

# IP Quality of Service: Building Blocks

Bahri Okuroğlu and Sema Oktuğ

Department of Computer Engineering, İstanbul Technical University, Maslak, İstanbul, Turkey

[okuroglu@itu.edu.tr](mailto:okuroglu@itu.edu.tr) , [oktug@itu.edu.tr](mailto:oktug@itu.edu.tr)

**Abstract:** RED (Random Early Detection) is the most popular active queue management algorithm, although it has some weaknesses. Recently, another active queue management algorithm, BLUE, was proposed and shown that it is more successful in controlling the queue length when high number of flows are active on ECN capable networks. In this report, RED and BLUE algorithms are studied and shown that BLUE on ECN incapable networks is not as successful as on ECN capable networks.

Differentiated Services architecture suggests that RIO (Red with In and Out) style queue management algorithms are to be used on each AF (Assured Forwarding) queue to offer different levels of services for different priorities at each AF class. Inspired of BLUE's success over RED on ECN capable networks, we developed a simple alternative to RIO, BIO (BLUE with In and Out). BIO, which runs two different BLUE algorithms for in and out packets, was expected to achieve lower loss rates while maximizing link utilization for high number of active flows on AF queues.

However, due to the self-configuring architecture of the algorithm, it is observed that BIO marks packets too aggressively and degrades utilization. In this paper, the properties of BIO are also explained and the results obtained are generalized to all self-configuring multi priority queue management algorithms.

**Keywords:** Active Queue Management, Congestion Control, TCP, Differentiated Services, RIO

## A. INTRODUCTION

The benefits of the stateless architecture of the IP (Internet Protocol) [1] have enabled the rapid growth of the Internet. With this enormous growth, network congestion, caused by the stateless architecture of IP has become more apparent. As the achievement of network efficiency and the reduction of the loss rate became major problems, new mechanisms are required to meet the expectations of today's applications since the architecture of the Internet is not designed to support these kinds of applications.

In this report we're studying works on these issues. We will mainly concern on the Quality of Service (QoS) mechanisms for IP networks, and

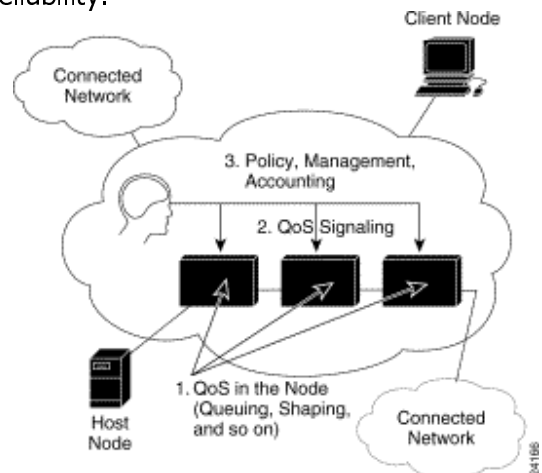
talk about the congestion control mechanisms from the IP QoS perspective.

## B. MARRIAGE OF TERMS QUALITY AND SERVICE

Quality can encompass many properties in networking, but people generally use quality to describe the process of delivering data in a reliable manner or even somehow in a manner better than normal. This method includes the aspect of data loss, minimal (or no) induced delay or latency, jitter, and the capability to determine the most efficient use of network resources.

People generally use service to describe something offered to the end-users of any network, such as end-to-end communications or client-server applications. Services can cover a broad range, from electronic mail to desktop video, from Web browsing to chat rooms.

QoS is the capability to differentiate between traffic or service types so that users can treat one or more classes of traffic differently than the other types. Service quality determines the characteristics of the offered service to the user and is measured in terms of delay, jitter and reliability.



**Figure 1 Generic end-to-end QoS architecture [2]**

To offer end-to-end QoS to users, three expectations should be met. First, gateways on the network should offer some kind of service differentiation to the data packets. Next,

signaling should be implemented to synchronize the routers' differentiation. Finally policy, management and accounting features should be supported.

Different alternatives on each protocol layer can be used to offer service differentiation on gateways. On physical level different links to the same destination, which is used as backup in normal, can be used for differentiation. High priority packets can be sent over the higher quality link. However, since the routing decision is based on the destination address, services offered to up and down traffic differ.

Another alternative is the data link layer protocols with native QoS support, such as ATM. But, there exist several problems with IP over ATM; such as the signaling overhead of ATM and inefficient segmentation of IP packets to ATM cells. About 20% of the link bandwidth is used for the padding bits and ATM header overhead [3].

To offer service differentiation at higher layers, different kinds of queuing and scheduling mechanisms should be used.

### C. INTEGRATED SERVICES

In 1994, IETF formed Integrated Services (IS) working group to develop a solution to the IP QoS issue. The architecture and two offered services, guaranteed [6] and controlled-load [7], beyond best-effort service are standardized in IETF. On IS architecture [4], end-nodes establish connection with the required QoS specification before transmitting packets similar to ATM. Source node sends an RSVP [5] packet destined to the destination. This PATH packet, is transmitted to the destination using the standard routing protocols. Destination sends back a RESV message with the QoS parameters it requests. RESV follows the reverse-path of the PATH messages and routers on the path reserves the required resources. Routers run a kind of admission control mechanism to decide to accept the flow to the network.

Guaranteed service, guarantees that datagrams will arrive within the guaranteed delivery time and will not be discarded due to queue overflows, provided the flow's traffic stays within its specified traffic parameters. This service is intended for applications, which need a

firm guarantee that a datagram will arrive no later than a certain time after it was transmitted by its source. For example, some audio and video "play-back" applications are intolerant of any datagram arriving after their playback time. Applications that have hard real-time requirements will also require guaranteed service.

The end-to-end behavior provided to an application by a series of network elements providing controlled-load service tightly approximates the behavior visible to applications receiving best-effort service *under unloaded conditions* from the same series of network elements.

However the IS architecture has inherent scalability problems because of the huge quantity of state information maintained by the gateways and the multi-field classification which gateways perform and is relatively an expensive process.

Also the new is architecture is completely different than the Internet's current stateless architecture. IS requires changes on both end-applications and networks.

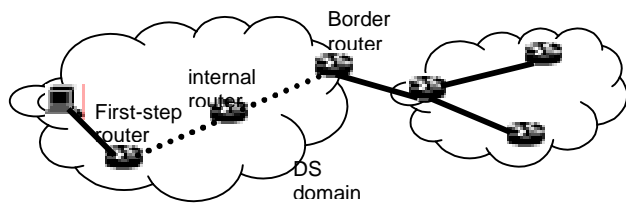
### D. DIFFERENTIATED SERVICES

Because of the problems in the IS architecture, Differentiated Services (DS) working group is formed in IETF. The main purpose of the DS architecture [8] is to develop a scalable solution to the IP QoS problem.

This architecture is based on DS domains, which is managed by a central authority. The gateways on the borders of this domain classify the packets and mark the DS field [9] of the packets to indicate the service level to be offered by the network. This decision is based on the Service Level Specification (SLS), which is agreed on by the neighbor network management. This classification is a multi-field (MF) classification, which means that multiple fields of the packets like source and IP addresses, ports, flowid are used in classification.

Internal nodes perform behavior aggregate (BA) classification on packets, which means that only the DS field of the packets is checked. Different forwarding treatments (PHB, Per Hop Behavior) are offered to the packets based on this classification. Most common method of service differentiation is using scheduling

algorithms for different number of queues for each output interface.



**Figure 2 DiffServ Architecture**

The forwarding behavior offered on a gateway is named as PHB, and currently two standardized PHBs by IETF.

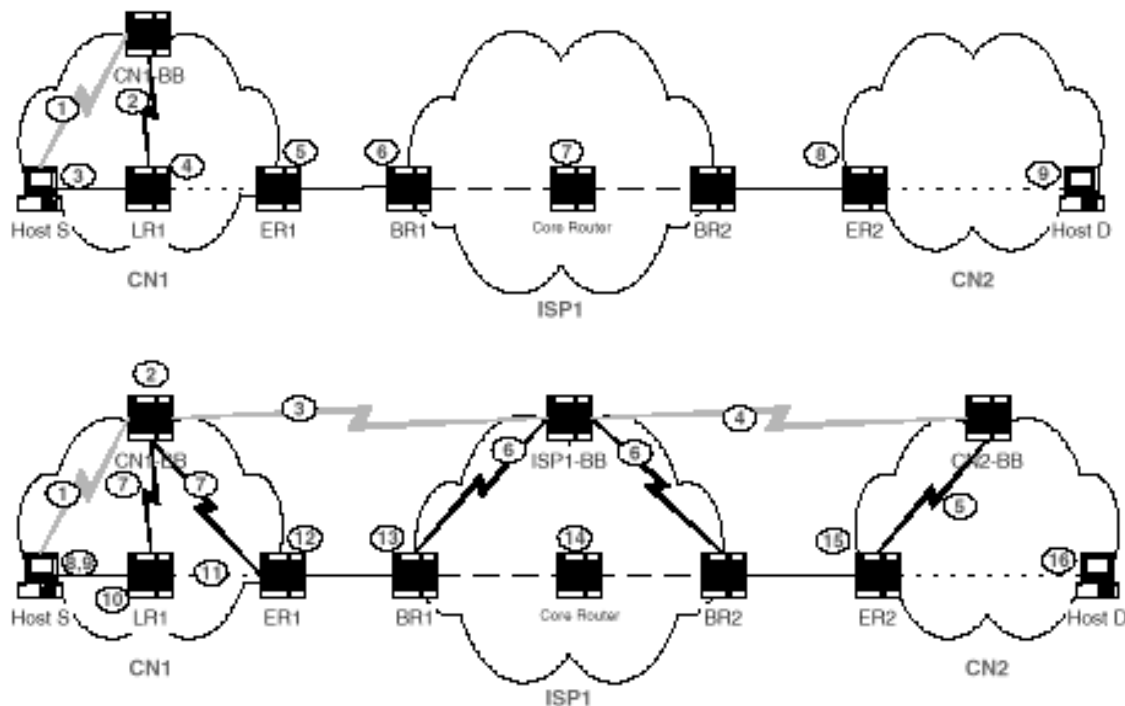
Assured Forwarding (AF) PHB [10] group is a means for a provider DS domain to offer different levels of forwarding assurances for IP packets received from a customer DS domain. Four AF classes are defined, where each AF class is in each DS node allocated a certain amount of forwarding resources (buffer space and bandwidth). Within each AF class IP packets are marked (again by the customer or the provider DS domain) with one of three possible drop precedence values. In case of congestion, the drop precedence of a packet determines the relative importance of the packet within the AF class. A congested DS node tries to protect packets with a lower drop precedence value from being lost by preferably discarding packets with a

higher drop precedence value. In a DS node, the level of forwarding assurance of an IP packet thus depends on (1) how much forwarding resources has been allocated to the AF class that the packet belongs to, (2) what is the current load of the AF class, and, in case of congestion within the class, (3) what is the drop precedence of the packet.

Expedited forwarding (EF) [11] can be used to build a low loss, low latency, low jitter, assured bandwidth, end-to-end service through DS domains. Such a service appears to the endpoints like a point-to-point connection or a "virtual leased line".

### *Resource Allocations in ISP Domains*

Given the SLs, ISPs must decide how to configure their boundary routers so that they know how to handle the incoming traffic. This process is called *Resource Allocation*. For static SLs, boundary routers can be manually configured with the classification, policing and shaping rules. Resources are therefore statically allocated for each customer. Unused resources can be shared by other customers. For a dynamic SL, resource allocation is closely related to the signaling process. The bandwidth broker (BB) in



**Figure 3 Service distribution in a DS domain**

the customer domain uses RSVP to request for resources from its ISP. At the ISP side, the admission control decisions can be made in a distributed manner by the boundary routers or by a BB. If boundary routers are directly involved in the signaling process, they are configured with the corresponding classification, policing and shaping rules when they grant a request. If a BB is involved rather than the boundary routers, then the BB must configure the boundary routers when it grants a request.

#### 1) *Delivery of AF Service with a static SLS*

1. Host S sends a RSVP message to the local Bandwidth Broker CN1-BB, requesting for AF service for its traffic.
2. If CN1-BB grants the request, it will configure leaf router LR1 so that LR1 can set the DS field of the packets of this flow with AF1, code. CN1-BB will then reply to host S. Otherwise, an error message is sent to Host S.
3. Host S sends packets to leaf router LR1.
4. If LR1 is configured to mark the traffic, it will set the DS field of the conforming packets with AF11 code. Nonconforming traffic will be marked with AF13 code to give lower priority.
5. Every router from LR1 (exclusive) to ER1 (inclusive) does a BA classification. Packets enter the AF1 queue. RIO is applied on the AF1 queue.
6. BR1 polices the traffic. All *out* traffic (AF13) remains *out*. If the *in* traffic (AF11) exceeds its bit-rate, the excess packets' DS field will be set to AF13 code. All packets enter the AF1 queue. RIO is applied on the queue.
7. All routers between boundary router BR1 and BR2 (inclusive) perform BA classifications and apply RIO on their AQs.
8. ER2 performs the same operations as BR1.
9. The packets are eventually delivered to host D.

#### 2) *Delivery of EF Service with a dynamic SLS*

1. Host S sends a RSVP PATH Message to the local Bandwidth Broker CN1-BB

2. CN1-BB makes an admission control decision.
  - If the request is denied, an error message is sent back to host S. The signaling process ends.
3. The request is accepted by CN1-BB. CN1-BB sends the PATH Message to ISP1-BB.
4. ISP1-BB makes an admission control decision.
  - If the request is denied, an error message is sent back to CN1-BB. Sender S will be notified.
  - If the request is accepted, ISP1-BB sends the PATH Message to CN2-BB.
5. CN2-BB makes an admission control decision.
  - If the request is denied, an error message is sent back to ISP1-BB. Sender S will be notified.
  - If the request is accepted, CN2-BB will use LDAP or RSVP to set the classification and policing rules on router ER2. CN2-BB will then send an RSVP RESV Message to ISP1-BB.
6. When ISP1-BB receives the RESV Message, it will configure the classification and policing rules on router BR1, and the policing and reshaping rules on router BR2. It will then send the RESV Message to CN1-BB.
7. When CN1-BB receives the RESV Message, it will set the classification and shaping rules on router LR1, so that if the traffic of the admitted flow is non-conformant, LR1 will shape it. CN1-BB will also set the policing and reshaping rules on router ER1. CN1-BB will then send the RESV Message to host S.
8. When host S receives the RESV Message, it can start transmitting data.
9. Host S sends packets to leaf router LR1.
10. Leaf router LR1 performs a MF classification. If the traffic is non-conformant, LR1 will shape it. LR1 will also set the DS field of the packets to EF code. All packets enter the EF queue.
11. Each intermediate router between leaf router LR1 and ER1 performs a BA classification, puts the packets into the EF queue, and sends them out.
12. ER1 performs a BA classification and reshapes the traffic to make sure that the

negotiated peak rate is not exceeded. Reshaping is done for the aggregation of all flows heading toward BR1, not for each individual flow.

13. BR1 classifies and polices the EF traffic. Excess EF packets are dropped.
14. Intermediate routers between leaf router BR1 and BR2 (inclusive) perform BA classifications. BR2 also reshapes the EF traffic.
15. ER2 classifies and polices the EF traffic. Excess EF packets are dropped.
16. The EF packets are delivered to host D.

#### E. ACTIVE QUEUE MANAGEMENT MECHANISMS

TCP [12] tries to offer ordered and reliable transmission over an unreliable network (IP). It uses sequence numbers (SEQ) to prevent reordering, and packet acknowledgements (ACK) to offer a reliable transmission. Waiting ACK of each packet before sending another prevents from exploiting the available bandwidth. To overcome this limitation, window based transmission mechanism is employed. In this architecture the TCP agent maintains two different windows. While the receiver window (rwnd) is defined by the information extracted from the ACK packets, there is no way of getting information from the network to define the congestion window (cwnd). Rwnd defines the amount of data the receiver can accept, and cwnd defines the amount of data the network can accept. TCP congestion control mechanisms are employed in order to define the value of the cwnd. [22]

Active queue management mechanisms are congestion control algorithms, which are run on the gateways to detect congestion earlier and to send implicit or explicit feedback to the end-nodes. Due to the advantages, usage of the active queue management architectures on the gateways is recommended by IETF [13].

These algorithms have many advantages over classical drop-tail queues. Although the main advantage is the detection of the congestion earlier and reduction in the drop rate, these mechanisms also avoid the global synchronization and bias against bursty flows.

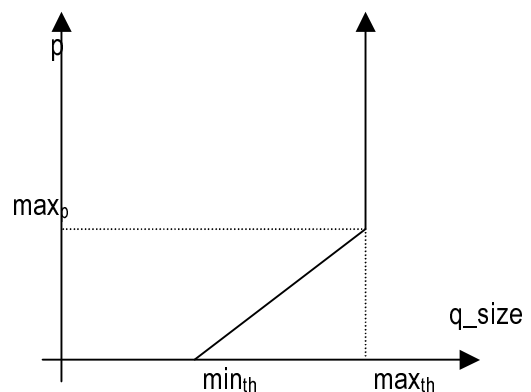
Active queue management algorithms either drop or just mark packets in order to control the

queue length. When packets are dropped to control congestion, standard TCP congestion control algorithms are efficient for the end-nodes. For the marking case, ECN-capable TCP must be employed on end-nodes. To prevent high loss rates caused by the packet drops to control the queue length, IETF is considering deployment of explicit congestion notification (ECN) [14].

By the use of ECN, gateways can mark packets instead of dropping them, and end nodes can take the proper action to decrease the size of the congestion window which indicates how much data the network can handle and is managed by the end node TCP.

#### RED

The RED [15] gateway calculates the average queue size, using a low-pass filter with an exponential weighted moving average. The average queue size is compared to two thresholds, a *minimum* threshold and a *maximum* threshold. When the average queue size is less than the minimum threshold, no packets are marked. When the average queue size is greater than the maximum threshold, every arriving packet is marked. If marked packets are in fact dropped, or if all source nodes are cooperative, this ensures that the average queue size does not significantly exceed the maximum threshold.



**Figure 4 RED parameters**

When the average queue size is between the minimum and the maximum threshold, each arriving packet is marked with probability  $p_a$ , where  $p_a$  is a function of the average queue size  $avg$ . Each time that a packet is marked, the

probability that a packet is marked from a particular connection is roughly proportional to that connection's share of the bandwidth at the gateway. The general RED gateway algorithm is given in Figure 5.

Thus the RED gateway has two separate algorithms. The algorithm for computing the average queue size determines the degree of burstiness that will be allowed in the gateway queue. The algorithm for calculating the packet-marking probability determines how frequently the gateway marks packets, given the current level of congestion. The goal is for the gateway to mark packets at fairly evenly-spaced intervals, in order to avoid biases and to avoid global synchronization, and to mark packets sufficiently frequently to control the average queue size.

```

for each packet arrival
    calculate the average queue size  $avg$ 
    if  $min_{th} - avg < max_{th}$ 
        calculate probability  $p_a$ 
        with probability  $p_a$ :
            mark the arriving packet
    else if  $max_{th} - avg$ 
        mark the arriving packet

 $avg = (1 - wq) * avg + wq * q$ 
 $p_b \leftarrow \max_p(avg - min_{th}) = (max_{th} - min_{th})$ 
 $p_a \leftarrow p_b / (1 - count - p_b)$ 

```

**Figure 5 RED algorithm**

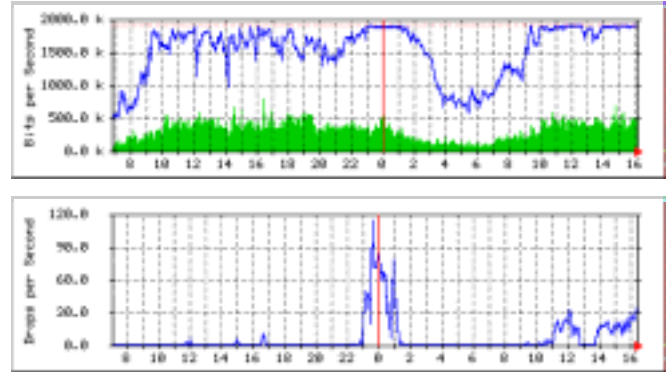
To send congestion indication to the end nodes packets can be both marked with ECN or dropped. While packet-dropping rate can be reduced with the use of ECN, dropping packets ensures that proper action is taken to the congestion even without the cooperating end nodes with adaptive transmission protocols like TCP.

RED's primary design goal is to avoid congestion by controlling the average queue length. Additional goals are avoiding global synchronization and bias against bursty traffic.

The effectiveness of the RED algorithm on congestion control and reduction of the loss rate is proven by both simulations and real-world experiences [23, 16].

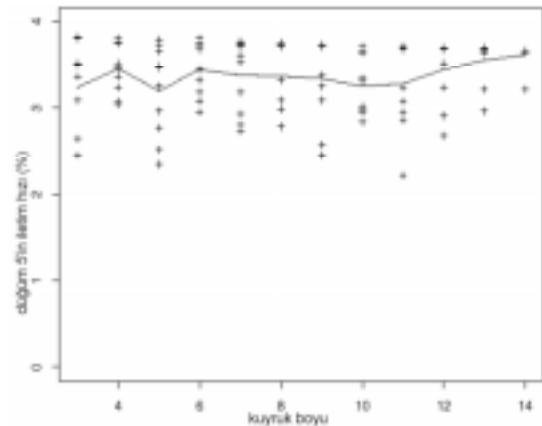
Figure 6 shows the usage of the RED algorithm on a link. In this experiment, RED is enabled on the link at Friday 10. Figures show that RED

achieves lower loss rates while it is keeping the link utilization about 100%.

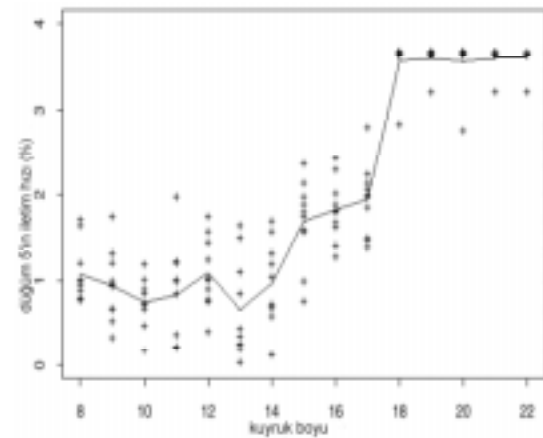


**Figure 6 RED performance [16]**

Figure 7 and Figure 8 show the throughput of the bursty flow in percentage to the total throughput between the smooth flows. Here RED is shown that ensuring the fairness to the bursty flows such as TCP traffic.



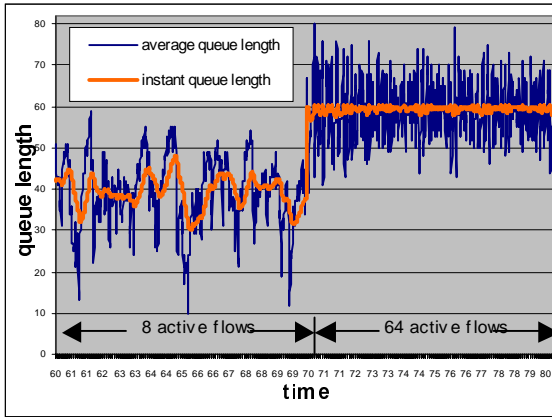
**Figure 7 Bursty-flow share in bw usage (RED)**



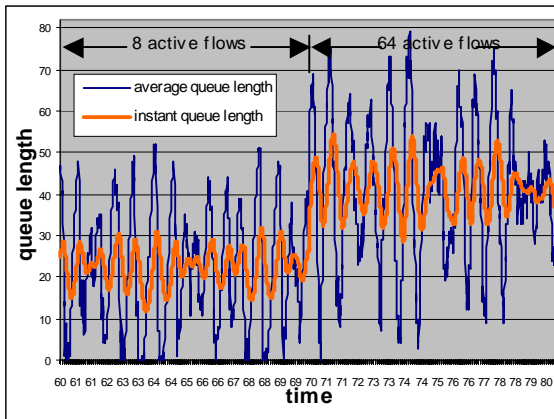
**Figure 8 Bursty-flow share in bw usage (Drop-tail)**

However, the algorithm has parameter selection problems [24, 25]. In order to benefit from the algorithm its parameters must be arranged properly.

Moreover, the marking aggressiveness of the algorithm is insensitive to the number of active flows on the gateway. When the bottleneck link is shared equally between the active flows, marking one packet to send congestion feedback decreases the total throughput by the rate of  $1-1/(2N)$  [17]. Since only the average queue length and  $\max_p$  parameter are used in calculation of RED's packet marking probability, RED cannot mark packets proportionally to the number of active flows. This may result in the loss of queue control.



**Figure 9 8 and 64 active flows with conservative RED configuration**



**Figure 10 8 and 64 active flows with aggressive RED configuration**

Simulations performed with different number of active flows and different RED configurations shows RED's problem. When the marking strategy is conservative, RED behaves well for 8 active flows over the bottleneck link. For 64

active flows over the same configuration this parameter setting can result the loss of the queue control and queue overflows.

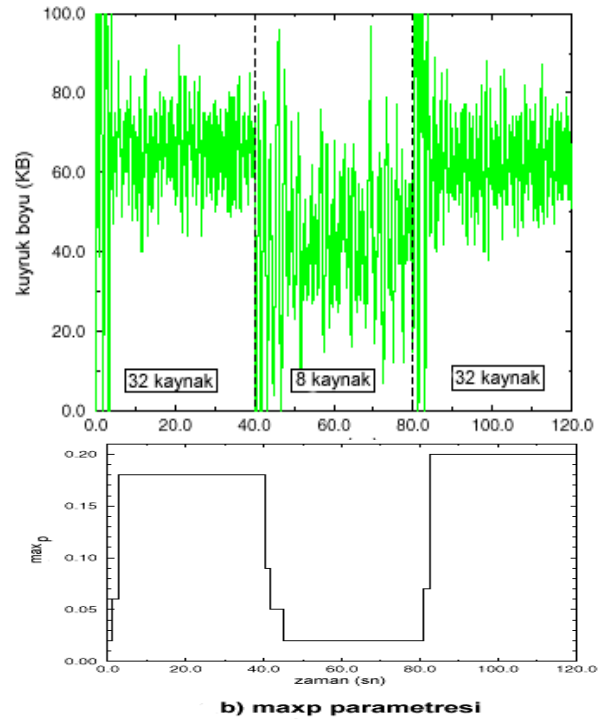
```

Every Q(ave) Update:
  if (minth < Q(ave) < maxth)
    status = Between ;
  if (Q(ave) < minth && status != Below)
    status = Below ;
    maxp = maxp /  $\alpha$ ;
  if (Q(ave) > maxth && status != Above)
    status = Above ;
    maxp = maxp *  $\beta$ ;

```

**Figure 11 ARED algorithm**

When the marking probability is increased to keep the queue in control, RED is successful for 64 active flows. When the flow count is decreased to 8, this aggressive configuration results underutilization of the queue.



**Figure 12 ARED performance**

Figure 11 represents an on-line algorithm, ARED [17], for adaptively changing the red parameters according to the observed traffic. The idea behind this algorithm is to infer whether or not red should become more or less aggressive by examining the variations in average queue length. If the average queue length oscillates around minth, then the early detection

mechanism is being too aggressive. If on the other hand, it oscillates around  $max_{th}$ , then the early detection mechanism is being too conservative. Based on the observed queue length dynamics, the algorithm adjusts value of  $max_p$ . In particular, it scales  $max_p$  by constant factors of  $\alpha$  and  $\beta$  depending on which threshold it crosses.

In Figure 12, it is shown that the ARED algorithm can successfully configure according to the traffic characteristics on the gateway.

### BLUE

BLUE [18] is a recently developed active queue management mechanism which has a completely different marking strategy compared to RED. BLUE assumes that queue length does not directly reflect the congestion level. Hence it does not update the packet marking probability with the queue length. Instead it uses queue overflow and idle event history to update the packet marking probability ( $p_m$ ). Packet loss due to the queue overflow means that the marking is not aggressive enough and  $p_m$  should be increased. Similarly, the queue idle event occurs as a result of the aggressive marking policy therefore the  $p_m$  parameter should be decreased. This mechanism effectively allows BLUE to *learn* the correct rate it needs to send back congestion notification.

```

For each packet loss:
    if ((now - last_update) > freeze_time )
         $p_m = p_m + d_i$ ;
        last_update = now;
For link idle event:
    if ((now - last_update) > freeze_time )
         $p_m = p_m + d_d$ ;
        last_update = now;

```

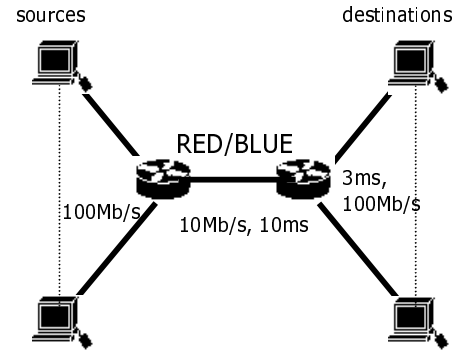
**Figure 13 The BLUE Algorithm**

BLUE uses three parameters: The first two parameters determine the amount by which  $p_m$  is incremented in case of the queue overflows ( $d_i$ ) or is decremented when the link is idle ( $d_d$ ). The last parameter is the minimum time interval between two successive updates ( $freeze\_time$ ).

Performance of the BLUE algorithm is evaluated by simulations and an experimental testbed at [18]. Even it was shown that BLUE beats RED when ECN is used in [18], BLUE has

not been evaluated for different levels of RTTs and without ECN support.

In this report, performance of the BLUE algorithm and the RED algorithm is evaluated and compared by simulations using ns simulation tool [19]. The network architecture used in the simulations is shown in Figure 14. 20 source nodes are connected through a bottleneck link to the 20 destination nodes. Each node runs 50 simultaneous exponential traffic applications with 2ms ON, 3ms OFF times with 1000 bytes packets. All the nodes are connected to the 10Mb/s, 10ms bottleneck with 100Mb/s links. The delays of the links between the sources and the bottleneck are selected differently for each link to differentiate the RTTs of flows. Flows are started randomly in the first 70 seconds of a 200-second simulation and the measurements are taken during the second half of the simulation duration. All nodes employ TCP Reno.



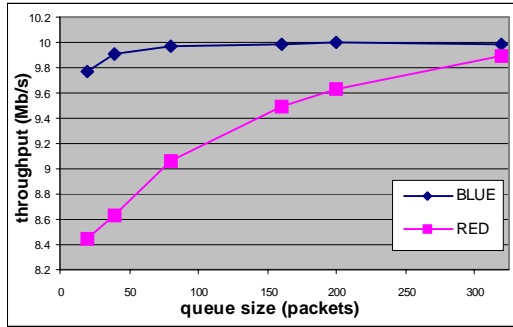
**Figure 14 Simulation network architecture**

RED or BLUE queue is used at the bottleneck link. Simulations are repeated for various queue sizes. RED queue is configured as  $min_{th} = q\_size/4$  and  $max_{th} = 3min_{th}$  where  $q\_size$  is the queue capacity. The marking limit of RED,  $max_p$ , is selected as 1 which is the best selection for this quantity of flows [20].  $w_q$  parameter is selected as 0.002. BLUE queue is configured as  $d_i = 0.0005$ ,  $d_d = 0.001$  and  $freeze\_time = 10ms$ . These BLUE parameters are the optimized values to increase the throughput and decrease the loss rate [16]. The link delays between the sources and the bottleneck are selected as  $(2i+200)ms$  where  $i$  is the source node index between 0 and 19. All the nodes are ECN capable and gateways mark the packets instead of dropping them.

Figure 15 reveals that BLUE performs better than RED for each queue size, when throughput is considered. BLUE converges to the 100% link

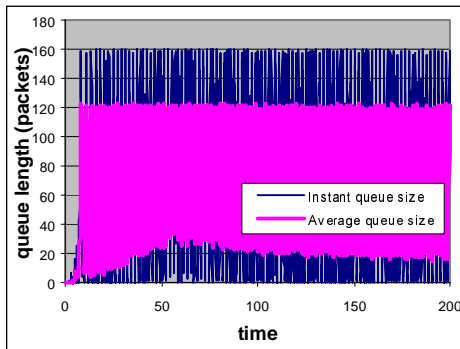


utilization using small queues. However, RED can reach that efficiency only for large queues.

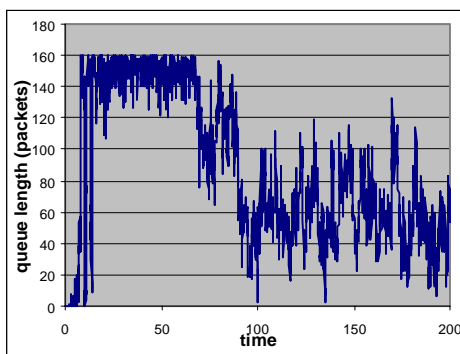


**Figure 15 Total throughput over bottleneck link for both RED and BLUE (large RTT)**

The inefficient link usage of RED is caused by its inability to control the queue length for high number of active bursty flows. As it is seen in Figure 16, both the instant queue length and the average queue length of RED oscillate between the queue limits. As a result, queue overflows just after a link idle event and becomes empty just after a queue overflow.



**Figure 16 RED average and instant queue size graph for  $q_{\max} = 160$  (large RTT)**

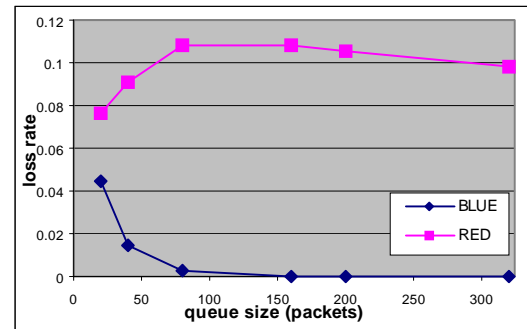


**Figure 17 Queue size for BLUE when  $q_{\max} = 160$  (large RTT)**

Figure 17 shows that BLUE can control the queue successfully over time once the behavior

of the traffic is learned. When new sources are activated, BLUE suffers from the changing traffic load. Once the traffic is stabilized, BLUE can control the queue successfully since it *knows* how to mark the packets. This control keeps the link at the high throughput condition.

In Figure 18, drop rates obtained for RED and BLUE can be compared. The queue instability occurring in RED causes large drop rates besides low link utilization. These drops are mainly forced drops, which are caused by queue overflows. However, as shown Figure 18, when queue size is increased BLUE converges to zero drop rates. Since no queue overflow is occurred and ECN is used to mark packets, packet loss can be kept at minimum rate.



**Figure 18 Total drop rate over bottleneck link for both RED and BLUE (large RTT)**

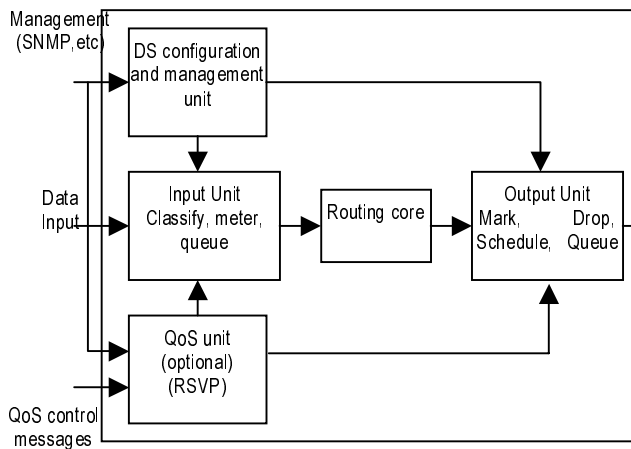
#### F. DIFFERENTIATED SERVICES MECHANISMS ON ROUTERS

Building blocks of a DS router is given in Figure 19. Here packets enter the router from the input unit. They are classified, metered and queued here. The routing core determines the output unit, which the packets will be sent. In the output unit, packets are marked, scheduled and queued to offer service differentiation.

Diffserv operating parameters are monitored and provisioned through the configuration and management interface. Monitored parameters include statistics regarding traffic carried at various Diffserv service levels. These statistics may be important for accounting purposes and/or for tracking compliance to Traffic Conditioning Specifications (TCSs) negotiated with customers.

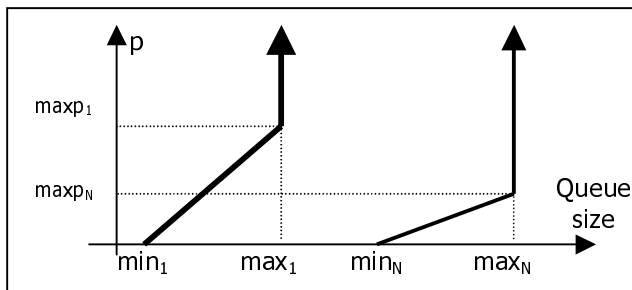
Diffserv routers may snoop or participate in either per-microflow or per-flow-aggregate

signaling of QoS requirements e.g. using the RSVP protocol. Snooping of RSVP messages may be used, for example, to learn how to classify traffic without actually participating as a RSVP protocol peer. Diffserv routers may reject or admit RSVP reservation requests to provide a means of admission control to Diffserv-based services or they may use these requests to trigger provisioning changes for a flow-aggregation in the Diffserv network. A flow-aggregation in this context might be equivalent to a Diffserv BA or it may be more fine-grained, relying on a multi-field (MF) classifier



**Figure 19 Building blocks of the DS router**

To offer service differentiation output unit may employ different queues served with a WRR style scheduler. Here, RED queue is used for BE traffic, simple DT queue is used for EF traffic. For each AF class, RIO (RED with In and Out) [21] style queue management algorithm should be used to be able to serve packets with different priorities in one queue.



**Figure 20 RIO parameters**

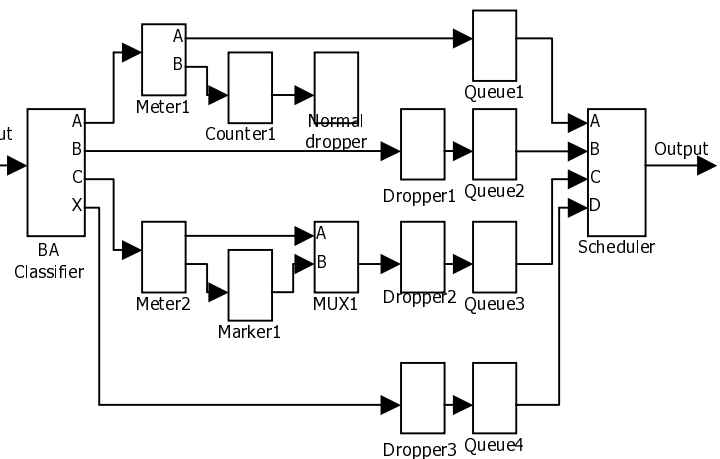
RIO runs N (3) different parameterized RED algorithm on the queue to mark each packet respective to its priority. It begins to drop low priority packets much before the high priority

packets. When the congestion increases high priority packets are dropped.

**Table 1 Sample SLS**

DS Field	PHB	Profile	Behavior
001001	EF	Profile1	Drop out-of-profile packets
001100	AF11	Profile2	Shape to the profile, drop tail when full
001101	AF21	Profile3	Mark out packets' DS field as 001000, drop tail when full
None	BE	None	RED-style packet drop

To comply with the SLS given in Table 1, routers should be configured similar to the Figure 21.



**Figure 21 Sample router configuration**

## G. BLUE WITH IN AND OUT (BIO)

3 different priority packets from one AF class can be served using a RIO queue to offer DS functionality. RIO provides different levels of services to each priority.

RIO is an algorithm based on RED. For high number of active flows on the gateway, RED loses the control of the queue. BLUE is shown that performing better than RED for this kind of configuration. Inspired of BLUE's success, we developed a simple alternative to RIO, named BIO (Blue with In and Out).

With the algorithm in Figure 22, two distinct BLUE algorithms are run for in and out packets. While total number of packets are compared with the out limits ( $max_{out}$  and  $min_{out}$ ), only the in packet count is used in comparison with the in limits ( $min_{in}$  and  $max_{in}$ ), similar to RIO.

```

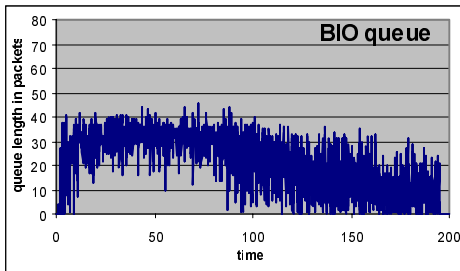
On packet enqueue
if (packet is in profile)
    mark packet with  $p_{m,in}$ 
else
    mark packet with  $p_{m,out}$ 
if  $Q_{len,in} > \max_{in}$  and  $(\text{now-last\_in\_inc}) > \text{in\_hold}$ 
     $p_{m,in} = p_{m,in} + d_{i,in}$ 
    last_in_inc = now
if  $Q_{len} > \max_{out}$  and  $(\text{now-last\_out\_inc}) > \text{out\_hold}$ 
     $p_{m,out} = p_{m,out} - d_{i,out}$ 
    last_out_inc = now
On link ready for transmission event
if link is idle or  $Q_{len,in} < \min_{in}$ 
    if  $(\text{now-last\_in\_dec}) > \text{in\_hold}$ 
         $p_{m,in} = p_{m,in} - d_{i,in}$ 
        last_in_dec = now
if link is idle or  $Q_{len} < \min_{out}$ 
    if  $(\text{now-last\_out\_dec}) > \text{out\_hold}$ 
         $p_{m,out} = p_{m,out} - d_{i,out}$ 
        last_out_dec = now

```

**Figure 22 Two-level BIO algorithm**

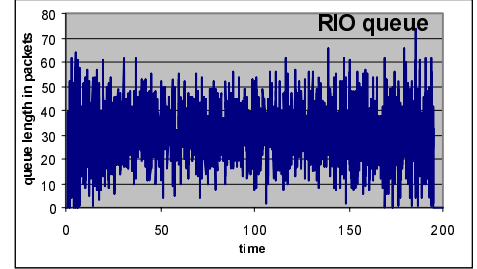
Performance evaluation of the algorithm is tested on network given in Figure 14. Ten nodes are used with 100 active flows, which are transferring an infinite-length file. RIO and BIO queues are used for AF queues on router1. Bottleneck link bandwidth is 33Mbps and all other links are 100Mbps. Each node-pair's RTT is chosen as  $RTT_i = 2(i/2+1)*20+2$  ms where  $i$  is the node-pair index between 0 and 9. Source nodes' queues on links to router1 are 100 packets long with each packet are 1000 bytes. While the even nodes' target rate is configured as 5Mbps, it is chosen as 1Mbps for odd nodes. All the flows on a node is measured according to the node's target rate. RIO parameters are chosen as  $\min_{out}=10$ ,  $\max_{out}=30$ ,  $\max_{p_{out}}=0.2$ ,  $\min_{in}=40$ ,  $\max_{in}=70$ ,  $\max_{p_{in}}=0.02$  and  $w_q=0.002$  as similar to [20]. BIO parameters are chosen as  $d_{i,in}=0.0005$ ,  $d_{d,in}=0.01$ ,  $d_{i,out}=0.001$ ,  $d_{d,out}=0.01$ ,  $\max_{in}=qlim$ ,  $\min_{in}=\max_{in}/3$ ,  $\max_{out}=\min_{in}$  and  $\min_{out}=\max_{in}/3$ .

Simulations are run for 200 ms and all the sources are randomly started in first 70 ms of the simulation. Bandwidth measurements are performed for the period of [90-190] ms.



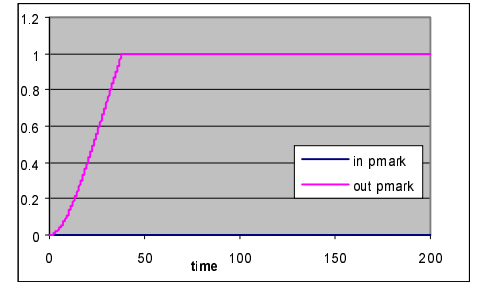
**Figure 23 BIO queue length in packets**

Figure 23 and Figure 24 show the queue lengths for BIO and RIO queues during the simulation.



**Figure 24 RIO queue length in packets**

The main observation is that queues do not reach the queue limit, and are successfully controlled by RIO. RIO algorithm's success in controlling the queue against RED is a result of marking out packets early. It is seen that BIO marks packets too aggressively after the flows' startup. The reason behind is the out packet marking probability,  $p_{m,out}$ . As seen in Figure 25,  $p_{m,out}$  reaches to 1 and remains there till the end of the simulation.



**Figure 25 In and out packet marking probability of BIO**

The result obtained is caused by the self-configuring architecture of the out packet marking probability in BIO.  $p_{m,out}$  can be decreased by queue idle events, but the average queue length is almost higher than  $\max_{out}$ .

A solution may be not auto-updating  $p_{m,out}$ , and setting it as multiple of  $p_{m,in}$ . Another alternative is comparing out limits with a function,  $f(Q_{len,in}, Q_{len,out})$  of in and out packet count. Studied algorithm sets  $f(Q_{len,in}, Q_{len,out}) = Q_{len,in} + Q_{len,out}$ .

## H. CONCLUSION

Active queue management algorithms detect congestion earlier and convey notification to sources by dropping or marking packets. These

mechanisms increase throughput and decrease loss rate.

RED, the most popular active queue management algorithm, has inherent problems besides its proven benefits. A new active queue management algorithm, BLUE, can perform better with the use of ECN in the case high number of flows are active at the gateways.

In this paper, the performance of RED and the performance of BLUE are compared by changing RTT values and ECN support under heavy bursty traffic. It is shown that: When ECN, which helps in the reduction of the loss rate, is used with RED; it loses the queue control in case of large number of active flows. For some conditions RED behaves like a drop-tail queue, for others queue size oscillates between the queue limits. However, ECN supported BLUE algorithm can control queue successfully and decrease the loss rate for central gateways with high number of flows active. When ECN support disappears, the performance of BLUE becomes closer to that of RED.

Today ECN is not deployed widely, thus BLUE's benefits cannot be observed in real world. Even if BLUE cannot surpass the performance of RED without ECN support, it can still race with it, and for ECN capable hosts and networks it will help to decrease the packet loss rate.

Inspired of the success of the BLUE against RED for high number of flows, a new queuing algorithm, BIO, to be used in Differentiated Services nodes instead of RIO is proposed.

BIO was expected to achieve lower loss-rate and higher link utilization. However, due to the two level packet discard algorithm and self-configuring architecture, BIO performs worse than RIO.

These results also show that any self-configuring multi-level queuing algorithm will fail to find proper marking probability for lower priority traffic classes.

Work has been done to make changes to the BIO algorithm to carry the success of BLUE to the DS nodes.

## I. REFERENCES

- [1] **Information Sciences Institute University of Southern California**, 1981, INTERNET PROTOCOL, Request for Comments: 791
- [2] **Cisco Documentation**, Quality of Service (QoS) Networking, [http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito\\_doc/qos.htm](http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/qos.htm)
- [3] **Ferguson, P., Huston, G.**, 1998, Quality of Service in the Internet: Fact, Fiction, or Compromise, <http://www.employees.org:80/~ferguson/inet-qos.ps>
- [4] **Braden, R., Clark, D., Shenker, S.**, 1994, Integrated Services in the Internet Architecture: an Overview, Request for Comments: 1633
- [5] **Braden, R., Zhang, L., Berson, S., Herzog, S., Jamin, S.**, 1997, Resource ReSerVation Protokol (RSVP) – Verison 1 Functional Specification, Request for Comments: 2205
- [6] **Shenker, S., Partridge, C., Guerin, R.**, 1997, Specification of Guaranteed Quality of Service, Request for Comments: 2212
- [7] **Shenker, S., Partridge, C., Guerin, R.**, 1997, Specification of the Controlled-Load Network Element Service, Request for Comments: 2211
- [8] **Blake, S., et.al.**, 1998, An architecture for Differentiated Services, Request for Comments: 2475
- [9] DS field
- [10] **Heinanen, J., Baker, F., Weiss, W., Wroclawski, J.**, 1999, Assured Forwarding PHB Group, Request for Comments: 2597
- [11] **Jacobson, V., Nichols, K., Poduri, K.**, 1999, An Expedited Forwarding PHB, Request for Comments: 2598
- [12] **Defense Advanced Research Projects Agency Information Processing Techniques Office**, 1981, TRANSMISSION CONTROL PROTOCOL, Request For Comments: 793
- [13] **Braden, B., et. al.**, 1998, Recommendations on Queue Management and Congestion Avoidance in the Internet, Request For Comments: 2309
- [14] **Ramakrishnan, K., Floyd, S.**, 1999, A Proposal to add Explicit Congestion Notification (ECN) to IP, Request for Comments: 2481
- [15] **Floyd, S., Jacobson, V.**, 1993, Random Early Detection Gateways for Congestion Avoidance, IEEE/ACM Transactions on Networking, <http://www.aciri.org/floyd/papers/early.pdf>
- [16] **Doran, S.**, 1998, EBONE Interface Graphs, <http://adm.ebone.net/~smd/red-1.html>
- [17] **Feng, W., Kandlur, D., Shin, K.**, 1999, A Self-configuring RED Gateway, INFOCOM '99, <http://www.eecs.umich.edu/~wuchang/work/CSE-TR-349-97.ps.Z>
- [18] **Feng, W., Kandlur, D., Saha, D., Shin, K.**, 2000, BLUE: A new class of active queue management, <http://www.eecs.umich.edu/~wuchang/blue/>
- [19] NS: Network Simulator, <http://www-mash.cs.berkeley.edu/ns/>
- [20] **Okuroğlu, B., Oktuğ, S.**, 2000, TCP/IP'de Aktif Kuyruk Yönetim Mekanizmaları ve Internet İçin Kaliteli Hizmet, MS Thesis, İstanbul Technical University, Control and Computer Engineering
- [21] **Clark, D., Fang, W.**, Explicit Allocation of Best Effort Packet Delivery Service, <http://diffserv.lcs.mit.edu/Papers/exp-alloc-ddc-wf.pdf>
- [22] **Jacobson, V.**, 1988, Congestion Avoidance and Control, Proceedings of SIGCOMM '88, Stanford, CA, ACM, <http://www-nrg.ee.lbl.gov/papers/congavoid.pdf>
- [23] **Jacobson, V.**, Congestion Avoidance and Control, Proceedings of SIGCOMM '88, August 1988, pp.314-329.
- [24] **Labrador, M., Banerjee, S.**, 1999, Packet Dropping Policies for ATM and IP Networks, IEEE Communications Surveys, vol 2, no 3, <http://www.comsoc.org/pubs/surveys>
- [25] **May, M., Bolot, J., Diot, C., Lyles, B.**, 1999, Reasons not to deploy RED, INRIA Sophia-Antipolis, ENSIM, Sprintlabs, <http://www-sop.inria.fr/rodeo/personnel/mmay/may-red.html> <http://www-sop.inria.fr/rodeo/personnel/mmay/papers/slides-wwqos-sprint.ps>