TCP Capture Avoidance in Wireless Networks Based on Path Length and Path Density

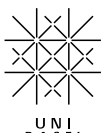
Evgeny Osipov and Christian Tschudin

Technical Report CS-2005-003
University of Basel
April 10, 2005

Abstract

We propose a distributed ingress throttling scheme for wireless networks based on path length and path density such that TCP fairness is guaranteed. To this end we introduce a model for the sustainable load permitted in the MAC interference zone. This model is then calibrated using simulations for a worst case scenario. We were able to validate the desired performance of our scheme in simulations as well as real world experiments. In this paper we also compare our solution to other schemes for TCP capture mitigation.

Keywords:
Ad Hoc networks, TCP capture, TCP performance,TCP fairness, IEEE 802.11.



Computer Science Department, Computer Networks Group, Bernoullistrasse 16, CH – 4056 Basel Switzerland. Phone: +41 61 267 05 55 Fax: +41 61 267 05 59 URL: http://cn.cs.unibas.ch/

1 Introduction

Poor TCP performance in mobile ad hoc networks (MANETs) is one of the stumbling blocks which prevents these networks from wide commercial deployment. In this paper we consider the well known and yet unsolved problem of TCP capture in MANETs. With TCP capture, only few connections obtain access to the network's transmission capacity, severely degrading the performance of other flows.

Several attempts solve this problem were undertaken during the last years. These solutions target either a modification of TCP mechanisms in order to make the protocol more "wireless friendly" [6], [7], [8] or a modification of the MAC protocol [2], [3], [4]. While some approaches solve the problem for particular cases, they are helpless in others. In [13] we have identified scenarios where functionality of the proposed algorithms would not eliminate the severe unfairness between TCP flows.

On the other hand, even for the cases where the proposed solutions are functional, we foresee that they will face considerable deployment problems because of the wide installation base of IEEE 802.11 products and the popularity of the existing TCP/IP protocol stack. In our approach we attempted to find a solution where modifications would touch only the sources of communications leaving TCP, MAC as well as the functionality of the forwarding nodes unchanged.

1.1 TCP in Wireless Networks

The conceptual difference of multi-hop communications in the wireline Internet and wireless MANETs is the nature of the transmission medium on the forwarding path. In the Internet the hops are carried on dedicated links. Thus, the transmission capacity of the links between hops are separated. This implies that a transmission originated on one link does not collide with the transmission ongoing on the link one or more hops away. In MANETs however we have a "super-shared" medium where multi-hop links belong to the same radio collision domain. This distinctive property of MANETs is well understood by the community. An illustrative example of TCP adaptation to this property is the research line dealing with optimization of TCP congestion window (CWND) [9], [12]. Obviously, the CWND cannot grow arbitrary large since even for a single isolated TCP flow its data segments collide with each other and acknowledgments on the forwarding path and congestion might occur already at the source of the flow leading to degradation of its performance.

1.2 Methodology

Our investigation aims at understanding the global behavior of multiple multi-hop TCP connections under worst case placement of nodes. Therefore, in the first stage of our research, we introduce a network model which facilitate the analysis of multiple TCP connections. After that we motivate and construct the worst case scenario. The major outcome of this stage is the definition of a *critical network load* beyond which a TCP flow begins loosing packets hence increases chances for the competing connections to capture the capacity.

In the second stage we derive an ingress rate which ensures that the sum of the loads produced by all TCP flows inside the network does not exceed the critical load.

In our solution the bound is a function of: (a) the number of hops for a particular TCP connection and (b) the number of competing nodes on the path of the connection (path density).

These parameters are obtainable from a reactive ad-hoc routing protocol as described in Section 4. For the experimental evaluation we configured this parameter in advance since we have a perfect knowledge about the flows in the constructed topologies.

1.3 Contribution and structure of the paper

The main contribution of this paper is the derivation of an upper bound on transmission rate at sources of TCP flows and the specification of a set of mechanisms to enforce this bound in MANETs which allow all competing TCP flows to obtain fair service. Besides the fact that our solution does not modify the existing standards of the IEEE 802.11 and TCP, it requires the configuration of schedulers at ingress points. Thus no changes to the forwarding nodes are needed.

The rest of the paper is organized as follows. The bulk of our solution is presented in Section 2 where we develop our ideas starting from introducing the TCP capture problem, giving the definitions and assumptions and then presenting and analyzing our network model. Following the evaluation of the proposed mechanisms in Section 3 we discuss the implications of the obtained rate bound and implementation issues further in Section 4. We present an overview of the related approaches in Section 5 before concluding with Section 6.

2 TCP Fairness and Sustainable Network Load

2.1 The TCP capture problem

TCP capture, which was described in [10], is characterized by expiration of the exponential TCP retransmission timer for one flow due to congestion created by other flows. The winning TCP flows utilize the temporal weakness of the affected flow by increasing their transmission rates. It has been shown by theoretical analysis presented e.g. in [11] and experimental studies in [4], [10], [13] that it is not possible to avoid TCP capture in wireless networks even with optimizing TCP settings. The work in [13] also indicates that the capture normally appears when some forwarding nodes of one TCP flow are located outside the reception zones but within the interference zones of competing forwarders.

One important metric to assess TCP capture is the unfairness index u which we define as the normalized distance (1) of the actual throughput of each flow from the corresponding optimal value.

$$u = \frac{\sqrt{\sum_{i=1..n} (X_{opt_i} - X_{act_i})^2}}{\sqrt{\sum X_{opt_i}^2}}. \tag{1}$$

In this formula X_{opt_i} is the ideal throughput of flow i obtained under fair share of the network capacity. In order to compute this value we divide the throughput of the corresponding flow obtained when running alone in the network by the number of competing flows. X_{act_i} is the actual throughput of the same flow achieved while competing with other flows. This index takes the values between 0 and 1 and reflects the degree of global user dissatisfaction, hence the value of 0 corresponds to perfectly fair communications and 1 represents the opposite case.

Figure 10 shows the TCP unfairness index for a family of network topologies, which clearly demonstrates the TCP capture problem of wireless multihop networks.

2.2 Definitions and Assumptions

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

2.2.1 Communication and Interference Ranges

Conventionally when talking about transmission ranges of the IEEE 802.11 enabled devices, a single transmitting node is considered. For the single node the radio transmission ranges are defined as schematically shown in Figure 1.

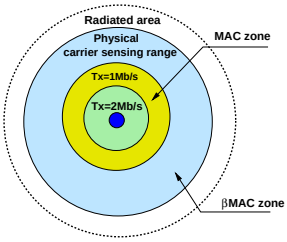


Figure 1: Communication ranges of the *base* IEEE 802.11 devices.

In order to facilitate readability of the paper we denote the area of assured data reception up to the slowest transmission rate (1 Mb/s) as the *MAC zone* of a node. We call the area beyond the MAC zone but within the physical carrier sensing range the *βMAC zone* (beyond MAC).

In this paper we extend the definitions of the transmission ranges for the entire multi-hop *association* as illustrated in Figure 2 and define MAC region, βMAC region as follows:

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

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

The above defined communication regions imply two different ways of nodes placement for several associations with respect to each other, hence two distinct cases of a network formation. The first case is where nodes of several associations partly or completely share the MAC regions of each other. The second case is where the associations are located solely in βMAC regions of each other.

In this paper we consider only the first case as it offers an advantage of direct communication between nodes of distinct associations. We address the issues regarding the second case in [14].

One requirement of our work is that the sources of all TCP connections sharing a common MAC region have full information about the *number* of competing flows. In Section 4 we show that this requirement is feasible in real MANETs using functionalities of reactive ad hoc routing protocols.

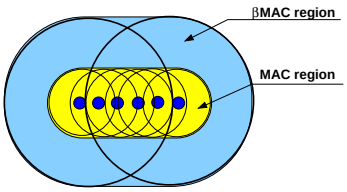


Figure 2: Communication ranges for the association (Only βMAC zones of the end nodes of the association are shown. Internode distance is 126 m).

2.2.2 Experimental and Simulation Setup

Here we describe the common settings for all simulations (with the network simulator NS-2 [16]) and real-world experiments. In this paper we focus on the original standard of the IEEE 802.11 with the highest data transmission rate equals 2 Mb/s.

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

In all setups we used TCP Newreno as the most popular variant of TCP. We performed real-world experiments with the maximum TCP data segment MSS equals 600 B. In simulations we operated with the broader range of MSS sizes from 200 B to 1000 B. All real-world and simulations values presented in this paper are means over six and 30 samples respectively.

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

2.2.3 Radio Transmission model and sizes of communication zones for a node

For the worst case analysis we rely on the radio transmission model of the original IEEE 802.11 standard used in NS-2. Therefore all values for sizes of respective MAC/βMAC zone of a node, MAC/βMAC region of an association are specific only for this model. Using the parameters of 914 MHz Lucent WaveLAN DSSS radio interface, the radius of MAC zone of a node is $r_{MACzone} = 250$ m and the radius of the βMAC zone is $r_{\beta MACzone} = 550$ m.

We define the minimum internode distance (d_{min}) for a multi-hop hop association the distance which satisfies the condition

$$d_{min}(N_i, N_{i+1}) = \frac{r_{MACzone}}{2} + \delta. \tag{2}$$

This condition dictates that any two nodes N_i and N_{i+2} communicating through another node N_{i+1} (a two hops communication) should be located just outside the MAC zones of each other. Therefore in order to satisfy this condition the intermediate node N_{i+1} should be located half way between N_i and N_{i+2}. Hence assuming for simplicity $\delta = 1$ m in a straight line topology $d_{min} = 125 + \delta = 126$ m.

2.3 Network Model and Worst Case Scenario

We represent the entire mobile ad-hoc network as a "black box" depicted in Figure 3.

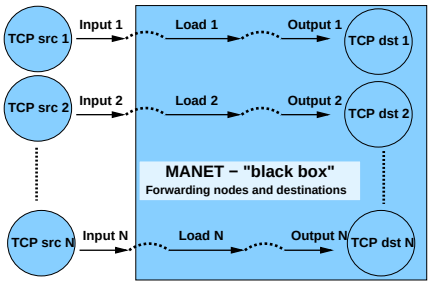


Figure 3: A wireless network as a "black box".

The two actors to differentiate in our network model are the sources of TCP flows and the set of all forwarding nodes sharing their MAC zones plus the destinations. Such separation of sources from other network devices is straight forward. It is the sources that introduce the load into the network and other devices just respond to their input by either forwarding the traffic or generating acknowledgments.

In MANETs sources can at the same time be forwarding or destination nodes for other flows. In general these nodes might not be located in geographical proximity of each other or even not share directly their MAC zones. The important point is that the load they generate into the network sums up in the black box formed by all or some of the forwarding nodes or destinations.

2.3.1 Operation of the model

The black box has two major quantitative characteristics: *capacity* (C_{net}) and *load* (L_{net}). The capacity term has its conventional meaning: It is the physical limit on the amount of traffic which can be transmitted inside the box per second.

In this network model we handle an entire association as a single entity. We say that association i brings load L_a^i consisting of transmissions from all devices (sources, forwarders and destinations) to the network and it consumes part of the black box's capacity C_{net}. The load of each distinct association i is determined by the *transmission rate* (or simply *rate*) of TCP data segments from the interface queue at the source node.

Obviously, the network load is the sum of the loads from all the competing associations $L_{net} = \sum L_a^i$. We define the term *critical load* ($L_{net}^{crit} = L_{TCP}^{crit} + L_{ACK}^{crit}$) as the threshold network load beyond which one or more TCP flows inside the MANET box begin to suffer from the unfairness of capacity distribution.

2.3.2 Competing Associations and Worst Case Placement

We decided to build our analysis considering the worst case of node placement for the competing associations: If we are able to find a solution which minimize the unfairness in the worst case it will bound the solutions for other cases from above.

As the worst case we consider a placement of nodes such that they interact with maximal number of other nodes. First of all we place the nodes of each association on the minimum

internode distance as defined by (2). After that in order to comply with our network model, we place all competing associations inside a stripe of width 250 m as shown in Figure 4. Under such placement of nodes every node share its MAC zone with maximal number of nodes. Placing some or all nodes of the competing association on distance more than 250 m from any node of the sample association would violate respectively the worst case condition of our model and our requirement about having at least one node in the MAC zone of any node of the sample association. The relative directions of the competing TCP connections does not play any role in our model, therefore we assume that all flows proceed from left to the right.

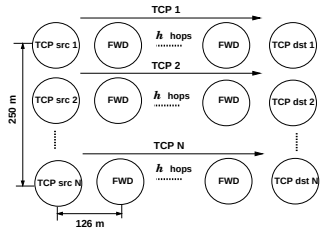


Figure 4: Worst case scenario.

We build our reasoning first by considering the case where all flows span over a path of equal length. The major reason for this is to observe the conditions for the TCP capture to appear in the case where the opportunities of each flow are equal in terms of TCP settings and the path lengths. In the case where competing flows have variable routes shorter flows have an additional advantage with respect to our sample flow as shown in Appendix A. In Section 3.3 we relax this assumption and consider TCP flows with routes of different lengths.

2.4 Critical network load

In this section we elaborate on the concept of the critical load in the MANET box. Estimating the critical load in our model is straight forward if we consider the native property of TCP. A single TCP flow in a steady state is a perfect estimator of the available bandwidth in the network. This property is utilized in the wireline Internet for estimation of the bandwidth delay product – the amount of traffic a pipe can accommodate. It is used to dimension the CWND parameter at sources to prevent local congestion [1].

It was shown in [9] that a single TCP flow achieves highest throughput in a multi-hop wireless network when the maximum CWND is scaled at sources as $h/4$, where h is the number of hops traversed by the flow. Further studies of the effect of TCP parameters on the end-to-end performance in [12] refine this bound and show that with the window size equals $3 \cdot h/2$ TCP achieves optimal throughput on a general topology. This is exactly the property we are looking for. Assuming that the source optimally configures CWND in MANETs a single TCP flow in isolation will generate as much traffic that will maximally use the capacity of the network on one hand and prevent losses of packets on the other. We also confirm this motivation by experiments in Section 3.

For the estimation of the critical load inside the MANET box we let one multi-hop TCP flow run alone in the network and measure the resulting load on the path. The load we measure is the joint transmission rate from all nodes of the association for this flow in bits per second. In order to

estimate this parameter in the case of one minimum distance (126 m) straight line connection we performed a series of experiments in NS-2 on such a connection for different sizes of TCP data segment. The resulting mean values are shown in Figure 5.

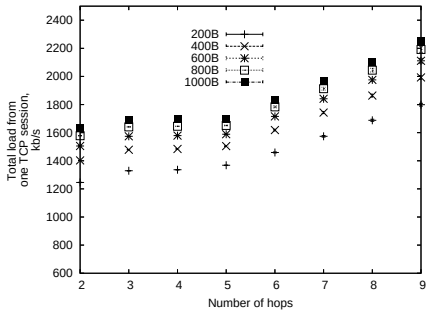


Figure 5: Critical load of the multi-hop network.

For computation of the transmission rate bound at sources (5) we need to estimate the part of the total critical load contributed by TCP data segments only. We measured this value during the simulations and the mean values are reflected in Figure 6.

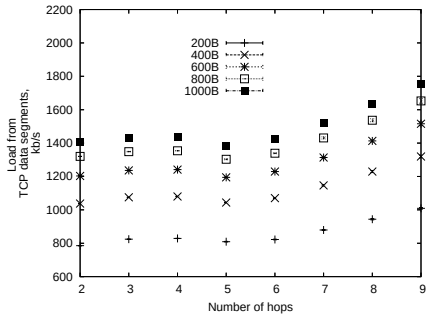


Figure 6: Part of the critical load contributed by transmission of TCP data segments.

2.5 Ingress Rate Throttling

In order to prevent any flow from loosing packets in the network, the total load generated by all competing sessions should be less or equal the critical load ($\sum L_a^i \leq L_{net}^{crit}$). Since generation of acknowledgments is determined by the arrival of the data segments this condition can be rewritten as in (3).

$$\sum_{i=1}^{N} L_{TCP}^{i} =\leq L_{TCP}^{crit}. \tag{3}$$

where *N* is the number of competing TCP flows. In order to control the load of TCP data segments generated by each flow we need to find it's dependency on the transmission rate of data

segments at sources. Considering the fact that for an isolated multi-hop TCP flow in a lossless case the intermediate nodes transmit as many TCP data segments as were issued by the source node we can express L^i_{TCP} as a function of the transmission rate at the source node (4).

$$L^i_{TCP} = r^i_{TCP} \cdot h. \tag{4}$$

The term h in the above formula is the number of hops on the forwarding path of the TCP connection. At a first glance such representation of TCP functionality is awkward from the point of view of TCP modeling. Indeed it is the rate at sources which depends on the arrival of acknowledgments which in turn depends on the load of TCP data segments on the path. However, from the functional point of view, the source node is the entity with embedded possibility to control its output rate hence it is natural to model TCP load as a function of the source's rate.

Assuming all flows follow the paths of equal length (see Section 2.3.2) denote $\rho_{path} = h \cdot (N - 1)$ the path density for the sample connection. This is the number of competing nodes sending TCP data segments. Now combining (3) and (4) we derive the bound for the sources transmission rate (5).

$$r_{TCP} \leq \frac{\alpha \cdot L^{crit}_{TCP}}{h + \rho_{path}}. \tag{5}$$

In the above formula $\alpha \geq 1$ is a load correction factor specific to the fixed delay scheduler of the interface queue as explained below.

The estimated value of L^{crit}_{TCP} allows us to compute the rate bound (5). Now we need to enforce it at sources of TCP flows. The following subsection defines the place where this functionality will be implemented.

2.5.1 Rate Bound Enforcement

Schematically the architecture of a wireless node is presented in Figure 7. For simplicity of presentation we assume that one source originates only one TCP flow. We also assume that the interface queue is either logically or physically divided into three queues: One for data packets originated at this node, the second one for all other data packets and the third for all packets of the routing protocol. Note that the last two queues are also the default structure of the IFQ in NS-2. All three queues are drop-tail in nature. Upon arrival from the routing layer all packets are classified according to their source IP addresses and type and placed into the corresponding queue.

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

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

2.5.2 Delay Parameter and Load Correction

The transmission rate bound (5) is used to set parameter Δ of the scheduler for the queue of locally generated packets described above. We compute the delay parameter as $\Delta = \frac{l_{packet}}{r_{TCP}}$, where l_{packet}

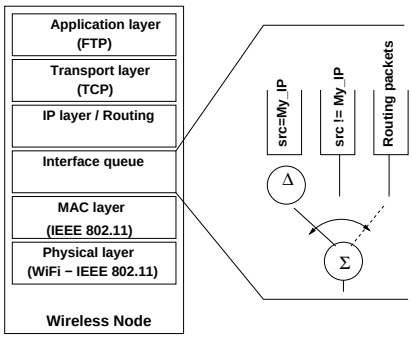


Figure 7: Structure of a IEEE 802.11 enabled (NS-2) node.

is the size of the outgoing packet in bits including the header overheads introduced by all layers of the TCP/IP protocol stack. Note that for estimation of L_{TCP}^{crit} we measured the actual transmission rate from all nodes at the MAC layer hence it also accounts for all transmission overheads.

In a next step, we need to compute Δ, configure the scheduler at sources of TCP connections and compare the resulting network load with the critical value. We performed a series of simulations on scenarios with various numbers of parallel three hop connections. We varied the number of TCP connections from 2 to 9 and each connection was configured to use 600 bytes TCP data segments. Note that when increasing the number of connections, the path density (ρ_{path}) for every connection grows linearly from 4 to 32 nodes. Initially for computation of Δ we set the load correction factor $\alpha = 1$. The results of this experiment are shown in Figure 8.

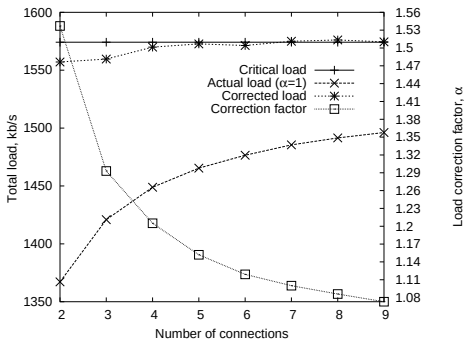


Figure 8: Load correction factor α.

In the figure the upper line represents the critical load for the experimental network from Figure 5. The lower curve is the actual total load after applying the rate throttling according to the computed bound. In the first place we observe a decrease in the total load in the case when the rate limitation is applied. This decrease is caused by two major reasons. Firstly due to the fact that we use a fixed delay scheduler the outgoing TCP traffic obtain a CBR nature. Therefore each individual throughput, hence the total network load, is decreased. Secondly this decrease happens because of the interaction of independent transmissions from forwarding nodes at the MAC layer. Indeed, for the estimation of L_{TCP}^{crit} we had only one TCP connection in the network. In this case

TCP data segments interact only with each other and own acknowledgments on the forwarding path. In presence of multiple competing associations the delay introduced by MAC layer due to contention of each competing node for the channel is applied to packets already delayed at sources by this reducing the individual throughput of each flow and the total network load.

We compensate for this phenomenon by introducing the load correction factor α. We iteratively estimated this parameter for different sizes of TCP data segments, route lengths and the number of connections. To estimate α we first compute the difference ratio between the actual total load and the critical load for $\alpha = 1$, this ratio becomes the new value for α. Then we increase the rate bound applying this value at every source, run the experiment again and compute new value of the correction factor. We repeat the last step until the actual load in the network differs from the estimated critical load on no more than 1%. In our sample network with three hops connections and MSS=600B the resulting values for α and the total load produced by TCP connections with this values are illustrated in Figure 8.

The shape of the curve for the correction factor has its natural explanation. In topologies with a low path density the number of nodes contending for the channel is less. Having lower degree of contention on the path the individual rates of the competing flows can be increased in larger steps. For a larger path density, even a small increase of the individual rates results in significant increase of the load.

From Figure 8 we observe an exponential decrease of the correction factor when the number of nodes in the network grows linearly. The analysis of this parameter in networks with longer routes and different sizes of TCP data segments reveals the same exponential law for its scaling, however the actual values for α are different for different values of MSS and route lengths. Because of the space limit those results are excluded from the paper.

2.6 Summary

In this section we presented a network model for the sustainable load permitted along the path of a sample TCP connection. We computed the bound on ingress rates which ensures that the total load from multiple TCP connections does not overflow the critical load beyond which TCP capture can occur. We also calibrated the model for different path densities.

3 Validation Through Simulation and Real-World Experiments

In this section we present validation of our solution by simulations and real-world experiments. We begin the performance evaluation by simulations. Firstly we analyze the dynamics of TCP performance in the sample set of networks used in the previous section. Then we test our findings on a broader class of networks. After that we assess the validity of the ingress throttling in case of variable length flows by performing experiments in NS-2 and a real-world testbed.

3.1 Three hops networks

In the previous section we monitored the resulting network load produced by three hops TCP flows when applying the rate throttling according to the derived bound. Now we evaluate the change in the performance with respect to the two parameters: The unfairness index (1) and the total TCP throughput in the network. The results are displayed in Figure 9.

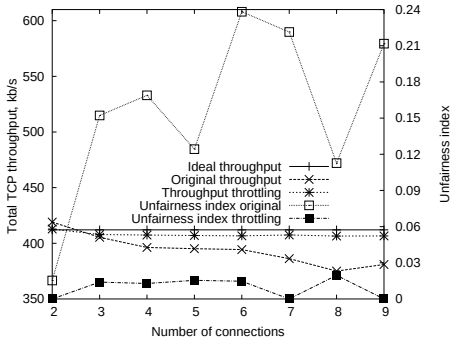


Figure 9: Total TCP throughput (left) and unfairness index (right) for the scenario shown in Figure 12.

As we observe from the figure the total TCP throughput, hence the load created by transmission of TCP data segments, in the case of multiple TCP connections is always lower than the throughput generated by single session (the straight line in the figure). By this we confirm our motivation from Section 2.4 to consider the load from single multi-hop connection as the maximally achievable in the network.

From the figure we also observe the positive effect of the source throttling with respect to both metrics. Applying our bound the resulting total TCP throughput is close to the ideal. The deviation of individual throughput for each flow from the optimal value (not shown in the graph) is ± 1.5 kb/s. This has its direct implication on the unfairness index. With the source throttling enabled we have homogeneous behavior for all number of competing connections in comparison to the case without rate limitation. On average the unfairness index is 0.0095 which corresponds to almost perfect fairness in every considered network.

3.2 Scaling the Network

In order to show the effect of our solution on a broader range of networks we performed a series of experiments for a family of worst case topologies. This time we scaled the network in two dimensions: the lengths of connections and the number of competing flows. Since the unfairness metric is valid for evaluation of two or more flows we varied the number of competing TCP flows from 2 to 9. The route lengths for each flow is scaled from 1 to 9 hops. The results are summarized in Figures 10 and 11.

We observe that the unfairness virtually vanishes when our rate bound is implemented.

3.3 Flows of variable lengths

Finally we address the effect of the rate throttling in networks with variable route lengths. As an illustrative example we choose the simple scenario from Appendix A depicted in Figure 12. This time we apply our rate bound to every flow generated at node N0.

As we discuss in Section 4 it is feasible to obtain the information about the number of competing connections extending the functionalities of the ad-hoc routing protocol. However we foresee that obtaining information about the overlap length of every competing flow is much more complex task. Therefore when computing bounds for every individual flow the value of L_{TCP}^{crit} should

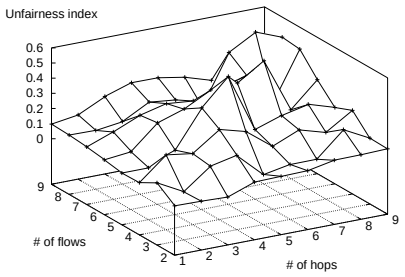


Figure 10: TCP unfairness index (plain TCP over IEEE 802.11) for the family of worst case scenarios shown in Figure 4.

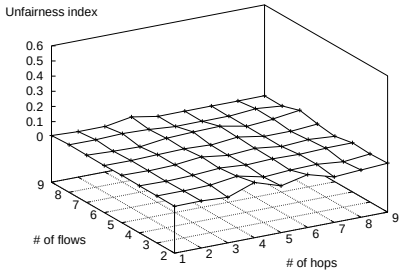


Figure 11: TCP unfairness index (TCP over IEEE 802.11 with our ingress throttling scheme) for the family of worst case scenarios shown in Figure 4.

be chosen based on the assumption that the competing TCP flows follow a path of the same length as the flow for which the bound is computed.

We take the corresponding values of L_{TCP}^{crit} from Figure 6 and compute (5) with actual route length of every flow, $N = 3$ and parameter α computed for corresponding cases. The results of this experiment are depicted in Table 1. Note that the corresponding values of the original performance are taken from Table 3 (b) in Appendix A.

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

Table 1: The effect of rate throttling in the case of flows with variable lengths.

This time we observe that some unfairness (9% in simulations and 4% in the testbed) remains in the network. This is because the critical load for longer flows is larger than for shorter ones (see Figure 6). Therefore the generated load after applying the rate throttling is larger. However

this behavior is acceptable taking into account the overall increase in performance in comparison to the original behavior.

4 Discussion

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

4.1 Gathering path length and path density information

The two parameters that are needed to compute the rate bound (5) apart from the L_{TCP}^{crit} are the length of the route for a connection and the number of competing associations. In the case that a reactive ad-hoc routing protocol is used to provide connectivity in multi-hop wireless network, the first parameter of interest is easily obtainable from the route reply (RREP) message received by the source.

In order to obtain the second parameter we need to enrich the routing protocol with a functionality of gathering information about the number of ongoing connections in the neighborhood of one hop of every node of the establishing connection. We have started to implement such information gathering and dissemination mechanisms in LUNAR [15], which is a L2.5 routing solution. LUNAR is well suited for this task as it establishes independent paths between peer nodes (while as DSR and AODV are aggregating routing information among routing paths), but will not discuss routing protocol modifications further in this paper.

4.2 Excess capacity and utilization

In our scheme a sample session will be guaranteed a share of the *available* network capacity for its entire duration considering its worst case communication, namely when it always has data to transmit e.g. long file transfer. If this is the case our scheme provides both perfect fairness and good network utilization. However, in reality we will also have a number of short-living communications and flows with a number of short transmission bursts e.g. web browsing. In this case the share of capacity will not be fully utilized. When several connections are originated from the same source the leftover capacity can be reused by the active flows, otherwise the network will be underutilized until the route for the inactive connection will be removed. Therefore it is essential to have an effective dynamic mechanism to update the ingress nodes with the path density information. Amongst the reactive ad hoc routing protocols we see again LUNAR as a perfect candidate which satisfies our criteria as it has embedded functionality of rebuilding the entire path for all flows every three seconds.

While presented for TCP communications the scheme is valid for UDP transport as well since in order to achieve “TCP-friendliness” the UDP flows should be throttled according to the same bound (5).

4.3 Wireless stub and transit networks

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

The first case corresponds to the normal functionality of the described solution with the correction that the sender's CWND should be scaled with respect to the path length from the source to the egress gateway.

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

4.4 Further developments

Outlining the major concepts and mechanisms of the ingress rate control in MANETs we also highlighted a set of open questions and problems. Firstly, further developments concerns formal analysis of the critical network and TCP loads and the load correction factor α which we determined by iterated simulations. This has direct implication on the tightness of the obtained bound. Secondly, the effect of the rate limitation in the case of availability of multiple transmission rates should be studied. Finally we need a detailed study of the proposed mechanisms in the case of node mobility, including the effects of routing algorithms and convergence time.

5 Related Work

A few papers attempt to formally model a single TCP flow over multi-hop wireless network. In [17] the authors formally characterize the throughput of a single multi-hop TCP connection in wireless network. The work describes an approach to model TCP transmissions over IEEE 802.11 network with enabled RTS/CTS exchange as an embedded Markov chain. Although simulations show good accuracy of the model, its extension for the case without RTS/CTS handshake and generalization to the total network load generated by the connection is yet to be developed.

One line of work which indirectly deals with the problem of TCP capture attempts to improve TCP congestion control mechanism in order to increase its performance in wireless networks [6], [7] and [8]. The common point of these studies is that they target TCP modifications in order to increase the throughput and make the protocol more "wireless friendly". However, as it is shown in [11], increasing TCP throughput as an ultimate goal of the network optimization problem leads to severe unfairness between the flows. The authors argue as we do that the focus should be placed on fairness issues.

A number of attempts to solve the problem of TCP capture were undertaken during several years. Altogether they can be characterized as an attempt to bring the known quality of service mechanisms from the wireline Internet to the wireless domain. The following papers are representative for this group. The work in [3] proposes a distributed algorithm for exchanging the status of local queues and scheduling information between all nodes in the neighborhood of one hop. For this purpose the authors suggest modifications to the distributed coordination function of MAC 802.11 to implement broadcasting of such information. Further on this information is used

to compute relative weights for all packets waiting for transmission in all stations in a manner that mimics the behavior of the weighted fair queuing scheduling in the wireline Internet.

Another technique for enhancing TCP fairness in ad hoc networks is distributed neighborhood RED [4]. The idea behind *nRED* is to coordinate the dropping of TCP segments in wireless nodes by overhearing the transmission in the neighborhood of two hops from each node and estimating the size of a virtual queue of the whole neighborhood.

Amongst the approaches targeting the solution of the TCP capture problem we distinguish the work in [5] as it has common parts with the approach described in this paper. The authors propose to limit the arrival rate from the interface queue to the transmission buffer on MAC layer depending on the observations of the departure rate in the past. The common point of this work to our approach is that the solution assumes rate limitation at a transmitting node implemented on the link layer, thus leaving the functionality of the MAC protocol unchanged. However, this approach differs from ours in several aspects. Firstly, their solution assumes rate throttling in forwarding nodes as well as in sources of communications while in our approach we limit the rate at sources of TCP flows only leaving the functionality of the forwarding nodes unchanged. Secondly, the rate limits in [5] are computed based on heuristics while for computation of our rate bound we use actual properties of MANET's transmission medium and TCP communications. Finally their approach was tested in simple scenarios while we consider the performance of our solution in the worst case.

6 Conclusions

In this paper we presented a methodology to derive adaptive bounds on TCP transmission rates at sources which allow guaranteed capture free communications. We focused on the case where all flows have information about (a) the number of connections ongoing in the one-hop neighborhood of forwarding nodes along the path to the destination, and (b) the length of their routing path.

The main result of this work is that by limiting the transmission rate at sources of TCP flows according to the derived bound, TCP capture does not occur and almost perfect fairness is achieved for all involved flows as we showed with simulations as well as with real world experiments. Moreover, our scheme does not require changes to TCP or IEEE 802.11 and is implementable by enhancing routing protocols and the use of traffic policing at ingress nodes.

References

[1] R.W. Stevens, TCP/IP illustrated volume 1: The protocols, Addison-Wesley Professional Computing Series.

[2] K.Tang and M. Gerla, "Fair sharing of MAC under TCP in wireless ad hoc networks" , In *Proc. IEEE MMT'99*, Venice, Italy, Oct 1999.

[3] N.H. Vaidya, P. Bahl and S. Gupta, "Distributed fair scheduling in a wireless LAN", In *Proc. MobiCom'00*,Boston, MA, USA 1999.

[4] K. Xu, M. Gerla, L. Qi, and Y. Shu, "Enhancing TCP fairness in ad hoc wireless networks using neighborhood RED" , In *Proc. MobiHoc'03*, Annapolis, MD, USA, 2003.

[5] L. Yang, W.K.G. Seah, and Q. Yin, "Improving fairness among TCP flows crossing wireless and wired networks", In *Proc. of the 4th ACM international symposium on Mobile ad hoc networking and computing*, Annapolis, MD, USA, 2003.

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

[7] R. Ludwig and R. H. Katz, "The Eifel algorithm: making TCP robust against spurious retransmissions", *ACM Comp. Commun. Rev.*, vol. 30, no. 1, Jan. 2000, pp. 30-36.

[8] S. Bhandarkar, N. Sadry, A.L.N. Reddy, N. Vaidya, "TCP-DCR: A novel protocol for tolerating wireless channel errors", *IEEE Transactions on Mobile Computing*, Feb. 2004.

[9] M. Gerla, K. Tang, and R. Bagrodia. TCP performance in wireless multi-hop networks. In *Proc. of IEEE WMCSA'99*, New Orleans, LA, USA, Feb. 1999.

[10] K. Xu, S. Bae, S. Lee and M. Gerla, "TCP behavior across multi-hop wireless networks and the wired Internet" , In *Proc. WoWMoM'02*, Atlanta, GA, USA, Sep. 2002.

[11] M. Johansson and L. Xiao, "Cross-layer optimization of wireless networks using nonlinear column generation", In *IEEE Transactions on Wireless Communications*, 2004.

[12] V. Kawadia, Protocols and Architectures for Wireless Adhoc Networks, Ph.D. Thesis, [Online]. Available: `http://black.csl.uiuc.edu/~prkumar/ps_files/04_07_kawadia_thesis.pdf`.

[13] Ch. Tschudin and E. Osipov, "Estimating the Ad Hoc Horizon for TCP over IEEE 802.11 Networks", In *Proc.MedHoc'04*, Bodrum, Turkey, Jun. 2004.

[14] E. Osipov, "Empirical Upper Bound on TCP Transmission Rate for Guaranteed Capture-Free Communications in Multi-hop IEEE 802.11 Based Wireless Networks", Technical Report CS-2005-001, University of Basel, 2005.

[15] Ch. Tschudin, R. Gold, O. Rensfelt and O. Wibling, "LUNAR: a Lightweight Underlay Network Ad-hoc Routing Protocol and Implementation", In *Proc. NEW2AN'04*, St. Petersburg, Russia, Feb. 2004.

[16] Network Simulator NS-2, [Online]. Available: `http://www.isi.edu/nsnam/ns/`.

[17] A.A. Kherani and R. Shorey, "Throughput analysis of TCP in multi-hop wireless networks with IEEE 802.11 MAC", In *Proc. IEEE WCNC'04*, Atlanta, USA, Mar. 2004.

Appendix A - Dynamics of TCP performance through consequent optimization of parameters

As we stated in Section 1 our main goal is to explore a solution to TCP capture problem in MANETs which does not require modifications to TCP protocol or standard MAC 802.11. Here we apply the existing findings from the research literature on increasing TCP performance in MANETs which satisfy our criteria. This section serves three major purposes: Firstly we introduce the problem area addressed by this paper, secondly we prepare an illustrative background for the overview of the related work and our motivation, and finally we address the node and TCP configuration choices which we will use throughout the paper.

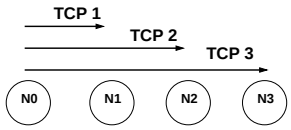


Figure 12: Simple test scenario for experiments and simulations.

We perform a series of experiments in real-world testbed and NS-2 with different network and protocol configurations and study the dynamics of TCP performance consequently introducing changes. We evaluate the performance of TCP with respect to the total TCP throughput in the network and unfairness index (1).

On a simple topology depicted in Figure 12 with only three TCP flows we show that optimizing the protocol and network configurations the TCP performance increases however the unfairness of communications remains.

6.1 Disabling RTS/CTS exchange

We first evaluate the effect of RTS/CTS handshake on our two TCP performance metrics. In [12] it is shown that on all topologies disabling the RTS/CTS exchange improves TCP throughput. The results of the first experiment are presented in Table 2. As we observe disabling RTS/CTS exchange improves the total throughput on almost 100 kb/s. Although the unfairness is slightly larger in the second case in total disabling RTS/CTS exchange is beneficial for TCP communications in MANETs. We account for this observation and disable RTS/CTS handshake in all further simulations and real-world experiments.

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

Table 2: The effect of RTS/CTS handshake on TCP performance.

6.2 Optimizing CWND

An important parameter which influences TCP performance is the value of the maximum congestion window (CWND). We pick the $3 \cdot h/2$ formula from [12] i.e., CWND(TCP1)=2MSS, CWND(TCP2)=3MSS, CWND(TCP3)=4MSS. The results of this measurement (simulation and experiment) are reflected in Table 3 (a).

(a) One common queue

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

(b) Separated queues

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

Table 3: The effect of CWND and interface queue optimization on TCP performance (no RTS/CTS exchange).

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

6.3 Splitting the IFQ at sources

A simple optimization to the previous problem is to split the incoming packets of different flows arriving to the interface queue at the source node into separate queues and serve them in a round robin manner.

We account for the last observation and perform our third experiment configuring the interface queue at the source node to have three logical queues one for each TCP flow. In the test bed we implemented this queue splitting by using the traffic controller (*tc*) embedded in Linux.

We performed the experiments with the default and optimal values of CWND now with optimized structure of the interface queue. The results are presented in Table 3 (b). As we expected in both cases the resulting total throughput in the network increased in comparison to the similar experiments in Table 3 (a).

In the experiments with optimal configuration of the congestion window in addition to some increase of the total TCP throughput both in simulations and real-world test we observed that the fairness index value remains on the same level as in the case of disabled RTS/CTS handshake and the default window size. The last observation confirms our assumption about the positive effect of combining the optimal TCP parameters and splitting the interface queue.