Performance analysis for a DiffServ-enabled network: The case of Relative Service

Christos Bouras[1,2], Afrodite Sevasti[1,2]
[1]Research Academic Computer Technology Institute-RACTI, Kolokotroni 3, 26221 Patras, Greece
[2]Dept. of Computer Engineering and Informatics, University of Patras, 26500, Patras, Greece
{bouras, sevastia}@cti.gr

Abstract

A lot of research work has recently focused on the exploitation of the DiffServ framework towards building reliable networking services that provide deterministic quality guarantees. In this work we are attempting to devise a DiffServ-based service model for relative differentiation of traffic in a network serving aggregated traffic. The service model proposed aims at providing a minimum capacity during congestion and a bounded average end-to-end delay to profile-conforming traffic, obtaining thus a relatively better performance when compared to the best-effort service. We provide the framework on which the proposed service model is based and we present results from its deployment in a simulation environment.

1. Introduction

The introduction of the Differentiated Services (DiffServ) framework has been a significant development in the direction of providing differential treatment and Quality of Service (QoS) to IP packets in contemporary backbone networks. DiffServ does not make explicit per-flow resource reservation, and therefore, although scalability is obtained, the quality guarantees provided to IP traffic, by mechanisms introduced so as to obtain differentiation, are compromised. Network dimensioning and appropriate resources' provisioning are therefore key factors for such guarantees to be achieved. DiffServ provides a set of mechanisms used in building Per-Hop-Behaviors (PHBs) and services. The definition and implementation of successful services built according to the Assured Forwarding (AF) PHB, remains a relatively unexplored subject of research.

As explained in [1], a number of parameters affect the use of the differentiation mechanisms used (such as packet marking and RED/WRED/RIO for packet dropping) so that differentation is very hard to become determinstc. A number of solutions have been proposed so far. In [3], an approach to the so–called 'Assured Service' is made, according to which an assurance or at least an estimation of the transmission rate for a flow can be provided with the use of appropriate markers for marking packets as 'in' or 'out' of profile. In [2] it is concluded that differences in RTT for individual flows affect only the capacity observed by these flows individually and that individual flows with a certain contracted capacity receive less capacity than aggregates with the same contracted capacity.

In [7] it is emphasized how important it is for all flows participating in a relative differentiation service schema to encounter the same loss rate. In [8], the effect of RTT on the throughput in the case of relative service differentiation is analysed. In [9] the two major deficiencies of classical RED are identified with respect to achieving fair relative service differentiation: congestion notification obtained is not dependent upon the number of active flows and the benefits of RED are not useful when RED is using packet dropping for congestion notification in contrast to packet marking (ECN).

In [10], the problem of unfairness of bandwidth allocation among TCP flows is solved with the adoption of Core-Stateless Fair Queueing (CSFQ). The proposal made requires a minor change to the TCP received window size adjustment. [11] proposes a number of improvements to the dropping, marking, shaping algorithms used by contemporary DiffServ-based services so that they will become more user friendly. In [12] Random Eearly Management is introduced in order to overcome the problems caused by RED in relative service differentiation.

[13] and a number of related publications are based on classical linear feedback control theory and stochastic differential equation analysis to examine how the closed-loop system of TCP traffic and RED behaves. It is proved that the packet flow within the system queue depends upon the queue length, which does not allow for a steady-state regulation and the closed-loop adjustments are too fast leading to instability.

Throughout the course of this work, it will become obvious that provision of bandwidth guarantees and bounds on quality metrics, such as end-to-end delay, are not trivial to obtain. There are several factors and prerequisites that must hold, most of which are difficult to obtain in realistic conditions. The contribution of this work is the definition and evaluation of an AF-based service, referred to as Relative Service from now on, that

is straightforward to deploy in a transport network. Our aim has been to investigate the methodology and configuration required for the Relative Service in order to obtain reliable transmission rate guarantees to profile-conforming traffic. We also attempt to enhance the quality provided by such a service, by proposing a methodology for the Relative Service deployment that provides guarantees for a bounded average queuing delay for conforming or in-profile packets.

In section 2 of this paper the definition of the proposed Relative Service followed by a description of the queue management mechanism used in section 3 are provided. A brief analysis of the proposed service's framework is outlined in section 4. Section 5 presents our experimental study while section 6 refers to our future steps and the conclusions of our work on the Relative Service.

2. Relative Service definition

Through the definition of the AF PHB, the DiffServ framework anticipates for services that attempt to provide to IP traffic a service that is qualitatively better than that of the traditional IP 'best-effort' model, without deterministic, high-assurance quality guarantees. Our study demonstrates that especially in the case of networks serving aggregated traffic, queue management mechanisms (such as WRED) do not affect the distribution of capacity between differently marked traffic aggregates. However, we claim that these mechanisms can be efficiently exploited in order to control the average occupancy of the queues serving Relative traffic, obtaining thus a reduction on the average queuing delay for Relative traffic packets.

We define Relative Service as one that ensures with very high probability and a bounded average end-to-end delay the delivery of packets for flows or aggregates of flows that conform to a predetermined profile. Although AF-based services do not provide for strict delay/jitter guarantees, since they allow the use of non-reserved resources for the transmission of non-conformant packets when such resources are available, we propose the enhancement of the Relative Service with a quantitative guarantee: a bounded average queuing delay for in-profile packets that translates into a bounded average end-to-end delay for in-profile traffic.

The provision model for Relative Service between a customer (C) of the service and a transport domain (TD) providing the service, is defined on the basis of a Service Level Agreement (SLA) offered from the TD to C as follows (refer to Figure 1):

- The SLA specifies that each aggregate S_i of Relative traffic injected from C to the TD at an ingress interface of TD (such as A) will be policed according to a token bucket (r_i, b_i) policer
- The provision of Relative Service to the aggregate S_i ensures that the in-profile traffic of S_i will be carried through TD up to the point where S_i exits TD (at an egress interface such as C) with a transmission rate that is equal to or larger than r_i
- The SLA guarantees that all in-profile packets of S_i transported from A to C over a period t will face a bounded average delay D metered between points A and C

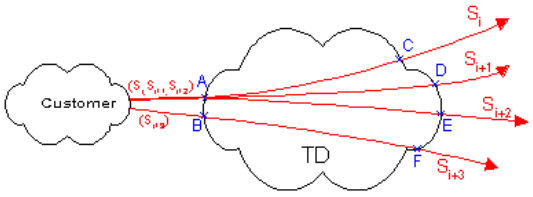


Figure 1. Provision model for the Relative Service

Our proposed service is straightforward to deploy on a backbone domain. Possible uses or applications of the Relative Service would be:

- The provision of an assured transmission rate with minimised packet losses due to congestion between two edges of a domain
- The controlled resources' allocation to flows sharing a congested transmission link

In general, Relative Service can be used by a wide variety of applications. It is definitely not suitable for jitter-sensitive applications, but appropriate provisioning and high-speed connectivity allow the Relative Service to be appropriate for bandwidth demanding applications with inherent buffering to cope with instant fluctuations in delay.

3. Queue management

A significant component of the service model we propose and the AF-based services in general, is that of active queue management. WRED is a queue management mechanism that is mainly used in core backbone routers rather than edge routers. RED (on which WRED is based) uses the average queue occupancy as a parameter in a random function that determines whether the congestion avoidance mechanisms (e.g. packet drops) should be triggered. As the average queue occupancy increases so does the packet drop probability. According to RED, when the average queue occupancy (Q_{avg}) is below a minimum threshold, $\min_{th}$, then no packet is dropped, while when Q_{avg} exceeds $\min_{th}$ the packet drop probability increases linearly up to a maximum value $\max_p$. When

Q_{avg} exceeds $\max_{th}$, then the packet drop probability is equal to 1, thus all packets are dropped. In WRED, packet drop probability for each packet arriving to the queue is calculated according to a single value of Q_{avg}, but the values of $\min_{th}$, $\max_{th}$, $\max_{p}$ differ according to the color that each packet has been colored with.

There are two alternatives for the values of $\min_{th}$, $\max_{th}$, $\max_{p}$ configured for three levels of packet coloring ('green', 'yellow' and 'red'):

- Case 1: $\min_{th}^{green} = \min_{th}^{yellow} = \min_{th}^{red}$ and $\max_{th}^{green} = \max_{th}^{yellow} = \max_{th}^{red}$, with $\max_{p}^{yellow} > \max_{p}^{green}$ and $\max_{p}^{red} > \max_{p}^{yellow}$
- Case 2: $\max_{th}^{green} > \min_{th}^{green} = \max_{th}^{yellow} > \min_{th}^{yellow} = = \max_{th}^{red} > \min_{th}^{red}$ and $\max_{p}^{green} > \max_{p}^{yellow} > \max_{p}^{red}$

Part of our experimental study will investigate the proper configuration of WRED for controlling the average length of a router queue for the purposes of the Relative service.

4. Framework analysis

According to the Relative Service definition already presented, the capacity perceived by the packets of Relative traffic has to be equal to or larger than the capacity guaranteed by the SLA signed between the TD and each one of its Relative Service customers, in the absence of congestion. It should also be equal to the guaranteed capacity during congestion with a very high probability. In both cases, during congestion or in the absence of it, the drops of in-profile or 'green' packets should be minimal.

Depending on the scheduling algorithm used, the router queues that serve Relative traffic in any topology have to be assigned with enough capacity resources to serve at least the contracted capacity of Relative traffic. If Q_R is the queue serving Relative traffic, then for the service rate r of Q_R, the following must hold:

$$r > \sum r_i \qquad (1)$$

with $\sum r_i$ being the sum of the token bucket rate parameters for each of the Relative traffic aggregates ($S_i, S_{i+1}, S_{i+2}, S_{i+3}$ for Figure 1) served through Q_R. In the case of the Cisco Modified Deficit Round Robin (MDRR) scheduling mechanism of the Cisco GSRs ([4]) that was used in our experiments, Relative traffic must be served by queues to which a weight (and quantum) is configured to ensure an adequate value of r in all the routers of a TD offering the service. Similarly, other scheduling mechanisms (e.g. Weighted Fair Queuing) must be configured according to the same principle.

As far as queue management is concerned, we will show how it is possible to bound the average value of the queuing (and thus end-to-end) delay perceived by 'green' packets during congestion. We propose the use of WRED with adjustable $\min_{th}$ and $\max_{th}$ parameters configured in a way that relates the anticipated Relative queue size with the bandwidth-delay product of the TD links.

More specifically, $\max_{th}$ is adjusted (over consecutive intervals) to equal the number of MTU-sized packets that can be transmitted during time equal to the current bandwidth-delay product of the TD links. We are thus proposing that the network is self-tuned at regular intervals during congestion, so that $\max_{th}$ is applied to router queues serving TCP Relative traffic during each interval according to:

$$\max_{th} = \frac{2 \times r \times one-way-delay}{MTU \times 8} \approx \frac{r \times RTT}{MTU \times 8} \quad (2)$$

where RTT is an averaged RTT value measured for Relative traffic packets at the previous interval. This approach assumes a homogeneous topology in terms of links' capacities and r values, however in the future we intend to investigate topologies with heterogeneous links and bottlenecks. Intuitively, in such cases, tuning of $\max_{th}$ will be mostly needed in the routers where bottlenecks initiate.

Based on this brief theoretical approach, the experimental approach that follows attempts to clarify how different configurations of the Relative queues' service rate and WRED thresholds affect the performance perceived by Relative traffic.

5. Experimental approach of the Relative Service

For the experimental analysis presented in the sequel, a minimal topology, simulating the backbone link of a transport network serving aggregated traffic (MAN/WAN) was used. The simulations were carried out using the ns-2 simulator ([5]) and a number of additional modules, such as that of generating background traffic, as presented in [6].

In Figure 2(a) the topology used in the first set of experiments, for controlling the average queue size with the use of WRED, is presented. The backbone link is the one connecting nodes 0 and 1, of 10 Mbps capacity. For the evaluation of the performance in each of the first set of experiments an HTTP server and a number of clients were attached to nodes 0 and 1 respectively. As an indication of performance, the throughput obtained by the HTTP server was observed throughout the experiments. The total of traffic (background and HTTP server traffic)

was served by a single queue in node 0 and node 1, to which WRED was applied. For the evaluation of WRED performance on the control of the queue size during congestion, the backbone link was congested with traffic up to 120% of its capacity.

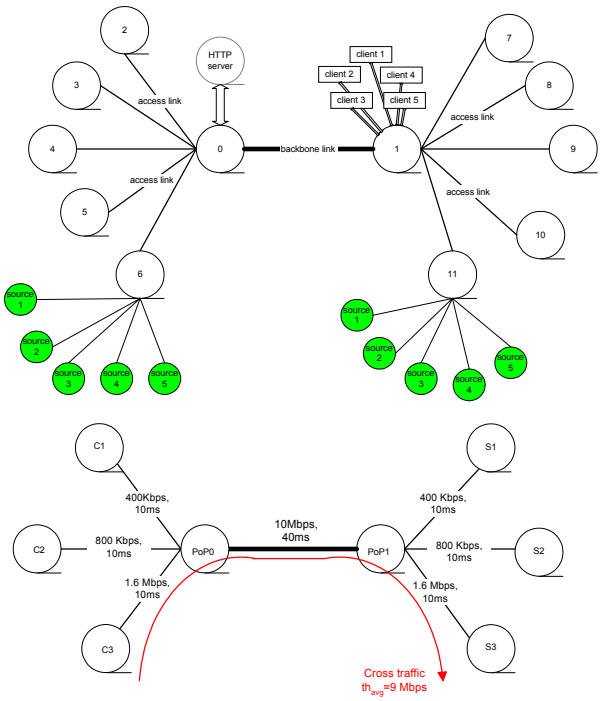


Figure 2. Topologies for conducted experiments

The rest of the experiments, for the dimensioning of the proposed Relative Service on the PoPs of a backbone network, were conducted over the topology of Figure 2(b). Here the backbone link is the one connecting PoP_0 and PoP_1 and the part of the topology used for generating the background traffic is omitted for simplicity. There exist three customers (C_1, C_2, C_3) attached to PoP_0 with different access capacities and equally three sinks (S_1, S_2, S_3) of traffic generated by the customers. Background traffic with a realistic synthesis and an average throughput of 9 Mbps was generated over the backbone link, co-existing with high-priority UDP traffic and the TCP Relative traffic injected by (C_1, C_2, C_3) in PoP_0. In this second topology, background traffic, Relative traffic and high-priority traffic were served by different router queues with the queue serving high-priority traffic always having strict priority over the other two queues (use of MDRR strict priority scheduling).

5.1. Control of average queue size with the use of WRED

In this section, a set of experiments that aim at investigating how the queuing delay perceived by TCP Relative traffic packets can be controlled with the use of queue management mechanisms are presented. This goal can obviously be achieved by controlling the average size of queues that serve Relative traffic across the backbone of a domain.

The experiments that will follow are using WRED as a queue management algorithm in a simplified topology so that the mechanism for controlling the average queue size and its effects can be more straightforward. Based on equation (2), we have conducted a series of iterative experiments over the topology of Figure 2(a) so that $\max_{th}$ is calculated according to equation (2) and the average RTT value of the previous experiment, $\min_{th} = 0{,}15 \times \max_{th}$ and $\max_p = 0{,}2$.

Figure 3(a) displays the distribution of the queue sizes obtained in a series of experiments for a set of different $\max_{th}$ values. It can be noted how the reduction of $\max_{th}$ shifts the distribution of the queue size to the left. Figure 3(b) displays the average Relative queue size as a percentage of the series of $\max_{th}$ values used. It is observed how the configuration of $\max_{th}$ according to measured values of RTT perceived by TCP flows results in an average service queue size that remains within the interval

$$\{0.2 \ldots 0.25\} \times \frac{RTT \times bandwidth}{MTU \times 8} \qquad (3)$$

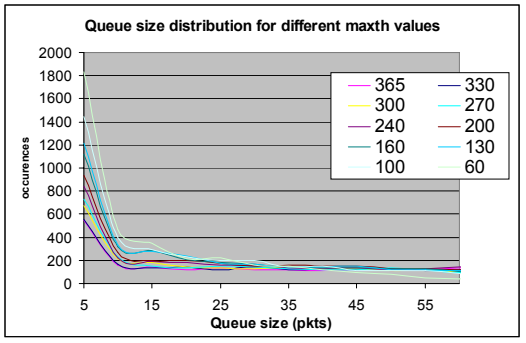
As a result, according to the current topology and the scheduling mechanism used, it appears to be possible to bound and estimate the average delay perceived by Relative traffic packets as they cross a number of transmission queues along the transmission path.

The performance perceived in the experiments carried out, as the throughput achieved by the HTTP server was seriously degraded for values of $\max_{th}$ that were less than 60. For these values, as deducted from Figure 3(a) the corresponding $\min_{th}$ (<9) results in a significant packet drop figure. Obviously, the decrease of $\max_{th}$, $\min_{th}$ values should not under any circumstances be infinite, because there seems to exist a turning point, after which the perceived performance is significantly reduced. The equilibrium point, at which a reliable bound on queue size and thus on queuing delay can be offered, without performance degradation, seems to require further analysis.

We have also conducted another set of experiments for the topology of Figure 3(b), examining the behaviour of WRED in two different cases:

- Using a static WRED parameters configuration

- Using a dynamic WRED parameters' configuration, according to RTT values measured in the experiments, and adjusting the value of $\max_{th}$ according to equation (3) at intervals of 50ms.



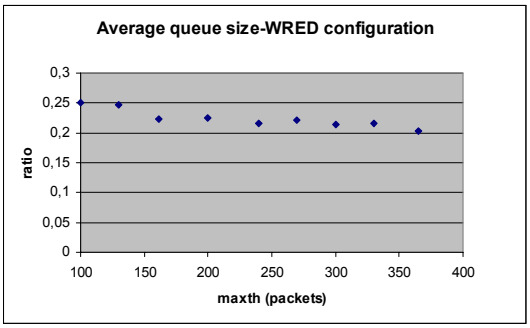
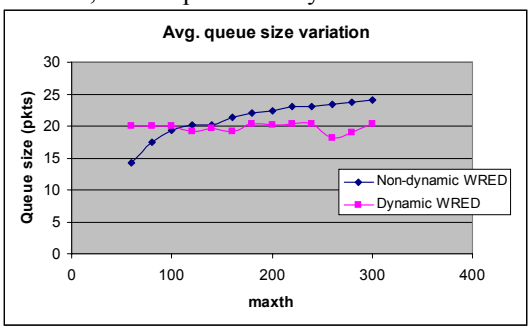


Figure 3. The distribution of service queue size and the average service queue size for different maxth values

Figure 4 depicts the results of the experiments conducted, where for the static case the x-axis enumerates the $\max_{th}$ values used for the series of experiments whereas for the dynamic case the x-axis depicts for each experiment the initial value of $\max_{th}$, since during these experiments, the value of $\max_{th}$ is dynamically adjusted.

It can be easily noticed how the dynamic WRED case succeeds in achieving a lower average queue size variation than using a static configuration for WRED parameters. The advantages of using a dynamic WRED are further reinforced by observing how the throughput of the HTTP flow is preserved in a stable value in comparison to the fluctuation of the static WRED case. It is also important to notice that for the specific experimental set-up, only a subset of the different static WRED configurations ($\max_{th}$={120 … 180}) succeed in obtaining the ideal throughput (of 1Mbps) for the HTTP flow.

The conclusion that can be drawn is the potential to control the average occupancy of a queue serving aggregated TCP traffic by properly configuring WRED parameters, taking into consideration the perceived by traffic RTT. This performance achievement can be exploited by the Relative Service in providing bounded average end-to-end delay guarantees to conforming traffic and thus an improved quality, in relation to best-effort service, will be perceived by end-users.



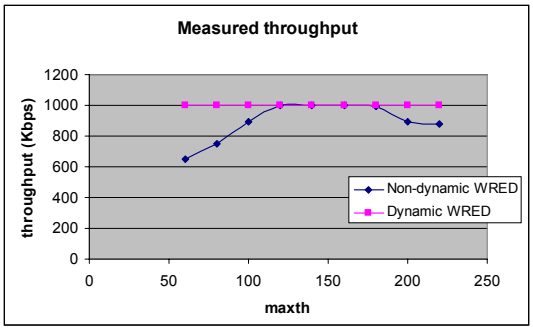


Figure 4. Comparison of static and dynamic WRED wrt average queue size and measured throughput

5.2. Relative Service dimensioning

The rest of the experiments that will be presented for the dimensioning of the proposed Relative Service at the PoPs of a backbone network, aim at investigating:

- Weight values' ($weight_{REL}$) configuration for the queue that serves Relative traffic in each router
- WRED configuration and handling of Relative traffic that falls out of the predetermined profiles

For these experiments, the topology of Figure 2(b) was used. The synthesis of high priority UDP traffic injected to PoP_0 is shown in Table 1. Relative traffic comprises of TCP Constant Bit Rate (CBR) sources with transmission rate of 100Kbps and packet sizes of 500 bytes, occupying all the available capacity in the access links of the customers C_1, C_2, C_3, after the high-priority traffic is served. In this way, a worst-case scenario is anticipated for, where all customers mark all of their non-high-priority traffic as Relative.

Background traffic is also inserted to the topology as depicted in Figure 2(b). In all the experiments, high-priority traffic is inserted at second 300.

5.2.1. **Relation between the policing profile and throughput provisioning.** In the experiments that follow, the Relative traffic injected by the customers was policed according to the profiles r_1=240Kb, b_1= 5000

bytes for customer C_1, r_2 = 400Kb, b_2 = 5000 bytes for customer C_2 and r_3 = 1Mb, b_3 = 5000 bytes for customer C_3. The packets not obeying to the aforementioned profiles are colored as 'yellow', while the conforming packets are marked as 'green'. High-priority UDP traffic was measured to use on average 1100Mbps of the backbone link so that the total of in-profile Relative traffic ($\sum r_i$) comprises the 18,4% of the rest of the available capacity.

	High-priority traffic	**Packet sizes**
C_1	2 VoIP flows ('on' rate: 80 Kbps)	188 bytes
C_2	1 MPEG video flow (r_{avg} :333 Kbps)	200 bytes
C_3	1 VBR flow (r_{avg} :448 Kbps), 2 VoIP flows ('on' rate: 80 Kbps)	1500 & 188 bytes

Table 1. High-priority traffic synthesis

Initially it was investigated, how the configuration of a service rate (r) for the queue serving Relative traffic affects the throughput perceived by Relative traffic and the queue size for that particular queue. For these experiments, the out-of-profile packets of Relative traffic were dropped for simplicity. Table 2 demonstrates how the Relative capacity is distributed among the three customers. As already mentioned, MDRR strict priority scheduling was used for our experiments, carried out for different configurations of the *weight* parameter of the Relative traffic queue.

$weight_{REL}$	BE	C1	C2	C3
30	7002	215	384	949
20	7005	216	384	952
10	7271	192	238	727

Table 2. Capacity distribution (in Kbps) on the backbone link

One can observe that in cases where $weight_{REL} = 30$ and $weight_{REL} = 20$, so that $\sum r_i < r$ and $\sum r_i \approx r$ correspondingly, then Relative traffic obtains 90-95% of the capacity predetermined on the contracted profiles for each customer C_i. There seems to be no significant gain in approaching 100% of the contracted capacity for each customer by introducing over-provisioning (in the case where $weight_{REL} = 30$).

In the case where $weight_{REL} = 10$, so that $\sum r_i > r$, the capacity of Relative traffic is significantly reduced. Figure 5 shows how the throughput of Relative traffic for the three customers is measured for the cases where $weight_{REL} = 20$ and $weight_{REL} = 10$.

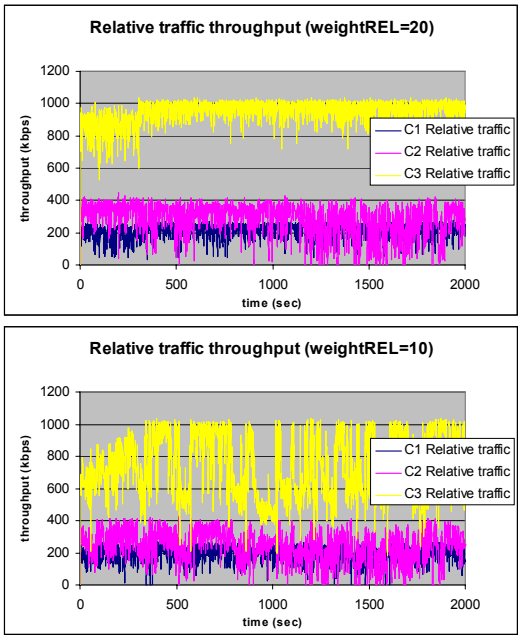


Figure 5. The throughput per customer in different provisioning circumstances

Figure 6 shows the distribution of the Relative traffic queue size for the three $weight_{REL}$ configurations. The conclusion is that with an appropriate provisioning of transmission rate (according to equation (1)), the in-profile Relative traffic is provided with the contracted capacity during congestion.

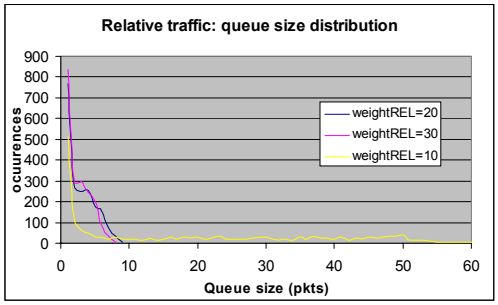


Figure 6. Relative queue size distribution

Also, Figure 6 shows how, in this simplistic case of out-of-profile packets' dropping, the recommended provisioning ($weight_{REL} = 30$ and $weight_{REL} = 20$) results in a size of the queue serving Relative traffic that does not exceed a few packets (10 packets actually) and thus a negligible queuing delay for 'green' packets.

5.2.2. WRED configuration According to the methodology already presented in section 5.1, the appropriate $\min_{th}, \max_{th}$ values for the Relative Service in the topology of Figure 2(b) with $weight_{REL} = 20$ were obtained. Three different cases for serving out-of-profile ('yellow') Relative traffic packets were investigated:

- Serving 'yellow' packets by the queue serving best-effort traffic (case A)
- Serving 'yellow' packets by the queue serving Relative traffic, with a lower probability than 'green' ones either so that $\min_{th}^{green} = \min_{th}^{yellow}$ but $\max_{th}^{green} = \max_{th}^{yellow}$, while $\max_{p}^{yellow} > \max_{p}^{green}$ (case B) or that $\max_{th}^{green} > \min_{th}^{green} = \max_{th}^{yellow} > \min_{th}^{yellow}$ and $\max_{p}^{yellow} > \max_{p}^{green}$ (case C)

In case A, Relative traffic increases its perceived bandwidth in the expense of background, best-effort traffic, compared to the case in which Relative 'yellow' packets are dropped (see Table 3).

	Drop yellow packets	Case A
Background traffic	7005	6838
Relative traffic -C1	216	341
Relative traffic -C2	284	388
Relative traffic -C3	949	1092

Table 3. Throughput (in Kbps) achieved by Relative traffic in two different cases

In this way, Relative traffic, apart from the guaranteed capacity, obtains access to more bandwidth, while the queue serving it, preserves a restricted size (Figure 7). Packets are thus perceiving a satisfactory RTT.

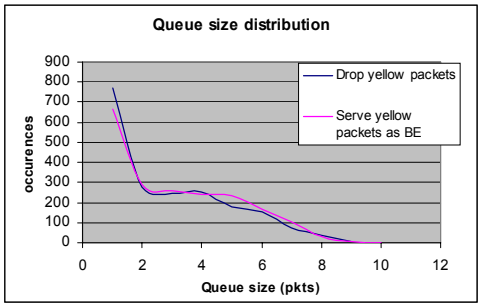


Figure 7. Queue length distribution for Relative queue

In case B, three different WRED configurations were tested:

B_1: $\min_{th}^{green} = \min_{th}^{yellow}$ =22, $\max_{th}^{green} = \max_{th}^{yellow}$ =149, $\max_{p}^{green} = 0{,}2$, $\max_{p}^{yellow} = 0{,}4$

B_2: $\min_{th}^{green} = \min_{th}^{yellow}$ =15, $\max_{th}^{green} = \max_{th}^{yellow}$ =100, $\max_{p}^{green} = 0{,}2$, $\max_{p}^{yellow} = 0{,}4$

B_3: $\min_{th}^{green} = \min_{th}^{yellow}$ =15, $\max_{th}^{green} = \max_{th}^{yellow}$ =100, $\max_{p}^{green} = 0{,}2$, $\max_{p}^{yellow} = 0{,}8$

Table 4 provides the results of the relevant experiments. One of the major outcomes is that, with few deviations, the different WRED configurations do not significantly differentiate the distribution of capacity among the aggregated flows of each customer. The distribution of Relative capacity in the three cases of WRED enforcement does not differ significantly from the distribution obtained without the use of WRED. It is important to note, though, how the introduction of WRED succeeds in reducing the mean and maximum length in packets of the queue serving Relative traffic, and thus the delay perceived by in-profile Relative traffic packets.

As far as the comparison of different configurations for $\min_{th}, \max_{th}, \max_{p}$ is concerned, one can notice how in cases B_1 and B_2, reducing the values for $\min_{th}, \max_{th}$ succeeds in reducing the average queue size by 50%, without violating the guaranteed capacity for Relative traffic. It must also be pointed out how increasing the difference between the value of $\max_{p}^{yellow}$ and the value of $\max_{p}^{green}$ (cases B_2 and B_3) de-stabilizes the operation of WRED and fails to obtain the desired result: the reduction of the average length for the queue serving Relative traffic. Also during the experiments it was measured that case B_2 achieves a lower average queue size.

	B_1	B_2	B_3	no WRED
Background traffic	6992	6973	6682	6826
Relative traffic-C1	352	342	341	347
Relative traffic-C2	512	494	514	400
Relative traffic-C3	1035	1067	1040	1086
"Green" queue size:avg(max) pkts	22 (125)	13 (100)	20 (125)	47 (250)
"Yellow" queue size:avg(max) pkts	6 (88)	3 (70)	5(88)	11 (98)
Max RTT (sec)	419	259	364	1021

Table 4. Comparative results of four different WRED configurations (in Kbps)

In case C, we make a comparison between the WRED configuration of case B2 and the configuration $\max_{th}^{green}$ =100, $\min_{th}^{green} = \max_{th}^{yellow}$ =40, $\min_{th}^{yellow}$ =15, $\max_{p}^{green} = 0{,}2$, $\max_{p}^{yellow} = 0{,}4$. In Table 5 for a series of experiments conducted, it is presented how the configuration of the parameters $\min_{th}, \max_{th}, \max_{p}$ according to Case 2 of section "Queue management" affect negatively the performance of WRED, by increasing the average queue size for Relative traffic.

	C	B_2	no WRED
Background traffic	6939	6973	6826
Relative traffic-C1	348	342	347
Relative traffic-C2	499	494	400
Relative traffic-C3	1055	1067	1086
"Green" queue size: avg/max	24 /107	13/100	47/250
"Yellow" queue size: avg/max	6 /88	3/70	11/98
Max RTT (sec)	512	259	1021

Table 5. Comparative results of three different WRED configurations (in Kbps)

In this case as well, the two different WRED configurations do not significantly differentiate the capacity distribution to the aggregated Relative traffic flows to which they are applied. The experiments conclude in favor of the WRED configuration of Case 1 (section "Queue management") for the purposes of the Relative service model.

6. Future work and conclusions

Our future work on the Relative Service model will focus on further testing its implementation in both a simulating and a realistic environment. Extensive testing should be performed on the model's performance for heterogeneous network topologies with one or more bottlenecks. We also intend to test the service model under various congestion levels. Another point of out future research will be to focus on the frequency of the WRED tuning intervals and how does varying the length of these intervals affect the delay bound achieved. Finally we aim at testing the service model in non-equilibrium conditions, where the average load of Relative traffic fluctuates, in order to observe how successfully are the capacity and average delay bound guarantees honoured as the Relative traffic queues' management mechanism is self-tuned.

The conclusions drawn from the work presented here are positive towards the adaptation of a simple model for relative service differentiation on a backbone network. We have shown how such a service must be designed and configured on network elements. Our study demonstrates that especially in the case of networks serving aggregated traffic, queue management mechanisms (such as WRED) do not affect the distribution of capacity between differently marked traffic aggregates. However, experimental data presented show that it is feasible to bound the average queue length in a router serving aggregated traffic of different priority levels with the use of such a queue management mechanism. This mechanism, when accompanied by proper provisioning of service rate for Relative traffic, is experimentally confirmed to provide minimum rate guarantees and a bounded average queuing delay to in-profile traffic.

Acknowledgements

This work has been partially funded by GRNET S.A. during the project "IP QoS: Design and implementation of a QoS architecture over GRNet". The authors also wish to explicitly thank D. Kalogeras for the fruitful discussions on the topic of this paper.

References

[1] B. Nandy, N. Seddigh and P. Pieda, "Diffserv's Assured Forwarding PHB: What Assurance does the Customer Have?", NOSSDAV'99, New Jersey, USA, June 1999

[2] I. Yeom and N. Reddy, "Impact of marking strategy on aggregated flows in a differentiated services network", Proc. of IWQoS'99 Workshop, London, UK, June 1999

[3] M. May, J-C. Bolot, A. Jean-Marie and C. Diot, "Simple performance models of tagging schemes for service differentiation in the Internet", Proc. of Infocom '99, New York, USA, March 1999

[4] Cisco 12000 Series Internet Router: Frequently Asked Questions, found at: http://www.cisco.com/warp/public/63/gsrfaq_11085.shtml

[5] S. McCanne and S. Floyd, "ns Network Simulator", available at: http://www.isi.edu/nsnam/ns/

[6] C. Bouras, D. Primpas, A. Sevasti, A. Varnavas, 'Enhancing the DiffServ architecture of a simulation environment', 6th IEEE International Workshop on Distributed Simulation and Real Time Applications, USA, October 2002

[7] D. Lin and R. Morris, 'Dynamics of random early detection', Proc. of ACM SIGCOMM '97, pp. 127--137, October 1997

[8] B. Nandy, N. Seddigh and P. Pieda, 'Diffserv's Assured Forwarding PHB: What Assurance does the Customer Have?', Proc. of NOSSDAV'99, New Jersey, June 1999

[9] W. Feng, D. Kandlur, D. Saha and K. Shin, 'A Self-Configuring RED Gateway', Proc. of IEEE INFOCOM, pp. 1320-1328, 1999

[10] R. Kapoor, C. Casetti and M. Gerla, 'Core-Stateless Fair Bandwidth Allocation for TCP flows', IEEE ICC2001, Helsinki, Finland, June 2001

[11] F. Azeem, A. Rao, X. Lu, and S. Kalyanaraman. 'TCP-Friendly traffic conditioners for differentiated services', Internet Draft, IETF, February 1999

[12] S. Athuraliya, V. H. Li, S. H. Low and Qinghe Yin, 'REM: Active Queue Management', 17th International Teletraffic Congress (ITC), December 2001

[13] C. V. Hollot, V. Misra, D. Towsley, and W. Gong. 'A control theoretic analysis of RED', In Proc.of IEEE INFOCOM, April 2001