QoS Buffer Management of Multimedia Networking with GREEN Algorithm

Lain-Chyr Hwang, Cheng-Yuan Ku, Steen J. Hsu, and Huan-Ying Lo

Abstract: **The provision of QoS control is a key of the successful deployment of multimidia networks. Buffer management plays an important role in QoS control. Therefore, this paper proposes a novel QoS buffer management algorithm named GREEN (Global Random Early Estimation for Nipping), which extends the concepts of ERD (early random drop) and RED (random early detection). Specifically, GREEN enhances the concept of "Random" to "Global Random" by globally considering the random probability function. It also enhances the concept of "Early" to "Early Estimation" by early estimating the network status. For performance evaluation, except compared with RED, extensive simulation cases are performed to probe the characteristics of GREEN.**

Index Terms: **QoS, buffer management, RED, GREEN.**

I. INTRODUCTION

Multimedia communication is the trend in the coming epoch. The corresponding solutions of multimedia communication are not only the broadband/high-speed networking but also the differentiated service. For matching different QoS (Quality of Service) requirements, many traffic control mechanisms are developed. This paper will study a buffer management scheme.

In the past two decades, there were many researchers using buffer management schemes to improve the QoS, e.g., Drop Tail [1] and [2], Random Drop [1], Early Random Drop (ERD) [1], and Random Early Detection (RED) [3]–[5]. With Drop Tail, when buffer is full, packets are dropped. With Random Drop, when a packet arrives and finds the buffer full, a randomly chosen packet in the buffer is dropped. With ERD, if the queue length exceeds a predefined threshold, each arriving packet may be dropped with a predefined and fixed dropping probability.

As regards RED, it does not directly use the queue size for the buffer management. It uses a moving average of queue size and two thresholds. If the average queue size is less than the minimum threshold, the arriving packet is accepted. If the average queue size is greater than the maximum threshold, the arriving packet is dropped. When the average queue size is between the two thresholds, the packet may be marked with a marking probability, which is a function of average queue size. However, the advantage of using average queue size instead of queue size is not doubtless. Maybe the usage of average queue size is able to accommodate bursty traffic and transient congestion [3], but it also results in slow reaction to the variation of network status. Besides, in RED, marking a chosen packet instead of dropping the packet is a good idea. This idea may come from leaky bucket schemes [6]–[8] or ATM traffic controls [7], which deal with the violation traffic by marking. But the meanings of these two "markings" are somewhat different. The marking in RED means either dropping or tagging, while the meaning in leaky bucket generally means tagging only. Additionally, there are some variants of RED or ERD. Pippas [9] considered a RED variation for delay control, where the average queue length was replaced by the average delay. Hasegawa [10] used an enhanced RED (in fact, it is ERD, not RED) for multiple TCP connections to improve the fairness of the system.

This paper proposes a novel QoS buffer management scheme, named Global Random Early Estimation for Nipping (GREEN) [11]. It enhances RED scheme in two aspects. One is global consideration of the random probability. Different from other buffer management algorithms, which used only network statuses as the parameters of random probability function, GREEN uses additionally QoS requirements, which are described in the service contract and can be assigned to intermediate nodes at call setup. In this way, the goodput of the system can be improved. The goodput in general is defined as the valid throughput seen by upper layer. The other is the usage of Early Estimation instead of early detection. At a previous time, it can estimate that buffer congestion will happen or not. If it forecasts there will be congestion, a next arriving packet will be nipped (marked if we use the term in RED) with a larger probability. In this way, the decision marking for nipping a packet or not will be faster, because some calculations have been finished beforehand. Consequently, GREEN is able not only to improve the system goodput, but also to reduce the time of decision-making.

The remainder of this paper is organized as follows. The GREEN algorithm is given in Section II. And, numerical examples are given in Section III. Finally, Section IV concludes this paper and points some further works.

II. GREEN ALGORITHM

A. The Meaning of GREEN

GREEN still uses the main stream of buffer management, the combination of "Early" and "Random." However, GREEN elaborates them to a most complete realm. First, GREEN globalizes the random probability of choosing a packet. It points out that the random probability can be a function with parameters not only of network statuses but also of QoS requirements. The

Manuscript received June 29, 2001.

L.-C. Hwang is with the Department of Electrical Engineering, I-Shou University, Taiwan, e-mail: lain@isu.edu.tw.

C.-Y. Ku is with the Department of Information Management, National Chung Cheng University, Taiwan, e-mail: coopercy@ms16.hinet.net.

S. J. Hsu and H.-Y. Lo are with the Department of Information Engineering, I-Shou University, Taiwan, e-mail: steenhsu@isu.edu.tw, m883301m@isu.edu.tw.

This work is partially supported by Nation science council, Taiwan, Republic of China under contract number NSC 90-2213-E-214-030.

Table 1. Notation specification.

Notation	Meaning	Initial Value
P_n	Nipping probability	Null
T_n	Nipping threshold	Given
Δ_q	Queue length variation	0
q	Queue length observed by current arrival packet	Null
q_{pre}	Queue length observed by previous arrival packet	0
w_q	Weight for calculating queue length variation	Given
P_{e_pre}	Entropy probability calculated at previous packet arrival	0
P_e	Entropy probability calculated at current packet arrival	Null
B_u	Used link bandwidth averaged during two consecutive packet arrivals	0
B_a	All the link bandwidth	Given
P_l	Loss ratio after a packet nipped/dropped	0
$\hat{P}_l$	Estimated loss ratio, assuming the arriving packet nipped	Null
w_l	Weight for calculating estimated loss ratio	Given
l	Loss ratio requirement	Given
c_p	Number of packets entering buffer since last packet nipped	0
c_b	Total size of the c_p packets	0
L	Packet size of current arrival	Null
$L_{\max}$	Maximum packet size	Given
d	Delay requirement	Given
k	Buffer size	Given

Global Random probability of choosing a packet to be punished is named Nipping probability in GREEN. Under the consideration of Global Random, the invalid packets that conflict the QoS requirements will be nipped. This can save the buffer resource for the valid packets to improve goodput. Secondly, GREEN uses "dual Early," not only "Early." It uses Early Estimation. Specifically, when making the decision of punishing a packet or not, GREEN uses not only network statuses (e.g., queue length) but also their variations. The variations can be used to do estimation. Because estimation is earlier than detection, that is the reason to call it "dual Early." An advantage of estimation over detection is the speeding up of the decision-making, because some steps of calculations for decision-making are calculated beforehand. The last letter of the acronym GREEN stands for Nipping. The meaning of "Nipping" is the same as the term "Marking" in RED. Three reasons to use nipping instead of marking are listed below.

1. The term "mark" is confusing, because the same term "mark" has been used in leaky bucket and in ATM priority bit. Accordingly, "marking" would be easily misunderstood as just tagging, not discarding.
2. An idea comes from the saying "Nip in the bud, stop the development of." That is, "nipping" a violation packet can mean "destroying" (discarding) it or "just taking a bite of" (tagging) it. In other words, "nipping" specifies the meaning of "discarding directly or bypassing after setting some bits" better than "marking."
3. Make the acronym elegant, the same motive as the last letter D in RED. In fact, it is also easy to misunderstand the meaning of D in RED as Discard [5], not Detection.

B. *The Algorithm of GREEN*

The notations and initial values are given in Table 1. The threshold for decision-making is named nipping threshold and is denoted by T_n. If a packet arrives at and finds the queue length over the nipping threshold, a probability, named nipping probability P_n, is calculated to decide on nipping the packet or not. The core of GREEN is the calculation of nipping probability, which is similar to the core of RED. Hence, GREEN keeps the property of uniform random variable in RED. That is, GREEN uses a variable c_p to count the number of packets entering the buffer since last nipped packet. And similar to RED, nipping probability is a function of another probability, which is called entropy probability in GREEN.

The Global Random property of GREEN is exhibited at the components of nipping probability, which consists of not only network status (e.g., queue length and link utilization) but also QoS requirements (e.g., delay requirement and loss requirement). It is natural to nip a packet, if some of its QoS requirements cannot be met. Hence, the nipping probability can be expressed by

$$P_n = \max\left\{U(q/B_a - d), [P_{e\_pre}/(1 - c_p P_{e\_pre})]U(l - \hat{P}_l)\right\}. \tag{1}$$

where $U(x)$ is the unit step function defined as

$$U(x) = \begin{cases} 0, & \text{if } x < 0, \\ 1, & \text{if } x \geq 0. \end{cases} \tag{2}$$

The two unit step functions $U(q/B_a - d)$ and $U(l - \hat{P}_l)$ are included into (1) under the consideration of QoS requirements for the Global Random property of GREEN, and painstakingly arranged in a way to make the conditions listed in Table 2 reasonable. The estimated loss ratio $\hat{P}_l$ in $U(l - \hat{P}_l)$ is obtained by a moving average with a weight w_l, and expressed by

$$\hat{P}_l = w_l(L/(c_b + L)) + (1 - w_l)P_l. \tag{3}$$

This estimation value is calculated under the assumption that the arrival packet has been nipped. If its nip makes the loss ratio requirement not satisfied, the arrival packet will not be nipped as far as possible.

One more thing to mention about is the term $P_{e\_pre}/(1 - c_p P_{e\_pre})$ in (1). It is possible for some large c_p to make the term greater than 1 or negative. The same thing may also happen in

Table 2. The relation between nipping and requirements.

Delay requirement ($U(q/B_a - d)$)	Loss requirement ($U(1 - \hat{P}_l)$)	Nipped or not
Satisfied (0)	Satisfied (1)	Nipped if necessary/Not if possible
Satisfied (0)	Non-satisfied (0)	Not (surly)
Non-satisfied (1)	Satisfied (1)	Nipped (surly)
Non-satisfied (1)	Non-satisfied (0)	Nipped (surly)

RED, but not mentioned in [3]. To avoid $P_{e\_pre}/(1 - c_p P_{e\_pre})$ greater than 1 or negative, the authors introduce a constraint to keep it in [0, 1], i.e.,

$$0 \leq P_{e\_pre}/(1 - c_p P_{e\_pre}) \leq 1, \tag{4}$$

then

$$0 \leq \min(1/P_{e\_pre}, 1/P_{e\_pre} - 1) = (1 - P_{e\_pre})/P_{e\_pre}. \tag{5}$$

Hence, if $c_p > (1 - P_{e\_pre})/P_{e\_pre}$, then $P_{e\_pre}/(1 - c_p P_{e\_pre})$ is assigned to 1. Thus, $P_{e\_pre}/(1 - c_p P_{e\_pre})$ is constrained to be in [0,1].

The last paragraph has shown the Global Random property of GREEN. This paragraph will show the Early Estimation property of GREEN. It is achieved by pre-calculation of some parts of nipping probability function. In fact, those parts make up the entropy probability P_e. The entropy probability is used to represent the normalized entropy (not greater than 1) to which the strength of nipping a packet is corresponding. The entropy is in terms of buffer occupancy and link bandwidth occupancy. For Early Estimation purpose, the entropy will be obtained by estimation. In the aspect of buffer occupancy, a queue length is estimated. If queue length variation is denoted by Δ_q and calculated by moving average with a weight w_q, then it can be expressed by

$$\Delta_q \leftarrow w_q(q - q_{pre}) + (1 - w_q)\Delta_q. \tag{6}$$

The arrow represents the recursive relation between a new Δ_q and an old Δ_q, like a common use of Eq. (1) in [3]. By using $q + \Delta_q$ to estimate the queue length of next arrival time, a component $\max(q + \Delta_q, 0)/k$ contributes to P_e. The 0 in the maximum operation is to make the probability not less than 0. The component $\max(q + \Delta_q, 0)/k$ indicates that the nipping probability of next arrival packet should be larger, if the (estimated) queue length of next packet arrival time is larger. In the aspect of link utilization, the entropy probability should be larger, if the available bandwidth is smaller, i.e., the link utilization is larger. According to this situation, a component B_u/B_a contributes to the entropy probability. It is the busy probability (busy ratio) of the link, where B_u is the used link bandwidth averaged during the inter-arrival time between this packet and previous packet. It indicates nipping the arrival packet with larger probability, if the available bandwidth is smaller. According to these factors mentioned above, the entropy probability can be expressed by

$$P_e = \min\{[\max(q + \Delta_q, 0)/k](B_u/B_a), 1\}. \tag{7}$$

The min operation is used to constrain the entropy probability less than 1. Note that this entropy probability is not used for the calculation of nipping probability of this arrival packet. It is used for next arrival packet. That is the true essence of "Early Estimation." The overall GREEN algorithm can be described in Fig. 1. Notice that in Fig. 1, unlike the estimation of the queue length variation Δ_q, the P_l will not be replaced by $\hat{P}_l$ until a packet is really nipped. The $\hat{P}_l$ is just an imaged estimation value by assuming the arrival packet nipped, not a true statistic of loss ratio except the arrival packet is truly nipped or dropped.

```
Initialization: Set the initial values
                (see Table 1)
For each packet arrival

  Δq ← wq(q − qpre) + (1 − wq)Δq
  Pe = min{[max(q + Δq, 0)/k](Bu/Ba), 1}
  qpre = q
  P̂l = wl(L/(cb + L)) + (1 − wl)Pl

  if (Buffer full)
    Drop the packet
    Pl = P̂l
    cp = 0
    cb = 0
  else
    if (q > Tn)
      if (cp > (1 − Pe_pre)/Pe_pre)
        Pn = (U(q/Ba − d), U(l − P̂l))
      else
        Pn = max{U(q/Ba − d),
                 [Pe_pre/(1 − cpPe_pre)]U(l − P̂l)}
        Pe_pre = Pe
        if (a random value<Pn)
          Nip the packet
          Pl = P̂l
          cp = 0
          cb = 0
        else
          cb+ = L
          cp + +
    else
      Accept the packet
      cp = 0
      cb = 0
```

Fig. 1. The GREEN algorithm.

It's imaginable that GREEN is more complex than RED, but it is reasonable to get more additional features by paying some cost. However, it does not mean that there are more calculations needed at the decision moment. Because GREEN considered many other conditions (e.g., buffer full, counter over some

limit, etc.), there are many other steps in GREEN. These conditions should be also considered in RED, but are not shown there. When comparing RED with GREEN, the cores of them are the calculations of p_a (RED) and P_n (GREEN), respectively. In RED, first the average queue length should be found, and then the p_b, and finally the p_a to make the decision. In GREEN, first P_l should be found and then the P_n. No calculation is needed for P_{e_pre}, which is found in parallel at last packet arrival. By quantity comparison, RED needs 2 additions, 4 or 5 subtractions, 4 or 3 multiplications, 2 divisions, and 2 logical comparisons. Furthermore, there are a linear function and a power function for nonempty queue. On the other hand, GREEN needs 2 additions, 3 subtractions, 4 multiplications, 3 divisions, and 3 logical comparisons, so it is obvious that GREEN is faster than RED to make a decision.

III. NUMERICAL EXAMPLES

The arrival process is modeled as a discrete ON-OFF random process. For simplification, the transition probability from ON to OFF is denoted by α, and the transition probability from OFF to ON is $1 - \alpha$. In order to make a reasonable scale of the sojourn time of ON/OFF state relative to the packet inter-arrival time and to avoid zero sojourn time, the final sojourn time is obtained by adding the original sojourn time by one and then multiplying by 10. In the following examples, the α is selected to be 0.2. The arrival rate at ON state is constant and at OFF state is 0. The packet length L is assumed uniform distributed from 1 unit to $L_{\max}$ units. The buffer size and the nipping threshold are 100 units and 50 units, respectively. The maximum packet length is 20 units. The delay and the loss ratio requirements are 75 units and 1, respectively. Setting the loss ratio to 1 is for making the factor $U(l - \hat{P}_l)$ in (1) have no effect to simplify the observation of characteristics. The weights for calculating the queue length variation and the estimated loss ratio are both equal to 0.6. Two inter-arrival times 0.91 and 0.96 are used respectively. The parameters in the counterpart RED are the same, except the maximum threshold, the minimum threshold, and the weight for calculating the average queue length are 50, 25, and 0.002, respectively, which are the suggested values in [3]. We used the well-known values in [3], although the new recommendation values can be found in [12]. The above values are used throughout the examples unless otherwise stated. In each simulation, 100000 packets are generated. The goodput ratios shown in Figs. 2–9 are the total packet size ratios of valid accepted packets to all generated packets. When comparing GREEN with RED, the higher goodput ratio means the higher goodput. This paper uses goodput ratio instead of goodput, because from the value of goodput ratio one can easily get the rough number of accepted and valid packets. It makes more sense than goodput, which is the mean of valid accepted packet size in a time unit, in some ways.

A. Case 1: Inter-Arrival Time

The inter-arrival times at ON state are changing from 0.91 to 0.98. The simulation results are shown in Figs. 2 (a)–(c). From Fig. 2(a), it can be seen that as the inter-arrival time increases,

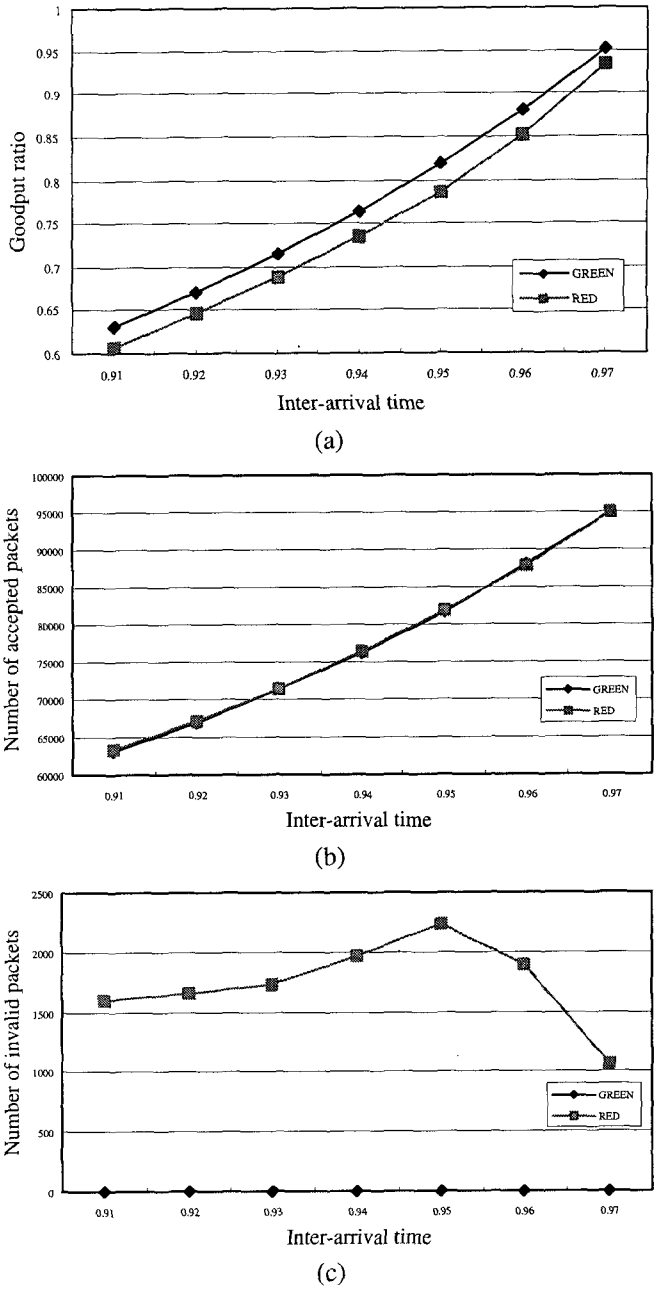


Fig. 2. (a) Goodput ratios for changing inter-arrival time; (b) the numbers of accepted packets for inter-arrival ratio; (c) the numbers of invalid packets for inter-arrival ratio.

the goodput ratio increases. That is, the number of valid accepted packets increases when the inter-arrival time increases. This characteristic is true for both GREEN and RED. That is because larger inter-arrival time results in smaller arrival rate to cause loose network status, which makes the nipping probability in GREEN and the marking probability in RED smaller. When comparing GREEN with RED, it is obvious that GREEN outperforms RED. It is expectable because the numbers of accepted packets of RED and GREEN are almost equal, and there are many invalid (e.g., not satisfy the delay requirement) accepted packets for RED, but none for GREEN; see Figs. 2(b) and (c).

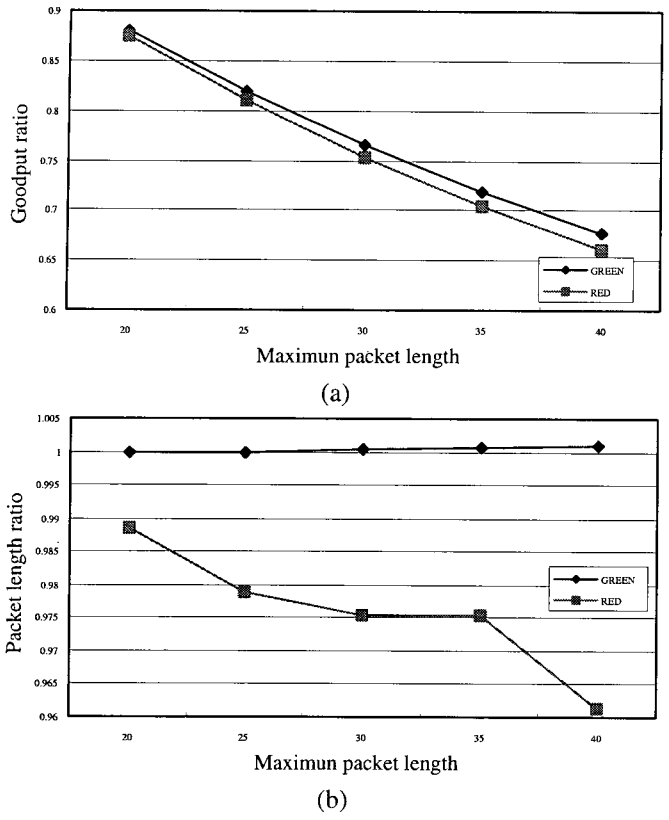


Fig. 3. (a) Goodput ratios for changing maximum packet length; (b) packet length ratios for changing maximum packet length.

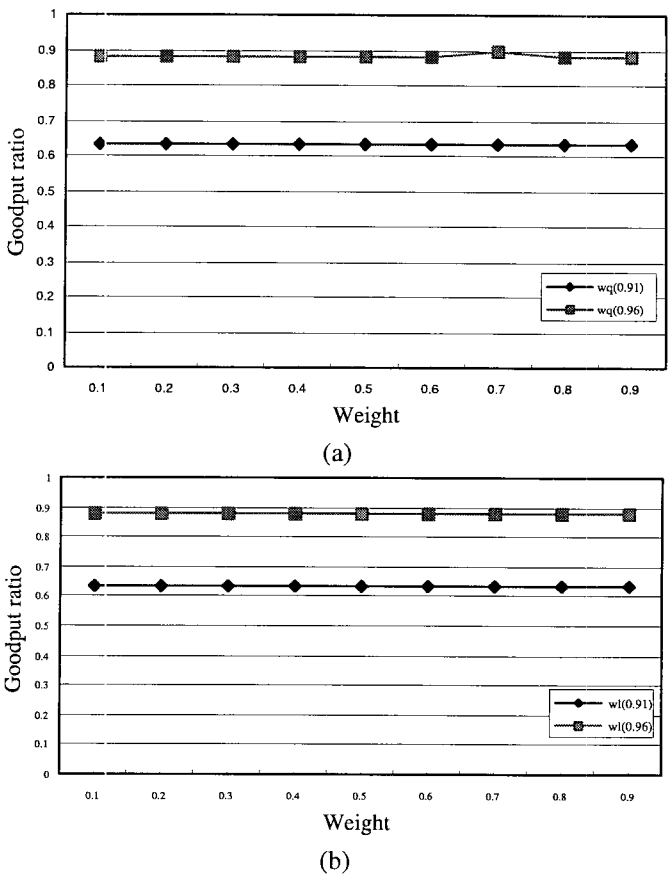


Fig. 4. (a) Goodput ratios for changing weight w_q; (b) goodput ratios for changing weight w_l.

B. Case 2: Maximum Packet Length

The inter-arrival time is 0.96. The maximum packet lengths are changing from 20 to 40. The simulation results are shown in Figs. 3(a) and (b). As can be seen in Fig. 3(a), the goodput ratios are decreased with larger maximum packet length, because larger packet length is more difficult to enter the buffer. From Fig. 3(b), it can be seen the packet length ratio of RED is smaller when the maximum packet length is larger. The packet length ratio is obtained from goodput ratio divided by acceptance ratio, where acceptance ratio is the ratio of the number of accepted packets to the number of total packets generated. The packet length ratio represents the ratio of the mean of accepted packet length to the mean of generated packet length. The results in Fig. 3(b) illustrate the smaller packets enter the buffer more easily than larger packet in RED, but it is not true in GREEN. That is because RED prefers smaller packet. The phenomenon results from the factor $L/L_{\max}$ of marking probability in RED. For the consistence between longer packets and short packets, we did not include it into (1). However, for some purpose, the factor can be included to improve the performance. It will be discussed shortly in next session.

C. Case 3: Weights

The weights for calculating the queue length variation and the estimated loss ratio are both changing from 0.1 to 0.9. When one weight is changing, the other weight is kept at 0.6. Owing to the difference of weight assignment between GREEN and RED, no counterpart is needed in this example. The simulation results are shown in Figs. 4(a) and (b). Seen in the figures, the influences of weights are not obvious. This means the weights can be selected arbitrarily. The advantage is that the system performance will not be spoiled by a wrong weight. However, here the temporal system performance has been not yet considered, which can be considered in the future work.

D. Case 4: Loss Ratio Requirement

The loss ratio requirements are changed depending on the inter-arrival times. The simulation results are shown in Figs. 5(a) and (b). The goodput ratios are not monotonically increasing or decreasing when loss ratio requirements increase. The reason can be derived from (1). When the loss ratio requirement is smaller, it means GREEN will accept packets as much as possible. The factor $U(l - \hat{P}_l)$ in (1) is almost 0, i.e., the second part in the max operation is almost 0. The factor $U(q/B_a - d)$ becomes the dominative term. Under this situation, the queue length easily increases to cause the packets over the delay requirement or over the buffer capacity, so many packets are lost to cause a smaller goodput ratio. On the other hand, when the loss ratio requirement is large enough, the factor $U(l - \hat{P}_l)$ is almost 1 and nipping probability is usually large to nip packets, so many packets are lost to cause a smaller goodput ratio. When loss ratio requirement is moderate, the above situations can be avoided to get a larger goodput ratio.

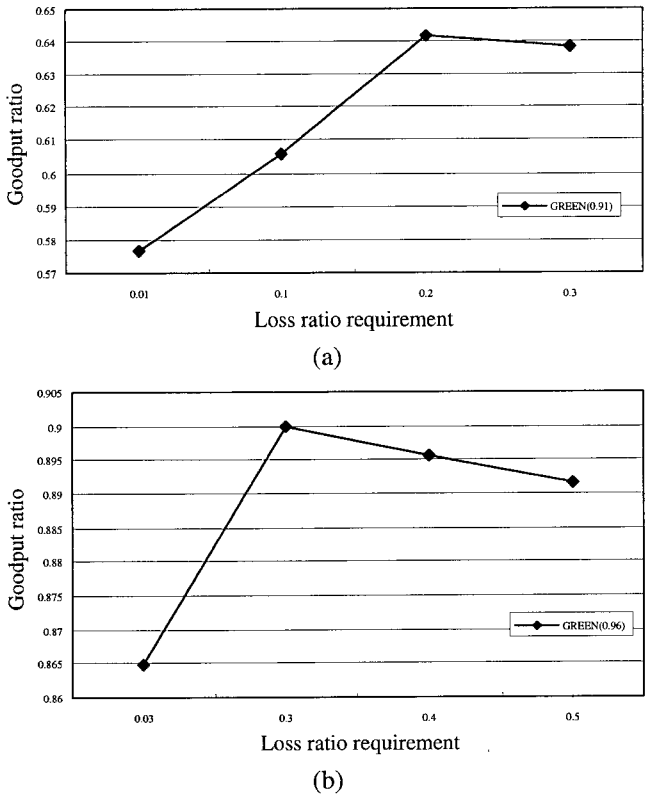


Fig. 5. (a) Goodput ratios for changing loss ratio requirement; (b) goodput ratios for changing loss ratio requirement.

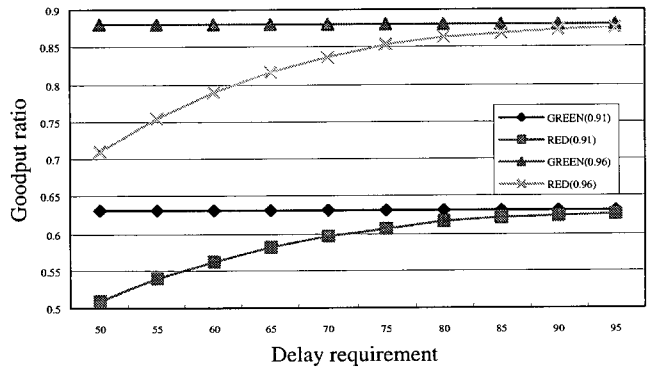


Fig. 6. Goodput ratios for changing delay requirement.

E. Case 5: Delay Requirement

The delay requirements are changed from 50 to 95. The simulation results are shown in Fig. 6. RED gets larger goodput ratio for larger delay requirement. However, in GREEN the delay requirements did not noticeably affect the goodput ratio. That is, GREEN can retain the high goodput ratio. It is a good characteristic of GREEN. The reason is specified below. When delay requirement is small, packets are nipped owing to over-delay requirement. This also makes the second part in max operation of (1) low, because the c_p is reset to 0 usually. On the other hand, when delay requirement is large, packets are nipped mainly corresponding to the second part in max operation of (1). Like the energy conservation law in physics, the nipping probabilities under these two situations are close to each other. The goodput ratios for RED are increasing when the delay requirements increase and approach to those of GREEN. Its reason is trivial. That results from the decrease of over-delay packets. Furthermore, the difference between RED and GREEN is more obvious when the inter-arrival time is large, because the goodput ratio is larger at larger inter-arrival time to make the absolute difference larger. In fact, the relative differences for small inter-arrival time and larger inter-arrival time are close to each other.

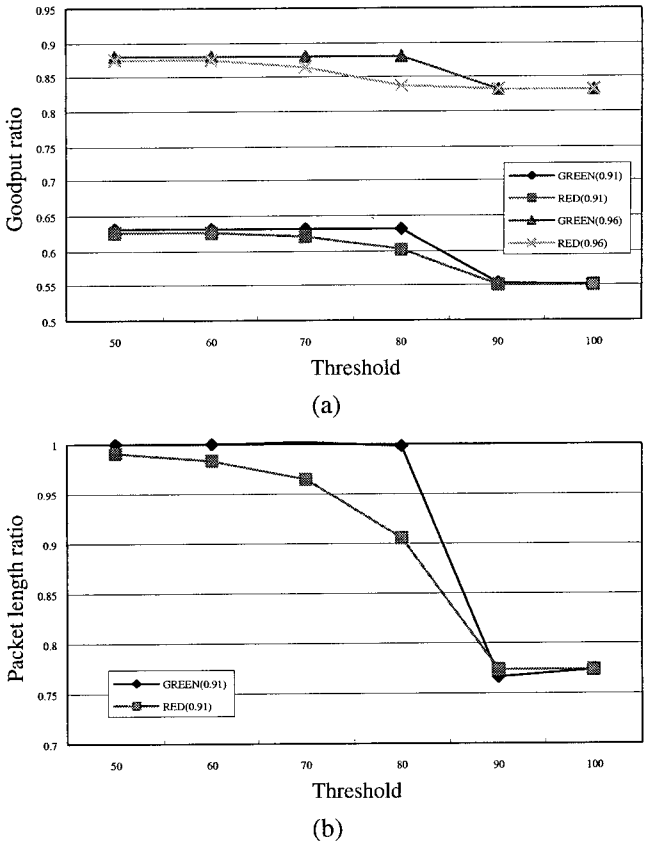
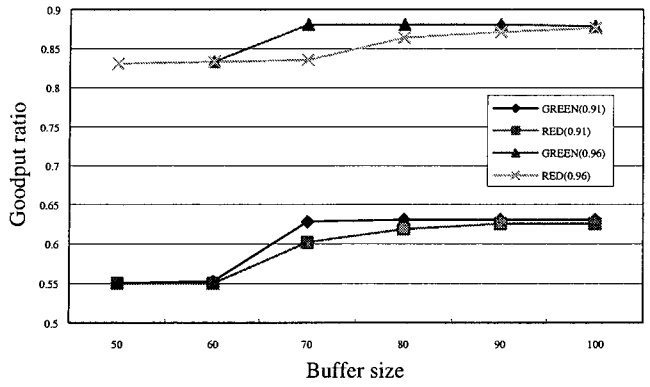


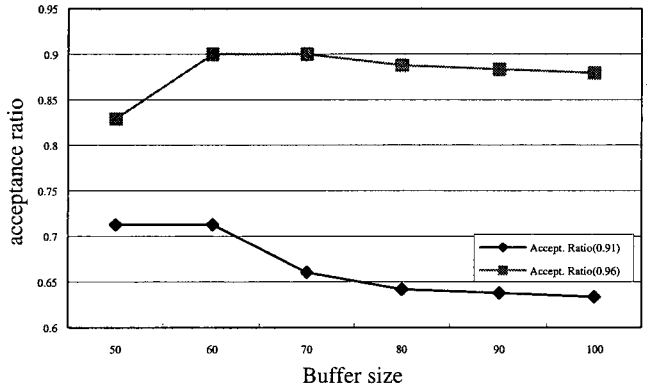
Fig. 7. (a) Goodput ratios for changing nipping threshold; (b) packet length ratios for changing nipping threshold.

F. Case 6: Nipping Threshold

The delay requirement is 100 units. Setting the delay requirement to 100 is for making the delay requirement factor $U(q/B_a - d)$ in (1) have no effect in order to simplify the observation of characteristics. The maximum threshold of RED is equal to the nipping threshold of GREEN, and the minimum threshold is a half of the maximum threshold. The simulation results are shown in Figs. 7(a) and (b). From Fig. 7(a), for GREEN, the goodput ratios are similar except when the nipping thresholds approach to buffer size. The reason is, when the nipping threshold approaches to buffer size, there is less buffer space to accommodate larger packets, which will be dropped to decrease the goodput ratio. That is, when the nipping threshold approaches to the buffer size, the system will prefer smaller packets. It can be seen from packet length ratio in Fig. 7(b). RED has the similar phenomena and prefers smaller packets more obviously.



(a)



(b)

Fig. 8. (a) Goodput ratios for changing buffer size; (b) acceptance ratios for RED.

G. Case 7: Buffer Size

The delay requirement is 100 units. The buffer sizes are changing from 50 to 100. The simulation results are shown in Figs. 8(a)–(b). From Fig. 8(a), the goodput ratios increase obviously in the beginning, but not in the subsequence. In the beginning, the buffer size increases to decrease the blocking probability. Nevertheless in the subsequence the buffer size is large enough to have no packets blocked, so increasing buffer size does nothing helpful. For RED, the phenomenon in Fig. 8(b) is interesting. The acceptance ratios decrease, while the goodput ratios increase (see Fig. 8(a)). Although RED was not proposed by us, we try to explain this phenomenon. Owing to buffer size increasing, the average queue size also increases. It will easily mark packets, so the acceptance ratio decreases. However, larger buffer size may accommodate larger packet to increase the goodput ratio.

H. Case 8: Buffer Size, Delay Requirement, and Nipping Threshold in a Ratio

In this case, the buffer size, the delay requirement, and the nipping threshold are in a ratio of 100:75:50. The buffer sizes are changing from 50 to 100. The maximum threshold of RED is equal to the nipping threshold of GREEN, and the minimum threshold is a half of the maximum threshold. The simulation results are shown in Fig. 9. The results are somewhat similar

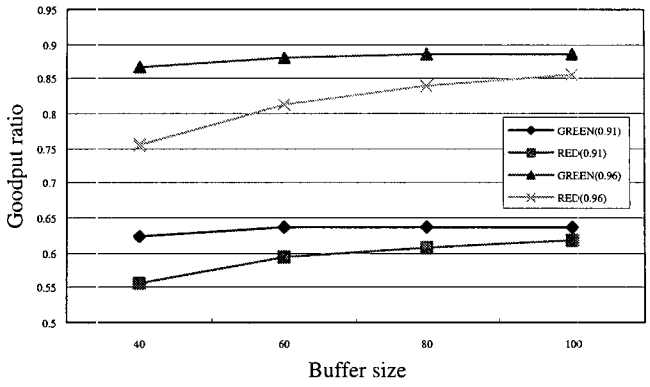


Fig. 9. Goodput ratios for changing buffer size, nipping threshold, delay requirement simultaneously.

to those in Case 7. For GREEN, the goodput ratio increases only in the very beginning. The reason is when the buffer size increases, the delay requirement and the nipping threshold increase, too. So, for buffer size large enough, the delay requirement and the nipping threshold are also large enough to make no packets dropped or nipped for over buffer size or delay requirement. The dominative term in nipping probability function becomes the entropy probability. Furthermore, for each buffer size, owing the input process is the same, the estimated loss ratio and the estimated queue length in (2) are similar to one another, so is the entropy probability. This results in the similar nipping probability, and then similar goodput ratio.

The assumptions of fixed QoS requirements in all numerical examples are suitable for DiffServ [13] (core router), where packets of a same class with the same QoS are accommodated in a queue. In this way, the QoS requirement information is fixed for each queue. If multiple queues share a common buffer, the per-class or per-flow informations should be kept for GREEN algorithm. Even in this way, the authors believe per-class information is small enough under the DiffServ architecture.

IV. CONCLUSIONS AND FUTURE WORKS

In multimedia communication networks, buffer management algorithms are the key components of the QoS control schemes. Hence, GREEN was proposed to attain the aim of QoS control in buffer management. GREEN elaborates the main stream of buffer management algorithms that combines "Early" with "Random" to a most complete realm. In fact, the proposed GREEN algorithm is only an example algorithm. GREEN algorithm can be designed from different viewpoints to get different effects. For example, add a factor $1 - L/L_{\max}$ into (1) if larger packets are favored, or add a factor $L/L_{\max}$ into (1) if smaller packets are favored.

This paper just makes an outset of the research on GREEN buffer management family, which does the buffer management depending on the information of QoS requirements additionally and estimates some factors for decision-making beforehand. So, here is only the simple system discussed. In a future differentiated service environment, there will be multi-class or multi-session in a network node. The buffer management should

handle this situation efficiently; so multi-class or multi-session buffer management should be a further study issue like [10]. Under this condition, the delay requirements and loss requirements may be diverse to increase the complexity of GREEN buffer management. Furthermore, like previous studies, GREEN in this paper mainly punishes violative packets by dropping them. However, in some aspects, tagging a violative packet and putting it into the network for further observation instead of dropping it directly will get better performance. Therefore, to consider tagging instead of dropping or to integrate tagging with dropping is another interesting problem.

REFERENCES

[1] E. Hashem, "Analysis of random drop for gateway congestion control," *Rep. LCS TR-465, Lab. for Comput. Sci., M.I.T.*, p. 103, 1989.

[2] L. Zhang, "A new architecture for packet switching network protocols," *MIT/LCS/TR-455, Lab. for Comput. Sci., M.I.T.*, 1998.

[3] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Networking*, vol. 1, no. 4, pp. 397–413, 1993.

[4] O. Elloumi and H. Afifi, "Improving RED algorithm performance in ATM networks," in *Proc. IEEE GLOBECOM'97*, vol. 2, 1997, pp. 1062–1066.

[5] R. Guerin and V. Peris, "Quality-of-service in packet network: Basic mechanisms and direction," *Computer Networks*, vol. 31, pp. 169–189, 1999.

[6] K.V. Waal, M. Dirksen, and D. Brandt, "Implementation of a police criterion calculator based on the leaky bucket algorithm," in *Proc. IEEE GLOBECOM'93*, 1993, pp. 713–718.

[7] J.M. Pitts and J.A. Schormans, *Introduction to ATM design and Performance*, Wiely, New York, 1997.

[8] D. Gan and S. McKenzie, "Traffic policing in ATM networks with multimedia traffic: The super leaky bucket," *Computer Commun.*, vol. 22, pp. 439–450, 1999.

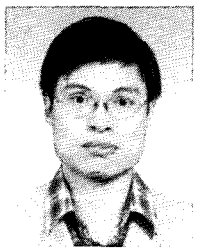
[9] J.B. Pippas and I.S. Venieris, "A RED variation for delay control," in *Proc. IEEE ICC'00*, vol. 1, pp. 475–479, 2000.

[10] G. Hasegawa *et al.*, "Comparisons of packet scheduling algorithms for fair service among connections on the Internet," in *Proc. IEEE INFOCOM'00*, vol. 3, 2000, pp. 1253–1262.

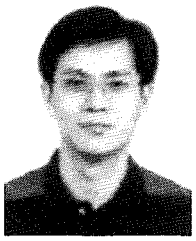
[11] L. C. Hwang *et al.*, "Extend Red to Green in Buffer Management," in *Proc. 8th IFIP Workshop on Performance Modelling and Evaluation of ATM & IP Networks (IFIP'00)*, Ilkley, West Yorkshire, U.K., July 17–19, 2000, pp. 34/1–34/9.

[12] Available at http://www.aciri.org/floyd/REDparameters.txt.

[13] S. Blake *et al.*, "An architecture for differentiated services," Internet RFC 2475, Dec. 1998.

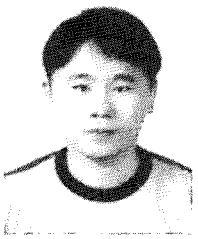


Lain-Chyr Hwang was born in Taiwan on 26 December 1965. He received his B.S. degree in Electrical Engineering from the National Sun Yat-Sen University, Kaohsiung, Taiwan in 1998, his M.S. degree in Communication Engineering (1990) and his Ph.D. in Electronic Engineering (1994) from National Chiao-Tung University, Hsinchu, Taiwan. From 1995 to 1996 he was with Telecommunications Laboratories, Chunghwa Telecom Co., Ltd, Taiwan as an associate researcher. There, he was involved in network planning of the intelligence network and personal communication services. He was an associate professor in the department of Management Information Systems of the I-Shou University, Taiwan, from August 1996 through July 2000, and in the department of Electrical Engineering, since August 2000. His research interests include performance evaluation and QoS control of multimedia communication networks.



Cheng-Yuan Ku was born in Taipei, Taiwan, on October 14, 1965. He received the B.S. degree in control engineering from National Chiao Tung University, Taiwan in 1987 and the M.S. and Ph.D. degrees in electrical engineering from Northwestern University in 1993 and 1995, respectively. From 1995 to 1999, he was with Department of Information Engineering, I-Shou University. Since 1999, he joined the Department of Information Management, National Chung Cheng University. His current research interests include cryptography, stochastic systems, dynamic programming, and network management. He is the member of INFORMS, IEEE, and IEICE.

Steen J. Hsu was born in Taiwan, 1963. He received the M.S. and Ph.D. degrees in computer science and information engineering from National Chiao Tung University, Taiwan, in 1992 and 1999, respectively. He is an assistant professor in the Department of information engineering at I-Shou University, Taiwan, since 1999. His current research interests include network reliability analysis, network traffic management, QoS routing and wireless LAN.



Huan-Ying Lo was born in Taiwan on 19 December 1972. He received his B.S. degree in Computer Science Information Engineering from the Chung Hun University, Taiwan in 1999, his M.S. degree in Information Engineering from I-Shou University, Kaohsiung, Taiwan, in 2001. His research interests include performance QoS control of multimedia communication networks.