On the Properties of Adaptive Additive Increase

Christos V. Samaras, Vassilis Tsaoussidis
Department of Electrical and Computer Engineering
Democritus University of Thrace, Greece
{csamaras, vtsaousi}@ee.duth.gr

Abstract

We discuss the impact of fixed window increase rules on network utilization, fair resource allocation and overall transmission effort. We exploit the dynamics of adaptive additive increase as the strategy of choice for regulating transmission in accordance with the current level of contention. We show that, based on the additive increase rate of flows, a contention-oriented responsive strategy can be designed to (i) increase utilization under low-contention, (ii) reduce overhead and transmission effort under heavy loads, and (iii) maintain system capability to reach fairness.

Key Words- Network Contention, TCP, AIMD, Adaptive Transmission, TCP Enhancement

1. Introduction

We examine TCP's additive increase impact on end-system/network performance. We depart from the idea that system-wide rate increase, given a fixed *increase parameter a*, depends strictly on network contention. In turn, frequency of congestion events, persistence of congestion, and system stability, largely depend on contention also. Retransmission overhead, goodput to throughput proportion, and possibly fairness and smoothness, may be affected too.

In this context, we seek a way to increase stability, reduce overhead, and adjust system behavior to current conditions of contention. One way to achieve this is by adapting additive increase in accordance with current contention. Ultimately, we adjust *parameter a* in a manner that the collective rate increase for the participating flows remains fixed at a target value. In simple terms, contention fluctuation does not translate into aggregated rate increase/decrease. However, this target alone cannot characterize an efficient system. That is, we also need to satisfy concurrently the condition of fairness; and judge system behavior not only in terms of achieved throughput gains, but also in terms of invested effort. The reasoning behind our evaluation strategy is mainly rooted on the strategic impact of *parameter a* on system behavior. The increase rule determines the capability of a flow to exploit available resources; the pace of congestion events and hence the convergence speed to equilibrium; the amount of retransmitted packets and hence the wasted effort of a flow. In this context, we target a system, which satisfies three main objectives: (i) high utilization, (ii) fairness, and (iii) minimal overhead. We claim that a system can have these properties *iff* it exhibits a contention-oriented transmitting behavior.

In order to design a contention-oriented increase/decrease transmission scheme, we need to:

- be able to estimate contention, and
- exploit the corresponding relation of additive-increase-value & contention.

Contention estimation is a research subject in its own right; we do not address presently this issue but rely on the fact that related studies exist with relatively satisfying results (e.g. [6], [7]). That said, we use predetermined scenarios where contention fluctuation is known *a priori* and we focus mainly on the corresponding relation between contention and additive increase rules.

In order to investigate the need for (and the dynamics of) adaptive additive increase, we follow three evaluation stages. At the first stage, we use a fixed $\alpha = 1$, as it is used in standard protocols, for varying levels of contention and we criticize that selection of $\alpha = 1$. At the second stage, we investigate other values of *parameter a* with varying levels of contention; we observe that the rate of increase depends strictly on contention. We further observe that the increase rate should grow in reverse proportion to contention. In particular, through simulations, we demonstrate that a relatively small value for *increase parameter a* contributes to better system performance when contention is high. Moreover, a relatively large additive increase value favors performance in low contention scenarios. At the third stage we investigate the corresponding relation between aggressiveness and contention. A constant additive increase value does not constitute an optimal choice because, as we show in our experiments, it does not keep pace with contention variation. We present good indications that adaptive additive increase for TCP effectively enhances system performance. Departing from there, we propose Adaptive Additive Increase Multiplicative Decrease (A-AIMD).

By and large, we show that with A-AIMD system utilization is increased; less packet retransmissions occur during congestion; bandwidth is better exploited during low contention; and fairness is either enhanced or maintained at acceptable levels. The proposed mechanism has the potential for deployment indeed: no change in network functionality is required; and only a slight modification in TCP sending side is needed.

The paper is organized as follows. In Section 2, we provide related work. Next, in Section 3, we elaborate on the significance of an adaptive, contention-oriented transmission strategy. In Section 4, we present our evaluation methodology, and in Section 5, we discuss simulation results. Finally, we conclude the paper in Section 6.

2. Related Work

Related work was focused mainly on two objectives. The first was to exploit the adjustment strategy of *parameter a*, while the second was to exploit the impact of contention on selected network or protocol control strategies.

The first objective has been extensively targeted during the past years and various congestion control mechanisms have been proposed and evaluated. Relevant studies include: standard TCP versions based on AIMD algorithm such as TCP New-Reno [2] and TCP SACK [3]; General AIMD congestion control [4], a generalized version of AIMD algorithm that parameterizes the *additive increase value a* and the *multiplicative decrease ratio β*; and TFRC (TCP-Friendly Rate Control) [5], which is a mechanism for equation-based congestion control, where the sender explicitly adjusts its sending rate as a function of the packet loss rate reported periodically by the receiver. For a comprehensive list of congestion control approaches, see [10]. In conclusion, increase strategies have been studied mainly as a necessary consequence of regulating parameter β, not as a strategic goal itself that impacts contention-oriented behavior. Here, we highlight the role of increase strategy alone, when contention varies.

The second objective was the target of significant previous research on various congestion control schemes, but no relation was established between congestion control and network contention. A contention-oriented strategy dominated the retransmission scheduling in the recent work presented in [6] and [7]. Also, authors in [8] observe, at a very preliminary level, that information on network contention should be retrieved and exploited for Active Queue Management as well. They propose an adaptive model for tuning RED/ECN algorithm based on contention level. Again, contention-oriented strategies for setting the transport protocol increase policy dynamically have not been established.

3. Adaptive Additive Increase

We demonstrate that an enhanced transport protocol capable of sensing network contention can improve system performance without sacrificing network stability. More specifically, we modify TCP sender so that it reacts more aggressively when contention is relatively low and more conservatively when contention reaches higher levels.

Applying the network model presented in [1], we consider *n* users sharing a bottleneck link. Users adjust their load (namely, their congestion window) according to the binary feedback received by the network. The network operates in discrete time slots, thus forming a synchronous feedback and control loop. Each time slot represents an interval at the beginning of which users set their load level based on the network feedback received at the previous interval. When feedback is 1, users increase their congestion window; when feedback is 0, users decrease their congestion window and consequently, their sending rate.

If during time slot *t*, user's *i* load is $x_i(t)$ then the total load at the bottleneck resource is the sum of all users' load, $\Sigma x_i(t)$. When a positive network feedback is received, the sender increases its window by *a* (i.e., additive increase = α), and at the following time slot $t + 1$ his load becomes:

$x_i(t + 1) = a + x_i(t),$ *where a is the additive increase value.*

At time $t + 1$, total load at the bottleneck resource becomes:

$$\sum_{i=1}^{n} x_i(t+1) = \sum_{i=1}^{n} (a + x_i(t)) = n \cdot a + \sum_{i=1}^{n} x_i(t).$$

Similarly, after two time slots ($t + 2$), total load becomes:

$$\sum_{i=1}^{n} x_i(t+2) = \sum_{i=1}^{n} (a + x_i(t+1)) = n \cdot a + \sum_{i=1}^{n} x_i(t+1) =$$

$$= n \cdot a + \sum_{i=1}^{n} (a + x_i(t)) = 2 \cdot n \cdot a + \sum_{i=1}^{n} x_i(t).$$

Consequently, after *k* time slots and without congestion, total network load becomes:

$$\sum_{i=1}^{n} x_i(t+k) = k \cdot n \cdot a + \sum_{i=1}^{n} x_i(t) = A + B \qquad (1)$$

$$\text{where } A = k \cdot n \cdot a, \text{ and } B = \sum_{i=1}^{n} x_i(t).$$

Equation (1) reveals that network load dynamics (and network resources utilization, by extension) is determined by two terms: term *B* that characterizes the *initial* state of the network system (at time *t*), and term *A* that essentially forms the driving force for changing current network state. Notably, term *A* is based on:

- *contention level*, which is expressed by the number of network flows (coupled with their current load), and
- senders' *aggressiveness*, controlled by additive increase specific value.

The efficiency of network usage is defined by the proximity of the total load to a desired level:

$$\sum x_i(t) = X_{goal}$$

where X_{goal} denotes the desired network load level and can be equated with bottleneck capacity, in the context of our analysis. Network is operating efficiently as long as total allocation $X(t)=\Sigma x_i(t)$ is close to X_{goal}. Overload ($X(t)>\Sigma x_i(t)$) or underload ($X(t)<\Sigma x_i(t)$) are both undesirable and are considered inefficient.

When current network load exceeds bottleneck capacity, congestion is encountered and network state is adjusted accordingly. Senders' reaction to congestion will eventually decrease network load and free up bandwidth resources. The capability of network flows to efficiently exploit available resources is therefore dictated by term A in (1): current number of users and additive increase parameter jointly affect network performance, smoothness, and responsiveness.

Given some initial network state (B in (1)), the time required for the network to approach the desired load level X_{goal} – and subsequently reach congestion – is determined by term A. So, it is:

$$A_{max} + B = X_{goal} \Leftrightarrow A_{max} = X_{goal} - B$$

$$where \quad A_{max} = k_{\max} \cdot n \cdot a \qquad (2)$$

In essence, k_{max} reflects frequency of reaching and exceeding X_{goal}, thus resulting in congestion. Therefore, k_{max} mirrors frequency of congestion. For a given initial network state, term A_{max} is a constant quantity. Additionally, (2) shows that: for a fixed *increase parameter a*, when contention increases (namely, higher n), the required time k_{max} up to congestion decreases; congestion events appear more frequently; and network system becomes unstable because frequent packet drops result in high system overhead. Improving network efficiency requires preserving stability and maintaining system overhead at acceptable levels. As seen in (2), in order to regulate frequency of congestion, *additive increase value a* should be adjusted in accordance with contention level n.

4. Evaluation Methodology

Our simulations use the packet-level simulator ns-2 [9]. We evaluate the performance of adaptive additive increase in a wide range of constant and variable contention scenarios. Simulations are conducted on a single-bottleneck dumbbell topology (Fig. 1) with bottleneck capacity of 10Mbps. Propagation delay is 2ms for the edge links and 30ms for the bottleneck link, thus producing a round-trip link delay of 68ms. Senders are neither limited by their access link capacity (which is set to bottleneck capacity, namely 10Mbps) nor by the maximum congestion window value allowed (which is set to a relatively high value that was never exceeded during simulation runs). Thus, congestion on the bottleneck link is the only limiting factor for sending rates. In other words, a single flow can potentially use the overall bottleneck capacity in the absence of competing flows. The data packet size is 1000 bytes. The flows RTTs are equivalent and the sources are long-lived FTP connections. Influence and performance of short-lived connections, which comprise a significant part of current Internet traffic, are left for future work. In this work, our aim is to exploit the benefits of adaptive transmission control against standard TCP constant sending behavior, and not the interaction between transport and application protocols or queuing disciplines. For the same reason, drop-tail is the only dropping policy used in our simulations. The bottleneck buffer size is adjusted to the *Bandwidth-Delay product*. Each simulation lasts for 120 seconds.

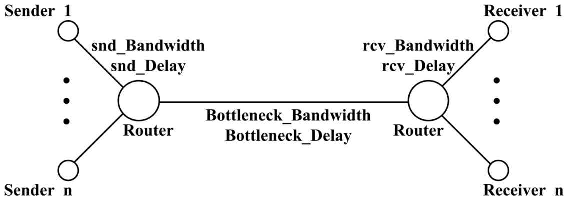

Figure 1. Dumbbell topology

We evaluate system performance based on four distinct metrics, which we define below. (In the following metrics, n denotes the number of flows.)

(i). Flow *Throughput* is measured at the sending side, includes packet headers and retransmitted packets, and illustrates the total transmission effort of the sender including the retransmission effort. *System Throughput* is the sum of throughput of all flows and *reflects* system utilization.

$$Throughput = \frac{Total\ Data\ Sent}{Transmission\ Time}$$

$$System\ Throughput = \sum_{i=1}^{n} Throughput_i$$

Note that system throughput and utilization are not identical since system utilization cannot exceed network capacity. Instead, throughput measured at the sender may do. Therefore:

$$System\ Throughput = System\ Utilization,$$

$$when \ System\ Throughput \leq Network\ Capacity.$$

(ii). *Goodput* of a flow is calculated at the receiver and refers to the original data received, i.e. excluding packet headers overhead and retransmitted packets. *System Goodput* is the sum of goodput of all flows and is used to measure the overall system efficiency in bandwidth utilization.

$$Goodput = \frac{Original\ Data\ Received}{Transmission\ Time}$$

$$System\ Goodput = \sum_{i=1}^{n} Goodput_i$$

Hence, an increase in the difference between system throughput and goodput indicates an increase in the number of retransmitted packets and implies wasted bandwidth but also wasted effort, which occasionally can be translated as energy cost, as well.

(iii). *Wasted Transmission Effort* is also measured (see [11] for more details); given the fixed packet size in our simulations, wasted effort can be calculated as the percentage of retransmitted packets for all flows:

$$Wasted\ Effort = \frac{Retransmitted\ Packets}{Total\ Packets\ Sent}$$

(iv). Fairness index is defined in [1] as:

$$Fairness = \frac{\left(\sum_{i=1}^{n} Throughput_i\right)^2}{n\sum_{i=1}^{n} (Throughput_i)^2}.$$

5. Results and Discussion

Next, we demonstrate conclusive performance results for adaptive additive increase. It is shown that an adaptive *increase parameter a* outperforms standard TCP for which α = 1. Adaptation strategy is based on the following simple concept:

- when contention is relatively low, raise *parameter a*;
- should contention reach relatively high levels, reduce *parameter a*.

In this work, we do not propose any sophisticated mechanism for dynamic TCP additive increase. However, a recent work of ours that contributes to contention estimation can be viewed at [6].

In our simulations, additive increase adaptivity is not incorporated as a responsive procedure; instead, at this stage of our work, it is predetermined. In constant contention scenarios, we evaluate a wide range of fixed values for *parameter a*. However, in variable contention scenarios, each simulation comprises a distinct number of constant contention periods. Thus, additive increase value remains constant for a certain contention period, and is modified accordingly for each of the consecutive contention periods. A sufficiently large number of increase-value/contention-level pairs are evaluated, in order to characterize comparatively system performance and observe system dynamics.

5.1. Impact of Fixed-Rate Additive Increase when Contention is Constant

Using the topology in Fig. 1, additive increase impact is evaluated in constant contention scenarios. More specifically, number of flows remains constant and takes values in the range 10-160 across different simulation runs. Also, additive increase value is fixed for a single simulation and its value varies between 0.1 and 2.0 from one simulation to another. Apart from that, rest of TCP functionality is kept unchanged. Different combinations of additive-increase/contention-level reveal that standard window growth function of TCP in congestion avoidance (namely, α = 1) is not always an appropriate choice (Figures 2-4).

Based on Fig. 2, one may fall into a false conclusion about the dominance of α = 0.1. Although goodput-wise value 0.1 appears as the additive increase value of choice, Figures 2 and 4 reveal that certain trade-offs need careful consideration. For example, α = 0.1 achieves lower goodput (Fig. 2) when number of flows is less than 50. Also, we see that for up to 70 flows, α = 0.1 corresponds to worse fairness (Fig. 4).

Increase in contention induces deviation of goodput curves (Fig. 2). This underlines the importance of adaptive additive increase: in intense contention conditions, a lower value for *parameter a* results in remarkably better performance.

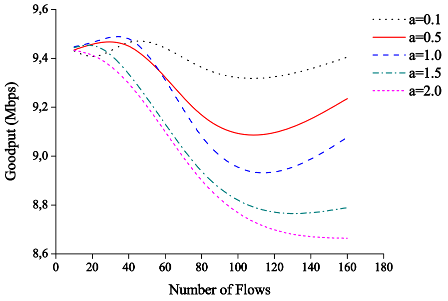

Figure 2. System goodput (Mbps)

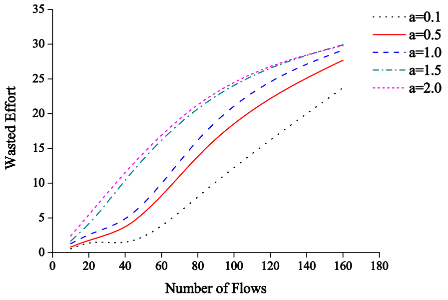

Figure 3. Wasted effort (%)

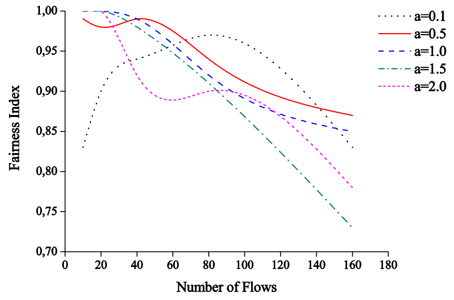

Figure 4. Fairness index

5.2. Impact of Adaptive Additive Increase when Contention Increases

We evaluate the impact of adaptive additive increase in contention increase scenarios. Each simulation lasts for 120 seconds and comprises four constant contention periods. More specifically, the initial number of flows is 10, and it is doubled every 30 seconds. Thus, at simulation time 0, 30, 60, and 90 seconds, the total number of flows is 10, 20, 40, and 80 respectively. We note that we have tried a wide range for number of flows, from 5 through 160 flows, with similar results.

However, *increase parameter a* is adapted in a predetermined way. In our simulations, TCP modifies additive increase value each time a new contention period begins. Thus, at simulation time 0, 30, 60, and 90 seconds, each TCP flow adjusts its additive increase value to *a1*, *a2*, *a3*, and *a4* respectively. A combination of increase parameter values during a simulation forms an adaptive TCP version (see Table 1), which decreases *parameter a* when contention increases.

We use NewReno as reference protocol and all the graphs in Figures 5, 6, and 7 refer to the *comparative* performance of *adaptive additive increase* and NewReno. From the three relevant graphs, we observe that various adaptive TCP versions (e.g. A, B, D, and E) outperform in all aspects the standard TCP version. We also observe that the trade-off between goodput and fairness still exists. This is highlighted in point C where goodput and retransmissions exhibit higher performance, however fairness is degraded. In conclusion, if we judge on the basis of system capacity to reach equilibrium with high utilization, then point A demonstrates a clear-cut advantage. Note that point A traverses a spectrum of very aggressive to very conservative increase rates. Instead, point C reaches too conservative rates, which reflect a more-or-less stable utilization at high levels (Fig. 5), minor retransmissions (Fig. 6) but significantly less fairness (Fig. 7). The latter was expected since the flows rarely had the opportunity to adjust the windows backwards and reduce their size gap. Note, however, that even an increase step of 0.05, when contention is high, may suffice to maintain fairness at the same level with NewReno (see point B in Fig. 7).

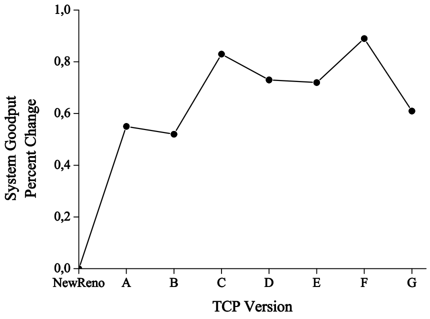

Figure 5. System goodput percent change

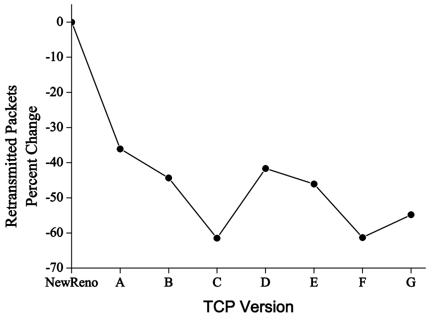

Figure 6. Retransmitted packets percent change

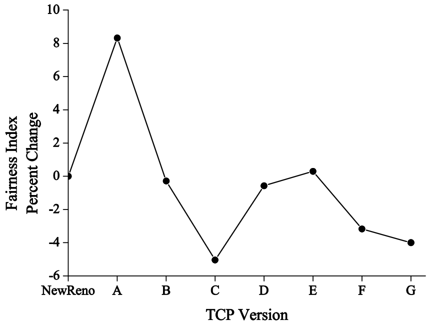

Figure 7. Fairness index percent change

Table 1. Adaptive TCP versions

TCP Version	Increase Parameter Value			
	a_1	a_2	a_3	a_4
A	2.0	1.0	0.5	0.1
B	2.0	1.0	0.25	0.05
C	2.0	1.0	0.1	0.01
D	1.5	1.0	0.5	0.1
E	1.0	0.7	0.4	0.1
F	1.0	0.5	0.1	0.05
G	0.5	0.4	0.2	0.1

5.3. Impact of Adaptive Additive Increase when Contention Decreases

We set up contention increase and decrease scenarios in a similar way. In the contention decrease scenario, the initial number of flows is 80, and it is halved every 30 seconds. We evaluate a number of adaptive TCP versions, which are shown in Table 2.

In our last scenario, resource supply exceeds the demand and hence we investigate the potential to exploit available bandwidth. In this context, we do not evaluate the fairness potential of the selected increase strategies.

Point J (which corresponds to point C of Table 1 but in reverse order) demonstrates clear-cut advantage here (see Figures 8 and 9). Although this may appear initially as a contradictory result, it is reasonable indeed. Since fairness is not an issue here, a wide range of values for *parameter a* results in better goodput with less retransmissions.

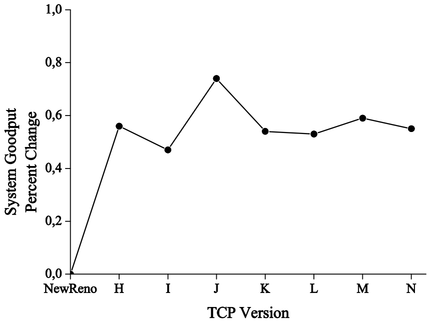

Figure 8. System goodput percent change

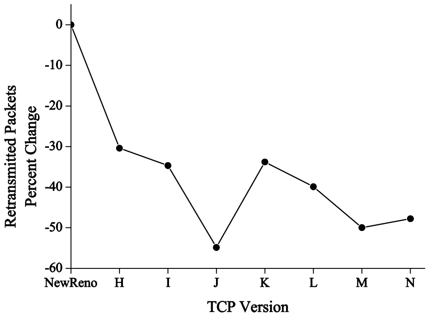

Figure 9. Retransmitted packets percent change

Table 2. Adaptive TCP versions

TCP Version	Increase Parameter Value			
	a_1	a_2	a_3	a_4
H	0.1	0.5	1.0	2.0
I	0.05	0.25	1.0	2.0
J	0.01	0.1	1.0	2.0
K	0.1	0.5	1.0	1.5
L	0.1	0.4	0.7	1.0
M	0.05	0.1	0.5	1.0
N	0.1	0.2	0.4	0.5

6. Conclusions

We studied the potential of adaptive increase strategies for transport protocols. This can be summarized in three major conclusions:

1) Adaptive strategy should correspond to contention level.

2) Adaptive additive increase has the potential to (i) cancel the possibility of congestion collapses, (ii) improve the effort/gain dynamics of protocol behavior, and (iii) maintain or improve the efficiency/fairness tradeoff among flows.

3) Adaptive additive increase strategy relies on the capability to monitor contention and its success depends on the accuracy and granularity of contention estimation strategies.

Our results justify conclusions 1 and 2. Further study is required for exploiting the possibility and impact of false estimations.

7. References

[1] D. Chiu and R. Jain, “Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks,” Journal of Computer Networks and ISDN Systems, vol. 17, no. 1, pp. 1-14, June 1989.

[2] S. Floyd, T. Henderson, and A. Gurtov, “The NewReno Modification to TCP's Fast Recovery Algorithm,” RFC 3782, April 2004.

[3] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, “TCP Selective Acknowledgement Options,” RFC 2018, October 1996.

[4] Y. R. Yang and S. S. Lam, “General AIMD Congestion Control,” in Proceedings of IEEE International Conference on Network Protocols 2000, Osaka, Japan, November 2000.

[5] S. Floyd, M. Handley, J. Padhye, and J. Widmer, “Equation-Based Congestion Control for Unicast Applications,” in Proceedings of ACM SIGCOMM 2000, Stockholm, Sweden, September 2000.

[6] Ioannis Psaras, Vassilis Tsaoussidis, and Lefteris Mamatas, “CA-RTO: A Contention Adaptive Retransmission Timeout,” in Proceedings of ICCCN 2005, San Diego, California, U.S.A., October 2005.

[7] Ioannis Psaras and Vassilis Tsaoussidis, “Why TCP timers (still) don't work well,” The International Journal of Computer and Telecommunications Networking Computer Networks and ISDN Systems, 2007.

[8] Li Lei, Pan Yong, and Shi Hongbao, “An Adaptive Model of RED/ECN Parameters,” in Proceedings of WCC-ICCT 2000.

[9] The Network Simulator, ns-2, http://www.isi.edu/nsnam/ns/.

[10] L. Mamatas, T. Harks, V. Tsaoussidis, “Approaches to Congestion Control in Packet Networks,” Journal of Internet Engineering, vol. 1, issue 1, Kleidarithmos Press, 2007.

[11] L. Mamatas, V. Tsaoussidis, and C. Zhang, “Protocol Behavior: More Effort, More Gains?”, in Proceedings of IEEE PIMRC 2004, Barcelona, Spain, September 2004.