Protocol Behavior: More Effort, More Gains?

L. Mamatas and V. Tsaoussidis
Dept. Of Electrical and Computer Engineering
Demokritos University of Thrace, Greece
Email: emamatas@ee.duth.gr, vtsaousi@ee.duth.gr

Abstract

We investigate the behavior 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. Our perspective is novel indeed: 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 metrics to capture protocol behavior: 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 behavior, 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 behavior.

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 modeled on the Additive Increase/Multiplicative Decrease 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].

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 [4, 5, 10, 11] propose TCP-friendly protocols 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 favoring smoothness using a gentle backward adjustment upon congestion, at the cost of lesser responsiveness (i.e., 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) [8]. As we show, implementation of such strategy requires ocassionally 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 behavioral spectrum reflects the effort a protocol expends. The real issue, therefore, is how much this effort is invested into efficient transmission.

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, 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 [7] 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, ½); (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[1] 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 present our testing methodology and we define new performance metrics. In section 4 we analyze the results of our experiments and in section 5 we highlight our conclusions.

2. Trading α For β

A throughput equation for standard TCP is first introduced in [6]. GAIMD [10] extends the equation to include parameters α and β:

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

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 [10]) could provide a rough guide to achieve friendliness.

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

Authors of [10] derive from (1) and (2) a simple relationship for α and β:

$$\alpha=4(1-\beta^2)/3 \quad (3)$$

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* [4], which reflects the *uninterrupted growing lifetime of congestion 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

$$n = (1-\beta) * W / \alpha + 1 \ RTTs \quad (4)$$

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:

$$W = B * RTT / N \quad (5)$$

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 modeled 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 favorable 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-β) is applied only once. However, multiple packet drops could cause the window size to be decreased 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 2 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

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

responsive TCP may be a desirable behavior. We confirm our statements experimentally in section 4.

3. Experimental Methodology
3.1 Testing Plan

We have implemented our testing plan on the ns-2 network simulator. The network topology used as a test-bed is the typical single-bottleneck *dumbbell*, as shown in Figure. 1. The link's capacity (bw_bottleneck) is 10Mbps, unless it is explicitly stated otherwise. 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 packet error rate (PER). The number of flows occasionally changes for the different scenarios. The simulation time was fixed at 120 seconds, a time-period deemed appropriate to allow all protocols to demonstrate their potential.

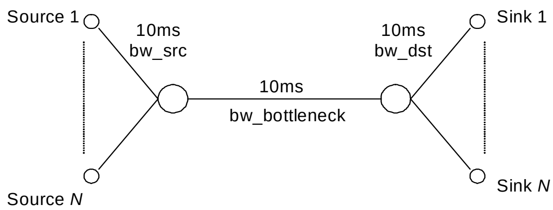

Figure 1. Network topology

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 equation [10]. 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 first scenarios, ftp flows are entering the system within the first seconds. All flows are fixed, during the rest 118 seconds. In order to evaluate how efficiently and fairly the protocols can exploit available bandwidth, we used, additionaly, scenarios with graduated contention decrease.

3.2 Performance Metrics

Our evaluation plan calls for common, as well as non-traditional metrics. We used traditional metrics for protocol efficiency, and fairness.

The system goodput is used to measure the overall system efficiency in bandwidth utilization. The system Goodput is defined as :

Goodput= Original_Data / 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 is the amount of time required for the data delivery.

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

$$Fairness = \frac{(\sum_{i=0}^{n} Throughput_i)^2}{n(\sum_{i=0}^{n} Throughput_i^2)}$$

where $Throughput_i$ is the Throughput of the i_{th} flow and n the flow number.

In order to capture the amount of *extra* energy expended, we introduce a new metric. Extra Energy Expenditure (3E) takes into account the difference of achieved Throughput from maximum Throughput ($Throughput_{max}$) for the given channel conditions, the difference of Goodput from Throughput, attempting to locate the Goodput as a point within a line that starts from 0 and ends at $Throughput_{max}$. The metric 3E takes values from 0 to 1, attempting to capture both distances.

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

where a=1 and b=0.3

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

We need to introduce another metric as well, in order for us to capture the level of *Unexploited Available Resources* (UAR). That is, how well did we exploit the windows of opportunities for successful transmissions. Reasonably, the case of Goodput=Throughput=0 should not give us at this point a minor (as with the 3E metric) but a major penalty.

$$UAR = 1 - [a\frac{Throughput}{Throughput_{max}} + b\frac{Goodput}{Throughput}]$$

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

4. Results and Discussion

4.1 Low error rate favors responsive protocols

The first scenario simulates a heterogeneous environment with random transient errors increasing from 0.01 to 0.1 PER. We used 30 flows and a 10Mbps bottleneck, a relatively low-contention environment. The following results show that the responsive protocol outperforms the smooth one. Also, its aggressive behavior favors both Extra Energy Expenditure (3E) and Unexploited Available Resources Index (UAR) :

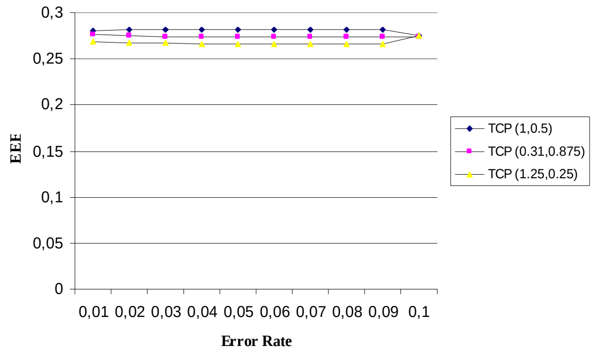

Figure 2. EEE & Low Error-Rate

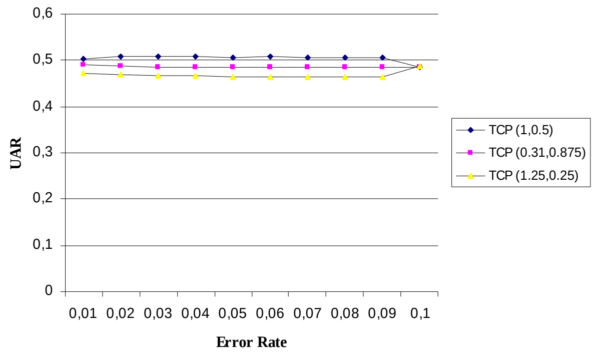

Figure 3. UAR & Low Error Rate

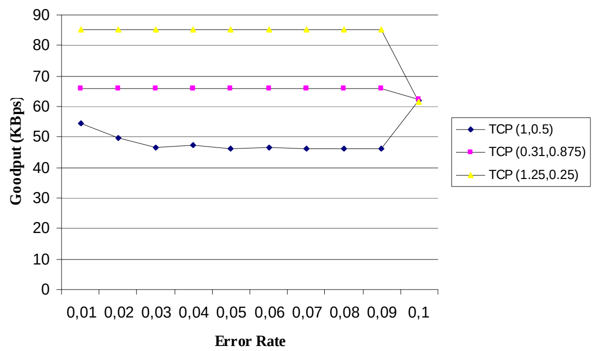

Figure 4. Goodput & Low Error Rate

4.2 A macroscopic view of the Effort/Gain dynamics

In the last scenario we used handoffs with duration 0.2 seconds in a 10Mbps bottleneck. We measured performance, ranging the number of flows from 10 to 100.

We can observe that, better resource and energy exploitation may have a positive impact on protocol goodput, although the reverse is also possible. See, for example the contrasting outcome with less and more effort, in figures 5,6,7 and 2,3,4, respectively.

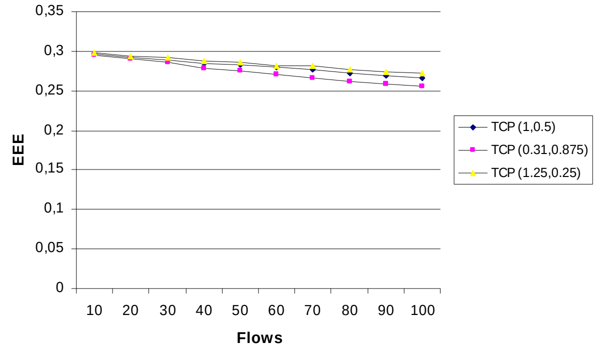

Figure 5. EEE & Effort/Gain dynamics

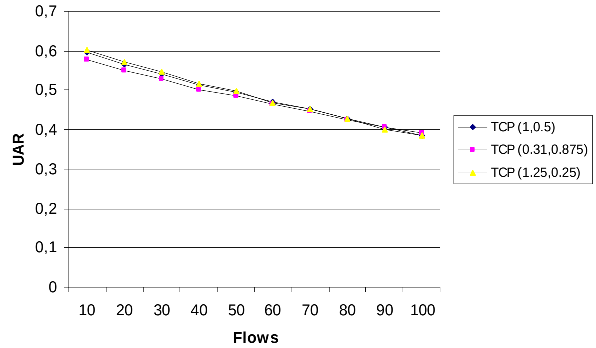

Figure 6. UAR & Effort/Gain dynamics

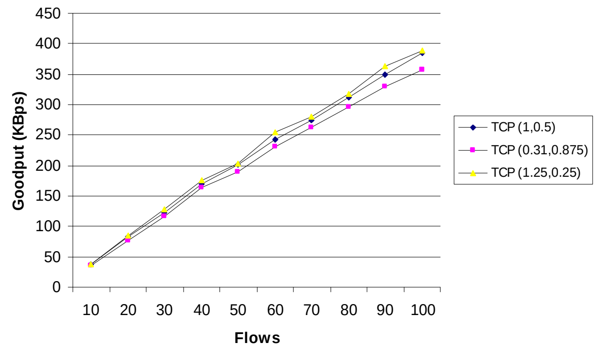

Figure 7. Goodput & Effort/Gain dynamics

4.3 Observations with contention decrease

The next scenario presented here intends to provide a framework for characterizing protocol behavior when bandwidth becomes available rapidly in heterogeneous networks. We measure Extra Energy Expenditure (3E), Unexploited Available Resources Index (UAR) and Goodput for a range of flows from 10 to 20. We used a 0.2 PER. All flows are entered in the system within the first two seconds. For the rest 112 seconds we have a graduated contention decrease, starting from 10 flows and repeating the experiment for 11, 12 upto 20 flows. At each stage we reduce the number of flows to half every Decrease_Step seconds, where Decrease_Step, is the step needed, in order for the last flow to exit at the 120th second.

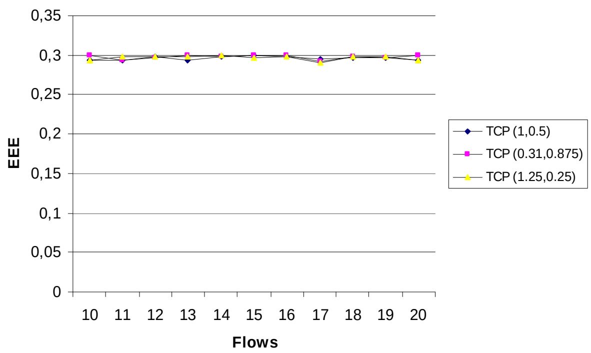

Figure 8. EEE & Contention Decrease

Although, according to the 3E metric, protocol behavior appears stable, the UAR index indicates that available resources are not exploited very well by the smooth protocol.

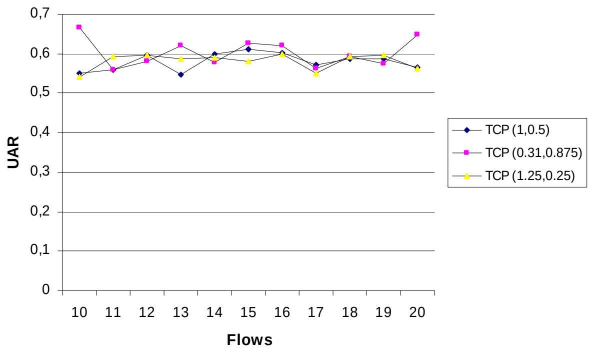

Figure 9. UAR & Contention Decrease

4.4 Error rate increase cancels responsive TCP's advantages

In the following scenario, we used 30 flows, a 10Mbps bottleneck and a variable error-rate from 0.01 to 0.4 PER. During low error rate the responsive protocol has better return for its effort, however, when error-rate exceeds 0.1, these advantages are canceled (see figures 10, 11, 12).

We summarize below the difference in Fairness, Extra Energy Expenditure, Unexploited Available Resources Index and Goodput.

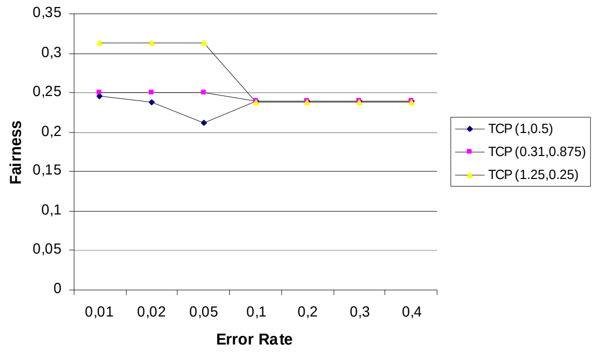

Figure 10. Fairness & Error-Rate

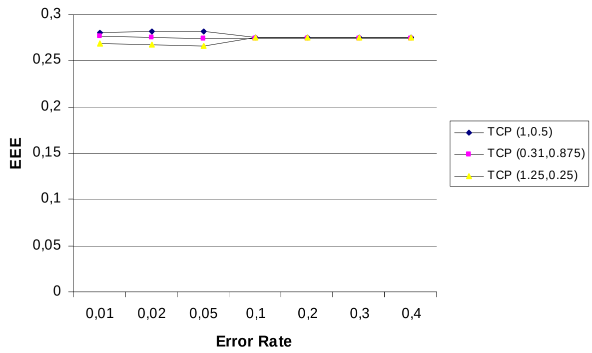

Figure 11. EEE & Error-Rate

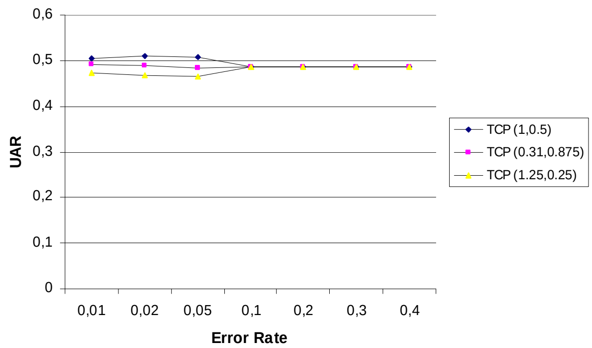

Figure 12. UAR & Error-Rate

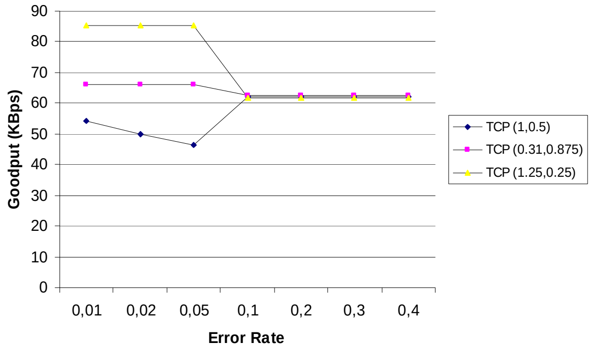

Figure 13. Goodput & Error-Rate

We can see that the responsive protocol is favored, initially. After a certain point, which is relevant to the specifics of the experiment (which in our case is 0.1), the smooth protocol may even become more efficient (in goodput) and fair, while it expends less extra energy. When the fair-share grows, due to higher bandwidth (100Mbps), the previous behavior is indicated more clearly (see figures 14, 15).

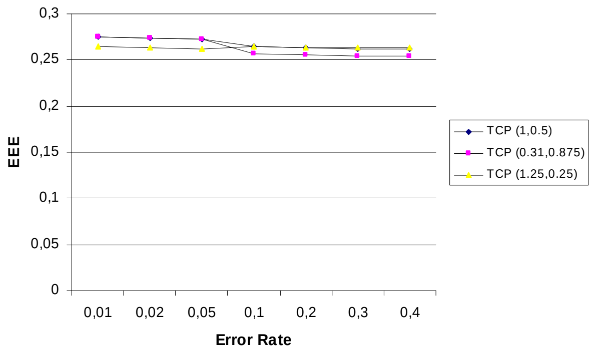

Figure 14. EEE & Error-Rate

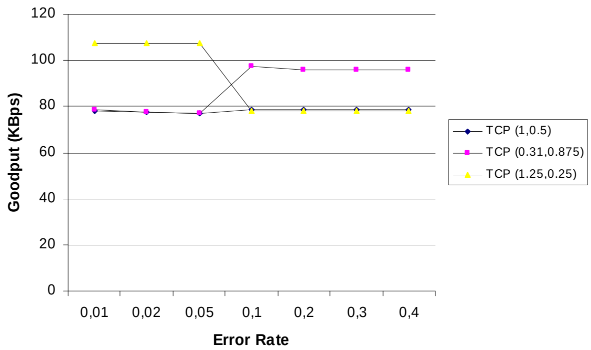

Figure 15. Goodput & Error-Rate

5. Conclusions and future work

We have shown that smooth/responsive protocols do not always have a conservative/aggressive behavior respectively, as it is was traditionally believed. We have predicted through a basic analysis and confirmed experimentally a better behavior of conservative protocols in a high error-rate environment, in contrast to the aggressive ones. In case of low error-rates and sufficient availability of bandwidth, the situation is reversed.

If TCP's traffic could be shaped to conform to detected network characteristics, system peformance metrics such as goodput, fairness, energy expenditure and resource utilization would be handled better.

Initially, we plan to work towards a measurement based detection of network characteristics, such as the one presented in [9]. Departing from there, we plan to apply error recovery tactics which integrate the adaptive strategy, in accordance with the results shown here.

6. References

[1] M. Allman, V. Paxson, and W. Stevens, "TCP Congestion Control", RFC2581, April 1999.

[2] D.-M. Chiu and R. Jain, "Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks", Computer Networks and ISDN Systems, 17(1):1-14, 1989.

[3] S. Floyd, "Congestion Control Principles", RFC 2914, September 2000.

[4] S. Floyd, M. Handley and J. Padhye, "A Comparison of Equation-based and AIMD Congestion Control", May 2000. URL: http://www.aciri.org/tfrc/.

[5] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-Based Congestion Control for Unicast Applications", Proceedings of ACM SIGCOMM 2000, August 2000.

[6] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP Throughput: A Simple Model and its Empirical Validation", ACM SIGCOMM 1998, August 1998.

[7] V. Tsaoussidis, H. Badr, X. Ge, K. Pentikousis, "Energy/Throughput Tradeoffs of TCP Error Control Strategies", *5th IEEE Symposium on Computers and Communications* IEEE ISCC 2000, July 2000.

[8] V. Tsaoussidis and I. Matta, "Open issues on TCP for Mobile Computing", Journal of Wireless Communications and Mobile Computing, Wiley Academic Publishers, Issue 2, Vol. 2, February 2002.

[9] V. Tsaoussidis and C. Zhang "TCP-Real: Receiver-oriented Congestion Control", The Journal of Computer Networks COMNET, Elsevier Science pp 477-497, Volume 40, Issue 4, November 2002.

[10] Y.R. Yang and S.S. Lam, “General AIMD Congestion Control”, Proceedings of the 8th International Conference on Network Protocols”, Osaka, Japan, November 2000.

[11] Y.R. Yang, M.S. Kim and S.S. Lam, “Transient Behaviors of TCP-friendly Congestion Control Protocols”, Proceedings of IEEE INFOCOM 2001, April 2001.