Cross-layer Optimization to Maximize Fairness among TCP Flows of different TCP Flavors

Toktam Mahmoodi, Vasilis Friderikos, Oliver Holland, Hamid Aghvami
Centre for Telecommunications Research, King's College London,
Strand, London WC2R 2LS, UK
{toktam.mahmoodi, vasilis.friderikos, oliver.holland, hamid.aghvami}@kcl.ac.uk

***Abstract*—A significant body of recent research has analyzed the problematic behavior of TCP over wireless links, and a plethora of modifications to TCP have been proposed in order to increase its performance in such contexts. Two schools of thought have emerged: the first proposes changes to the end-to-end protocol, while the second explores the potential to enhance lower layers as a means to improve the end-to-end performance of TCP. This paper focuses on the latter, and in contrast to most research in this area, which thus-far has concentrated on a single TCP flavor, examines the case where different TCP flavors are competing over a wireless link. To this end, we present and assess a cross-layer solution that involves the adaptation of lower layer characteristics (i.e., the coding rate) based on the detected TCP flavor, in order to maximize the fairness among TCP flows. Through extensive numerical investigations, we show that the proposed scheme considerably improves the fairness over wireless links among different TCP flavors. Our approach also has a minimal effect on the aggregate throughput of the TCP flows, and in cases where the packet error rate is very low, has a small positive effect on throughput.**

***Index Terms*—Transmission Control Protocol, FEC, Fairness, Wireless Packet Transmission.**

I. Introduction

Transmission Control Protocol (TCP) is the most widely used reliable end-to-end transport protocol over the Internet. It is expected that as the take-up of wireless technologies further increases, TCP will used much more over wireless networks. A significant body of work has therefore evolved over the past few years aimed at improving the performance of TCP over wireless networks [1]. Many such enhancement algorithms fit into one of two categories, the first of which proposes changes to the end-to-end protocol, and the second of which explores the potential to enhance lower layers in order to optimize end-to-end performance.

The mass of previous work on cross-layer design to improve wireless TCP performance has thus far assumed that all competing TCP flows are of the same flavor. However, with the spiraling number of TCP flavors in operation over the Internet [2], the end-to-end performance characteristics of TCP have become increasingly diverse. In this paper, we therefore study a more general framework where the different TCP flows are heterogeneous in nature, i.e., they can be based on different TCP flavors (such as Reno, NewReno, or TCP Westwood). In this context, because TCP flavors react differently to random packet losses, a significant degree of unfairness among TCP flows can surface in terms of achieved end-to-end rates. The performances seen by the different TCP flows are therefore not only dependent on packet loss rates across the wireless link, but also dependent on the TCP flavors of competing flows. Hence cross-layer designs to improve end-to-end TCP performance should also take into consideration the mix of TCP flavors at the wireless link.

To this end, based on the constraints imposed by the wireless link, we detail an optimization problem that strives to maximize Jain's fairness index in order to achieve a fairer allocation with respect to realized end-to-end throughputs of TCP flows. Moreover, through introducing an algorithm which dynamically enhances link-layer characteristics, we propose a means to maintain fairness among different TCP flavors competing over wireless networks. This suggested algorithm builds on our past work on top-down cross-layer TCP optimization (e.g., [3] and [4]), and is compatible with active solutions to improve end-to-end performances of reliable services, such as [5].

This paper is structured as follows. In the next section, we review the literature in terms of fairness studies among TCP flows over wireless networks. In Section III, we briefly introduce some common TCP flavors, and discuss their performances and procedures under packet losses. We also discuss TCP throughput modeling in Section III, thereby verifying some aspects of our utilized analytical models. In Section IV, we use Jain's fairness index as an objective function for our optimization framework, and propose a heuristic approach to solve the resulting problem. Various numerical results are presented in Section V, through which we argue the validity of our approach. This paper concludes in Section VI.

II. Related Work

As wireless networks become increasingly heterogeneous, it is important for service providers to ensure that the QoS obtained by different users and applications remains relatively equal. Since the majority of applications in the Internet use TCP, TCP's fairness has been well studied in the literature. The unfairness among TCP flows in WLAN networks is investigated in [6] and [7], where it has been shown that the base station's buffer size affects fairness. Reference [6] takes into account that the majority of applications involve download rather than upload, and proposes a rate control mechanism which modifies the TCP advertised window size in order to avoid loss in the downlink buffer. In [7], a smoother

978-1-4244-2324-8/08/$25.00 © 2008 IEEE.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2008 proceedings.

rate control mechanism is proposed to improve fairness in two highly congested scenarios that may cause starvation to TCP connections. The first scenario studied is the case where packets which belong to multiple TCP flows are competing in the WLAN base station transmission buffer, and the second scenario is where the base station is congested with TCP ACKnowledgements (ACKs) to be transmitted to the mobile users. The coefficient of variation of throughput is evaluated as the "unfairness index" in that paper, whereby the authors show that the performance of the above scheme is not affected by the version of TCP (Reno and NewReno are studied). However, neither in that publication, nor in the literature in general, has the effect of combining multiple TCP versions on fairness degradation been extensively studied.

Various different fairness measures have been proposed in the literature. *Jain's Index*, which was conceived to measure fairness in computer networks [8], is a very well used measure of fairness thanks to its advantageous mathematical properties. Jain's index is independent of the scale of the allocation metric, and is bounded between 0 and 1. In addition, Jain's index is continuous such that any change in allocation also changes the fairness. In this paper, we therefore utilize Jain's index to measure fairness among TCP flows in a wireless network.

Another issue important to this paper is the identification of the TCP flavor. Referring to the literature in this area, TCP flavors can be identified via the mechanism presented in [9]. Here, the TCP flavor and state are determined by monitoring changes in the estimated congestion window (cwnd), where the cwnd can be estimated passively at any point within the network (i.e., at routers) using this approach. Of the TCP flavors, TCP Reno, and TCP NewReno are covered by [9], but TCP Westwood is not investigated. We therefore use our own mechanism to identify TCP Westwood, whereby if a DupACK-triggered loss indication does not result in an approximate halving of the cwnd, the flavor is assumed to be TCP Westwood by deduction.

III. TCP Flavors and Packet Losses

For the vast majority of the time, TCP connections are likely to be in one of two phases: slow start or congestion avoidance. In the slow start phase, TCP increases its cwnd exponentially, leading to a doubling of the size of the cwnd per Round Trip Time (RTT). In the congestion avoidance phase, which is initiated upon the cwnd reaching the slow-start threshold (ssthresh), the cwnd is increased linearly (by one packet per RTT).

Packet losses in a TCP connection can be inferred by the detection of Duplicate ACKnowledgements (DupACKs), or by retransmission timer expirations. DupACKs (i.e., acknowledgements where the sequence number has not been incremented) are returned by the TCP receiver as an immediate response to receiving an out-of-order segment. From the sender's perspective however, DupACKs might be caused by a number of other issues (e.g., re-routing and traffic shaping) in addition to packet loss. Hence, to be conservative, loss detection is trigged only upon receiving three consecutive DupACKs.

A. TCP Congestion Control in the Presence of Losses

In the presence of losses, the behavior of TCP congestion control varies dependent on the TCP flavor. In this paper, we concentrate on the TCP Reno (TCPR), TCP NewReno (TCPNR), and TCP Westwood (TCPW) (which is built on TCPR) flavors. Characteristics of these versions of TCP can be summarized as follows.

TCPR congestion control [10], which can be considered the baseline for modern TCP implementations, supports fast retransmit and fast recovery upon segment losses. In TCPR, when a sender detects a segment loss through a retransmission timer expiration, the cwnd is set to one segment and the ssthresh is set to half of the FlightSize, where the FlightSize is the amount of outstanding data in flight within the network. If it detects packet loss through incoming DupACKs, the TCP sender invokes fast retransmit, which performs a retransmission of the lost segment immediately without having to wait for timer expiry. Fast recovery, which is used in conjunction with fast retransmit in TCPR and later flavors, sets the ssthresh to half of the FlightSize and the cwnd to the ssthresh plus three segments; this is deemed appropriate because although a loss has happened, packets are still getting through hence any reversion to slow-start would be far too severe. Upon receipt of the next ACK for new data, the cwnd is set to ssthresh, and congestion avoidance phase resumes. This received ACK therefore acknowledges all segments sent between the initial lost segment and its retransmission (including segments that triggered DupACKs, as well as those transmitted since and that were already in flight).

TCPR is known to generally not recover efficiently if there are multiple losses in a single flight of packets. TCPNR on the other hand presents a modification to the fast recovery algorithm of TCPR to improve recovery from multiple packet losses per window [11]. In the case of TCPNR, the ACK for new data is a partial ACK. The algorithm then retransmits the first unacknowledged segment, and deflates the congestion window by the amount of new data acknowledged by the cumulative acknowledgement field. Upon receipt of an ACK which acknowledges all segments, fast recovery phase exits.

The mechanisms described above can handle competition fairly well through cwnd and ssthresh changes in response to congestion-related losses. However, in shared medium access networks the available bandwidth for a TCP flow is highly variable dependent on channel utilization and medium access protocol dynamics. If a sudden change in available bandwidth occurs, TCP may be too slow to converge to this bandwidth. Moreover, TCPR/TCPNR are not usually robust when random (e.g., wireless) losses occur, as they misinterpret these losses as being caused by congestion. Alternatively, TCPW [12], which only requires modifications to the sender-side TCP, has been proposed to solve these problems. In response to a packet loss as detected by DupACKs, TCPW sets the cwnd and ssthresh to an *estimated eligible bandwidth* (BWE), which is calculated

978-1-4244-2324-8/08/$25.00 © 2008 IEEE.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2008 proceedings.

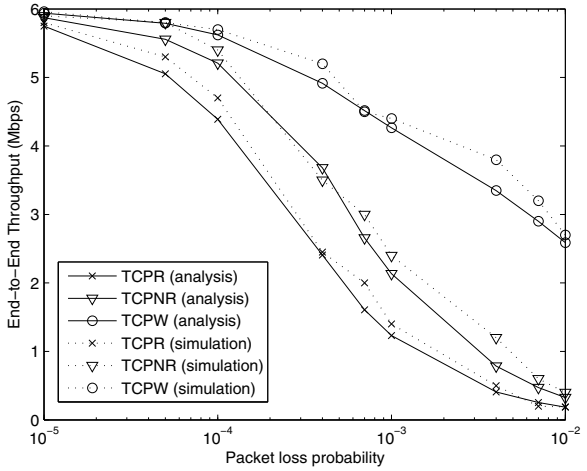


Figure 1. Single-flow analytical model and simulation results comparison for TCPR, TCPNR, and TCPW.

by low-pass filtering the rate of incoming ACKs (i.e., if ACKs are being returned at a certain rate, then packets are getting to the receiver at that same rate hence the network can support that rate). Moreover, if a loss is the result of DupACKs, the values for cwnd and ssthresh are set to
$ssthresh = BWE \cdot RTT_{min}/SegmentSize$,
$cwnd = \min(cwnd, ssthresh)$,
whereas in the case of a retransmission timer expiring under TCPW, these values are set to
$ssthresh = \max(BWE \cdot RTT_{min}/SegmentSize, 2)$,
$cwnd = 1$.

B. TCP Throughput Modeling

The three TCP flavors described previously are analytically modeled in the literature [13][14][15][16]. These models express TCP throughput as a function of the Packet Error Rate (PER) and the end-to-end RTT that each TCP flow experiences. Using these models, we study the effect of PER on the TCP throughput of each flavor. Furthermore, each TCP flavor is also simulated in a single-flow scenario in OPNET, and the results are compared. OPNET implementations of TCPR and TCPNR are based on the existing RFCs; the simulation code for TCPW in OPNET has been created by ourselves based on adapting the available ns-2 model.

The same conditions of 100ms RTT, 6Mbps bottleneck link, and a random packet loss rate varied in range [10^{-5}, 10^{-2}] are applied to the analytical model and the OPNET simulations; moreover, the simulations are all performed over a download file size of 16MB (approximately 11,000 packets) where packet size is 1460B. Figure 1 clearly shows that these three versions of TCP react significantly different when the packet loss increases. In addition, Figure 1 gives a comparison of the OPNET TCP models, with the analytical models.

For simplicity in using the analytical TCP throughput models in our framework, without loosing the generality, we assume that the packet loss is detected only via DupACKs, hence TCP does not face timer expiration.

Under the assumption of independency of packet losses between rounds, and that the sender's rate is not limited by the receiver's advertised window, a closed form for TCPR throughput in the steady state is proposed in [13] and revised in [14]. This gives the TCPR throughput, $B_R(p)$, as follows,

$$B_R(p) = \frac{\frac{1-p}{p} + E[W]}{\overline{RTT}\left(\frac{b}{2} \cdot E[W] + b + 1\right)}. \quad (1)$$

In the above expression, b represents the number of download packets that each acknowledgement applies to (in the most commonly used implementations of TCP, the default value of this is 1), p is the packet loss probability, and $\overline{RTT}$ is the average value of the RTT. Packet loss is only detected via DupACKs. Finally, $E[W]$, the expectation of the cwnd, is defined by,

$$E[W] = -\frac{3b-2}{3b} + \sqrt{\frac{8(1-p)}{3bp} + \left(\frac{3b-2}{3b}\right)^2}. \quad (2)$$

An analytical model for TCPNR throughput is proposed in [15], where the same assumptions as above apply. This model gives

$$B_{NR}(p) = \frac{1-p+E[W]}{RTT \cdot (1/2E[W] + b(1/2E[W]+1)+1)}, \quad (3)$$

where,

$$E[W] = \begin{cases} -\frac{3b+2\bar{\delta}}{3b} + \\ \sqrt{\frac{8(1-p)}{3pb)} + \frac{4\left(\bar{\delta}^2+\bar{\delta}-2\right)}{3b} + \left(\frac{3b+2\bar{\delta}}{3b}\right)^2}, & \bar{\delta} < E[W] \\ -\frac{3b-1}{3b+1} + \sqrt{\frac{8(1-p)}{p(3b+1)} + \left(\frac{3b-1}{3b+1}\right)^2}, & \text{otherwise.} \end{cases} \quad (4)$$

and $\bar{\delta}$ is the average number of segments lost per loss event.

A TCPW analytical throughput model is proposed in [16]. In this model, it is assumed that the system is always in the congestion avoidance phase, and that only a single packet loss occurs in each cycle. Denoting T as the RTT excluding queuing delay in buffers, μ as the packet transmission rate (assumed to be constant), and B_k^* as the burst at which the pipe capacity is reached, throughput model is formulated. k^* is determined from $k^* = C - W_0 + 1$, where C is the pipe capacity and W_0 is the initial congestion window. Assuming that the buffer can hold up to B packets, the packet number dropped due to buffer overflow, n_{of}, can be formulated as follows,

$$n_{of} = s_{k^*} + 2(C+B), \quad (5)$$

where s_k is the first packet of burst B_k,

$$s_k = 1 + (W_0 - 1)(k-1) + \frac{k(k-1)}{2}. \quad (6)$$

Denoting the number of the burst that contains packet number n as k_n, and the offset of the packet in burst B_{k_n} as r_n,

$$k_n = \left\lfloor -W_0 + \tfrac{3}{2} + \sqrt{W_0^2 - W_0 - \tfrac{7}{4} + 2 \cdot n} \right\rfloor, \quad r_{k_n} = n - s_{k_n}. \quad (7)$$

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2008 proceedings.

The instant at which the nth ACK is received is then given by

$$t\left(W_0,n\right)=\begin{cases} T_{k_n}+\frac{r_n}{\mu}, & n\leq s_{k^*},\\ T_{k^*}+\frac{n-s_{k^*}}{\mu}, & n>s_{k^*}. \end{cases} \tag{8}$$

The probability $p_{W_0,n}$ that packet n is dropped is expressed as

$$p_{W_0,n}=\begin{cases} p\left(1-p\right)^{n-1}, & n<n_{of}\left(W_0\right),\\ \left(1-p\right)^{n-1}, & n=n_{of}\left(W_0\right). \end{cases} \tag{9}$$

Finally, denoting π_W as the asymptotic probability of the initial window size being W_0, the average throughput is given by

$$B_W=\sum_{W_0=2}^{C}\pi_W\sum_{n=1}^{n_{of}}\frac{n-1}{t(W_0,n)}p_{W_0,n}. \tag{10}$$

IV. Fairness among TCP flows in Multiple-Flavor Scenarios

In the case where all flows are based on the same TCP flavor, congestion control algorithms guarantee fairness among TCP flows. However, in cases where multiple TCP flavors are coexisting in the network, fairness is affected by the different reactions of TCP flavors to packet losses. In this work, the well-known *Jain's Fairness Index* is used to measure fairness [8]. Jain's index is defined by the following equation:

$$J(\mathbf{P})=\frac{\left(\sum_{i=1}^{n}x(p_i)\right)^2}{n\cdot\sum_{i=1}^{n}x(p_i)^2}, \tag{11}$$

where n is the number of TCP flows, p_i the associated PER for flow i, $[\mathbf{P}]_i=p_i$, and $x(p_i)$ expresses the ratio between the TCP throughput and the optimal throughput that can be achieved by each TCP flow as shown in Equation (12).

$$x(p_i)=\frac{B_k\left(p_i\right)}{B_{k_{Optimal}}}. \tag{12}$$

A. Problem Definition

We consider n TCP flows, denoted by $i=1..n$, where each TCP flow i can be served by any one three flavors Reno, NewReno or Westwood, enumerated by $k=1..3$. These n flows compete for the limited capacity of the wireless link, where their throughputs are affected by the wireless error rate as described in Section III-B. Therefore, the throughput of each flavor is given $B_k\left(p_i\right)$ as a function of the corresponding flow's PER, p_i. Assume that the probability of a packet being in error can be adjusted according to $p_i\cdot 10^{\pm e}$ due to the specified link-layer error recovery algorithm. Therefore, the description of TCP throughput can be adjusted to $B_k\left(p_i\cdot 10^{\pm e}\right)$. Our aim is to maximize fairness among the n flows chosen among k TCP flavors. As mentioned above, $x(p_i)$ is the normalized TCP throughput with the optimal rate value which can be achieved by each flow. The optimal TCP rate for flow i is defined as the TCP throughput when all the other $n-1$ users have the same TCP flavor and the PER is the minimum feasible value, $p_i\cdot 10^{-e}$. In this work, we consider an optimization problem with the objective function being to maximize Jain's fairness index. In addition, we illustrate that maximizing the fairness index under this framework does not affect the overall TCP throughput significantly. The proposed non-linear optimization problem is outlined below:

$$\text{(P)}:\quad \text{maximize } J(\mathbf{P})=\frac{\left(\sum_{i=1}^{n}x(p_i)\right)^2}{n\cdot\sum_{i=1}^{n}x(p_i)^2},$$

subject to,

$$\sum_{i=1}^{n}B_k\left(p_i\right)\leq W,\quad \forall k\in\{1..3\}, \tag{13}$$

$$p_i\cdot 10^{-e}\leq p_i\leq p_i\cdot 10^{e},\quad \forall i\in\{1..n\}, \tag{14}$$

$$0\leq p_i\leq 1,\quad \forall i\in\{1..n\}, \tag{15}$$

where W is the channel capacity which depends on the wireless access method.

The TCP throughput is a non-linear but differentiable function over $p_i\cdot 10^{\pm e}$. The details of the congestion control algorithm behavior and the throughput modeling for TCPR, TCPNR and TCPW, are described in Section III, hence we utilize the throughput expressions of Equations (1), (3), and (10). It is assumed that TCP connections are sufficiently long-lived and are greedy in competing to access bandwidth.

B. Using the Logarithmic Barrier Method to Solve the Optimization Problem

The above problem needs to be solved in real time, at base stations. Here we therefore transform the problem such that Newton's method can be applied ([17], Chapter 9.5). It is noted that the constrained problem (P) can be approximated as an unconstrained optimization problem with the Logarithmic barrier function ([17], Chapter 11.2). However, through approximating with the Logarithmic barrier function, the inequality constraints of Equations (13)-(15) can be implicit in the objective function (see problem P). The optimization problem can therefore be rewritten as

$$\text{minimize } Y(\mathbf{P})=-J(\mathbf{P})-1/t\cdot\Phi(\mathbf{P}), \tag{16}$$

where

$$\begin{aligned}\Phi(\mathbf{P})=\quad & \log\left(-\sum_{i=1}^{n}B_k(p_i)+W\right)+\\ & \sum_{i=1}^{n}\log\left(-B_k(p_i)+B_k(p_i\cdot 10^{-e})\right)+\\ & \sum_{i=1}^{n}\log\left(B_k(p_i)-B_k(p_i\cdot 10^{e})\right)+\\ & \sum_{i=1}^{n}\log\left(p_i\right)+\sum_{i=1}^{n}\log\left(1-p_i\right).\end{aligned} \tag{17}$$

The parameter t sets the accuracy of the approximation; in the other words, the approximation becomes more precise as the parameter t grows. The modified objective function is convex, therefore Newton's method can be used to solve it.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2008 proceedings.

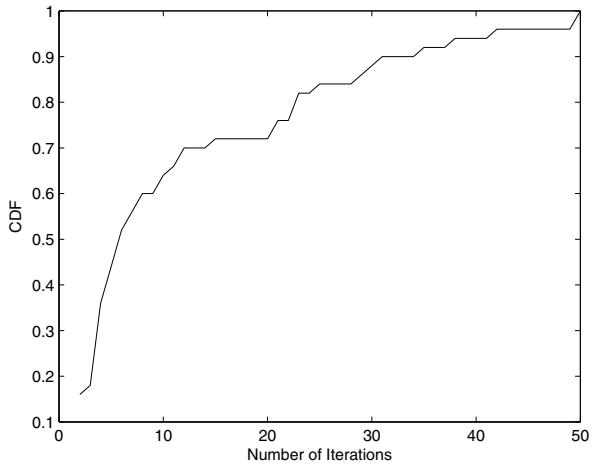


Figure 2. CDF of the number of iterations required in solving the problem using Newton's method.

We define the Hessian Matrix to formulate the Newton step:

$$H = -\nabla^2 J(\mathbf{P}) + (-1/t) \cdot \nabla^2 \Phi(\mathbf{P}), \tag{18}$$

giving Δp_{nt}, the Newton step as,

$$\Delta p_{nt} = H^{-1} * (\nabla - J(\mathbf{P}) + (-1/t) \cdot \nabla \Phi(\mathbf{P})), \tag{19}$$

where ∇ and ∇^2 are the first and the second gradient.

The optimal values of p_i are the results of the Newton iterations initiated from the current operation point of each flow. This iteration is described in further detail as follows:

1) Start from the current operational point as $p_{k=1}$,
2) Compute the first gradient, $\nabla Y(\mathbf{P})$, and the second gradient, H, of the objective function,
3) Compute Δp_{nt} $(= p_{k+1} - p_k)$ from Equation (19),
4) If $\|\nabla Y(\mathbf{P})\| \leq \varepsilon$ stop,
   else go back to 2.

We increase parameter t in sequential steps, whereby as mentioned earlier, increasing t increases the accuracy of the approximation. When t is large, the problem is difficult to solve by Newton's method, as its Hessian varies rapidly near the boundaries. In our problem, it can be seen that setting t to five leads to the optimal region.

It can be seen in Figure 2 that, given this approximation, the problem is solved in a small number of iterations. Figure 2 shows that, the heuristic approximation is solved in a maximum 50 iterations, while the average number of required iterations is approximately 12. Moreover, in 70% of cases, the optimal value is attained in less than 15 iterations.

V. Numerical Results

The optimization problem (P), has been solved using MATLAB's optimization Toolbox. TCP's PER is assumed to be a random variable in the range [10^{-5}, 10^{-2}], and e is 2, thus the exponent of PER is bounded by ± 2.

It is assumed that all the flows terminate at the same point in the wired network, which results in an equal RTT for all flows, set as 100ms in this work. Thus, unfairness arises solely from reactions of TCP flavors to packet losses.

Our simulation scenarios need to be defined over a system which supports adaptive coding, such as a WLAN. We therefore assume an IEEE802.11a network with a 54Mbps data rate. 15 mobile users connect to the end-host via a unique wireless access point, and each user receives a single TCP flow. The optimization framework is implemented in the access point through which all 15 flows pass. The PER can be varied by changing the code rate of the Forward Error Correction (FEC) scheme. It can be assumed that over a small period of time the wireless channel is fixed, therefore the operating values of PER are entered to the optimization framework, and the optimal values are computed for each flow. To examine the fairness level achieved, we study three different Scenarios over a small time frame.

In Scenario 1, there are five flows of each TCP flavor (TCPR, TCPNR, and TCPW). In Scenario 2, there are nine flows of TCPR, four flows of TCPNR, and two flows of TCPW. Finally, in Scenario 3, there are three flows of TCPR, eight flows of TCPNR, and four flows of TCPW. Under the three scenarios, Figures 3(a)-5(a) presents results showing the achieved fairness over the wireless link for our proposed cross-layer optimization of the link-layer, compared with the fairness among TCP flows that would be achieved otherwise. It is clear that our optimization approach has an extremely positive effect on the fairness achieved among the flows. In these three scenarios, we can see up to 50% increase in the fairness index. The increment in the fairness index is considerable in the high values of PER where, the end-to-end performance is more affected by the errors. As the probability of a packet in error gets smaller, the improvement in the fairness is also less significant.In average over the range of PER values, fairness is increased approximately 12.5% using our scheme.

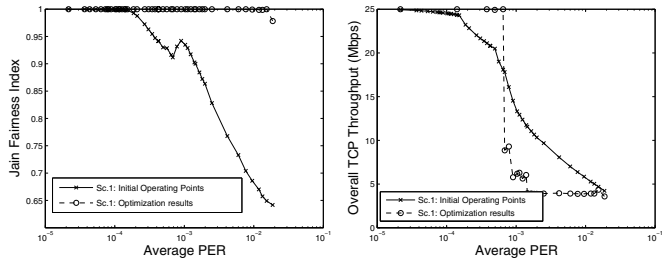
The effect of our fairness maximization scheme on TCP throughput also needs to be investigated. The aggregated TCP throughput, for the three studied scenarios, are plotted against the average PER in Figures 3(b)-5(b). It can be seen that, given a small PER, the aggregated TCP throughput remains the same and in some cases slightly increased under which case the improvement in the fairness is not considerable. As the PER increases however, the improvement in the fairness index becomes significant while the aggregate TCP throughput is decreased. From Figures 3(b)-5(b) we can see that at the PER value of 10^{-3}, overall throughput starts to decrease.

To summarize, the results for the three studied scenarios show that the fairness index is increased on average by 12.5%, or potentially by up to 50% through our scheme, while across all PERs the decrease in aggregate throughput through our scheme is approximately 4% on average.

VI. Conclusion and Future Work

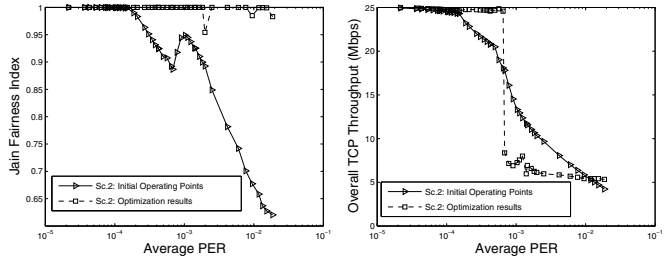
In this paper, we have presented a cross-layer mechanism to optimally set the FEC rate at the link-layer for packet transmissions, based on the end-to-end TCP flavor for each flow as detected at the wireless link. We have demonstrate a

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2008 proceedings.



(a) Fairness Index (b) Aggregated TCP throughput

Figure 3. Results for our cross-layer optimization scheme compared with what would otherwise be achieved, for scenario 1.



(a) Fairness Index (b) Aggregated TCP throughput

Figure 4. Results for our cross-layer optimization scheme compared with what would otherwise be achieved, for scenario 2.

framework to maximize fairness among competing TCP flows of different flavors across wireless links. Numerical results for our framework indicate up to 50% improvement in the Jain's fairness index as compared with a conventional fixed coding choice, while the overall TCP throughput is minimally affected (or is actually improved in cases of low packet error rate, due to the implied coding benefits of our scheme). Moreover, we present a heuristic optimization approach to find the optimum link-layer coding in the wireless base station, whereby convergence of the proposed scheme is studied.

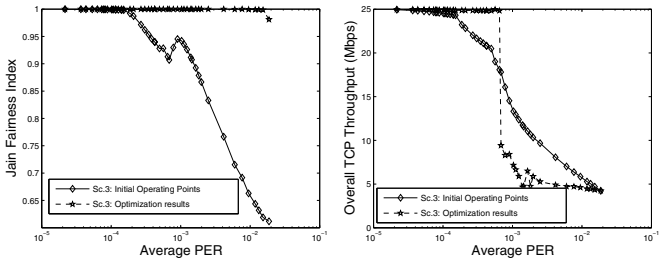
Although we study three well-used TCP flavors in this paper, it would also be useful to expand the frame work to include TCP options (e.g., SACK). Another area of interesting future work would be to investigate the performance of the proposed scheme in advanced network-level simulations.

VII. Acknowledgment

The work reported in this paper has formed part of the Delivery Efficiency Core Research Programme of the Virtual Centre of Excellence in Mobile & Personal Communications, Mobile VCE, www.mobilevce.com. This research has been funded by EPSRC and by the Industrial Companies who are Members of Mobile VCE. Fully detailed technical reports on this research are available to Industrial Members of Mobile VCE.

References

[1] V. Tsaoussidis and I. Matta, "Open Issues on TCP for Mobile Computing," *Wireless Comm. and Mobile Computing*, vol. 2, pp. 3–20, Feb. 2002.



(a) Fairness Index (b) Aggregated TCP throughput

Figure 5. Results for our cross-layer optimization scheme compared with what would otherwise be achieved, for scenario 3.

[2] A. Medina, M. Allman, and S. Floyd, "Measuring the Evolution of Transport Protocols in the Internet," *Comp. Comm. Review*, vol. 35, pp. 37–52, Apr. 2005.

[3] T. Mahmoodi, V. Friderikos, O. Holland, and A. H. Aghvami, "Cross-Layer Design to Improve Wireless TCP Performance with Link-Layer Adaptation," *Proc. IEEE VTC Fall*, pp. 1504–1508, Maryland, Oct. 2007.

[4] T. Mahmoodi, O. Holland, V. Friderikos, and A. H. Aghvami, "Cross-Layer Optimization of the Link-Layer based on the Detected TCP Flavor," *proc. IEEE PIMRC'08*, France, Sep. 2008.

[5] O. Holland, T. Mahmoodi, and A. H. Aghvami, "A Software Download Management Module," *Proc. IEEE VTC Fall*, pp. 1984–1989, Maryland, Oct. 2007.

[6] S. Pilosof, R. Ramjee, Y. Shavitt, and P. Sinha, "Understanding TCP fairness over Wireless LAN," *Proc. IEEE INFOCOM'03*, vol. 2, pp. 863–872, California, Mar. 2003.

[7] N. Blefari-Melazzi, A. Dett, I. Habib, A. Ordine, and S. Salsano, "TCP Fairness Issues in IEEE 802.11 Networks: Problem Analysis and Solutions Based on Rate Control," *IEEE Trans. Wireless Comm.*, vol. 6, pp. 1346–1355, Apr. 2007.

[8] R. Jain, D. Chiu, and W. Hawe, "A Quantitative Measure Of Fairness And Discrimination For Resource Allocation In Shared Computer Systems," DEC Research Report TR-301, Sep. 1984.

[9] S. Jaiswal, *Measurement in the Middle: Inferring End-End Path Properties and Characteristics of TCP Connections through Passive Measurement*. PhD thesis, University of Massachusetts Amherst, Sep. 2005.

[10] M. Allman, V. Paxson, and W. Stevens, "TCP Congestion Control," *IETF RFC 2581*, Apr. 1999.

[11] S. Floyd, T. Henderson, and A. Gurtov, "The NewReno Modification to TCP's Fast Recovery Algorithm," *IETF RFC 3782*, Apr. 2004.

[12] R. Wang, K. Yamada, M. Y. Sanadidi, and M. Gerla, "TCP with Sender-side Intelligence to handle Dynamic, Large, Leaky Pipes," *IEEE J Sel. Areas Comm.*, vol. 23, pp. 235–248, Mar. 2005.

[13] J. Padhye, V. Firoiu, D. F. Towsley, and J. F. Kurose, "Modeling TCP Reno Performance: A Simple Model and Its Empirical Validation," *IEEE/ACM Trans. Net.*, vol. 8, pp. 133–145, Apr. 2000.

[14] Z. Chen, T. Bu, M. Ammar, and D. F. Towsley, "Comments on Modeling TCP Reno Performance: A simple model and its Empirical Validation," *IEEE/ACM Trans. Net.*, vol. 14, pp. 451–453, Apr. 2006.

[15] R. Dunaytsev, Y. Koucheryavy, and J. Harju, "TCP NewReno Throughput in the Presence of Correlated Losses: The Slow-but-Steady Variant," *Proc. IEEE INFOCOM'06, Global Internet Workshop*, pp. 115–120, Spain, Apr. 2006.

[16] A. Zanella, G. Procissi, M. Gerla, and M. Y. Sanadidi, "TCP Westwood: Analytic Model and Performance Evaluation," *Proc. IEEE GLOBECOM'01*, vol. 3, pp. 1703–1707, Texas, Dec. 2001.

[17] S. Boyds and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.