Regular Paper

Congestion Control Using Multilevel Explicit Congestion Notification

Arjan Durresi,[†1] Leonard Barolli,[†2] Raj Jain[†3]
and Makoto Takizawa[†4]

Congestion remains one of the main obstacles to the Quality of Service (QoS) on the Internet. We think that a good solution to Internet congestion should optimally combine congestion signaling from network and source reaction, with the following as its main goals: minimum losses and delays, maximum network utilization, fairness among flows, and last but not least, scalability of the solution. The solution should not significantly increase the complexity of router operations. In this paper, we present a new traffic management scheme based on an enhanced Explicit Congestion Notification (ECN) mechanism. Our Multilevel ECN (MECN) conveys more accurate feedback information about the network congestion status than the current ECN. We have designed a TCP source reaction that takes advantage of the extra information provided about congestion. Therefore, MECN responds better to congestion by allowing the system to reach the stability point faster, which results in better network performance. We use control theoretical tools verified by *ns*2 simulations to show that MECN can outperform up to twenty times in term of throughput the *de facto* standard RED/ECN.

1. Introduction

There is a strong demand for Quality of Service (QoS) in the Internet. One key element of QoS is traffic management. Since the network traffic is bursty, it is difficult to make any QoS guarantees without proper control of traffic. Currently, Internet Protocol (IP) only has minimal traffic management capabilities. The packets are dropped when the queue exceeds the buffer capacity. The Transmission Control Protocol (TCP) uses the packet drop as a signal of congestion and reduces its load [1]. While in the past this strategy has worked satisfactorily, now we need better strategies for two reasons [2]. First, the bandwidth of the networks, as well as the distances, are increasing. For very high distance-bandwidth product networks, packet drop is not the optimal congestion indicator. Several megabytes of data may be lost in the time required to detect and respond to packet losses. Therefore, a better strategy for traffic management in IP networks is required. Second, a large part of the traffic, particularly voice and video traffic does not use TCP. Continuous media traffic uses User Data Protocol (UDP). The proportion of UDP traffic is increasing at a faster pace than TCP traffic. The UDP traffic is congestion insensitive in the sense that UDP sources do not reduce their load in response to congestion [3]. Despite the fact that a number of schemes have been proposed for congestion control, the search for new schemes continues [2),4)~25)]. A survey of various congestion control algorithms proposed for use in routers can be found in Refs. 26) and 27).

The research in this area has been going on for at least two decades, for the following reasons: first, there are requirements for congestion control schemes that make it difficult to get a satisfactory solution; second, there are several network policies that affect the design of a congestion scheme. Thus, a scheme developed for one network, traffic pattern, or service requirements may not work on another network, traffic pattern, or service requirements.

The proposed solutions expand over a wide spectrum of improvements. At one end of this spectrum are simpler, more incremental and more easily employable changes to the current TCP. Examples of such proposed solutions are RED [4] and ECN [2]. At the other end of the spectrum, there are solutions with more powerful changes that result in new transport protocols with higher performance but with less chance to be deployed in a large scale on the Internet at least in the immediate future. An example of such solution is XCP [19]. Other proposals, such as REM [17], Proportional Integral Controller [18], HighSpeed TCP [28], and Quick Start TCP [29] reside along the simplicity-deployability spectrum. The choice among all

†1 Department of Computer Science, Louisiana State University, USA
†2 Department of Information and Communication Engineering, Fukuoka Institute of Technology (FIT)
†3 Washington University in St. Louis, USA
†4 Tokyo Denki University

these solutions depends on the tradeoff between performance and practical use that will better fit the Internet. Because of the size and multidimensional complexity of the Internet, the robustness in heterogeneity is valued over efficiency of performance, which leads to favor evolution compared to revolution of changes. For this reason, in our solution we propose minimal changes to ECN and try to derive the maximum performance improvements out of them.

Recognizing the need for a more direct feedback of congestion information, the Internet Engineering Task Force (IETF) has come up with Explicit Congestion Notification (ECN) method for IP routers [2),30)]. A bit in the IP header is set when the routers are congested. ECN is much more powerful than the simple packet drop indication used by existing routers and is more suitable for high distance-bandwidth networks. In this paper, we extend and justify with theoretical and simulation results the enhancements to ECN based on multilevel ECN (MECN), which we presented in Refs. 31) and 34). Our Multilevel ECN (MECN) conveys more accurate feedback information about the network congestion status than the current ECN. We have designed a TCP source reaction that takes advantage of the extra information provided about congestion. Therefore, MECN responds better to congestion by allowing the system to reach the stability point faster, which results in better network performance as shown in our results in this paper. Another proposal to use two bits for signaling congestion is presented in Ref. 25), but their scheme is different from our MECN. The scheme presented in Ref. 25) measures the congestion by measuring the traffic load, while MECN uses the queue length for the same purpose. We would like to stress that, so far, all congestion control schemes used for TCP/IP are based on measurement of the queue. Therefore, our approach would require minimal change on routers.

The remaining of the paper is organized as follows. In Section 2, we present the MECN scheme. In Section 3, we derive the equations for Delay Margin (DM) and Sensitivity using the linearized fluid flow model. In Section 4, we verify the results of Section 3 using *ns*2 simulations. In Section 5, we discuss the parameter setting and implementation issues. The conclusions of the study are given in Section 6.

2. Multilevel Explicit Congestion Notification (MECN)

2.1 Marking Bits at the Router

The current proposal for ECN uses two bits in the IP header (bits 6 and 7 in the TOS octet in IPv4, or the traffic class octet in IPv6) to indicate congestion. The first bit is called ECT (ECN-Capable Transport) bit. This bit is set to 1 in the packet by the traffic source if the source and receiver are ECN capable. The second bit is called the CE (Congestion Experienced) bit. If the ECT bit is set in a packet, the router can set the CE bit in order to indicate congestion. The two bits specified for the purpose of ECN can be used more efficiently to indicate congestion, since using two bits we can indicate four different levels. If non ECN-capable packets are identified by the bit combination of ''00'', we have three other combinations to indicate three levels of congestion. In our scheme the bit combination ''01'' - indicates no congestion, ''10'' - indicates incipient congestion and ''11'' - indicates moderate congestion.

Packet drop occurs only if there is severe congestion in the router and when the buffer overflows. So, with packet-drop we have four different levels of congestion indication and appropriate action could be taken by the source TCP depending on the level of congestion. The four levels of congestion are summarized in **Table 1**. The marking of CE, ECT bits is done using a multilevel RED scheme. The RED scheme has been modified to include another threshold called the mid_{th}, in addition to the min_{th} and max_{th}. If the size of the average queue is between min_{th} and mid_{th}, there is incipient congestion and the CE, ECT bits are marked as ''10'' with probability p_1. If the average queue is between mid_{th} and max_{th}, there is moderate congestion and the CE, ECT bits are marked as ''11'' with probability p_2. If the average queue is above the max_{th} all packets are dropped. The packet dropping policy of RED is shown in **Fig. 1**. The modified packet marking/dropping policy of MECN is shown in

Table 1 Router response to congestion (probabilistic marking of CE and ECT bits and packet dropping).

Congestion State	CE bit	ECT bit
No Congestion	0	1
Incipient Congestion	1	0
Moderate Congestion	1	1
Severe Congestion	Packet	Drop

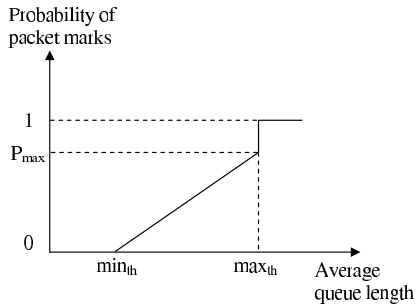

Fig. 1 Probabilities of marking packets in RED.

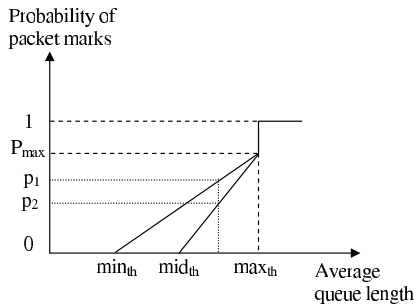

Fig. 2 Probabilities of marking packets in MECN.

Fig. 2.

We would like to stress that the major advantage of MECN as compared to other congestion management schemes is that it conveys more accurate feedback information about the network congestion status than the current ECN. We have designed, as shown in Section 2.3, a TCP source reaction that takes advantage of the extra information provided about congestion. This is the reason why MECN responds better to congestion by allowing the system to reach the stability point faster, which results in better network performance as shown in our simulation results.

2.2 Feedback from Receiver to Sender

The receiver reflects the bit marking in the IP header, to the TCP ACK. Since we have three levels of marking instead of two-level marking in the traditional ECN, we make use of three combination of the 2 bits 8, 9 (CWR, ECE) in the reserved field of the TCP header, which are specified for ECN. In ECN, the bit combination `''00''` - indicates no congestion and `''01''` - indicates congestion. And in piggybacked acknowledgments, `''10''` and `''11''` - indicated noncongestion and congestion respectively, with the receiver source indicating that the congestion window has been reduced.

In our scheme, if the source has to indicate that the congestion window has been reduced, then the congestion information has to wait for the next packet. In this case, the congestion information is ignored. But, this will not cause any major problems to the scheme, because if the congestion is persistent then a lot of packets are going to get marked and the received source will eventually get the congestion information. So in the new scheme, `''00''` - will indicate congestion window reduced, `''01''` - will indicate no congestion, `''10''` - will indicate mild congestion and `''11''` - will indicate heavy congestion. The packet drop is recognized using traditional ways, by timeouts or duplicate ACKs. The marking in the ACKs CWR, ECE bits is shown in **Table 2**.

Table 2 End Host reflecting congestion information (Marking of CWR and ECE bits).

Congestion State	CWR bit	ECE bit
Congestion Window Reduced	0	0
No Congestion	0	1
Incipient Congestion	1	0
Moderate Congestion	1	1

2.3 Response of TCP Source

We believe that the marking of ECN should not be treated the same way as a packet drop, since ECN indicates just the start of congestion, and the buffers still have space. But, there is not actual congestion and the buffers still have space. And now with multiple levels of congestion feedback, the TCP's response needs to be refined.

We have implemented the following scheme. When there is a packet-drop the `''cwnd''` is reduced by $\beta_3 = 50\%$. This is done for two reasons: first, a packet-drop means severe congestion and buffer overflow and some severe actions need to be taken; second, to maintain backward compatibility with routers which do not implement ECN.

For other levels of congestion, such a drastic step as reducing the `''cwnd''` to half is not necessary and might make the flow less vigorous. When there is no congestion, the `''cwnd''` is allowed to grow additively as usual. When the marking is `''10''` (incipient congestion), `''cwnd''` is decreased by $\beta_1\%$. When the marking is `''11''` (moderate congestion) the `''cwnd''` is decreased multiplicatively not by a factor of 50% (as for a packet drop), but by a factor $\beta_2\%$ less than 50% but more than $\beta_1\%$. **Table 3** shows the TCP source responses and the value of βs we have implemented.

Another method could be to decrease additively the window, when the marking is 10 (in-

Table 3 TCP source response.

Congestion State	CWND Change
No Congestion	Increase 'cwnd' additively
Incipient Congestion	Decrease multiplicatively by $\beta_1 = 20\%$
Moderate Congestion	Decrease multiplicatively by $\beta_2 = 40\%$
Severe Congestion	Decrease multiplicatively by $\beta_3 = 50\%$

cipient congestion), instead of maintaining the window. This again will be analyzed in future study.

If the average queue length is less than mid_{th}, then the modified-TCP congestion windows corresponding to the marks ''10'' keep increasing by one every round-trip time in congestion avoidance mode, thus linearly increasing the sending rates of these flows. Consequently, the average queue length will keep increasing unless some marks ''11'' are received by the sources, which correspond to operating in the region where the average queue length is larger than mid_{th}. We can thus conclude that the steady-state average queue length is larger than mid_{th}.

3. Mathematical Modeling of ECN and MECN Schemes

3.1 ECN Mathematical Model

We first give a very brief introduction to an already performed linearization of the ECN scheme. For detailed derivation refer to Ref. 32). The following ignores the TCP slow start and time out mechanisms, thus providing a model and analysis during the congestion avoidance mode only.

The open loop transfer function of the linear model of TCP/RED dynamics in the case of N flows traversing through a single router [32] is then given by:

$$G_0(s) = \frac{\mathrm{e}^{-R_0 s}}{\left(\frac{s}{K}+1\right)(R_0 s+1)} \cdot \frac{K_0}{\frac{R_0^2 C}{2N}s+1}, \tag{1}$$

where N = Number of flows, R_0 = Steady state Round Trip Time (RTT) and C = Link capacity of the router in packets/sec with

$$K_0 = L_{RED_0}\frac{(R_0 C)^3}{(2N)^2}, \tag{2}$$

and K defined by $K = -C\ln(1-\alpha)$, where α is RED's averaging weight and $L_{RED_0} = P_{\max}/(\max_{\text{th}} - \min_{\text{th}})$ is the slope of the probability of packet mark function shown in Fig. 1. The $\max_{\text{th}}$ and $\min_{\text{th}}$ are the maximum and minimum thresholds in packets which are set at the router.

3.2 MECN Mathematical Model

In the following, we do the linearization of the new system following the same method of Ref. 32) for ECN. In fact, we derive the transfer functions when the average queue length is between mid_{th} and $\max_{\text{th}}$.

We ignore the TCP slow start and time out mechanisms, thus providing a model and analysis during the congestion avoidance mode only.

The dynamics of the new TCP in our scheme are derived as follows:

$$\begin{aligned}\dot{W}(t) = {} & \frac{1}{R(t)} \\ & -\frac{W(t)}{\beta_1}\frac{W(t-R(t))}{R(t-R(t))}Prob_1(t-R(t)) \\ & -\frac{W(t)}{\beta_2}\frac{W(t-R(t))}{R(t-R(t))}Prob_2(t-R(t))\end{aligned} \tag{3}$$

$$\dot{q}(t) = \begin{cases} \frac{N(t)}{R(t)}W(t) - C & \text{if } q(t) > 0 \\ \max\left\{0, \frac{N(t)}{R(t)}W(t) - C\right\} & \text{if } q(t) = 0 \end{cases} \tag{4}$$

where $Prob_1$ is the probability of receiving a mark ''01'' and $Prob_2$ is the probability of receiving a mark ''11'', thus $Prob_2 = p_2$ and $Prob_1 = p_1(1-p_2)$.

Using similar techniques as in Ref. 32) a linear model of Eq. (3) and Eq. (4) can be derived.

We first assume that the number of TCP flows and the outgoing link capacity are constant. The operating point $(W_0, q_0, R_0, p_{1_0}, p_{2_0})$ defined by $\dot{W}(t) = 0$ and $\dot{q}(t) = 0$ satisfies

$$W_0^2\left(\frac{p_{1_0}}{\beta_1}(1-p_{2_0}) + \frac{p_{2_0}}{\beta_2}\right) = 1 \tag{5}$$

$$p_{1_0} = (q_0 - \min_{\text{th}})L_{RED_1} \tag{6}$$

$$p_{2_0} = (q_0 - \text{mid}_{\text{th}})L_{RED_2} \tag{7}$$

$$W_0 = \frac{R_0 C}{N} \tag{8}$$

$$R_0 = \frac{q_0}{C} + T_p \tag{9}$$

with $L_{RED_1} = P_{\max}/(\max_{\text{th}} - \min_{\text{th}})$, $L_{RED_2} =$

$P_{\max}/(\max_{\text{th}} - \text{mid}_{\text{th}})$ as shown in Fig. 2.

Next, the time-varying nature of the RTT delay in the terms "$t - R(t)$" is ignored and these terms are approximated by "$t - R_0$". However, the queue length still depends on the RTT in the dynamic Eq. (4).

Let's define

$$f(W, W_R, q, q_R, p_{1_R}, p_{2_R}) = \frac{1}{\frac{q}{C} + T_p} - \frac{WW_R}{\frac{q_R}{C} + T_p}\left[\frac{p_{1_R}}{\beta_1}(1 - p_{2_R}) + \frac{p_{2_R}}{\beta_2}\right] \quad (10)$$

$$g(W, q) = \frac{N}{\frac{q_R}{C} + T_p}W - C \quad (11)$$

where $W_R(t) = W(t - R_0)$, $q_R(t) = q(t - R_0)$, $p_{1_R}(t) = p_1(t - R_0)$ and $p_{2_R}(t) = p_2(t - R_0)$.

By evaluating partials of f and g at the operating point and using similar techniques as in Ref. 32), we obtain the linearized transfer function and neglecting some high-frequency dynamics, the linearized dynamics of TCP-MECN results in the open-loop transfer function

$$G(s) = \frac{1}{\left(\frac{R_0^2 C}{2N}s + 1\right)(R_0 s + 1)} \cdot K_{\text{MECN}} \cdot \frac{\mathrm{e}^{-R_0 s}}{\frac{s}{K} + 1} \quad (12)$$

with

$$K_{\text{MECN}} = \frac{R_0^3 C^3}{2N^2} \cdot \left[\frac{1 - p_{2_0}}{\beta_1}L_{RED_1} + \left(\frac{1}{\beta_2} - \frac{p_{1_0}}{\beta_1}\right)L_{RED_2}\right], \quad (13)$$

where $L_{RED_1} = P_{\max}/(\max_{\text{th}} - \min_{\text{th}})$, $L_{RED_2} = P_{\max}/(\max_{\text{th}} - \text{mid}_{\text{th}})$ as shown in Fig. 2.

The p_{1_0} and p_{2_0} are solutions of Eq. (5) with

$$p_{1_0} = \frac{P_{\max}}{\max_{\text{th}} - \min_{\text{th}}}(q_0 - \min_{\text{th}}) \quad (14)$$

$$p_{2_0} = \frac{P_{\max}}{\max_{\text{th}} - \text{mid}_{\text{th}}}(q_0 - \text{mid}_{\text{th}}). \quad (15)$$

Similarly to Ref. 32), we assume that the low-pass filter pole K is less than the corner frequencies of the new TCP, and that it dominates the closed-loop system behavior. The unity-gain crossover frequency ω_g (i.e. $|G(j\omega_g)| = 1$) thus satisfies:

$$\omega_g \ll \min\left\{\frac{2N}{R_0^2 C}, \frac{1}{R_0}\right\}. \quad (16)$$

Then, at low frequency we have

$$G_0(s) \approx \mathrm{e}^{-R_0 s}\frac{K_0}{\frac{s}{K} + 1} \quad (17)$$

and

$$G(s) \approx \mathrm{e}^{-R_0 s}\frac{K_{\text{MECN}}}{\frac{s}{K} + 1}. \quad (18)$$

3.3 Sensitivity Transfer Function

Ideally, we would like small oscillations around the steady state queue. If the average queue is settling at a value greater than mid_{th}, the queue is not very likely to become empty, and thus we are more concerned with oscillations in the queue length at steady state that lead to jitter (delay variation).

We would like to design the MECN scheme such that the magnitude of the sensitivity transfer function is reduced compared to RED. Ideally, a transfer function with low sensitivity means better tracking of the steady state value.

In order to analyze the performance improvement, we compute the ratio between the sensitivity transfer functions of MECN and ECN for the case where the queue settles above the mid_{th}:

$$\frac{S(s)}{S_0(s)} = \frac{(1 + G(s))^{-1}}{(1 + G_0(s))^{-1}} \quad (19)$$

$$\approx \frac{1 + K_0}{1 + K_{\text{MECN}}} \cdot \frac{1 + \frac{s}{K(1+K_0)}}{1 + \frac{s}{K(1+K_{\text{MECN}})}} \quad (20)$$

where we neglected the dynamics of the time-delays of Eq. (17) and Eq. (18).

If $K_{\text{MECN}} > K_0$, we have a performance improvement for $\omega \in [0, K(1 + K_0)]$ with a sensitivity function reduced such that $\left|\frac{S}{S_0}\right| \approx \frac{1+K_0}{1+K_{\text{MECN}}}$ with $\frac{1+K_0}{1+K_{\text{MECN}}} < 1$.

3.4 Stability Analysis

The delay margin DM is also a parameter of major interest. The DM is a measure of the stability of the system (low oscillations). The phase margins of the systems without delay are (see for example Ref. 33))

$$PM(\omega_g) \approx \pi - \tan^{-1}\left(\frac{\omega_g}{K}\right) \quad (21)$$

where ω_g is such that $|G(j\omega_g)| = 1$, i.e. $\omega_{g_0} = K\sqrt{K_0^2 - 1}$ for the traditional TCP-ECN, and $\omega_g = K\sqrt{K_{\text{MECN}}^2 - 1}$ for TCP-MECN.

The DM, which represents how much the RTT can be increased without violating stability of the feedback system (see for example Ref. 33)), is then

$$DM(\omega_g) \approx \frac{PM(\omega_g)}{\omega_g} - R_0 \quad (22)$$

$$\approx \frac{\pi - \tan^{-1}\left(\frac{\omega_g}{K}\right)}{\omega_g} - R_0. \quad (23)$$

If $K_{\mathrm{MECN}} > K_0$, we have $\omega_g > \omega_{g_0}$, and since $DM(\omega_g)$ is a decreasing function of ω_g, we have a decrease in the DM by using MECN.

The reasons for studying stability in this region are: first, queue oscillations around here can lead to packets being dropped if the queue crosses the $\max_{\mathrm{th}}$; and second, if the queue oscillations reach low values (including zero), which mean that very few or no packet are being served, then the throughput (link utilization) will be low. But, this is the price we pay for having an improved performance while still using such a simple feedback control mechanism. Increasing K_{MECN} further will mean more oscillations that will lead to packets drop.

3.5 Tradeoff of Performance Improvement and Stability Margins

In **Fig. 3**, we plot the DM of Eq. (23) and the performance improvement defined by $(1 + K_{\mathrm{MECN}})/(1 + K_0) > 1$ as a function of K_{MECN}. We clearly see the following tradeoff: K_{MECN} should be chosen as large as possible for good performance improvement, but it should be less than some value to ensure a sufficient DM. For example, in Fig. 3, we see that for the case of $K_0 = 0.1$, $R_0 = 0.2\,\mathrm{s}$ and $K = 0.1$, we can have a DM of 1 s while having a performance improvement $(1 + K_{\mathrm{MECN}})/(1 + K_0) \approx 13.3$ with $K_{\mathrm{MECN}} = 13.64$. By performance improvement, we mean better tracking of the steady state queue, resulting in less jitter.

3.6 Stability and Performance Analysis when Steady State Queue Length Settles Between $\mathbf{min_{th}}$ and $\mathbf{mid_{th}}$

On the other hand, if the average queue settles at a value less than $\mathrm{mid}_{\mathrm{th}}$, it is more important to have a good throughput and thus to have good stability margins to prevent the queue from becoming empty too often as described in Section 3.4.

In this region, we have $L_{RED_2} = 0$. Thus, the open-loop transfer function is

$$G(s) \approx \mathrm{e}^{-R_0 s} \frac{K_1}{\frac{s}{K} + 1}, \quad (24)$$

with

$$K_1 = \frac{R_0^3 C^3}{2\beta_1 N^2} L_{RED_1}. \quad (25)$$

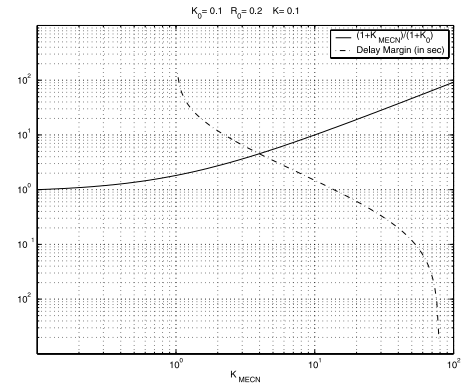

Fig. 3 Tradeoff between performance improvement and DM.

Since $\beta_1 > 2$, we have $K_1 < K_0$, and from the analysis carried out in Section 3.4 we have an increase in the DM as desired.

4. NS2 Simulations

In this section, we use *ns*2 simulations to prove the results obtained in the previous section and also show how to tune the MECN, so that tradeoffs among various performance metrics can be achieved from the algorithm. We are also interested in showing that MECN performs better than ECN, not only in throughput but also in average delay and jitter under various scenarios and traffic mixtures. Because MECN introduces minimal changes to ECN, we compare MECN to ECN and not to solutions with more powerful changes that result in new transport protocols such as XCP [19),25)].

4.1 Simulation Configurations

We used three different simulations configurations to capture a wide range of scenarios and to prove the robustness of the MECN algorithm. The first one is a simple FTP configuration shown in **Fig. 4**. A number of sources S_1, S_2, S_3, ..., S_n are connected to a router R_1 through 10 Mbps, 2 ms delay links. Router R_1 is connected to R_2 through a 1.5 Mbps, 40 ms delay link and a number of destinations D_1, D_2, D_3, ..., D_n are connected to the router R_2 via 10 Mbps, 4 ms delay links. The link speeds are chosen so that congestion will happen only between routers R_1 and R_2 where our scheme is tested. An FTP application runs on each source. Reno TCP is used as the transport agent (the modifications were made to the Reno TCP). The packet size is 1,000 bytes and the acknowledgment size is 40 bytes. The number of sources is varied to alter the congestion

level. The weight used for queue averaging is $\alpha = 0.002$.

The next simulation configuration is used to model a traffic pattern closer to that of today's Internet. The configuration is shown in **Fig. 5**. The model consists of:

(1) 25 exponential traffic sources running over UDP, which creates a self-similar traffic;
(2) 30 web clients and a server, to model the web traffic;
(3) 5 FTP sources to model the bulk data transfer.

The third simulation configuration is used to study the effect of the algorithm on multiple congested gateways. The configuration is shown in **Fig. 6**. It is a typical parking lot configuration. Different flows in the network, travel for different lengths. There are $N + 1$ routers in the network, R_0 to R_N. At routers R_0 to R_{N-1} flows enter the network and at router R_N all flows leave the network. At each router five flows enter the network. In our experiment, we used a configuration with four routers. The throughput of the system was found by adding up the throughput of the individual flows. We intend to show that a system which uses MECN on all routers has a better overall throughput than a system which uses ECN.

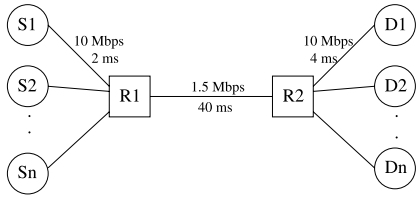

Fig. 4 Network configuration for `ns2` simulations.

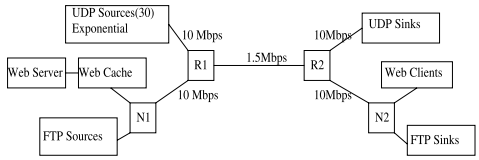

Fig. 5 Simulation configuration for Web traffic.

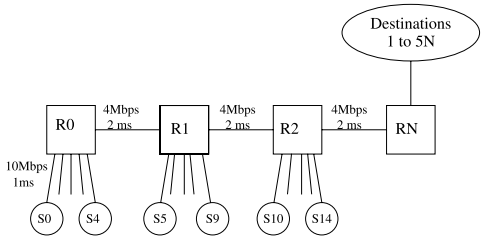

Fig. 6 Simulation configuration for multiple congested gateways.

4.2 Simulations Results

In this section, we use FTP configuration as shown in Fig. 4. We first use low threshold levels. For analyzing ECN, we set $\min_{\text{th}}$ as 1 and $\max_{\text{th}}$ as 4 packets. For analyzing MECN, we set $\min_{\text{th}}$ as 1, mid_{th} as 2 and $\max_{\text{th}}$ as 4 packets. **Figure 7** shows the instantaneous and average queue of ECN where the oscillations in the queue are very high and the queue goes to zero often. This results in a substantial reduction in the throughput of the router. However, the MECN scheme shown in **Fig. 8** gives a higher throughput because the oscillations are reduced. This leads to a higher throughput (link utilization). The control inference of this observation is the fact that the instantaneous queue better tracks the steady-state queue with MECN compared to ECN, thus improving performance. **Figure 9** compares the

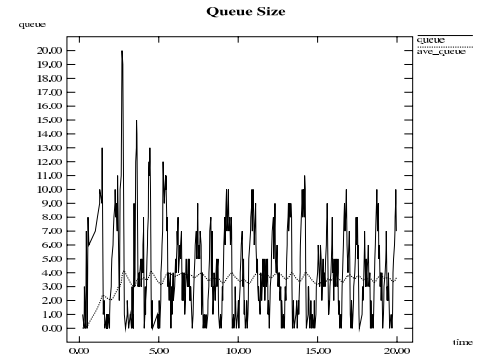

Fig. 7 Queue size of ECN for lower delay, $\min_{\text{th}} = 1$ and $\max_{\text{th}} = 4$.

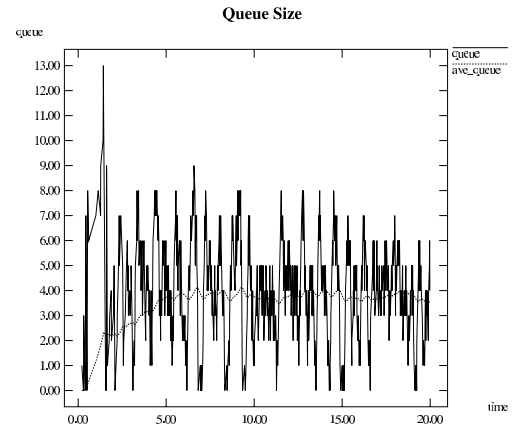

Fig. 8 Queue size of MECN for lower delay, $\min_{\text{th}} = 1$, $\text{mid}_{\text{th}} = 2$, and $\max_{\text{th}} = 4$.

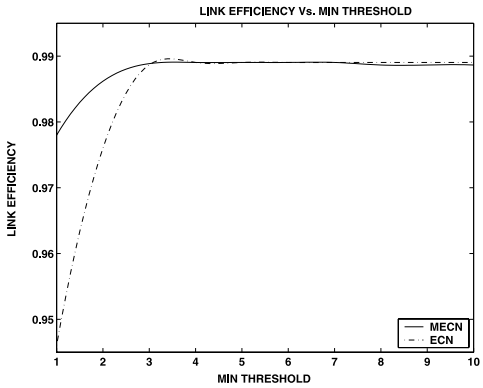

Fig. 9 Comparison of link efficiency for ECN and MECN.

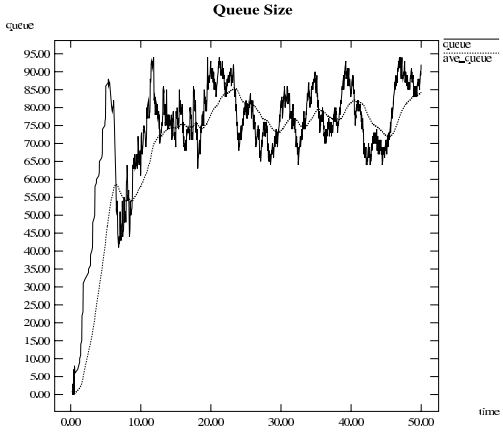

Fig. 10 Queue size of ECN for higher delay.

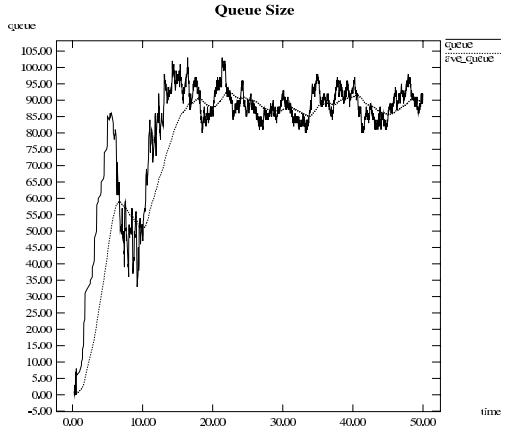

Fig. 11 Queue size of MECN for higher delay.

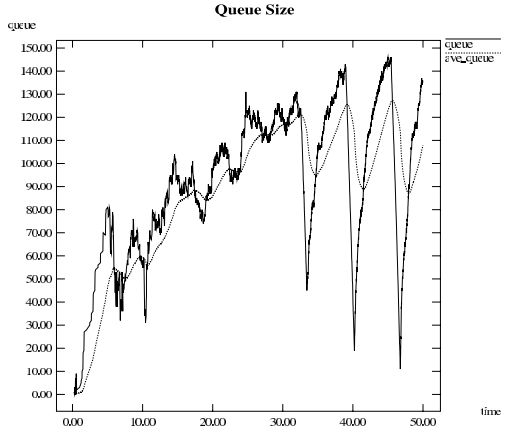

Fig. 12 Effect of parameter tuning on MECN: low K_{MECN} and jitter is 15 ms.

link efficiency of ECN with MECN scheme for low thresholds used in Fig. 7 and Fig. 8, respectively. From Fig. 7 and Fig. 8, we observe that MECN should give a better link efficiency than ECN. This is clearly shown in Fig. 9, where we compare the throughput of ECN and that of MECN by varying $\min_{\mathrm{th}}$. One important observation from Fig. 9 is that MECN provides a much higher throughput (up to 20 times) compared to ECN at the low delay ranges (small $\min_{\mathrm{th}}$).

In **Fig. 10** and **Fig. 11**, we compare the performance of ECN and MECN respectively, over a broader range of thresholds. We set $\min_{\mathrm{th}}$ as 30, $\mathrm{mid}_{\mathrm{th}}$ as 60 and $\max_{\mathrm{th}}$ as 90 packets. An increase in throughput beyond a certain $\max_{\mathrm{th}}$ is not possible. However, the MECN scheme outweighs the ECN scheme, thus the oscillations in the queue are reduced, and that is the reason why MECN reduces the jitter compared to ECN, as shown later in Fig. 16.

From **Fig. 12**, **Fig. 13** and **Fig. 14**, we see that we can improve the performance of MECN by a proper choice of K_{MECN}. For the first case (Fig. 12), we have $N = 40$ and $P_{\max} = 0.1$. In the second case (Fig. 13), we have $N = 40$ and $P_{\max} = 0.2$, thus we have increased K_{MECN} by increasing $P_{\max}$ from 0.1 to 0.2, using Eq. (13). When we compare the two graphs, we see that the oscillations in the queue decrease drastically in the second case and as a result, jitter decreases from 30 ms to 15 ms. The jitter calculated above is the average end-to-end jitter from source to destination for individual flows. Now, we can further extract better performance by still increasing K_{MECN}. In the third case (Fig. 14), we have $N = 20$ and $P_{\max} = 0.2$, thus we increase K_{MECN} by decreasing N and the jitter decreases to 7 ms. The reduction of jitter is a good result for real-time applications that are sensitive to jitter.

As shown above, for any given traffic level N we can tune the MECN parameters using the

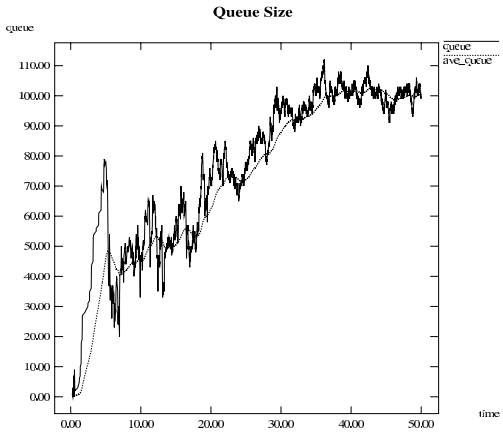

Fig. 13 Effect of parameter tuning on MECN: high K_{MECN} and jitter is 15 ms.

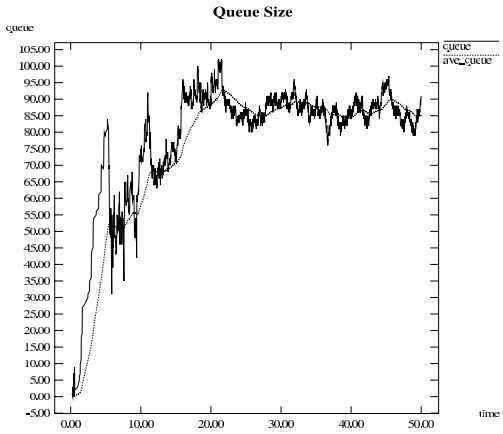

Fig. 14 Effect of parameter tuning on MECN: higher K_{MECN} and jitter is 7 ms.

results in Section 3 to obtain the needed trade-off among delay, throughput and jitter performance. As shown by Fig. 12, Fig. 13 and Fig. 14, the jitter in MECN can be reduced drastically compared to ECN.

Figure 15 shows a plot of throughput versus average delay, which is a useful metric for analyzing performance of network. Ideally, we would like to have high throughput without introducing delay to the traffic. We can clearly see the improvement in performance of MECN over ECN. It is a well known fact that throughput can be traded-off with average delay. We want higher throughput with lesser delay. From the figure, we can see that MECN provides up to 112% more throughput than ENC for the minimal delay. On the other hand MECN gives the same throughput as ECN with less than half of ECN average delay. MECN* indicates fine tuned MECN (larger K_{MECN}) using Eq. (13).

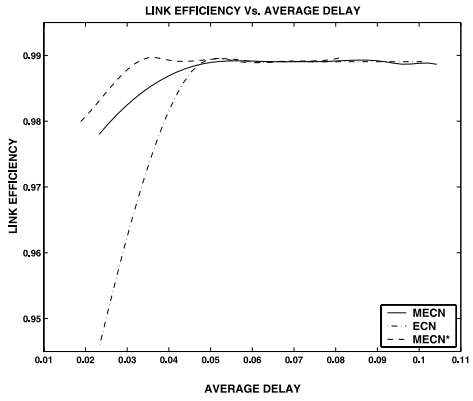

Fig. 15 Comparison of link efficiency vs. average delay for ECN and MECN.

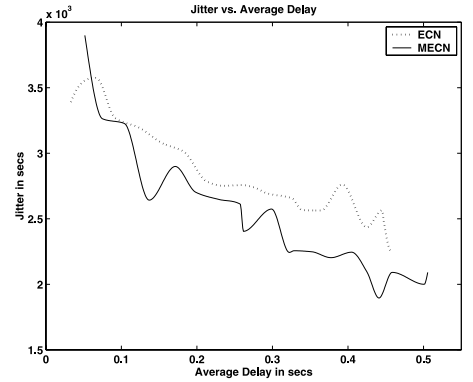

Fig. 16 Jitter vs. average delay for MECN and ECN.

In this case, for MECN* and MECN we use the same parameters used in Fig. 12 and Fig. 14, respectively. As we can see, better performance can be obtained by having a higher K_{MECN}.

Figure 16 and **Fig. 17** are generated by simulations using the simple FTP simulation configuration. Figure 16 shows that MECN consistently has up to 146% less jitter than ECN, which leads to a much smoother traffic pattern. Figure 17 shows the corresponding link efficiencies of the two systems for the same simulation for a given average delay. These two graphs prove that MECN gives up to 7.6 times higher throughput performance at low delays, and also better jitter performance (the measured jitter at the router). A lower jitter is advantageous not only for these sources but also for the whole network, since other audio/video applications running on UDP will also benefit from using MECN at the router.

4.3 Results for Internet Traffic Model

The Internet traffic model described in Section 4.1 and shown in Fig. 5, is used to test the

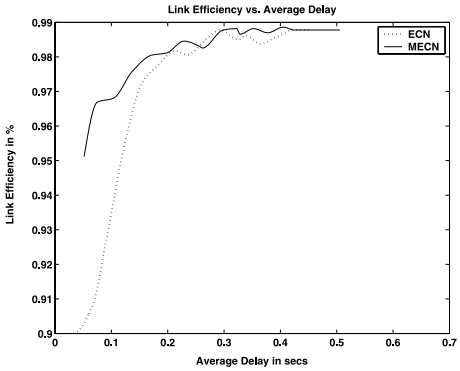

Fig. 17 Link efficiency vs. average delay for MECN and ECN using FTP traffic model.

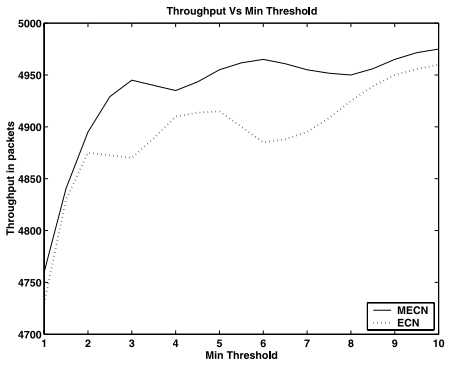

Fig. 19 Throughput vs. Min_{th} for multiple congested gateways.

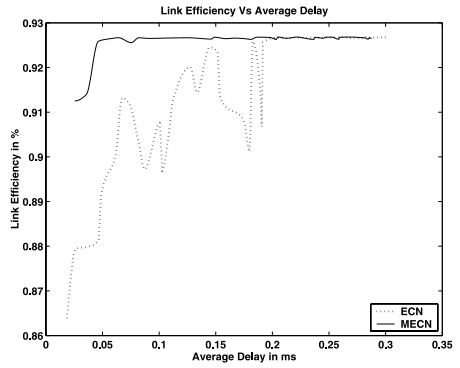

Fig. 18 Link efficiency vs. average delay for ECN and MECN using Web traffic model.

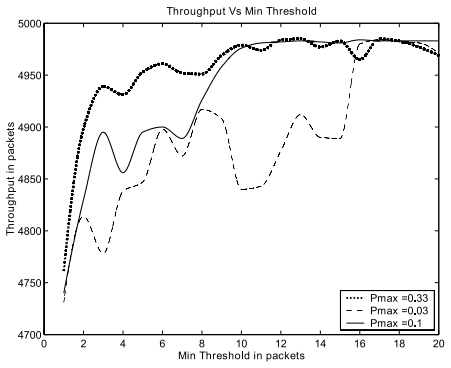

Fig. 20 Throughput vs. Min_{th} for MECN for different $P_{\max}$.

MECN algorithm under typical network traffic. In this model, we would like to simulate typical high bandwidth Internet conditions, which is shown to be characterized by self-similar traffic. We measure the "self-similarity" of our model traffic by using the Hurst parameter. This parameter indicates the degree of traffic self-similarity. Usually Hurst parameter of web traffic is between 0.75 and 0.85. For our mixed model, the Hurst parameter was measured using variance-time plot [35] and was found to be 0.807.

It is found that MECN performs better than ECN for many different RTTs, but only a sample result is shown in **Fig. 18**, where MECN provides up to 12 times more throughput than ECN for low delays. Similar results were observed for RTTs in the range 20–300 ms. Also, the results get better with the increase of percentage of TCP traffic in the network, since only TCP is sensitive to the algorithm.

4.4 Multiple Congested Gateways

The MECN was tested under various network configurations. **Figure 19** shows the plot of throughput versus Min_{th} for ECN and MECN, using the multiple congested gateways configuration defined in Section 4.1 and shown in Fig. 6. The system throughput here is defined as the sum of throughput of the individual flows (the throughput of individual flow is the total number of packets that the receiver received during the simulation time). From the figure, we can see that by using MECN the overall performance of the network increases up to 147.2%, since the performance of each link in the system is increased.

Even though the control theory derived in Section 3 does not exactly capture the dynamics of this configuration, the guidelines derived in that section seems to work even in the case of multiple congested links. This is because the guidelines where derived for a single link and as the efficiency of each link in the system is increased the total network performance also increases. **Figure 20** shows the throughput of MECN system for three different $P_{\max}$. An increase in $P_{\max}$ leads to an increase in K_{MECN} (from Eq. (13)) and a corresponding increase

in the performance. The figure confirms this result. Let us call MECN1 the scheme with $P_{\max} = 0.33$ (largest K_{MECN}), MECN2 with $P_{\max} = 0.1$ (medium K_{MECN}) and MECN3 with $P_{\max} = 0.03$ (smallest K_{MECN}). From Fig. 20, we can see that MECN1 outperforms MECN2 and MECN3 in terms of throughput up to 147% and 3.25 times respectively, while MECN2 provides up to 2.6 times throughput than MECN3.

5. Discussion on Parameters and Implementation

The MECN parameters enable to obtain wide ranges of tradeoffs among throughput, delay, jitter and RTT.

The $\min_{\text{th}}$ determines how soon the packet marking is commenced, and therefore influences the minimum delay. The $\max_{\text{th}}$ determines the amount of burst the link can tolerate. The higher the maximum threshold, higher the link delay and vice versa.

In setting the MECN parameters, the results of Section 3 are a good guidance. So, using Eq. (13) and varying its components, we would like to select a larger K_{MECN} such that we can get the needed performance improvement, captured by Eq. (20), which enable us to keep small the queue oscillations. On the other hand, in the low delay area, small queue oscillations lead to higher throughput because the queue length reaches not often zero. At the same time, small queue oscillations lead to lower jitter.

While improving the performance, following the conclusions of Section 3.4, we have to be careful not to make zero the DM in Eq. (23), which represents how much the RTT can be increased without leading to oscillations.

The implementation of the MECN scheme is going to be similar to ECN since they follow the similar architecture, that means the congestion is measured by measuring the queue length.

6. Conclusions

In this paper, we have studied the performance of a MECN scheme. We propose minimal changes to ECN and try to derive the maximum performance improvements out of them. Two bits are now used to indicate four levels of congestion. The major advantage of MECN is that it conveys more accurate feedback information about the network congestion status than the current ECN. We have designed a TCP source reaction that takes advantage of the extra information provided about congestion. Therefore, MECN responds better to congestion by allowing the system to reach faster the stability point, which results in better network performance as shown in our simulation results. We have explained the performance improvement of MECN over ECN using classical control theory tools and these results have been validated by *ns*2 simulations. For low thresholds (delay), MECN obtains up to twenty times more throughput than ECN. For higher thresholds (delay), the improvement is seen in the reduction up to 146.6% of the jitter experienced by the flows.

Acknowledgments This research work has been partially supported by National Science Foundation Awards NSF-CNS-9980637 and NSF-CNS-0413187.

References

1) Stevans, W.: TCP Slow Start, Congestion Avoidance, Fast Retransmit and Fast Recovery, RFC 2001 (1997).
2) Ramakrishnan, K. and Floyd, S.: A Proposal to Add Explicit Congestion Notification (ECN) to IP, RFC 2481 (1999).
3) Floyd, S.: TCP and Explicit Congestion Notification, *ACM SIGCOMM Computer Communications Review*, Vol.24, No.5, pp.8–23 (Oct. 1994).
4) Floyd, S. and Jacobson, V.: Random Early Detection Gateways for Congestion Avoidance, *IEEE/ACM Trans. on Networking*, Vol.1, Issue 4, pp.397–413 (Aug. 1993).
5) Clark, D.D. and Fang, W.: Explicit Allocation of Best-effort Packet Delivery Service, *IEEE/ACM Trans. on Networking*, Vol.6, No.4, pp.362–373 (Aug. 1998).
6) Feng, W., Kandlur, D., Saha, D. and Shin, K.: Blue: A New Class of Active Queue Management Algorithms, *Technical Report UM-CSE-TR-387-99*, University of Michigan (1999).
7) Kalyanaraman, S., Jain, R., Fahmy, S., Goyal, R. and Vandalore, B.: The Erica Switch Algorithm for ABR Traffic Management in ATM Networks, *IEEE/ACM Trans. on Networking*, Vol.8, No.1, pp.87–98 (Feb. 2000).
8) Floyd, S. and Fall, K.: Promoting the Use of End-to-End Congestion Control in the Internet, *IEEE/ACM Trans. on Networking*, Vol.7, Issue 4, pp.458–472 (Aug. 1999).
9) Floyd, S. and Fall, K.: Router Mechanisms to Support End-to-End Congestion Control, *LBL Technical Report* (Feb. 1997).
10) Mathis, M., Semke, J., Mahdavi, J. and Ott, T.: The Macroscopic Behaviors of the TCP

congestion Avoidance Algorithm, *Computer Communications Review*, Vol.27, No.3, pp.67–82 (July 1997).
11) Chiu, D. and Jain, R.: Analysis of the Increase/Decrease Algorithms for Congestion Avoidance in Computer Networks, *Journal of Computer Networks and ISDN Systems*, Vol.17, No.1, pp.1–14 (July 1989).
12) Floyd, S. and Henderson, T.: The NewReno Modification to TCP's Fast Recovery Algorithm, Internet RFC 2582, Experimental (Apr. 1999).
13) Mathis, M., Mahdavi, J., Floyd, S. and Romanow, A.: TCP Selective Acknowledgment Options, IETF Internet RFC 2018 (Oct. 1996).
14) Ramakrishnan, K.K. and Jain, R.: A Binary Feedback Scheme for Congestion Avoidance in Computer Networks, *ACM Trans. on Computer Systems*, Vol.8, No.2, pp.158–181 (May 1990).
15) Jain, R., Kalyanaraman, S. and Viswanathan, R.: Rate Based Schemes: Mistakes to Avoid, AF-TM 940882 (Sep. 1994).
16) Jain, R., Kalyanaraman, S. and Viswanathan, R.: Ordered BECN: Why We Need a Timestamp or Sequence Number in the RM Cell, ATM Forum/94-0987 (Oct. 1994).
17) Athuraliya, S., Low, S., Li, V. and Yin, Q.: REM Active Queue Management, *IEEE Network Magazine*, Vol.15, No.3, pp.48–53 (Aug. 2001).
18) Hollot, C., Misra, V., Towsley, D. and Gong, W.B.: On Designing Improved Controllers for AQM Routers Supporting TCP Flows, *Proc. INFOCOM-2001*, San Francisco, CA, Vol.3, pp.1726–1734 (2001).
19) Katabi, D., Handley, M. and Rohrs, C.: Internet Congestion Control for Future High Bandwidth-delay Product Environments, *Proc. ACM SIGCOMM-2002*, pp.69–102 (2002).
20) Wang, C., Li, B., Hou, T., Sohraby, K. and Lin, Y.: LRED: A Robust Active Queue Management Scheme Based on Packet Loss Ratio, *Proc. IEEE INFOCOM-2004*, Vol.1, pp.1–12 (Mar. 2004).
21) Jin, C., Wei, D.X. and Low, S.H.: FAST TCP: Motivation, Architecture, Algorithms, Performance, *Proc. IEEE INFOCOM-2004*, pp.2490–2501 (Mar. 2004).
22) King, R., Riedi, R. and Baraniuk, R.: TCP-Africa: An Adaptive and Fair Rapid Increase Rule for Scalable TCP, *Proc. IEEE INFOCOM-2005*, Vol.3, pp.1838–1848 (Mar. 2005).
23) Low, S., Andreq, L. and Wydrowski, B.: Understanding XCP: Equilibrium and Fairness, *Proc. IEEE INFOCOM-2005*, Vol.2, pp.1025–1036 (Mar. 2005).
24) Wu, Q. and Rao, N.: A Class of Reliable UDP-based Transport Protocols based on Stochastic Approximation, *Proc. IEEE INFOCOM-2005*, Vol.2, pp.1013–1024 (Mar. 2005).
25) Xia, Y., Subramanian, L., Stoica, I. and Kalyanaraman, S.: One More Bit is Enough, *Proc. ACM SIGCOMM-2005*, Philadelphia, PA (Aug. 2005).
26) Low, S.H., Paganini, F. and Doyle, J.C.: Internet Congestion Control, *IEEE Control Systems Magazine*, Vol.22, pp.28–43 (Oct. 2002).
27) Medina, A., Allman, M. and Floyd, S.: Measuring the Evolution of Transport Protocols in the Internet, *ACM CCR*, Vol.35, No.2, pp.35–51 (Apr. 2005).
28) Floyd, S.: HighSpeed TCP for Large Congestion Windows, IETF, RFC 3649 (Dec. 2003).
29) Jain, A., Floyd, S., Allman, M. and Sarolahti, P.: Quick-Start for TCP and IP, IETF Internet-draft, draft-amit-quick-start-04.txt (Feb. 2000).
30) Floyd, S.: TCP and Explicit Congestion Notification, *ACM SIGCOMM Computer Communications Review*, Vol.24, No.5, pp.8–23 (Oct. 1994).
31) Durresi, A., Sridharan, M., Liu, C., Goyal, M. and Jain, R.: Traffic Management Using Multilevel Explicit Congestion Notification, *Proc. 5th World Multi-Conference on Systemics, Cybernetics and Informatics* (*SCI-2001*), *ABR over the Internet*, Orlando, FL, pp.12–17 (July 2001).
32) Hollot, C., Misra, V., Towsley, D. and Gong, W.B.: Analysis and Design of Controllers for AQM Routers Supporting TCP Flows, *IEEE Trans. on Automatic Control*, Vol.47, No.6, pp.945–959 (June 2002).
33) Özbay, H.: *Introduction to Feedback Control Theory*, CRC Press LCC, Boca Raton, FL (1999).
34) Quet, P.F., Chellappan, S., Durresi, A., Sridharan, M., Özbay, H. and Jain, R.: Guidelines for Optimizing Multi-Level ECN Using Fluid Flow Based TCP Model, *Proc. ITCOM-2002*, Quality of Service over Next Generation Internet, Boston, pp.106–116 (July 2002).
35) Leland, W., Taqqu, W., Willinger, W. and Wilson, D.: On the Self-similar Nature of Ethernet Traffic, *IEEE/ACM Trans. on Networking*, Vol.2, pp.1–15 (Aug. 1994).

(Received May 15, 2006)
(Accepted November 2, 2006)
(Released February 7, 2007)

Arjan Durresi received the B.S.E.E., M.S. and Ph.D. (all summa cum laude) all in Electronic and Telecommunications, in 1986, 1991 and 1993, respectively and a Diploma of Superior Specialization in Telecommunications from La Sapienza University in Rome, Italy and Italian Telecommunications Institute in 1991. He is currently an Assistant Professor in the Department of Computer Science, Louisiana State University. His current research interests include network architectures, heterogeneous wireless networks, security, QoS routing protocols, traffic management, satellite networks, and bioinformatics. Dr. Durresi has authored more than 100 papers in refereed Journals and International Conference proceedings. He is an area editor for the Ad Hoc Networks Journal. He was Program Co-Chair of AINA-2006 and is Workshops Co-Chair of AINA-2007. Dr. Durresi has recived many Research Awards. He received the appreciation certificate from IEEE Computer Society in 2005. He is a senior member of the IEEE.

Leonard Barolli received B.E. and Ph.D. degrees from Tirana University and Yamagata University in 1989 and 1997, respectively. Presently, he is a Professor at the Department of Information and Communication Engineering, Fukuoka Institute of Technology (FIT). Dr. Barolli has published more than 180 papers in refereed Journals and International Conference proceedings. He was an Editor of the IPSJ Journal and has served as an Guest Editor for many Special Issues of International Journals. Dr. Barolli has been a PC Member of many International Conferences. He was a PC Co-Chair of AINA-2003, PC Chair of AINA-2004, PC Chair of ICPADS-2005, General Co-Chair of AINA-2006 and Workshops Chair of iiWAS-2006. He is Workshops Co-Chiar of AINA-2007. His research interests include network traffic control, fuzzy control, genetic algorithms, ad-hoc networks, sensor networks and P2P systems. He is a member of SOFT, IPSJ, and IEEE.

Raj Jain is a Professor of Computer Science and Engineering at Washington University in St. Louis. He is also a Co-founder and Chief Technology Officer of Nayna Networks, Inc. He was a Professor of Computer and Information Sciences at The Ohio State University in Columbus until August 2002 and then an Adjunct Professor until August 2004. He is a Fellow of IEEE, a Fellow of ACM and is on the Editorial Boards of Computer Communications, Journal of High Speed Networks, Mobile Networks and Nomadic Applications, International Journal of Virtual Technology and Multimedia and International Journal of Wireless and Optical Communications. He has 14 patents, more than 40 journal and magazine papers, and more than 60 conference papers. His papers have been widely referenced and he is known for his research on congestion control and avoidance, traffic modeling, performance analysis, and error analysis.

Makoto Takizawa received his B.E., M.E. and Ph.D. degrees from Tohoku University, Japan in 1973, 1975, 1984, respectively. From 1975 to 1986, he worked for Japan Information Processing Developing Center (JIPDEC). He has been a Full professor of the Department of Computers and Systems Engineering, Tokyo Denki University (TDU) since 1986. He was Dean of the Graduate School of Science and Engineering, Tokyo Denki University from April of 2001 to March of 2005. He is a Golden Core Member and a Board of Governors (BoG) of IEEE Computer Society. He is a senior member of IEEE. He has been a PC member, PC Chair and General Chair of many International Conferences. He is Steering Committee Chair of IEEE AINA International Conference. He has published more than 70 Journal papers and 250 International Conference papers. His research interest includes network protocols, group communication, distributed systems, fault-tolerant systems and P2P systems. He is a member of IEEE, ACM, and IPSJ.