Published in IET Communications
Received on 23rd January 2009
Revised on 2nd December 2009
doi: 10.1049/iet-com.2009.0394

ISSN 1751-8628

Optimal design of forward error correction for fairness maximisation among transmission control protocol flavours over wireless networks

T. Mahmoodi[1] *V. Friderikos*[2] *O. Holland*[2] *A. Hamid Aghvami*[2]

[1]*Intelligent Systems and Networks Group, Department of Electrical and Electronic Engineering, Imperial College, London SW7 2BT, UK*
[2]*Centre for Telecommunications Research, King's College London, Strand, London WC2R 2LS, UK*
E-mail: t.mahmoodi@imperial.ac.uk

Abstract: A plethora of modifications to transmission control protocol (TCP) have recently been proposed, a major aim being to improve its performance over wireless links. Two schools of thought have emerged: the first investigates changes to the transport-layer protocol, whereas the second explores the potential to enhance the characteristics of lower layers to improve the end-to-end performance of TCP. This study focuses on the latter, and, in contrast to most research in this area, which thus-far has concentrated on a single TCP flavour, examines the case where different TCP flavours are competing over a wireless link. To this end, the authors present and assess a cross-layer solution to adapt the coding rate at the link-layer based on the detected TCP flavour, to maximise fairness among TCP flows. Through both analysis and simulation, the authors show that the proposed scheme considerably improves the fairness among different TCP flavours that compete over a wireless link. Furthermore, the proposed approach has minimal detrimental effect on the aggregate throughput of TCP flows.

1 Introduction

Among the tapestry of solutions that have made the development and success of the Internet possible, the transmission control protocol (TCP), and numerous variations thereof, have become the norm for the provision of a reliable end-to-end unicast transport layer. As the take-up of wireless technologies further proliferates, it is expected, for compatibility reasons, that TCP will be used much more often over wireless networks. A significant amount of research has therefore evolved over the past few years, aimed at improving the performance of TCP over wireless networks [1]. Many such enhancements 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 optimise end-to-end performance.

The mass of prior art on designs to improve TCP over wireless links has thus far assumed that all competing flows are of the same flavour. However, with the spiralling number of TCP flavours in existence over the Internet [2], the end-to-end performance characteristics of TCP, particularly in cases of random (wireless) packet loss, have become increasingly diverse. In this paper, we study a more generalised framework, whereby the different competing TCP flows are heterogeneous in nature, that is, they can be based on different TCP flavours (such as for example Reno, New Reno, Westwood, or using the Selective ACKnowledgement (SACK) option). In this context, because TCP flavours react differently to random packet losses, a significant degree of unfairness among flows can surface in terms of achieved end-to-end throughput. The performances seen by the different TCP flows therefore depend not only on the packet loss rates over the wireless link, but also on the exact combination of currently utilised TCP flavours over that wireless link. Hence, designs to improve end-to-end TCP performance should also take into consideration the mix of TCP flavours operating over the wireless link.

To this end, based on the constraints imposed by the wireless link, we detail an optimisation problem that strives to maximise Jain's fairness index in respect of the throughputs achieved among the competing TCP flows. Moreover, we propose to improve fairness among the different TCP flavours competing over wireless networks, by introducing an algorithm that dynamically enhances the link-layer coding rate [3].

Our major contributions in this paper are as follows. We consider a scenario where competing TCP flows are of different flavours, thus have various reaction to packet loss. In this scenario, the following issues are discussed:

1. A novel analytical framework is proposed to maximise fairness by adapting the packet error rate (PER) that can be seen by each corresponding TCP flow.

2. In order to adopt the TCP PER, we select the link-level forward error correction (FEC) coding rate dynamically according to the results of our proposed optimisation problem.

3. Through extensive simulation studies, it has been shown that our novel cross-layer scheme considerably improves the fairness over wireless links among flows comprised of different TCP flavours. We further investigate the effect of the proposed scheme on the aggregate achieved throughput of the competing heterogeneous TCP flows. The network level simulator OPNET is used that provides real TCP implementations. Thus, our simulation results not only study the performance of our proposed scheme but also investigate its effect on the real TCP.

Despite the fact that adapting the FEC rate with respect to the TCP performance has been studied in the literature [4–6], issues regarding the existence of various competing TCP flavours have not been previously reported. It is also worthwhile to mention that, our main objective is to enhance the end-to-end performance (e.g. fairness among TCP flows) with no modifications to the actual transport layer, but with modifying the wireless link layer. Although we discussed this problem in [3], details of the simulation modelling and investigations on the performance of the proposed scheme in the existence of the real TCP is only examined in this paper – using OPNET modeller. Moreover, existence of the SACK option is studied in this paper in addition to the previously studied TCP flavours.

The remainder of this paper is organised as follows. In the next section, we review the literature in terms of reliability of link layer that provided by FEC scheme, and its adaptation to enhance the performance of TCP. The research works involving fairness among TCP flows over wireless networks are also reviewed in this section. Moreover, the cross-layer interaction methods between TCP and link layer are addressed. In Section 3.1, we discuss TCP throughput modelling, and verifying our utilised analytical expressions against real TCP implementations in our simulation platform. In Section 4, we use Jain's fairness index as an objective function for our optimisation framework, and propose a heuristic approach to solve the resulting problem. In Section 5, performance investigation of the scheme is carried out within the framework of a network simulation platform. Finally, this paper concludes in Section 6.

2 Background study

In this section the state of the art in the three related topics are addressed. Firstly, we discuss the related literature which investigates the effect of link-level FEC on the performance of TCP. Thus, some proposals for adapting the FEC coding rate in order to improve the performance of TCP are presented. Secondly, the related literature in fairness among TCP flows over wireless networks is studied. To this end, it is worthwhile to mention that neither the proposals to enhance TCP performance with FEC adaptations nor the research works that study fairness among TCP flows, have considered the co-existence of various TCP flavours. Finally, the methods to transfer cross-layer information from TCP at the end host to the link layer at the wireless base station are investigated.

2.1 FEC adaptation to improve TCP performance

In wireless networks, in order to counter the effects of random losses, some degree of additional reliability is often provided at the link layer. This reliability is usually achieved through link level automatic repeat requests (ARQ) in conjunction with FEC.

ARQ over the wireless link is more rapidly reactive compared to TCP's acknowledgement control loop. This is because TCP operates over a much longer delay path than the wireless link, and because packets (and retransmissions, or coded FEC packets) at the link layer are usually much smaller than the TCP packets, because of link layer fragmentation. The effect of ARQ persistency on TCP performance is studied in the literature [7], and TCP-aware ARQ algorithms are detailed [8].

FEC coding allows the receiver to correct a proportion of errors caused by the wireless channel by adding the redundant bits. We should note that when a channel has a low signal-to-noise ratio, adding the redundancy, can even decrease the performance of the wireless link [9, Chapter 8.2]. In high signal-to-noise ratios, increasing the redundancy improve the reliability of FEC; this will also increase the channel load and the processing/reception delay. This trade-off is studied in [4] where the achieved TCP performance with respect to the FEC rate has been shown and the gains that can be attained in the end-to-end TCP throughput has been studied. In [5], a utility function is defined to jointly tune power, FEC and ARQ for improving the performance of TCP. These works, perform their study assuming a single version of TCP (TCP Reno (TCPR) in most cases).

Moreover, in [6] the optimal design of a hybrid FEC/ARQ scheme with respect to the TCP throughput is presented. Although both TCPR and TCP NewReno (TCPNR) versions are studied in [6], the co-existence of these two flavours is not investigated. It is worthwhile noting that this research work investigates a single TCP flow scenario. Therefore it can be seen that the mass of previous work on the FEC rate selection with respect to the TCP performance have neither considered the co-existence of various flavours nor the effect of such a combination on the FEC rate selection.

2.2 Fairness among TCP flows in wireless networks

Since the majority of applications on the Internet use TCP, TCP's fairness has been well studied in the literature. The unfairness among TCP flows in wireless local area network (WLAN) is investigated in [10, 11], where it has been shown that the base station's buffer size affects fairness. Pilosof *et al.* [10] take into account that the majority of applications involve download rather than upload, and proposes a rate control mechanism that modifies the TCP advertised window size in order to avoid loss in the downlink buffer. In [11], a smoother 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 that 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 to be transmitted to the mobile users. In that paper, authors evaluate the coefficient of variation of throughput as the 'unfairness index', thereby demonstrating that the scheme's performance is not affected by the version of TCP (the Reno and NewReno versions 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 studied in detail.

Various different fairness measures have been proposed in the literature. The Jain's fairness index [12], which was conceived to measure fairness in computer networks, is a very well used measure of fairness in both wired and wireless networks [13], thanks to its advantageous mathematical properties. The 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 utilise Jain's index to measure fairness among TCP flows in a wireless network.

2.3 Cross-layer interactions

In this work we present a top-down cross-layer interaction between TCP and link layer. Therefore our proposed scheme requires the TCP flavor and the end-to-end round trip time (RTT) of each flow to be known at the wireless access point (AP).

Referring to the literature in this area, TCP flavours can be identified via the mechanism presented in [14]. Here, the TCP flavour 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. This mechanism can be implemented in any point in the middle of the end-to-end path, such as wireless AP, thus identify the TCP flavour of the end-host TCP.

The timing information of TCP timing can be included in the Timestamp option of the TCP header [15]. The Timestamp option regarding RFC 1323 [16], is placed in the TCP header. TCP headers containing this option will increase from 20 bytes in size to 32 bytes. The receiver then echoes the Timestamp value in the acknowledgement, thus allowing the sender to calculate an RTT for each received ACK. Despite the fact that RTT can be identified by reading Timestamp from the TCP header, the TCP header may be encrypted, thus cannot be read at the link layer of the wireless AP. In this respect, various methods are presented in the literature to estimate RTT either actively or passively at an interior measure point [17]. Passive estimation of RTT can be performed with high precision as experiments show that 90% of the passive measurements are within 10% of the RTT that ping would measure [17]. Either reading the TCP Timestamp option or estimating the RTT using the above methods can be used to identify RTT at the wireless AP.

3 TCP flavours in the presence of losses

For a TCP connection, packet losses can be inferred by the detection of Duplication 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 the receipt of an out-of-order packet. 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.

In the presence of losses, the behaviour of TCP congestion control varies dependent on the TCP flavour. In this paper, we concentrate on three TCP flavours: TCPR [18], TCPNR [19] and TCP Westwood (TCPW) [20] (we use the model built based on TCPR), and examine the reaction of each TCP to the packet loss in the next section. Furthermore, we study the presence of the SACK option in TCPR.

3.1 Performance comparison of various TCP flavours

The three TCP flavours in our study are analytically modelled in the literature [21–23] – the presented model in [21] is revised in [24]. Using the SACK option is also analytically

doi: 10.1049/iet-com.2009.0394

modelled in [25]. These models express the TCP throughput as a function of the PER and the end-to-end RTT that each TCP flow experiences. Although the above models are presented over the wired networks, and wireless networks faces higher error rate thus more rapid changes in the cwnd, they are still valid and have been widely used to investigate the behaviour of TCP over wireless networks [26].

Using the above-mentioned models, we study the effect of PER on the TCP throughput of each flavour. Furthermore, each TCP flavor is also simulated in a single-flow scenario in OPNET, and the results of the simulation and analysis are compared.

To achieve this, we have developed the simulation code for TCPW in OPNET based on the model that has been already available in ns-2 [27]. The same conditions of 100 ms RTT, 6 Mbps bottleneck link and a random packet loss rate varied from 10^{-5} to 10^{-2} are applied to the analytical model and the OPNET simulations; moreover, the simulations are all performed over a download file size of 16 MB (approximately 11 000 packets) where the packet size is 1460 B. Fig. 1 clearly shows that these four versions of TCP react significantly different when the packet loss increases. In addition, Fig. 1 compares the OPNET TCP models with the analytical models.

In the rest of this paper, we assume that packet loss is detected only via DupACKs, such that TCP does not face timer expiration. Timer expiration occurs mainly from the loss of an ACKnowledgement (ACK) packet, which can be attributed to the use of cumulative acknowledgement. Thus, the probability of ACK loss occurring is low – much lower than for data packets – even in wireless channels with high error probability. Therefore generality is not affected by this assumption.

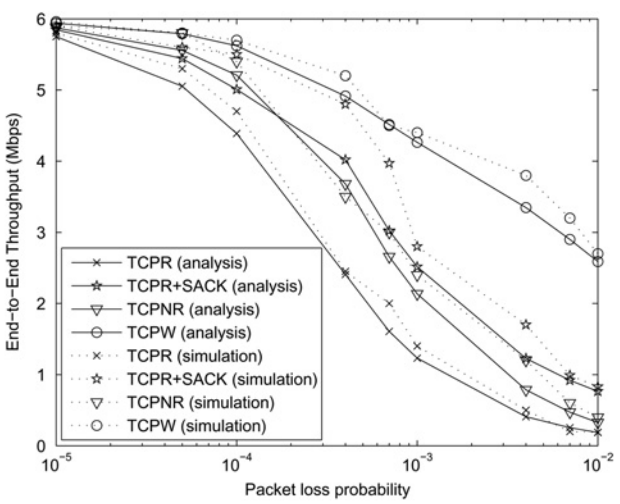

Figure 1 *Single-flow analytical model and simulation results comparisons for TCPR, TCPR + SACK, TCPNR, and TCPW*

3.2 Modelling TCPR throughput

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 [21] and revised in [24]. This gives the TCPR throughput, $B_R(p)$, as follows

$$B_R(e) = \text{MSS}\frac{(1-e)/e + E[W]}{\overline{\text{RTT}}((b/2)E[W] + b + 1)} \tag{1}$$

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 is 1 – , e is the packet loss probability and $\overline{\text{RTT}}$ is the average value of the RTT. Finally, $E[W]$, the expectation of the cwnd, is defined by

$$E[W] = -\frac{3b-2}{3b} + \sqrt{\frac{8(1-e)}{3be} + \left(\frac{3b-2}{3b}\right)^2} \tag{2}$$

3.3 Modelling TCPR throughput with enabled SACK option

In the presence of SACK option in TCPR, we use the model which is presented and validated in [25] where its Reno model presents similar results to [21]. Given the assumption that a loss happened in the jth packet of the round when the flow has a cwnd of h. Thus

$$B_{\text{SACK}}(e) = \frac{\text{MSS}}{e}\cdot\left(\frac{2}{c_{wm}(c_{wm}+1)} \times \sum_{h=1}^{c_{wm}}\sum_{j=1}^{h}(2 + r(n) + r(a, n))\overline{\text{RTT}}\right)^{-1} \tag{3}$$

where it is assumed that the cwnd varies uniformly between 1 and c_{wm}. MSS is the TCP maximum segment size, and $r(n)$ is the number of rounds required to transmit n packets. With the initial cwnd value of n, it takes $r(a, n)$ rounds to transmit a packets in the congestion avoidance phase.

3.4 Modelling TCPNR throughput

An analytical model for TCPNR throughput is proposed in [22], where the extra assumption of correlated losses to the above assumptions applies. This model gives

$$B_{\text{NR}}(e) = \text{MSS}\frac{1 - e + E[W]}{\overline{\text{RTT}}\cdot(1/2E[W] + b(1/2E[W]+1)+1)} \tag{4}$$

doi: 10.1049/iet-com.2009.0394

where

$$
E[W] = \begin{cases} -\dfrac{3b+2\bar{\delta}}{3b} + \sqrt{\dfrac{8(1-e)}{3eb} + \dfrac{4(\bar{\delta}^2+\bar{\delta}-2)}{3b} + \left(\dfrac{3b+2\bar{\delta}}{3b}\right)^2}, & \bar{\delta} < E[W] \\ -\dfrac{3b-1}{3b+1} + \sqrt{\dfrac{8(1-e)}{e(3b+1)} + \left(\dfrac{3b-1}{3b+1}\right)^2}, & \text{otherwise} \end{cases} \tag{5}
$$

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

3.5 Modelling TCPW throughput

A TCPW analytical throughput model is proposed in [23]. 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. The value of k^* is determined from $k^* = C - W_0 + 1$, where C is the pipe capacity and W_0 is the initial cwnd. Assuming that the buffer can hold up to B packets, the packet number dropped because of buffer overflow, n_{of}, can be formulated as $n_{\mathrm{of}} = s_{k^*} + 2(C+B)$. The determination of s_k that is the first packet of burst b_k is given by

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

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

$$
\begin{aligned} k_n &= \left\lfloor -W_0 + \frac{3}{2} + \sqrt{W_0^2 - W_0 - \frac{7}{4} + 2n} \right\rfloor \\ r_{k_n} &= n - s_{k_n} \end{aligned} \tag{7}
$$

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

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

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

$$
p_{W_0,n} = \begin{cases} e(1-e)^{n-1}, & n < n_{\mathrm{of}}(W_0) \\ (1-e)^{n-1}, & n = n_{\mathrm{of}}(W_0) \end{cases} \tag{9}
$$

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

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

4 Fairness among TCP flows in multiple-flavour scenarios

When all flows are based on the same TCP flavour, congestion control algorithms guarantee fairness among TCP flows. However, in cases where multiple TCP flavours are coexisting in the network, fairness is affected by different reactions of TCP flavours to packet losses. In this work, the well-used fairness index in wired and wireless networks [13], Jain's index [12], is used as a notation of fairness as defined by the following equation

$$
J(\boldsymbol{E}) = \frac{\left(\sum_{i=1}^{n} x(e_i)\right)^2}{n \sum_{i=1}^{n} x(e_i)^2} \tag{11}
$$

where n is the number of TCP flows, $[\boldsymbol{E}]_i = e_i$ is the associated PER for flow i and $x(e_i)$ expresses the ratio between the TCP throughput and the optimal throughput that can be achieved by each TCP flow: $x(e_i) = B_k(e_i)/B_{k_{\mathrm{Optimal}}}(e_i)$. The optimal TCP rate for flow i in this paper, is defined as the throughput of the corresponding flow when all the other $n-1$ flows have similar TCP flavour to flow i.

4.1 Problem definition

We consider n TCP flows, denoted by $i = 1 \ldots n$, where each TCP flow i can be served by any one of the four TCP versions, that is, Reno, Reno + SACK, NewReno or Westwood, enumerated by $k = 1 \ldots 4$. These n flows compete for the limited capacity of the wireless link, W, and their throughputs are affected by the wireless error rate as described in Section 3.1. Therefore the throughput of each flavour, $B_k(e_i)$, is given as a function of the corresponding flow's PER, e_i. Given $e(\mathrm{init})_i$ the initial value of PER for each TCP flow i, we further assume that the probability of a packet being in error can be adjusted according to $e_i = e(\mathrm{init})_i \times 10^{\pm\xi}$ because of the specified link-layer error recovery algorithm [FEC code rate (R_i), in this case]. Therefore the description of TCP throughput can be adjusted to $B_k(e(\mathrm{init})_i \times 10^{\pm\xi})$ with either increasing or decreasing the FEC coding rate. Our aim is to maximise fairness among the n flows that use different TCP flavours. As mentioned previously, $x(e_i)$ is the normalised TCP throughput with the optimal rate value which can be achieved by each flow. The optimal TCP rate is calculated for the minimum feasible value of PER for the corresponding flow, $e(\mathrm{init})_i \times 10^{-\xi}$. Under the above assumptions, we formulate an optimisation problem with the objective function being to maximise the Jain's fairness index. In addition, we illustrate that maximising the fairness index under this framework does not affect the overall TCP throughput significantly. The proposed non-linear

IET Commun., 2010, Vol. 4, Iss. 10, pp. 1196–1206
doi: 10.1049/iet-com.2009.0394

optimisation problem is outlined below

$$(\text{P})\text{:maximise } J(\boldsymbol{E}) = \frac{\left(\sum_{i=1}^{n} x(e_i)\right)^2}{n\sum_{i=1}^{n} x(e_i)^2}$$

$$\text{subject to } \sum_{i=1}^{n} B_k(e_i) \le W, \quad \forall k \in \{1..4\} \tag{12}$$

$$e(\text{init})_i \times 10^{-\xi} \le e_i \le e(\text{init})_i \times 10^{\xi}, \quad \forall i \in \{1..n\} \tag{13}$$

$$0 \le e_i \le 1, \quad \forall i \in \{1..n\} \tag{14}$$

Although PER of e_i is a discrete variable, assuming that TCP packet size can be changes accordingly e_i can be converted to a continuous variable. The TCP throughput is a non-linear but differentiable function over $[e_i \times 10^{-\xi}, e_i \times 10^{\xi}]$. The details of the congestion control algorithm behaviour and the throughput modelling for TCPR, TCPNR and TCPW are described in Section 3.1. Hence, we utilise the throughput expressions of equations (1), (3), (4) and (10), and we assume that TCP connections are sufficiently long lived.

4.2 Link-layer FEC code rate selection

Let us define TCP packets' size MSS bytes, which are fragmented into link-layer frames of m bytes and later coded into blocks of M bytes to transmit over the wireless link. We assume a convolutional coding with the code rate equal to R where $R = m/M$. For the convolutional coding, Hamming distance is defined as the distance between two code words, in which the minimum distance of the code is called minimum free distance and denoted by d_{free} [9, Chapter 8.2]. The block error rate (l) over the block size of M is defined as a function of d_{free} which is also dependent on the R and the bit error rate (BER) of the wireless link (b): $l = f(R; b)$. Given the fixed value of b – at the same condition over wireless channel – and having different values for R, the value of l and consequently e are varied. In other words, increasing the code rate R results in increasing the PER e and vice versa.

Given different code rates of R, the values of function f is calculated for the wireless channel BER in the range $[10^{-6}, 0.1]$. Thus, Fig. 2 shows the wireless channel BER, b, against the PER of TCP, e, for various values of R. The specifications of our utilised encoder/decoder and further discussion on Fig. 2 is given in Section 5.2. Knowing the current BER of the channel, the optimal code rate for the link-level FEC can be found by moving horizontally on the curve of Fig. 2 from the initial PER of the corresponding TCP flow to its desired PER that is the result of solving problem (P). Assuming the initial PER for flow i is e_i, the desired PER is in the range $[e_i \times 10^{-\xi}, e_i \times 10^{\xi}]$, thus the feasible PERs shown on the y-axis should be within this range.

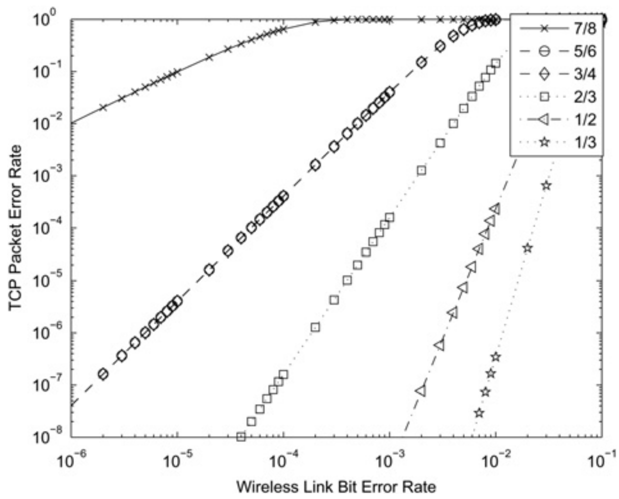

Figure 2 *TCP packet loss against link BER in various convolutional code rates with $K = 7$, $g_0 = 171$ and $g_1 = 133$*

4.3 Observations based on numerical investigations

To observe the performance of such a scheme, the optimisation problem (P) has been solved using MATLAB's optimisation Toolbox. As an illustrating example we assume 15 TCP flows with PER as a random variable in the range $[10^{-5}, 10^{-2}]$, and ξ is 2, thus the exponent of PER is bounded by ± 2. It is assumed that the bottleneck is wireless link, thus the PER is only affected by the loss at the wireless link. Moreover, all flows terminate at the same point in the wired network, which results in an equal RTT for all flows (100 ms in this section). Thus unfairness arises solely from the reactions of TCP flavours to packet losses.

In order to strive for generality, three combinations of TCP flavours are studied. The first scenario consists of four TCPR, four TCPR + SACK, four TCPNR and three TCPW. In the second scenario, there are three TCPR, five TCPR + SACK, five TCPNR and two TCPW. Finally, the combination of TCP flavours in the third scenario is three TCPR, three TCPR + SACK, three TCPNR and six TCPW. All the three scenarios are simulated over a network configuration as shown in Fig. 3. Results in terms of average and maximum improvement in the Jain's index

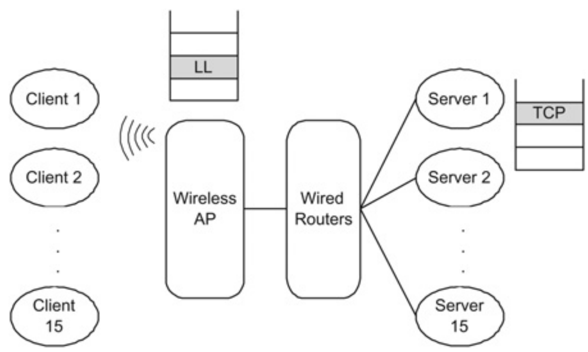

Figure 3 *Investigated network architecture*

Table 1 Fairness improvement as achieved by our cross-layer scheme

	Average increase, %	Maximum increase, %
scenario one	15	41
scenario two	19	38
scenario three	12	33

are given in Table 1. The increase of up to 41% in the Jain's fairness index can be seen. Moreover, over a range of investigated values of PER, the average increase in the Jain's index is 20%.

4.4 Using the logarithmic barrier method to solve the optimisation problem

The above optimisation problem – P – needs to be solved in real time, at the base stations. In this section, we transform the problem such that Newton's method can be applied [28, Chapter 9.5]. It is noted that the constrained problem (P) can be approximated as an unconstrained optimisation problem with the logarithmic barrier function [28, Chapter 11.2]. Therefore through approximating with the logarithmic barrier function, the inequality constraints of equations (12)–(14) can be implicit in the objective function. Assuming inequality constraints are a set of functions $f_i(x)$ such that $f_i(x) \leq 0 \; \forall i \in \{1, \ldots, m\}$, they can be added to the objective function as $\sum_{i=1}^{m} I\_(f_i(x))$. In this term, indicator $I\_(u)$ can be defined as

$$I\_(u) = \begin{cases} 0, & u \leq 0 \\ \infty, & u > 0 \end{cases}$$

Although indicator $I\_$ can convert the optimisation problem (P) to the unconstrained problem, it is not differentiable and Newton's method cannot be applied. Therefore the key idea of the barrier method is to approximate the indicator $I\_$ by a differentiable function such as $\widehat{I}\_(u) = -1/t \log(-u)$, so that Newton's method can be applied. Parameter $t > 0$ sets the accuracy of the approximation; in the other words, the approximation becomes more precise as the parameter t increases. The optimisation problem can therefore be rewritten as follows

$$\text{minimise } Y(\boldsymbol{E}) = -J(\boldsymbol{E}) - 1/t\Phi(\boldsymbol{E}) \tag{15}$$

where

$$\Phi(\boldsymbol{E}) = \log\left(-\sum_{i=1}^{n} B_k(e_i) + W\right) + \sum_{i=1}^{n} \log\left(-B_k(e_i) + B_k(e_i \times 10^{-\xi})\right) + \sum_{i=1}^{n} \log\left(B_k(e_i) - B_k(e_i \times 10^{\xi})\right) \tag{16}$$

The modified objective function is convex, therefore Newton's method can be used to solve it. We define the Hessian matrix, $\boldsymbol{H}$, to formulate the Newton step, as follows

$$\boldsymbol{H} = -\nabla^2 J(\boldsymbol{E}) + (-1/t)\nabla^2\Phi(\boldsymbol{E}) \tag{17}$$

giving Δe_{nt}, the Newton step as

$$\Delta p_{nt} = \boldsymbol{H}^{-1}(\nabla(-J(\boldsymbol{E})) + (-1/t) \cdot \nabla\Phi(\boldsymbol{E})) \tag{18}$$

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

The optimal values of e_i are the results of the Newton iterations initiated from the current operation point of each flow. Given ε a relatively small value, the mentioned iterations are described in further detail in Algorithm 1.

Algorithm 1. Newton iterations to solve problem P
1. Start from the current operational point as $e_{k=1}$.
2. Compute the first gradient, $\nabla Y(\boldsymbol{E})$, and the second gradient, $\boldsymbol{H}$, of the objective function.
3. Compute Δe_{nt} $(=e_{k+1} - e_k)$ from equation (18).
4. If $\parallel \nabla Y(\boldsymbol{E}) \parallel \leq \varepsilon$ stop,
else go back to 2.

We increase parameter t in sequential steps, but we should note that 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, setting t to five can lead to the optimal region.

It can be seen in Fig. 4 that given this approximation, the problem is solved in a small number of iterations. Fig. 4 shows that the heuristic approximation is solved in a maximum of 50 iterations, whereas the average number of

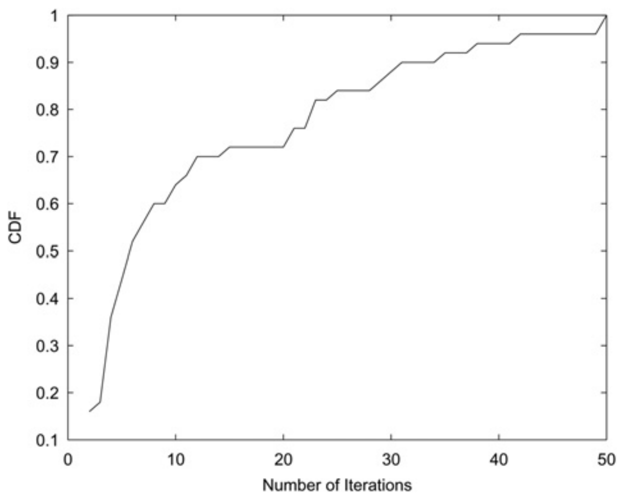

Figure 4 *CDF of the number of iterations required in solving the problem using Newton's method*

doi: 10.1049/iet-com.2009.0394

required iterations is approximately 12. Moreover, in 70% of cases, the optimal value is attained in less than 15 iterations.

5 Simulation study

To further investigate the performance of the proposed scheme and its effect on real TCP implementations, OPNET simulation platform is used. The implementation of TCP in OPNET is based on its RFCs. TCPR and TCPNR as well as the SACK option were all supported already within OPNET; however, TCPW is implemented for this research work within the OPNET environment based on the available model from Network Simulator 2 [27]. We have verified through comparison with published simulation results and by comparison with an analytical model (Fig. 1), as well as by liaising with the designers of TCPW, that our implementation is correct.

5.1 Simulation parameters

In our simulated scenarios, wireless clients set up connections with wired servers via a single wireless AP and wired routers (see Fig. 3). Each wireless client connects to a unique server, where the bottleneck is assumed to be at the wireless link. The wireless channel is modelled with an ITU indoor path loss model ($\mathrm{PL} = 30\log(d) + 20\log(f_c) + 48$ where d is the users' distance from the AP in meter and f_c is the operating frequency in GHz) and lognormal shadowing (standard deviation 4 dB). Therefore low mobility users are assumed, and channel parameters are assumed to be constant over each 2 s interval. This is because, every 2 s, information on the channel error rate is updated at the AP, whereby the optimisation problem is then triggered to renew the corresponding code rate of each flow. In the different simulated scenarios, the distances of wireless clients from the AP within the range of 20–70 m (see Fig. 5), and RTTs are varied between 20 and 300 ms. Mobile users are distributed uniformly over the cell with radius R. To uniformly distribute the mobile users, the angle θ is chosen uniformly and the radius is $r = R\sqrt{z}$, where z is generated uniformly between 0 and 1 [29].

Other specific simulation characteristics are as follows:

- Simulation duration: 120 s.

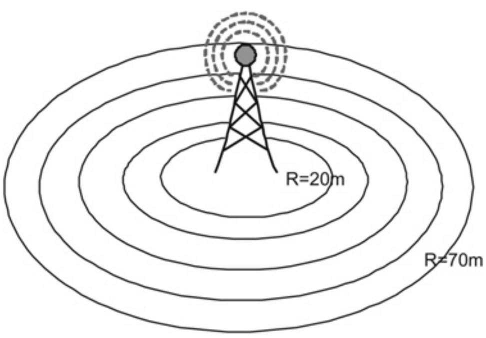

Figure 5 *Defined cells around the wireless AP*

- FTP servers: 16 MB file download size.

- HTTP servers: HTTP1.1, webpage interarrival time (s): lognormal ($\mu = 0.22$, $\sigma = 2.64$), HTML object size (KB): lognormal ($\mu = 7.90$, $\sigma = 1.76$), number of images per webpage: gamma ($k = 0.14$, $\theta = 40.32$), image size (KB): (lognormal $\mu = 7.51$, $\sigma = 2.17$) where μ and σ are from the mean and standard deviation of the normal distribution, and k and θ are the shape and scale parameters of gamma distribution [30].

- TCP maximum segment size (MSS): 1460 B.

- MAC packet size: 365 B.

- MAC layer specification: IEEE 802.11.

- Physical layer characteristic: OFDM (802.11a).

- Operating frequency: $f_c = 5.4$ GHz

- Data rate: 6 Mbps.

Note that we have concentrated on 802.11a, as the properties associated with orthogonal frequency division multiplexing (OFDM) has led to its consideration as a candidate for new generation of wireless networks, although the proposed scheme can be applied to different wireless networks.

5.2 Convolutional encoder/decodder

In this section, we discuss the encoder/decoder that is used to attain the simulation results in more detail. We assume the NASA standard convolutional encoder/decoder which is well implemented for example in Actel enc/dec core [31], and can support selectable code rates of 1/3, 1/2, 2/3, 4/5, 5/6, 7/8. The constraint length K is equal to 7 and the polynomials are $g_0 = 171$, $g_1 = 133$. The minimum distance, d_{free}, for the described enc/dec are calculated from the convolutional Trellis diagram with $K = 7$ and are equal to 15, 10, 6, 5, 4 and 3, respectively, for the six mentioned code rates [31]. Given MSS the Ethernet value of 1460 B and m, 365 B, the values of function $f(R_c, b)$ can be derived numerically. Fig. 2 shows the TCP PER, e, against wireless link BER, b, for the various code rates. As discussed earlier, knowing the BER of the wireless channel, the link-level FEC code rate for the corresponding TCP flow can be found by moving horizontally in the curve of Fig. 2 from the initial value of PER to its desired value that is the result of solving problem (P).

Various techniques are presented in the literature to estimate the BER of the wireless channel for example, transmitting pilots, modelling the channel with all the known effects or using the soft output from the rake receiver [32]. An accurate way to estimate the BER in the multicarrier system is the measurement and/or simulation of all possible symbols that is not feasible. Therefore the presented methods in the literature introduce sets of the OFDM symbols for the estimation of

BER, thus the complexity is reduced while the accuracy is relatively maintained [33].

5.3 Simulation results

The existing 15 wireless clients are connected to 15 servers (as plotted in Fig. 3). The combination of TCP flavors are four TCPR, four TCPNR, four TCPW, and three TCPR with enabled SACK option. The cell is assumed to have a circular shape with the wireless AP at the centre. In the scenarios one, and three users are randomly distributed in the cell using the method described earlier [29]. In scenario two all users keep the fix distance from the AP in each simulation run. Distance of users from the wireless AP–d–refers to a random location on a circle around the AP (as plotted in Fig. 5) with the radius d. Locating mobile users on these circles and keep their distance from the AP constant for each simulation run, keep the path loss effect fix, thus examine the random slow fading effect. On the other hand, changing the distance from the AP – moving to another circle – include the effect of path loss which is the deterministic part of the channel loss. Every presented simulation scenario is performed five times and the results are the average of these five simulations.

Our benchmark is a system which simply adapts its FEC code rate based on the channel quality. This code rate is assumed to be taken from one of six available rates (see Fig. 2) based on the channel BER, and is updated once every 2 s. We compare the results of our TCP-aware optimisation scheme with this benchmark. The results in terms of fairness index against RTT, and mobile users' distances to the AP are presented and compared with the benchmark problem. Aggregated end-to-end throughput is also studied in the scenarios detailed in the sequel.

5.3.1. Simulation scenario one: various end-to-end RTT values: The first simulated scenario studies the achieved fairness by our proposed scheme in different end-to-end path RTT values. To this end, in consequent run of simulation scenario one, the RTTs to the 15 FTP servers in the wired part of the network, are varied from 20 to 300 ms. In this scenario, users are uniformly distributed within the cell with radius $R = 60$ m – the uniform distribution of users in the cell is with respect to the earlier described method. In Fig. 6 that shows the achieved fairness index against RTT, the average improvement of 30% in fairness index can be seen.

5.3.2 Simulation scenario two: changing users' distance from the AP: In scenario two, the effect of moving mobile users further(closer) from(to) the wireless AP on the achieved fairness is investigated. In this respect, in each run of simulation scenario two, the all 15 mobile users are located on the specific circle – as shown in Fig. 5 – that results in changing the users' distance from the AP in the from 20 to 70 m. The same set of uniformly distributed random RTT values are in place for all of the simulation runs of scenario two, which are in the range [20 ms, 300 ms]. Moreover, the 15 servers are FTP servers. The achieved

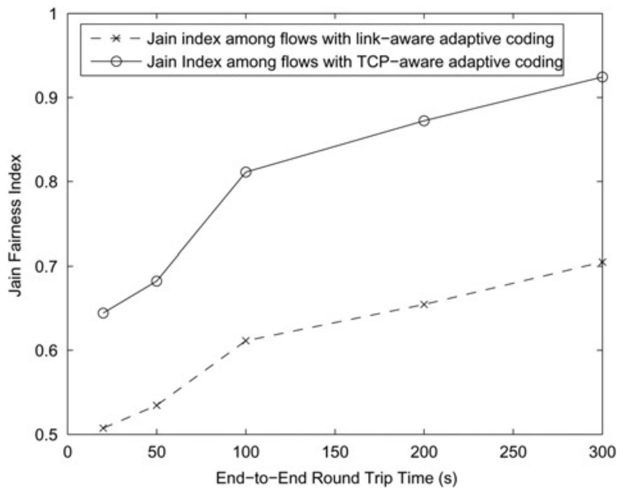

Figure 6 *Scenario one: Jain's index against end-to-end RTT (ms)*

fairness in this scenario is plotted in Fig. 7, where the average of 30% improvement can be observed.

Additional observation from this figure reveals two main points. Firstly, the level of achieved fairness is not affected by moving all the 15 mobile users further(closer) from(to) the wireless AP – unlike the observation from Fig. 6. Secondly, having the diversity among end-to-end RTTs in this scenario, which is caused by assigning random RTTs to each TCP flow, results in the high level of achieved fairness using our scheme – over 85% fairness for every run of the simulations in Fig. 7.

In addition to the achieved fairness among competing TCP flows, we examine the overall end-to-end throughput accomplished using our proposed scheme. Previously, we claim that using this scheme fairness is increased significantly with not considerable degradation in the overall throughput. This can be seen in Fig. 8, in which the aggregated end-to-end throughput is plotted against the users' distance from the AP. The results present in this figure show that the overall

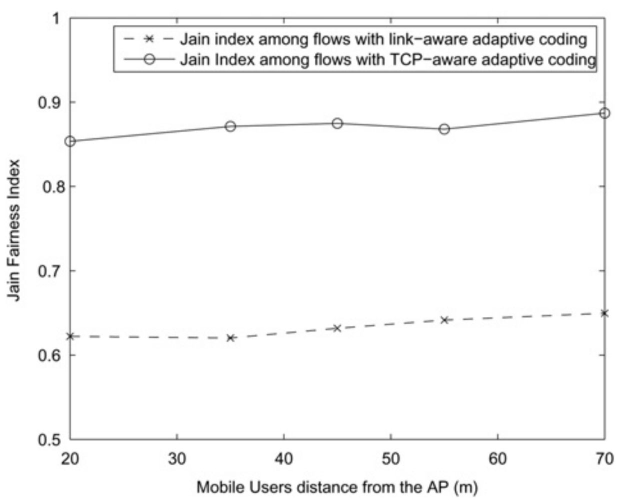

Figure 7 *Scenario two: Jain's index against users distance from the AP (m)*

doi: 10.1049/iet-com.2009.0394

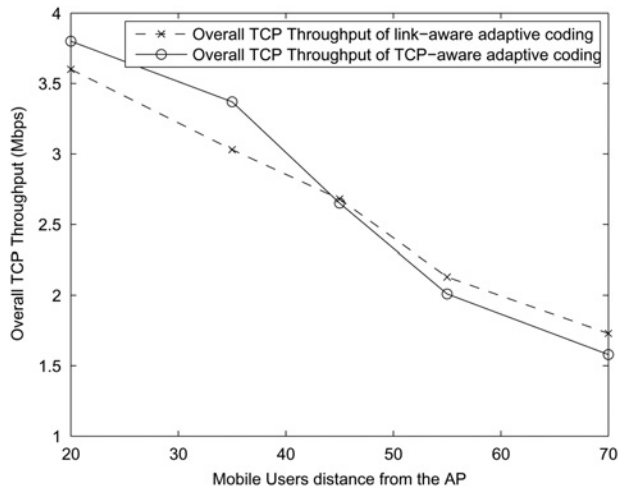

Figure 8 *Scenario two: overall end-to-end throughput against users distance from the AP (m)*

TCP throughput is slightly increased when users are close to the AP and slightly decreased as users get further from the AP.

5.3.3 Simulation scenario three: different traffic types of FTP and HTTP: Scenario three investigates the performance of our proposed scheme under different traffic types: FTP and HTTP traffic. In the two runs of this scenario the RTT values are random values uniformly distributed in the range [20 ms, 300 ms], and users are uniformly distributed within the cell with radius $R = 60$ m. The first run of scenario three is performed with 15 FTP servers, and the seond run is performed with 15 HTTP servers with the properties described in Section 5.1. Table 2 presents the results of this scenario. From the values in Table 2, we can see that the improvement in the fairness index in both the traffic types is significant, and approximately 30%. It is also clear that the fairness index is not affected by the different traffic types.

5.3.4 Overall results: The results presented in Figs. 6–8 and Table 2 show not only that the fairness index is improved using our scheme – by approximately 30% – but also the overall TCP throughput is minimally affected. More generally, the proposed optimisation framework yields improvements in fairness in a variety of conditions regarding end-to-end RTTs, users' distribution within the cell, and different traffic types. The improvement in fairness is significant in both the analysis (Table 2) and the OPNET simulation scenarios, while at the same time the effect on the end-to-end performance is minimal. Moreover, increasing the diversity among the end-to-end RTT, more significant improvement in fairness among the corresponding TCP flows can be seen.

Table 2 Jain's index in scenarios three with FTP and HTTP applications

	Jain's fairness index	
traffic type	FTP	HTTP
link-aware adaptive coding	0.615	0.653
TCP-aware adaptive coding	0.832	0.875

6 Conclusions

In this paper, we have presented a cross-layer mechanism to dynamically set the FEC rate at the link layer on a per-flow basis, given the end-to-end TCP flavour of that flow as detected at the wireless link. Utilising information on the TCP flavour of each flow, we have detailed a framework to improve fairness among heterogeneous TCP flows that compete over a wireless links. To allow real-time implementation, we have presented a heuristic to ascertain the optimal link-layer coding rate at the wireless base station. The convergence properties of this proposed heuristic have been studied.

Our analysis indicates that using the proposed scheme fairness can be improved significantly; moreover, overall TCP throughput is minimally affected. Comparative simulation studies using the OPNET simulation platform have also been performed, where the utilised implementations of different TCP flavours precisely match their respective RFCs. These simulations, under various packet loss probabilities and RTT conditions, show an improvement in fairness of approximately 30%.

7 Acknowledgments

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.

8 References

[1] TSAOUSSIDIS V., MATTA I.: 'Open issues on TCP for mobile computing', *Wirel. Commun. Mob. Comput.*, 2002, **2**, (1), pp. 3–20

[2] MEDINA A., ALLMAN M., FLOYD S.: 'Measuring the evolution of transport protocols in the internet', *ACM Comput. Commun. Rev.*, 2005, **35**, (2), pp. 37–52

[3] MAHMOODI T., FRIDERIKOS V., HOLLAND O., AGHVAMI A.H.: 'Cross-layer optimization to maximize fairness among TCP flows of different TCP flavors'. Proc. IEEE GLOBECOM'08, New Orleans, LA, USA, November 2008

[4] BARAKAT C., ALTMAN E.: 'Bandwidth tradeoff between TCP and link-level FEC', *Comput. Netw.*, 2002, **39**, pp. 133–150

[5] BARMAN D., MATTA I., ALTMAN E., AZOUZI R.: 'TCP optimization through FEC, ARQ and transmission power tradeoffs'. Proc. Second Int. Conf. on Wired/Wireless Internet Communication, Frankfurt, Germany, 2004

[6] VACIRCA F., VEBDUCTIS D., BAIOCCHI A.: 'Optimal design of hybrid FEC/ARQ schemes for TCP over wireless links with Rayleigh fading', *IEEE Trans. Mob. Comput.*, 2006, **5**, (4), pp. 289–302

[7] VACIRCA F., VENDICTIS A.D., TODINI A., BAIOCCHI A.: 'On the effects of ARQ mechanisms on TCP performance in wireless environments'. Proc. IEEE GLOBECOM'03, San Francisco, CA, USA, December 2003, vol. 2, pp. 671–675

[8] MAHMOODI T., FRIDERIKOS V., HOLLAND O., AGHVAMI A.H.: 'Cross-layer design to improve wireless tcp performance with link-layer adaptation'. Proc. IEEE VTC Fall, Baltimore, MD, USA, October 2007, pp. 1504–1508

[9] PROAKIS J.G.: 'Digital communications' (McGraw-Hill, 1995, 3rd edn.)

[10] PILOSOF S., RAMJEE R., SHAVITT Y., SINHA P.: 'Understanding TCP fairness over wireless LAN'. Proc. IEEE INFOCOM'03, San Francisco, CA, USA, March 2003, vol. 2, pp. 863–872

[11] BLEFARI-MELAZZI N., DETT A., HABIB I., ORDINE A., SALSANO S.: 'TCP fairness issues in IEEE 802.11 networks: problem analysis and solutions based on rate control', *IEEE Trans. Wirel. Commun.*, 2007, **6**, (4), pp. 1346–1355

[12] JAIN R., CHIU D., HAWE W.: 'A quantitative measure of fairness and discrimination for resource allocation in shared computer systems'. DEC Research Report TR-301, September 1984

[13] ELENA LOPEZ-AGUILERA F.R.A.D., MARTIN HEUSSE, CASADEMONT J.: 'Performance of wireless LAN access methods in multicell environments'. Proc. IEEE GLOBECOM'06, San Francisco, CA, USA, November 2006

[14] JAISWAL S.: 'Measurement in the middle: inferring end-end path properties and characteristics of TCP connections through passive measurement'. PhD dissertation, University of Massachusetts Amherst, September 2005

[15] STEVENS W.R.: 'TCP/IP illustrated, vol. I (The protocols)' (Addison Wesley, 2000)

[16] JACOBSON V., BRADEN R., BORMAN D.: 'TCP extensions for high performance'. IETF RFC 1323, May 1992

[17] JIANG H., DOVROLIS C.: 'Passive estimation of TCP round-trip times', *ACM Compnt. Commun. Rev.*, 2002, **32**, (3), pp. 75–88

[18] ALLMAN M., PAXSON V., STEVENS W.: 'TCP congestion control'. IETF RFC 2581, April 1999

[19] FLOYD S., HENDERSON T., GURTOV A.: 'The NewReno modification to TCP's fast recovery algorithm'. IETF RFC 3782, April 2004

[20] WANG R., YAMADA K., SANADIDI M.Y., GERLA M.: 'TCP with sender-side intelligence to handle dynamic, large, leaky pipes', *IEEE J. Sel. Areas Commun.*, 2005, **23**, (3), pp. 235–248

[21] PADHYE J., FIROIU V., TOWSLEY D.F., KUROSE J.F.: 'Modeling TCP Reno performance: a simple model and its empirical validation', *IEEE/ACM Trans. Netw.*, 2000, **8**, (2), pp. 133–145

[22] DUNAYTSEV R., KOUCHERYAVY Y., HARJU J.: 'TCP NewReno throughput in the presence of correlated losses: the slow-but-steady variant'. Proc. IEEE INFOCOM'06, Global Internet Workshop, Barcelona, Spain, April 2006, pp. 115–120

[23] ZANELLA A., PROCISSI G., GERLA M., SANADIDI M.Y.: 'TCP Westwood: analytic model and performance evaluation'. Proc. IEEE GLOBECOM'01, St. Antonio, Texas, USA, November 2001, vol. 3, pp. 1703–1707

[24] CHEN Z., BU T., AMMAR M., TOWSLEY D.F.: 'Comments on modeling TCP Reno performance: a simple model and its empirical validation', *IEEE/ACM Trans. Netw.*, 2006, **14**, (2), pp. 451–453

[25] SIKDAR B., KALYANARAMAN S., VASTOLA K.S.: 'Analytic models for the latency and steady-state throughput of TCP Tahoe, Reno, and SACK', *IEEE/ACM Trans. Netw.*, 2003, **11**, (6), pp. 959–971

[26] BIAZ S., VAIDYA N.H.: 'De-randomizing congestion losses to improve TCP performance over wired-wireless networks', *IEEE/ACM Trans. Netw.*, 2005, **13**, (3), pp. 596–608

[27] 'The TCP westwood model for the ns2 network simulator,' www.cs.ucla.edu/nrl/hpi/tcpw

[28] BOYDS S., VANDENBERGHE L.: 'Convex optimization' (Cambridge University Press, 2004)

[29] GÖRG C., VORNEFELD U.: 'Uniform distribution of users in a circle'. The 100 most frequent errors in performance evaluation

[30] LEE J.J., GUPTA M.: 'A new traffic model for current user web browsing behavior' (White paper, Intel Corporation, 2007)

[31] 'Convolutional encoder and viterbi decoder' (4i2i Communication Ltd., 2001, 3rd edn.)

[32] SMIT L., SMIT G., HURINK L., KOKKELER A.: 'Soft output bit error rate estimation for WCDMA'. Proc. Personal Wireless Commun., Venice, Italy, September 2003

[33] VANDERSTEEN G., WAMBACQ P., ROLAIN Y., SCHOUKENS J.: 'Efficient bit-error-rate estimation of multicarrier transceivers'. Proc. Design, Automation and Test in Europe, Munich, Germany, March 2001, pp. 164–168

doi: 10.1049/iet-com.2009.0394