Comprehensive performance model of differentiated service with token bucket marker

H. Su and M. Atiquzzaman

Abstract: Differentiated services (DiffServ) networks use two type of routers: edge routers mark the packet headers according to their service profile, while the core routers implement queue management to provide service differentiation between packets according to information in their packet headers. Most previous studies on the performance of DiffServ have considered either the effect of the edge marking scheme or the effect of core queue management on the throughput of TCP applications. To study the combined effect of marking at the edge router and queue management at the core router on the performance of DiffServ, the authors propose a simple model that takes into consideration the committed information rate and maximum allowed burst size of the marker, in addition to the number of flows in an aggregate, round trip time, and the probability of a packet dropping at the core network. Observations from the model are presented, and simulations were performed to validate the accuracy of the model. The model shows that the performance of DiffServ is largely flow-based rather than desired-aggregate-based, and the DiffServ network suffers from unfairness issues.

1 Introduction

The Internet has successfully provided a worldwide data communication service for the past few decades. However, it does not provide any quality of service (QoS) guarantee to applications. Emerging new service types, such as real-time audio/video applications, demand QoS support from the Internet. The differentiated services (DiffServ) network [1] is currently being developed by IETF to provide coarse-grained QoS to Internet applications.

Instead of providing a guarantee to individual flows, which may cause scalability problems, DiffServ provides statistical QoS to a few predefined service classes over a long time scale. In a DiffServ network, a service subscriber first sets up a service profile with an Internet service provider (ISP) regarding the desired type of service. At the ingress of the DiffServ network, edge routers classify all packets passing through them into several predefined service classes, and mark the packets with different drop precedences (such as in/out packets) [1] according to the subscriber's service profile. The traffic that conforms to the service profile are marked with low drop priority and will receive better service, while the nonconformant part of the traffic are marked with high drop priority and receive a best effort service. The token bucket marker is one of the most commonly used markers in edge routers. Variants of the token bucket marker [2, 3] were proposed to mark packets with multidrop priorities. Service differentiation of packets is provided by core routers, by using an active queue management scheme [4], such as RED with in and out (RIO) [5], according to the preassigned service classes and drop precedences carried in the packet headers.

Although Diffserv overcomes the scalability problem found in previous proposals to provide QoS to Internet applications, the extent to which the service goals of DiffServ can be achieved is still uncertain. Many recent research results have shown that the service goals promised by the DiffServ network are not likely to be achieved in many situations [6, 7] due to the TCP congestion control algorithm [8], which was developed for the best effort service and has no knowledge about the priorities of packets.

It is well known that for a long-lived TCP flow under the assumption of constant round-trip time (RTT) and low to medium periodic packet losses, the TCP congestion control window ($CWND$) follows a saw-tooth like pattern. Based on this assumption, Mathis *et al.* [9] described the behaviour of a single TCP flow by

$$BW = \frac{(MSS)c}{(RTT)\sqrt{p}} \tag{1}$$

where MSS is the maximum segment size, RTT is the round-trip time, p is the probability of packet loss in the network, and c is a constant. Under the assumption that packet losses are periodic, and the destination acknowledges every packet, c is equal to $\sqrt{3/2}$. This model captures only the congestion avoidance phase of the TCP congestion control algorithm [8]. Padhye *et al.* [10] developed a more accurate model by considering both congestion avoidance and retransmissions caused by time out. Although the result of the latter is more accurate, the result is not as intuitive as the former one.

Yeom and Reddy [11] have extended Padhye's work to evaluate the performance of a DiffServ network, while Baines *et al.* [6] have extended Mathis's work to model a DiffServ network. Their results and conclusions were similar. Although the effect of the committed information

IEE Proceedings online no. 20030588

doi:10.1049/ip-com:20030588

Paper first received 25th April and in revised form 11th December 2002

H. Su is with the School of Computing, Armstrong Atlantic State University, Savannah, Georgia 31419, USA

M. Atiquzzaman is with the School of Computer Science, University of Oklahoma, Norman, OK, 73019-6151, USA

rate (*CIR*) was considered in both papers [11, 6], either of them considered the effect of the marker settings on the performance of DiffServ. As a basic building block of the DiffServ proposal, markers play an important role in achieving the service goals of DiffServ [1]. Research on DiffServ markers [7, 12] has shown that the choice of markers and their settings have a significant effect on the performance of DiffServ.

In this paper, we incorporate the effect of token bucket marker settings in our proposed model. We develop a simple, yet accurate, DiffServ performance model that incorporates the two required parameters of a token bucket marker, viz. the *CIR* and maximum allowed burst size (*B*). In contrast to previous studies, our proposed model integrates the effect of token bucket marker settings, packet dropping probabilities in the core network, and the number of flows in a traffic aggregate. We focus on modelling the performance of a DiffServ network with a token bucket marker in the presence of multiple flows in an aggregate. Based on our proposed model and simulations, we find that the performance of a DiffServ with a token bucket marker under TCP congestion control algorithm is flow based rather than the desired aggregate based. This leads to unfairness problems among sources having different number of flows per aggregate.

2 Proposed model

We first describe the assumptions and notation used in our model.

2.1 Assumptions

In this paper, we assume that the loss in the DiffServ network is low to medium and is periodic. Periodic loss means that whenever the TCP congestion control window reaches some maximum window size, a packet loss occurs. We also assume constant round-trip times for packets. It is well known [8, 9] that under these assumptions, the TCP congestion control window follows a perfect saw-tooth like pattern as shown in Fig. 1. Although this assumption may not be true in the real world, simulation results in Section 3 show good accordance with results from the model, and our proposed model is simple and intuitive. It should be noted that this assumption has also been used and validated by previous researchers [6, 9].

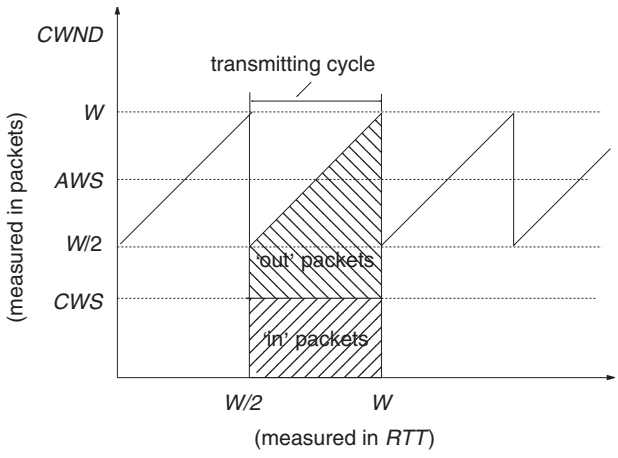

Fig. 1 *Congestion window in largely over-provisioned network*

We also assume all TCP flows from a source go to the same destination (i.e. all flows belonging to an aggregate experience the same *RTT* and have the same *MSS*. Under this assumption, all flows in an aggregate will share the network resource fairly and will encounter the same loss.

The bandwidth achieved by an aggregate emitting n flows with different *RTT* and *MSS* has been shown [6, 9] to be given by

$$\sum_{i=1}^{n} \frac{(MSS_i)c}{(RTT)_i\sqrt{p_i}}$$

where n is the number of flows from a source, p_i is the packet drop probability in the network for the ith flow, and $c = \sqrt{3/2}$ for periodic loss. Therefore, under our assumption of all flows having the same *RTT* and *MSS*, the bandwidth of an aggregate is given by

$$BW = \frac{n(MSS)c}{(RTT)\sqrt{p}} \tag{2}$$

2.2 Notations

We use the following notation in our model.

Flow, connection: a flow or connection means a TCP connection going from a TCP source to a TCP destination. There may be multiple flows or connections going from a source to a destination

Aggregate: aggregate refers to the collection of all flows/connections going from the same TCP source to the same TCP destination

CWND: the TCP congestion window size measured in packets

W: maximum congestion window size measured in packets (see Fig. 1) for a TCP connection. We assume that whenever $CWND = W$, a packet loss occurs and *CWND* is reduced to $W/2$, after which *CWND* increases by one segment per *RTT*. It will therefore take $W/2$ time units (measured in *RTT*) for *CWND* to reach *W* again

BW: achieved bandwidth, in bytes per second (byte/s). This represents the average bandwidth achieved by an aggregate

MSS: maximum segment size in bytes

AWS: average congestion window size of a TCP connection, measured in packets. It is defined as the average value of the congestion window size, and reflects the average bandwidth obtained by a TCP flow. It can therefore be expressed as $(BW)(RTT)/MSS$ for an aggregate with single flow. From Fig. 1, it can also be expressed as $(3/4)W$. Therefore

$$AWS = \frac{(BW)(RTT)}{MSS} = \frac{3}{4}W \tag{3}$$

Transmitting cycle: a cycle during which *CWND* goes from $W/2$ to *W* (see Fig. 1). In each cycle, there are $(W/2)^2 + (1/2)(W/2)^2 = (3/8)W^2$ packets being transmitted. Also we know that there are $1/p$ packets being transmitted in one cycle due to the periodic loss assumption [6]. So we obtain

$$W = \sqrt{\frac{8}{3p}} \tag{4}$$

CIR: committed information rate, in bytes per second, is the desired bandwidth of an aggregate. It is also used as the token arrival rate of the token bucket marker

CWS: committed window size for a TCP flow, in packets, which is defined as

$$CWS = \frac{(CIR)(RTT)}{MSS} \tag{5}$$

for an aggregate with single flow. It is the equivalent of CIR in TCP congestion control window diagrams

P_{MIn}, P_{MOut}: in and out packets marking probabilities, respectively. They represent the probability that an 'in' or 'out' packet will be marked by the token bucket marker. For the 'in'/'out' packet marking scheme, $P_{MIn} = 1 - P_{MOut}$

B: token bucket size measured in bytes

B_p: token bucket size measured in packets, i.e.

$$B_p = \frac{B}{MSS} \tag{6}$$

p_i, p_o: 'in' and 'out' packet dropping probability, respectively, in the core network. They can be measured by many network tools

p: total packet loss probability in the core network. This includes loss of both low and high priority packets. The total packets loss probability in the core network can be expressed as

$$p = (P_{MIn})\,p_i + (P_{MOut})\,p_o = p_i + P_{MOut}(p_o - p_i) \tag{7}$$

Maximum effective token bucket size: minimum of the maximum possible accumulated tokens in one transmitting cycle and the token bucket size.

Since we assume that the network resource is fairly shared among all flows in an aggregate, therefore for an aggregate with n flows, (3), (5) and (6) can be rewritten for individual flows as

$$AWS = \frac{(BW)(RTT)}{n(MSS)} = \frac{3}{4}W \tag{8}$$

$$CWS = \frac{(CIR)(RTT)}{n(MSS)} \tag{9}$$

$$B_p = \frac{B}{n(MSS)} \tag{10}$$

2.3 Model

The main task in developing our model is to express the packet loss probabilities P_{MIn} and P_{MOut} as a function of CIR and B, i.e. finding the relationship between the packet loss probability and the token bucket parameters. To help us find this relationships, we divide the congestion window into four regions depending on the value of CWS in Fig. 1.

(i) Region $CWS < W/2$: By substituting (8) and (9) into it, we can express CIR for this region as $CIR < 2(BW)/3$. This corresponds to a network that is largely over-provisioned, with the total CIR much less than the available bandwidth. We call this network 'largely over provisioned'.

(ii) Region $W/2 \le CWS \le AWS$: Again, with the help of (8) and (9), we find that this boundary condition is equivalent to $2(BW)/3 \le CIR \le BW$. Clearly, the network is operating almost at its full capacity. We call it a 'nearly provisioned network'.

(iii) Region $AWS \le CWS \le 4(AWS)/3$: For this region, $BW \le CIR \le 4(BW)/3$. The level of network provisioning is slightly higher than the available bandwidth; we call it a 'slightly under provisioned network'.

(iv) Region $CWS > 4(AWS)/3$: we have $CIR > 4(BW)/3$, which means that the network is largely under provisioned. We call it a 'largely under-provisioned network'.

We will now derive our model for all these four regions.

2.3.1 Largely over-provisioned network ($CWS < W/2$):

As shown in Fig. 1, due to the fact that CWS is less than $W/2$ (the minimum window size) in this regision, it is easy to see that the token bucket is always empty under previous assumptions (recall that token arrival rate at the marker is usually set to CIR). So, in this region, the token bucket size setting B does not have any effect on the achieved bandwidth, and all packets above CWS will be marked as 'out' packets. Note that the probability that a packet is marked as an 'out' packet is given by the ratio of all packets marked as 'out' sent during a cycle to the total number of packets sent during that cycle. Therefore, the probability of a packet being marked as 'out' in this region is (see Fig. 1 for reference)

$$P_{MOut} = \frac{(3/8)(W^2) - (CWS)(W/2)}{(3/8)(W^2)}$$

$$= 1 - \frac{4(CWS)}{3(W)} \tag{11}$$

Substituting this, together with (4) into (7) and solving for $1/\sqrt{p}$, we obtain (note that we only need the positive root)

$$\frac{1}{\sqrt{p}} = \left(\frac{CWS}{2c}\right)\left(\frac{p_o - p_i}{p_o}\right) + \sqrt{\left(\left(\frac{CWS}{2c}\right)\left(\frac{p_o - p_i}{p_o}\right)\right)^2 + \frac{1}{p_o}} \tag{12}$$

where $c = \sqrt{3/2}$. Substituting, this together with (9), into (2), and after simplification, we obtain

$$BW = \left(\frac{CIR}{2}\right)\left(\frac{p_o - p_i}{p_o}\right) + \sqrt{\left(\left(\frac{CIR}{2}\right)\left(\frac{p_o - p_i}{p_o}\right)\right)^2 + \left(\frac{n(MSS)c}{(RTT)\sqrt{p_o}}\right)^2} \tag{13}$$

Note that in this region, the dropping probability of 'in' packets in the DiffServ core is very small because of the low network load. If we assume $p_i = 0$, we can simplify (13) to

$$BW = \frac{CIR}{2} + \sqrt{\left(\frac{CIR}{2}\right)^2 + \left(\frac{n(MSS)c}{(RTT)\sqrt{p_o}}\right)^2} \tag{14}$$

2.3.2 Nearly provisioned network ($W/2 \le CWS \le AWS$):

In this region, for each cycle, if the window size is less than the CWS, all packets will be marked as 'in' packets, and tokens can be accumulated in the token bucket for future usage. However, if the window size is greater than CWS, we assume that all the committed part of the traffic will be marked as 'in' packets. If there are enough tokens accumulated in the bucket, the excess packets will be marked as 'in' packets; otherwise, they will be marked as

'out' packets. Therefore, the setting of B has an effect on BW in this region. Note that, during a transmitting cycle, tokens can only be accumulated in the bucket if $CWND < CWS$. Therefore, the maximum possible accumulated tokens for a cycle can be given by $(1/2)(CWS-W/2)^2$ as shown in Fig. 2. If $B_p \geq (1/2)(CWS-W/2)^2$, the excess bucket size ($B_p-(1/2)(CWS-W/2)^2$) has no effect on the result. We further subdivide this region into two small regions depending on the value of B_p.

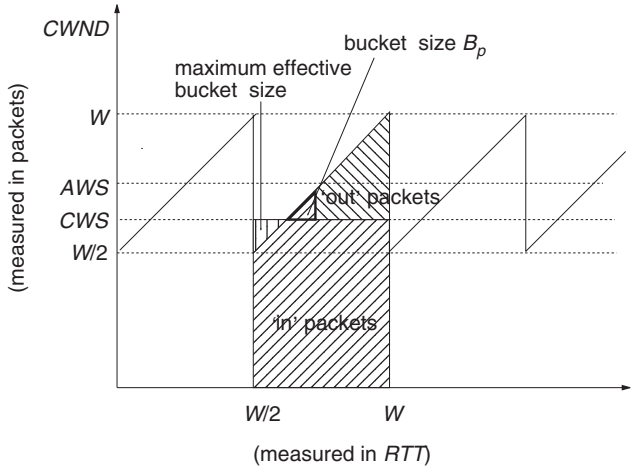

Fig. 2 *Congestion window in nearly-provisioned network*

Case 1: $0 \leq B_p \leq (1/2)\ (CWS-W/2)^2$: In this region, the probability of packets being marked as 'out' packets is given by the ratio of packets exceeding $CWS-B_p$ and the total amount of transmitted packets in a cycle, i.e.

$$P_{MOut} = \frac{(1/2)(W-CWS)^2 - B_p}{(3/8)W^2} \tag{15}$$

Using similar technique as in Section 2.3.1, we can obtain BW as:

$$BW = \frac{2(CIR)}{b} + \sqrt{3\left[\frac{n(MSS)B}{(RTT)^2 b} - \frac{p_i(CIR)^2}{b^2(p_o-p_i)} + \left(\frac{n(MSS)}{RTT\sqrt{b(p_o-p_i)}}\right)^2\right]} \tag{16}$$

where $a = \sqrt{8/3}$ and $b = 2(p_i)/(p_o-p_i) + a^2$.

In this region, it is still reasonable to assume that $p_i = 0$. Substituting it into (16), we obtain a rather intuitive equation

$$BW = \frac{3}{4}(CIR) + \frac{3}{4}\sqrt{2\left[\frac{n(MSS)B}{(RTT)^2} + \left(\frac{n(MSS)}{(RTT)\sqrt{p_o}}\right)^2\right]} \tag{17}$$

The above equation will be discussed further in Section 4.

Case 2: $B_p > (1/2)(CWS-W/2)^2$: Using similar reasoning as in case 1 above, in this subregion, regardless of the size of B_p, the effective bucket size is $(1/2)(CWS-W/2)^2$ packets; therefore, the probability of a packet being marked as Out is

$$P_{MOut} = \frac{\frac{1}{2}(W-CWS)^2 - \frac{1}{2}(CWS-\frac{W}{2})^2}{(3/8)W^2} \tag{18}$$

As before, we can obtain the BW as

$$BW = \frac{CIR}{2}\frac{p_o-p_i}{p_o} + \sqrt{\left(\frac{CIR}{2}\frac{p_o-p_i}{p_o}\right)^2 + \left(\frac{n(MSS)c}{(RTT)\sqrt{p_o}}\right)^2} \tag{19}$$

2.3.3 Slightly under-provisioned network (AWS $\leq$ CWS $\leq$ (4/3)AWS):

As shown in Fig. 3, we also divide this region into two subregions depending on the value of B_p.

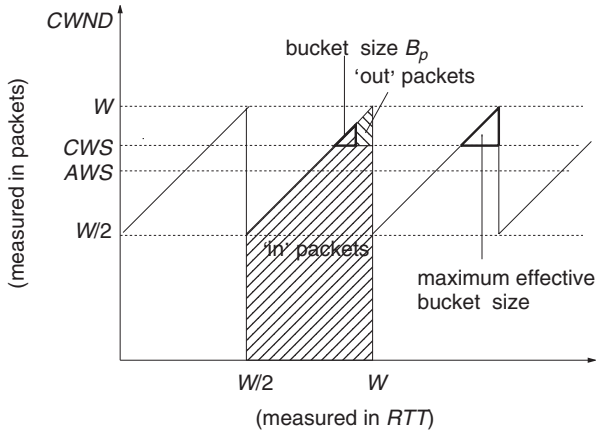

Fig. 3 *Congestion window in slightly under-provisioned network*

Case 1: $0 \leq B_p \leq (1/2)(W-CWS)^2$: In this region, the probability of packets being marked as 'out' packets is the ratio of packets exceeding $CWS-B_p$ and the total amount of transmitted packets in one cycle, i.e.

$$P_{MOut} = \frac{\frac{1}{2}(W-CWS)^2 - B_p}{(3/8)W^2} \tag{20}$$

Using similar technique as used in Section 2.3.1, we can obtain BW as

$$BW = \frac{2(CIR)}{b} + \sqrt{3\left[\frac{n(MSS)B}{(RTT)^2 b} - \frac{p_i(CIR)^2}{b^2(p_o-p_i)} + \left(\frac{n(MSS)}{(RTT)\sqrt{b(p_o-p_i)}}\right)^2\right]} \tag{21}$$

where $a = \sqrt{8/3}$ and $b = 2p_i/(p_o-p_i) + a^2$. In contrast to earlier cases, in this region, we cannot assume that $p_i = 0$ because the loss probability of 'in' packets is large.

Case 2: $B_p > (1/2)(W-CWS)^2$: For this subregion, all packets will be marked as 'in' packets. Therefore, the probability of packets being marked as 'out' is

$$P_{MOut} = 0 \tag{22}$$

so, $p = p_i$. In this case, neither CIR nor B_p has any effect on achieved bandwidth; the network is indeed a best effort network. The BW is (see (2))

$$BW = \frac{n(MSS)c}{(RTT)\sqrt{p_i}} \tag{23}$$

2.3.4 Largely under-provisioned network (CWS > (4/3)(AWS)):

For this region, all packets will be

marked as 'in' packets as shown in Fig. 4. The probability of packets being marked as 'out' is $P_{MOut} = 0$, so, $p = p_i$ for this case. Neither *CIR* nor B_p has any effect on the achieved bandwidth; the network is indeed a best effort network. The *BW* is given by (see (2))

$$BW = \frac{n(MSS)c}{(RTT)\sqrt{p_i}} \tag{24}$$

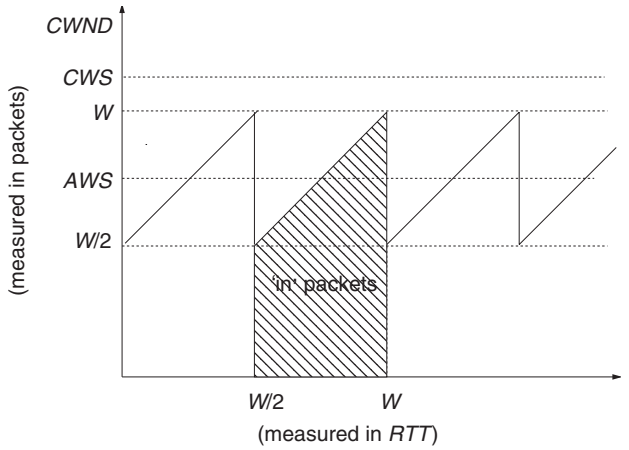

Fig. 4 *Congestion window in largely under-provisioned network*

This is the same formula used in modelling the current best effort network [9], i.e. the DiffServ network losses it ability to provide service differentiation.

3 Simulation validation and results

In this Section, we will use simulation to validate the accuracy of our model. A set of simulations have been performed using the NS-2 network simulator [13] with a DiffServ patch from the Nortel Networks (which is now part of the NS-2 package). Figure 5 shows the simulation topology used in this study. There are two TCP sources: one sends traffic from source 0 to destination 0, the other sends traffic from source 1 to destination 1, which we call 'aggregate 0' and 'aggregate 1', respectively. E0, E1, E2 are edge routers. E0 and E1 are responsible for monitoring and marking traffic aggregate 0 and 1, respectively. Both of them use a token bucket marker to mark the traffic. Token arrival rates are set to their *CIR*s, respectively, while the token bucket size is set to 30 000 bytes for both markers. Choosing the above value of the bucket size ensures that the network goes through all the cases mentioned in Section 2.3 when *CIR* varies between 1 Mbyte/s and 13 Mbyte/s.

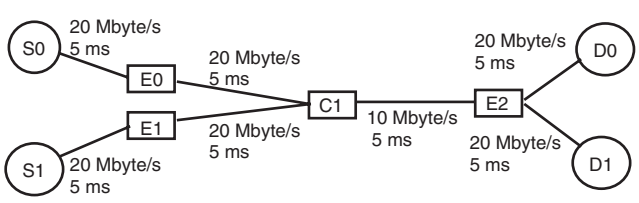

Fig. 5 *Simulation topology*

In Fig. 5, C1 is the core router which implements the RIO scheme. We use the notation $\{x, y, z\}$ to represent minimum threshold, maximum threshold and weight parameter, respectively, of a RED queue [4]. The setting of the RIO router is {20, 40, 0.1} and {40, 80, 0.02} for the 'out' and in packets, respectively, for the given set of results. We use SACK TCP [14] because it helps to keep the TCP operating in the congestion avoidance phase even with a higher loss rate. Also, in this study, all TCP flows are long lived FTP applications with $MSS = 1500$ bytes and $RTT = 40$ ms. In the rest of this Section, we present results for two cases: single flow and multiple flow, depending on whether an aggregate has one or multiple flows.

3.1 Single flow case

In this simulation, each aggregate has only one flow. The *CIR* for aggregate 1 is fixed at 1 Mbyte/s, while the *CIR* for aggregate 0 increases from 1 Mbyte/s to 13 Mbyte/s. This corresponds to the network going from over-provision to under-provision. Figure 6 shows the achieved bandwidth against *CIR* for this simulation. In addition to the measured bandwidth and the estimated bandwidth from our proposed model for both aggregates, we have included information regarding *CIR* of aggregate 0, and 2/3 and 4/3 of measured bandwidth for aggregate 0 in the Figure to ease the understanding. To help the reader to associate the simulation results with our proposed model given in Section 2.3, we divide Fig. 6 into six regions (R1–R6) depending on the relationship between *CIR*, *BW* and *B*.

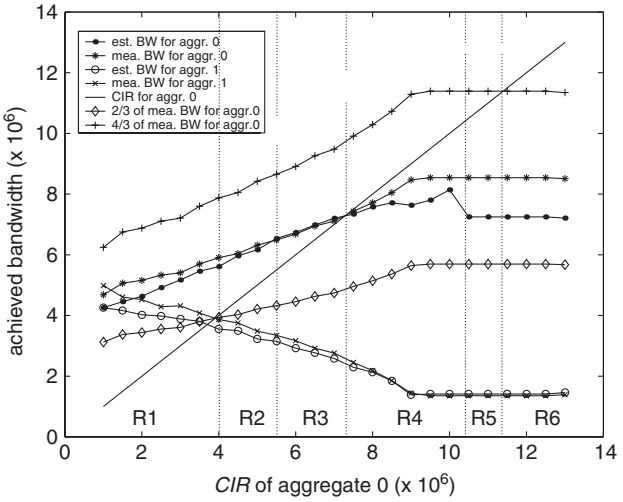

Fig. 6 *Achieved bandwidth against CIR of aggregate 0 in single flow case*

3.1.1 Aggregate 0: For aggregate 0, we divide Fig. 6 into six regions as below.

Region R1: In this region, *CIR* of aggregate 0 is between 1 and 4 Mbyte/s. We can see that *CIR* is less than two-thirds of the measured bandwidth. The estimated bandwidth is calculated using (13). Although the results from simulation and model differ slightly, they are close.

Region R2: For this region, $4 < \text{CIR} \leq 5.5$ Mbyte/s. *B* is greater than the maximum effective token bucket size, and the estimated bandwidth is calculated using (19). The simulation and modelling results are close.

Region R3: With $5.5 < CIR \leq 7.4$ Mbyte/s, the network is near provisioned and *B* is less than maximum effective token bucket size. Equation (16) has been used to calculate the estimated bandwidth. The results from simulation and model are very close. This means that by taking *B* into consideration, our model gives a very good estimation of the achievable bandwidth in a DiffServ network.

Region R4: For $7.4 < \text{CIR} \leq 10.4$ Mbyte/s, model and simulation results begin to diverge slowly because some of the modelling assumptions (for example, the low to medium packet dropping probability assumption) no longer hold. This corresponds to a slightly under-provisioned network with a small token bucket case. As *CIR* increases, the amount of traffic approaches the limit of the bottleneck link, and the dropping probability increases.

Regions R5 and R6: For $CIR > 10.4$ Mbyte/s, both the estimated and measured bandwidth stop increasing. This corresponds to either the slightly under-provisioned network with a large token bucket case or the largely underprovisioned case. In these two regions, the network load exceed its capacity, all packets are 'in' packets and the network looses its ability to provide service differentiation.

3.1.2 Aggregate 1: From Fig. 6, we can see that even at a rather high level of congestion, aggregate 1 still achieves its service rate even though aggregate 0 does not achieve its contracted rate. It indicates that DiffServ network favors small service profile subscribers. This causes an unfairness problem as will be discussed in Section 4.

3.2 Multiple flows case

In this simulation, aggregates 0 and 1 have four and one flows, respectively. The *CIR* for aggregate 1 is still fixed at 1 Mbyte/s, while *CIR* for aggregate 0 increases from 1 Mbyte/s to 13 Mbyte/s. Therefor, the network goes from over-provisioned to under provisioned. Figure 7 shows the achieved bandwidth against *CIR* of aggregate 0 for this simulation.

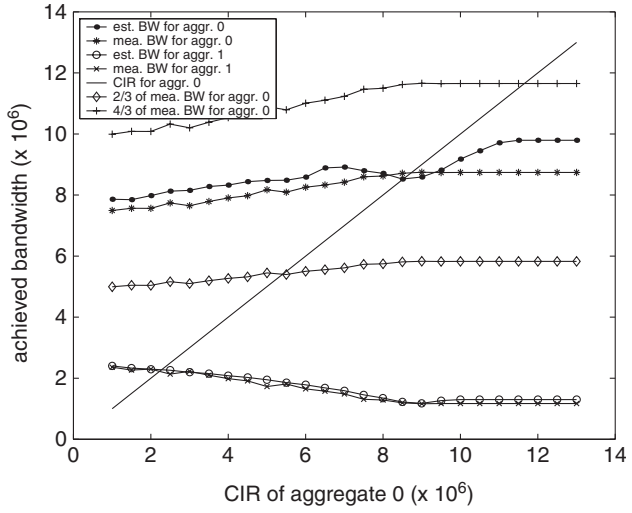

Fig. 7 *Achieved Bandwidth plotted against CIR of aggregate 0 in the multiple flows case*

As in the single flow case, results from simulation and model are close to each other even at a rather high load. This proves the validity of our proposed model. It should be noted that when *CIR*s of aggregate 0 and 1 are all 1 Mbyte/s, the bandwidth gained by aggregate 0 is about four times that gained by aggregate 1. Clearly, this is because of the fact that aggregate 0 has four flows, while aggregate 1 has only one. This indicates that the performance of DiffServ is flow based rather than aggregate based for a largely over-provisioned network.

4 Observation

Several simulations with different sets of RIO parameters and token bucket settings were run. The results of these simulations were similar to those given in Section 3. These simulations validate the accuracy of our model. In choosing the combined parameter settings, we should try to make sure that the modelling assumptions made in Section 2.1 are still valid, otherwise, the results from simulation and model may not be comparable. We now present some observations based on the proposed model.

Observation 1: For largely over-provisioned network, all aggregates can achieve their profiles, and the excess bandwidth is shared among aggregates depending on the number of flows in the aggregates, *MSS*, *RTT*, etc.

This can be explained by (14), which is copied here for easy reference. In this case we assume that $p_i = 0$.

$$BW = \frac{CIR}{2} + \sqrt{\left(\frac{CIR}{2}\right)^2 + \left(\frac{n(MSS)c}{(RTT)\sqrt{p_o}}\right)^2} \qquad (25)$$

We see that for a largely overprovisioned network, each aggregate can achieve its service rate *CIR*. The term $n(MSS)c/(RTT)\sqrt{p_o}$ reflects the amount of excess bandwidth that will be obtained by an aggregate. Clearly, an aggregate with more flows (higher value of n) will be able to gain more excess bandwidth. For the same values of *RTT* and *MSS*, the excess bandwidth is equally shared among all the flows (considering all the aggregates) instead of aggregates. Simulation of multiple flows case (see Section 3.2) verified this observation, where at $\text{CIR} = 1$ Mbyte/s for aggregate 0, with the same *RTT* and *MSS*, the excess bandwidth obtained by aggregate 0 (with four flows) is about four times that of aggregate 1 (with only one flow). Because the service charge in a DiffServ network is applied to aggregates rather than flows, this creates unfairness among the subscribers. Subscribers paying the same charge should equally share the excess bandwidth regardless of the number of flows in their aggregates.

Observation 2: For a near provisioned network with a small token bucket size, about three quarters of the *CIR* is not sensitive to the parameter settings, the other part depends on parameter settings, such as *B*, *RTT*, packet loss probability, etc.

This is drawn from (17). Here, we still assume that the packet dropping probability of 'in' packets is small. Equation (17) is copied here for easy reference

$$BW = \frac{3}{4}CIR + \frac{3}{4}\sqrt{2\left[\frac{n(MSS)B}{(RTT)^2} + \left(\frac{n(MSS)}{(RTT)\sqrt{p_o}}\right)^2\right]} \qquad (26)$$

For all aggregates having the same settings, aggregates with more flows will acquire more of the excess bandwidth. This equation also shows that only 3/4 of the source's desired service rate is guaranteed; the rest depends on the parameter settings. So, there is a possibility that an aggregate with small number of flows will not be able to achieve its service rate, while an aggregate with a large number of flows will receive more bandwidth than its subscription. This causes an unfairness problem. In this region, increasing *B* will slightly increase the achieved bandwidth.

Observation 3: There is a trade-off between the token bucket size (*B*) and the probability of packet dropping. Once *B* exceeds some threshold, an aggregate can not get any more bandwidth by further increasing *B*.

As discussed in Section 2.3.2, there is a maximum possible accumulative token threshold for each congestion

avoidance cycle. Once B reaches this threshold, any further increase in B does not have any effect on the achieved bandwidth. Note that the token bucket size is indeed the maximum allowed burst size. As B increases, the traffic becomes more bursty and increases the probability of packet dropping at the core router, which in turn leads to a reduction in the achieved bandwidth. There is, therefore, a balance between the setting of B and the probability of packet drop in the core network. This agrees with the simulation results in [7].

Observation 4: The performance of DiffServ is flow-based rather than the desired-aggregate-based. With same settings, an aggregate with more flows acquires more bandwidth than the one with fewer flows. The excess bandwidth is distributed in proportion to the number of flows in an aggregate instead of *CIR*s of that aggregate.

This is evidenced in (25) and (26). An aggregate with more flows will acquire more excess bandwidth than aggregates with fewer flows, and the excess bandwidth is shared among flows rather than aggregates. As discussed in observation 1, this is not the desired behaviour. The underlying reason for this is that the excess bandwidth is acquired by sending 'out' packets. While the rate of sending 'in' packets is aggregate-based, the rate of sending 'out' packets is largely flow based. All these 'out' packets experience the same best effort service, causing the excess bandwidth to be shared among flows instead of aggregates.

Observation 5: For a slightly under-provisioned network with a relative large B or a largely under provisioned network, the DiffServ network loses its ability to provide service differentiation due to the fact that all packets are 'in' packets. The DiffServ network reduces to a best effort network. Large subscribers lose their services guarantee, however, the small service subscribers may still be able to achieve their services.

For a DiffServ network operated in this situation, all the promises of the DiffServ are broken. All packets are 'in' packets and receive the same service that is similar to what in the current internet. It is clear that a well provisioned network is the precondition for service differentiation.

5 Conclusion

We have proposed a new comprehensive model that, unlike previous models, takes the two main components of a DiffServ network; the marker at the edge network and the queue at the core network; into consideration. The effect of flows is also considered in the model. We derived an intuitive relationship among the achieved bandwidth and the *CIR*, maximum burst size or token bucket size B, drop probability of in/out packets, number of flows in aggregates, and the *RTT*.

Simulations have been performed to validate the accuracy of our model. Results from our model proved to be very close to results from simulation, for both single flow and multiple flow cases. Several observations were made based on our model. Our model can help network engineers to understand the behaviour of a DiffServ network.

To alleviate the unfairness problem of DiffServ, we strongly believe that new mechanisms, such as aggregate based markers, have to be added to the DiffServ suite to improve the network performance.

6 References

1 Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., and Weiss, W.: 'An architecture for differentiated services'. RFC 2475, December 1998

2 Heinanen, J., and Guerin, R.: 'A two rate three color marker'. RFC 2698, September 1999

3 Heinanen, J., and Guerin, R.: 'A single rate three color marker'. RFC 2697, September 1999

4 Floyd, S., and Jacobson, V.: 'Random early detection gateways for congestion avoidance', *ACM/IEEE Trans. Netw.*, 1993, **1**, (4), pp. 397–413

5 Clark, D., and Fang, W.: 'Explicit allocation of best effort packet delivery service', *IEEE/ACM Trans. Netw.*, 1998, **6**, (4), pp. 362–373

6 Baines, M., Nandy, B., Pieda, P., Seddigh, N., and Devetsikiotis, M.: 'Using TCP models to understand bandwidth assurance in a differentiated services network'. Nortel Technical Report, July 2000

7 Yeom, I., and Narasimha Reddy, A.L.: 'Realizing throughput guarantees in a differentiated services network'. Proc. ICMCS, Florence, Italy, 7–11 June 1994, Vol. 2, pp. 372–376

8 Jacobson, V., and Karels, M.J.: 'Congestion avoidance and control'. Proc. SIGCOMM'88, Stanford, CA, USA, August 1988, pp. 314–329

9 Mathis, M., Semke, J., Mahdavi, J., and Ott, T.: 'The macroscopic behavior of the TCP congestion avoidance algorithm', *Comput. Commun. Rev.*, 1997, **27**, (1), pp. 67–82

10 Padhye, J., Firoiu, V., Towsley, D., and Kurose, J.: 'Modeling TCP throughput: A simple model and its empirical validation'. SIGCOMM'98, Vancouver, BC, Canada, September 1998

11 Yeom, I., and Narasimha Reddy, A.L.: 'Modeling TCP behavior in a differentiated services network'. Texas A&M Technical Report, May 1999

12 Feng, W., Kandlur, D.D., Saha, D., and Shin, K.G.: 'Adaptive packet marking for providing differentiated services in the internet'. Proc. Int. Conf. on Network protocols, Austin, TX, USA, October 1998

13 'Network simulator 2 (ns-2)'. University of California at Berkeley, CA, USA, available via http://www.isi.edu/nsnam/ns/ 1997

14 Mathis, M., and Mahdavi, J.: 'Forward acknowledgement: refining TCP congestion control'. Proc. SIGCOMM'96, Stanford, CA, USA, August 1996, pp. 281–291