic does not introduce enough overhead to force TCP flows into the "wrong" slow start phase. What happens here is that TCP itself reduces the rate below the throttling limit, by this reducing further possibilities of packet losses in the network. However, beyond 30 nodes the broadcast bursts caused by every lost packet are long enough to cause the invocation of the slow start phase at a TCP sender; by this resulting in a stammering nature of the TCP flow progress and worse fairness figures.

AODV-HELLO

The most unstable effect on the quality of TCP sessions is introduced by the traffic patterns produced by AODV-HELLO. In all experiments the *unsmoothness* of the competing TCP flows was between one and two at best. Figure 6.7 shows

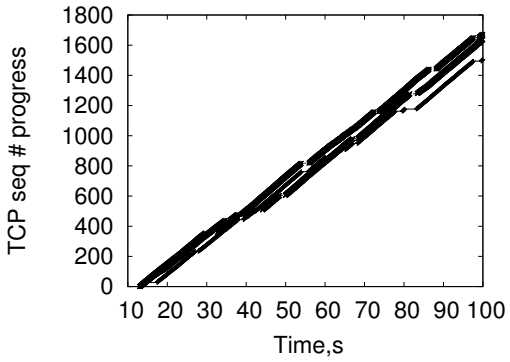


Figure 6.7: Best smoothness under HELLO-based AODV.

a typical arrival process of the data segments for three competing TCP flows in a small network of 10 nodes in total. As we can observe from the graph, on average the interruptions in TCP progress are not that dramatic. This is due to the small size of HELLO messages. However, their frequent and independent emissions from multiple nodes does not allow any of the three TCP sessions to progress smoothly through the network. Even though we could not observe a perfectly smooth flows as we did in the case of AODV-LL, we decided to delimit the admissible operation range where $0 < Unsmoothness < 2$ and the unfairness index (Equation 2.1) is less than 10%. We labeled this horizon for AODV-HELLO as "weak RT-horizon" in Figure 6.6*a* .

LUNAR

As for the effect of the routing traffic pattern generated by LUNAR, we observe that it is very similar to that of AODV-HELLO. The smaller shaded area in Figure 6.6*b* shows the admissible operation range for the original LUNAR with forced three seconds complete route re-discoveries. The difference is that the horizon with LUNAR is stable, meaning that the *unsmoothness* metric for all flows within the AOR is strictly less than 1 and the fairness is close to perfect. The stability of LUNAR's horizon can be explained by less frequent initiations of the broadcast traffic than in the case of AODV-HELLO. Certain similarities in

the right border of the AOR are explained by the fact that LUNAR's three second forced route rediscovery interval is chosen with the reference to the HELLO interval in AODV: It corresponds to two HELLO rounds which are needed by AODV to determine a change of the route. However in the case of LUNAR we have a strict right border. This can be explained by the difference in the nature of broadcast patterns generated by the two protocols. If in the case of AODV-HELLO we have short frequent one hop broadcasts, in the case of LUNAR we have less frequent but more massive flooding waves, which with increasing the number of nodes in the network result in appearance of the *routing capture*.

Opportunistic LUNAR

As can be seen in Figure 6.6*b* original LUNAR has a narrow admissible operation range. As an experiment we changed the routing traffic patterns generated by LUNAR to one which closely resembles the error-driven pattern of AODV-LL. To do this we disabled the forced three second complete route rediscovery mechanism. Instead, we retain a route as long as there are packets arriving to the forwarding engine: With every new data packet we shift the route timeout three seconds into the future. This modification allowed us to create an "opportunistic" version of LUNAR.

This change allowed us to significantly extend the horizon. Moreover we achieved even better characteristics in comparison to AODV-LL. Now the right edge of the admissible operation range stretches to bigger networks. This is because we do not interpret every loss of a packet as an indication of the route breakage as it is the case with AODV-LL. Instead, we react on invocations of slow starts in TCP which are less frequent events, given that our ingress throttling scheme is deployed in all sources. By doing this we significantly decrease the frequency of broadcast bursts.

6.3.4 Equivalent routing load

In the beginning of the current section we presented our reasoning for why the estimation of the routing load is complex and not an obvious task. Recapitulating, this is due to the difficulties to capture the spatial effect of the broadcast traffic on a particular data session. Because of this the averaged metrics for routing load

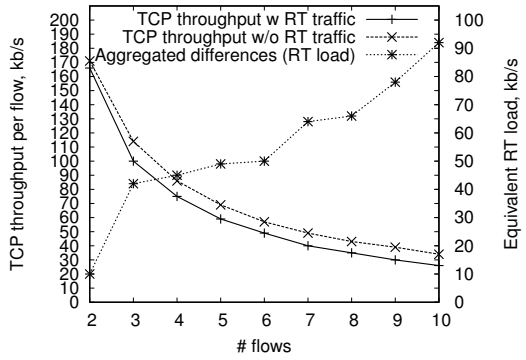


Figure 6.8: Equivalent routing load and TCP throughput per flow at the maximum network size (AODV with the link layer feedback, 30 nodes, see Figure 6.6*a*).

do not reflect the spatially distinct effects of the routing traffic on the ongoing TCP communications. In the previous subsection we found a way to quantify the stable interoperation of reactive routing protocols and TCP communications. The *RT-horizon* represents the critical network size where TCP flows maximally use the network capacity which is not consumed by transmission of routing messages. Beyond this horizon, TCP flows begin to suffer from the capture problem, now caused by the routing activity. The direct consequence of the *RT-capture* is that a part of the network capacity remains unused during the silence periods of TCP. The important property of the stability of communications within the *RT-horizon* is that we can indirectly estimate the routing load produced by one or another protocol. We define an *"equivalent routing load"* metric as follows.

Assuming that our ingress throttling is deployed in all sources and that we measure the total TCP throughput in the network without routing ($Thr^{tot}_{norouting}$), in the case where all competing flows are active the network is fully utilized. Now we measure the total TCP throughput on the right border of the admissible operation range ($Thr^{tot}_{withrouting}$). In this case the competing TCP flows maximally use the capacity that is left from the routing traffic. The difference $Thr^{tot}_{norouting} - Thr^{tot}_{withrouting}$ reveals the reduction in the throughput in bits per second due to the routing activity. Figure 6.8 and Table 6.1 show the dynamics of

Protocol	AODV-HELLO		original LUNAR	
Number of flows	2	3	2	3
Net size	9	9	18	15
Equivalent routing load, kb/s	80	82.2	40	39.4
TCP throughput per flow, kb/s	131	87	150	102

Table 6.1: Equivalent routing load of AODV-HELLO and the original LUNAR at the edge of admissible operation range.

this metric with increasing the number of competing flows when different routing protocols are used.

When we compare the equivalent routing loads of AODV-LL (Figure 6.8), AODV-HELLO and the original LUNAR (Table 6.1) for commonly used numbers of flows, we immediately see the reason for the very small AOR of the later two protocols. The equivalent routing load in the case of AODV-HELLO in nine nodes network and two flows is 8 times larger than the corresponding load of AODV-LL in 30 nodes networks. The corresponding difference between the original LUNAR and AODV-LL is respectively 4 times.

As for the modified opportunistic LUNAR protocol, its maximum "equivalent routing load" was in the order of 10 kb/s (not shown in the table). This is because the routing activity in this case happen only during the path establishment of each flow. After that the ingress throttling scheme prevents invocation of slow starts in the stable network as we have in our experiments. Therefore no further routing traffic is involved after all sessions were successfully established until the end of simulations.

6.4 Summary

In this chapter we analyzed the effect of different routing traffic patterns on the performance of TCP during stable operations of MANETs. We found that quantifying the routing load even in stable networks without mobility is difficult. We highlighted the fact that the current quantitative and qualitative performance met-

rics for the evaluation of routing protocols do not reflect their actual effect on TCP in MANETs.

We suggested to use the main property of our ingress throttling scheme for an indirect measurement of the routing load and quantification of the routing effect on TCP performance. Using this technique we were able to show that the routing traffic itself can be a reason for TCP unfairness in MANETs; We call this phenomenon the *RT-capture*.

We analyzed two routing protocols with four distinct routing traffic patterns and were able to identify the operational region of MANETs (in terms of the number of simultaneously active TCP sessions and the number of network nodes) beyond which the traffic generated by the routing protocol significantly degrades the performance of TCP. We call this region the *"admissible operation range"* of MANETs.

Our major conclusion regarding the effect of routing traffic patterns on TCP communications is that periodic, non error-driven broadcasts of even short messages is harmful for data communications and leads to narrowing the operational region of MANETs.

Chapter 7

A path density protocol for MANETs

In Chapter 4 we presented an adaptive distributed capacity allocation scheme for multi-hop wireless networks, which we briefly summarize for convenience here. After that we introduce in Section 7.1 the problem of path density gathering and describe the major design options for the path density protocol. We show how our protocol was piggybacked onto an existing routing protocol. After that in Section 7.2 we report on performance results of the path density protocol obtained using simulations and real-world measurements. We discuss open issues in Section 7.3. The summary of the presented material follows in Section 7.4.

Recapitulation of the ingress throttling scheme

The ingress throttling scheme enforces the max-min fairness framework for mobile ad hoc networks. We showed in Chapter 5 that by throttling the output rate at ingress nodes we achieve both an increase in the total TCP throughput and almost perfect fairness. We compute the limit on the output rate at ingress nodes of TCP session by the following formula:

$$r_i^{ingress} \leq Thr_{max}(h, MSS, TX_{802.11}) \cdot \phi_i^{bottleneck}. \tag{7.1}$$

In the above formula $Thr_{max}(h, MSS, TX_{802.11})$ is the maximal throughput of a single TCP flow traversing h hops, transmitting data segments of size MSS, while $TX_{802.11}$ is the used transmission rate on the physical layer along the path of the flow. The maximal throughput can be experimentally or formally estimated in advance for all possible combinations of the input parameters. The parameter $\phi_i^{bottleneck}$ is the C-load share of TCP session i in its bottleneck L-region.

In order to compute the delay parameter Δ for the scheduler at the interface queue and assuming an a priori availability of the estimates for the maximal TCP throughput $Thr_{max}(h, MSS, TX_{802.11})$ we need an indication of which value from the available estimates to use as well as the value of the C-load share in the bottleneck of the particular connection.

As for the first issue, out of the three input parameters, the value of MSS can be detected locally for every packet going to the scheduler. The available transmission rates at the physical layer on the path, $TX_{802.11}$, and the path length h should be provided by the network. In the case of a reactive ad hoc routing protocol, the later parameter is easily obtainable from the route reply (RREP) message received by the source. As for the former parameter, the routing protocol should be equipped with a functionality to detect the available transmission rate on the path. In this chapter we assume that all communications in MANETs happen with equal IEEE 802.11 physical layer transmission rate.

Finally, the C-load share distribution algorithm should be implemented in the network in order to provide the sources with the $\phi_i^{bottleneck}$ parameter. In this chapter we present an implementation of the algorithm for C-load share distribution for a special class of MANETs where all the competing TCP connections share at least one common bottleneck L-region. Recall from Section 4.2.6 that in this case the computation of the bottleneck C-load shares $\phi_i^{bottleneck}$ becomes the task of finding the *path density* for every competing connection. Now with $\phi_i^{bottleneck} = 1/\rho_{max}$ the resulting formula for the ingress rate limit is then transformed as follows.

$$r_i^{ingress} \leq \frac{Thr_{max}(h, MSS, TX_{802.11})}{\rho_{max}}. \tag{7.2}$$

In the above formula ρ_{max} is the maximal density of the competing connec-

tions among all existing L-regions. We refer this parameter as to the *path density* for the particular connection. Therefore, the overall goal of our implementation is to gather the density information along the path of a particular connection and report this value to the source so that it can locally compute Equation (7.2).

7.1 Measuring the path density

The concept of a path density (PD) implicitly exists in the major quality of service (QoS) architectures developed for the wireline Internet. For instance, in services which require a before hand reservation of network resources, this information can be extracted from the state (either aggregated or per-flow) established by a resource reservation protocol like RSVP. In the simplest case we can count the number of entries identifying the ongoing flows in every router on the path of the particular session and report this number to the source. In MANETs, however, obtaining the path density information is difficult due to frequent topology changes and the specifics of the wireless transmission medium: In MANETs, connections might not share common nodes but still do compete with each other inside L-regions.

7.1.1 Problem statement

How can one discover the number of connections competing for the transmission medium along a path of the particular session? The problem is illustrated in Figure 7.1. In the figure, Connection 1 is our sample connection which attempts to discover the *path density* along its path. The correct scheme should report five cross-connections plus the main Connection 1 itself. When any two flows partly share their forwarding paths (as is the case with flows one and four in Figure 7.1) the common nodes should treat them as different connections. However when two or more connections completely share the path from a source to a destination the forwarding nodes should treat this case as one end-to-end session. In the later case the source nodes should perform additional shaping actions as described in Chapter 4. The forwarding nodes of Connection 1 should also take care of distributing the known information about the ongoing connections in the corresponding one-

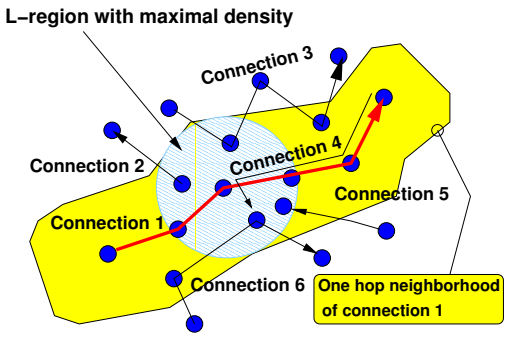


Figure 7.1: Counting competing connections along the path of the particular session.

hop L-region. In our example this is needed to introduce the presence of flows two and three to flows five and six and vice versa, since their forwarding nodes in general might be outside the communication regions of each other but still be within the common L-region.

7.1.2 General solution scheme

We developed two different approaches for gathering the path density information: A *stateless* route request driven and a *state-full* route reply driven scheme. Further on we will refer to the first scheme as SL-PDP (StateLess PDP) and to the second scheme as SF-PDP (StateFull PDP). In both approaches every node in the network maintains one state variable *ND* (the neighborhood density)[1]. This is the number of cross-connections in the neighborhood of two hops. With every new connection appearing in the neighborhood this value is incremented by one. The state is periodically aged and is decremented by the number of connections which were silent during this period.

[1]Considering a class of MANETs with at least one common L-region, having the state variable ND is sufficient to identify the density of the L-region.

The stateless approach does not keep a per-connection state in the forwarding nodes and utilizes the fact that a *route request* (RREQ) message indicates the desire of a node to communicate with another node. In this scheme every RREQ issued by a source advertises the presence of a connection to all nodes which receive its original or re-broadcasted copy. There are two major weaknesses of this scheme that we discovered when experimentally assessing the functional correctness. Firstly, SL-PDP also counts the failed connection setups as existing connections and maintains this information at least for the duration of the aging timer. Secondly, since flooding is used for dissemination of RREQs we cannot control the range of their distribution. As a result, the information about the presence of a connection is spread over more than two hops. We do not discuss further the functionality of SL-PDP here and refer to [55] for more information.

The statefull approach is free from these weaknesses since it counts only active connections and controls the range of information dissemination. The main idea of this scheme is that on reception of a *route reply* (RREP) message all involved nodes establish a state corresponding to the end-to-end session. A node delays the announcement of a connection to the neighborhood until it sees the first data packet from this connection, as only this event indicates that the connection is active. The announcement of active connections in one hop neighborhood is done every time a node sends or re-broadcasts a route request message. We now describe SF-PDP in more details.

7.1.3 Statefull path density gathering

In SF-PDP we identify the presence of a communication session between two nodes in the network by a source-destination pair (SD). In this scheme every node maintains a table of known SD pairs (*SD_table*). The entries in this table can be of four types: (a) Local active, (b) local inactive, (c) one-hop active and (d) two-hops active as explained in the following subsections. The ND state variable is the number of all "local active" and "non-local active" entries in the SD_table.

A. Adding "local inactive" SD entry

When a route request message for a particular connection successfully reaches the destination the RREP message is returned to the source. However, reception of the RREP message by an intermediate node does not indicate the success of the route establishment procedure since the message itself can be lost on the path to the source. SF-PDP treats the event of RREP reception as an indication of a possible connection through this node. Upon reception of a RREP a SD entry corresponding to this connection is inserted to the SD_table and is marked as inactive. The operation of SF-PDP during registration of a connection in the network is given by the following pseudo-code.

```
// On reception of
// rrep(S, D, forwardto_addr, hop_count, max_density)
Insert_SD(rrep.S, rrep.D);
// ND is unchanged;
Update_SD(rrep.S, rrep.D, LOCAL_INACTIVE);
// End-to-end density reporting (see Section 2.4)
if (ND > rrep.max_density) then
    SetRREP(rrep.max_density, ND);
```

B. Adding "local active" SD entry

The only definite indication of the active connection in any forwarding node is the event of receiving a first data packet. Therefore, when a local inactive SD record is created, the forwarding engine is instructed to catch the first packet in either direction of the connection. The SD entry becomes active only when the first data packet arrives to the node as reflected in the pseudo-code below.

```
// On reception of the first data packet pkt(S, D, DATA)
ND++;
Update_SD(pkt.S, pkt.D, LOCAL_ACTIVE);
```

C. Adding "non-local active" SD entry

The non-local active entries are those which identify connections ongoing in the neighborhood of two hops but not passing through this node. The information

about such connections is distributed with broadcast route request messages for any connection originated or re-broadcasted from any of the one-hop neighbors. Before issuing or re-broadcasting the route request message, SF-PDP examines the local SD_table and appends all "local active" and "one-hop active" SD pairs to a "competitor list". After that it piggybacks the constructed list to the RREQ message and transmits it. The following pseudo-code reflects the above operations.

```
// On issuing or re-broadcasting of
// rreq(S, D, replyto_addr, hop_count, competitor_list)
new_competitor_list = nil;
foreach (c in SD_table) do{
  if (c.location_type == LOCAL_ACTIVE)
  then
    add(new_competitor_list, c, ONEHOP_ACTIVE);
  elseif (c.location_type == ONEHOP_ACTIVE) then
    add(new_competitor_list, c, TWOHOP_ACTIVE);
}
SetRREQ(rreq.competitor_list, new_competitor_list);
```

When a route request message is received by a node (see the pseudo-code below), the SD entries from the RREQ's *competitor list* are inserted to the local SD_table and marked as "non-local active" of the corresponding location. After that SF-PDP removes the competitor list from the received RREQ message and appends its own list as described above.

```
// On reception of RREQ
foreach (n in rreq.competitor_list) do{
  if (!iselementof(SD_table, n)) then{
    Insert_SD(n.S, n.D);
    ND++;
  }
  Update_SD(n.S, n.D, n.location);
}
```

By replacing the *competitor list* with every retransmission of RREQ we do not allow the information about local connections to spread further than one hop.

D. Aging of SD entries

Every entry in SD_Table is assigned a timer. The duration of the timer, after which the entry will be removed and ND is decremented, is the same as the route expiration time.

7.1.4 End-to-end density reporting and smoothing

The overall goal of PDP is to discover and to deliver the maximum path density along a path to the corresponding source node. This information is used by the interface queue to configure the delay of emission of the locally generated data packets as described in Chapter 4.

A *max_density* field is added to the route reply message and can be modified by every node which receives and forwards the RREP message. When a forwarding node receives a route reply message, PDP examines the local ND state and compares its value to the *max_density* field in the RREP message. The higher value proceeds further to the source.

As we see, the path density state is refreshed with every route reply message. Therefore the frequency of its update depends on the frequency of route refreshes initiated by the source. Here we assume that route refreshes are periodically initiated. In the next subsection we discuss how this requirement can be combined with reactive ad hoc routing protocols.

Due to the high dynamics with which new flows might enter and leave the ad hoc network, the path density reported to the source may fluctuate in time. Therefore, in order to avoid reconfiguring the scheduler on the interface queue too often we allow sources to *smooth* the path density by computing a sliding average.

7.1.5 Integration in ad hoc routing protocols

Our primary goal for the implementation of PDP was to avoid introducing additional message exchanges to gather the path density information. This is because from the functional point of view the operations of PDP are very similar to operations of the route establishment phase. Implementing an additional stand-alone protocol for dissemination of PDP information would in the worst case double the capacity already consumed by a routing protocol. We decided to re-use the

existing mechanism of message dissemination of the LUNAR [64] ad hoc routing scheme. We also considered possibility of PDP integration in other reactive routing protocols, i.e AODV [57] and DSR [35], and in the proactive OLSR protocol [16] as we discuss below.

The route refresh period in proactive routing is normally larger than in reactive schemes. In OLSR, for example, the default refresh period is 16 seconds. We consider it too large assuming the high dynamics of data flows and mobile nodes in ad hoc networks. Based on these observation we do not see proactive routing protocols to be suitable for dissemination of PDP information.

As for AODV and DSR, we see a number of difficulties for the integration of PDP in these protocols. As stated in Section 7.1.1, the flows which share a part of their path to the destination should be treated by our scheme as separate connections. This requirement places a major limitation on the carrier protocol: It should not perform path aggregation. However, this functionality is embedded in both AODV and DSR. Firstly, path aggregation is the normal operation for routes to the same IP subnet. Secondly, even for the cases where IP level aggregation is not used, AODV and DSR provide gratuitous route replies: When a forwarding node is part of the path to a specific destination, it may return the RREP to the source without further propagation of the route request message. Integration of PDP into AODV or DSR would require changes to the core functionality of these L3 routing protocols.

7.1.6 Integration of PDP in LUNAR

LUNAR is a L2.5 reactive routing scheme [64]. The major difference of LUNAR from AODV and DSR is its position in the TCP/IP protocol stack. Since LUNAR is located below IP layer, the IP level route aggregation which is present in L3 routing schemes is not an issue. The route request messages in LUNAR always propagate through the full path from the source to the destination using their node addresses. That is, gratuitous replies are not used either. LUNAR is well suited for integration with PDP also because of its frequent information update: LUNAR re-establishes the entire path every three seconds as shown in Figure 7.2. We store PDP messages in two new fields of RREQ and RREP messages of LUNAR as indicated by the italic font in Figure 7.2.

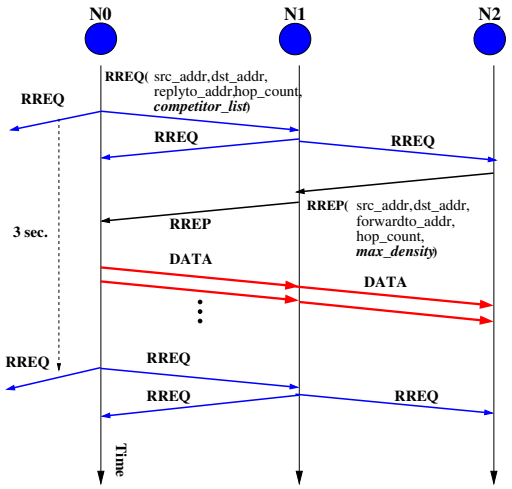


Figure 7.2: Main operations of LUNAR and integration of PDP (PDP specific parameters shown in *italics*): Establishment of a two hop connection.

7.2 Experiments

The single code base both for the Linux kernel and the network simulator ns-2 [82] allowed us to debug and extensively test the PDP+LUNAR functionality on a wide range of static and dynamic simulation scenarios as well as in real world. We implemented and tested both stateless and statefull PDP schemes. Once we obtained a stable protocol we generated a real world version and performed a correctness test in a test-bed as described in Section 7.2.2. While performing a preliminary evaluation of SL-PDP in the simulator we found the drawbacks of this scheme described in Section 7.1.2. As a result we decided to go on with SF-PDP only. In this section we present the results from testing the protocol both in simulations and the real-world test-bed.

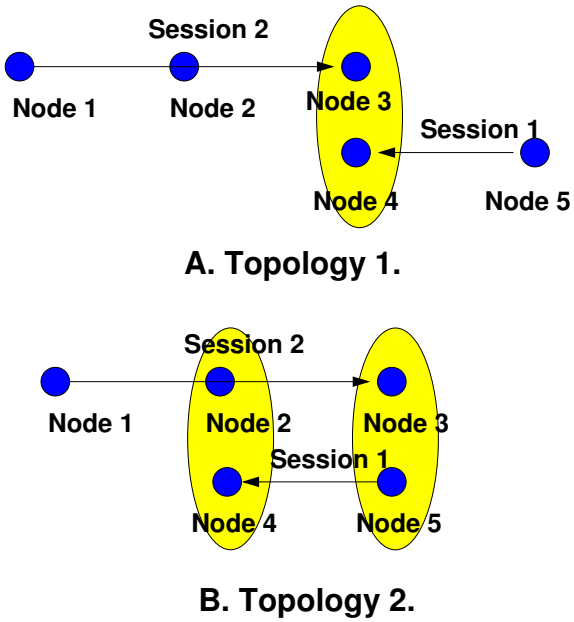


Figure 7.3: The two topologies for our qualitative real world-experiments.

7.2.1 Metrics

In all experiments we used correctness and convergence time metrics to evaluate the performance of PDP. Correctness is evaluated with respect to the "path density" as reported to the end nodes of the session (the path density in our experiments was also reported to destinations because we used bidirectional traffic in the experiments).

7.2.2 Static Scenarios (Real World)

The real-world results were obtained from a setup of five DELL Latitude laptops with ZyXEL ZyAIR B-100 wireless interfaces. We use the Linux operating system with 2.6 kernel and the LUNAR implementation that is available from [87].

We present an evaluation of the functional correctness of PDP on two static scenarios depicted in Figure 7.3. While all nodes in reality are located in the reception range of each other (hence the same radio interference domain) we configured LUNAR so that a node can hear the data transmission only from a selected set of nodes and discards packets from others. In this sense only Nodes 3 and 4 in Figure 7.3*a* "share" the regions of assured data reception of each other. In the second case (Figure 7.3*b*), Node 2 is able to hear transmissions from Node 4, and Node 3 from Node 5. In the first scenario Node 3 can not here the RREQ messages issued by the source Node 5. In the same way RREQ messages issued by Node 1 do not reach Node 4. What Nodes 3 and 4 hear are only the backwards RREQ message. The second scenario addresses the case where a node can hear both the RREQ messages issued by the source and the destination nodes of the particular association. Let us consider Node 4, the destination for Session 1. While sharing the "common" transmission range with Node 2, Node 4 hears both the RREQs issued by the source Node 1 and the backwards RREQs issued by Node 3. These two types of RREQs have swapped source and destination addresses. In order to avoid wrong counting such RREQs as two different connections, Node 4 should check the presence of such *symmetrical* SD pairs before inserting the new record in the local SD table and treat them as the indication to one connection. In both experiments PDP should report the same density information to all nodes in both cases.

In both scenarios Session 1 starts at time 1 second and finishes at time 30 seconds. Session 2 establishes the path 10 seconds after Session 1. The duration of Session 2 is 10 seconds. We used the ping protocol to generate traffic of the corresponding sessions. The interval between two consequent ICMP requests is 100 milliseconds. We repeated both real world experiments up to 10 times and obtained the same behavior of our two metrics. Figure 7.4 shows the results from one of these runs.

As is visible from the time diagrams, PDP correctly reports the path density to the corresponding end nodes of the two connections within 1.5 – 3 seconds from the start time of a session. The delay in dissemination of the information is explained by the default route refresh interval in LUNAR. After the session is established the earliest time a new RREQ message is generated by the destination in the backwards direction is after 1.5 seconds.

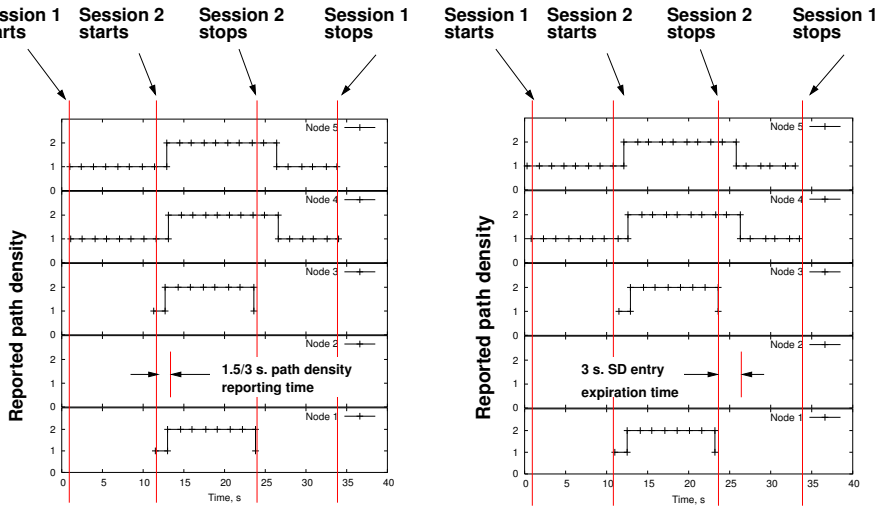


Figure 7.4: Reported path density in real-world experiments.

7.2.3 Dynamic Scenario (Simulation)

We evaluated the behavior of PDP in presence of mobility in simulations only. This time we seek a quantitative assessment of the path density information provided by PDP. We studied the scenario shown in Figure 7.5.

In the scenario we have three TCP sessions each following a path of two hops. Initially all nodes are located outside the range of assured reception of each other. After five seconds from the simulation start the middle nodes of sessions one and three begin a movement towards the middle node of Session 2. The speed of the nodes is 10 m/s. In their final destination both Nodes 2 and 8 are in the range of assured reception of each other and Node 5. The mobile nodes remain in this position for 80 seconds; After that they start moving back to their original position.

The goal of this experiment is to evaluate the dynamics of PDP based source throttling. We performed two experiments on this scenario. First, the path density information was ignored and TCP flows might transmit without rate limitation. In

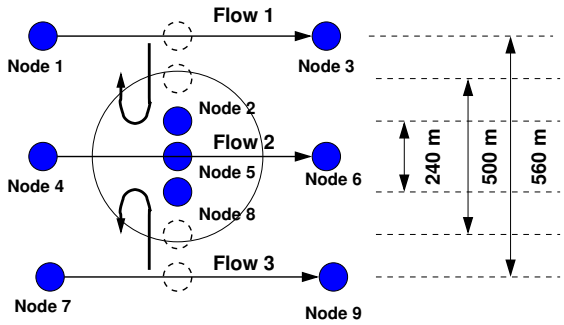


Figure 7.5: Topology 3 for experiments on a dynamic scenario.

the second experiment we configured the scheduler on the interface queue with a throttling limit dynamically computed according to the path density information reported by PDP. Figure 7.6 presents the results of these experiments.

As we observe from the figure, the slope of TCP sequence numbers curves is the same for all three flows provided that the path density is taken into account, which means fair sharing of the network's capacity. Moreover, in this case the progress of all TCP flows is very smooth and free from interruptions when compared to the case where the path density information is ignored. In addition to the smoothness of the TCP flows, the resulting total TCP throughput (not shown in the figure) remains the same as in the case without rate limitation at sources.

To sum up, the results of the last experiments illustrate the overall goal of our fairness framework: If flows in a MANET are aware about the presence of each other and reduce their rate accordingly, none of the competitors is able to capture the capacity as it happens for the plain combination of IEEE 802.11 MAC and TCP. The latter case is represented in Figure 7.6 by flow TCP 2 (w/o path density) which worsens the performance of TCP flows 1 and especially 3.

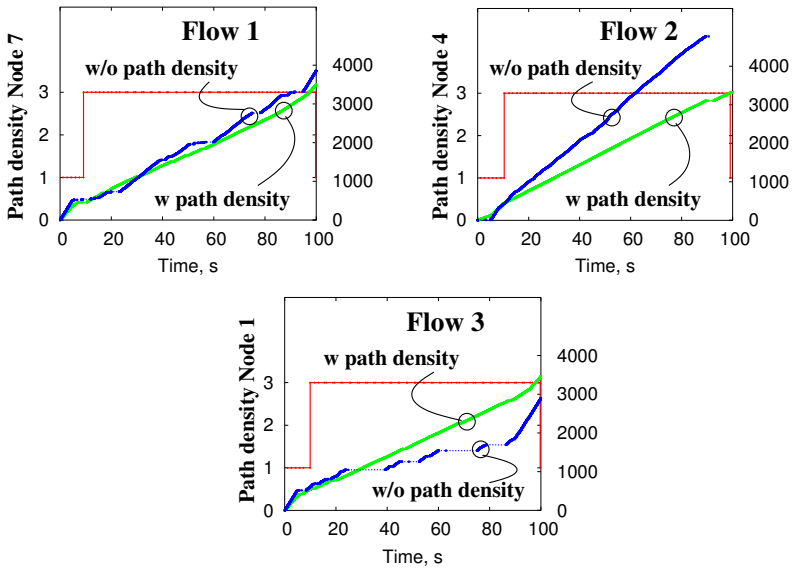


Figure 7.6: Path density and TCP sequence numbers progress for the topology in Figure 7.5.

7.3 Future developments

The path density gathering scheme presented in this chapter is a distributed protocol for the on-demand discovery of an ad hoc network state. The PDP protocol plus our ingress rate throttling approach permits to implement a capacity allocation scheme with guarantees on fairness of communications. In this section we briefly list possible extensions and generalizations of PDP.

- **Gathering C-load shares**

In order to implement the load share distribution algorithm for a general case MANET the suggested PDP protocol should be modified. This would permit MANET nodes to distribute not only the identities of the associations inside the L-region but also actively participate in deciding on the membership of the particular

association in one or another L-region. In order to do this the route update message issued by sources should carry the currently used load share. sertation.

- **Transmission rate aware ad hoc routing**

One of the parameters of our ingress rate limit formula (7.1) is the transmission rate at the physical layer along the path of the particular connection. The majority of the existing ad hoc routing protocols, however, do not have a functionality to either actively choose a route with the desired transmission rate or at least to report to the sources the actually *used* transmission rate on the path. The recent results from [3, 18] indicate the feasibility of adding this feature to the ad hoc routing schemes.

- **Controlled information dissemination patterns**

In order to resolve the dilemma of having a dynamic information updates and to avoid the usage of periodic broadcast invocations we suggest to use a special kind of periodic "keep alive" messages by reactive ad hoc routing schemes. These messages should be end-to-end in nature (i.e. they should be forwarded along the unicast path towards the destination by the routing layer), but transmitted to the broadcast address on the MAC layer. The end-to-end transmission of such messages along a single path implies that the underlying broadcast is controlled by the source. The rate of these messages can then be accounted in the calculation of the ingress rate limit, and thus would not add extra load to the network. The underlying broadcast transmission should allow dissemination of PDP messages in the same way as is described in this chapter.

- **Congestion-aware routing**

In addition to the admission control in sources we see potentials of our PDP protocol for congestion-aware routing. Indeed, the ND state kept in all forwarding nodes is an excellent indicator of the neighborhood's load. If a forwarding node – instead of re-broadcasting the newly arriving route request – would first examine the neighborhood density, it can estimate the chance for this flow to obtain an acceptable service while being forwarded through this area. If a neighborhood is overloaded with existing connections the route request can simply be discarded.

Assuming a network with relatively high node density, the "surviving" route requests would discover less loaded paths.

7.4 Summary

In this chapter we presented an approach for gathering path density information in a distributed way at run-time. In order to build the corresponding path density protocol (PDP) we need to resort to storing per-connection state in the forwarding nodes. We showed how PDP messages can be piggybacked inside LUNAR messages without introducing new transmission events and tested our implementation with combined simulation and real-world measurements. The design of PDP greatly benefited from being able to instantly switch between simulations and real world experiments during the developing, debugging and performance measuring phases.

With the specification of the path density protocol we showed the feasibility of our ingress throttling scheme for fair TCP communications in IEEE 802.11 based *mobile* ad hoc networks. By this we conclude the technical development of the topic of this dissertation. In the next chapter we give an overall summary of the presented material, describe the insights obtained during the thesis and state open research issues.

Chapter 8

Summary and outlook

It is fair to say that mobile ad hoc networks, which we considered in this thesis, are currently outliving their embryonic phase. Still, and to a large extend, it is not clear whether multihop wireless networks will be directly used by end users as a means of communications since MANET-specific applications are yet to appear. In this respect, the benefits of the infrastructure-less networks exist mainly in the minds of researchers.

However, already now we are witnessing the first attempts to use these networks for providing Internet connectivity to large residential areas. An illustrative example of such work is the project "Technology For All"[1] undergoing in the US. The deployment of the infrastructure-less wireless backbone in a large neighborhood permits to drastically reduce the deployment cost in comparison to the deployment cost of the traditional wire and fiber lines[2].

The later example highlights the needs to consider the use of traditional network services with TCP and UDP as transport protocols in MANETs. Through the history of its development and tuning to the specifics of the wireline Internet medium, TCP has become a mature protocol for reliable data transmission.

[1] [Online]. Available: `http://www.techforall.org`

[2] The following deployment cost estimates are taken from [13]. Deployment of a new fiber infrastructure is around $200,000 per linear mile. The deployment of the wireless multihop backbone is estimated to be around $25,000 per square mile.

Adopting the protocol in wireless networks, reveals its serious performance problems, which lead to an *unstable* and an *unpredictable* service for the end users.

In this dissertation we focused mainly on TCP performance in these networks. In the foreground of this work is the problem of severe unfairness between multiple multihop TCP connections. The major question addressed is whether multihop wireless networks are practically capable to provide the end users with the *reliable* data delivery service in the same *stable* way as this is done by the wireline Internet. This thesis suggests a practically implementable set of network mechanisms which allows fair TCP communications over IEEE 802.11 based multihop wireless networks.

In this chapter we summarize the lessons learned and insights obtained before in Section 8.2 we outline the open research topics which appeared to be outside the scope for this thesis.

8.1 Summary of the major observations and insights

A. General approach for MANET research: Synergy between research and reality

Working with multihop wireless networks we have very mixed feelings about the synergy between the existing research work on MANET architectures and the situation in reality. On a very superficial level these feelings can be expressed as follows. In the Internet of the recent years we witness stably operating networks and see research results which claim that one can do even better. In contrast, for MANETs nearly every comparative study or evaluation of a stand alone protocol or solution reports good results and promising performance in large scale ad hoc networks, while in reality basic network services cannot be assured even within small networks of a dozen of nodes.

One important point that we emphasize in this thesis is the need for reality oriented research in the MANET community. This, in the first place, means making realistic assumptions on the scale of MANETs. As we illustrated, the IEEE 802.11 technology itself places strict limitations on the number of supported nodes and the feasible data rates. Therefore, if we are talking about IEEE 802.11 based networks, it is incorrect to illustrate good scalability based on simplified radio

transmission models which for example do not account for interferences or use smaller transmission ranges. As we showed, the large radio transmission ranges and radio interferences play the key role in the formation of severe unfairness in MANETs.

Throughout the text of the thesis our main goal was to understand the operational range of current MANETs in terms of the number of simultaneously active data sessions, the number of nodes actively participating inside a single MANET and the path length of each flow. Looking at the problem of the synergy between the research and reality on the basis of *"ad hoc horizons"* highlights the need for conducting more intensive theoretical and optimization studies within the feasible operation limits of MANETs.

Another important point in this context is the impact of the proposed changes to existing well-established and popular protocols. From the researcher's point of view we obviously should not stick to the existing standards and should consider any suitable modifications and changes, which might lead to improvements. However, we believe that such "re-engineering" attempts will take longer to arrive at optimal solutions. The reason is that normally, introducing certain changes at MAC or TCP layer improves their joint performance in some specific scenarios; however, it remains on the same poor level in other cases because of the inefficient piecewise patching of such solid systems including TCP and MAC protocols. Our approach has been to find a corrective solution in the standard-free space which in a smart way makes the existing protocols to interoperate nicely.

B. Performance analysis and used metrics

One of the important lessons we learned during the development of the dissertation is the method of qualitative and quantitative assessment of the performance of the major MANETs components, such as TCP and routing protocols. We found that the right choice of illustrative metrics is a very challenging task. We also learned that in MANETs the performance evaluation of different communication layers of the TCP/IP protocol stack in isolation is to a large extend an incorrect approach. It is relatively easy to illustrate a favorable performance of a protocol at the particular layer of the communication stack. At the same time the performance of other MANET components might be far from acceptable. On the other

hand, evaluating the performance inside the particular layer is not an obvious task either. While the usage of one set of metrics leads to very optimistic conclusions, we were able to find other combinations, which sometimes indicate a completely opposite situation. In the rest of this subsection we will give particular examples for our performance evaluation experience.

Evaluating the performance of TCP communications

The problem we encountered while evaluating the performance of TCP communications is that the conventional performance metrics (e.g. the end-to-end throughput of an individual flow, the combined (total) throughput in the network, various unfairness indexes) considered in isolation highlight only one side of the overall TCP quality in MANETs. In particular, we showed that while the unfairness index might be low, which implies a good end-to-end throughput of each of the competing TCP flows, the progress quality of the data transfer is unacceptable for some of the end-users. In order to capture this side of the TCP quality we needed to introduce new metrics: The *"no-progress time"* of each session and the *"unsmoothness"* of a flow. While the first metric quantitatively characterizes the level of the service degradation for the particular TCP flow, the second metric qualitatively illustrates the deviation of the actual TCP arrival process from the estimated smooth progress. Such joint consideration of the conventional and the newly defined metrics allowed us to grasp a broader picture of TCP quality in MANETs.

Evaluating the performance of ad hoc routing protocols

Through the evaluation of ad hoc routing protocols we found that the conventional metrics for assessing the impact of a routing protocol on data traffic are not illustrative at all when matter comes to the evaluation of the *TCP + routing* interactions. We showed that the "average routing load" metric, commonly used in the comparative studies of different routing schemes, is meaningless for judging the impact on TCP communications. We also found it difficult to construct a new metric based on the direct observation of packet level interaction between the routing protocol and TCP wherefore we suggested a methodology for an indirect

analysis of routing protocols.

C. Cross-layer approach in wireless research

The term "cross-layer development" is very popular nowadays in the MANET research community. As we learned through the work on the dissertation different people understand this concept differently. Here are just two typical examples of this non-uniform understanding.

- ***Using link layer or network feedback for correcting TCP congestion control***

As it is discussed in the dissertation one way to improve the TCP performance is to make the congestion control mechanism aware about the reasons for packet losses in the network. This is to avoid an erroneous invocation of the slow start phase as a reaction on packet losses due to bit errors and not due to the network congestion.

- ***Using link layer feedback for route maintenance***

On the routing layer it is important to dynamically react on the link breakage in order to locally repair or re-establish the failed route quickly. One way of achieving this is to use explicit notification from the MAC layer up to the routing layer in the case of unsuccessful packet transmission to the next hop.

We consider both cases as examples of "partial" cross-layering because the feedback mechanism is too layer-specific. At several moments we experienced the situation where tuning the parameters of the particular protocol to overcome one set of problems produced other new problems. This in turn led to new information needed to be exchanged between network nodes and additional modifications to the considered protocol. In our opinion it is very difficult to trace all problems induced by new cross-layer optimizations which lead directly to a "feature interaction nightmare". Moreover, at the end, such a MANET-specific sequential patching of existing protocols based on new feedback information might lead to serious incompatibilities between the wireless and wireline domain.

Instead of partial cross-layering based on isolated mechanisms like link layer feedback, in this thesis we attempted to first characterize the properties of a whole layer. For example, we showed how understanding the properties of TCP and the limitations of the IEEE 802.11 technology allows to create a middle layer mechanism (the ingress throttling scheme), which optimally glues together the two technologies.

Future cross-layer proposals should also account for the mutual influence of all involved protocols (*TCP + routing + MAC*) and not only for the particular combinations like *TCP + MAC*, *TCP + routing*, *routing + MAC* etc. In the scope of this dissertation this methodology is illustrated by the studies of the impact of routing traffic on data communications. After optimization of the *TCP + IEEE 802.11 MAC* performance we showed that ad hoc routing itself becomes a reason for the poor TCP performance in MANETs.

D. Development of new protocols and mechanisms for MANETs – the reality oriented approach

Finally, summarizing the experience obtained during the development of this thesis we would like to comment on the reality oriented approach for the development of new protocols for MANETs. The boom of ad hoc networking research, which we are still witnessing nowadays, resulted in a large number of MANET-specific protocols and network mechanisms. However, the dominant tendency which can be observed in many publications is the extensive usage of simulations to show the performance of the proposals. Unfortunately, the conclusions drawn in such studies are in many cases far from those that the developers working with real-world wireless networks experience.

In our group from the very beginning of the development of a new protocol we tried to be as close in our assumptions to the reality as possible. The power of this approach is shown in the part of the thesis concerning the development of the *path density protocol*. The usage of the single code base both for the network simulator and the real operating system allowed us to instantly switch between simulation-based experiments and tests in the real-world testbed. Firstly, we are able to extensively test the functionality of our scheme on a wide range of network topologies in simulations. Once the desired and stable behavior of the protocol is

achieved, we are able to instantly compile the real-world version and verify its operations in the real-world testbed.

8.2 Future research in wireless networking

In this section we present three open research problems which we think should be attacked with high priority.

a) New lower layer protocols for MANETs

Major attention should be paid to minimizing the destructive effect of long-ranging radio interferences. Ideally one would wish for solutions at the physical layer. In parallel, MAC protocols should account for transmissions ongoing more than one hop away.

b) Routing that is data traffic-friendly and QoS-aware

Looking at the current activities of the MANET working group on specification of a new generation reactive routing protocol, it is easy to see the traces of previous suggestions to utilize the HELLO mechanism for connectivity maintenance between the neighbors[3]. In contrast, our analysis of different routing traffic patterns made it obvious that the extensive use of frequent broadcast transmissions becomes a reason for severe unfairness between TCP sessions. We showed that a "TCP-friendly" routing protocol should use mainly an error-driven form of broadcast invocations.

The QoS aspects of routing should be addressed in two ways. Firstly, new routing protocols should favor the routes through less loaded geographical regions. Our implementation of PDP creates the ground for embedding this functionality in reactive routing schemes. Secondly, the desired transmission rate criteria should be introduced to the routing path establishment procedure in order to provide the availability of multi-rate communications. The current routing

[3] In the recent draft for the next generation reactive routing protocol *DyMo* [14] this mechanism is specified inherently from AODV.

schemes using broadcast transmissions at the base transmission rate of 1 Mb/s for the path establishment, result in the situation where the data exchange is not possible with a higher data rate due to longer hops being favored by the routing protocol.

c) MANET-specific simplified reliable transport protocol

Our suggested *ingress throttling scheme* together with the *path density protocol* provides the sources of communications with all necessary congestion control functionalities. In this sense the existence of the additional congestion control mechanism at the transport layer (TCP) is redundant in our MANET environment. One could therefore create a new transport layer protocol with a simple re-transmission mechanism for reliable data transfer in MANETs which relies on the lower layers to do the congestion control job. In our opinion such a "splitting of concerns" approach will reduce the implementation complexity of the transport protocol and facilitate its formal analysis.

We see the creation of a new wireless-specific transport protocol particularly interesting for the development of new autonomic network architectures [75, 73]. In these networks wireless segments are native network components and not only an extension for the infrastructure based networks. Therefore it is essential to include the support for "wireless friendliness" in the network design from the very beginning and not to adopt a "patch" approach which, to a large extend, we are witnessing with TCP over mobile ad hoc networks today.

Bibliography

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

[2] G. Anastasi, M. Conti, and E. Gregory, *IEEE 802.11 ad hoc networks: protocols, performance and open issues*. Wiley-IEEE Press, 2004.

[3] B. Awerbuch, D. Holmer, and H. Rubens, "High throughput route selection in multi-rate ad hoc wireless networks," in *Proc. WONS'04*, St. Moritz, Switzerland, 2004.

[4] P. Bahl, R. Chandra, and J. Dunagan. SSCH: Slotted seeded channel hopping for capacity improvement in IEEE 802.11 ad-hoc wireless networks. [Online]. Available: http://research.microsoft.com/users/bahl/Papers/Pdf/SSCH.pdf

[5] A. Bakre and B. Bardinath, "Implementation and performance evaluation of indirect TCP," *IEEE Transactions on Computers*, no. 46, 1997.

[6] H. Balakrishnan, V. Padmanabhan, S.Seshan, and R. H. Katz, "A comparison of mechanisms for improving TCP performance over wireless links," in *Proc. ACM SIGCOMM'05 workshops*, Philadelphia, PA, USA, Aug. 2005.

[7] D. Bertsekas and R. Gallager, *Data networks*. Prentice-Hall, 1992.

[8] S. Bhandarkar, N. Sadry, A. Reddy, and N. Vaidya, "TCP-DCR: A novel protocol for tolerating wireless channel errors," *IEEE Transactions on Mobile Computing*, Feb. 2004.

[9] J.-Y. L. Boudec, "Rate adaptation, congestion control and fairness: A tutorial," EPFL, Feb. 2005. [Online]. Available: http://ica1www.epfl.ch/PS_files/LEB3132.pdf

[10] B. Braden, D. Clark, J. Crowcroft, B. Davie, S. Deering, D. Estrin, S. Floyd, V. Jacobson, G. Minshall, C. Partridge, L. Peterson, K. Ramakrishnan, S. Shenker, J. Wroclawski, and L. Zhang, "Recommendations on queue management and congestion avoidance in the Internet," IETF RFC 2309. [Online]. Available: http://www.rfc-editor.org/rfcsearch.html

[11] J. Broch, D. Maltz, D. Johnson, Y-C.Hu, and J. Jetcheva, "A performance comparison of multi-hop wireless ad hoc network routing protocols," in *Proc. ACM MobiCom'98*, Dallas, TX, USA, 1998.

[12] K. Brown and S. Singh, "M-TCP: TCP for mobile cellular networks," *ACM Computer Communication Reviews*, Jul. 1997.

[13] J. Camp, E. Knightly, and W. Reed, "Developing and deploying multihop wireless networks for low-income communities," in *Proc. Digital Communities 2005*, Napoli, Italy, Jun. 2005.

[14] I. Chakeres, E. Belding-Royer, and C. Perkins, "Dynamic MANET on-demand (DYMO) routing," IETF draft (work in progress), 2005. [Online]. Available: http://www.ietf.org/internet-drafts/draft-ietf-manet-dymo-02.txt

[15] Ch.Tschudin and E. Osipov, "Estimating the ad hoc horizon for TCP over IEEE 802.11 networks," in *Proc.MedHoc'04*, Bodrum, Turkey, Jun. 2004.

[16] T. Clausen and P. Jacquet, "Optimized link state routing protocol (OLSR)," IETF RFC 3626, Oct. 2003. [Online]. Available: http://www.rfc-editor.org/rfcsearch.html

[17] S. Corson and J. Macker, "Mobile ad hoc networking (MANET): Routing protocol performance issues and evaluation considerations," IETF Infromational RFC 2501, 1999. [Online]. Available: http://www.rfc-editor.org/rfcsearch.html

[18] D. D. Couto, D. Aguayo, J. Bicket, and R. Morris, "A high throughput path metric for multihop wireless routing," in *Proc. ACM MobiCom'03*, San Diego, CA, USA, Sep. 2003.

[19] S. R. Das, C. Perkins, and E. M. Royer, "Perfortmance comparison of two on-demand routing protocols for ad hoc networks," in *Proc. IEEE Infocom'00*, Tel-Aviv, Israel, Mar. 2000.

[20] T. Dyer and R. Boppana, "A comparison of TCP performance over three routing protocols for mobile ad hoc networks," in *Proc. ACM MobiHoc'01*, Long Beach, CA, USA, Oct. 2001.

[21] H. Elaarag, "Improving TCP performance over mobile networks," *ACM Computing Surveys*, vol. 34, no. 3, Sept. 2003.

[22] S. ElRakabawy, A. Klemm, and C. Lindemann, "TCP with adaptive pacing for multihop wireless networks," in *Proc. ACM MobiHoc'05*, Urbana-Campaign, Illinoise, USA, May 2005.

[23] Z. Fu, P. Zerfos, H. Luo, S. Lu, L. Zhang, and M. Gerla, "The impact of multihop wireless channel on TCP throughput and loss," in *Proc. Infocom'03*, San Francisco, USA, Apr. 2003.

[24] Z. Fu, X. Meng, and S. Lu, "How bad TCP can perform in mobile ad hoc networks," in *Proc. IEEE ICC'02*, 2002.

[25] M. Gerla, K. Tang, and R. Bagrodia, "TCP performance in wireless multihop networks," in *Proc. of IEEE WMCSA'99*, New Orleans, LA, USA, Feb. 1999.

[26] A. Gurtov, "Effect of delays on TCP performance," in *Proc. PWC'01*, 2001.

[27] A. A. Hanbali, E. Altman, and P. Nain, "A survey of TCP over mobile ad hoc networks," INRIA, France, Tech. Rep. 5182, May 2004.

[28] M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda, "Performance anomaly of 802.11b," in *Proc. IEEE Infocom'03*, San Francisco, CA, USA, 2003.

[29] G. Holland and N. Vaidya, "Analysis of TCP performance over mobile ad hoc networks," in *Proc. ACM MobiCom'99*, Seattle, WA, USA, 1999.

[30] X. Huang and B. Bensaou, "On max-min fairness and scheduling in wireless ad-hoc networks: analytical framework and implementation," in *Proc. ACM MobiHoc'01*, Long Beach, CA, USA, 2001.

[31] P. Jacquet and L. Viennot, "Overhead in mobile ad-hoc network protocols," INRIA, France, Tech. Rep. 3965, 2000.

[32] S. Jaiswal, G. Iannaccone, C. Diot, K. J, and D. Towsley, "Inferring TCP connection characteristics through passive measurements," in *Proc. IEEE INFOCOM 2004*, 2004.

[33] L. B. Jiang and S. C. Liew, "Proportional fairness in wireless LANs and ad hoc networks," in *Proc. IEEE Wireless Communication and Network Conference (WCNC'05)*, Mar. 2005.

[34] M. Johansson and L. Xiao, "Cross-layer optimization of wireless networks using nonlinear column generation," *IEEE Transactions on Wireless Communications*, 2004.

[35] D. Johnson, D. Maltz, and Y.-C. Hu, "The dynamic source routing protocol for mobile ad hoc networks (DSR)," IETF draft (work in progress), 2003. [Online]. Available: http://www-2.cs.cmu.edu/~dmaltz/dsr.html

[36] J. Jubin, "Current packet radio network protocols," in *Proc. Infocom'85*, Mar. 1985.

[37] R. Kahn, "The organization of computer resources into a packet radio network," *IEEE Trans. Comm*, vol. COM-25, no. 1, Jan. 1977.

[38] V. Kawadia, "Protocols and architectures for wireless adhoc networks," Ph.D. dissertation, University of Illinois at Urbana-Champaign, USA, 2004. [Online]. Available: http://black.csl.uiuc.edu/~prkumar/ps_files/04_07_kawadia_thesis.pdf

[39] V. Kawadia and P. Kumar, "Experimental investigations into TCP performance over wireless multihop networks," in *Proc. ACM SIGCOMM'05 workshops*, Philadelphia, PA, USA, Aug. 2005.

[40] A. Kherani and R. Shorey, "Throughput analysis of TCP in multi-hop wireless networks with IEEE 802.11 MAC," in *Proc. IEEE WCNC'04*, Atlanta, USA, Mar. 2004.

[41] S.-B. Lee, G.-S. Ahn, and A. Campbell, "Improving UDP and TCP performance in mobile ad hoc networks with INSIGNIA," *IEEE Communication Magazine*, Jun. 2001.

[42] S.-J. Lee, E. Belding-Royer, and C. Perkins, "Scalability study of the ad hoc on-demand distance vector routing protocol," *International Journal of Network Management*, vol. 13, no. 2, Mar./Apr. 2003.

[43] J. Li, C. Blake, D. D. Couto, H. Lee, and R. Morris, "Capacity of ad hoc wireless networks," in *Proc. ACM MobiCom'01*, Rome, Italy, Jul. 2001.

[44] J. Liu and S. Singh, "ATCP: TCP for mobile ad hoc networks," *IEEE JSAC*, vol. 19, no. 7, 2001.

[45] R. Ludwig and R. H. Katz, "The Eifel algorithm: making TCP robust against spurious retransmissions," *ACM Comp. Commun. Rev.*, vol. 30, no. 1, Jan. 2000.

[46] R. Ludwig, "Eliminating inefficient cross-layer interactions in wireless networking," Ph.D. dissertation, Aachen University of Technology, Germany, 2000.

[47] H. Lundgren, D. Lundberg, J. Nielsen, E. Nordström, and C. Tschudin, "A large-scale testbed for reproducible ad hoc protocol evaluations," in *Proc.*

3rd annual IEEE Wireless Communications and Networking Conference (WCNC 2002), Orlando. FL, USA, Mar. 2004.

[48] S. Mascolo, C. Casetti, M. Gerla, M. Sanadidi, and R. Wang, "TCP westwood: Bandwidth estimation for enhanced transport over wireless links," in *Proc. of the 7th annual international conference on Mobile computing and networking*, Italy, 2001.

[49] K. Nahm, A. Helmy, and C.-C. J. Kuo, "TCP over multihop 802.11 networks: issues and performance enhancement," in *Proc. ACM MobiHoc'05*, Urbana-Campaign, Illinoise, USA, May 2005.

[50] R. Oliveira, "Addressing the challenges for TCP over multihop wireless networks," Ph.D. dissertation, University of Bern, Switzerland, Jun. 2005.

[51] R. Oliveira and T. Braun, "A delay-based approach using fuzzy logic to improve TCP error detection in ad hoc networks," in *Proc. IEEE Wireless Communications and Networking Conference (WCNC'04)*, Atlanta, USA, Mar. 2004.

[52] ——, "A dynamic adaptive acknowledgment strategy for TCP over multihop wireless networks," in *Proc. IEEE Infocom'05*, Miami, USA, Mar. 2005.

[53] E. Osipov, "Empirical upper bound on TCP transmission rate for guaranteed capture-free communications in multi-hop IEEE 802.11 based wireless networks," University of Basel, Switzerland, Tech. Rep. CS-2005-001, Feb. 2005.

[54] E. Osipov and Ch.Tschudin, "A path density protocol for MANETs," in *Proc. IEEE ICPS Workshop on Multi-hop Ad hoc Networks: from theory to reality (REALMAN'05)*, Santorini, Greece, Jul. 2005.

[55] ——, "A path density protocol for MANETs (Extended version)," University of Basel, Switzerland, Tech. Rep. CS-2005-004, May 2005.

[56] E. Osipov, C. Jelger, and Ch.Tschudin, "TCP capture avoidance in wireless networks based on path length and path density," University of Basel, Switzerland, Tech. Rep. CS-2005-003, Apr. 2005.

[57] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc on-demand distance vector (AODV) routing," IETF RFC 3561, Jul. 2003. [Online]. Available: http://www.rfc-editor.org/rfcsearch.html

[58] D. Perkins and H. Hughes, "Investigating the performance of TCP in mobile ad hoc networks," *International Journal of Computer Communications*, vol. 25, no. 11-12, 2002.

[59] B. Radunovic and J.-Y. L. Boudec, "Rate performance objectives of multihop wireless networks," in *Proc. IEEE Infocom'04*, Hong Kong, 2004.

[60] C. Rohner, E. Nordström, P. Gunningberg, and C. Tschudin, "Interactions between TCP, UDP and routing protocols in wireless multi-hop ad hoc netwrorks," in *Proc. IEEE ICPS Workshop on Multi-hop Ad hoc Networks: from theory to reality (REALMAN'05)*, Santorini, Greece, Jul. 2005.

[61] R. Stevens, *TCP/IP illustrated volume 1: The protocols*. Addison-Wesley, 1993.

[62] K. Tang and M. Gerla, "Fair sharing of MAC under TCP in wireless ad hoc networks," in *Proc. IEEE MMT'99*, Venice, Italy, Oct. 1999.

[63] Y. Tian, K.Xu, and N. Ansari, "TCP in wireless environments: problems and solutions," *IEEE Radio Communications*, Mar. 2005.

[64] C. Tschudin, R. Gold, O. Rensfelt, and O. Wibling, "LUNAR: a lightweight underlay network ad-hoc routing protocol and implementation," in *Proc. NEW2AN'04*, St. Petersburg, Russia, Feb. 2004.

[65] N. Vaidya, P. Bahl, and S. Gupta, "Distributed fair scheduling in a wireless LAN," in *Proc. ACM MobiCom'00*, Boston, MA, USA, 2000.

[66] K. Viswanath and K. Obraczka, "Modeling the perfortmance of flooding in wireless multi-hop ad hoc networks," in *Proc. SPECTS'04*, 2004.

[67] K. Xu, S. Bae, S. Lee, and M. Gerla, "TCP behavior across multi-hop wireless networks and the wired Internet," in *Proc. WoWMoM'02*, Atlanta, GA, USA, Sep. 2002.

[68] K. Xu, M. Gerla, L. Qi, and Y. Shu, "Enhancing TCP fairness in ad hoc wireless networks using neighborhood RED," in *Proc. ACM MobiHoc'03*, Annapolis, MD, USA, 2003.

[69] S. Xu and T. Saadawi, "Does the IEEE 802.11 MAC protocol work well in multihop wireless ad hoc networks?" *IEEE Communications Magazine*, Jun. 2001.

[70] G. Xylomenos, F. C. Polyzos, P. Mähönen, and M. Saaranen, "TCP performance issues over wireless links," *IEEE Communications Magazine*, no. 54-58, Apr. 2001.

[71] L. Yang, W. Seah, and Q. Yin, "Improving fairness among TCP flows crossing wireless and wired networks," in *Proc. of the 4th ACM international symposium on Mobile ad hoc networking and computing*, Annapolis, MD, USA, 2003.

[72] X. Yu, "Improving TCP performance over mobile ad hoc networks by exploiting cross-layer information," in *Proc. ACM MobiCom'04*, Philadelphia, USA, Sep. 2004.

[73] ANA: Autonomic network architectures website. [Online]. Available: http://www.csg.ethz.ch/research/projects/ANA

[74] AODV-UU implementation, Uppsala University. [Online]. Available: http://core.it.uu.se/AdHoc/AodvUUImpl

[75] Autonomic communication website. [Online]. Available: http://www.autonomic-communication.org

[76] Documentation for network simulator ns-2. [Online]. Available: http://www.isi.edu/nsnam/ns/ns-documentation.html

[77] *Economist*, A brief history of Wi-Fi, Jun. 10, 2004.

[78] Friis transmission equation. [Online]. Available: http://en.wikipedia.org/wiki/Friis_Transmission_Equation

[79] IEEE 802.11 standards portal, IEEE. [Online]. Available: http://www.ieee.org/

[80] "Internet protocol," IETF RFC 791. [Online]. Available: http://www.rfc-editor.org/rfcsearch.html

[81] Mobile ad-hoc networks (MANET), the IETF working group. [Online]. Available: http://www.ietf.org/html.charters/manet-charter.html

[82] Network simulator ns-2. [Online]. Available: http://www.isi.edu/nsnam/ns/

[83] "Recommendations on queue management and congestion avoidance in the Internet," IETF Infromational RFC 2309. [Online]. Available: http://www.rfc-editor.org/rfcsearch.html

[84] The Rice University Monarch Project: Mobile networking architectures. [Online]. Available: http://www.monarch.cs.rice.edu

[85] "TCP NewReno," IETF RFC 3782. [Online]. Available: http://www.rfc-editor.org/rfcsearch.html

[86] "Transmission Control Protocol," IETF RFC 793. [Online]. Available: http://www.rfc-editor.org/rfcsearch.html

[87] Uppsala University ad hoc implementation portal. [Online]. Available: http://core.it.uu.se/AdHoc/ImplementationPortal

[88] Wireless Application Protocol forum. [Online]. Available: http://www.wapforum.org

Curriculum Vitae

Evgeny Alexandrovitch Osipov

1976	Born in Novosibirsk, Russia.
1983 – 1993	Primary and secondary school #5 in Krasnoyarsk, Russia.
1993 – 1998	Faculty of Computer Science at Krasnoyarsk State University of Technology, Russia. Awarded the qualification “Mathematician” with a Major in Organization and Technology of Information Security. Diploma with Honors.
1998 – 1999	Pre-doctoral school in Communication Systems. EPFL, Swiss Federal Institute of Technology, Lausanne, Switzerland
1999 – 2003	Licentiate of Engineering, Institute of Microelectronics and Information Technology at KTH, Royal Institute of Technology, Stockholm, Sweden
2004 – 2005	PhD, Computer Science Department, University of Basel, Switzerland.