INTERNATIONAL JOURNAL OF COMMUNICATION SYSTEMS
Int. J. Commun. Syst. (in press)
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/dac.894

Effort/gains dynamics in heterogeneous networks

L. Mamatas*,† and V. Tsaoussidis

Department of Electrical and Computer Engineering, Demokritos University of Thrace, Greece

SUMMARY

We investigate the behaviour of TCP(α, β) protocols in the presence of wireless networks. We seek an answer to strategic issues of maximizing energy and bandwidth exploitation, without damaging the dynamics of multiple-flow equilibrium. We take a fresh perspective on protocol design: What is the return of the effort that a protocol expends? Can we achieve more gains with less effort? We study first the design assumptions of TCP(α, β) protocols and discuss the impact of equation-based modulation of α and β on protocol efficiency. We introduce two new measures to capture protocol behaviour: the 'Extra Energy Expenditure' and the 'Unexploited Available Resource Index'. We confirm experimentally that, in general, smoothness and responsiveness constitute a tradeoff; however, we show that this tradeoff does not graft its dynamics into a conservative/aggressive behaviour, as it is traditionally believed. We uncover patterns of unjustified tactics; our results suggest that an adaptive congestion control algorithm is needed to integrate the dynamics of heterogeneous networks into protocol behaviour. Copyright © 2007 John Wiley & Sons, Ltd.

Received 23 January 2006; Revised 15 March 2007; Accepted 16 April 2007

KEY WORDS: congestion control; TCP; transport layer; energy efficiency

1. INTRODUCTION

Transmission control of reliable protocols, as exemplified by TCP [1], is based on somewhat 'blind' increase/decrease window mechanism that exploits the bandwidth availability dynamically and, meanwhile, avoids persistent congestion. The adjustments are modelled on the additive increase/multiplicative decrease (AIMD) algorithm from the perspective of fair resource allocation and efficient resource utilization [2]. AIMD is the core algorithm of standard TCP and is becoming the core algorithm of all transport protocols that support congestion control functions [3].

Several different mechanisms/protocols have been proposed regarding the transport layer. A thorough analysis of the different approaches to congestion control in transport protocols can be

*Correspondence to: L. Mamatas, Department of Electrical and Computer Engineering, Demokritos University of Thrace, Greece.
†E-mail: emamatas@ee.duth.gr

found in [4]. For example, TFRC [5, 6] calculates Throughput by incorporating the loss event rate, round-trip time and packet size. TCP-Vegas [7] estimates the level of congestion using Throughput-based measurements. TCP-Vegas demonstrates that measurement-based window adjustments is a viable mechanism; however, the corresponding estimators can be improved. In TCP-Westwood [8], the sender continuously measures the effective bandwidth used by monitoring the rate of returned ACKs. TCP-Real [9] uses wave patterns: a wave consists of a number of fixed-sized data segments sent back to back, matching the inherent characteristic of TCP to send packets back to back. The protocol computes the data-receiving rate of a wave, which reflects the level of contention at the bottleneck link. Bimodal congestion avoidance and control mechanism [10] compute the fair share of the total bandwidth that should be allocated for each flow, at any point, during the system's execution. TCP-Jersey [11] operates based on an 'available bandwidth' estimator to optimize the window size when network congestion is detected.

However, a concern is how efficiently do protocols administer the network resources; that is, whether the (whatever) gain is proportional to the expended effort. Also, how can we measure effectively the effort-/gain-relative performance of a transport protocol? The evaluation of the proposed transport protocols needs to focus on the effort/gain dynamics of their corresponding mechanisms too.

The problems of standard TCP have been mainly investigated from two different perspectives, namely the application requirements and the characteristics of the underlying networks. The former expounds the impact of the transmission gaps caused by halving the transmission rate during congestion on the quality of delay-sensitive applications. Authors in [5, 12–14] propose TCP-friendly protocols (which are defined as the protocols that share the available bandwidth fairly with applications built on TCP) that satisfy two fundamental goals: (i) to achieve smooth window adjustments. This is done by reducing the window decrease ratio during congestion. (ii) To compete fairly with TCP flows. This is approached by reducing the window increase factor according to a steady-state TCP Throughput equation. It has been effectively established that TCP can achieve application-oriented improvements by favouring smoothness (which is defined as the level of window oscillations) using a gentle backward adjustment upon congestion, at the cost of lesser responsiveness (which is the speed to approach an equilibrium)—through moderated upward adjustments. The latter perspective unfolds the need for error detection and classification that would permit a responsive strategy, oriented by the nature of the error detected (congestion in wired networks *versus* transient random errors in wireless networks) [15]. As we show, implementation of such strategy requires occasionally a more responsive TCP. Our approach, however, is dominated by the distinctive characteristics and requirements of wireless networks: we address issues of energy and wireless error recovery, through a parallel study of a smooth/responsive protocol design and an aggressive/conservative outcome. Note that the conservative-through-to-aggressive behavioural spectrum reflects the effort a protocol expends. The real issue, therefore, is how much this effort is invested into efficient transmission.

Our contribution centres around that issue, precisely. In order to measure protocol efficiency we introduce two new measures of performance: extra energy expenditure (EEE) quantifies the additional effort expended throughout protocol operations that did not return corresponding gains. Unexploited available resources (UAR) is the measure that quantifies the missed opportunities for error-free transmission that a protocol experiences. Traditional measures cannot capture precisely such behaviour since they lack a parameter that corresponds to optimal performance.

TCP(α, β) protocols parameterize the congestion window increase value α and decrease ratio β, where the sender's window size is increased by α if there is no packet loss in a round-trip time,

DOI: 10.1002/dac

and the window is decreased to β times the current value if there is a loss indication. We discuss the impact of the smoothness/responsiveness tradeoff on protocol performance, assuming that it follows strictly the friendliness-oriented α/β tradeoff. A natural question is therefore 'under what network conditions can we achieve efficiency; and how do we define efficiency'. Having shown in previous work [16] that a protocol for wireless networks may need to be occasionally more conservative and occasionally more aggressive, we attempt to explore how this tradeoff is shaped by the responsive or smooth protocol strategy. In our discussion below, we refer to three classes of TCP(α, β) protocols: (i) standard New Reno TCP(1, $\frac{1}{2}$); (ii) responsive TCP(α, β), with *relatively* low β value and high α value; and (iii) smooth TCP(α, β), with *relatively* high β value and low α value.

We compare the performance of our TCP(α, β) versions in heterogeneous (wired and wireless) networks and in static and dynamic[‡] environments. Based on the assumptions of equation-based congestion control and on experimental data, we arrive at the conclusion that protocols, which are based entirely on the α/β tradeoff may be adequate for specific applications, networks and scenarios; however, they are inappropriate for several other occasions.

We organized the paper as follows: we give an overview of TCP(α, β) protocols in Section 2 and we discuss their inherent assumptions. In Section 3 we define new performance measures. In Section 4 we present our testing methodology and in Section 5 we analyse the results of our experiments. Finally, in Section 6 we highlight our conclusions.

2. TRADING α FOR β

A Throughput equation for standard TCP was first introduced in [17]. GAIMD [14] extends the equation to include parameters α and β:

$$T_{\alpha,\beta}(p, \mathrm{RTT}, T_0, b) = \frac{1}{\mathrm{RTT}\sqrt{\dfrac{2b(1-\beta)}{\alpha(1+\beta)}p} + T_0 \min\left(1, 3\sqrt{\dfrac{(1-\beta^2)b}{2\alpha}p}\right) p(1+32p^2)} \tag{1}$$

where p is the loss rate; T_0 is the retransmission timeout value; b is the number of packets acknowledged by each ACK. The overall Throughput of TCP-Friendly (α, β) protocols is bounded by the average Throughput of standard TCP($\alpha = 1$, $\beta = 0.5$), which means that Equation (2), which is derived from (1) (see [14]), could provide a rough guide to achieve friendliness:

$$T_{\alpha,\beta}(p, \mathrm{RTT}, T_0, b) = T_{1,0.5}(p, \mathrm{RTT}, T_0, b) \tag{2}$$

Authors of [13] derive from (1) and (2) a simple relationship for α and β:

$$\alpha = 4(1-\beta^2)/3 \tag{3}$$

Based on experiments, they propose a $\beta = 7/8$ as the appropriate value for the reduced the window (i.e. less rapidly than TCP does). For $\beta = 7/8$, (3) gives an increase value $\alpha = 0.31$.

The observations of the window dynamics and event losses are frequently assumed within a time period of a *congestion epoch* [5], which reflects the *uninterrupted growing lifetime of congestion*

[‡] From the perspective of the participating flows with criterion whether their number is fixed or not.

DOI: 10.1002/dac

window. More precisely, a congestion epoch begins with βW packets, increased by α packets per RTT and reaching a congestion window of W packets, when a packet is dropped. The congestion window is then decreased to βW. Hence, a congestion epoch involves

$$n = (1 - \beta) * W/\alpha + 1\text{RTTs} \tag{4}$$

Assuming that the capacity of the bottleneck link is B packets per second and the number of active flows going through the bottleneck router is N, and assuming a control system as in [2], we further calculate that

$$W = B * \text{RTT}/N \tag{5}$$

We can easily observe that *it takes several RTTs for a small α to pay back the bandwidth credit of a high β.*

Equation (1) is modelled by calculating the average Throughput over a congestion epoch, which is associated with several RTTs. Since Equation (1) gives the *steady-state* TCP Throughput, in a dynamic network where conditions changing rapidly, friendliness might not be attained. More precisely, based on (4) we conclude that (1) and (2) can be achieved at a time n RTTs or later since multiple drops will extend further the time of convergence. Based on (4) and (5), we further conclude that the time period required for (1) and (2) to hold is in reverse proportion to contention within a fixed bandwidth channel; the smaller the number of flows, the larger the window and therefore the longer the convergence time. By the same token, the fact that a responsive protocol can exploit bandwidth better suggests that lower contention is a favourable case for such protocols.

This analysis implies that smooth protocols may be more aggressive (since they consume temporarily more bandwidth) in the presence of transient errors, while they may behave more conservatively, due to their low increasing rate, when multiple drops force the multiplicative decrease factor to adjust the congestion window back to its initial value. This can be justified by a hidden assumption behind (3): when packet drops occur at the end of the congestion epoch, the window decreasing by a factor of $(1-\beta)$ is applied only once. However, multiple packet drops could cause the window size to decrease multiple times, or they could also cause the retransmission timer to expire. At the end, it is possible that the window size and the ssthresh could be decreased down to two segments, even with smooth backward adjustments. Under such scenarios, the performance of applications (including real-time applications) is not affected by how slowly the sender reduces its sending rate, but rather by how fast it can recover from the error and restore its sending rate. Note that our scenario is not unrealistic. For example, in mobile networks, burst-correlated errors and handoffs generate this kind of error pattern. The aggressiveness of responsive TCP may be a desirable behaviour. We confirm our statements experimentally in Section 5.

3. MEASURES FOR EVALUATING EFFORT/GAINS DYNAMICS

For a proper evaluation of effort/gains dynamics, we propose new measures to monitor:

- the effort expended from a protocol;
- the effective utilization of available resources;
- the achieved gains of the effort from the applications viewpoint.

Additionally, the new measures need to be combined with traditional measures.

DOI: 10.1002/dac

The system Goodput measures the overall system efficiency in bandwidth utilization. The system Goodput is defined as

$$\text{Goodput} = \text{Original\_Data}/\text{Connection\_Time}$$

where Original_Data is the number of bytes delivered to the high-level protocol at the receiver (i.e. excluding retransmitted packets and overhead) and Connection_Time represents the amount of time required for the data delivery.

Fairness is captured by the Fairness Index, derived from the formula given in [2] and defined as

$$\text{Fairness} = \frac{\left(\sum_{i=1}^{n} \text{Throughput}_i\right)^2}{n\left(\sum_{i=1}^{n} \text{Throughput}_i^2\right)}$$

where Throughput_i is the Throughput of the ith-flow and n is the flow number. This Fairness Index provides a sort of 'average-case' analysis used by most researchers. In order to conduct a 'worst-case' analysis and provide a tight bound on Fairness, we propose the Worst-Case Fairness as

$$\text{Worst-case Fairness} = \frac{\min_{1\leqslant i\leqslant n} \text{throughput}_i}{\max_{1\leqslant i\leqslant n} \text{throughput}_i}$$

The range of Worst-case Fairness is within [0, 1] (1 represents the higher Fairness). To demonstrate why Worst-case Fairness is introduced, consider a scenario of 6 flows, with Throughputs 9, 9.5, 8.5, 9, 9 and 6Mbps, respectively. The traditional 'average-case' Fairness Index is 0.982, while the Worst-case Fairness is 0.667. Compare this scenario with a perfectly fair case in which all flows achieve 9.5 Mbps, and both the 'average-case' Fairness Index and Worst-case Fairness Index are 1.0. The difference between the first scenario and the ideal case cannot be obviously distinguished by the 'average-case' Fairness Index. In the first scenario, the system is fair in general, but is particularly unfair to the 6th flow. This unfairness to a very small fraction of flows can only be captured by the Worst-case Fairness.

In order to validate the efficiency of the protocols in real-time/multimedia applications, we assume an application, which demands to receive at least one packet every 100 ms. Due to the sending window fluctuation and the transmission gaps of TCP(α, β), there are instances of data being unavailable to the application (because the packets were delayed more than 100 ms). We use the Realtime Performance Index in order to measure the protocol's real-time performance:

$$\text{Realtime Performance Index} = \frac{\text{Packets\_Delivered\_in\_Time}}{\text{Packets\_Delivered}}$$

where Packets_Delivered_in_Time is the number of packets which have been received by the application in time and Packets_Delivered the total number of packets received by the application.

DOI: 10.1002/dac

In order to capture the amount of *extra* energy expended, we introduce a new metric. Extra Energy Expenditure (EEE or 3E) [18] attempts to capture the *extra* energy expended due to protocol operation and not just the expended energy. That is, a protocol may transmit when there are windows of opportunities for error-free transmission, without expending extra energy, or *vice versa*. In contrast, it may miss opportunities for transmission, expending energy (even in an idle state) and extending communication time. 3E attempt to capture extra energy expenditure as an associated result of Goodput, Throughput, and maximum Throughput, each one represented as a moving point on a line. 3E takes into account the difference of achieved Throughput from maximum Throughput ($\text{Throughput}_{\text{max}}$) for the given channel conditions along with the difference of Goodput from Throughput, attempting to locate the Goodput as a point within a line that starts from 0 and ends at $\text{Throughput}_{\text{max}}$. The metric 3E takes values from 0 to 1, attempting to capture both distances:

$$\text{EEE} = a\frac{\text{Throughput} - \text{Goodput}}{\text{Throughput}_{\text{max}}} + b\frac{\text{Throughput}_{\text{max}} - \text{Throughput}}{\text{Throughput}_{\text{max}}}$$

where $a = 1$ and $b = 0.3$.

The a and b parameters follow the behaviour of a specific network device. In many cases, a sophisticated energy efficient protocol consumes more energy than it is designed to, due to lack of sophistication of the network device. The EEE metric should be adjusted to the network device in order to follow accurately the impact of the network communication on the specific battery's lifetime. In [19], we show how this adjustment can be made.

The ideal EEE is the EEE produced by an ideal device. We assume that an ideal network device is energy efficient and sophisticated in the sense that its states correspond always to the states of the transport protocol (i.e. when the protocol suspends transmission the device remains on an idle state). Therefore, this device allows the transport protocol to operate on its maximum energy efficiency. According our results, the EEE metric normalized with the parameters $a = 1$ and $b = 0.3$ behaves almost ideally. Practically, we assume that the ideal network device consumes the 30% of the energy in the idle state (set by parameter b).

When Goodput approaches Throughput, which approaches 0, the extra expenditure is only due to waiting time (probably in an idle state). We assume that the extra expenditure at this stage is 0.3 (the first term is 0). Instead, when $\text{Goodput} = \text{Throughput} = \text{Throughput}_{\text{max}}$ the extra expenditure is 0, since all the expended energy has been invested into efficient transmissions. Also, when $\text{Throughput}_{\text{max}} = 100$, $\text{Throughput} = 99$, $\text{Goodput} = 1$, the extra expenditure due to unsuccessful retransmission increases to an almost maximum value (0.993).

We need to introduce another metric as well, in order to capture the level of Unexploited Available Resources [18]. That is, how well are the windows of opportunities exploited for successful transmissions. More precisely, holding transmission when conditions call for transmission will perhaps result in minor energy expenditure but have a great cost on protocol Goodput. Reasonably, the case of $\text{Goodput} = \text{Throughput} = 0$ should not give us at this point a minor (as with the 3E metric) but a major penalty:

$$\text{UAR} = 1 - \left[a\frac{\text{Throughput}}{\text{Throughput}_{\text{max}}} + b\frac{\text{Goodput}}{\text{Throughput}}\right]$$

DOI: 10.1002/dac

where $a=0.5$ and $b=0.5$.[§] The UAR index ranges also from 0 to 1, expressing also a negative performance aspect.

The protocol efficiency can be studied from another perspective. Overhead is used as a measure to realize the protocol transmission effort to complete reliable data delivery:

$$\text{Overhead}=\frac{\text{Bytes\_Send}-\text{Original\_Bytes}}{\text{Bytes\_Send}}=1-\frac{\text{Goodput}}{\text{Throughput}}$$

Bytes_Send is the total bytes transmitted by TCP senders, while Original_Bytes is the number of bytes delivered to the higher level protocol by receivers, excluding retransmitted packets and TCP header bytes. This measure captures the portion of consumed bandwidth, or the percentage of the transmission energy (a scarce resource in mobile computing), that is wasted on packet retransmissions and protocol header overhead. It differs from EEE because it only captures the extra energy expenditure due to retransmissions.

4. EXPERIMENTAL METHODOLOGY

4.1. Evaluation plan

We have implemented our testing plan on the ns-2 network simulator. In our experiments, we used two different network topologies: (i) a typical single-bottleneck dumbbell (Figure 1); (ii) a complex topology with multiple bottlenecks and cross traffic (Figure 2). The router R1 is the bottleneck for the main traffic, which includes TCP flows between 'source nodes' to 'sink nodes'. The router R3 is another bottleneck for the competing main traffic and cross traffic, which includes TCP flows between 'peripheral source nodes' and 'peripheral sink nodes'. We used equal number of source and sink nodes. We simulated a heterogeneous (wired and wireless) network with ns-2 error models, which were inserted into the access links at the sink nodes. The Bernoulli model was used to simulate link-level errors with configurable bit error rate (BER). To simulate bursty wireless errors, a two-state (On/Off) error model is used, with the On/Off phase sojourn times exponentially distributed. The Off state is error free and the On state is configured with BER values. The Off and On states correspond to the Good and Bad states of a wireless channel, respectively. Bursty errors occur due to a large number of reasons associated mostly with movement of mobile terminals. Error models were configured on both (forward and reverse) directions of the link traffic. We did not use an ARQ mechanism in the link layer. Furthermore, occasionally we included mobility in order to monitor the behaviour of the network and its impact on the application in a situation of frequent handoffs. The scenarios with handoffs do not include lossy links. The number of flows occasionally changes for the different scenarios. The simulation time was fixed at 60 s, a time period deemed appropriate to allow all protocols to demonstrate their potential. Similar results can be attained from scenarios with higher simulation times (e.g. 120 s).

Due to the deterministic nature of the experiments, statistical validity is not an issue. In order to validate our statements, we selected and evaluated three protocols that satisfy the TCP-friendly

[§] The *a* and *b* values in both EEE and UAR indices can be modelled on the behaviour of a specific wireless network device.

DOI: 10.1002/dac

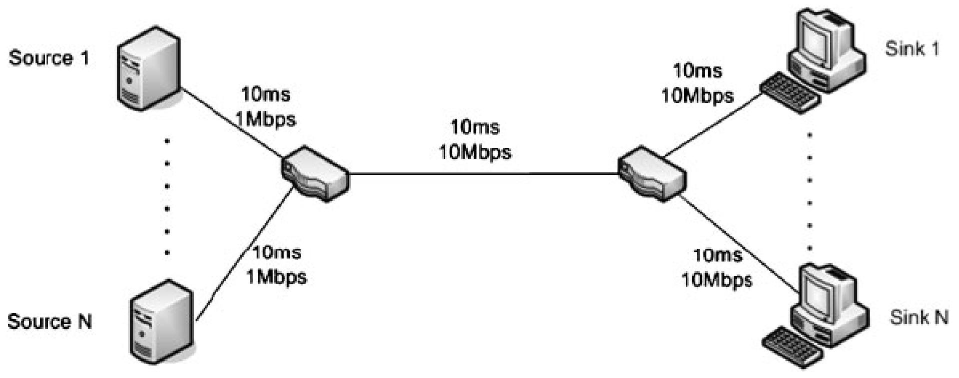


Figure 1. Dumbbell topology.

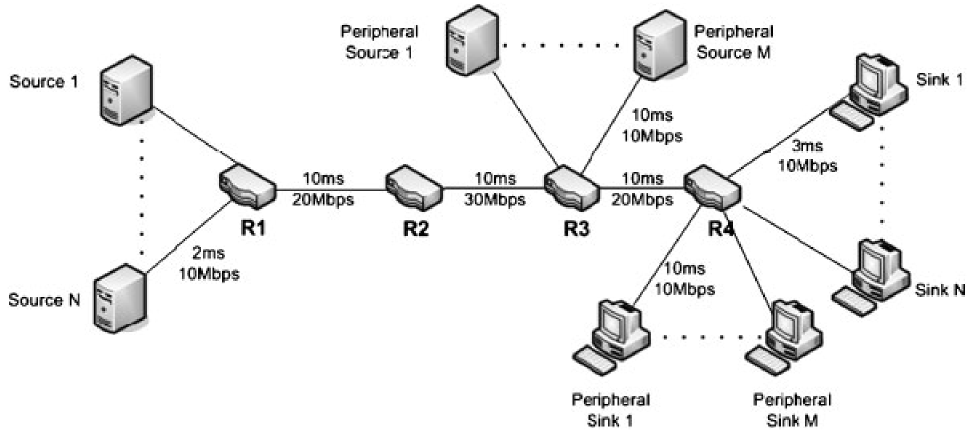


Figure 2. Complex topology.

equation [14]. We used standard New-Reno TCP (1, 0.5), a responsive New-Reno TCP (1.25, 0.25) and a smooth New-Reno TCP (0.31, 0.875).

In the scenarios without graduated contention decrease, FTP flows are entering the system within the first 2 s. All flows are fixed, during the remaining 58 s. In order to evaluate how efficiently and fairly the protocols can exploit available bandwidth, we used, additionally, scenarios with graduated contention decrease. All scenarios use FTP flows as the offered traffic.

5. RESULTS AND DISCUSSION

In this work, we comment on five different scenarios:

1. A simple wired scenario.
2. A wireless scenario with errors.

DOI: 10.1002/dac

3. A wireless scenario with errors and graduated contention increase.
4. A wireless scenario with long handoffs.
5. A wireless scenario with short handoffs.
6. A scenario with a complex topology.

We focus on protocol behaviour with respect to effort expended and gains achieved. Although effort expended is a rather unified measure (i.e. packets transmitted over time), achieved gain is application specific and cannot be expressed in a unified manner.

5.1. Simple wired scenario

In the simple wired scenario, while the responsive TCP improves the Fairness Index (Figure 3), the smooth TCP has less EEE (Figure 4). An adaptive transport protocol could follow, in this situation, either a responsive or a smooth transmission tactic, in order to be fair or energy efficient. Hence, there is a tradeoff between Fairness and Energy Efficiency. The responsive TCP does not exploit the network resources efficiently in cases of flows less than 40 (Figure 6). This results in an increased UAR index (Figure 5). Based on the effort/gains perspective, we note that the increased effort that the responsive TCP expends (Figure 4) makes the protocol more fair (Figure 3). However, it has no performance gains (Figure 6). In case of real-time/multimedia applications, the traditional TCP achieves more gains, as indicated by its Realtime Performance Index (Figure 7).

5.2. Wireless scenario with errors

In our second scenario, we simulated a heterogeneous environment with somewhat extreme, but random errors (error rate 0.02 BER). We measured performance, ranging the number of flows from 10 to 100. Based on Figure 10, we note that the responsive TCP has an increased UAR index. In this situation, the responsive TCP follows a conservative strategy. Therefore, there is not always a direct relationship between responsiveness and aggressiveness. Although the responsive TCP is not the most efficient (Figure 11), it appears more fair (Figures 8 and 9). This result is interesting and calls for further research.

5.3. Wireless scenario with errors and graduated contention increase

We simulated a heterogeneous environment with random errors (error rate 0.01 BER). Additionally, we gradually increased the contention level. While the protocols exhibit similar behaviour in terms of Goodput and Energy Efficiency (Figures 15 and 14), the responsive protocol is more fair (Figures 12 and 13) because it adjusts faster to the corresponding contention level. However, the smooth protocol has a better Realtime Performance Index, since the smooth behaviour reduces the packet jitter (Figure 16). Therefore, the same effort (Figure 14) can result in different gains for different transport mechanisms. A more sophisticated transport protocol, which is aware of the application's demands, could select a different strategy to reach the desired gain.

5.4. Wireless scenario with long handoffs

In this scenario, we used handoffs with 1-s duration. We measured performance, ranging the number of flows from 10 to 100. Although the smooth TCP is less fair (Figure 17), it outperformed the other protocols in terms of Extra Energy Expenditure (Figure 18) due to their increased overhead (Figures 20 and 21). The increased UAR index (Figure 19) shows that the smooth protocol

DOI: 10.1002/dac

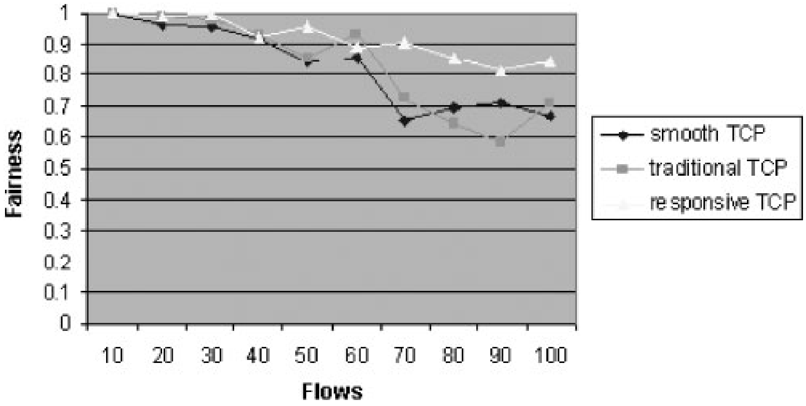


Figure 3. Fairness of TCP variations in a simple wired scenario.

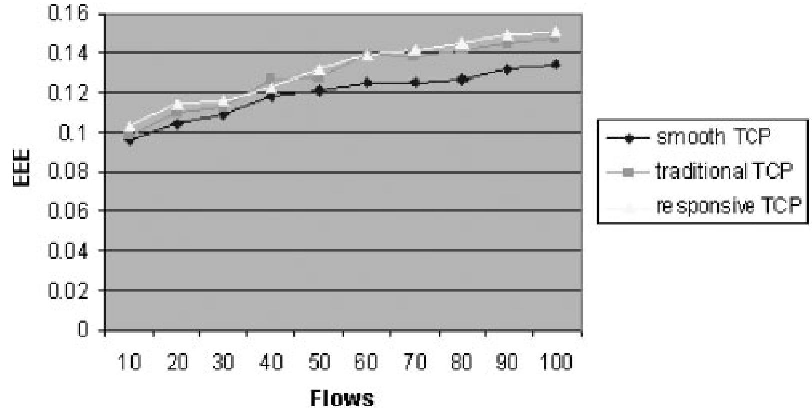


Figure 4. EEE of TCP variations in a simple wired scenario.

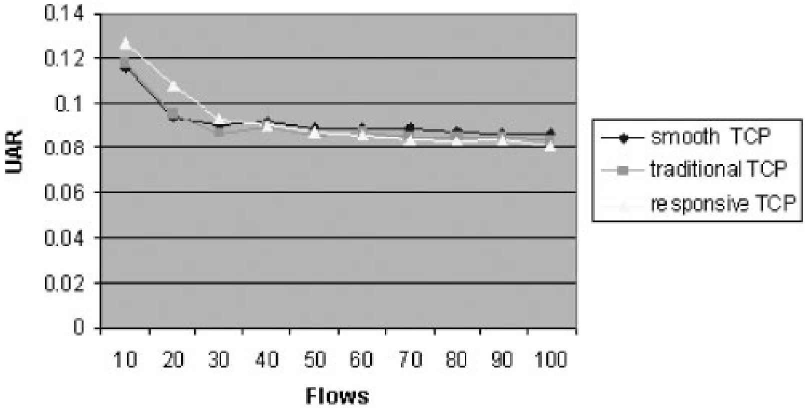


Figure 5. UAR of TCP variations in a simple wired scenario.

DOI: 10.1002/dac

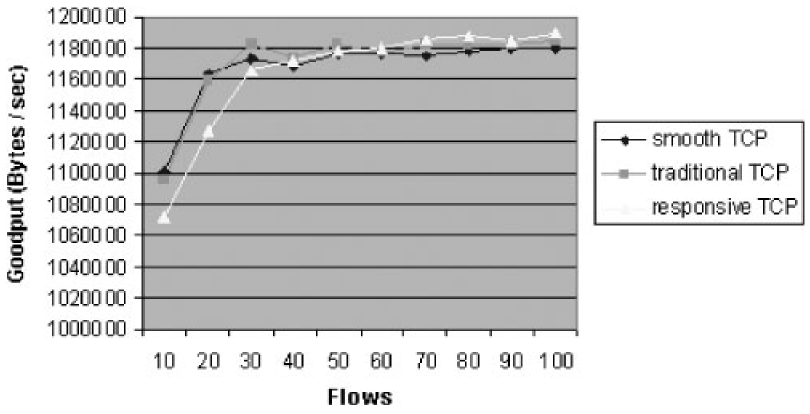


Figure 6. Goodput of TCP variations in a simple wired scenario.

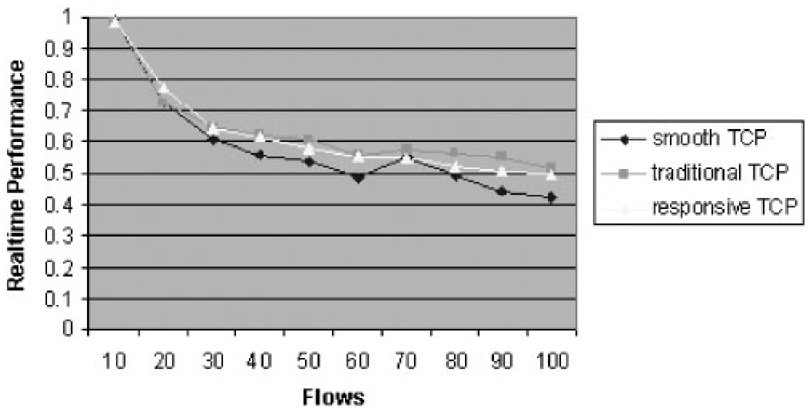


Figure 7. Real-time performance of TCP variations in a simple wired scenario.

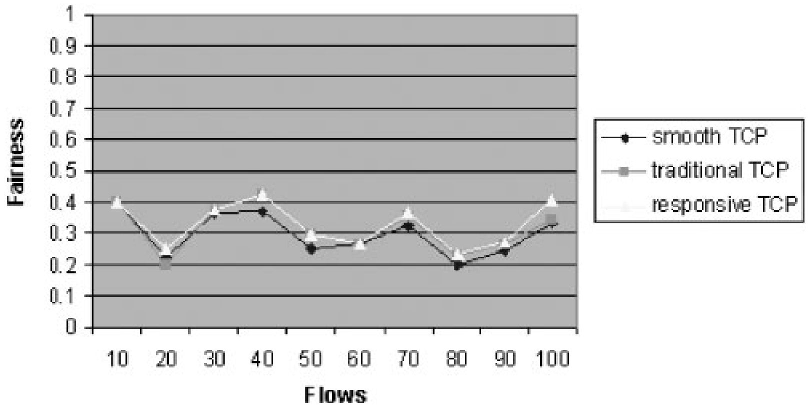


Figure 8. Fairness of TCP variations in a lossy environment.

DOI: 10.1002/dac

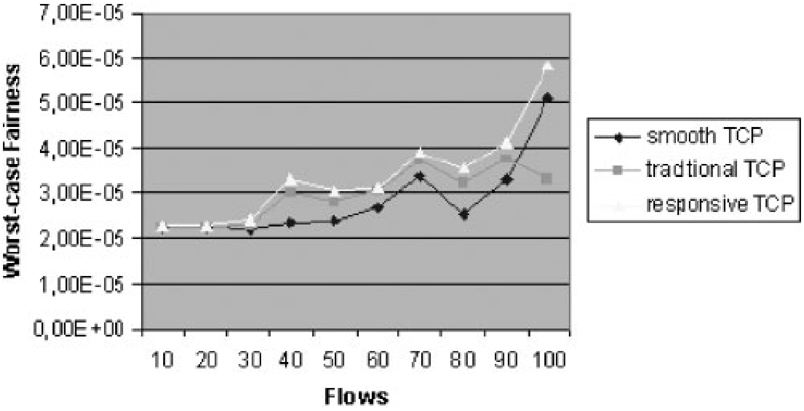


Figure 9. Worst-case fairness of TCP variations in a lossy environment.

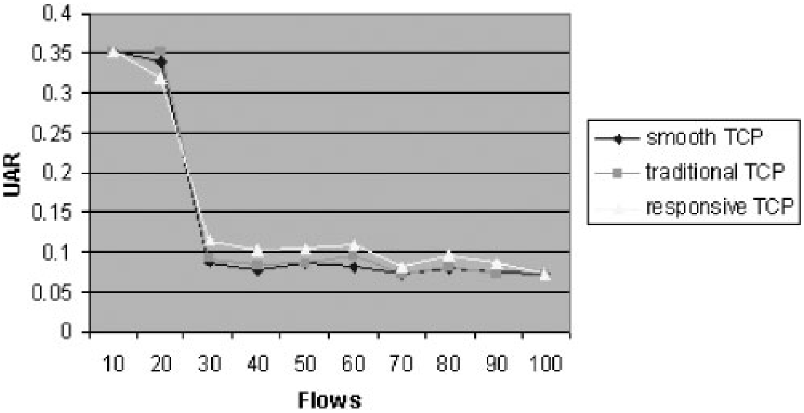


Figure 10. UAR of TCP variations in a lossy environment.

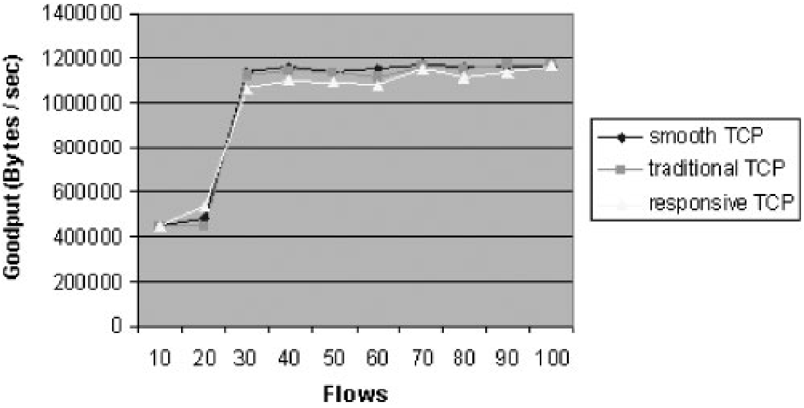


Figure 11. Goodput of TCP variations in a lossy environment.

DOI: 10.1002/dac

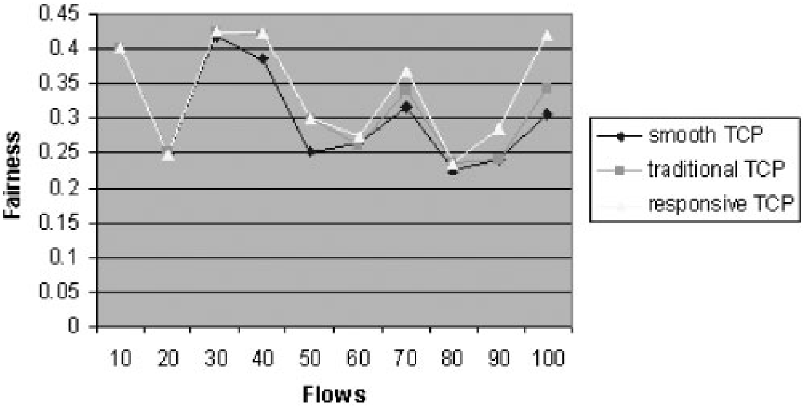


Figure 12. Fairness of TCP variations in heterogeneous scenario with graduated contention increase.

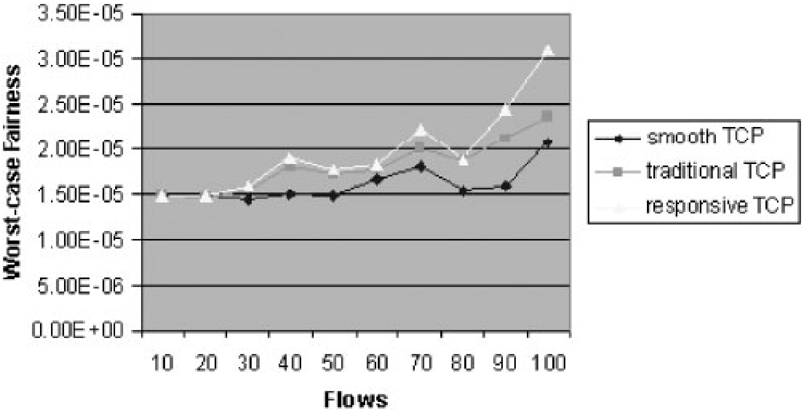


Figure 13. Worst-case fairness of TCP variations in heterogeneous scenario with graduated contention increase.

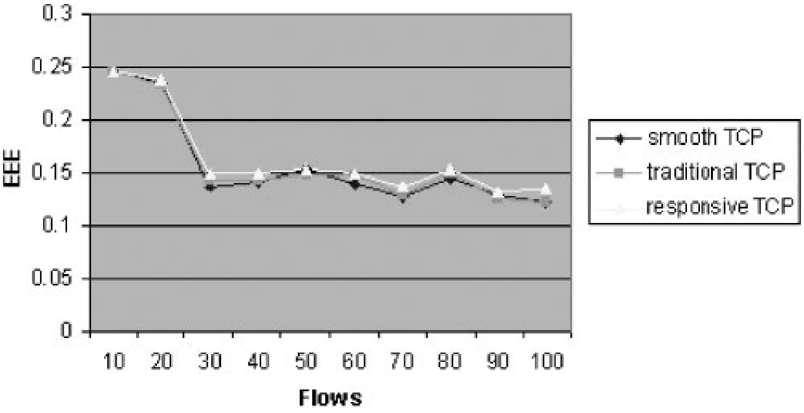


Figure 14. EEE of TCP variations in heterogeneous scenario with graduated contention increase.

DOI: 10.1002/dac

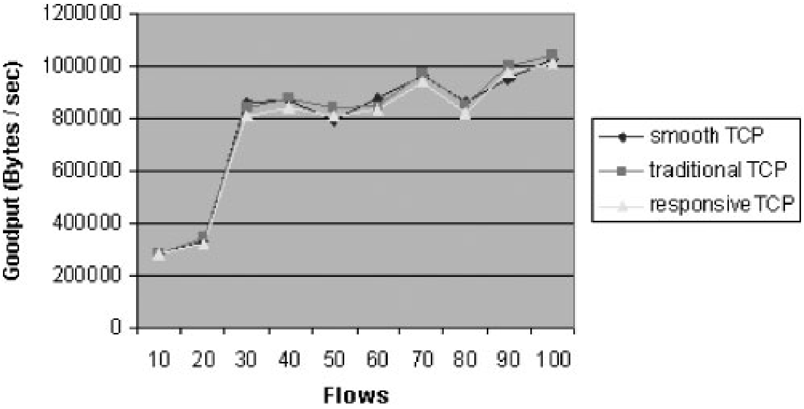


Figure 15. Goodput of TCP variations in heterogeneous scenario with graduated contention increase.

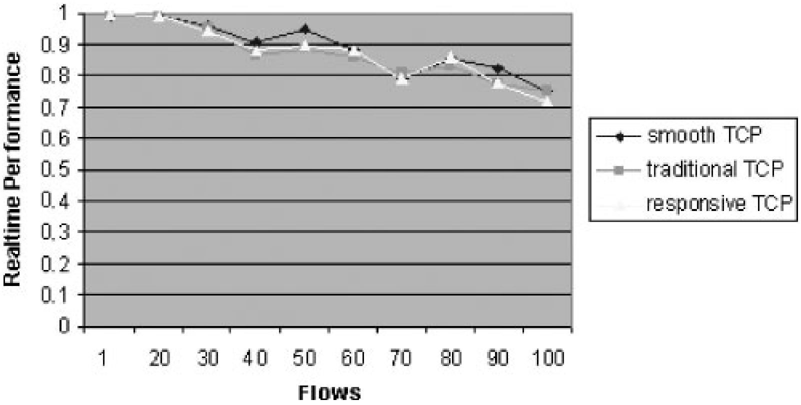


Figure 16. Real-time performance of TCP variations in heterogeneous scenario with graduated contention increase.

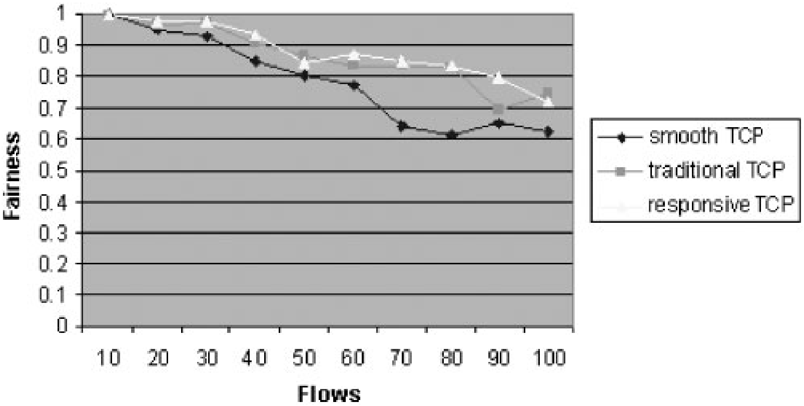


Figure 17. Fairness of TCP variations in a scenario with long handoffs.

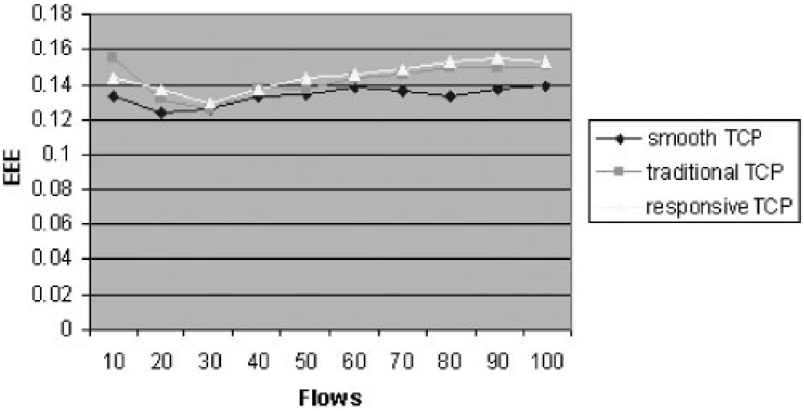


Figure 18. EEE of TCP variations in a scenario with long handoffs.

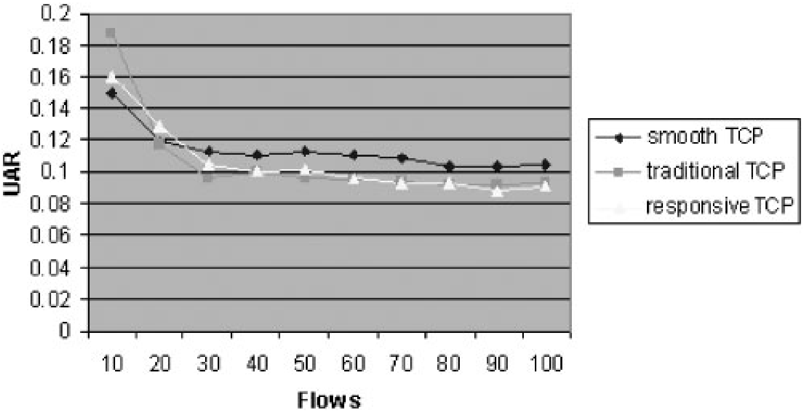


Figure 19. UAR of TCP variations in a scenario with long handoffs.

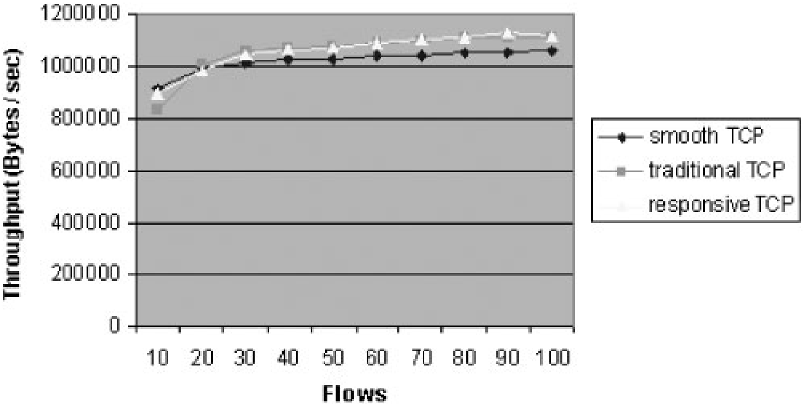


Figure 20. Throughput of TCP variations in a scenario with long handoffs.

DOI: 10.1002/dac

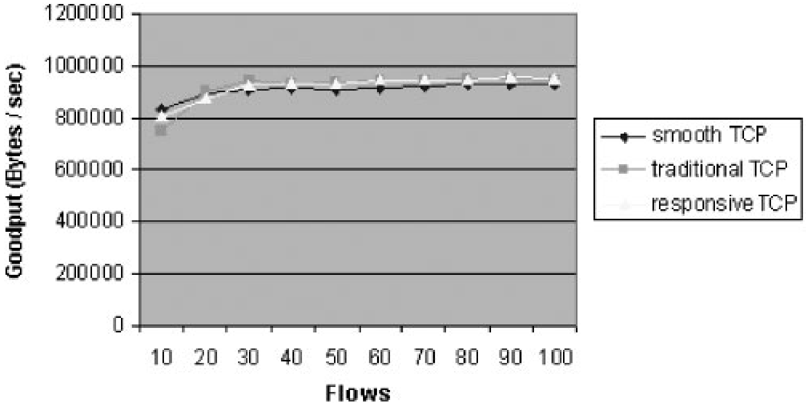


Figure 21. Goodput of TCP variations in a scenario with long handoffs.

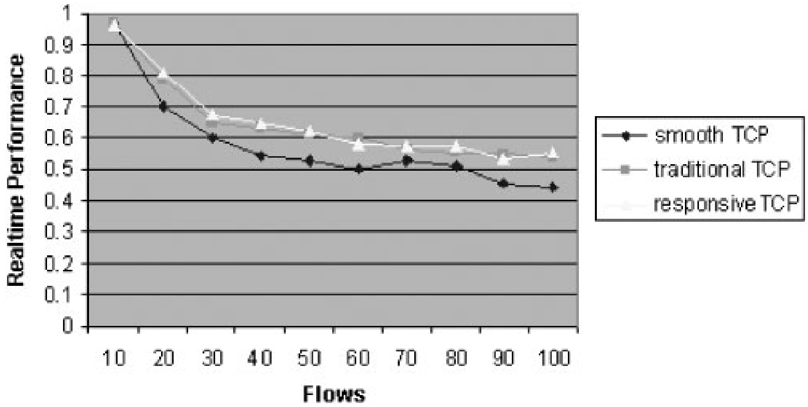


Figure 22. Real-time performance of TCP variations in a scenario with long handoffs.

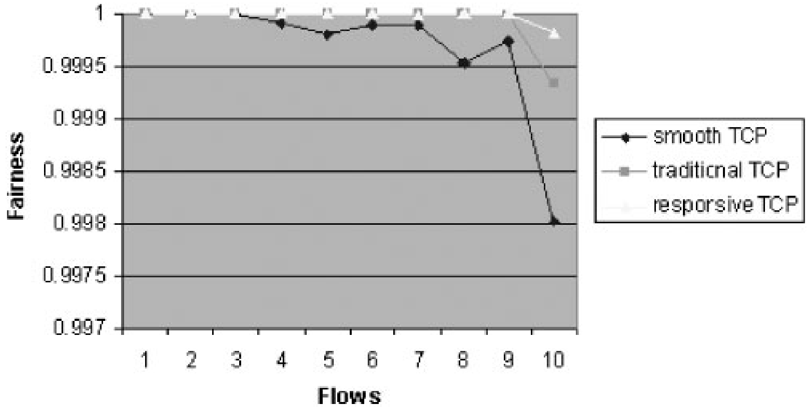


Figure 23. Fairness of TCP variations in a scenario with short handoffs.

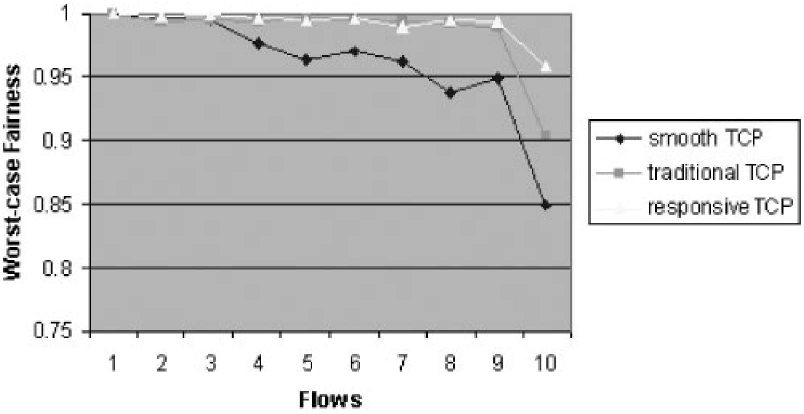


Figure 24. Worst-case fairness of TCP variations in a scenario with short handoffs.

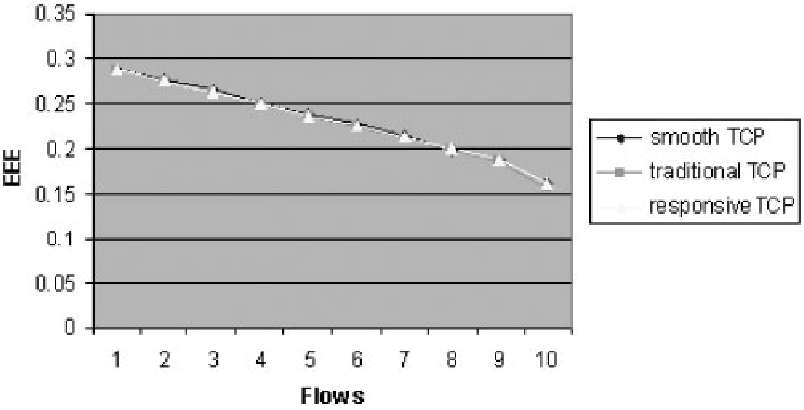


Figure 25. EEE of TCP variations in a scenario with short handoffs.

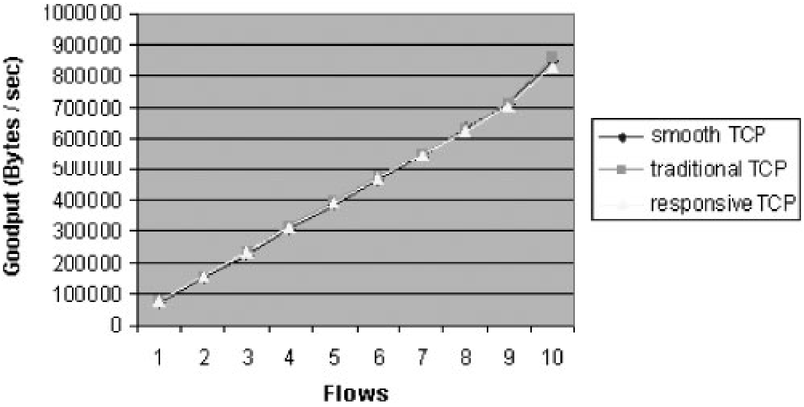


Figure 26. Goodput of TCP variations in a scenario with short handoffs.

DOI: 10.1002/dac

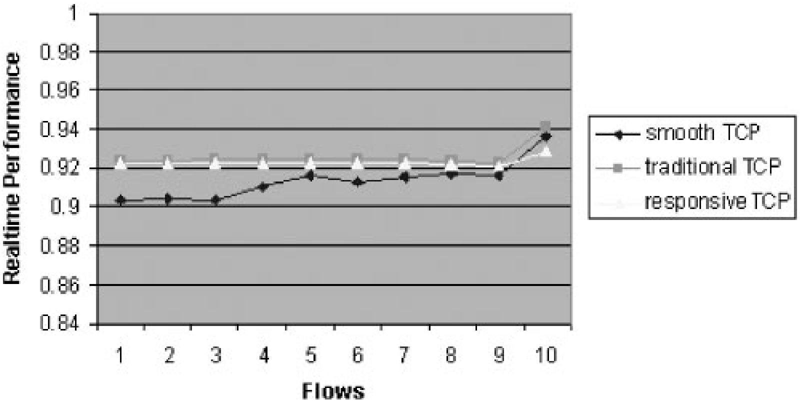


Figure 27. Real-time performance of TCP variations in a scenario with short handoffs.

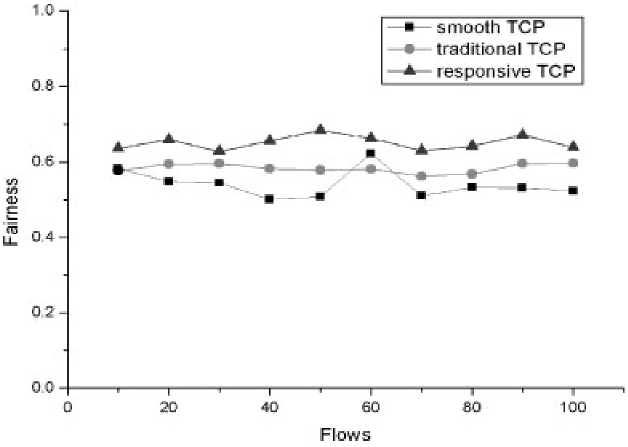


Figure 28. Fairness of TCP variations in a complex topology.

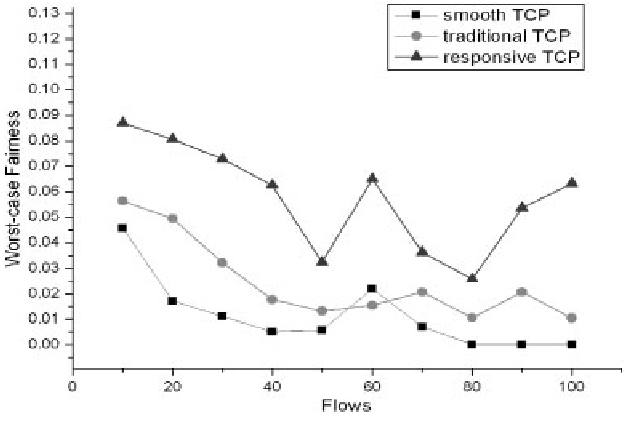


Figure 29. Worst-case fairness of TCP variations in a complex topology.

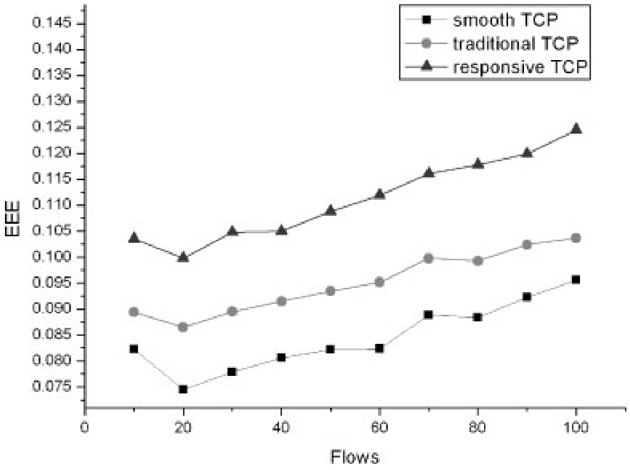


Figure 30. EEE of TCP variations in a complex topology.

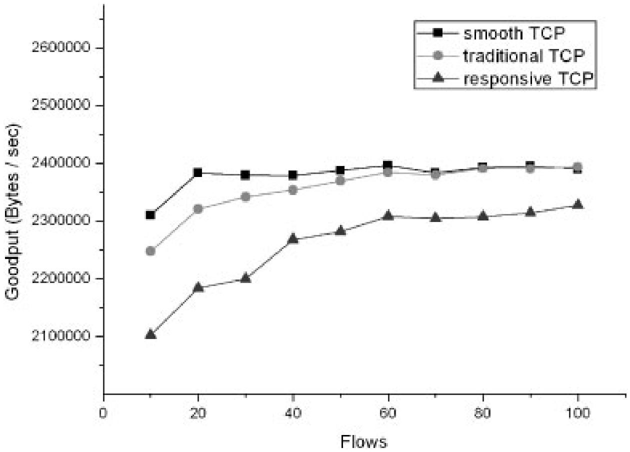


Figure 31. Goodput of TCP variations in a complex topology.

has not exploited the available resources efficiently. Therefore, a more sophisticated congestion control algorithm can improve its efficiency. The lower effort of the smooth TCP (Figure 18) does not impact Goodput significantly (Figure 21). However, smooth TCP is not suitable for real-time/multimedia applications because of its increased packet jitter (Figure 22).

5.5. *Wireless scenario with short handoffs*

In this scenario, we used handoffs with low duration (0.1 s). We measured performance, ranging the number of flows from 1 to 10. While all protocols exhibit similar behaviour in terms of Energy

DOI: 10.1002/dac

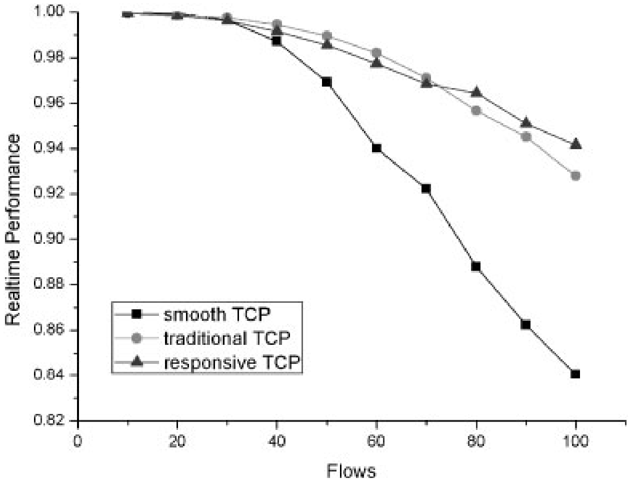


Figure 32. Real-time performance of TCP variations in a complex topology.

Efficiency and Goodput (Figures 25 and 26), smooth TCP is unfair (Figures 23 and 24) and achieves the worst Realtime Performance (Figure 27). This result calls for further investigation into whether a smooth behaviour is always more suitable for applications demanding low packet jitter. From the effort/gain perspective, which is our present focus, we note that different transport mechanisms may promote different targets. For example, although the smooth protocol is more suitable for applications that demand high Fairness (e.g. distributed applications), it is not appropriate for multimedia/real-time applications (Figure 27).

5.6. Scenario with a complex topology

In the last scenario, we used a complex topology with multiple bottlenecks and cross traffic. We ranged the number of flows from 10 to 100. While smooth TCP achieves the best performance in terms of Goodput (Figure 31—especially for less than 60 flows), its *Fairness* potential is degraded (Figures 28 and 29). Furthermore, smooth TCP appears energy efficient but inappropriate for real-time/multimedia applications due to very high packet jitter (Figure 32). We can see a clear tradeoff between Fairness-Realtime Performance (Figures 28, 29, 32) and Goodput-Energy Efficiency (Figures 30 and 31). Responsive TCP improves Fairness (Figures 28 and 29) and Realtime Performance (Figure 32) trading off performance in Goodput (31) and Energy Efficiency (Figure 30). Traditional TCP seems a good choice maintaining all performance aspects in acceptable levels.

6. CONCLUSIONS AND FUTURE WORK

We have shown that smooth/responsive strategies do not always correspond to conservative/aggressive behaviour, respectively, as it is traditionally believed. Based on a primary analysis, which was also confirmed experimentally, we have shown that different network conditions call

DOI: 10.1002/dac

for different smoothness/responsiveness tactics. Furthermore, smooth protocols are not always suitable for multimedia/real-time applications. For example, dynamic scenarios that require more frequent adjustments of the transmission strategy (such as scenarios with handoffs or dynamic contention level) require responsive protocols.

We have also shown that effort/gains dynamics are not straightforward and impact system performance in terms of Goodput, Fairness, Energy Efficiency and Realtime Performance. Based on the behavioural patterns we exploited here, we plan to work towards a measurement based algorithm (such as [9]) that monitors network condition and triggers appropriate responses.

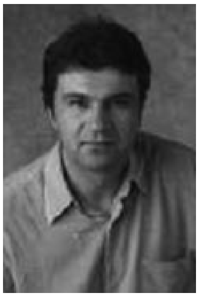
REFERENCES

1. Allman M, Paxson V, Stevens W. TCP congestion control. *RFC2581*, 1999.
2. Chiu D-M, Jain R. Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. *Computer Networks and ISDN Systems* 1989; **17**(1):1–14.
3. Floyd S. Congestion control principles. *RFC 2914*, September 2000.
4. Mamatas L, Harks T, Tsaoussidis V. Approaches to congestion control in packets networks. *Journal of Internet Engineering* (JIE) 2007; **1**(1):22–33.
5. Floyd S, Handley M, Padhye J. *A Comparison of Equation-based and AIMD Congestion Control*. May 2000. http://www.aciri.org/tfrc/
6. Handley M, Floyd S, Pahdye J, Widmer J. TCP friendly rate control (TFRC): protocol specification. *RFC 3448*, January 2003.
7. Brakmo L, Peterson L. TCP vegas: end to end congestion avoidance on a global Internet. *IEEE Journal on Selected Areas in Communication* 1995; **13**(8):1465–1480.
8. Casetti C, Gerla M, Mascolo S, Sansadidi MY, Wang R. TCP westwood: end-to-end congestion control for wired/wireless networks. *Wireless Networks Journal* 2002; **8**:467–479.
9. Tsaoussidis V, Zhang C. TCP-real: receiver-oriented congestion control. *The Journal of Computer Networks* (COMNET) 2002; **40**(4):477–497.
10. Attie PC, Lahanas A, Tsaoussidis V. Beyond AIMD: explicit fair-share calculation. *Proceedings of ISCC*, Kemer-Antalya, Turkey, June 2003.
11. Xu K, Tian Y, Ansari N. TCP-Jersey for wireless IP communications. *IEEE JSAC* 2004/05; **22**(4): 747–756.
12. Floyd S, Handley M, Padhye J, Widmer J. Equation-based congestion control for unicast applications. *Proceedings of ACM SIGCOMM 2000*, Stockholm, Sweden, August 2000.
13. Yang YR, Kim MS, Lam SS. Transient behaviors of TCP-friendly congestion control protocols. *Proceedings of IEEE INFOCOM 2001*, Anchorage, AK, U.S.A., April 2001.
14. Yang YR, Lam SS. General AIMD congestion control. *Proceedings of the 8th International Conference on Network Protocols*, Osaka, Japan, November 2000.
15. Tsaoussidis V, Matta I. Open issues on TCP for mobile computing. *Journal of Wireless Communications and Mobile Computing* 2002; **2**(2):3–20.
16. Tsaoussidis V, Badr H, Ge X, Pentikousis K. Energy/throughput tradeoffs of TCP error control strategies. *Fifth IEEE Symposium on Computers and Communications IEEE ISCC 2000*, Antibes-Juan les Pins, France, July 2000.
17. Padhye J, Firoiu V, Towsley D, Kurose J. Modeling TCP throughput: a simple model and its empirical validation. *ACM SIGCOMM 1998*, Vancouver, B.C., Canada, August 1998.
18. Mamatas L, Tsaoussidis V. Protocol behavior: more effort, more gains? *The 15th IEEE International Symposium on Personal*, *Indoor and Mobile Radio Communications*, Barcelona, Spain, September 2004.
19. Mamatas L, Tsaoussidis V. Transport protocol behavior and energy-saving potential. *Sixth International Workshop on Wireless Local Networks* (*WLN 2006*), Tampa, FL, 2006.

DOI: 10.1002/dac

AUTHORS' BIOGRAPHIES

Lefteris Mamatas is a PhD student in the Demokritos University of Thrace. He graduated in 2003 from the Department of Electrical and Computer Engineering, Demokritos University of Thrace. His research interests are focused on transport protocols over wired/wireless networks. He had published 4 journal and 12 conference papers.



Vassilis Tsaoussidis received a BSc in Applied Mathematics from Aristotle University, Greece; a Diploma in Statistics and Computer Science from the Hellenic Institute of Statistics; and a PhD in Computer Networks from Humboldt University, Berlin, Germany (1995). Vassilis held faculty positions in Rutgers University, New Brunswick, SUNY Stony Brook and Northeastern University, Boston. In May 2003, Vassilis joined the Department of Electrical and Computer Engineering of Demokritos University, Greece. His research interests lie in the area of transport/network protocols, i.e. their design aspects and performance evaluation. His COMNET (Computer Networks) Research Group includes seven PhD students.

Vassilis is an editor-in-chief for the *Journal of Internet Engineering* and an editor for *IEEE Transactions in Mobile Computing*, the *Journal of Computer Networks*, the *Journal of Wireless Communications and Mobile Computing*, and the *Journal of Mobile Multimedia*. He participates in several Technical Program Committees in his area of expertise, such as INFOCOM, GLOBECOM, ISCC, EWCN, LCN, NETWORKING, and several others. Vassilis graduated two PhD students in the United States: Adrian Lahanas, now a lecturer at the University of Cyprus; and Chi Zhang, now Assistant Professor at Florida International University.

DOI: 10.1002/dac