

# Implementation and Performance Analysis of Active Queue Management Mechanisms

Stanislav Mišković, Grozdan Petrović, and Ljiljana Trajković

**Abstract**—In this paper, we describe active queue management (AQM) mechanisms that are employed in the Internet Protocol (IP) routers. We address the functionality and implementation of AQM blocks in IP routers. We also describe the benefits of the AQM mechanisms and justify the need for their implementation. Simulation results illustrate the fairness and the efficient use of resources in networks with the AQM support.

**Keywords**—Active queue management, random early detection, congestion avoidance, TCP/IP.

## I. INTRODUCTION

The Internet is based on the Transport Control Protocol/Internet Protocol (TCP/IP) protocol stack, where the IP protocol controls communication on network layer and offers a connectionless and best effort service. This design provides greater efficiency, flexibility, and robustness than the connection oriented service. The drawback is the unpredictable state of the next hop in a communication path. Packets may be reordered and/dropped at any network node. Traffic management mechanisms are employed in order to provide efficient exchange of data. Most Internet traffic is transported by the TCP end-to-end transport protocol. Its main functions are flow control, congestion control, and congestion avoidance. In today's Internet, end-to-end congestion control may cause increased levels of control traffic, increased drop rate, spurious packet retransmissions, spurious TCP timeouts and back-offs, increased processing on the transport layer, and increased packet delay and delay jitter.

The Internet growth requires expansion of congestion control to network midpoints (routers and switches). This may be achieved by departing from the current best effort service model to a guaranteed quality of service (QoS) framework. Such architecture may be only employed if network domains belong to a single or compliant administrative control.

Recent recommendations of the Internet Engineering Task Force (IETF) advocate the development and deployment of active queue management (AQM) mechanisms [1]–[3] and more sophisticated scheduling algorithms. The key advantage

of AQM is its capability to seamlessly cooperate with the TCP traffic management and its congestion control process. AQM usually requires less processing and a smaller number of traffic state variables than scheduling algorithms. AQM may be gradually implemented within the Internet architecture without causing negative effects on neighboring network nodes. Examples of deployed AQM mechanisms are Random Early Detection (RED) [1], its improved variants gentle RED [2] and adaptive RED [3], and Explicit Congestion Notification (ECN) [4]. It is expected that AQM will improve the current best-effort service by reducing the negative effects of non-managed queues such as FIFO buffers with the DropTail queuing discipline currently dominant in the Internet routers.

This paper is organized as follows: The introduction is given in Section 1. We describe implementation of AQM in the Internet routers in Section 2. The need for AQM and description of expected benefits of its wider Internet deployment is addressed in Section 3. We describe RED, its improved variants gentle RED and adaptive RED, and also analyze the effects of ECN. In Section 4, we provide an overview of the current status of AQM and the Internet developments. The performance criteria are given in Section 5, while simulation platform and scenarios are described in Section 6. Simulation results for two test network topologies are given in Sections 7 and 8. We conclude with Section 9.

## II. IMPLEMENTATION OF THE AQM IN IP ROUTERS

The two main reasons for traffic congestion are bandwidth bottlenecks and limited buffers. While bandwidth may only be increased by allocating additional network resources, queuing may be also improved by employing better queue management algorithms.

We describe the architecture of an AQM block in IP routers. A network router consists of input interfaces, backplane (switching fabric), control logic, and output interfaces, as shown in Fig. 1. Its performance is predominantly determined by the speed of packet forwarding and blocking characteristics of the switching fabric. Switching devices range from a simple PC based router to a parallel backplane and multiprocessor device such as the Cisco CRS-1 router. Design of high performance backplanes originated from technologies such as the Clos backplane used in telephone networks and the Asynchronous Transfer Mode (ATM).

Stanislav Mišković is with Institute Mihajlo Pupin, Volgina 15, 11000 Belgrade, Serbia and Montenegro (e-mail: miskovic@yubc.net).

Grozdan Petrović is with Faculty of Electrical Engineering, Bul. kralja Aleksandra 73, 11000 Belgrade, Serbia and Montenegro (e-mail: gpetrovic@etf.bg.ac.yu).

Ljiljana Trajković is with Simon Fraser University, Burnaby, BC, Canada (e-mail: ljilja@cs.sfu.ca).

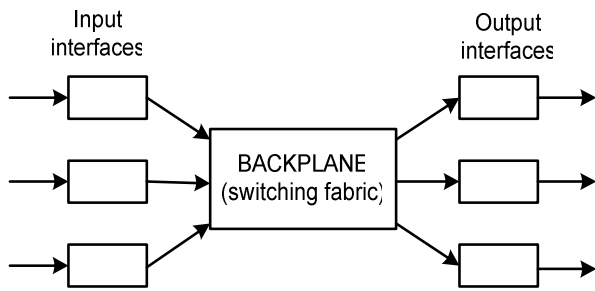


Fig. 1. Basic building blocks of a router.

Packet queues may be implemented at input and/or output router interfaces. Each queue may have AQM logic and may be further enhanced with ECN in order to reduce the overall packet drop rate. The block diagram of a router interface with a queuing support for the QoS framework is shown in Fig. 2. The policing block is an administrative tool used to limit traffic rate. The classifier reads the QoS information carried in the IP header and directs packet to a queue with an appropriately assigned priority. Multiple queues in the buffer block may be configured to function within the QoS framework by employing AQM for each QoS class. Nevertheless, it is a common practice not to deploy AQM in the highest priority queues [5] because the highest priority traffic will be allocated adequate bandwidth. The scheduler combines outputs of various queues in packet streams that are sent to the output of the router interface. In the absence of a QoS policy, the role of the classifier is simplified and buffers are not divided in multiple priority queues. In such case, only the AQM block may improve the end-to-end packet exchange.

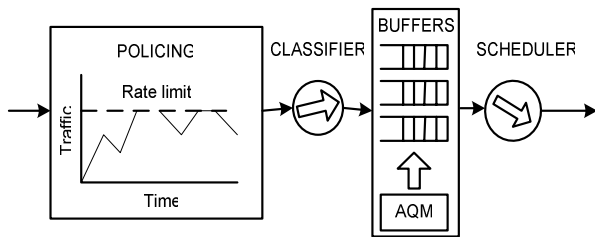


Fig. 2. Block diagram of a router interface. Classifier classifies incoming packets to buffers according to their priority. Scheduler generates a unified packet stream according to buffer priority.

### III. THE ACTIVE QUEUE MANAGEMENT MECHANISMS

#### A. Negative effects of DropTail queue discipline

The FIFO buffering with the DropTail queuing discipline is commonly used in the Internet routers. DropTail is a rather simple discipline that does not rely on estimating traffic properties. It admits packets to a queue until the queue length reaches the *buffer capacity*. Subsequent packets are discarded (or marked) until the buffer space becomes available. The DropTail mechanism does not employ congestion notification. Traffic bursts are common in packet networks, and, hence, an almost full DropTail queue may cause multiple packet drops.

This negatively affects both the use of network resources and the operation of the TCP congestion control mechanisms. Traffic measurements and network simulations identified lockout, bias towards shorter round-trip-time (RTT), and global synchronization as problems caused by the DropTail discipline, as illustrated in Fig. 3. A lockout occurs when a flow seizes most of the available bandwidth. Bias towards shorter RTT flows, similar to a lockout, occurs when a flow with shorter RTT assumes throughput advantage due to faster TCP recovery. Global synchronization [6] begins when multiple flows repeatedly lose packets simultaneously. This leads to high oscillations in the usage of the available network bandwidth.

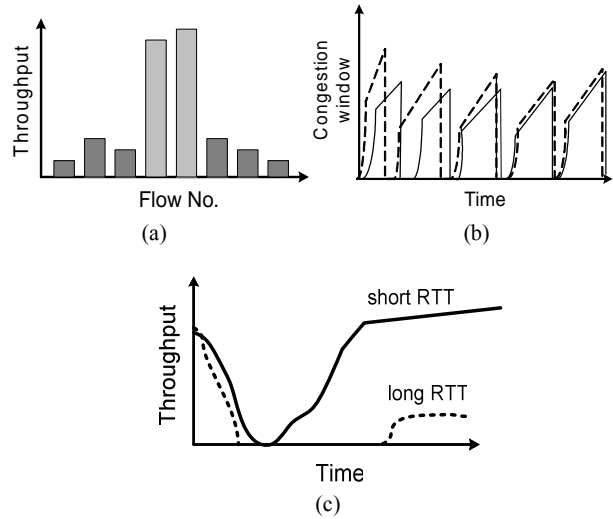


Fig. 3. Negative effects of the DropTail discipline: (a) lockout effect, (b) global synchronization effect, and (c) short RTT bias.

#### B. Benefits of the Active Queue Management

A key AQM feature is its cooperation with the TCP congestion control mechanisms. TCP Reno is the most widely deployed TCP protocol. It controls traffic flow and traffic congestion by monitoring packet drop rate and sequencing the end-to-end acknowledgements.

RED mechanism is widely implemented and deals with the onset of congestion by employing probabilistic random packet drops. Packets are discarded equally across the flows. This cancels the negative effects of DropTail, improves the fair sharing of network resources, reduces the end-to-end delay, and reduces the overall packet drop rate.

AQM improves the end-to-end packet delays and most AQM mechanisms try to optimize buffering vs. delay tradeoff. This is accomplished by keeping the average queue length lower than the *buffer capacity*. Hence, a packet spends less time on average in queues. Internet traffic is bursty [7]. Because the AQM mechanisms control *only* the average queue length, the instantaneous queue length could extend to the entire buffer space and thus compensate for the temporary traffic bursts. If the periods of full buffer occupancy seldom occur, the overall packet drop rate will be reduced and the

packet delay shortened. This enhances the TCP performance because TCP reacts to increased packet loss with dramatic reduction of its throughput.

### C. Random Early Detection and Explicit Congestion Notification

The RED mechanism best perform in cooperation with traffic management protocols that register congestion through packet drops, such as TCP Reno. RED is implemented in most commercial operating systems (Linux, Windows XP, and Windows 2000). Network equipment vendors also employ proprietary variants of RED, such as Cisco's proprietary Weighted Random Early Detection (WRED).

RED does not classify traffic. Instead, it requires five configuration parameters: *buffer capacity*, lower threshold  $min_{th}$ , upper threshold  $max_{th}$ , maximum drop probability  $max_p$ , and weight coefficient  $w_q$ . RED continuously estimates average queue length ( $avg$ ), and instantaneous queue length ( $q$ ):

$$avg_i = (1 - w_q) \times avg_{i-1} + w_q \times q. \quad (1)$$

Threshold parameters  $min_{th}$  and  $man_{th}$  divide the buffer in three areas, as shown in Fig. 4. The value of  $avg$  controls the behavior of the RED management No packets are discarded if  $avg$  is smaller than threshold ( $min_{th}$ ). RED acts if  $avg$  is between lower ( $min_{th}$ ) and upper ( $max_{th}$ ) thresholds by dropping packets with drop probability linearly proportional to  $avg$ . These probabilistic drops are called *early drops*. They serve as an indication of an upcoming congestion. An optimal operation of the RED mechanism should maintain the queue length  $avg$  within the ( $min_{th} - max_{th}$ ) area. RED functions as DropTail when  $avg$  increases beyond  $max_{th}$ .

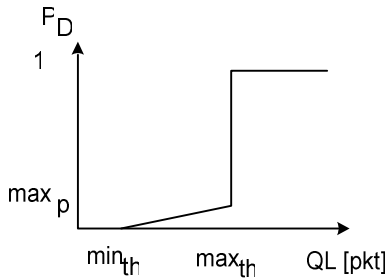


Fig. 4. Operation of the RED queuing discipline. Labels:  $P_D$  is drop probability,  $max_p$  is the maximum drop probability, and  $QL$  denotes queue length.

To avoid behavior similar to DropTail, the RED parameters need to be configured based on traffic characteristics. This is not easily accomplished and it has been shown that RED may remain locked in a DropTail behavior [8]. Two RED modifications are shown in Fig. 5. Gentle RED [2] continues with early drops above  $max_{th}$  with an increased drop probability. Adaptive RED [3] adjusts the slope of the drop probability function. Other proposed RED improvements are FRED [9] and RED-PD [10], which classify flows and, therefore, increase the number of observed state variables.

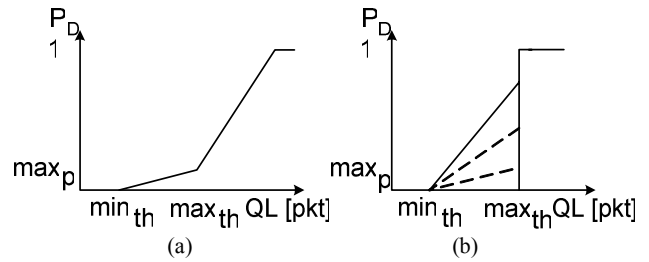


Fig. 5. RED algorithms: (a) Gentle RED and (b) Adaptive RED.

The IETF encourages implementation of the ECN framework [4] as an upgrade to AQM and TCP. ECN was originally introduced by technologies such as DNA (DECbit), Frame Relay (FECN and BECN mechanisms), and ATM (EFCI bit). When combined with RED, ECN operates within the ( $min_{th} - max_{th}$ ) RED area by marking packets, rather than employing early packet drops. ECN functions are advertised in both TCP and IP headers. TCP advertises support for ECN during connection establishment. ECN flags in the network layer are included in the Type of Service (ToS) field of the IP header. Network midpoints can insert congestion markers into the flags. Packet marking makes the exchange of congestion notifications faster. If both end points and certain routers on the path of a TCP flow support ECN, congestion notifications may be advertised within a single RTT period, as shown in Fig. 6. Standards define that a TCP source node must react to the ECN marking as if the packet was dropped [4]. It is assumed that any other action would be unfair to the majority of TCP flows that do not support ECN.

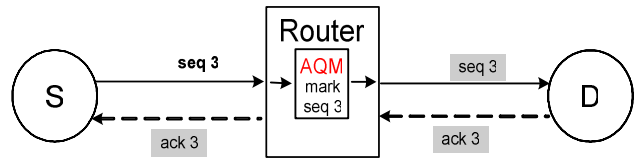


Fig. 6. ECN marking process: congestion notification markers are exchanged within one RTT period. Labels: S and D denote source and destination of the TCP connection. Seq 3 and ack3 are segment with sequence number 3 and its acknowledgement, respectively. AQM mechanism in the router marks seq 3. The marker is sent back to the TCP source node via ack 3.

## IV. RELEVANT RESEARCH

Performance of an Internet protocol depends on traffic properties and, ideally, RED parameters should be configured in accordance to traffic parameters. Interactions between Internet traffic and Internet protocols are studied through modeling, simulation, and test-bed experiments.

Explosion of variable state space is a typical problem for modeling of protocols. Details that may be omitted during a modeling process need to be carefully chosen. A model for RED mechanism depends on the number of TCP flows, average size of the TCP window, drop probability, marking probability, roundtrip link delay, and link rates [11]. The RED model may be used to correctly select the RED parameters. Even though measuring model parameters in real-

time may be difficult, the model may be used to evaluate the network performance.

Simulation approaches are popular due the existing detailed implementation of protocols in network simulators. A key benefit of the simulation approach is the possibility to test protocols during their development phase. The expected benefits of the RED mechanism have been demonstrated through simulations [1], [3]. Nevertheless, in the case of the basic RED algorithm, lack of fairness and the bias toward flows with shorter RTT intervals have been observed [9], [12]. Simulation of DropTail, RED, and RED with ECN mechanisms in a simple network topology with one server, one bottleneck link, and several client nodes showed that ECN may improve fairness and reduce packet drop rate [12]. Since AQM was mainly designed to prevent congestion, its occasional lack of fairness could be eliminated by employing packet scheduling. Fair scheduling algorithms increase the number of observed traffic state variables. Most scheduling algorithms are derived from the fair queuing algorithm [13]. Simulation showed that inappropriately configured RED behaves as DropTail [8]. Oscillations may occur in a RED queue, even when  $avg$  is maintained in the  $(min_{th} - max_{th})$  RED area. These observations motivated the development of adaptive RED [3].

The main criticisms of the RED mechanism emanated from experiments in laboratory test-beds. RED is inefficient when small buffers are used [14]. The performance of RED is very sensitive to the parameter selection [15] and RED may be difficult to configure in a multi-router environment. It was also noted that DropTail penalizes non-responsive traffic more than RED [14]. Evaluation of the RED performance with HTTP transfers from the end-user perspective showed that basic RED algorithm has no advantages over DropTail until the used capacity of the bottleneck link reaches 90% [15]. Even then, choosing the RED parameters that provided the best link utilization resulted in inferior response times. It was also observed that RED parameter  $max_p$  is highly sensitive to the number of active connections supported by a queue.

The growing number of Internet applications that do not rely on TCP could diminish positive effects of AQM. The IETF is currently working on several frameworks to preserve TCP and AQM congestion control in the presence of new services. Proposals include TCP friendly rate control [16] and the development of adaptive multimedia codecs [17].

## V. PERFORMANCE MEASURES

In this paper, we compare the two improved variants of RED with DropTail and address fairness, drop rate, throughput, and the RED queue size.

We use Jain's fairness index [18] and cumulative diagram of TCP acknowledgements. The fairness index  $FI$  represents data exchange fairness at a specific time instance:

$$FI = \frac{\left( \sum_{i=1}^n x_i \right)^2}{n \sum_{i=1}^n x_i^2}, \quad (2)$$

where  $x_i$  is a measure of exchange rate (throughput or goodput). Optimum fairness is indicated by  $FI = 1$  and it decreases as  $FI$  approaches 0.

A cumulative diagram of TCP acknowledgements is a temporal representation of ACKs for all flows passing through a link. At a chosen time instance, the width of the diagram is a fairness measure: the wider the diagram the poorer the fairness. This diagram may also indicate forming of privileged flow groups that can be identified by the separation between its members and the remaining traffic flows. The diagram also shows fluctuations of fairness with time.

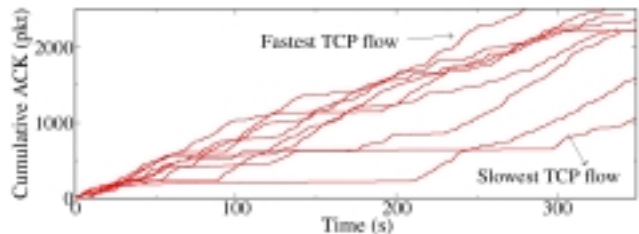


Fig. 7. Cumulative diagram of the TCP acknowledgements.

Packet drops are measured by the drop rate and drop percentage. The drop percentage is calculated as the ratio of dropped packets and the total number of packets on the link ingress. Data exchange rate is measured as the goodput efficiency and throughput. We define goodput efficiency as a sum of all goodputs divided by the bandwidth of the bottleneck link.

## VI. SIMULATION SCENARIOS

We simulate DropTail, gentle RED, adaptive RED, and the ECN upgrade to RED mechanisms and followed recommendations for simulation setup [19]. We use ns-2 [20] as a simulation platform. Ns-2 is open source software written in C++ and OTcl. It is widely used in research community. It offers numerous configurable parameters for network protocols, topology elements, traffic generators, and packet delay and packet drop generators.

Infinite FTP sources are used in simulation scenarios. The background traffic is generated by a constant bit rate (CBR) source using UDP transport. The rate of the CBR traffic is 10% of the bottleneck bandwidth. Even though the Internet traffic is more complex, we simulated the worst case scenario for comparison between the modified RED mechanisms and DropTail. Less aggressive traffic sources would have generated less congestion and, therefore, AQM effects could not be fully observed. To simulate equivalent levels of congestion with traffic sources such as HTTP, additional traffic generators would be required and simulation results would be more difficult to process.

TCP NewReno, the improved versions of TCP Reno, was used as the transport protocol. Most simulation scenarios employed the equal number of TCP NewReno and TCP SACK connections. Since SACK is already rather robust, we upgraded the NewReno connections with ECN in several

simulation scenarios. This enabled us to observe the benefits of ECN on both transport and network layers. Transport layer generated 1,460-byte segments, which is common in the current Internet [17].

We deployed DropTail, gentle RED, adaptive RED, and combinations of gentle RED with ECN and adaptive RED with ECN in network midpoints. Configuration of RED threshold parameters ( $min_{th}$  and  $max_{th}$ ) were not optimized and the default values  $max_{th} = 3 \times min_{th}$ , and  $min_{th} = 0.25 \times buffer\ capacity$  [2] was used. The assumption is that most network engineers would not have resources to adjust RED parameters to traffic characteristics and would, instead, use the recommended values. This choice of the threshold values also gives an advantage to DropTail over RED.

We monitor the behavior of the queue management mechanisms with respect to the *bandwidth-delay* product. We simulated networks where the *buffer capacity* is lower than the *bandwidth-delay* product. Tests were also performed with scenarios where the *buffer capacity* is larger than the *bandwidth-delay* and where the congestion management protocols dominate the congestion control.

The two topologies used in simulations are shown in Fig. 8. Traffic sources are connected to nodes  $s_i$  ( $i = 0, 1, 2, \dots$ ). Destinations are nodes  $d_j$  ( $j = 0, 1, 2, \dots$ ). Queue disciplines are employed in routers  $r_k$  ( $k = 0, 1, 2$ ). Delay is set to 40 ms, which is an average value for link delay in the Internet.

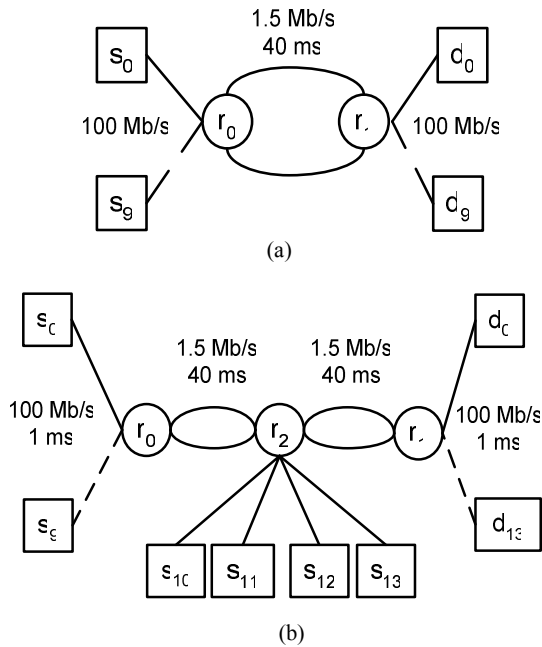


Fig. 8. Two test network topologies: (a) net10 and (b) netMultiC.

In the simulation scenario with the net10 topology, shown in Fig. 8(a), we vary the number of connections and the *buffer capacity*. In the simulation scenario with the netMultiC topology, shown in Fig. 8(b), the number of connections is constant while the *buffer capacity* varies. The interaction between the AQM mechanisms and DropTail is observed by varying the order and the type of queue disciplines employed on links  $r0 - r2$  and  $r2 - r1$ .

## VII. SIMULATION OF THE NET10 TOPOLOGY

### A. Two TCP flows

In this simulation scenario, two TCP flows are connected to infinite FTP traffic generators. Simulation results for the goodput efficiency and drop percentage are shown in Fig. 9. Neither RED mechanisms nor DropTail perform optimally when *buffer capacity* is under the *bandwidth-delay* threshold. The difference between RED and DropTail was most noticeable for  $buffer\ capacity = 4$  pkt when all RED variants have identical performance.

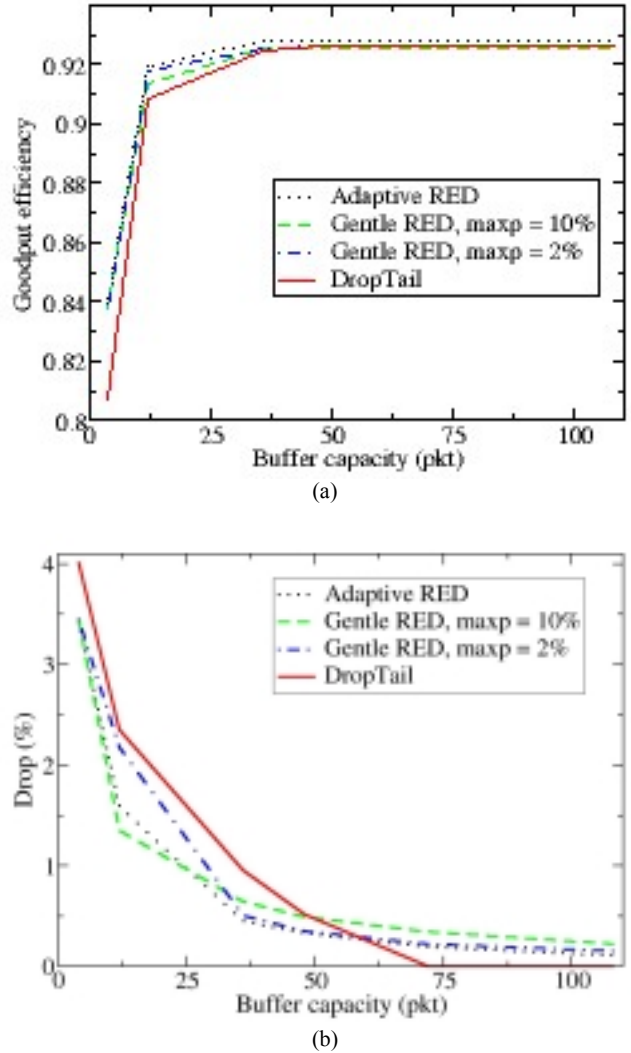


Fig. 9. Topology net10 with two TCP flows and one UDP flow: (a) goodput efficiency and (b) drop percentage. Labels:  $max_p = 10\%$  and  $max_p = 2\%$  denote the maximum drop probability parameter for gentle RED.

Events in the RED buffer are shown in Fig. 10. Simulation results indicate that while the parameter *avg* remains within the  $(min_{th} - max_{th})$  RED area, the instantaneous queue length frequently leaves the  $(min_{th} - max_{th})$  RED area. The queue

length oscillates with high amplitude of oscillations. Nevertheless, the early packet drops provide adequate feedback to the TCP congestion control, which explains the superior performance of the RED mechanisms.

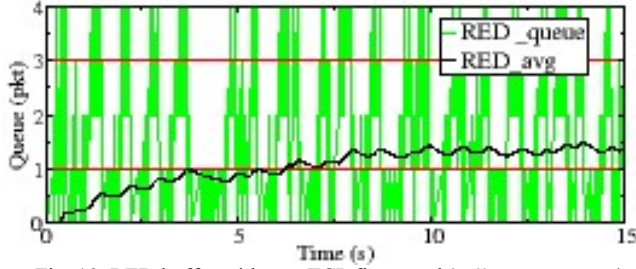


Fig. 10. RED buffer with two TCP flows and  $buffer\ capacity = 4$  pkt. Labels: RED\_queue denotes the instantaneous queue length and RED\_avg denotes the  $avg$  parameter.

Simulations lasted approximately 1,000 RTT intervals with two simultaneous TCP connections. Multiple packet drops were frequent due to the insufficient  $buffer\ capacity$ . We also recorded variations of TCP congestion window. Neither RED mechanisms nor DropTail exhibited signs of global synchronization [6]. Hence, synchronization effects were avoided with fast retransmit and fast recovery states of the TCP Reno implementations.

Increasing the  $buffer\ capacity$  above the  $bandwidth\text{-}delay$  threshold quickly improves both goodput and the packet loss when the capacity of the RED buffer was set to 36 packets. The gentle RED mechanism maintained the  $avg$  parameter in the  $(min_{th} - max_{th})$  RED area near  $min_{th}$  threshold. Nevertheless, the instantaneous queue length frequently decreased below  $min_{th}$ . Simulation results for the adaptive RED mechanism are shown in Fig. 11. The  $avg$  value remained within the  $(min_{th} - max_{th})$  RED area. The queue length remained within the  $(min_{th} - max_{th})$  RED area until  $t = 60$  s, when one TCP flow was penalized and a lockout occurred. Analysis revealed that the RED averaging coefficient  $w_q$  was too sensitive to the variations of the queue length. Such behavior is a consequence of a small number of TCP flows.

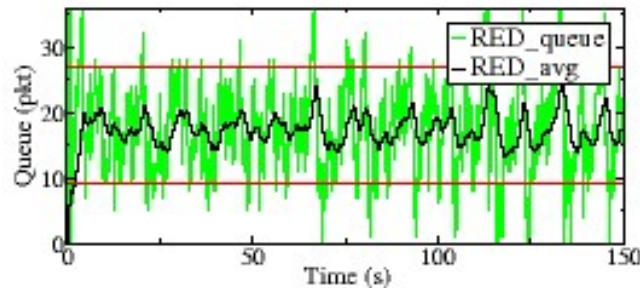


Fig. 11. Adaptive RED buffer with two TCP flows and  $buffer\ capacity = 36$  pkt. Labels: RED\_queue denotes the instantaneous queue length and RED\_avg denotes the  $avg$  parameter.

We also measured the impact of RED on fairness. The reference was a setup when DropTail achieved almost ideal fairness. We then replaced DropTail with gentle RED and

adaptive RED and measured the number of successfully exchanged TCP segments. Finally, we also upgraded RED with the ECN mechanism. RED could not maintain optimal fairness when number of connections was small. The ECN enhancement helped improve the fairness. The simulation results are shown in Table I.

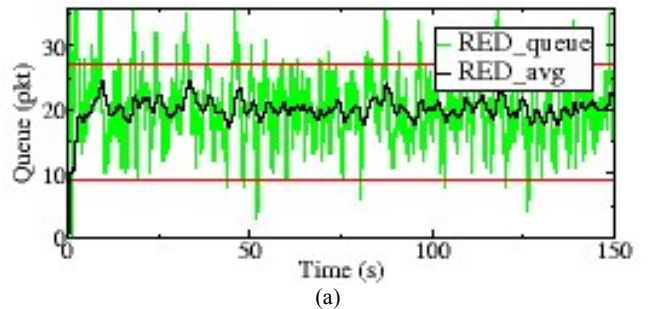
TABLE I  
IMPACT OF RED VARIANTS ON FAIRNESS IN TESTS WITH TWO TCP FLOWS

Total bytes exchanged	DropTail	Gentle RED ( $max_p = 10\%$ )	Adaptive RED
TCP flow f1	12,883,040	13,550,260	11,877,100
TCP flow f2	12,831,940	12,090,260	13,865,620
Flow f1 - f2	51,100	1,460,000	-1,988,520
With ECN			
TCP flow f1	12,883,040	12804200	12198300
TCP flow f2	12,831,940	12874280	13561940
Flow f1 - f2	51,100	-70,080	-1,363,640

### B. Simulations with six TCP flows

A moderate increase in the number of active TCP connections increased the goodput efficiency in cases of insufficient buffers. Percentage of dropped packets doubled.

Queue size in RED buffers is shown in Fig. 12. In the case of gentle RED shown in Fig. 12(a), the parameter  $avg$  doubled while remaining within the  $(min_{th} - max_{th})$  RED area. The queue length also mostly remained in the  $(min_{th} - max_{th})$  RED area. Underflows were less frequent, while the number of overflows increased. The adaptive RED mechanism shown in Fig. 12(b) exhibited fewer variations and maintained the  $avg$  parameter in the middle of the  $(min_{th} - max_{th})$  RED area. Variations of  $avg$  were smaller in comparison to tests with two TCP flows. The number of queue length overflows and underflows from the  $(min_{th} - max_{th})$  RED area was reduced. The ability of adaptive RED to keep the  $avg$  parameter in the  $(min_{th} - max_{th})$  RED area may be of benefit to applications sensitive to packet delay variations.



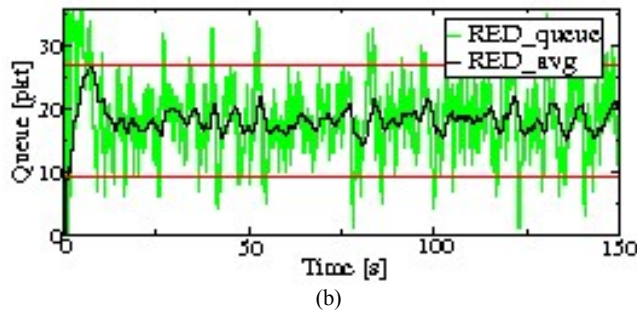


Fig. 12. RED buffer with six TCP flows and  $buffer\ capacity = 36$  pkt: (a) gentle RED and (b) adaptive RED. Labels: RED\_queue denotes the instantaneous queue length and RED\_avg denotes the avg parameter.

The fairness index is shown in Fig. 13. The RED mechanisms performed better than DropTail when  $buffer\ capacity$  was inadequate. This difference in fairness decreased as the  $buffer\ capacity$  increased. The fairness index of DropTail decreased for large  $buffer\ capacity$ . Gentle RED showed the best overall fairness index. The fairness performance of the adaptive RED was often worse than DropTail. However, it exhibited small variations and continuously increased with the increase of the  $buffer\ capacity$ . While the main function of adaptive RED is adjusting the avg parameter in the middle of the ( $min_{th} - max_{th}$ ) RED area, this did not lead to the optimal fairness or the optimal drop rate [21].

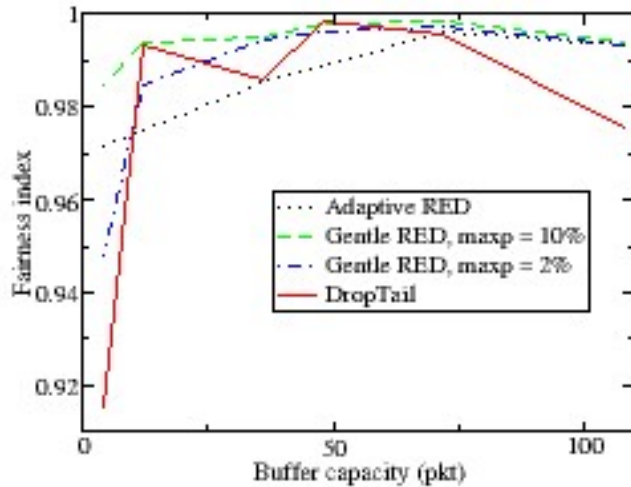


Fig. 13. Fairness index in the case of six TCP connections. Labels:  $max_p = 10\%$  and  $max_p = 2\%$  denote the maximum drop probability parameter for gentle RED.

### C. Simulations with eighteen TCP flows

We simulated heavily congested network with eighteen FTP connections. In comparison to previous test scenarios, goodput and drop performances changed significantly. Simulation results are shown in Fig. 14. There is no visible

trend in performance as the  $buffer\ capacity$  increased. The RED mechanisms show no advantage compared to DropTail.

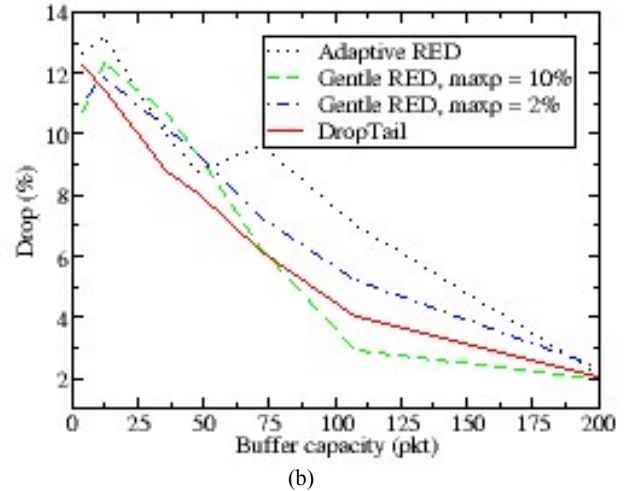
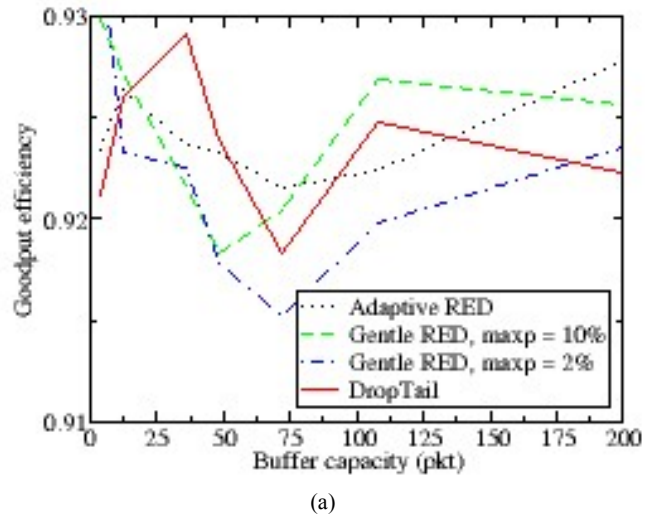


Fig. 14. Topology net10 with eighteen TCP flows and one UDP flow: (a) goodput efficiency and (b) drop percentage. Labels:  $max_p = 10\%$  and  $max_p = 2\%$  denote the maximum drop probability parameter for gentle RED.

The RED  $buffer\ capacity$  was set to 36 pkt. The RED mechanisms could not maintain the avg parameter within the ( $min_{th} - max_{th}$ ) RED area. While the performance of gentle RED was acceptable due to probabilistic dropping of packets when queue size exceeded  $max_{th}$ , the adaptive RED performed poorly. The adaptive RED case is shown in Fig. 15. These results illustrate that both the number of connections and the traffic volume influence the performance of a queue discipline.

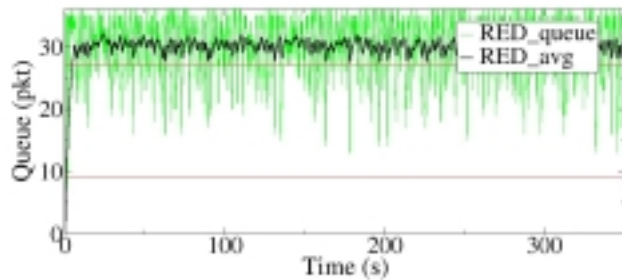


Fig. 15. Adaptive RED buffer with eighteen TCP flows and *buffer capacity* = 36pkt. Labels: RED\_queue denotes the instantaneous queue length and RED\_avg denotes the *avg* parameter.

In order to analyze excessive packet drop of the adaptive RED for *buffer capacity* = 72 pkt, we increased the simulation time and used smaller increment steps for *buffer capacity* in the range [36, 108] pkt. Results shown in Fig. 16 indicate that packet drop rate increased due to the adaptive RED mechanism. A finite time was needed to adjust the *avg* parameter [3]. At *buffer capacity* = 36 pkt, the adaptive RED could not adjust the *avg* parameter in the ( $min_{th} - max_{th}$ ) RED area. When *buffer capacity* was increased to 48 pkt, the adjustment was completed after 800 s. Finally, for the *buffer capacity* = 72 pkt, the adjustment was completed within 40 s. In the case of small buffer capacities when the *avg* parameter was in the middle of the ( $min_{th} - max_{th}$ ) RED area, the packet drop rate of the adaptive RED was greater than the drop rate of gentle RED.

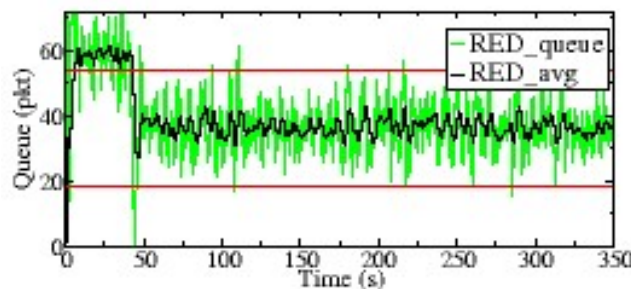


Fig. 16. Adaptive RED buffer with eighteen TCP flows and *buffer capacity* = 72 pkt. Labels: RED\_queue denotes the instantaneous queue length and RED\_avg denotes the *avg* parameter.

The fairness index is shown in Fig. 17. The scenario with eighteen TCP flows exhibits smaller fairness index than the setup with six TCP flows when *buffer capacity* is relatively low ( $\leq 36$  pkt). Adaptive RED algorithm does not always exhibit the best performance, but its variations in fairness are the smallest. In most cases, DropTail showed lower fairness than RED mechanisms.

We also evaluated the fairness in the presence of ECN mechanism. The connections were equally divided between TCP NewReno and TCP SACK. The ECN support was given to NewReno, being the less robust protocol implementation. We used the scenario with inadequate *buffer capacity* (12 pkt), to emphasize the effect of ECN. Simulation results show that TCP NewReno without ECN exchanged fewer packets than SACK. The low fairness index of TCP NewReno

connections is indicated in Fig. 18(a). With ECN, the fairness of NewReno connections improved and their goodput increased, as shown in Fig. 18(b). Packet drop rate of the TCP NewReno connections notably decreased. However, the presence of ECN significantly reduced the goodput of SACK connections and indicated ECN's negative effect on fairness.

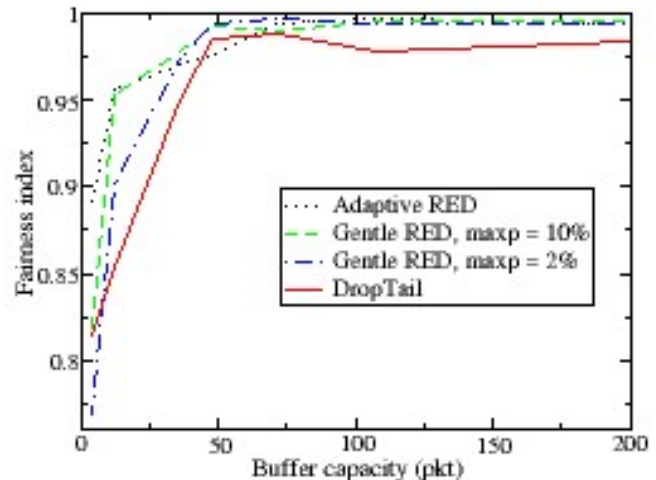


Fig. 17. Fairness index in the case of eighteen TCP connections. Labels:  $max_p = 10\%$  and  $max_p = 2\%$  denote the maximum drop probability parameter for gentle RED.

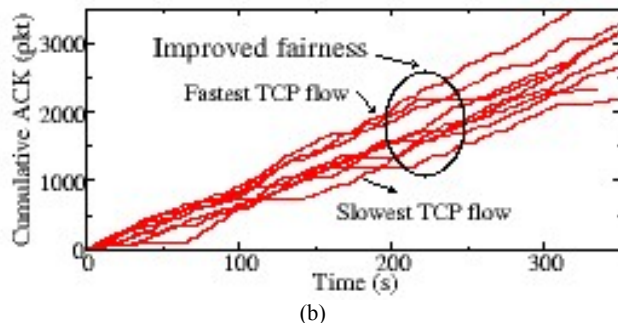
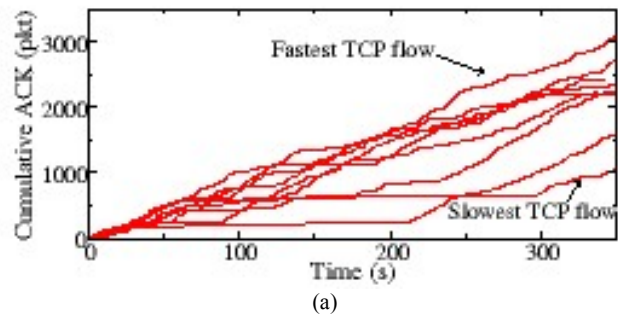


Figure 18. Goodput and fairness in the case of nine TCP NewReno and nine TCP SACK flows. (a) NewReno without ECN and (b) NewReno with ECN support.

We also observed the impact of packet delay variations on fairness. The simulation setup was identical to the study of ECN influence on fairness. Paths of NewReno+ECN and

SACK connections have different packet delays. Simulation results shown in Fig. 19 illustrate the unpredictable behavior of DropTail even in the case of small delay difference and adequate buffers. The gentle and adaptive RED mechanisms were less affected by the change in packet delays.

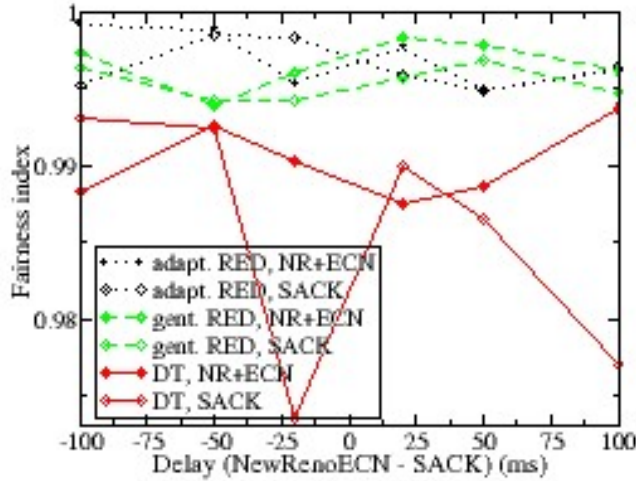


Fig. 19. Fairness in presence of delay difference between 9 TCP NewReno+ECN and 9 TCP SACK flows. Buffer capacity is 200 pkt, delay(NR-SACK) = -100 ms denotes that SACK connections are delayed 100 ms.

### VIII. SIMULATIONS OF NETMULTIC TOPOLOGY

We also simulated interactions between AQM mechanisms and DropTail in a network with *netMultiC* topology shown in Fig. 8 (b) [14]. The inter-router links  $r0-r2$  and  $r2-r1$  were the network bottlenecks. There were 27 TCP flows. Nine NewReno+ECN and nine SACK connections traversed both bottleneck links. The background traffic was provided with nine TCP NewReno connections exchanging data through the bottleneck link  $r2-r1$ . Simulation tests were performed with two buffer capacities (72 pkt and 300 pkt) and various choices of the queue disciplines deployed in the bottleneck links: RED-DT denotes that AQM was configured on link  $r0-r2$  and DropTail was used on link  $r2-r1$ . We tested RED-DT, DT-RED, and DT-DT configurations.

Simulation results of the total goodput are shown in Table II. The goodput of “inter node” TCP NewReno connections depends on the queue discipline deployed in router  $r2$ . With the DropTail configured in router  $r2$ , goodput of the “inter node” flows was extremely low. This was unexpected because DropTail has been shown to be biased towards flows with shorter RTT intervals. The cumulative acknowledgements diagram revealed that this effect was due to the number of the “end node” NewReno+ECN and SACK flows. The “inter node” flows started aggressively, thus halting the “end node” flows. This aggressive start flooded buffers in  $r2$  router and, eventually, halted the “inter node” connections. The “end node” connections eventually recovered and gained most bandwidth. The number of the “end node” flows and their superior congestion control mechanisms made them dominant throughout the remaining test interval. When RED variants

were deployed in router  $r2$ , fairness between groups of flows significantly improved. Increase in buffer capacity to 200 pkt improved the distribution of goodput in all simulation scenarios.

TABLE II

TOTAL GOODPUT IN THE NETMULTIC TOPOLOGY. THE “END NODES” REFER TO NEWRENO WITH ECN AND SACK FLOWS. THE “INTER NODES” REFER TO NEWRENO CONNECTIONS.

Total goodput (Mb/s)		Buffer capacity 72 pkt		Buffer capacity 300 pkt	
		RED 10%	Adapt. RED	RED 10%	Adapt. RED
RED-DT	end nodes	1.367	1.361	1.158	1.158
	inter. nodes	0.039	0.043	0.222	0.222
DT-RED	end nodes	0.854	0.883	0.899	0.915
	inter. nodes	0.496	0.466	0.488	0.469
DT-DT	end nodes	1.374		1.158	
	inter. nodes	0.028		0.222	

Packet drop rates are shown in Table III. RED improves fairness when configured on router  $r2$ . RED increases the packet drop rate. It controls the traffic management, since DropTail in router  $r1$  does not drop any packets. Smaller packet drop rate with DropTail in  $r2$  router are not desirable because it causes an unfair distribution of available bandwidth. Increase of buffer size to 300 pkt reduces the packet drop rate. However, when RED mechanisms dominate traffic management, the adaptive RED drops more packets than the gentle RED.

TABLE III

TOTAL DROP RATE IN TESTS ON NETMULTIC TOPOLOGY. THE “END NODES” REFER TO NEWRENO+ECN AND SACK FLOWS. THE “INTER NODES” REFER TO NEWRENO CONNECTIONS.

Total drop rate (kb/s)		Buffer capacity 72 pkt		Buffer capacity 300 pkt	
		RED 10%	Adapt. RED	RED 10%	Adapt. RED
RED-DT	end $r0-r2$	13.920	15.326	0.000	0.000
	end $r2-r1$	21.600	22.663	17.074	17.074
	inter. nodes	17.040	21.051	17.143	17.143
DT-RED	end $r0-r2$	0.000	0.000	0.000	0.000
	end $r2-r1$	104.503	98.331	16.629	26.983
	inter. nodes	64.869	62.606	18.514	23.006
DT-DT	end $r0-r2$	24.000		0.000	
	end $r2-r1$	14.263		17.074	
	inter. nodes	11.863		17.143	

### IX. DISCUSSION AND CONCLUSION

In this paper, we compared improved variants of the RED mechanism with the DropTail queuing discipline. Simulation results confirm that both gentle RED and adaptive RED are sensitive to the choice of their parameters. This was observed in scenarios when the average queue length could not be kept within  $(min_{th} - max_{th})$  RED area. However, even when the

RED variants did not perform optimally, they often outperformed DropTail. The largest differences were observed in the case of small buffers with small or moderate number of flows. Several tests indicated that the performance of DropTail depends on a flow of events in the network and led to unexpected packet drops even when the network provided adequate resources. The RED mechanisms generally exhibited more consistent performance with the variation of number of connections, *buffer capacity*, and packet delay.

Unlike [14], our results dealing with interactions between AQM and DropTail take into account the background TCP traffic. We found that RED mechanisms may improve fairness in environments with mixed queue disciplines. The RED variants may also assume all traffic management functions when they may significantly increase the packet drop rate. Addition of ECN improved fairness between flows that supported ECN. However, the presence of ECN may reduce throughput of connections without ECN. ECN also reduces the overall packet drop rate. Simulations indicate that NewReno with ECN support may have advantage over SACK [21].

The implementation of AQM is beneficial in a general network environment. Nevertheless, it may not always improve network performance and AQM parameters need to be adjusted to traffic properties, number of active traffic flows, and physical network parameters. Simulation results indicated that AQM almost always results in a more consistent traffic behavior.

Future research direction may include the implementation of higher order queue length estimation statistics in the RED estimation and decision process. Simulations indicated that instantaneous queue length may often depart from the  $(min_{th} - max_{th})$  RED area even when the average queue length (*avg*) remains within it, which may be prevented by including the estimation of *avg* variation within the RED decision mechanism. Furthermore, deploying AQM support in network midpoints could enable them to dynamically negotiate the AQM parameters.

## REFERENCES

- [1] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. on Networking*, vol. 1, no. 4, pp. 397–413, August 1993.
- [2] S. Floyd. (March 2000). Recommendation on using the "gentle" variant of RED. [Online]. Available: <http://www.icir.org/floyd/red/gentle.html>.
- [3] S. Floyd, R. Gummadi, and S. Shenker. (August 2001). Adaptive RED: an algorithm for increasing the robustness of RED's active queue management. [Online]. Available: <http://www.icir.org/floyd/papers/adaptiveRed.pdf>.
- [4] K. Ramakrishnan, S. Floyd, and S. Black, "The addition of explicit congestion notification (ECN) to IP," *RFC 3168*, Sept. 2001.
- [5] D. Hucaby, *CCNP BCMSN Exam Certification Guide*, Cisco Press, 2003, pp. 422–430.
- [6] L. Zhang, S. Shenker, and D. D. Clark, "Observations of a congestion control algorithm: the effects of two way traffic," in *Proc. ACM SIGCOMM*, Zürich, Switzerland, Sept. 1991, pp. 123–147.
- [7] W. E. Leland, M. Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of Ethernet traffic (extended version)," *IEEE/ACM Trans. on Networking*, vol. 2, no. 1, pp. 1–15, Feb. 1994.
- [8] W. C. Feng, "Improving Internet congestion control and queue management algorithms," Ph.D. Thesis, University of Michigan, Ann Arbor, MI, 1999.
- [9] D. Lin and R. Morris, "Dynamics of random early detection," in *Proc. ACM SIGCOMM*, Cannes, France, Sept. 1997, pp. 127–137.
- [10] R. Mahajan and S. Floyd, "Controlling high-bandwidth flows at the congested router," ICSI Technical Report TR-01-001, Apr. 2001.
- [11] J. Chung and M. Claypool, "Analysis of active queue management," in *Proc 2nd IEEE International Symposium on Network Computing and Applications (NCA)*, Cambridge, MA, USA, Apr. 2003, pp. 359–366.
- [12] K. Pentikousis and H. Badr, "An evaluation of TCP with explicit congestion notification," *Annals of Telecommunications*, vol. 59, no. 1–2, 2004.
- [13] A. Demers, S. Keshav, and S. Shenker, "Analysis and simulation of a fair queuing algorithm," *J. Internetworking: Research and Experience*, vol. 1, no. 1, pp. 3–26, Oct. 1990.
- [14] M. May, J. Bolot, C. Diot, and B. Lyles, "Reasons not to deploy RED," in *Proc. 7th. Int. Workshop on Quality of Service (IWQoS'99)*, London, 1999, pp. 260–262.
- [15] M. Christiansen, K. Jeffay, D. Ott, and F. D. Smith, "Tuning RED for Web traffic," *IEEE/ACM Trans. on Networking*, vol. 9, no. 3, June 2001, pp. 249–264.
- [16] M. Handley, S. Floyd, J. Padhye, and J. Widmer, "TCP friendly rate control (TFRC): protocol specification," *RFC 3448*, Jan. 2003.
- [17] S. Floyd and J. Kempf, "IAB concerns regarding congestion control for voice traffic in the Internet," *RFC 3714*, Mar. 2004.
- [18] R. Jain, *The Art of Computer Systems Performance Analysis*. New York: Wiley, 1991.
- [19] S. Floyd and V. Paxson, "Difficulties in simulating the Internet," *IEEE/ACM Trans. on Networking*, vol. 9, no. 4, pp. 392–403, Aug. 2001.
- [20] Network simulator–ns-2. [Online]. Available: <http://www.isi.edu/nsnam/ns/>.
- [21] S. Mišković, "Simulation study of TCP congestion control mechanisms on the transport layer," M.Sc. Thesis, Faculty of Electrical Engineering, Belgrade University, Belgrade, Serbia and Montenegro, 2004.