A Survey on Congestion Control for Mobile Ad-Hoc Networks

Christian Lochert* Björn Scheuermann Martin Mauve

University of Düsseldorf, Computer Science Department,
Universitätsstr. 1, D-40225 Düsseldorf, Germany
Phone: +49 211 81-11634, Fax: +49 211 81-11638
{lochert, scheuermann, mauve}@cs.uni-duesseldorf.de

Abstract

Congestion control is a key problem in mobile ad-hoc networks. The standard TCP congestion control mechanism is not able to handle the special properties of a shared wireless multihop channel well. In particular the frequent changes of the network topology and the shared nature of the wireless channel pose significant challenges. Many approaches have been proposed to overcome these difficulties. In this paper, we give an overview over existing proposals, explain their key ideas and show their interrelations.

Index Terms

Mobile Ad-Hoc Networks, Wireless Multihop Networks, Congestion Control, TCP, Survey

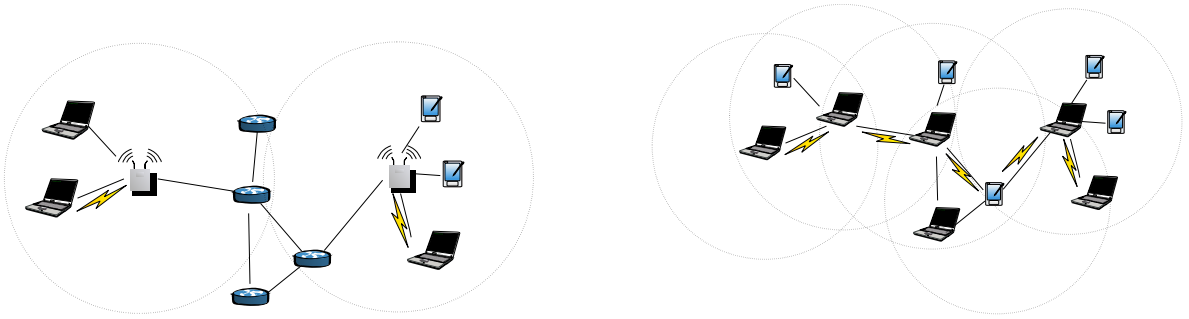
I. Introduction

In this survey paper we consider the problem of congestion control in mobile ad-hoc networks (MANETs). In most wireless networking environments in productive use today the users' devices communicate either via some networking infrastructure in the form of base stations and a backbone network, examples are WLANs, GSM/UMTS, and 4G Networks (see Figure 1(a)), or directly with their intended communication partner, e.g. using 802.11 in ad-hoc mode. In contrast a mobile ad-hoc network does not have an infrastructure and still the devices do not need to be within each other's communication range to communicate. Instead, the end-users' (mobile) devices also act as routers, and data packets are forwarded by intermediate nodes to their final destination (see Figure 1(b)). MANETs are applicable in situations where no infrastructure is available; a common example is a disaster relief scenario. They are also the foundation for vehicular ad-hoc networks, where communication between cars is used to increase vehicle safety and driving comfort. There are also related multihop wireless networks, e. g. wireless mesh networks or wireless sensor networks. These networks share some of the congestion control related problems with MANETs.

Much research effort has been put into the MANET area. For example, a large variety of approaches has been put forward how to perform routing in MANETs. However, it has also been observed that the transport layer functionality needs to be adapted to the specific properties of MANETs. In particular the congestion control mechanisms implemented in the common transport protocol used today, i. e., TCP, do not deal properly with the specific effects occurring in MANETs.

* Corresponding Author

This is a preprint of an article published in Wiley Wireless Communications and Mobile Computing 7 (5), pp. 655–676, June 2007. http://www.interscience.wiley.com.



(a) Infrastructure-based wireless network.

(b) Mobile ad-hoc network.

Fig. 1. Comparison of wireless network architectures.

This is due to the fact that TCP has originally been designed for the Internet, a network with fundamentally different properties.

As a consequence, appropriate congestion control is widely considered to be a key issue for MANETs. Several distinct congestion-related problems have been identified and tracked down, including severe throughput degradation and massive fairness problems. They have been shown to originate from the MAC, routing, and transport layers, as discussed, e. g., in [22], [24], [49], [52]. There is a large variety of ideas on how to overcome the difficulties. In this survey paper we provide an overview of existing attempts to solve the congestion control problem in mobile multihop ad-hoc networks. We do not consider approaches aimed at improving congestion control or TCP performance over single-hop wireless networks. Schemes which are specifically targeted at multicast flows are likewise not described here.

The rest of this paper is structured as follows. First we will give a short introduction to the congestion control problem in general and to some specific properties of mobile ad-hoc networks that are relevant in the congestion control context in section II. In the then following sections, we introduce a structure on the proposed solutions, grouping them in categories based on the problem that they focus on. Section III describes approaches that deal primarily with problems due to mobility-induced route failures. Random wireless losses are the main concern of the proposals in section IV. In section V approaches are discussed that deal with the special properties of a shared medium. Self contention between data- and ACK-traffic is the main concern of the approaches in section VI. In section VII we describe proposals that artificially limit the packet output of TCP in order to adapt TCP to MANETs. Finally, there are a number of approaches specifically tailored to the properties of MANETs. These are not based on TCP. They are discussed in section VIII. Some approaches can be assigned to multiple sections, since they address more than one key problem. In these cases we present them in the section where they provide the most significant contribution. Finally, some general observations and concluding remarks follow in section IX, and an outlook on possible future research directions is given in section X.

II. Problem Statement

This section provides an introductory overview of the congestion problem. After a brief outline of the problem in general we will give an overview how congestion is handled on the Internet, namely the TCP congestion control mechanism. Thereafter we show why mobile ad-hoc networks pose new challenges to congestion control.

A. The congestion problem

In a network with shared resources, where multiple senders compete for link bandwidth, it is necessary to adjust the data rate used by each sender in order not to overload the network. Packets that arrive at a router and cannot be forwarded are dropped, consequently an excessive amount of packets arriving at a network bottleneck leads to many packet drops. These dropped packets might already have travelled a long way in the network and thus consumed significant resources. Additionally, the lost packets often trigger retransmissions, which means that even more packets are sent into the network. Thus network congestion can severely deteriorate network throughput. If no appropriate congestion control is performed this can lead to a congestion collapse of the network, where almost no data is successfully delivered. Such a situation occurred on the early Internet, leading to the development of the TCP congestion control mechanism [28].

B. TCP congestion control

On the Internet, congestion control is in the responsibility of the transport layer, more precisely of the Transmission Control Protocol (TCP). TCP combines congestion control and reliability mechanisms. This combination allows to perform congestion control without the need for explicit feedback about the congestion state of the network, and without direct participation of the intermediate nodes. To detect network congestion TCP simply observes occurring packet losses. Since on the Internet missing packets are almost always caused by congestion, a missing packet is interpreted as a sign for network congestion.

TCP uses cumulative acknowledgments: a TCP receiver always acknowledges the end of the so-far correctly and completely received data when a new segment arrives. If segments are received out-of-order, i. e., some data is missing between the already known and the newly arriving data, the last acknowledgment is sent again (duplicate ACK).

In TCP a window-based additive increase, multiplicative decrease mechanism is employed. The window size is increased by one segment (i. e., additively) in every round-trip time when no packet losses occur. In case of the reception of a duplicate acknowledgment a TCP sender will first assume that some packet reordering has occurred in the network. But upon reception of the fourth copy of an acknowledgment (Triple Duplicate ACK, TDACK) a congestion loss is asumed. In this case the missing segment is repeated and the window size is cut in half (multiplicative decrease).

Additionally, TCP uses a timeout that depends on the measured round-trip time of the connection. If this retransmission timeout (RTO) elapses without an acknowledgment TCP concludes severe congestion. Then the window size is reduced to one and the unacknowledged segment is sent again. The timeout until the next retransmission attempt if still no acknowledgment arrives is doubled. Thus this timeout grows exponentially.

During the first phase of a connection and after a timeout a mechanism named slow start is employed. It allows for a faster convergence to the correct window size. While slow start is active, the window size is not increased by one segment size for every round-trip time, but instead for every received acknowledgment. This means that during this phase the window size grows exponentially.

A number of extensions and modifications of TCP have been proposed. Two of them are particularly relevant here, since they have been adopted for some MANET congestion control approaches. When the delayed acknowledgments (DACK) [7] option is used, an acknowledgment is sent only for every second segment, instead of for every single one, or after some timeout at the receiver has expired. This reduces the number of acknowledgment packets. The selective acknowledgment (SACK) [39] option allows for a more fine-grained feedback on missing segments

than the standard cumulative ACK scheme. It is particularly useful if multiple segments are missing in the same round-trip period.

C. Congestion in mobile ad-hoc networks

TCP congestion control works very well on the Internet. But MANETs exhibit some unique properties that greatly affect the design of appropriate protocols and protocol stacks in general, and of a congestion control mechanism in particular. As it turned out, the vastly differing environment in a mobile ad-hoc network is highly problematic for standard TCP.

Paramount amongst the specific properties of MANETs are the node mobility and a shared, wireless multi-hop channel. Route changes due to node mobility as well as the inherently unreliable medium result in unsteady packet delivery delays and packet losses. These delays and losses must not be misinterpreted as congestion losses.

The use of a wireless multi-hop channel allows only one data transmission at a time within the interference range of one node. Thus geographically close links are not independent from each other. This influences the way network congestion manifest itself tremendously. Internet routers typically are dedicated hosts connected by high bandwidth links. When congestion occurs on the Internet, it is usually concentrated on one single router. In contrast, congestion in MANETs affects a whole area because of the shared medium. Not nodes are overloaded, but regions of the network.

Although this depends on the network type, packet losses which are not caused by network congestion can be much more frequent in wireless networks. This can lead to wrong reactions of TCP congestion control. Moreover, observing packet losses is much harder, because transmission times and thus also round trip times vary much more.

Furthermore, due to the comparatively low bandwidth of mobile ad-hoc networks, one single sender is able to—be it accidentally or intentionally—cause a collapse of the network due to congestion. The extreme effect of a single traffic flow on the network condition can cause severe unfairness between flows. Thus wireless multihop networks are much more prone to overload-related problems than traditional wireline networks like the Internet. Therefore an appropriate congestion control is absolutely vital for network stability and acceptable performance.

Table I summarizes the major problems and observations that have been made in MANET congestion control research so far. It is also indicated in which sections of this paper the respective aspect is discussed.

D. Application scenarios for MANET congestion control

Since the application scenarios for multihop wireless networks are so heterogeneous, the appropriate congestion control solution for a specific network and application will largely depend on the properties and the purpose of the respective network. One single, general-purpose solution for all possible scenarios is thus unlikely to exist. This is also reflected in the proposals discussed in this paper. Most do not represent complete, ready-to-use protocols, but rather solutions for a subset of the identified problems, which can serve as building blocks for application-tailored protocol stacks. Some properties of the protocols are, however, important for a broader range of applications. Together with an overview of the approaches considered here these are displayed in Table II.

In the table it is shown whether certain network and application designs are possible with the respective congestion control approach. Assume for example, that the MANET is to be connected to some backbone network or the Internet. Then it is either necessary that the congestion control

TABLE I
OBSERVATIONS IN MANET CONGESTION CONTROL AND THE SECTIONS THEY ARE DISCUSSED IN.

Observation	in section(s)
Route failures trigger inappropriate TCP congestion control reactions.	III
The standard TCP retransmission timeout grows too fast in MANET environments.	III
After rerouting, the new route can have very different properties.	III
It is possible to foresee route failures by keeping track of the RSSI.	III
Random losses can occur on wireless links.	IV
The locally shared medium induces unfairness between TCP flows.	V
TCP has a long feedback path.	V
Data and acknowledgment packets interfere on the shared medium.	V, VI
TCP oversaturates the network.	V, VII
On the shared medium there is intra-flow contention between successive data packets.	V, VII
The TCP acknowledgment scheme generates a lot of packets.	VI
TCP traffic is bursty.	VII
TCP's basic design decisions do not fit a MANET environment well.	VIII

protocol does also work appropriately over wired networks, or that some sort of proxy or gateway is provided that connects the two parts of the network. In the column "Internet gateway" of the table it is indicated whether the approach will work in such an environment (✓), or if its design does not allow for a connection to wired backbone networks, at least not without modifications (–). If the scenario is not considered by the authors and it is either independent from the design decisions made in the protocol or unclear whether the approach is directly applicable this is indicated by ○.

The "TCP friendly" column indicates whether data flows controlled by the proposed approach and standard TCP flows can coexist in the same network and a fair sharing of the bandwidth can be expected. "Separated congestion control and reliability" means that the approach solves the two main tasks of TCP separately and is thus also applicable, for example, for UDP-like traffic. This is obviously not possible if the approach is heavily based on TCP. Likewise TCP-based approaches cannot satisfy the "smooth data rate" criterion. This indicates whether the protocol will deliver the packets at a constant rate with stable inter-arrival times, as opposed to TCP's typically very bursty delivery pattern. This is important especially for multimedia applications. "Routing independent" shows whether the approach is independent from the routing protocol. This applies when it does neither rely on a certain routing protocol nor on specific modifications or extensions at the routing layer. Finally, by "MAC independent" we denote that the protocol can be implemented independent from and without modifications to the MAC protocol. This can be a central aspect for a decision for or against a certain approach since depending on the hardware platform it might be impossible to modify the MAC implementation.

III. DEALING WITH ROUTE FAILURES

In a typical mobile ad-hoc network route failures occur frequently. The amount of time required for discovering a new route has a negative impact on the standard TCP congestion control mechanism. For a long time no packets are delivered and no acknowledgments are received, causing the TCP sender to reduce its window size dramatically, even though in fact no real congestion situation might exist. The approaches described in this section primarily tackle this problem.

One of the very first approaches to deal with the congestion control problem in MANETs at all is called **TCP-Feedback (TCP-F)**. It has been proposed by Chandran et al. in 1998 [8], and has become the basis for a lot of subsequent work in this area.

TABLE II
APPLICATION REQUIREMENTS.

	Approach	Ref.	based on	Internet gateway	TCP friendly	Sep. reliability and CC	Smooth data rate	Routing independent	MAC independent
Route Failures	TCP-F	[8]	TCP	✓	✓	–	–	–	✓
	ELFN	[27]	TCP	✓	✓	–	–	–	✓
	TCP-BuS	[31]	TCP	✓	✓	–	–	–	✓
	Fixed RTO	[17]	TCP	✓	✓	–	–	✓	✓
	ENIC	[47]	TCP	✓	✓	–	–	–	✓
	TCP-RC	[62]	TCP	✓	✓	–	–	–	✓
	ATCP	[38]	TCP	–	✓	–	–	✓	✓
	Cross-layer information awareness	[55]	TCP	–	✓	–	–	–	✓
	TCP-DOOR	[50]	TCP	✓	✓	–	–	✓	✓
	Preemptive routing	[25]	TCP	○	✓	–	–	–	✓
	Atra	[3]	TCP	○	✓	–	–	–	✓
	Signal-strength based link management	[32]	TCP	○	✓	–	–	–	–
	Multipath-TCP	[36]	TCP	–	✓	–	–	–	✓
Wireless Losses	TCP/RCWE	[26]	TCP	✓	○	–	–	–	✓
	ADTCP	[21]	TCP	✓	○	–	–	✓	✓
	ADTFRC	[23]	TFRC	✓	○	–	✓	✓	✓
	Edge-based approach	[14], [16]	TCP	✓	○	–	–	✓	✓
Shared Medium	LRED / adaptive pacing	[24]	TCP	○	✓	–	–	✓	–
	NRED	[51]	TCP	○	✓	–	–	✓	✓
	COPAS	[13]	TCP	○	✓	–	–	–	–
	CAR	[54]	TCP	○	✓	–	–	–	✓
	Split-TCP	[34]	TCP	○	○	–	–	✓	✓
	Alleviating self-contention	[5]	TCP	○	○	–	–	✓	–
	OPET	[59], [61]	TCP	○	○	✓	–	✓	–
	RE-TFRC	[35]	TFRC	✓	○	–	✓	✓	✓
ACK Traffic	Dynamic delayed ACK	[2]	TCP	✓	✓	–	–	✓	✓
	Dynamic adaptive ACK	[15]	TCP	✓	✓	–	–	✓	✓
	Preferred ACK Retransmission	[46]	TCP	○	✓	–	–	–	–
	Combining data and ACK packets	[56]	TCP	○	✓	–	–	✓	–
Output Limit	Dynamic congestion window limit	[12]	TCP	–	–	–	–	–	✓
	SCA	[42]	TCP	✓	–	–	–	✓	✓
	FeW	[41]	TCP	✓	–	–	–	✓	✓
	Non-work-conserving scheduling	[53]	TCP	✓	○	–	–	✓	✓
	RBCC	[60]	TCP	–	○	–	–	✓	✓
	C^{3}TCP	[33]	TCP	–	○	–	–	✓	–
	TCP-AP	[18]	TCP	–	○	–	–	✓	✓
Alternative Protocol	EXACT	[9], [11]	-	✓	–	✓	✓	✓	–
	ATP (Sundaresan)	[48]	-	–	–	✓	✓	✓	✓
	ATP (Liu)	[37]	-	○	–	○	○	✓	✓
	WXCP	[45]	XCP	–	–	–	–	✓	–
	TPA	[4]	-	○	–	–	–	✓	✓

The authors of TCP-F propose to disable the TCP congestion control mechanisms in case of network-induced, non-congestion related losses or timeout events caused by route failures. The TCP sender is notified explicitly when routing failures occur (RFN, Route Failure Notification) and when a new route has been discovered (RRN, Route Re-establishment Notification). When a route to the destination node is currently not available, i. e., after receiving an RFN message, the TCP-F sender enters a "snooze" state. In this state it freezes its TCP state values such as timers and window sizes.

An intermediate node generates an RFN message when it detects a link failure on the route. Once a node previously generating or forwarding an RFN learns about a new route to the destination node, it generates an RRN message and sends it to the source node. When the source node receives an RRN, it resumes the TCP session with the previously frozen state values. To prevent a TCP-F session from remaining in snooze state indefinitely in case an RRN message gets lost, an additional timeout is used as a fallback.

Holland and Vaidya analyze the impact of **Explicit Link Failure Notification (ELFN)** techniques on TCP performance [27]. ELFN means feedback from lower layers to notify TCP explicitly about link or routing failures. In case of such a failure, the sender enters *standby mode*, which is the equivalent to TCP-F's snooze state.

In contrast to the TCP-F proposal no explicit notification in case of a re-established route is used. Instead, a TCP-ELFN sender sends probe packets in regular intervals when in standby mode. A probe packet is not a special control packet. Instead the first data packet in the send queue is used for that purpose. Standby mode is left as soon as a probe packet is acknowledged by the destination node.

Likewise, for route failure notifications no special control packet is introduced in ELFN. The authors propose to either piggyback the notification message onto a route failure message sent by the routing protocol (as used, for example, in DSR), or to use an ICMP host unreachable message for that purpose.

ELFN has become very well-known and has served as a basis for many later approaches.

Monks at al. showed in [40] that although ELFN can indeed improve the performance of TCP, there are situations where a severe performance degradation is possible. This is particularly true for scenarios with many active connections. The reason is that an ELFN-like mechanism makes TCP behave more aggressive, i. e., a higher number of packets is present in the network. Therefore the MAC layer contention is generally higher. This in turn leads to more collisions, a higher packet loss rate on the MAC layer and eventually to false broken link detections; therefore wrong route failure notifications are sent and unnecessary route discoveries are performed. This observation is quite fundamental and should generally be taken into account in mobile ad-hoc networks. It does not only apply to ELFN or TCP-F, but also to many of the following approaches.

In **TCP with BUffering capability and Sequence information (TCP-BuS)** by Kim, Toh and Choi [31], the idea of explicit notifications from the routing layer is extended. Additional measures are proposed to improve TCP performance.

In TCP-BuS, intermediate nodes buffer packets in case of a route failure instead of dropping them; the intention is to not having to retransmit all these packets. To avoid timeouts at the source when buffered packets are delivered after the route is reconstructed, the timeouts for these packets are extended. Selective retransmission requests by the destination node allow a fast recovery from lost buffered packets without the need to wait for these extended timeouts to expire. A scheme for avoiding unnecessary retransmission requests is proposed.

In order to ensure the delivery of control messages like route failure notifications, a reliable

delivery scheme is suggested for these messages. To detect a re-established route both explicit notifications and probe packets, i. e., both the TCP-F and the ELFN way of detecting the new route, are examined.

In [17] Dyer and Boppana compare the performance of different TCP variants over three MANET routing protocols. Based on their finding that an exponentially growing retransmission timeout (RTO) is problematic in MANETs because of the route failures, they propose to keep the RTO fixed after the first retransmission. Their **Fixed RTO** scheme thus leads to a periodic sending of packets. Essentially this has the same effect as ELFN's probe packets. However, their scheme is much simpler because it does not interact with the routing layer, and due to the dependency of the RTO on the measured round trip time it is inherently adaptive.

They compare this mechanism with the use of TCP's selective acknowledgment (SACK) and delayed acknowledgment (DACK) extensions. The results of their ns-2 simulations show that the delayed and selective acknowledgment mechanisms of TCP yield only minor improvements, but with fixed RTO a performance increase similar to that of ELFN is possible.

In the **ENhanced Inter-layer Communication and control (ENIC)** scheme proposed by Sun and Man [47], too, ELFN-like route failure handling and the TCP SACK and DACK mechanisms are combined. In comparison to TCP-BuS, ENIC requires less assistance from the intermediate nodes. Like in ELFN, no separate notification messages in case of a route failure or a recovered route are transmitted. Instead the reuse of notifications inherent to the routing protocol is proposed. A buffering of packets at the intermediate nodes is not suggested, queued packets are dropped when a route breaks.

Not only the sender, but also the receiver is notified of route failures in ENIC. Like the sender it freezes its state, particularly that of the DACK mechanism. With a broken route, acknowledgment packets would not be able to reach the source node, anyway.

In ENIC, a new retransmission timeout value (*Temporary RTO*) is calculated after a route change in a heuristic fashion. It is based on the hop counts of the old and the new route. Thus of the approaches described here ENIC is the first one taking potential changes of route characteristics after a route reestablishment into account.

Zhou et al. in [62] focus especially on the problem of changing route properties after a route reestablishment. As an extension to the ELFN mechanism they propose to recalculate TCP's congestion window size (cwnd) and slow start threshold (ssthresh) parameters based on the properties of the new route. They call their approach **TCP-ReComputation (TCP-RC)**.

In TCP-RC, the route length and the round trip time—both relative to their previous values—are used to determine new values for cwnd and ssthresh. In ns-2 simulations they compare TCP-RC to plain TCP Reno and find performance improvements; however, it is not clear how much of this improvement is due to the ELFN mechanism alone, since this variant is not examined.

With **TCP for mobile ad hoc networks (ATCP)** Liu et al. [38] present a solution to several TCP problems in MANETs. ATCP not only handles route failures, but it is also intended to correctly handle longer periods of disconnection and to distinguish congestion-related from other losses. For the latter explicit congestion notification (ECN) messages are used.

Instead of invoking standard TCP congestion control in the case of non-congestion losses ATCP uses a TCP state freezing mechanism with an TCP-ELFN-like probe packet mechanism. A loss is considered to be non-congestion related if no ECN message is received.

An important point for the authors is to preserve standard TCP compatibility as far as possible. Therefore ATCP is implemented in an additional layer below the transport layer, reducing the interaction with TCP to a minimum.

A problem with explicit link failure notification schemes observed by Yu in [55] is that still a number of data packets and ACKs may get lost before the state is frozen. This has negative effects *after* the state is restored: missing packets or missing ACKs will then cause timeouts or duplicate acknowledgments. Yu's **cross-layer information awareness** scheme overcomes this by extensively using cached route information from the DSR routing protocol. Two mechanisms are introduced, *Early Packet Loss Notification (EPLN)* and *Best-Effort ACK Delivery (BEAD)*.

EPLN notifies TCP senders of the sequence numbers of lost packets that could not be salvaged. The sender can then disable the retransmission timer and retransmit the respective packets upon route reestablishment. BEAD generates ACK loss notifications at intermediate nodes and sends them towards the TCP receiver. This prevents ACKs from being permanently lost. A node forwarding such a loss notification may send an ACK with the highest affected sequence number to the TCP sender if it is able to do so. Otherwise the TCP receiver will retransmit an ACK with the highest sequence number when a new route is present.

Wang and Zhang—in contrast to most of the previously described approaches—propose a pure end-to-end mechanism. It is called **TCP with Detection of Out-of-Order and Response (TCP-DOOR)** [50]. They regard the deployment and maintenance of cross-layer solutions as too intricate in many scenarios.

In TCP-DOOR, data packets and/or ACKs delivered out-of-order are utilized as an indication of changed routing without the need of explicit feedback. While the receiving node can notify the sender about detected out-of-order data packets, the sender itself may notice ACKs arriving out-of-order.

Two mechanisms are suggested as a reaction to the occurrence of out-of-order events. When out-of-order packets are detected, the sender may temporarily disable TCP's congestion control mechanisms by keeping its state variables constant. Additionally, it may fall back to a somewhat older state. Thereby it reverts effects of congestion control mechanisms that might already have occurred. This last mechanism is called *Instant Recovery*. The expected effect is similar to that of freezing: after the reset "wrong" changes to TCP's parameters are reverted, and the connection continues to operate as if no route change had occurred.

Through simulation studies the authors have found that it is perfectly sufficient to detect out-of-order events either at the sender or at the receiver. A combination of both detection mechanisms does not yield significantly better results. The best performance is achieved by a combination of both possible reactions, that is, by temporarily disabling congestion control and performing Instant Recovery.

The authors recommend their approach primarily for mixed ad-hoc and wired network scenarios, where it would be particularly hard to adopt a feedback-based approach. Where possible, they suggest to use some feedback-supported method instead.

In [25] Goff et al. propose an early detection scheme for route failures. The idea of **preemptive routing** is to foresee route failures before they actually occur, and to initiate a new path discovery early. This is intended to avoid or at least to reduce disconnection times. Especially for TCP—with the negative effects that the disconnection periods have on its performance—significant improvements are expected by the authors.

Upon reception of a packet each node along a route looks at the received signal strength. If it is below a given threshold, a warning is sent to the source node of the route. To mitigate short-term effects like small scale fading, the use of an exponential average or of a verification of the measurement by sending some small ping-pong packets along the respective link are proposed. In simulations they evaluate their scheme and find a largely improved TCP performance when

preemptive routing is used.

In [3] Anantharaman et al. perform an analysis of TCP behavior in mobile ad-hoc networks, studying the factors that influence the protocol's behavior. They conclude that the schemes proposed so far, in particular the ELFN scheme, are only able to deal with part of the problems TCP is facing due to breaking links and route failures. They propose a set of three mechanisms with a somewhat broader scope, designed specifically for the DSR routing protocol [29]. They call the combination of these mechanisms the **Atra framework**.

The main aims of Atra are the minimization of route failures, their prediction and a fast notification of the source in case of a route failure. The mechanisms used to achieve these goals are called *Symmetric Route Pinning*, *Route Failure Prediction* and *Proactive Route Errors*. Symmetric Route Pinning forces TCP acknowledgment packets to use the same route as the corresponding data packets. Usually, different routes could be used in DSR. The authors' reasoning is that using different routes increases the probability of a route failure—there are two routes that may fail.

The Route Failure Prediction mechanism in Atra works like the preemptive routing scheme: each node along the route estimates the trend of received signal strength values. In case the Route Failure Prediction doesn't work, the third mechanism, Proactive Route Failures, notifies the sources of *all* connections that use the broken link. This is different from standard ELFN mechanisms, where only the source of the packet that could not be transmitted on the MAC layer is notified of the problem, meaning that every connection sharing the broken link has to detect the problem separately.

Another approach anticipating route failures is the **signal strength based link management** by Klemm et al. [32]. They point out that the 802.11 MAC cannot identify link breaks correctly in case of congestion. The situations when two nodes move out of transmission range and when a congested area does not allow a successful RTS transmission cannot immediately be distinguished. The authors aim to distinguish these cases better and to provide appropriate reactions in conjunction with the AODV routing protocol [43].

The first key idea is the same as that of preemptive routing: a history of node distances, estimated by the received RSSI values, is kept for the neighbors. If a used link is about to break the routing protocol is informed and a new route to the destination can be searched early. In addition to the signal strength based link management it adds mechanisms to ease the salvaging by temporarily increasing the transmission power. In case congestion prevents the transmission of data, the number of RTS/CTS retries before a is packet dropped is increased.

The main difference between route failure anticipation schemes and the approaches discussed before is that these schemes try to avoid performance degrading effects from occurring in the first place, instead of alleviating their effect. This concept is also the key idea in the **multipath TCP** scheme by Lim et al. [36]. They use existing multipath routing protocols to study the effects that these approaches have on the TCP performance. They compare two modes how the additional routes provided by the routing layer can be used: distributing the TCP packets to multiple routes in parallel and maintaining additional routes just as a backup in case of a route failure.

By simulation with Fixed RTO TCP (see above) as a transport protocol they find that it is not beneficial to use multiple paths in parallel. Using different paths—with different round trip times—makes TCP's RTT measurements unreliable. Additionally, out-of-order delivery effects produce many duplicate ACKs and thus trigger unnecessary congestion control reactions.

In contrast to that, maintaining one additional route as a fallback has a positive effect. In case of a route error the transmission can be continued quickly. The authors call this scheme *backup path multipath routing*. In some sense, it can be seen as a consequent variant of the route failure prediction schemes: instead of establishing an alternative path when a route failure is about to

occur, it is maintained from the beginning, in order to be always quickly available.

IV. Coping with Wireless Losses

A wireless link per se is much more prone to more or less random packet losses than a wireline connection. Such losses are detrimental for a transport protocol's performance if they are misinterpreted as congestion-induced packet drops.

This problem is not a central issue when 802.11-like MAC protocols are used, where the link layer provides single-hop reliability. In this case packets are dropped by the link layer only after a number of failed transmission attempts or failed RTS/CTS handshakes. This in turn usually happens when either a link is lost or when a lot of packet collisions occur. The latter means that there probably is a lot of traffic in the area around the currently forwarding node and thus congestion. Therefore although the packet loss has not been caused by a queue overflow in a router it can still be a valid congestion indicator.

However, it can well be argued that there might be networks without single hop reliability. In such a case the proposals presented in this section show how end-to-end transport protocols—designed to rely on missing packets as congestion indicators—can learn to deal with random losses. Some work in the area of distinguishing wireless losses from congestion losses has also been done in the area of wired-cum-wireless networks, see for example [6]. Here, we only focus on the approaches for mobile ad-hoc networks.

Güneş and Vlahovic propose to introduce three states in TCP senders [26]. This approach is called **TCP with Restricted Congestion Window Enlargement (TCP/RCWE)**. It is based on the Explicit Link Failure Notification mechanism.

In TCP/RCWE, link breaks and thus the corresponding losses are handled by ELFN. In addition, RCWE aims to deal well with random losses. To this end, the authors propose a mechanism based on a heuristic observing the value of the Retransmission Timeout (RTO). If the RTO increases the congestion window size is not increased. If the RTO decreases or remains constant the congestion window size is increased according to TCP's rules.

In ns-2 simulations, RCWE is found to cause a much smaller congestion window, leading to higher goodput and less packet losses. But it is compared only to standard TCP without ELFN, so it is not clear how much of the performance gain is due to ELFN alone. Since seemingly IEEE 802.11 with link layer reliability is used in the simulations, the observed performance gain might also occur only because cwnd is increased less often and thus is smaller on average. In this case, the observed effect would be the same as the one discovered later by Fu et al. in [24], leading to the congestion window size limitation schemes described in section VII.

To improve the reliability of bit error loss detection, Fu et al. in [21] combine multiple metrics instead of relying just on a single one. They call their approach **ADTCP**. They state that the main problem of end-to-end transport protocols in mobile ad-hoc networks is the noisiness of the measurements of indicators for certain network events.

Two metrics are proposed to detect network congestion. The *inter-packet delay difference* at the receiver, defined as the time elapsing between two successive packet arrivals, increases when congestion occurs. Additionally, the *short-term throughput*, describing the throughput in a certain time interval in the immediate past, decreases in case of congestion. These two metrics are combined to gain a more robust congestion indicator. In a similar way, out-of-order packet arrivals and the packet loss ratio are used to detect route changes and channel errors.

In ADTCP, the receiver detects the most probable current network state and includes this information into its feedback to the sender. Both sender and receiver behavior are altered appropriately, but

at the same time remain compatible with standard TCP. An ADTCP sender can talk to a standard TCP receiver and vice versa. ADTCP also behaves TCP-friendly.

Especially because of these last properties ADTCP might be an interesting option for scenarios with different protocols in the same MANET, with mixed wired and wireless infrastructure or with existing TCP-based applications.

For mobile multimedia communication Fu et al. also propose an adaption of TFRC called **ADTFRC** [23]. TFRC is a rate-based transport protocol originally developed as a TCP-friendly transport protocol for wired networks with smooth rate adaption properties [19].

ADTFRC applies the same ideas to TFRC that ADTCP applies to TCP. An identical combination of metrics and general mechanism are used to distinguish loss types and to provide receiver-based feedback. ADTFRC shares most of the benefits of ADTCP, especially the possibility of incremental deployment—ADTFRC, too, is compatible to standard TFRC communication partners.

De Oliveira et al. propose to use the measured round trip times to distinguish between congestion and medium losses [16]. In their **edge-based approach** the TCP congestion control reaction is circumvented if a medium loss is detected. Also, route failures are detected when a timeout occurs and no packets at all have been received for a longer period of time. In this case TCP enters an ELFN-like "probe mode", in which packets are transmitted at regular intervals in order to be able to detect a re-established route.

The specific mechanism to distinguish congestion from medium related losses is presented in [14]. It is based on fuzzy logic [57], [58]. The authors use ns-2 simulations with a single observed flow and some background traffic to examine the performance of their approach, and find a good detection accuracy, although the amount of samples needed can be high, causing the algorithm to be too slow for highly dynamic scenarios. The authors name some techniques that could be used to improve the fuzzy logic engine in this regard.

Given the approaches described in this section, the results presented in [12], [24] are of particular interest. The authors show that the assumption that more knowledge on the cause of packet losses solves the problems of TCP in MANETs may not be true. On the contrary, the resulting, more aggressive TCP behavior might lead to an even higher load on the network and thus more congestion problems.

V. Managing a Shared Medium

In a wireless network the medium is shared by all nodes in a certain area. Dealing with this property is a big challenge when one wants to perform congestion control in such networks: it makes congestion a spatial phenomenon, happening no longer in a node, but rather in an area. Several mechanisms paying attention to these special limitations have been proposed. They are presented here.

In a frequently cited work by Fu et al. [24] the authors show that for a given topology and traffic pattern there exists an optimal TCP window size, but TCP is unable to find it. Instead it uses larger windows, leading to dropped packets caused by link-layer contention. This observation has influenced research in wireless multihop congestion control significantly. Given this background the authors propose two mechanisms to improve TCP by earlier reaction to link overload—a distributed **Link RED (LRED)** and an **adaptive pacing** strategy.

The RED mechanism in wired networks [20] drops packets in router queues at random with a probability that increases linearly with the queue length. This makes TCP flows passing through the router regulate their bandwidth requirements before severe congestion occurs. LRED's intention is to make the TCP flows regulate their window size closer to the optimal region for MANETs.

In LRED, the probability for a drop is based on the observation of the number of transmission attempts needed on the MAC layer. The mechanism is enabled once a certain threshold is exceeded. With more frequent retries the probability to drop a packet is increased, because this situation is interpreted as a sign for local congestion.

The other technique—adaptive pacing—is also enabled once the LRED retransmission count threshold is reached. When active, the mechanism adds an additional packet transmission time to the sender's MAC backoff timer. Therefore the medium can be expected to be free also at the next node downstream, avoiding the so-called *exposed receiver problem*. This leads to a two hop coordination mechanism as the sender waits long enough that a packet can be forwarded from the receiver one hop further.

Xu et al. in [51] focus especially on TCP unfairness problems caused by the locally shared medium. Their proposal to improve TCP fairness, **Neighborhood RED (NRED)**, is like the previously discussed Link RED based on Random Early Detection.

In NRED, every node estimates the number of packets queued in its neighborhood in total, i. e., in all nodes in the vicinity. All these packets form a virtual, distributed neighborhood queue. If the length of this queue exceeds a threshold, packets start getting dropped with increasing probability.

NRED consists of three steps. The neighborhood queue size estimation is performed by analyzing the channel utilization. Therefore the transmissions of the neighbors are overheard. If the utilization exceeds a threshold the neighborhood is presumed to be in an early congested state. A drop probability is calculated and is sent explicitly to all neighbors to inform them. Each node calculates its own drop probability based on the received notifications. Incoming packets are dropped with this local probability, in total leading to RED-like packet dropping in the virtual neighborhood queue.

The authors of **COntention-based PAth Selection (COPAS)**, de Cordeiro et al. [13], focus on a problem of TCP in MANETs called the *capture problem*. Nodes can capture the medium unfairly and gain an advantage in comparison to others.

COPAS is an extension for reactive routing protocols. During route discovery all routes between a source and destination node are gathered. Then two disjoint routes are used to forward upstream TCP traffic and downstream acknowledgments respectively, in order to avoid effects where one of the two directions captures the medium. The decision which routes are chosen is based on congestion measurements performed during the discovery process. The measurements are based upon the backoff times for which the node had to wait before the medium became free. They are updated continuously during operation, and a route that becomes too congested is substituted by a better one.

Interestingly, the fact that two disjoint routes are used for forward and backward traffic is perfectly opposed to the Symmetric Route Pinning technique from the Atra Framework (see section III), where special care is taken to use the *same* route.

Ye et al. in [54] propose to extend the routing in order to separate flows spatially on the basis of distributed congestion information in their **Congestion Aware Routing (CAR)** approach. In contrast to most previous work in the congestion aware routing area they focus on TCP flows and the interaction with the congestion control mechanism.

First, they evaluate the theoretical benefit for spatial separation by simulating a centralized approach (Centralized CAR, CCAR). They assume that every node has total knowledge about source, destination, and route of every single TCP flow.

A decentralized approach (Distributed CAR, DCAR) is then described as a more realistic scenario. Every node locally calculates a congestion weight representing its local load situation and broadcasts it to its neighbors. The AODV routing protocol is used in an adapted form. Route discovery messages

are collected for some time until they are forwarded, and routes are chosen based on the path's aggregated weight. The destination node uses the path with the minimal weight to send the route reply message.

The performance analysis show that both approaches outperform shortest path routing protocols for long paths in terms of throughput. The centralized approach clearly outperforms the distributed procedure and is also favorable for short paths where the distributed approach fails. However, for obvious reasons it is not feasible in practice. As the main reason for the inferior performance of DCAR on comparison to CCAR the overhead for broadcasting the—potentially already outdated—congestion information has been identified by the authors.

Spatial separation of flows as in COPAS or CAR might actually be an effective means of reducing contention-related negative effects in MANETs. However, problems could arise in such a problem due to possible interactions between congestion control and routing. In particular feedback-loop oscillations can occur if congestion induces route changes, thus create network load in different areas of the network and eventually force other streams to change their routes.

A completely different approach to solve TCP's problems in MANETs has been proposed by Kopparty et al. in [34]. In **Split TCP**, congestion control and end-to-end reliability mechanisms are separated.

In intermediate nodes on the path between sender and receiver so-called *TCP proxies* are automatically established. The proxies subdivide the path into several independent segments. Each proxy buffers packets and transmits them either to the next proxy or to the final destination. *Local acknowledgments* are used to acknowledge packets within one segment. In addition to local acknowledgments end-to-end ACKs—potentially cumulative—are used to ensure reliability in case of a proxy failure.

Split TCP intends to alleviate mobility related effects by keeping other path segments functioning if a link breaks in one segment. Capture effects are expected to be less severe, since the transportation of data occurs in shorter stages and thus the size of the captured region is reduced.

Unlike most previously described approaches, Split TCP does not try to modify TCP behavior to introduce "knowledge" about specific properties of MANETs into the mechanisms. Instead, the long way for feedback from the sink back to the source is identified as a major problem and alleviated by shortening the feedback path.

The focus of Berger et al.'s scheme for **alleviating self-contention** [5] is on contention between packets belonging to the same transport layer flow. This can happen for both packets going into the same and packets traveling in the opposite direction.

The authors propose two mechanisms to make bidirectional data communication more reliable. *Quick Exchange (QE)* helps packets traveling in opposite directions, *Fast Forward (FF)* tries to reduce self-contention for packets with the same direction. Both are extensions to 802.11's RTS/CTS mechanism.

QE allows to exchange two packets in opposite directions by using only one exchange of RTS/CTS information. By adding an extra duration header to the first data transmission the network allocation vector of all other nodes in range is extended appropriately. A packet in the opposite direction with a piggybacked ACK can then be sent directly, i. e. without a second RTS/CTS. After both transmissions the original sender completes the procedure with an additional ACK.

FF speeds up the forwarding of a packet in the downstream direction. Like QE there is only one exchange of RTS/CTS information. But here the ACK is piggybacked with a new RTS packet for forwarding. This way, packets are forwarded faster over multiple hops to avoid self contention with other packets of the same flow. The use of this technique is restricted probabilistically because too frequent usage can lead to flow unfairness.

In simulation studies, the authors see a performance gain by their techniques, but also some problems in the interaction of FF with TCP: FF causes a high RTT variance, leading to suboptimal TCP performance.

Against the background of shared medium effects Zhai at al. in [61] and in [59] propose four mechanisms to reduce the impact that inter-flow and intra-flow contention have on the throughput and fairness in MANETs. They call their approach **Optimum Packet scheduling for Each Traffic flow (OPET)**. For reliable end-to-end communication they combine the proposed mechanisms with standard TCP.

The authors propose to assign a higher priority for the medium access to a node that has just received a packet. This is to give "downstream" packet transmissions a higher priority and thus to alleviate intra-flow contention. This has a similar background as the previously described Fast Forwarding mechanism. In addition, a hop-by-hop backward-pressure scheme keeps upstream nodes from sending further packets until the previous ones have been forwarded. This mechanism is tightly coupled to the 802.11 RTS/CTS mechanism, allowing the receiving node to send a "negative CTS" (NCTS) in order to signal that it is not willing to receive another packet of a certain flow. The upstream node then has to wait until its next hop explicitly gives the permission to continue.

The third mechanism limits the number of packets that source nodes can inject into their local queues, in order to prevent the sources from consuming too much local bandwidth for themselves. Finally, a flow-based round-robin scheme keeps, e. g., a single hop flow from occupying the medium excessively.

The basic ideas of OPET look very promising. However, especially the backward-pressure scheme might have some problems in networks that are more realistic than the static topologies in the authors' simulations. Since the scheme requires to wait for a potentially long period of silence for the next hop to allow to continue with the transmission after a NCTS, the scheme might prolong the time until broken links are detected.

Like Fu et al. had done before with ADTFRC (see section IV), Li et al. propose an adaption of TFRC to MANET requirements called **RE-TFRC** [35]. RE-TFRC is intended to alleviate negative MAC layer influences on the TFRC rate control mechanism. The authors show that unmodified TFRC produces a high load, beyond the MAC's saturation point. They introduce a rate estimation (RE) algorithm into TFRC. It uses a model for the round trip time to derive the loss rate equivalent to the load saturating the MAC layer capacity. The rate estimation algorithm is used to constrain the rate of TFRC.

The authors regard RTS/CTS-induced congestion as a major reason for the performance drop in an overload situation. Their rate estimation algorithm is based on a wireless multihop network model that is essentially equivalent to a collection of independent single hop wireless networks. This model is used to derive the optimal round trip time upon MAC saturation, which in turn serves as a basis for constraining the TFRC rate below this point.

RE-TFRC's approach to tackle the problems of transport protocols in wireless multihop networks by refining the modeling seems promising. However, the question arises whether the approach will work in more complex scenarios than evaluated by the authors, e. g., when interactions between multiple flows occur. It is also interesting to compare the findings in this paper to those of Chen and Nahrstedt in [10]. They show fundamental problems of equation-based congestion control in MANETs. Whereas in RE-TFRC it is observed that the offered load is beyond the MAC's saturation point, Chen and Nahrstedt report a too conservative behavior of TFRC.

VI. Handling ACK Traffic

Because of the shared medium, packets using the same route—or spatially close routes—in opposite directions severely affect each other. A very prominent example for this situation is the end-to-end acknowledgment traffic generated by transport protocols, causing intra-flow contention between data packets and acknowledgment packets traveling in opposite directions.

The question arises how the amount of ACK traffic or at least its negative impact on the performance of the forward channel can be minimized. This is closely related to the effects caused by the shared medium in general. Consequently there is some overlap with the previous section, and some of the approaches described there also consider the interplay between oppositely-directed data and ACK traffic. The work described in this section focuses solely on the acknowledgment traffic.

Altman and Jiménez in their **dynamic delayed ACK** approach follow this direction in [2]. Using delayed acknowledgments (DACK) in MANETs has been proposed before, e. g., in [17], [47]. Here, the authors extend the idea beyond standard DACK's combination of only two consecutive ACKs (see RFC 1122 [7]). In the Dynamic Delayed ACK scheme, only after a given number d of segments or after a certain, fixed timeout an acknowledgment packet is sent.

For $d = 2$ the authors observe significant performance improvements in their ns-2 simulations, which increase further for higher values of d of 3 or 4. However, these values might be problematic when TCP operates at a small window size. Therefore the authors propose to use a dynamic delayed ACK with d growing with an increasing packet sequence number, up to $d = 4$. Once this limit is reached, d is never decremented again. They state that a value of d depending on the current window size would probably lead to better results, but they do not want to introduce the additional changes required in order to make this information available at the receiver. Simulations demonstrate the performance gains possible with this approach, but are performed only with single flows in static networks.

De Oliveira and Braun in [15] identify some drawbacks in Altman and Jiménez' scheme. They criticize the fixed number of packets handled with a single ACK after the startup phase and the fixed ACK timeout. They design a new scheme called **dynamic adaptive acknowledgment** by applying concepts proposed in RFC 2581 (TCP Congestion Control) [1] to TCP in MANETs, and by a dynamic ACK timeout which is calculated based on the packet inter-arrival times at the receiver.

The concepts from RFC 2581 comprise an immediate acknowledgment upon out-of-order packets or packets filling a gap at the receiver. For the timeout a sliding average over the packet inter-arrival times is maintained. The authors argue that it is reasonable to wait at least for the time until the second next packet should arrive before a timeout occurs. If this packet is in-order and just delayed, everything is fine. Otherwise, if a packet is missing, the next packet will be out-of-order and will thus trigger an immediate acknowledgment.

In addition to these mechanisms, the parameter d is also changed dynamically. It grows additively up to a maximum value of 4. On out-of-order or gap-filling packets it is reduced to the Delayed ACK standard value of 2. Additionally cwnd is limited to a maximum of 4 segments.

The positive results obtained with delayed acknowledgment schemes in general are very interesting, because these techniques can be combined with many other approaches quite well, and are a promising way of reducing the number of packets and thus the contention for the shared medium. It could also be considered to use similar techniques for aggregating the feedback from the receiver to the sender in other transport protocols.

An in-network method for dealing better with acknowledgment traffic, the **preferred ACK retransmission**, is proposed in [46] by Sugano et al. As a basis for their system, they suggest to

combine two well-known approaches from the literature, namely ELFN messages and the DACK option for TCP on the Flexible Radio Network (FRN). FRN is a commercially available MANET system by Fuji Electric. It is not based on the IEEE 802.11 standard and uses fixed time slots on the medium. On this foundation, they put forward an additional improvement, intended to avoid repeated collisions of ACKs with data packets from the same stream. Their idea is to give acknowledgment packets a higher priority on the medium, by assigning them a shorter MAC retransmission interval.

Also tailored for the FRN is a technique proposed by Yuki et al. in [56]. They use a mechanism in intermediate nodes for **combining oppositely-directed TCP data and ACK packets** into one common frame.

Their main idea is to avoid using a whole FRN time slot for a very short ACK frame and at the same time to reduce the probability of packet collisions by combining a data and an ACK packet into a common frame if the transmitting node has packets of both kinds in its queue. This frame has two destination addresses, one for each of the two parts. One of the two next hop nodes delays forwarding the packet for one time slot, in order to avoid a collision when both parts are forwarded further.

The authors also discuss the applicability of their mechanism to generic ad-hoc networks. Although no results for other network systems than FRN are given, they consider the technique also useful in other environments, given that the composition and decomposition of frames containing multiple packets is feasible.

VII. Limiting TCP's Packet Output

Fu et al. showed in [24] that a small TCP congestion window can have beneficial effects on the performance in mobile ad-hoc networks (see section V). Following that, a number of approaches have been proposed that exploit more or less directly this effect.

In [12] Chen et al. take up Fu et al.'s observation. On this basis they establish a **dynamic congestion window limit** based on the broadcast characteristics of the wireless medium.

They argue that the congestion window limit (CWL) depends on the bandwidth-delay product (BDP) of the connection, and they show that the BDP cannot exceed the round-trip hop-count (RTHC) in a wireless multihop network. For the IEEE 802.11 MAC they give an even tighter bound of $\frac{\mathrm{RTHC}}{5}$.

They use DSR as a path-aware routing protocol to get information about the path length at the source node. This allows for setting the CWL dynamically depending on the path length of the connection.

To demonstrate the performance gain of their scheme ns-2 simulations have been conducted to compare it to TCP Reno with an unbounded congestion window. However, it should be noted that they also changed the maximum retransmission timeout of TCP in their simulations, setting it to 2 s as opposed to the 240 s given in RFC 1122 [7]. This might affect the simulation results.

This modification of the maximum retransmission timer is criticized by Papanastasiou and Ould-Khaoua in [42]. They also propose their own scheme, called **Slow Congestion Avoidance (SCA)**. Their approach is to limit TCP's window growth rate to a level below the standard of one segment per RTT. This is intended to reduce the number of packets in the network without putting hard constraints on the maximum window size like in CWL.

The SCA modification of the TCP window increment mechanism increases the window size by one segment after a given number of round trip times with successful acknowledgment receptions.

SCA seems to be an interesting approach, showing ways to deal with a MANET's shared channel properties without the need to use cross-layer information in the transport protocol. Further

investigation could be necessary in order to examine the properties of SCA, especially under different traffic loads.

An adaption of the TCP behavior to mobile ad-hoc networks by Nahm et al. [41] is similar to the ideas realized in SCA. They also propose to reduce the rate of the congestion window growth of TCP. They call their scheme **fractional window increment (FeW)**.

The reasoning for the growth rate limitation in FeW is the observation that TCP generally operates at a high loss rate in networks with a low bandwidth-delay product. But because losses in congested wireless networks are usually link layer losses rather than queue overflows, these losses also influence routing—the routing protocol will often assume a lost link. The FeW approach is to change TCP's operational range, in order to achieve a generally lower loss rate.

A mathematical analysis based on the TCP-friendly steady state throughput equation is used to deduce that such a shift of the operational range can be achieved by incrementing TCP's congestion window slower than in standard TCP. In practice this leads to a scheme with a non-integer increment in the window size per RTT. It is equivalent to SCA's window size increment only every n round trip times.

The promising results of SCA and FeW should definitely be taken into account by other researchers working on modified TCP variants for MANETs. However, it is not yet clear to which extent short connections with only relatively small amounts of data might suffer from the slower congestion window growth and the resulting slower convergence.

Yang et al. looked at MANETs that are connected to a wired backbone network. There, unmodified TCP exhibits severe unfairness. But in [53] they also point out that simply reducing the congestion window size will severely degrade the performance in such a setup: the small window size only allows for a small number of packets present in the network in parallel, and this affects the wired part of the network, too. As a remedy they propose a **non-work-conserving scheduling** mechanism for the wireless interfaces.

In their scheme, after transmitting a data packet on the wireless medium a timer is set. Before this timer expires no other data packet is allowed to be sent by the same node. The run time of the timer increases with a higher output data rate of the queue in the recent past. Therefore more aggressive nodes' queues are slowed down more. The authors observe greatly improved fairness in their simulations, but at the same time a significant throughput deterioration.

While the approaches described so far directly modify TCP's congestion window size, the non-work-conserving scheduling reduces the rate at which packets are allowed to be forwarded. This happens at every intermediate node and per interface queue. There are also proposals which add an additional rate based control mechanism to the output of the transport layer only in the source nodes.

In general the impact of those approaches is similar to the window size limitation schemes—the number of packets per time unit and per TCP sender is limited. However, rate-based mechanisms might be able to achieve additional benefits by reducing the inherent burstiness of TCP traffic and distributing packet transmissions more evenly. This might significantly reduce intra-flow contention. On the other hand, because multiple, stacked rate limitation and congestion control mechanisms are used the question of possible feedback-loop effects arises.

The **Rate-Based Congestion Control (RBCC)** mechanism proposed by Zhai et al. in [60] adds a leaky bucket mechanism beyond TCP's window-based rate control. In RBCC, a feedback field is added to the header, which is used by all intermediate nodes to provide feedback on the allowed maximum rate of the flow.

Each node on the route observes its local *channel busyness ratio*, defined as the fraction of time the medium is locally non-idle. It is used to modify the feedback field in the packets passing by,

in order to inform the source about the sustainable rate. The intermediate nodes aim to distribute the capacity fair between the flows. For that reason, state describing the flows passing through the local node is maintained, and an AIMD mechanism is used to converge to fairness.

Kliazovich et al. measure in **Cross-layer congestion control (C^{3}TCP)** [33] the bandwidth and delay within an end-to-end link by cumulating intermediate hops' measurements. Like in RBCC a feedback field is added to the link layer header. The measurements are gathered hop by hop from the intermediate nodes and stored in the feedback field. When an ACK is generated at the destination node the feedback included in the corresponding data packet is repeated and thus transmitted back to the sender. There it is used by an additional module—located beyond TCP in the protocol stack—to modify the receiver advertised window (rwnd) field in the ACK. In this field the receiver can limit the sender's window size. Its normal purpose is flow control, in C^{3}TCP it is used to limit the window size of the sender dynamically based on the measurements. In order to keep the TCP implementation unmodified, all C^{3}TCP logic is contained within the additional protocol module.

While RBCC and C^{3}TCP rely on the participation of intermediate nodes, ElRakabawy et al. propose a pure end-to-end approach [18]. They suggest to pace the packets allowed to be sent out by the congestion window adaptively. Their approach is called **TCP with Adaptive Pacing (TCP-AP)**. Like RBCC, TCP-AP is a hybrid between a window-based and a rate-based approach, adding rate-based mechanisms to TCP in order to avoid large bursts of packets.

As a metric to be evaluated in order to configure the pacing the *4-hop propagation delay* is defined. It describes the time between the transmission of a packet by the TCP source node and its reception by the node four hops downstream. Since TCP-AP is a pure end-to-end protocol, the 4-hop propagation delay cannot be measured directly. Instead, it is estimated using the RTT of the packets. The 4-hop propagation delay is chosen by the authors because a transmission currently in progress is assumed to interfere within a range of four hops, a number matching the common modeling assumptions as used, e. g., in the ns-2 simulator.

In addition to the 4-hop propagation delay, the coefficient of variation of RTT samples is proposed as a metric. Its purpose is to measure the degree of contention along the path. In combination with the 4-hop propagation delay it is used to establish a minimum time between two successive packet transmissions.

VIII. Alternative Protocol Designs

A variety of wireless peculiarities have been shown to be detrimental to TCP's end-to-end way of performing congestion control in mobile ad-hoc networks up to now. Consequently some researchers do not just try to trim TCP to perform better by adjusting the protocol's behavior. Instead they develop new reliable transmission protocols that are specifically tailored to cope with the characteristics of MANETs. While the authors of these approaches report to have obtain a broad range of improvements, in these approaches this necessarily comes at the cost of TCP compatibility. Moreover, most approaches are also limited to "clean" environments where no other transport protocols are used.

However, since MANETs can often be expected to be rather small, closed environments, such constraints can be perfectly reasonable. Additionally, it seems that it might even turn out to be absolutely necessary to rely on completely different queuing and congestion control paradigms than those used by TCP in networks with media properties like those of wireless multihop networks. At least there are results indicating very fundamental problems of TCP-like mechanisms in the presence of wireless interference [44].

The first representative of the alternative protocol category is the **EXACT** protocol by Chen et al. [9], [11]. EXACT is rate based and is supported by the network itself, i. e., by the intermediate

nodes. These nodes have dedicated state variables for all flows passing through them. All nodes determine their current bandwidth to their neighbors and calculate local fair bandwidth shares for all flows.

Explicit rate information is inserted into all passing packets by the intermediate nodes to transmit the minimum bandwidth at the bottleneck to the receiver of the flow. Each node checks whether the rate it can supply for the flow of a packet it processes is lower than the rate currently specified in the packet header. In this case the lower rate is written into the header before the packet is forwarded. Thus the bottleneck rate is reported in the end.

This mechanism is used twice, i. e. on two different header fields. One field contains the current rate of the sender and another one the rate requested by the sending application. On the one hand with this procedure it is possible for the intermediate routers not to give a flow more bandwidth than it needs, and on the other hand the sender is notified when it is allowed to increase its rate above the current level.

A *safety window* prevents the sender from overloading the network in case of a route failure. A sender is not allowed to have more unacknowledged packets underway than the size of the safety window.

EXACT may be used reliably (TCP-EXACT) or unreliably (UDP-EXACT). There are no retransmission timers, instead a SACK scheme with strictly monotonous increasing sequence numbers is used: when a segment not acknowledged by the receiver is too far apart (in terms of this sequence number) from the highest acknowledged segment, it is retransmitted.

Some limitations on EXACT's practical usage and scalability might be imposed by the fact that it requires explicit state information for each flow in each intermediate node.

Another protocol offers some similar properties to those of EXACT. It is also rate based and network supported. Sundaresan et al. tailored it to the specific needs in MANETs and called it **Ad-hoc Transport Protocol (ATP)** [48]. It does not use retransmission timeouts, strictly separates congestion control and reliability mechanisms and requires only limited feedback from the receiver. In contrast to EXACT, ATP does not require any flow-specific state variables in the intermediate nodes. All nodes calculate an exponential average of the delay of all packets passing through them. This delay consists of the time a packet had to wait in the node's local queue and of the time to wait for a free medium before it could be transmitted. These values are independent of the flows the packets belongs to.

Like the rate information in EXACT, the current delay value is piggybacked onto forwarded data packets if it is worse than the information currently in the packet's header. This way, the maximum delay over the packet's path is communicated to the receiver. The receiver aggregates this information and sends it back to the sender. Based on this information the sender can adapt its rate.

To find a good rate at the start of a new connection a probe packet is sent along the route collecting information from the intermediate nodes about the current state of the network. For the acknowledgments of data packets a selective ACK scheme is employed in ATP. It uses large SACK blocks and therefore requires only few feedback packets.

A second approach which is also called ATP, the **Application controlled Transport Protocol** by Liu and Singh [37], is based on the observation that TCP's throughput in MANETs is very low, while UDP achieves reasonable throughput, but suffers from a high packet loss rate. ATP is meant to be somewhere in between TCP and UDP—UDP with optional packet delivery status feedback.

The protocol supports packet acknowledgments, feedback is given to the application if an acknowledgment for a given packet has arrived or not. An application using ATP is expected to do retransmissions on its own, if they are necessary.

Leaving the decision upon whether retransmissions are necessary or not to the applications is an interesting approach to reducing the number of retransmitted packets. However, no other transport layer components are implemented, especially congestion control would have to be provided by the application itself. Therefore ATP's approach alone can probably not be sufficient to save a MANET from severe congestion problems.

Not a complete redesign, but a variation of the XCP [30] transport protocol for wired networks with high bandwidth-delay product is the **Wireless eXplicit Congestion control Protocol (WXCP)** by Su and Gross [45]. Although it shares some fundamental concepts with TCP, XCP is not compatible with standard TCP. WXCP uses explicit feedback from within the network and multiple congestion metrics. These are evaluated at the intermediate nodes, in order to avoid the necessity of probing for the highest available bandwidth.

The metrics used in each WXCP-enabled network node are the locally available bandwidth, the length of the local interface queue and the average number of required link layer retransmissions. The latter is specifically meant to help detecting self-interference within a flow, that is, packets belonging to the same flow contending for medium access in the same collision domain. The aggregate feedback is a function of the three metrics, weighting their relative influence.

Congestion and fairness control decisions are made separately in WXCP. The fairness controller tries to achieve time fairness instead of throughput fairness among flows, since throughputs of different links may not be the same in wireless networks. Time fairness, guaranteeing all flows equal medium access time as opposed to equal throughputs, is thus regarded as a better fairness metric by the authors.

WXCP is a window-based approach with some rate-based elements in it. The sender may switch from the window-based default to a slow rate-based control mechanism, if otherwise no further packets were allowed to be transmitted—due to a small congestion window and missing ACKs or duplicate ACKs. This is called the *discovery state*. It allows the sender to continuously examine the current packet loss pattern.

Additionally a pacing mechanism introduces rate-based ideas into WXCP. If the number of packets allowed to be sent out by the window mechanism exceeds a certain limit, the packets are paced to be evenly distributed over an RTT interval.

A transport protocol for mobile ad-hoc networks developed from scratch by Anastasi et al. is **TPA** [4]. Its congestion control mechanism is inspired by TCP, but designed to minimize the number of required packet retransmissions.

Packets are transmitted in *blocks* using a window-based scheme. A fixed number of packets is grouped into a block and transmitted reliably to the destination before any packet of the next block is transmitted. Packet retransmissions are not performed before every packet of a block has been transmitted once—thus a block is transmitted in several rounds: first every packet is transmitted once, then not yet acknowledged packets of this block are retransmitted until every packet of the block has been delivered and acknowledged.

If an ELFN mechanism is available, TPA can make use of it and enters a freeze state upon route failures, decreasing the window size to one. If ELFN is not available, TPA detects route failures by a number of consecutive timeouts. Like TCP it uses an estimate of the RTT to set the retransmission timeout. In case of route changes, new RTT values are given a greater weight in the sliding average in order to speed up the convergence against a correct new RTT measure.

For congestion control TPA uses a window mechanism with a tightly limited maximum window size. Actually, only two different cwnd values are used: a "large" window of 2 or 3 segments during normal operation and the minimum value of 1 when congestion is detected.

TPA shows that even a quite simple end-to-end protocol without additional intelligence in the intermediate nodes has the potential to increase throughput in comparison to TCP. However, it is not yet clear if these benefits can be maintained in more complex, dynamic scenarios. Additionally, for time critical applications the higher latency introduced by the protocol might be a problem.

IX. Conclusions

A look at the approaches proposing improved transport layers for mobile ad-hoc networks shows two major trends: on the one hand there is a quite large number of protocols which try to achieve improvements while maintaining compatibility with established, wide-spread protocols, mainly with TCP. They also try to avoid any cross-layer feedback in order to stay independent from any specific network stack below. On the other hand many approaches willingly sacrifice compatibility to gain more freedom in protocol design and hence to even better fit the specific needs of MANETs.

Both attempts to address the problems have their respective application areas. When heterogeneous networks—like wired-cum-wireless scenarios—are used, a wireless-centered protocol or a protocol depending on wireless-specific MAC layer feedback is not suitable. In contrast, in a completely wireless network with few, well-known and potentially newly developed applications a custom approach providing superior performance might be the best choice. In addition numerous intermediate scenarios can be imagined, where the design of the overall system architecture might depend largely on how much additional performance can be realized by choosing a non-standard protocol at the cost of a higher deployment and maintenance effort.

Another striking fact is that a number of design patterns occur in multiple protocols. These common ideas and concepts have either been developed in multiple contexts or have been adopted from earlier approaches. In Table III we show these patterns and identify the approaches they appear in.

X. Opportunities for Future Research

Because there are no comparative studies of protocols maintaining compatibility versus cross-layer or totally re-designed protocols it is hard to tell how much performance gain is possible at what cost. Studying this aspect in more detail than done up to now might lead to considerable insight and might help to decide which direction future research should put its main focus on.

Especially in the field of non-TCP compatible, network supported protocols there is still a lot of potential for the development of new protocol designs. Apart from the current, rate-based approaches, there might be fundamentally different ideas to be explored. In contrast to wired networks the routers in MANETs—that is, the intermediate nodes—can often be expected to have significant memory and computational power in relation to the available network bandwidth. Also, in many application scenarios a certain homogeneity of the nodes can be assumed. This offers many opportunities for a protocol design to exploit both cross-layer information and sophisticated support by the intermediate hops. Therefore in this field there is a large variety of potentially very innovative protocol design options.

Currently congestion control, as considered in this survey, is often seen as a transport layer issue. It is therefore very often combined with reliability mechanisms, in a TCP-like way. Therefore—like in the Internet—congestion control for unreliable, UDP-like traffic is not possible. A new perspective on this problem might be to realize congestion control in the MAC or network layer. After all, it might make sense to tackle the problem where it emerges. An exeedingly high network load is a problem closely associated with medium access and packet forwarding. Some approaches

already follow this direction and separate congestion control strictly from reliability measures. Here a wide spectrum is open for more fundamental research.

Furthermore, new applications especially developed for MANETs need different types of protocols. As mentioned before, there are many cases where the protocols will necessarily be extremely application specific. Examples are the emerging applications for MANETs in the field of car-to-car communication: this demands very specific, nongeneric congestion control and reliability mechanisms. Warning messages need to be sent fast and it has to be assured that every car driving towards an incident will be warned. Moreover, new applications in MANETs will require communication paradigms like multicast or geocast. In wired networks multicast is used very rarely, because it is often not supported by the network. But in mobile ad-hoc networks, where the network can be tailored to the application and bandwidth is especially scarce, it might in fact turn out to be vital for group communication scenarios. Congestion control for these non-unicast communication scenarios is also an open research issue.

REFERENCES

[1] M. Allman, V. Paxson, and W. Stevens. TCP Congestion Control. RFC 2581 (Proposed Standard), Apr. 1999. Updated by RFC 3390.

[2] E. Altman and T. Jiménez. Novel Delayed ACK Techniques for Improving TCP Performance in Multihop Wireless Networks. In *PWC '03: Proceedings of the IFIP-TC6 8th International Conference on Personal Wireless Communications*, pages 237–250, Sept. 2003. DOI: 10.1007/b12004.

[3] V. Anantharaman, S.-J. Park, K. Sundaresan, and R. Sivakumar. TCP performance over mobile ad-hoc networks: a quantitative study. *Wireless Communications and Mobile Computing*, 4(2):203–222, 2004. DOI: 10.1002/wcm.172.

[4] G. Anastasi, E. Ancillotti, M. Conti, and A. Passarella. TPA: A Transport Protocol for Ad Hoc Networks. In *ISCC '05: Proceedings of the 10th IEEE International Symposium on Computers and Communication*, pages 51–56, June 2005. DOI: 10.1109/ISCC.2005.147.

[5] D. Berger, Z. Ye, P. Sinha, S. Krishnamurthy, M. Faloutsos, and S. K. Tripathi. TCP-Friendly Medium Access Control for Ad-Hoc Wireless Networks: Alleviating Self-Contention. In *Proceedings of the 1st IEEE International Conference on Mobile Ad-hoc and Sensor Systems*, pages 214–223, Fort Lauderdale, Florida, USA, Oct. 2004.

[6] S. Biaz and N. H. Vaidya. "De-randomizing" congestion losses to improve TCP performance over wired-wireless networks. *IEEE/ACM Transactions on Networking*, 13(3):596–608, 2005. DOI: 10.1109/TNET.2005.850205.

[7] R. Braden. Requirements for Internet Hosts - Communication Layers. RFC 1122 (Standard), Oct. 1989. Updated by RFCs 1349, 4379.

[8] K. Chandran, S. Raghunathan, S. Venkatesan, and R. Prakash. A Feedback Based Scheme for Improving TCP Performance in Ad-Hoc Wireless Networks. In *ICDCS '98: Proceedings of the 18th International Conference on Distributed Computing Systems*, pages 472–479, May 1998.

[9] K. Chen and K. Nahrstedt. EXACT: An Explicit Rate-based Flow Control Framework in MANET (extended version). Technical Report UIUCDCS-R-2002-2286/UILU-ENG-2002-1730, Department of Computer Science, University of Illinois at Urbana-Champaign, July 2002.

[10] K. Chen and K. Nahrstedt. Limitations of Equation-Based Congestion Control in Mobile Ad Hoc Networks. In *ICDCSW '04: Proceedings of the 24th International Conference on Distributed Computing Systems Workshops - W7: EC*, pages 756–761, 2004. DOI: 10.1109/ICDCSW.2004.1284118.

[11] K. Chen, K. Nahrstedt, and N. Vaidya. The Utility of Explicit Rate-based Flow Control in Mobile Ad Hoc Networks. In *WCNC '04: Proceedings of the IEEE Wireless Communications and Networking Conference*, volume 3, pages 1921–1926, Mar. 2004.

[12] K. Chen, Y. Xue, and K. Nahrstedt. On Setting TCP's Congestion Window Limit in Mobile Ad Hoc Networks. In *ICC '03: Proceedings of the IEEE International Conference on Communications*, Anchorage, Alaska, May 2003. DOI: 10.1109/ICC.2003.1204525.

[13] C. de M. Cordeiro, S. R. Das, and D. P. Agrawal. COPAS: Dynamic Contention-Balancing to Enhance the Performance of TCP over Multi-hop Wireless Networks. In *Proceedings of the 10th International Conference on Computer Communication and Networks (IC3N)*, pages 382–387, Miami, FL, USA, Oct. 2002.

[14] R. de Oliveira and T. Braun. A Delay-based Approach Using Fuzzy Logic to Improve TCP Error Detection in Ad Hoc Networks. In *WCNC '04: Proceedings of the IEEE Wireless Communications and Networking Conference*, volume 3, pages 1666–1671, Mar. 2004.

[15] R. de Oliveira and T. Braun. A Dynamic Adaptive Acknowledgment Strategy for TCP over Multihop Wireless Networks. In *INFOCOM '05: Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 3, pages 1863–1874, Mar. 2005. DOI: 10.1109/INFCOM.2005.1498465.

[16] R. de Oliveira, T. Braun, and M. Heissenbüttel. An Edge-based Approach for Improving TCP in Wireless Mobile Ad Hoc Networks. In *DADS '03: Proceedings of the Conference on Design, Analysis and Simulation of Distributed Systems*, Orlando, USA, Mar. 2003.

[17] T. D. Dyer and R. V. Boppana. A Comparison of TCP Performance over Three Routing Protocols for Mobile Ad Hoc Networks. In *MobiHoc '01: Proceedings of the 2nd ACM International Symposium on Mobile Ad Hoc Networking & Computing*, pages 56–66, 2001. DOI: 10.1145/501422.501425.

[18] S. M. ElRakabawy, A. Klemm, and C. Lindemann. TCP with Adaptive Pacing for Multihop Wireless Networks. In *MobiHoc '05: Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 288–299, May 2005. DOI: http://doi.acm.org/10.1145/1062689.1062726.

[19] S. Floyd, M. Handley, J. Padhye, and J. Widmer. Equation-Based Congestion Control for Unicast Applications. *ACM SIGCOMM Computer Communication Review*, 30(4):43–56, 2000. DOI: http://doi.acm.org/10.1145/347057.347397.

[20] S. Floyd and V. Jacobson. Random Early Detection Gateways for Congestion Avoidance. *IEEE/ACM Transactions on Networking*, 1(4):397–413, 1993. DOI: 10.1109/90.251892.

[21] Z. Fu, B. Greenstein, X. Meng, and S. Lu. Design and Implementation of a TCP-Friendly Transport Protocol for Ad Hoc Wireless Networks. In *ICNP '02: Proceedings of the 10th IEEE International Conference on Network Protocols*, pages 216–225, Nov. 2002.

[22] Z. Fu, X. Meng, and S. Lu. How Bad TCP Can Perform In Mobile Ad Hoc Networks. In *ISCC '02: Proceedings of the 7th IEEE International Symposium on Computers and Communication*, pages 298–303, July 2002. DOI: 10.1109/ISCC.2002.1021693.

[23] Z. Fu, X. Meng, and S. Lu. A Transport Protocol for Supporting Multimedia Streaming in Mobile Ad-hoc Networks. *IEEE Journal on Selected Areas in Communications*, 21(10):1615–1626, Dec. 2003. DOI: 10.1109/JSAC.2003.816461.

[24] 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 *INFOCOM '03: Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 3, pages 1744–1753, Apr. 2003.

[25] T. Goff, N. B. Abu-Ghazaleh, D. S. Phatak, and R. Kahvecioglu. Preemptive Routing in Ad Hoc Networks. *Elsevier Journal of Parallel and Distributed Computing*, 63(2):123–140, Feb. 2003. DOI: 10.1016/S0743-7315(02)00059-X.

[26] M. Güneş and D. Vlahovic. The Performance of the TCP/RCWE Enhancement for Ad-Hoc Networks. In *ISCC '02: Proceedings of the 7th IEEE International Symposium on Computers and Communication*, pages 43–48, 2002. DOI: 10.1109/ISCC.2002.1021656.

[27] G. Holland and N. H. Vaidya. Analysis of TCP Performance Over Mobile Ad Hoc Networks. In *MobiCom '99: Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, pages 219–230, Aug. 1999. DOI: http://doi.acm.org/10.1145/313451.313540.

[28] V. Jacobson and M. J. Karels. Congestion Avoidance and Control. *Proceedings of SIGCOMM '88*, 18(4):314–329, 1988. DOI: http://doi.acm.org/10.1145/205447.205462.

[29] D. B. Johnson and D. A. Maltz. Dynamic Source Routing in Ad Hoc Wireless Networks. In T. Imielinski and H. Korth, editors, *Mobile Computing*, volume 353, chapter 5, pages 153–181. Kluwer Academic Publishers, 1996.

[30] D. Katabi, M. Handley, and C. Rohrs. Congestion control for high bandwidth-delay product networks. In *SIGCOMM '02: Proceedings of the 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pages 89–102, New York, NY, USA, 2002. ACM Press. DOI: http://doi.acm.org/10.1145/633025.633035.

[31] D. Kim, C.-K. Toh, and Y. Choi. TCP-Bus: Improving TCP Performance in Wireless Ad-Hoc Networks. In *ICC '00: Proceedings of the IEEE International Conference on Communications*, volume 3, pages 1707–1713, June 2000. DOI: 10.1109/ICC.2000.853785.

[32] F. Klemm, Z. Ye, S. V. Krishnamurthy, and S. K. Tripathi. Improving TCP Performance in Ad Hoc Networks using Signal Strength based Link Management. *Elsevier Ad Hoc Networks*, 3(2):175–191, 2005. DOI: 10.1016/j.adhoc.2004.07.005.

[33] D. Kliazovich and F. Granelli. Cross-layer congestion control in ad hoc wireless networks. *Ad Hoc Networks*, 4(6):687–708, Nov. 2006. DOI: 10.1016/j.adhoc.2005.08.001.

[34] S. Kopparty, S. Krishnamurthy, M. Faloutsos, and S. Tripathi. Split-TCP for Mobile Ad Hoc Networks. In *GLOBECOM '02: Proceedings of the IEEE Global Telecommunications Conference*, volume 1, pages 138–142, Nov. 2002.

[35] M. Li, C.-S. Lee, E. Agu, M. Claypool, and R. Kinicki. Performance Enhancement of TFRC in Wireless Ad Hoc Networks. In *DMS '04: Proceedings of the 10th International Conference on Distributed Multimedia Systems*, Sept. 2004.

[36] H. Lim, K. Xu, and M. Gerla. TCP Performance over Multipath Routing in Mobile Ad Hoc Networks. In *ICC '03: Proceedings of the IEEE International Conference on Communications*, volume 2, pages 1064–1068, May 2003.

[37] J. Liu and S. Singh. ATP: Application controlled Transport Protocol for Mobile Ad Hoc Networks. In *WCNC '99: Proceedings of the IEEE Wireless Communications and Networking Conference*, volume 3, pages 1318–1322, Sept. 1999. DOI: 10.1109/WCNC.1999.796951.

[38] J. Liu and S. Singh. ATCP: TCP for Mobile Ad Hoc Networks. *IEEE Journal on Selceted Areas in Communications (J-SAC)*, 19(7):1300–1315, July 2001. DOI: 10.1109/49.932698.

[39] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow. TCP Selective Acknowledgment Options. RFC 2018 (Proposed Standard), Oct. 1996.

[40] J. P. Monks, P. Sinha, and V. Bharghavan. Limitations of TCP-ELFN for Ad Hoc Networks. In *MoMuC '00: Proceedings of the 7th IEEE International Workshop on Mobile Multimedia Communications*, 2000.

[41] K. Nahm, A. Helmy, and C.-C. J. Kuo. TCP over Multihop 802.11 Networks: Issues and Performance Enhancement. In *MobiHoc '05: Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 277–287, 2005. DOI: http://doi.acm.org/10.1145/1062689.1062725.

[42] S. Papanastasiou and M. Ould-Khaoua. TCP congestion window evolution and spatial reuse in MANETs. *Journal of Wireless Communications and Mobile Computing*, 4(6):669–682, Sept. 2004. DOI: 10.1002/wcm.236.

[43] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc On-Demand Distance Vector (AODV) Routing. RFC 3561 (Experimental), July 2003.

[44] V. Raghunathan and P. R. Kumar. A Counterexample in Congestion Control of Wireless Networks. In *MSWiM '05: Proceedings of the 8th ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, pages 290–297, Oct. 2005. DOI: http://doi.acm.org/10.1145/1089444.1089496.

[45] Y. Su and T. Gross. WXCP: Explicit Congestion Control for Wireless Multi-hop Networks. In *IWQoS '05: Proceedings of the 12th International Workshop on Quality of Service*, June 2005.

[46] M. Sugano and M. Murata. Performance Improvement of TCP on a Wireless Ad Hoc Network. In *VTC '03-Spring: Proceedings of the 57th IEEE Vehicular Technology Conference*, volume 4, pages 2276–2280, Apr. 2003. DOI: 10.1109/VETECS.2003.1208794.

[47] D. Sun and H. Man. ENIC – An Improved Reliable Transport Scheme for Mobile Ad Hoc Networks. In *GLOBECOM '01: Proceedings of the IEEE Global Telecommunications Conference*, volume 5, pages 2852–2856, Nov. 2001. DOI: 10.1109/GLOCOM.2001.965950.

[48] K. Sundaresan, V. Anantharaman, H.-Y. Hsieh, and R. Sivakumar. ATP: A Reliable Transport Protocol for Ad-hoc Networks. In *MobiHoc '03: Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking & Computing*, pages 64–75, June 2003. DOI: http://doi.acm.org/10.1145/778415.778424.

[49] K. Tang and M. Gerla. Fair Sharing of MAC under TCP in Wireless Ad Hoc Networks. In *MMT '99: Proceedings of the Workshop on Multiaccess, Mobility and Teletraffic for Wireless Communications*, Oct. 1999.

[50] F. Wang and Y. Zhang. Improving TCP Performance over Mobile Ad-Hoc Networks with Out-of-Order Detection and Response. In *MobiHoc '02: Proceedings of the 3rd ACM International Symposium on Mobile Ad Hoc Networking & Computing*, pages 217–225, 2002. DOI: http://doi.acm.org/10.1145/513800.513827.

[51] K. Xu, M. Gerla, L. Qi, and Y. Shu. TCP Unfairness in Ad Hoc Wireless Networks and a Neighborhood RED Solution. *Wireless Networks*, 11(4):383–399, July 2005. DOI: 10.1007/s11276-005-1764-1.

[52] S. Xu and T. Saadawi. Does the IEEE 802.11 MAC protocol work well in multihop wireless ad hoc network. *IEEE Communications Magazine*, 39(6):130–137, June 2001. DOI: 10.1109/35.925681.

[53] L. Yang, W. K. G. Seah, and Q. Yin. Improving Fairness among TCP Flows crossing Wireless Ad Hoc and Wired Networks. In *MobiHoc '03: Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking & Computing*, pages 57–63, New York, NY, USA, June 2003. ACM Press. DOI: http://doi.acm.org/10.1145/778415.778423.

[54] Z. Ye, S. Krishnamurthy, and S. Tripathi. Use of Congestion-Aware Routing to Spatially Separate TCP Connections in Wireless Ad Hoc Networks. In *Proceedings of the 1st International Conference on Mobile Ad hoc and Sensor Systems (MASS)*, Fort Lauderdale, FL, USA, Oct. 2004.

[55] X. Yu. Improving TCP Performance over Mobile Ad Hoc Networks by Exploiting Cross-Layer Information Awareness. In *MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking*, pages 231–244, New York, NY, USA, 2004. ACM Press. DOI: http://doi.acm.org/10.1145/1023720.1023743.

[56] T. Yuki, T. Yamamoto, M. Sugano, M. Murata, H. Miyahara, and T. Hatauchi. Improvement of TCP Throughput by Combination of Data and ACK Packets in Ad Hoc Networks. *IEICE Transactions on Communications*, 87(9):2493–2499, Sept. 2004.

[57] L. A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.

[58] L. A. Zadeh. Fuzzy algorithms. *Information and Control*, 12:94–102, 1968.

[59] H. Zhai, X. Chen, and Y. Fang. Alleviating Intra-Flow and Inter-Flow Contentions for Reliable Service in Mobile Ad Hoc Networks. In *MILCOM '04: Proceedings of the IEEE Military Communications Conference*, volume 3, pages 1640–1646, Oct. 2004. DOI: 10.1109/MILCOM.2004.1495184.

[60] H. Zhai, X. Chen, and Y. Fang. Rate-Based Transport Control for Mobile Ad Hoc Networks. In *WCNC '05: Proceedings of the IEEE Wireless Communications and Networking Conference*, volume 4, pages 2264–2269, Mar. 2005. DOI: 10.1109/WCNC.2005.1424868.

[61] H. Zhai, J. Wang, and Y. Fang. Distributed Packet Scheduling for Multihop Flows in Ad Hoc Networks. In *WCNC '04: Proceedings of the IEEE Wireless Communications and Networking Conference*, volume 2, pages 1081–1086, Mar. 2004.

[62] J. Zhou, B. Shi, and L. Zou. Improve TCP performance in Ad hoc network by TCP-RC. In *PIMRC '03: Proceedings of the 14th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, volume 1, pages 216–220, Sept. 2003. DOI: 10.1109/PIMRC.2003.1264264.

Christian Lochert received his Diploma degree in Information Systems in 2003 from the University of Mannheim, Germany. Currently he is a Ph. D. student at the Heinrich Heine University, Düsseldorf, Germany. His current research interests include vehicular ad-hoc networks and congestion control in mobile ad-hoc networks.

Björn Scheuermann received his B. S. degree in Mathematics and Computer Science in 2004 and his M. S. in Computer Science in the same year, both from the University of Mannheim, Germany. Since 2005, he is a Ph. D. student at the Heinrich Heine University, Düsseldorf, Germany. His current research interests include congestion control in mobile ad-hoc networks and vehicular ad-hoc networks.

Martin Mauve received the M. S. and Ph. D. degrees in Computer Science from the University of Mannheim, Germany, in 1997 and 2000 respectively. From 2000 to 2003, he was an Assistant Professor at the University of Mannheim. In 2003, he joined the Heinrich Heine University, Düsseldorf, Germany, as a Full Professor and Head of the research group for computer networks and communication systems. His research interests include distributed multimedia systems, multimedia transport protocols, mobile ad-hoc networks and inter-vehicle communication.

TABLE III

COMMON DESIGN PATTERNS IN CONGESTION CONTROL MECHANISMS FOR MANETS

	Approach	Ref.	cross-layer	link failure notification	SACK	DACK	TCP param. recalc.	ECN	route failure pred.	adaptive routing	multiple paths	fixed RTO	cwnd/rate limit	multiple metrics	loss type dist.	adapt. backoff	early dropping (RED)	medium overhearing	frame combining	interm. node feedback
Route Failures	TCP-F	[8]	■	■	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·
	ELFN	[27]	■	■	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·
	TCP-Bus	[31]	■	■	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·
	Fixed-RTO	[17]	·	·	·	·	·	·	·	·	·	■	·	·	·	·	·	·	·	·
	ENIC	[47]	■	■	■	■	■	·	·	·	·	·	·	·	·	·	·	·	·	·
	TCP-RC	[62]	■	■	·	·	■	·	·	·	·	·	·	·	·	·	·	·	·	·
	ATCP	[38]	■	■	·	·	·	■	·	·	·	·	·	·	·	·	·	·	·	·
	Cross-layer IA	[55]	■	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·
	TCP-DOOR	[50]	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·
	Preempt. routing	[25]	■	·	·	·	·	■	·	·	·	·	·	·	·	·	·	·	·	·
	Atra	[3]	■	■	·	·	·	■	■	·	·	·	·	·	·	·	·	·	·	·
	S.-strength based LM	[32]	■	·	·	·	·	■	·	·	·	·	·	·	·	·	·	·	·	·
	Multipath-TCP	[36]	·	·	·	·	·	·	·	·	■	■	·	·	·	·	·	·	·	·
Wireless Losses	TCP/RCWE	[26]	■	■	·	·	·	·	·	·	·	·	■	·	■	·	·	·	·	·
	ADTCP	[21]	·	·	·	·	·	·	·	·	·	·	·	■	■	·	·	·	·	·
	ADTFRC	[23]	·	·	·	·	·	·	·	·	·	·	·	■	■	·	·	·	·	·
	Edge-based	[14], [16]	·	·	·	·	·	·	·	·	·	·	·	·	■	·	·	·	·	·
Shared Medium	LRED / adaptive p.	[24]	■	·	·	·	·	·	·	·	·	·	■	·	·	■	■	■	·	·
	NRED	[51]	■	·	·	·	·	·	·	·	·	·	·	·	·	·	■	■	·	·
	COPAS	[13]	■	·	·	·	·	·	·	■	·	·	·	·	·	·	·	·	·	·
	CAR	[54]	■	·	·	·	·	·	·	■	·	·	·	·	·	·	·	·	·	·
	Split-TCP	[34]	■	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·
	Allev. self-cont.	[5]	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	■	·
	OPET	[59], [61]	■	·	·	·	·	·	·	·	·	·	■	·	·	■	·	·	·	·
	RE-TFRC	[35]	·	·	·	·	·	·	·	·	·	·	■	·	·	·	·	·	·	·
ACK Traffic	Dyn. del. ACK	[2]	·	·	·	■	·	·	·	·	·	·	·	·	·	·	·	·	·	·
	Dyn. adapt. ACK	[15]	·	·	·	■	·	·	·	·	·	·	·	·	·	·	·	·	·	·
	Pref. ACK retr.	[46]	■	■	·	■	·	·	·	·	·	·	·	·	■	·	·	·	·	·
	Comb. data and ACK	[56]	■	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	■	·
Output Limit	Dyn. cwnd limit	[12]	·	·	·	·	·	·	·	·	■	■	·	·	·	·	·	·	·	·
	SCA	[42]	·	·	·	·	·	·	·	·	·	■	·	·	·	·	·	·	·	·
	FeW	[41]	·	·	·	·	·	·	·	·	·	■	·	·	·	·	·	·	·	·
	Non-work-c. sched.	[53]	·	·	·	·	·	·	·	·	·	·	·	·	■	·	·	·	·	·
	RBCC	[60]	■	·	·	·	·	·	·	·	·	■	·	·	·	·	■	·	■	
	C^3TCP	[33]	■	·	■	■	·	·	·	·	·	■	·	·	·	·	·	·	■	
	TCP-AP	[18]	·	·	·	·	·	·	·	·	·	■	·	·	·	·	·	·	·	·
Alternative Protocol	EXACT	[9], [11]	■	·	■	·	·	·	·	·	·	·	·	·	·	·	·	·	■	
	ATP (Sundaresan)	[48]	■	·	■	·	·	·	·	·	·	·	·	·	·	·	·	·	■	
	ATP (Liu)	[37]	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·
	WXCP	[45]	■	·	·	·	·	·	·	·	·	·	■	■	·	·	·	·	■	
	TPA	[4]	·	■	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·	·