Practical Rate-based Congestion Control for Wireless Mesh Networks

Sherif M. ElRakabawy and Christoph Lindemann

University of Leipzig, Department of Computer Science,
Johannisgasse 26, 04103 Leipzig, Germany

Abstract. We introduce an adaptive pacing scheme to overcome the drawbacks of TCP in wireless mesh networks with Internet connectivity. The pacing scheme is implemented at the wireless TCP sender as well as at the mesh gateway, and reacts according to the direction of TCP flows running across the wireless network and the Internet. TCP packets are transmitted rate-based within the TCP congestion window according to the current out-of-interference delay and the coefficient of variation of recently measured round-trip times. Opposed to the majority of previous work which builds on simulations, we implement a Linux prototype of our approach and evaluate its feasibility in a real 20-node mesh testbed. In an experimental performance study, we compare the goodput and fairness of our approach against the widely deployed TCP NewReno. Experiments show that our approach, which we denote as Mesh Adaptive Pacing (MAP), can achieve up to 150% more goodput than TCP NewReno and significantly improves fairness between competing flows. MAP is incrementally deployable since it is TCP-compatible, does not require cross-layer information from intermediate nodes along the path, and requires no modifications in the wired domain.

1 Introduction

In recent years, wireless mesh networks (e.g. [2], [9], [13]) have been within the focus of research in the networking community. Wireless mesh networks typically aim to provide cost-efficient Internet access with minimal infrastructure expenditure, which makes it particularly attractive for suburban areas with little or no broadband availability. Mesh nodes, which are typically wireless routers mounted on buildings, form a multihop backbone to forward packets hop-by-hop between the Internet and other mesh nodes. Mesh participants with mesh gateways (MGs), which have direct access to the Internet, can share it with other participants.

Within several research topics in wireless mesh networks, TCP performance has acquired great attention. Wireless mesh networks using IEEE 802.11 namely possess several properties, which are different to the wired Internet for which the widely deployed TCP NewReno implementation has been optimized. Opposed to wired networks, in IEEE 802.11 networks, the wireless channel is a scarce resource shared among nodes within their radio range. Furthermore, channel capture, hidden and exposed terminal effects [8], and the IEEE 802.11 medium access control constitute features of wireless mesh networks not present in a wired IP network. Since the congestion control of TCP NewReno is based on lost data packets, the size of its congestion window is overshooting rather than proactively sensing incipient congestion by monitoring the network traffic. Hence, TCP NewReno possesses quite poor performance in wireless mesh networks, as well as exhibits severe unfairness among competing TCP flows.

Besides TCP improvements for the Internet [16], a few approaches (e.g. [6], [9], and [21]) have been proposed for improving TCP performance in wireless mesh networks with Internet connectivity. Unfortunately, most of these approaches have been only designed and evaluated in network simulators. In this paper, we introduce and evaluate a feasible approach for improving TCP goodput and fairness in a real wireless mesh testbed rather than in simulators. We introduce a rate-based congestion control algorithm for TCP over real IEEE 802.11 mesh networks, implementing rate-based scheduling of transmissions within the TCP congestion window. We propose to distinguish the direction of the TCP flow: For wired-to-wireless TCP flows, we introduce an adaptive pacing scheme at the mesh gateway. For wireless-to-wired flows, we propose an adaptive pacing scheme at the TCP sender. The proposed approach is denoted as *Mesh Adaptive Pacing (MAP)*. In a performance study, we show that, depending on the current network state and traffic patterns, MAP can achieve up to 150% more goodput than the widely deployed TCP NewReno, and significantly improves fairness between competing flows. Opposed to previous proposals for improving TCP over IEEE 802.11 mesh networks, MAP does neither rely on modifications at the routing or the link layer nor requires cross-layer information from intermediate nodes along the path.

The remainder of this paper is organized as follows. Section 2 summarizes related work on TCP for wireless mesh networks, while Section 3 describes the miniaturized wireless mesh testbed which we have built for evaluating our approach. Section 4 introduces the Mesh Adaptive Pacing scheme. An experimental performance study of MAP versus TCP NewReno using our wireless mesh testbed is given in Section 5. Finally, concluding remarks are given.

2 Related Work

ElRakabawy et al. ([5], [6]) and Wei et al. [20] showed that pacing for TCP can improve goodput and fairness both for wired as well as for wireless multihop networks. The authors in [20] found out that pacing yields reduced burstiness of traffic, increased synchronization among flows as well as fragmented SACK blocks in a flow. In [5] and [6], TCP with Adaptive Pacing and Gateway Adaptive Pacing were introduced and evaluated using ns-2 [7]. The results showed that adaptive pacing yields significant performance improvement with respect to standard TCP. Opposed to [20], our approach is tailored for wireless mesh networks and not for the Internet, which possesses fundamentally different characteristics. Beyond [5], MAP supports flows between hosts in the Internet and wireless mesh nodes. Furthermore, opposed to [5] and [6], we evaluate our approach in a real mesh testbed rather than only in simulations.

In [17], Scheuermann et al. proposed a novel hop-by-hop congestion control protocol for multihop wireless networks. In their scheme, backpressure towards the source node is established implicitly by passively observing the medium. Sundaresam et al. [19] introduced ATP, and Anastasi et al. proposed TPA [1], which are both new transport protocols for multihop wireless networks. ATP employs pure rate-based transmission of packets, where the transmission rate is determined using feedback from intermediate nodes along the path. TPA uses a similar congestion control algorithm as TCP, in such that packets are transmitted window-based. Opposed to [1], [17] and [19] we consider mesh networks with Internet connectivity rather than pure multihop wireless networks.

Yang et al. [21] proposed a pacing scheme at the IP layer to improve TCP fairness in hybrid wireless/wired networks. They derived the pacing rate by the minimum transmission delay observed for a node, the most recent transmission delay, and a random delay. In contrast to [21], MAP employs adaptive pacing rather than static pacing. Employing such an adaptive pacing scheme, MAP does not lead to unnecessary goodput degradation if there is no contention between active flows.

Gambiroza et al. [9] studied TCP performance and fairness in wireless mesh networks comprising numerous wireless relay nodes and a connection to the wired Internet. They proposed a distributed link layer algorithm for achieving fairness among active TCP flows. MAP constitutes a modification at the transport layer rather than a modification at the link layer as in [9]. In contrast to [9], MAP does not require any control traffic for achieving fairness among active TCP flows.

Opposed to [5], [6], [9], [19], and [21], our approach is tailored and evaluated in a real mesh testbed rather than simulations. This makes our approach feasible and improves the reliability of the acquired results.

3 The Leipzig Wireless Mesh Testbed

To study the performance of MAP in reality and compare it to the widely deployed TCP NewReno, we built up a miniaturized wireless mesh testbed. The testbed, which is depicted in Figure 1, comprises 20 wireless mesh nodes. Each node consists of a Siemens ESPRIMO P2510 PC with an Intel Celeron 3.2 GHz processor and two IEEE 802.11b Netgear wireless PCI network interface cards (NICs) with Atheros chipsets. Opposed to other testbeds like ORBIT [14], each wireless card is connected to a variable signal attenuator and a 2.1dBi low-gain antenna. Using the variable attenuators, the signal power of the wireless PCI cards can be adaptively shrunk in order to limit the maximum transmission range of each node. Thus, large wireless mesh networks can be scaled down to a few meters, making quick

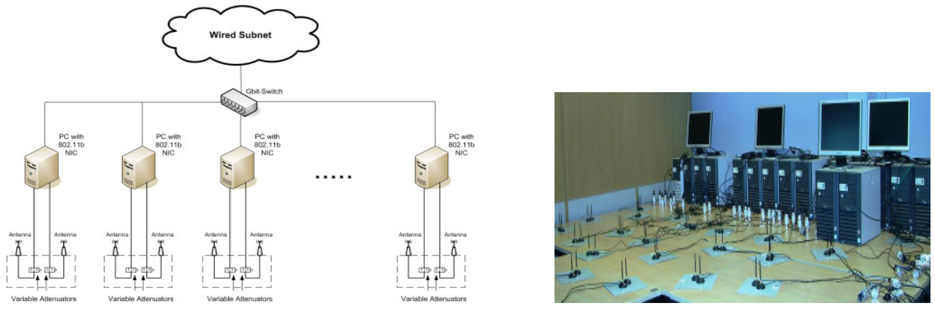


Fig 1: Leipzig wireless mesh testbed

topology and parameter modifications for efficient evaluation of network protocols possible. Besides single-radio communication, the testbed also supports dual-radio communication between mesh nodes by assigning a different channel to each of the two wireless PCI cards of a node. Testbed nodes run a SuSE Linux 10.2 operating system with a custom-compiled kernel version 2.6.18 with the high-resolution timer subsystem patch [10]. As driver for the wireless PCI cards, we employ the Linux Madwifi kernel device driver version 0.9.3.2 for Atheros chipsets. For mesh routing, we employ the Optimized Link State Routing Protocol (OLSR) version 0.5.2 for Linux [3] [12], which incorporates the ETX metric [4] for selecting routes based on the current loss probability of the links. Each wireless node further possesses a Gigabit Ethernet NIC, which is connected to the wired subnet through a Gigabit switch. This allows conducting wired-to-wireless and wireless-to-wired experiments with connections between the wired subnet and the wireless mesh domain. Thereby, any wireless mesh node can act as a mesh gateway.

4 The Mesh Adaptive Pacing Scheme

The main deficiency of TCP's congestion control algorithm over IEEE 802.11 is that it implements reactive rather than proactive congestion control. That is, a packet loss must actually occur before TCP starts throttling its transmission rate. Thus, proactive congestion control, as proposed by MAP, aims to avoid congestion before it actually occurs. Our approach adapts TCP's transmission in order to overcome the spatial reuse constraint as well as the link layer unfairness of IEEE 802.11. Thereby, the transmission rate is adjusted proactively according to the current load in the network rather than reactively after a packet loss actually occurs. Since our approach is only implemented at the wireless TCP sender and above the network layer of the mesh gateway, it is fully TCP-compatible. Specifically, modifications of TCP in the wired domain or changes in IEEE 802.11 are not required.

In general, the network entity at which adaptive pacing is implemented depends on the direction of a TCP flow. We distinguish between two flow directions: wireless-to-wired flows as well as wired-to-wireless flows. While wireless-to-wired flows describe the case where a wireless node constitutes the TCP source and a host in the wired domain is the TCP destination, wired-to-wireless flows correspond to the opposite flow direction. For wireless-to-wired flows, the adaptive pacing scheme is implemented at the wireless TCP source. For wired-to-wireless flows, adaptive pacing is implemented at the mesh gateway, which is responsible for forwarding the packets to the wireless domain according to the computed adaptive pacing rate.

Subsequently, we discuss the components of MAP in detail. Thereby, we use the term *wireless source entity* to refer to the wireless TCP source in case of wireless-to-wired flows, and to the mesh gateway (MG) in case of wired-to-wireless flows. In other words, the wireless source entity is the entity in the wireless domain at which adaptive pacing is implemented for a considered TCP connection. Accordingly, the *wireless destination entity* describes the mesh gateway in case of wireless-to-wired flows, and the wireless TCP destination in case of wired-to-wireless flows. The communication between the wireless source entity and the wireless destination entity corresponds to the wireless part of a TCP connection running across the wired and the wireless domain.

4.1 Network Load Adaptation

In order to achieve fairness between competing flows, MAP adapts its transmission rate according to the current load in the network. Thus, opposed to the aggressive strategy of standard TCP, it throttles its transmission rate to share the available bandwidth with other flows contending for the same channel. MAP identifies the current load in the vicinity by measuring the current degree of contention by means of recently measured wireless RTT samples. The term *wireless RTT* denotes the round-trip time in the wireless part of the network, i.e. the time taken for a TCP packet to be transmitted between a wireless node and MG plus the time taken for the corresponding TCP ACK packet to be transmitted between MG and the wireless node. The RTT delay in the wired part of the network does not need to be considered since the adaptive pacing approach is deployed only within the wireless domain.

MAP uses the coefficient of variation of recently measured wireless round trip times, cov_{RTT}, as a key measure for the degree of the contention on the network path. This measure is given by:

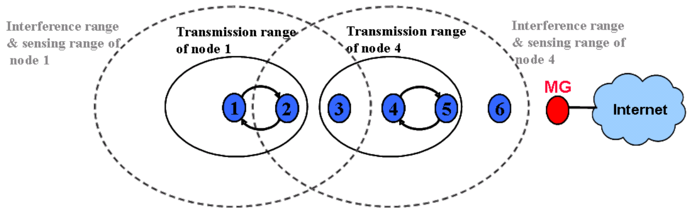
$$cov_{RTT} = \frac{\sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(RTT^{i}_{wireless} - \overline{RTT}_{wireless})^2}}{\overline{RTT}_{wireless}} \qquad (1)$$

Here, N is the number of considered wireless RTT samples, $\overline{RTT}_{wireless}$ is the mean of the samples, and $RTT^{i}_{wireless}$ denotes the value of the i-th wireless RTT sample. The coefficient of variation cov_{RTT} can be obtained purely end-to-end without provoking congestion or packet losses.

4.2 The Spatial Reuse Constraint

Besides the measure of contention on the network path, MAP also accounts for the spatial reuse constraint of IEEE 802.11 mesh networks. That is, due to the hidden terminal problem and the absence of perfect scheduling at link layer, concurrent wireless nodes in a chain cannot transmit simultaneously without causing collisions. A crucial factor that has a significant impact on the spatial reuse constraint of a mesh network is the carrier sensing range of wireless nodes. Physical carrier sensing is a mechanism incorporated in IEEE 802.11 [18], by which a wireless node senses the medium before it transmits a packet. Only if the sensed signal power is below a certain threshold, denoted as carrier sense threshold T_{cs}, does the node initiate a transmission. As the radio signal of a node attenuates with the distance, the range in which the node can sense the transmission of another node is limited. The carrier sensing range defines the range in which the current transmission of a node can be sensed by other nodes. The key role of the carrier sensing range lies in determining which hops on a chain of nodes are prone to be potential hidden terminals. That is, nodes which operate beyond each other's carrier sensing range on a chain comprise mutual hidden terminals. Thus, the transmission of each of the nodes cannot be sensed by the other node, respectively, resulting increased collision at link layer. Figure 2 shows a wireless chain of 6 nodes and a mesh gateway which is connected to the Internet. Assume a TCP connection is running between node 1 as a TCP source and a wired TCP host in the Internet as a TCP destination (i.e. a wireless-to-wired flow). We consider the wireless part of the communication, i.e. the transmission between the nodes of the chain. In this case, nodes 1 and 4 comprise mutual hidden terminals, since both nodes operate beyond each other's carrier sensing ranges. Namely, node 4 cannot sense the transmission from node 1 to node 2 and thus may transmit packets to node 5, resulting collisions with the ongoing transmission between nodes 1 and 2.

From the point of view of the TCP source, i.e. node 1, the first node which is positioned right at the border of its carrier sensing range, node 4 in this case, is the first node that comprises a potential hidden terminal. That means that collisions can be avoided if node 1 defers its transmission until node 4 finishes its transmission to node 5. Note that which node comprises the hidden terminal is mainly determined by the carrier sensing range and does not have to be the 4th node on the chain as given in Figure 2. This means that the hidden terminal varies with varying carrier sensing range. Let node i be the TCP source node and node (i+x), x ≥ 2, be the hidden terminal to node i. We refer to the time elapsed between transmitting a TCP packet by the TCP source node i and receiving the packet at node (i+x+1) as the *out-of-interference delay (OID)*. Note that the same circumstances apply for wired-to-wireless case, where MG would act as the wireless source entity, node 1 would act as the wireless destination entity, and node 4 would be the hidden terminal.
The challenge is to approximate *OID* by determining the hidden terminal for the wireless source entity node. In order to identify the hidden terminal for the wireless source entity, we have to determine the carrier sensing range in terms of number of hops. The next node right at the border of the carrier sensing range comprises the potential hidden terminal. Subsequently, we introduce the *Adaptive Out-of-Interference Delay* approach, which incorporates an effective way for estimating the carrier sensing range of the wireless source entity and approximating the out-of-interference delay accordingly.


Interference range & sensing range of node 1
Transmission range of node 1
Transmission range of node 4
Interference range & sensing range of node 4
1
2
3
4
5
6
MG
Internet
Fig. 2: Spatial reuse constraint: Hidden terminals in a chain are dependent on current carrier sensing range

4.3 Approximating the Out-of-Interference Delay

The main challenge in approximating the carrier sensing range of the wireless source entity, i.e. a wireless TCP source or the mesh gateway, lies in the lack of fundamental information such as transmission range and distance between wireless nodes. Such information can be easily inquired in simulations, but are not available in real life. As we set the preservation of the end-to-end semantics of TCP as a strict design goal, we introduce an approach for approximating the carrier sensing range purely end-to-end without any support from intermediate nodes. All parameters required for approximating the carrier sensing range are available at the wireless source entity and can be inquired from the IEEE 802.11 driver.

We approximate the carrier sensing range in terms of number of hops, not in meters, by estimating how many hops it takes for the transmission signal of the wireless source entity to get attenuated such that it falls below the carrier sensing threshold T_{cs}. That is, the first hop that comes after the threshold T_{cs} is undercut is a potential hidden terminal for the wireless source entity.

The first step towards estimating the carrier sensing range in terms of number of hops is to estimate the signal attenuation for the first hop on the path from wireless source entity to TCP destination. Let P_{out} be the actual outgoing signal power of the wireless source entity. Following the Equivalent Isotropically Radiated Power (EIRP) [15] equation we get:

$$P_{out} = P_{tx} + G_{ant} - A_{cab} - A_v \qquad (2)$$

where P_{tx} denotes the transmission power of the wireless NIC at the wireless source entity, G_{ant} denotes the signal gain of the mini antenna, and A_{cab} and A_v describe the signal attenuation caused by the coaxial cable and the variable attenuator, respectively. The parameters A_{cab} and A_v only correspond to the deployed testbed and are set to zero if no cables and/or no hardware attenuators are used in the mesh network. The signal attenuation for the first hop, L_1, is given by the difference between the received power P_{rx} at the second node in the wireless chain and the outgoing signal power from the wireless source entity, i.e. first node in the chain, P_{out}:

$$L_1 = P_{out} - P_{rx} \qquad (3)$$

The received power P_{rx} can be easily inquired from the IEEE 802.11 driver at the wireless source entity using the Received Signal Strength Indication mechanism (RSSI) [18] due to the link symmetry between the wireless source entity and the second node in the chain.

The next step is to derive an equation for estimating the signal attenuation for an arbitrary number of hops, n. Such an equation shall approximate the signal attenuation of the wireless source entity at nodes which are n hops away. The signal attenuation equation as described by the ITU-R indoor propagation model [15] is given by

$$L = 20\log_{10}(f_c) + 10p\log_{10}(d) \qquad (4)$$

where f_c denotes the frequency of the transmitted signal, i.e. a channel in the 2.4 GHz band in our case, p denotes the path loss exponent, and d describes the distance between transmitter and receiver in meters. The path loss exponent p depends on the operating environment of the wireless nodes and ranges from 2 for propagation in free space up to 5 in dense indoor environments. Due to findings from extensive measurements in our testbed and following [15], we set $p = 3$.

Let d_1 be the distance of the first hop in the chain, i.e. between the wireless source entity and the second wireless node, then we get according to Eq. 4

$$L_1 = 20\log_{10}(f_c) + 10p\log_{10}(d_1) \qquad (5)$$

Since the exact distance between the wireless nodes is unknown, we set the distance of the first n hops as $\sum_{i=1}^{n} d_i = d_1 n + \delta$, where δ determines the deviation between the distance $d_1 n$ and the actual distance of the first n hops. For the signal attenuation of the wireless source entity after n hops, L_n, we get:

$$\begin{aligned} L_n &= 20\log_{10}(f_c) + 10p\log_{10}(\textstyle\sum_{i=1}^{n} d_i) \\ &= 20\log_{10}(f_c) + 10p\log_{10}(d_1 n + \delta) \\ &= 20\log_{10}(f_c) + 10p\left(\log_{10}(d_1) + \log_{10}(n) + \log_{10}\left(1 + \frac{\delta}{d_1 n}\right)\right) \\ &= L_1 + 10p\log_{10}(n) + 10p\log_{10}\left(1 + \frac{\delta}{d_1 n}\right) \\ &= L_1 + 10p\log_{10}(n) + \varepsilon \end{aligned} \tag{6}$$

where ε describes the approximation error, which is determined by δ. Since ε is a logarithmic factor, its ratio to the overall attenuation L_n diminishes with increasing distance. In a real large-scale mesh network, $\sum_{i=1}^{n} d_i$ may well be determined more accurately, either by deploying localization techniques in IEEE 802.11 [11], or by using GPS localization. In case such localization information are available at the TCP source, an even more accurate approximation of L_n may be achieved.

Finally, we can derive the carrier sensing range H_{cs} in terms of number of hops for an h-hop chain:

$$H_{cs} = \min\{k \mid k \in \{1,2,...,h\} \wedge P_{out} - L_k < T_{cs}\} \tag{7}$$

In other words, H_{cs} is the smallest number of hops k for which the actually sensed power of the wireless source entity (i.e. P_{out} - L_k) is below the carrier sensing threshold T_{cs}. This implies that (H_{cs} + 1) is the first node in the chain which cannot sense the transmission of the wireless source entity, and thus comprises a potential hidden terminal.

By means of the estimated carrier sensing range H_{cs} as well as wireless RTT measurements at the wireless source entity, the out-of-interference delay *OID* of TCP data packets can be derived. The wireless RTT is composed of the sum of the delay experienced by the data packet on the way from the wireless source entity to wireless destination entity and the delay experienced by the ACK packet forwarded from the wireless destination entity to the wireless source entity. Each of these delays comprises the time to forward the packet over h wireless hops, where each forwarding requires a queuing delay t_q and transmission delays t_{data} and t_{ACK}, respectively. Using the measured wireless RTT, we get:

$$RTT_{wireless} = h\left(t_q + t_{data} + t_{LLD}\right) + h\left(t_q + t_{ACK} + t_{LLA}\right) \tag{8}$$

Here, t_{LLD} and t_{LLA} denote the average wireless link layer delay required for transmitting the TCP data packet and the TCP ACK packet, respectively. This delay comprises the transmission time of IEEE 802.11 control packets, link layer backoff, and potential retransmissions at link layer. Since information on link layer backoff and retransmissions are not available at the wireless source entity, we approximate t_{LLD} and t_{LLA} by defining the corresponding upper and lower bounds:

$$\frac{ACK_{LL}}{b_{base}} \le t_{LLD} \le \frac{ACK_{LL}}{b_{base}} + 3\left(\frac{s_{data}}{b} + cw_{cur} \cdot t_{slot}\right) \tag{9}$$

and

$$\frac{ACK_{LL}}{b_{base}} \le t_{LLA} \le \frac{ACK_{LL}}{b_{base}} + 3\left(\frac{s_{ACK}}{b} + cw_{cur} \cdot t_{slot}\right) \tag{10}$$

Here, ACK_{LL} denotes the link layer ACK size (14 bytes), b_{base} is the base bandwidth for transmitting of IEEE 802.11 control packets (1 Mbit/s), b is the bandwidth for data packets, cw_{cur} denotes the current size of the IEEE 802.11 contention window, s_{data} and s_{ACK} denote the packet sizes of TCP data and ACK packets, and t_{slot} corresponds to the IEEE 802.11 slot time. The lower bounds apply when the TCP packet (data or ACK) can be transmitted with no retransmissions. The upper bounds correspond to the case when it takes the maximum number of retransmissions to deliver the TCP packet. According to the IEEE specifications

[18], a total of 4 attempts (i.e. 3 retransmissions) are distinguished at link layer before the packet is dropped. We omit the DIFS and SIFS intervals [18] due to their negligible sizes. We consider the case with RTS/CTS deactivated. In case RTS/CTS is activated, the corresponding transmission times of the RTS and CTS packets at a bandwidth of b_{base} are considered in Eqs. 9 and 10. By choosing the median values within the upper and lower bounds for t_{LLD} and t_{LLA} we get an approximation error of at maximum 3%-6%.

Solving for t_q in Eq. 8 while using $t_{data} = s_{data}/b$ and $t_{ACK} = s_{ACK}/b$, we derive the average queuing delay as:

$$t_q = \frac{1}{2}\left(\frac{RTT_{wireless}}{h} - \frac{s_{data} + s_{ACK}}{b} - t_{LLD} - t_{LLA}\right) \tag{11}$$

Subsequently, we can estimate the out-of-interference delay of the TCP data packet:

$$OID = \left(H_{cs} + 1\right)\left(t_q + \frac{s_{data}}{b}\right) \tag{12}$$

The number of hops h on the network path to the wireless destination entity and the bandwidth of the wireless network interface b can be easily inquired without extra overhead from the kernel routing table and the IEEE 802.11 driver, respectively. Note that we use raw $RTT_{wireless}$ measures rather than EWMA-smoothed ones in order to be able to determine short-term RTT variations.

In theory, the maximum spatial reuse with minimum collisions can be achieved with a transmission rate $R_{max}=1/OID$. Thus, an upper bound for the capacity of a path with h hops in an IEEE 802.11 wireless mesh network is given by $h/(H_{cs} + 1)$ packets. Let $T_{one\text{-}way}$ denote the time a packet traverses from the wireless source entity to the wireless destination entity. This quantity can be computed as $T_{one-way} = OID \cdot h/\left(H_{cs} + 1\right)$. Subsequently, the number of packets in flight on the way from the wireless source entity to the wireless destination entity with a transmission rate of R_{max} is given by:

$$\tilde{P} = R_{max} \cdot T_{one-way} = \frac{1}{\left(H_{cs} + 1\right)} h \tag{13}$$

Thus, the number of packets in flight $\tilde{P}$ transmitted with the maximum transmission rate R_{max} reflects the maximum capacity of the network path.

4.4 The MAP Pacing Rate

The adaptive transmission rate of MAP accounts for both the current contention on the network path and the spatial reuse constraint. Thus, the transmission rate formula incorporates both cov_{RTT} and OID:

$$R = \frac{1}{\widehat{OID} \cdot (1 + 2cov_{RTT})} \tag{14}$$

The coefficient of variation quantifies the percentage of sample deviation from the mean. However, since we want to quantify the size of the spectrum in which the samples fluctuate around the mean, we double the value cov_{RTT} in the rate formula.

We average the measured out-of-interference delay samples and employ a reasonable history size N for the computation of the coefficient of variation using an exponentially weighted moving average (EWMA) with averaging weight α:

$$\widehat{OID} = \alpha \cdot \widehat{OID}_{old} + (1 - \alpha) \cdot OID \tag{15}$$

As validated by our experiments, suitable values for the EWMA weight α and the history size N are 0.7 and 50, respectively.

5 Comparative Performance Study

We evaluate the performance of MAP versus the widely deployed TCP NewReno by means of our Leipzig wireless mesh testbed. The considered performance measures are derived from 10 batches with 95% confidence intervals by the batch means method ([5], [6]). For all experiments, we set the TCP packet size to 1,460 bytes and the TCP receiver's advertised window to 64 packets. Consistent with previous work ([2], [9]), the RTS/CTS handshake is disabled, since it rather degrades TCP goodput due to the increased link layer overhead. We set the IEEE 802.11 data rate to 11 Mbit/s and the attenuation level of the variable attenuators to 16dB, unless otherwise stated. This provides a transmission range of 0.5m.

5.1 Wireless Chain Topology

The first topology we consider is an equally spaced wireless chain comprising 10 mesh nodes, where node 10 acts as mesh router to the wired subnet. Nodes in the chain are positioned such that only direct neighbors can communicate with each other over one hop. An FTP connection runs between the leftmost wireless node (i.e. node 1) and a wired host in the subnet.

In order to evaluate MAP versus TCP NewReno in a variety of different network conditions, we vary network-related parameters to reflect typical real world settings. For one, we consider the goodput of MAP and TCP NewReno in the wireless-to-wired case (i.e. from node 1 to the wired host), as well as in the wired-to-wireless case (i.e. from the wired host to node 1). Furthermore, we set the attenuation level of the variable attenuators such that the signal between nodes is either

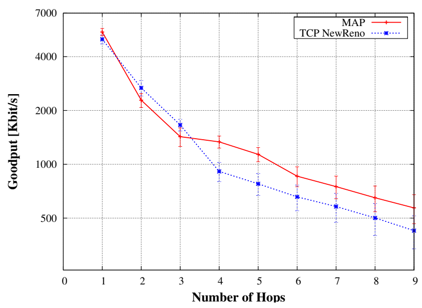


Fig. 3: Wireless chain: Goodput vs. number of hops for wireless-to-wired flow and high link quality

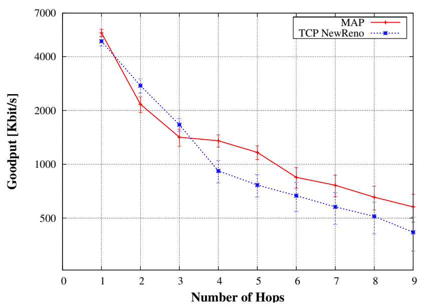


Fig. 5: Wireless chain: Goodput vs. number of hops for wired-to-wireless flow and high link quality

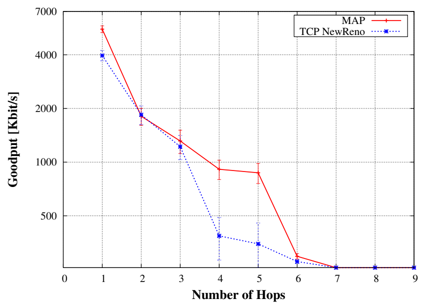


Fig. 4: Wireless chain: Goodput vs. number of hops for wireless-to-wired flow and low link quality

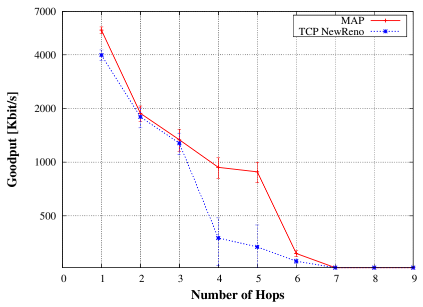


Fig. 6: Wireless chain: Goodput vs. number of hops for wired-to-wireless flow and low link quality

optimal (at low attenuation level) or very weak (at high attenuation level). Such high attenuation level and/or weak inter-node signal often occurs in real wireless mesh networks, in cases where either links between nodes suffer from high external interference or the distance between nodes is considerably large. Figures 3 to 6 show the results of this experiment, plotted as goodput versus chain length in terms of number of hops.

In Figure 3, we observe that the advantage of the MAP scheme evolves for a chain length of 4 hops and above. Specifically, MAP achieves around 46% more goodput than TCP NewReno. Up to a chain length of 3 hops, the goodput of MAP is similar to the goodput of TCP NewReno, with a slight advantage for TCP NewReno at 2 and 3 hops. The reason for such a turning point at 4 hops is the presence of hidden terminals for a chain length above 3 hops. As discussed before, TCP NewReno suffers from such hidden terminals due to its aggressive transmission strategy, whereas MAP takes advantage due to its adaptive pacing rate which reduces hidden terminal effects.

Figure 4 corresponds to the case where the signal strength between the wireless nodes is weak. In this case, we see that MAP significantly outperforms TCP NewReno, specifically up to 150% at 5 hops. The reason is that, in such a realistic environment where the signal between nodes is not optimal, the aggressive transmission of TCP NewReno greatly overwhelms the channel, resulting in severe packet loss rate. On the other hand, the adaptive pacing approach of MAP adjusts the transmission rate according to the current state of the channel, reducing packet loss and thus achieving more goodput. As the signal between nodes is at its lower limit, it does not suffice for delivering packets successfully for a chain of 7 to 9 hops.

Figures 5 and 6 show the goodput versus wireless chain length for a wired-to-wireless FTP flow, where the FTP connection runs between the wired host as TCP source to node 1 as TCP destination. The results are consistent with the findings in Figures 3 and 4. Such nearly identical match in the results is due to the fact that as a bottleneck, the wireless part of the network (i.e. the wireless chain) determines the performance of the considered transport protocol. Consequently, since the links between node 1 to MG are symmetric, both flow directions (i.e. from node 1 to MG and vice versa) deliver similar results.

5.2 Concurrent Flows Topology

As a second topology, we consider two concurrent FTP flows running between two wired hosts and a wireless mesh node, as illustrated in Figure 7. This scenario corresponds to the case in reality when a user in the wireless domain starts to simultaneously download content from two hosts in the Internet. In our case, two FTP connections run between the wired source hosts A and B through two mesh gateways (i.e. nodes 1 and 5) to the wireless destination node 3. Obviously, both FTP flows contend for the channel in the wireless part of the network. Our goal is to evaluate the performance of MAP and TCP NewReno in terms of fairness between the competing flows.

We consider the transient behavior of both flows by plotting the goodput of the flows over time. Figures 8 and 9 show the results for MAP and TCP NewReno, respectively. Considering MAP, we see how both flows share the available bandwidth equally over the lifetime of the connections. Since both flows experience similar interference, their cov_{RTT} values are also similar. Thus, according to Eq. 14, the MAP transmission rate is derived such that both flows acquire the same share of the bandwidth.

Opposed to MAP, TCP NewReno suffers from severe unfairness between the competing flows. As Figure 9 shows, during the lifetime of flow 1, it acquires the entire available bandwidth at cost of the completely starved flow 2. Not until about 10 seconds after flow 1 terminates is flow 2 able to take control of the bandwidth. The reason for such delay is the timeout interval of flow 2, which has to expire first before a new transmission attempt is performed.

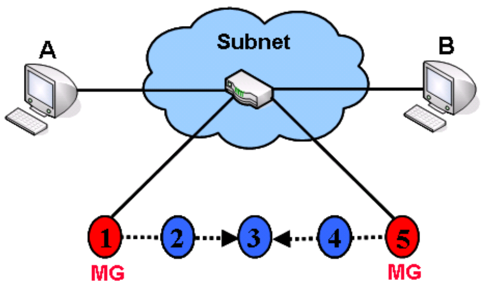


Fig. 7: Concurrent flows topology

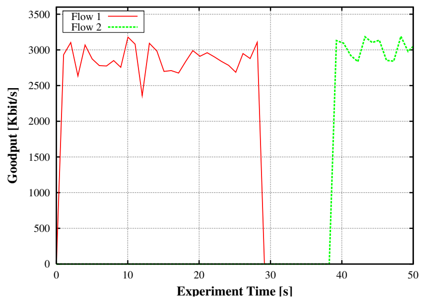


Fig. 9: Concurrent flows: Fairness of TCP NewReno

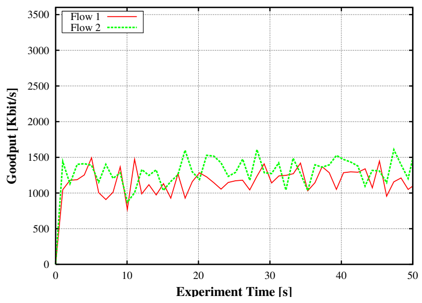


Fig. 8: Concurrent flows: Fairness of MAP

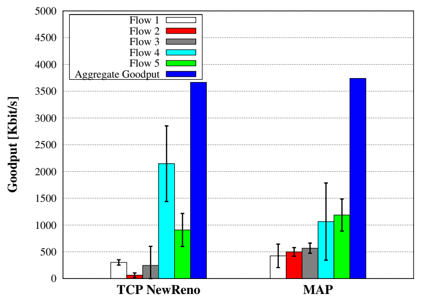


Fig. 10: Random topology: Individual and aggregate goodput for dual-radio communication

5.3 Random Topology

Random node topologies are typically found in community mesh networks such as [2] and are widely deployed in reality. We evaluate MAP and TCP NewReno in such topologies by considering random placements of the testbed's 20 antenna-stations. The 20 antenna-stations are distributed uniformly on a flat area of 2m x 3m such that full connectivity between each pair in the wireless network over one or more hops is granted. We consider 5 FTP flows and 5 different mesh gateways, which are also chosen randomly out of the testbed nodes. Each flow runs from a wired host in the subnet through one mesh gateway to a mesh node in the wireless network. Figure 10 shows the results using dual-radio communication, in which each of the non-overlapping channels 1 and 11 is assigned to a different wireless NIC.

In Figure 10, we see that TCP NewReno penalizes flows 1 to 3 in favor of the other 2 flows. Especially flow 4 acquires most of the available bandwidth. In contrast, MAP achieves more fairness among the competing flows, avoiding the starvation of any flow. The aggregate goodput of MAP and TCP NewReno is similar, since dual-radio communication reduces the effects of hidden terminals due to the reduced interference caused by the non-overlapping channels. Performance results acquired from a cross topology with four concurrent flows are consistent with the above findings. Unfortunately, we have to omit the corresponding curves due to space limitations.

6 Conclusion

We introduced MAP, an effective adaptive pacing scheme for improving goodput and fairness in wireless mesh networks with Internet connectivity. MAP operates at the wireless TCP source as well as at the mesh gateway, and transmits TCP packets by adapting the transmission rate according to the current network state. This results in reduced collisions at link layer, and thus improved goodput and fairness. MAP is fully TCP-compatible and relies solely on measurements of round trip times. Since it further requires no modifications at the routing layer or the link layer, MAP is easily deployable.

Opposed to most previous work, we implemented a real-world Linux prototype of MAP, which we evaluated in our Leipzig wireless mesh testbed. A comparative performance study showed that, depending on the current link quality, MAP achieves up to 150% more goodput than TCP NewReno and significantly improves fairness between competing flows. Experiments with dual radios indicated that the fairness problem of TCP NewReno persists for dual-radio communication, although the overall inter-link interference is reduced.

7 References

1. G. Anastasi, E. Ancillotti, M. Conti and A. Passarella, Experimental Analysis of a Transport Protocol for Ad hoc Networks (TPA), *Proc. ACM PE-WASUN Workshop*, Terromolinos, Spain, 2006.
2. J. Bicket, D. Aguayo, S. Biswas, and R. Morris, Architecture and Evaluation of an Unplanned 802.11b Mesh Network, *Proc. ACM MOBICOM*, Cologne, Germany, 2005.
3. T. Clausen and P. Jacquet, Optimized Link State Routing Protocol, RFC 3626, *http://www.ietf.org/rfc/rfc3626.txt*, October 2003.
4. D. De Couto, D. Aguayo, J. Bicket, and R. Morris, A High-Throughput Path Metric for Multi-Hop Wireless Routing, *Proc. ACM MOBICOM*, San Diego, CA, 2003.
5. S. ElRakabawy, A. Klemm, and C. Lindemann, TCP with Adaptive Pacing for Multihop Wireless Networks, *Proc. ACM MobiHoc*, Urbana-Champaign, IL, 2005.
6. S. ElRakabawy, A. Klemm, and C. Lindemann, TCP with Gateway Adaptive Pacing for Multihop Wireless Networks with Internet Connectivity, *Computer Networks*, 52, 2008.
7. K. Fall and K. Varadhan (Ed.), The ns-2 Manual, Technical Report, The VINT Project, UC Berkeley, LBL, and Xerox PARC, 2007.
8. Z. Fu, P. Zerfos, H. Luo, S. Lu, L. Zhang, and M. Gerla, The Impact of Multihop Wireless Channel on TCP Performance*, IEEE Transactions on Mobile Computing*, Vol. 4, Issue 2, March 2005.
9. V. Gambiroza, B. Sadeghi, and E. Knightly, End-to-End Performance and Fairness in Multihop Wireless Backhaul Networks, *Proc. ACM MOBICOM*, Philadelphia, PA, 2004.
10. T. Gleixner and D. Niehaus, Hrtimers and Beyond: Transforming the Linux Time Subsystems, *Proc. 8th OLS Linux Symposium*, Ottawa, Canada, 2006.
11. A. Haeberlen, E. Flannery, A. Ladd, A. Rudys, D. Wallach and L. Kavraki, Practical Robust Localization over Large-scale 802.11 Wireless Networks, *Proc. ACM MOBICOM*, Philadelphia, PA, 2004.
12. OLSR.ORG Implementation for Linux, *http://www.olsr.org*.
13. Freifunk Mesh Community, *http://start.freifunk.net/*.
14. D. Raychaudhuri, I. Seskar, M. Ott, S. Ganu, K. Ramachandran, H. Kremo, R. Siracusa, H. Liu and M. Singh, Overview of the ORBIT Radio Grid Testbed for Evaluation of Next-Generation Wireless Network Protocols, *Proc. IEEE WCNC*, New Orleans, LA, 2005.
15. S. Saunders, Antennas and Propagation for Wireless Communication Systems, *Wiley & Sons*, May 2007.
16. M. Savoric, H. Karl, M. Schläger, T. Poshwatta, and A. Wolisz, Analysis and performance evaluation of the EFCM common congestion controller for TCP connections, *Computer Networks 49 (2)*, October 2005.
17. B. Scheuermann, C. Lochert, and M. Mauve, Implicit Hop-by-Hop Congestion Control in Wireless Multihop Networks, Ad Hoc Networks 6 (2), April 2008.
18. IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, ISO/IEC 8802-11, August 1999.
19. K. Sundaresan, V. Anantharaman, H-Y. Hsieh, R. Sivakumar, ATP: A Reliable Transport Protocol for Ad Hoc Networks, *Transactions on Mobile Computing*, Vol. 4, Issue 6, November 2005.
20. D. Wai, P. Cao, and S. Low, TCP Pacing Revisited, *Proc. IEEE INFOCOM*, Anchorage, AK, USA, 2007.
21. L. Yang, W. Seah, and Q. Yin, Improving Fairness among TCP Flows crossing Wireless Ad Hoc and Wired Networks, *Proc. ACM MobiHoc*, Annapolis, MD, 2003.