Unbiased RIO: An Active Queue Management Scheme for DiffServ Networks[⋆]

Sergio Herrería-Alonso, Miguel Rodríguez-Pérez, Manuel Fernández-Veiga, Andrés Suárez-González, and Cándido López-García

Departamento de Enxeñería Telemática, Universidade de Vigo
Campus universitario, 36310 Vigo, Spain
sha@det.uvigo.es

Abstract. One of the more challenging research issues in the context of Assured Forwarding (AF) is the fair distribution of bandwidth among aggregates sharing the same AF class. Several studies have shown that the number of microflows in aggregates, the round trip time, the mean packet size and the TCP/UDP interaction are key factors in the throughput obtained by aggregates using this architecture. In this paper, we propose the Unbiased RIO (URIO) scheme, an enhanced RIO technique that improves fairness requirements among heterogeneous aggregates in AF-based networks. We validate URIO behaviour through simulation. Simulation results show how our proposal mitigates unfairness under many different circumstances.

1 Introduction

The Differentiated Services (DiffServ) architecture [1] is one of the most promising proposals to address Quality of Service (QoS) issues for data and multimedia applications in IP networks. The differentiated service is obtained through traffic conditioning and packet marking at the edge of the network combined with simple differentiated forwarding mechanisms at the core.

DiffServ defines a set of packet forwarding criteria called "Per Hop Behaviours" (PHBs). One of these packet-handling schemes standardised by the IETF is the Assured Forwarding (AF) PHB [2]. The basis of assured services is differentiated dropping of packets during congestion at a router. AF provides four classes of delivery for IP packets and three levels of drop precedence per class. Within each AF class, IP packets are marked based on conformance to their target throughputs. The Time Sliding Window Three Colour Marker (TSWTCM) [3] is one of the packet marking algorithms most used to work with AF. At the core of the network, the different drop probabilities can be achieved using the RIO (RED with In/Out) scheme [4], an active queue management technique that extends RED (Random Early Detection) gateways [5] to provide the service differentiation required.

⋆ This work was supported by the project TIC2000-1126 of the "Plan Nacional de Investigación Científica, Desarrollo e Innovación Tecnológica" and by the "Secretaría Xeral de I+D da Xunta de Galicia" through the grant PGIDT01PX132202PN.

Assured services should fulfil the two following requirements [6]:

1. Each aggregate should receive its subscribed target rate on average if there is enough bandwidth available (throughput assurance).
2. The extra unsubscribed bandwidth should be distributed among aggregates in a fair manner (fairness).

Unfortunately, as reported in [7–9], the fairness requirement of assured services cannot be met under some circumstances. Via simulation studies, these works confirm that the number of microflows in aggregates, the round trip time (RTT) and the mean packet size are critical factors for the fair distribution of bandwidth among aggregates belonging to the same AF class. Also, the interaction between responsive TCP traffic and unresponsive UDP traffic may impact the TCP traffic in an adverse manner.

Many smart packet marking mechanisms have been proposed to overcome these fairness issues. Adaptive Packet Marking (APM) [10] is one of these schemes able to provide soft bandwidth guarantees, but it has to be implemented inside the TCP code itself and thus, requires modifying all TCP agents. Intelligent traffic conditioners proposed in [11] handle some of these fairness problems using a simple TCP model when marking packets. However, they require external inputs and cooperation among markers for different aggregates. This feature complicates both implementation and deployment. Another marking algorithm based on a more complex TCP model is Equation-Based Marking (EBM) [12]. This scheme solves the fairness problems associated with heterogeneous TCP flows under diverse network conditions. EBM behaviour depends on the quality of the estimation of the current loss rate seen by TCP flows but, sadly, the calculation of this estimate is not an easy task and involves the deployment of the scheme awfully.

A different approach consists of addressing these problems by enhanced RIO queue management algorithms. One of the more interesting examples of these techniques is Dynamic RIO (DRIO) [6]. This scheme accomplishes per-flow monitoring at the core routers to share the available bandwidth among heterogeneous flows fairly. Since there can be thousands of active flows at the core of the network, DRIO needs to store and manage a great amount of state information and therefore, this renders it unscalable. In this paper, we first extend DRIO to work at the aggregate level. Applying DRIO to aggregated traffic instead of to individual flows reduces considerably the amount of stored information at the core improving scalability substantially. In addition, the mechanisms proposed for DiffServ are derived from a model that considers only aggregated traffic. Therefore, the application of DRIO to aggregates will be more natural and suitable for this kind of networks.

We then propose a new enhanced RIO queue management algorithm, namely Unbiased RIO (URIO), that uses proper level buffer usage policing to solve these fairness issues among heterogeneous aggregates. To validate the proposed schemes, we examine their behaviours under a variety of network conditions through simulation. Our simulation results show that URIO fulfils fairness requirement more satisfactorily than either RIO or DRIO.

The rest of the paper is organised as follows. Section 2 gives a brief overview of TSWTCM, RIO and DRIO schemes. In Section 3, we propose URIO, an enhanced version of RIO. Section 4 describes the simulation configuration used for the evaluation of RIO, DRIO and URIO schemes. In Section 5, we present the results obtained from the simulation experiments. We end the paper with some concluding remarks in Section 6.

2 Background

2.1 The TSWTCM Marker

The TSWTCM packet marking algorithm [3] defines two target rates: the Committed Information Rate (CIR) and the Peak Information Rate (PIR). Under TSWTCM, the aggregated traffic is monitored and when the measured traffic is below its CIR, packets are marked with the lowest drop precedence, *AFx1* (green packets). If the measured traffic exceeds its CIR but falls below its PIR, packets are marked with a higher drop precedence, *AFx2* (yellow packets). Finally, when traffic exceeds its PIR, packets are marked with the highest drop precedence, *AFx3* (red packets).

2.2 The RIO Scheme

RIO [4] uses the same dropping mechanism as RED, but it is configured with three sets of parameters, one for each drop precedence marking (colour). These different RED parameters cause packets marked with higher drop precedence to be discarded more frequently during periods of congestion than packets marked with lower drop precedence. Incipient congestion is detected by computing a weighted average queue size, since a sustained long queue is an evidence of network congestion. There are several ways to compute the average queue length. For example, in the RIO-C mode, the average queue size for packets of different colours is calculated by adding its average queue size to the average queue sizes of colours with lower drop precedence. That is, for red packets, the average queue size will be calculated using red, yellow and green packets. For the configuration of RIO parameters, [13] recommends the staggered setting (Fig. 1).

2.3 The DRIO Scheme

DRIO [6] was originally proposed to improve fairness among individual flows for AF-based services. Upon arrival of a red packet, DRIO takes effect when the average queue length falls between min_{th} and max_{th} RIO thresholds. In DRIO, a history list of the last red packets seen is maintained. If the history list is not full, the red packet is added to it. Otherwise, two elements are randomly selected from the history list and compared with the red packet. A hit (miss) occurs when the red packet and the selected element (do not) belong to the same flow. Therefore, three outcomes are possible:

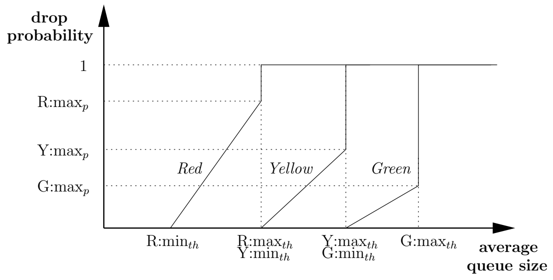


Fig. 1. Staggered setting for RIO

1. If a double hit occurs, the packet is dropped.
2. In the case of one hit and one miss, the packet is dropped with a certain probability. The value of this probability depends on the corresponding flow and it is adjusted accordingly to the number of hits.
3. Finally, if a double miss is obtained, the packet is enqueued and one of the two elements selected from the history list is replaced with it.

The number of hits is a good signal of whether a flow has sent excessive red packets. Thus, DRIO preferably discards packets from flows that have sent more red packets than their fair share. See [6] for a complete description of this scheme.

Albeit DRIO was originally proposed to be applied to individual flows, it can be adapted to work at the aggregate level with a minimum of variations. In fact, only an identification method for classifying packets from different aggregates is required. This method could be based on the value of a combination of several header fields, such as source and destination addresses, source and destination port numbers, protocol identifier, MPLS label, etc.

3 The Unbiased RIO Scheme

In this section, we present the Unbiased RIO (URIO) scheme, a modified version of RIO that improves fairness requirements in AF-based networks. As our extension of DRIO, URIO is also applied to aggregated traffic for scalability reasons. The technique employed by URIO to improve fairness requirement consists of discarding packets from aggregates that are using more than their fair share of the buffer. This technique is based on the one proposed in the Flow Random Early Drop (FRED) [14] scheme to isolate non-adaptive traffic.

3.1 URIO Operations

As in RIO, if the average queue length is less than min_{th}, no arriving packets are discarded. Likewise, when the average queue length is greater than max_{th}, all arriving packets are dropped. In fact, URIO only takes effect when the average queue length falls between min_{th} and max_{th} RIO thresholds. Therefore, URIO just acts when the router is likely to incur congestion.

```
/* Upon arrival of a packet $pkt$ from aggregate $i$ */
if ($avg\_queue\_length > max_{th}$) drop($pkt$)
else if ($avg\_queue\_length > min_{th}$) {
    if ($qbytes_i \leq fairqbytes$) {
        /* Non-greedy aggregate */
        $qbytes_i \leftarrow qbytes_i + pkt\_size$
        enque($pkt$)
    } else if (($pkt$ = UDP) || (rand() < $p_{TCPi}$)) {
        drop($pkt$)
    } else {
        $qbytes_i \leftarrow qbytes_i + pkt\_size$
        enque($pkt$)
    }
} else enque($pkt$)
```

Fig. 2. URIO operations

To implement this scheme, it is necessary to record for each active aggregate the three following variables:

1. The aggregate identifier: as described in the previous section, a combination of several header fields can be used.
2. The number of bytes queued in the buffer by the aggregate (*qbytes*).
3. The timestamp with the last arrival of a packet from the aggregate.

We also need to estimate the fair share of the buffer that corresponds to each active aggregate. An easily computable estimation is the fair number of queued bytes per aggregate (*fairqbytes*). This value is calculated dividing the total number of bytes queued in the buffer by the current number of active aggregates (n_a):

$$fairqbytes = \frac{\sum_{i=1}^{n_a} qbytes_i}{n_a} \ . \tag{1}$$

The operations of URIO are shown in Fig. 2. Upon arrival of a packet belonging to an aggregate i, URIO reads the number of bytes already queued by this aggregate. If $qbytes_i$ exceeds *fairqbytes*, we consider that the aggregate is being greedy and, therefore, we should decide whether to accept the incoming packet. TCP packets are discarded with some probability p_{TCP} proportional to the greed degree of the corresponding aggregate. However, packets from UDP flows are always discarded. UDP traffic is treated in a worse manner than TCP traffic since UDP sources are unresponsive to the losses during periods of congestion. This selective dropping allows responsive traffic to send bursts of packets, but prevents greedy traffic from monopolising the buffer. When a packet is finally queued, the *qbytes* variable of the corresponding aggregate must be incremented by the size of the packet. No extra operations must be accomplished when a packet leaves the queue.

In order to adapt to the current network conditions, periodically both the *qbytes* value of all active aggregates is set to 0 and the outdated aggregates are

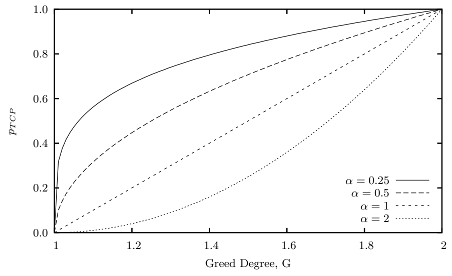


Fig. 3. Dropping probability vs. greed degree

purged based on their timestamp value. If the number of competing aggregates oscillates quickly, this reset timeout should be set to a small value. At the other extreme, if the number of aggregates is relatively constant over time, greater values can be used. The search for optimum values for the reset timeout is a topic for further research, as it does not affect the results presented in this paper.

3.2 Calculation of the p_{TCP} Dropping Probability

TCP packets from aggregates whose *qbytes* does not exceed *fairqbytes* are always queued in the buffer. Otherwise, the probability of dropping a TCP packet (p_{TCP}) should be proportional to the greed degree of the corresponding aggregate: the greater the greed degree of the aggregate, the higher the dropping probability. We define the greed degree G of an aggregate i as:

$$G_i = \frac{qbytes_i}{fairqbytes} \ . \tag{2}$$

Greedy aggregates verify that $G > 1$, so the p_{TCPi} probability can be obtained in the following manner:

$$p_{TCPi} = \min\left\{(G_i - 1)^{\alpha}, 1\right\} \ , \qquad \alpha > 0 \ , \tag{3}$$

where the α parameter controls the dropping probability in a inversely proportional manner. Figure 3 shows how the p_{TCP} probability varies with the greed degree for different values of α. The α parameter is a compromise: smaller values of α improve fairness at the expense of increasing packet loss ratios. Experimentally, we have found that the fairness condition is fulfilled satisfactorily with a moderate packet loss ratio when $\alpha = 0.5$. This result can be explained by the combined effect of the power-law dropping probability enforced by the router, coupled with the multiplicative-decrease reaction of TCP upon packet losses. For $\alpha < 1$, this leads to a sub-linear decrease of the sending rate only for the greedy sources that improves fairness.

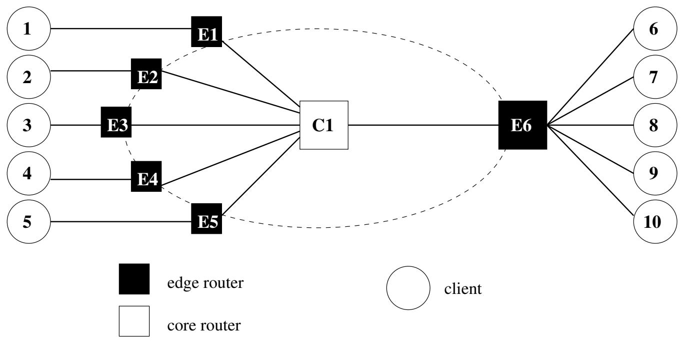


Fig. 4. Network topology. Each link has a 10 Mbps capacity and a 1 ms delay

4 Simulation Configuration

Figure 4 shows the network topology employed in simulations. We consider five competing aggregates sharing the same AF class: aggregate 1 runs between client 1 and client 6, aggregate 2 runs between client 2 and client 7, and so on. Therefore, all aggregates pass through a single bottleneck link between nodes C1 and E6. All TCP connections established are bulk data transfer and use the NewReno implementation. The packet size is set to 1000 bytes.

The marking scheme used in the edge routers is TSWTCM. Each edge node uses a single-level RED queue. The RED parameters $\{min_{th}, max_{th}, max_p, w_q\}$ used are {30,60,0.02,0.002} and the queue size is set to 100 packets.

The core node implements RIO, DRIO and URIO mechanisms. Three sets of RED thresholds are maintained, one for each drop precedence (Table 1). The physical queue size is capped at 250 packets and w_q equals 0.002. The average core queue size is calculated as in the RIO-C mode.

Each aggregate has the same target: a 0.5 Mbps CIR and a 1.0 Mbps PIR. This subscription level implies a light load network condition. The fair bandwidth that should be allocated for each aggregate is given by:

$$\text{target rate} + \frac{\text{extra bandwidth}}{\text{number of aggregates}} \; . \tag{4}$$

Therefore, ideally all the aggregates will obtain a throughput of 2 Mbps.

All the simulations were performed with ns-2 [15]. We run the simulations for 100 seconds. We measure the average throughput achieved by each aggregate

Table 1. Core RED parameters

Precedence	min_{th} *(pkts)*	max_{th} *(pkts)*	max_p
AFx1 (green packets)	200	240	0.02
AFx2 (yellow packets)	160	200	0.06
AFx3 (red packets)	40	160	0.12

over the whole simulation period. Each experiment is repeated 10 times, and then an average and a confidence interval with 95% confidence level are taken over all runs.

5 Simulation Results

Five experiments have been simulated to study the performances of the proposed schemes. The fairness issues examined are the effect of differing number of microflows in an aggregate, the effect of differing RTTs, the effect of differing packet sizes, the impact of TCP/UDP interaction and the effect of different target rates.

5.1 Impact of Number of Microflows

Each microflow represents a single TCP connection. In this test, each aggregate contains a different number of TCP flows that varies from 5 to 25. Under RIO, the aggregate with a larger number of microflows obtains a greater share of the bandwidth. With both DRIO and URIO, each aggregate obtains an equal amount of bandwidth (Fig. 5(a)).

5.2 Impact of RTT

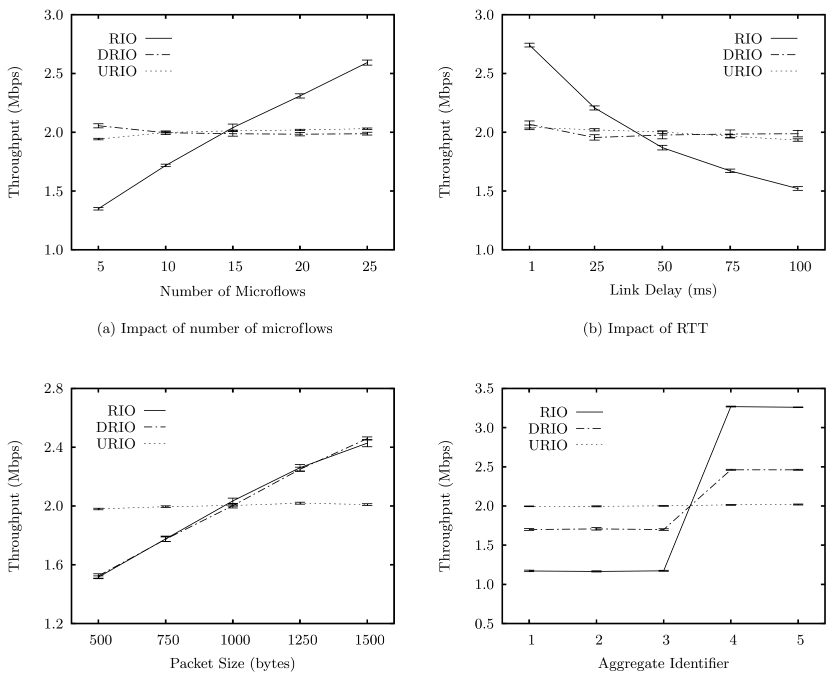
In this test, each aggregate comprises 10 TCP flows. In order to modify the RTT of the aggregates, we consider that the delay of each access link (the link that joins a client with its corresponding edge router) is different, varying from 1 to 100 ms. Under RIO, aggregates cannot achieve its proportional share of bandwidth. With both DRIO and URIO, this bias is fully overcome (Fig. 5(b)).

5.3 Impact of Packet Size

In this test, all the aggregates contain 10 TCP flows but each one has a different packet size, varying from 500 to 1500 bytes. Through RIO, the aggregate that is sending larger packets consumes more of the available bandwidth. DRIO cannot correct this bias either since it does not take into account that packets may have different sizes. Only URIO can make the sharing of bandwidth insensitive to the packet sizes of the aggregates (Fig. 5(c)).

5.4 TCP/UDP Interaction

In this test, there are two groups of aggregates. Aggregates 1, 2 and 3 comprise 10 TCP flows while aggregates 4 and 5 contain one UDP flow with a sending rate of 5 Mbps. With RIO, UDP traffic consumes more than its available throughput impacting TCP traffic negatively. Under DRIO, this bias is only slightly mitigated. As in the previous test, only URIO can completely isolate UDP traffic and share the extra bandwidth in a TCP-friendly manner (Fig. 5(d)).



(a) Impact of number of microflows (b) Impact of RTT

(c) Impact of packet size (d) TCP/UDP interaction

Fig. 5. Fairness evaluation

5.5 Impact of Target Rate

The previous experiments illustrate how URIO shares the extra bandwidth in a uniform manner among aggregates with equal traffic profiles. In a realistic environment, different users will contract different target rates and therefore, all aggregates will not have the same bandwidth expectations. In this test, each aggregate has different CIR and PIR values. The CIR value varies from 0.5 to 1.5 Mbps and the PIR value is set to the CIR value plus 0.5 Mbps. The expected fair share of bandwidth for each aggregate is obtained using Eq. (4). As shown in Table 2, RIO, DRIO and URIO schemes distribute bandwidth according to the subscribed profiles.

6 Conclusions

In this paper, we have proposed an active queue management scheme called URIO that improves fairness requirements for assured services. Our proposal outperforms RIO and DRIO schemes using a selective discard mechanism to protect aggregates that are consuming less than their fair share. We have evaluated the performance of URIO under diverse network conditions through simu-

Table 2. Impact of target rate

Aggregates		**Throughput** (Mbps)			
Id	*CIR*	*Expected*	*RIO*	*DRIO*	*URIO*
1	0.50 Mbps	1.50	1.51 ± 0.01	1.50 ± 0.01	1.49 ± 0.01
2	0.75 Mbps	1.75	1.75 ± 0.01	1.74 ± 0.01	1.74 ± 0.01
3	1.00 Mbps	2.00	2.00 ± 0.01	2.00 ± 0.01	1.99 ± 0.01
4	1.25 Mbps	2.25	2.24 ± 0.01	2.24 ± 0.01	2.24 ± 0.01
5	1.50 Mbps	2.50	2.47 ± 0.01	2.49 ± 0.01	2.49 ± 0.01

lation. Simulation results confirm that URIO distributes the bandwidth among heterogeneous aggregates in a fair manner.

References

1. Blake, S., Black, D., Carlson, M., Davis, E., Wang, Z., Weiss, W.: An architecture for differentiated services. RFC 2475 (1998)
2. Heinanen, J., Baker, F., Weiss, W., Wroclawski, J.: Assured forwarding PHB group. RFC 2597 (1999)
3. Fang, W., Seddigh, N., Nandy, B.: A time sliding window three colour marker (TSWTCM). RFC 2859 (2000)
4. Clark, D.D., Fang, W.: Explicit allocation of best effort packet delivery. IEEE/ACM Transactions on Networking **6** (1998) 362–373
5. Floyd, S., Jacobson, V.: Random early detection gateways for congestion avoidance. IEEE/ACM Transactions on Networking **1** (1993) 397–413
6. Lin, W., Zheng, R., Hou, J.: How to make assured services more assured. In: Proc. ICNP. (1999)
7. Ibanez, J.A., Nichols, K.: Preliminary simulation evaluation of an assured service. IETF Draft (1998)
8. de Rezende, J.F.: Assured service evaluation. In: Proc. IEEE GLOBECOM. (1999)
9. Seddigh, N., Nandy, B., Pieda, P.: Bandwidth assurance issues for TCP flows in a differentiated services network. In: Proc. IEEE GLOBECOM. (1999)
10. Feng, W.C., Kandlur, D., Saha, D., Shin, K.: Adaptive packet marking for maintaining end-to-end throughput in a differentiated-services internet. IEEE/ACM Transactions on Networking **7** (1999) 685–697
11. Nandy, B., Seddigh, N., Pieda, P., Ethridge, J.: Intelligent traffic conditioners for assured forwarding based differentiated services networks. In: NetWorld+Interop 2000 Engineers Conference. (2000)
12. El-Gendy, M., Shin, K.: Equation-based packet marking for assured forwarding services. In: Proc. IEEE INFOCOM. (2002)
13. Makkar, R., Lambadaris, I., Salim, J.H., Seddigh, N., Nandy, B., Babiarz, J.: Empirical study of buffer management schemes for diffserv assured forwarding PHB. Technical report, Nortel Networks (2000)
14. Lin, D., Morris, R.: Dynamics of random early detection. In: Proc. ACM SIGCOMM. (1997)
15. NS: Network simulator, ns. `http://www.isi.edu/nsnam/ns/` (2003)