Available online at www.sciencedirect.com
ScienceDirect
ELSEVIER
Int. J. Electron. Commun. (AEÜ) 60 (2006) 619 – 627
International Journal of Electronics and Communications AEÜ
www.elsevier.de/aeue

A new algorithm for traffic conditioners to improve proportional fairness in IP networks

Maria-Dolores Cano*, Fernando Cerdan

Department of Information & Communications Technologies, Polytechnic University of Cartagena, 30202 Cartagena, Spain

Received 22 December 2004

Abstract

The current trend in telecommunications networks addresses an All-IP vision, since most user applications are based on IP, and additionally, IP networks operation is simpler than other more complex ones. These new trends in multimedia communications are giving rise to an increasing need of quality of service (QoS). The Differentiated Services (DiffServ) QoS architecture will provide the means to help underlying technologies, like MPLS, to guarantee the QoS required to support data and real-time services. In DiffServ, QoS is mainly provided with traffic control at edge nodes, and this task is carried out by traffic conditioners. In this paper, we introduce a new policy function for traffic conditioning. Its ability to react to new network conditions makes end-users to get excess bandwidth in proportion to their contracted target rates. We call this method Proportional Excess Traffic conditionER (PETER). Through extensive simulations, we confirm the effectiveness of our approach in providing both a proportional excess bandwidth distribution and assured contracted target rates.

Keywords: Traffic conditioning; Assured forwarding; DiffServ; QoS

1. Introduction

The Differentiated Service (DiffServ) architecture [1] is the most promising approach to offer quality of service (QoS) in IP-based networks. DiffServ is intended to conceive a simple scheme that provides a range of QoS levels by moving complexity toward the edge of the network. The Type of Service packet header field from IP v.4 is substituted by the DiffServ field and new meanings are conferred to its bits: the six most significant bits form the DiffServ code point (DSCP), while the two less significant ones are currently unused. A group of mechanisms to handle packets of aggregated flows with different priorities according to the information carried in the DSCP is created. Thus, packets are classified and marked to receive a particular treatment on the nodes along their path. This treatment is known as per-hop behavior (PHB). Complex classification and conditioning functions (metering, marking, shaping and/or policing) need only to be implemented at boundary nodes, whereas interior nodes perform a set of forwarding PHBs to aggregates of traffic that have been appropriately marked. Notice that a boundary node is not compulsory a router device, but it can be the last hardware or software system the DiffServ domain administrator controls.

One of the current PHBs with the status of standard is the assured forwarding PHB (AF-PHB) [2,3]. The idea behind the AF-PHB is to assure a minimum throughput, the contracted target rate, to an end-user while enabling consuming excess bandwidth if the network load is lower than the maximum link utilization. Excess bandwidth is defined as the remaining available bandwidth once all connections have a throughput equal to their contracted target rates. Notice that we use the term throughput without considering retransmitted packets, which is usually called goodput. There are four independently forwarded AF instances. Within each AF

* Corresponding author. Tel.: +34 968325953; fax: +34 968325973.
E-mail address: mdolores.cano@upct.es (M.-D. Cano).

1434-8411/$ - see front matter © 2005 Elsevier GmbH. All rights reserved.
doi:10.1016/j.aeue.2005.12.003

instance, an IP packet is assigned one of three different levels of precedence. Packets inside the contracted user profile are called in-of-profile (in), while non-conformant packets are called out-of-profile (out). In this case only two levels of precedence are used. In case of employing three precedence levels, in packets are usually colored as green, and out packets are usually divided into yellow and red. When network congestion occurs, DiffServ nodes try to protect packets with a lower drop precedence value from being lost by preferably discarding packets with higher drop precedence.

Operators decide how to perform traffic conditioning (metering, marking, shaping and/or policing) to fulfill a service level agreement (SLA). Most related literature has focused on traffic conditioners for the AF-PHB Service, presenting different proposals to complete the AF goals. The first goal, assuring the contracted target rate of the final user, has been achieved for many of the published schemes. As regard to the second goal, using more bandwidth if the network load is low, there exists discrepancy in how to distribute the excess bandwidth. Some authors agree that a fair excess bandwidth sharing means an even distribution of spare bandwidth among all the sources composing the aggregate. On the contrary, other authors define a fair excess bandwidth sharing as the distribution of the spare bandwidth proportional to the contracted target rates of each source. In this work we follow the second approach.

In this paper, we introduce a new algorithm for proportional excess bandwidth sharing for the AF Service. With this approach, end-users get excess bandwidth in proportion to their target rates. The proposed traffic conditioner is placed next to the source of traffic (where the contract is established), but out of the reach of the final user. Basically, our proposal marks IP packets with one of two drop precedences (in and out, what simplifies the scheme) using the CB marker [4], and then the proportional excess traffic conditionER (PETER) algorithm is applied as a policy function. PETER adapts the source throughput to network conditions by discarding packets if necessary. To carry out this task, PETER needs some signaling from the edge network node. This signaling does not represent a problem since it runs in the user local loop (short distances). The traffic conditioner with PETER is evaluated through extensive simulations, and results show that the AF Service goals are widely satisfied.

Our study will focus on understanding and evaluating this solution in a single bottleneck. In this study we consider that all TCP sources try to obtain excess bandwidth if available. In situations where a TCP source is strongly controlled to produce only the contracted target rate, its quota of excess bandwidth will not be shared among other sources. However, this type of sources are out of the typical user service profile interested in assured forwarding services, where the likeliest demanding service will be downloading applications (music, documents, software updates, etc.) with a time scale of several minutes for TCP connections.

The rest of the paper is organized as follows: Section 2 presents related works. Section 3 describes the characteristics of our PETER proposal. Section 4 details the simulation topology and the simulation tool employed to conduct the performance evaluation. In Section 5 we present simulation results, and compare fairness and user contracts guarantees with results obtained for the classical time sliding window (TSW) algorithm (by TSW we understand the enhanced TSW version [5,4]). In Section 6, we discuss implementation issues. We end with conclusions in Section 7.

2. Related work

Studies done in [6–8] introduce algorithms for achieving proportional fairness in the AF-PHB Service. These proposals have in common the use of three colors for each AF-PHB instance. In [6], the authors present a random marking scheme for aggregated flows, i.e., marking is performed at the boundary node of a DiffServ domain, and the contract is specified for the aggregate of flows. The probability of marking a packet as green, yellow, or red is a function of aggregate transmission rate with respect to aggregate contracted target rate and peak rate. Fairness is evaluated by simulation, understood as follows: none of the flows is discriminated or experiences lower share of bandwidth than the others. El-Gendy et al. propose in [7] a marking algorithm called equation-based marking (EBM). This algorithm senses the current network conditions and adapts the packet marking probabilities accordingly. It uses a feedback control mechanism based on the TCP model, so adjusts the sending rate by sensing the level of congestion in the network via observation of packet losses. Although simulation results show a proportional distribution of excess bandwidth, the scheme presents a high level of complexity; mainly due to the need of computing a round trip time (RTT) estimation, target loss probabilities, and for each packet a marking probability. The study done in [8] focuses on how to provide proportional fairness among aggregates in a DiffServ network. As in [6], they concentrate on aggregates (only two in this case), and simulation results show a high level of fairness for networks with a medium provision level (20–70%). By provision level, or network load, we mean the percentage of the total available bandwidth that is contracted by the users.

To the best of our knowledge, there are only a few approaches to offer a proportional distribution of excess bandwidth using two levels of precedence (in fact only one). In [9], the authors propose traffic aware traffic conditioner (TATC). This algorithm allocates back out-of-profile bandwidth to in-profile bandwidth in proportion to the target rates, what presumably leads to a higher assured bandwidth for flows with high target rates. On the other hand, algorithms such as TSW [10] or enhanced time sliding window (ETSW) [5] were employed to compare the performance of EBM in terms of excess bandwidth sharing. Although neither TSW nor ETSW were thought to carry out a proportional distribution of excess bandwidth, the widespread use of TSW turned them into classical references. The lack of contributions that use two levels of precedence motivates us to employ an enhanced version of TSW [5,4] to perform comparisons with our proposal.

3. The PETER algorithm

As mentioned in the introduction, the AF defines four different instances. Packets are marked to belong to one of the four independently forwarded AF instances. Within each AF instance, there are up to three levels of precedence. PETER only uses two levels of precedence. Next, we describe the PETER algorithm.

Let us denote by c the link capacity and by b the sum of all contracted target rates of those sources that join in a boundary node. For two-color based traffic, we can define α_{ideal} as the quotient link capacity c minus b divided by b.

$$\alpha_{\text{ideal}} = \frac{\text{link capacity} - \Sigma_{i=1}^{n}\text{target rate}_i}{\Sigma_{i=1}^{n}\text{target rate}_i} = \frac{c-b}{b}. \quad (1)$$

Notice that the upper part of this fraction represents the excess bandwidth.

Suppose that we measure the ratio number of out packets divided by the number of in packets that leave the boundary node in the time interval (t_1, t_2). We call this value α_{m}, see Eq. (2). For simplicity, we assume that all packets have a similar size, but Eq. (2) can be also calculated with the sum of packet sizes. Then, if link utilization is about hundred percent (maximum efficiency), α_{m} and α_{ideal} should be almost equal. That is, if we subtract b from the link capacity c we obtain the excess bandwidth, and all packets marked as out represent the excess bandwidth. Similarly, all packets marked as in represent b. In consequence, the ideal situation yields to α_{m} equal to α_{ideal}.

$$\alpha_{\text{m}} = \frac{\Sigma_{t_1}^{t_2}\text{ out packets}}{\Sigma_{t_1}^{t_2}\text{ in packets}}. \quad (2)$$

Observe that α_{ideal} is a fixed value unless a user cancels his/her contracted target rate, modifies it or the boundary node gives service to a new user (a new contracted target rates is established). All these changes are communicated directly to the DiffServ domain administrator, who sets up the new α_{ideal} in the boundary node.

Considering an environment where sources come and leave frequently (balanced case), as it occurs in most of the real-life scenarios, the α_{ideal} that we should take into account these variations will not differ from the initially configured α_{ideal}. Nevertheless, in situations where the numbers of sources that come or leave is unbalanced, we do need to change the initially configured α_{ideal} in order to keep network efficiency. These unbalanced situations are easily detected because α_{m} is clearly different from the fixed value α_{ideal}. Once detected, and assuming some signaling, α_{ideal} is updated. Thus we are in a balanced case again, where we perform our evaluation study shown in next sections.

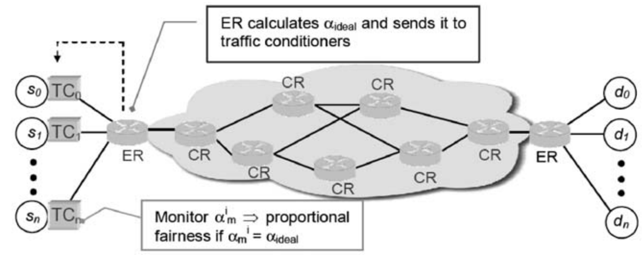

Fig. 1. General procedures using PETER (ER ≡ Edge Router, CR ≡ Core Router).

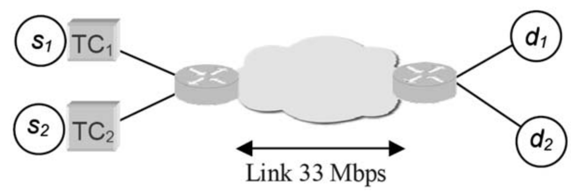

Fig. 2. Example topology.

If we measure the ratio α_{m} at each traffic conditioner, we can use α_{ideal} to achieve a fair distribution of excess bandwidth. Given the ideal value α_{ideal}, we compare it with the corresponding α_{m} value obtained at each traffic conditioner (α_{m}^i, where i is the source number). If α_{m}^i is less than α_{ideal} then the source i is not consuming its corresponding excess bandwidth. If both values coincide, then the source i consumes exactly its corresponding spare bandwidth. Finally, if α_{m}^i is greater than α_{ideal} then the source i is consuming bandwidth beyond its fair quota. Therefore, when it is detected a value of α_{m}^i greater than α_{ideal}, source i has to be penalized to decrease its throughput. Fig. 1 shows the general operation of PETER.

Let us give an example to illustrate PETER operation. Suppose that there are two sources, s_1 and s_2, whose contracted target rates are 1 and 10 Mbps, respectively (see Fig. 2). Link capacity is 33 Mbps, so α_{ideal} is in this case equal to 2. From Eq. (1):

$$\alpha_{\text{ideal}} = \frac{\text{link capacity} - \Sigma_{i=1}^{n}\text{ target rate}_i}{\Sigma_{i=1}^{n}\text{ target rate}_i} = \frac{33\,\text{Mbps} - (1+10)\,\text{Mbps}}{(1+10)\,\text{Mbps}} = 2.$$

For a fair excess bandwidth distribution s_1 should get 2 Mbps of the excess bandwidth, whereas s_2 should obtain 20 Mbps. Both values are in proportion to their service profiles:

- Excess bandwidth $33 - (1+10) = 22$ Mbps.

- Ratio between contracted target rates of s_1 and s_2 is 1/10, so s_2 should get ten times more excess bandwidth than s_1.
- We denote by x the portion of excess bandwidth of source s_2, so $x + x/10 = 22$.
- Therefore, s_2 should get 20 Mbps and s_1 should get 2 Mbps.

Each traffic conditioner measures at time intervals α_m^1 and α_m^2. If both values are equal to 2 (α_{ideal}), then s_1 gets 2 Mbps of excess bandwidth, two times its contracted target rate, see Eq. (2). The same applies to s_2 that gets 20 Mbps (again, two times its contracted target rate). On the contrary, if α_m^1 is equal to 3, then s_1 gets 3 Mbps of excess bandwidth, so it is stealing bandwidth that proportionally belongs to s_2. If α_m^1 is equal to 1, then s_1 only obtains 1 Mbps of spare bandwidth, so s_2 has to reduce its throughput because is consuming more bandwidth than allowed.

From this example, we extract that it is possible to know the behavior of the sources by comparing α_m^i and α_{ideal}. We have seen that with the relation $\alpha_m^i = \alpha_{ideal}$ the system works well, but the inequality identifies unfairness in the excess bandwidth distribution. In fact, if $\alpha_m^i \succ \alpha_{ideal}$ then the source i has to decrease its throughput. Because we are working with TCP sources, a packet loss makes sources to slow down. Since this is our goal, we employ packet discarding in the corresponding traffic conditioner i when the condition $\alpha_m^i \succ \alpha_{ideal}$ is detected.

There are different options to be applied for packet discarding. One of them consists of dropping packets if $\alpha_m^i \succ \alpha_{ideal}$ independently on the type of packet (in or out). The question associated to this option is that discarding in packets may cause problems in assuring contracted target rates. The solution we adopt is to discard only out packets when the condition $\alpha_m^i \succ \alpha_{ideal}$ is true. The pseudo-code of the PETER algorithm is shown below.

α_m^i = ratio out packets/in packets
if ($\alpha_m^i \preceq \alpha_{ideal}$)
    do not discard the packet
else
    if packet is in
        do not discard the packet
    else
        discard the packet

Before applying PETER as a policy function, packets are marked in the traffic conditioner. In this case, marking is done with the counters-based algorithm (CB) introduced in [4] and used in [11]. CB performs comparatively better than other marking schemes like TSW or Leaky Bucket (LB). Its main advantages are an easy configuration and high accuracy in guaranteeing the contracted target rates in heterogeneous scenarios. CB uses two counters and includes a simple mechanism to avoid accumulation of *credits* when a source stops transmitting data, for instance when a time out occurs. The pseudo-code of CB is written below.

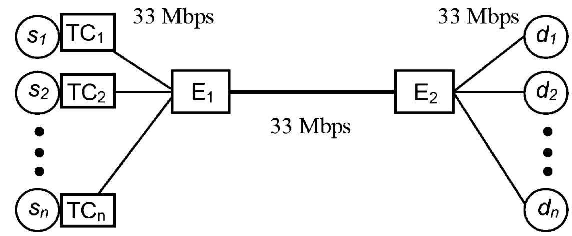

Fig. 3. Simulation topology.

Initially:
    $Counter_1 = 1$
    $Counter_2$ = link rate/contracted target rate
For each unit of time:
    $Counter_2 --$
    if $Counter_2 \preceq 0$
        $Counter_1 ++$
        $Counter_2$ = link rate/contracted target rate
    if there is a packet arrival
        if $Counter_1 \succ 0$
            packet marked as in
            $Counter_1 --$
        else
            packet marked as out

4. Simulation setup

In this section, we describe the simulation topology that we use to carry out simulations. The simulation tool for the sliding window protocol of TCP Reno sources was developed in [12] and was widely used in [13,14]. Some of its features are: TCP sources are long-lived for a worst-case study, that is, they have unlimited data to send; destinations only send acknowledgments, which are never lost or delayed; and the maximum window size equals the product bandwidth delay as usual for WAN environments.

The simulation topology is shown in Fig. 3. There are n TCP Reno sources ($s_1, s_2, \ldots, s_n$) transmitting at the link rate, which has been set to 33 Mbps. All sources send traffic to destinations ($d_1, d_2, \ldots, d_n$) through the edge node E_1. The bottleneck is placed between the edge nodes E_1 and E_2, since the sources transmit at link rate. α_{ideal} is computed at E_1 with the sum of the contracted target rates of sources s_1 to s_n and the serial interface link capacity to the next network node E_2.

For the AF PHB, edge nodes employ RIO (RED (In and Out packets)) [10]. RIO is the combination of two RED [15] algorithms with different drop probability curves so that out packets are likelier to be discarded. Each RED algorithm has three parameters (min, max, p) that define the normal operation phase [0, min), the congestion avoidance phase [min, max), and the congestion control phase [max, ∞). The probability of dropping an out packet depends on the total number of packets that arrive at the node, while the probability of dropping an in packet depends exclusively on the buffer occupancy of in packets. The RIO parameters employed in our simulations are [40/70/0.02] for in packets and [10/40/0.2] for out ones. RED parameters $\text{weight}_{\text{in}}$ and $\text{weight}_{\text{out}}$ used to calculate the average queue size were chosen equal to 0.002 as recommended in [15].

We evaluate the performance of the PETER algorithm in five different scenarios. Moreover, we obtain results varying the network provision level from 20% to 90% in all cases. Scenarios present the following characteristics.

- *Case* 1: As a first step, contracted target rates and round trip times are the same for all sources. This is the simplest scenario. Round trip time is fixed to 50 ms. Topology is composed of eight TCP Reno sources whose contracted target rates depend on the network load. For instance, for a network load of 20% the contracted target rate is 0.825 Mbps for all sources.
- *Case* 2: In this case, we evaluate the effect on performance if sources have different contracted target rates. Topology is composed of eight TCP Reno sources. The value of the contracted target rates depends on the network load. For instance, for a network load of 60% contracted target rates are 1, 1, 2, 2, 3, 3, 4 and 4 Mbps. Round trip time is the same for all connections and is equal to 50 ms.
- *Case* 3: We study the impact of round trip time variation among sources. Topology is composed of eight TCP Reno sources. The round trip time is established to 10, 20, 30, 40, 50, 60, 70 and 80 ms from source s_1 to source s_8. All sources have the same contracted target rate that depends on the network load as in the previous case.
- *Case* 4: For a more realistic case, we vary both the contracted target rate and the round trip time among sources. Topology is composed of eight TCP Reno sources. Round trip time is equal to 10, 20, 30, 40, 50, 60, 70 and 80 ms for source s_1 to source s_8, respectively. Contracted target rates depend on the network load.
- *Case* 5: In this last case we analyze the effect on performance if we increment the number of sources. Topology is composed of 16 TCP Reno sources. Round trip time is the same for all sources, 50 ms. The first four sources, s_1 to s_4, present a fixed contracted target rate of 1 Mbps. The rest of the sources, s_5 to s_{16}, have the same contract but its value depends again on the network load. For instance, for a network load of 20% contracted target rate is 0.216 Mbps for sources s_5 to s_{16}.

Simulation results have a confidence interval of 95% that has been calculated with a normal distribution function using 30 samples, with an approximate value of ±0.002 for fairness calculations and ±0.01 for achieved target rates.

5. Results

In this section we present and discuss simulation results. We evaluate the performance of the proposed traffic conditioner with PETER in terms of guarantees of achieving contracted target rates and fairness in excess bandwidth sharing. Results are also compared with the classical TSW. As we mentioned in the introduction section, we use an improved version of TSW to compare its performance with results obtained with PETER. This enhancement consists of using the most appropriated parameter configuration in TSW. The configuration guide we followed is included in [5,4].

To analyze the fairness of different schemes we use the definition given in [16], where the fairness index f is calculated as follows:

$$f = \frac{(\Sigma_{i=1}^{n} x_i)^2}{n\Sigma_{i=1}^{n} x_i^2}, \tag{3}$$

$$x_i = \frac{\text{throughput}_i - \text{contracted target rate}_i}{\text{contracted target rate}_i}, \tag{4}$$

where x_i is the excess throughput of source i divided by the contracted target rate of source i, see Eq. (4), and n is the number of sources that arrive to the boundary node. As f approximates to 1, the sharing of the spare bandwidth is fairer. We should mention the importance of improving the fairness index even in a small quantity. For the same example used in Section 3 (Fig. 2), two sources s_1 and s_2 with targets of 1 and 10 Mbps and a link rate of 33 Mbps, the fair distribution means that s_1 gets 2 Mbps and s_2 gets 20 Mbps of excess, respectively. This provides a fairness index of 1. In case s_1 obtains 3 Mbps and s_2 19 Mbps of excess, respectively, observe that s_2 obtains one megabit less of its corresponding excess, the fairness index decreases to 0.95 (only 0.05 points!). Therefore, small increments in the fairness index may represent noticeable improvement in the traffic conditioner performance.

5.1. Case 1: same contracted target rates and round trip times

In this case, eight TCP Reno sources contract the same target rate. Round trip time is set to 50 ms for all connections. This is the simplest scenario we can consider. Table 1 illustrates the strong assurance in achieving contracted target rates for a provision level of 60% with PETER. We see that end-users achieve their corresponding targets. Compared with TSW, the main difference is that the measured in packet throughput with TSW does not guarantee the end-user target. When two levels of precedence are used in the AF Service, in-of-profile packets must assure the user contracted target rate. In Table 1, compare columns 3 and 4 (in packet throughput) with column 2 (Target Rate). We see that TSW does not achieve the contract with the throughput of in packets. This is a well-known TSW behavior [10]. In this particular simulation, this fact does not represent a problem because targets are guaranteed with the total throughput (in plus out). However, it can be an inconvenience in more extensive topologies, because out packets have less priority and may be discarded at intermediate network nodes.

Table 1. Throughputs obtained in case 1 with eight TCP Reno sources that have contracted target rates of 2.5 Mbps (60% provision level)

s	TR	in packet throughput		Total throughput	
		PETER	TSW	PETER	TSW
1	2.5	2.48	2.15	3.84	4.19
2	2.5	2.48	2.08	3.85	4.04
3	2.5	2.49	2.11	3.87	3.91
4	2.5	2.48	2.19	3.84	4.16
5	2.5	2.49	2.13	3.86	3.94
6	2.5	2.49	2.13	3.85	3.94
7	2.5	2.49	2.16	3.88	4.17
8	2.5	2.49	2.19	3.86	4.20

Round trip time is set to 50 ms for all connections. ($s \equiv$ source; TR $\equiv$ Target Rate).

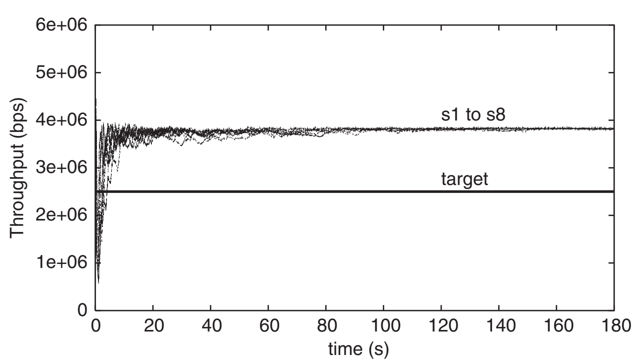

Fig. 4. Throughput in case 1 with PETER for a 60% provision level.

We observe in Fig. 4 the throughput obtained by all sources with PETER (provision level 60%). Since all sources have the same target (2.5 Mbps), the excess bandwidth is evenly distributed among them. Our PETER scheme performs as expected in this scenario. Dropping out packets when the relation $\alpha^i_{\mathrm{m}} \succ \alpha_{\mathrm{ideal}}$ is detected, makes the TCP source to slow down. Fig. 5 shows that both TSW and PETER present a fairness index above 0.9 in the whole range of provision level (20–90%), but PETER presents higher values always above 0.98.

5.2. Case 2: variation in contracted target rates and same round trip times

In this section, the topology consists of eight TCP Reno sources whose contracted target rates are variable. The round trip time is set to 50 ms. Table 2 includes the throughputs obtained for PETER and TSW with a 60% provision level. Even with variation of targets among the different connections, PETER shows strong guarantees in achieving the contracts. With TSW, targets are reached, thanks to out packets, as it happened in case 1.

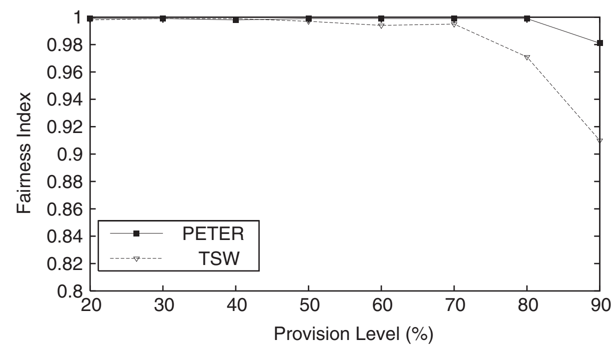

Fig. 5. Fairness index vs. provision level in case 1 with PETER and TSW.

Table 2. Throughputs obtained in case 2 with eight TCP Reno sources that have contracted target rates of 1–1–2–2–3–3–4 and 4 Mbps (60% provision level)

s	TR	in packet throughput		Total throughput	
		PETER	TSW	PETER	TSW
1	1	0.97	0.80	1.25	2.79
2	1	0.96	0.79	1.22	2.80
3	2	1.98	1.64	2.91	3.56
4	2	1.97	1.67	2.88	3.52
5	3	2.99	2.65	4.65	4.43
6	3	2.99	2.63	4.76	4.35
7	4	3.98	3.85	5.21	5.50
8	4	3.99	3.79	5.25	5.30

The round trip time is set to 50 ms for all connections.

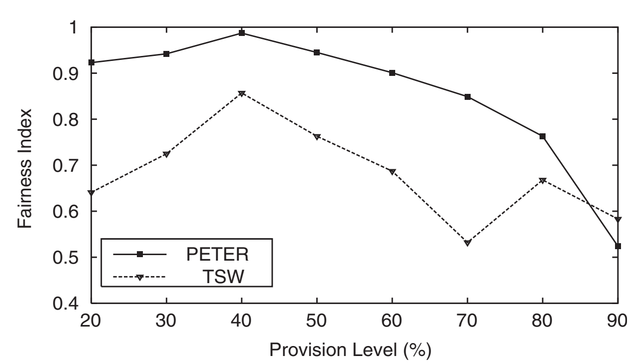

Fig. 6. Fairness index vs. provision level in case 2 with PETER and TSW.

Regarding the fairness in the distribution of excess bandwidth, we perceive in Fig. 6 that PETER presents a fairness index over the one obtained with TSW. The TSW curve increases until a network load around 40% and then it begins to decrease. This is due to the TSW behavior. It is known

Table 3. Throughputs obtained in case 3 with eight TCP Reno sources that have contracted target rates of 2.5 Mbps (60% provision level)

s	TR	in packet throughput		Total throughput	
		PETER	TSW	PETER	TSW
1	2.5	2.49	2.12	3.89	3.31
2	2.5	2.47	2.01	3.65	4.35
3	2.5	2.48	2.23	3.73	4.47
4	2.5	2.48	2.28	3.72	4.26
5	2.5	2.49	2.39	3.81	4.23
6	2.5	2.50	2.52	3.92	4.02
7	2.5	2.50	2.55	3.55	4.02
8	2.5	2.49	2.54	3.22	3.82

The round trip time is set to 10–20–30–40–50–60–70 and 80 ms for connections s_1 to s_8, respectively.

that TSW marks more in packets than allowed when network load is above 50% [5]. Moreover, TSW always favors sources with smaller contracted target rates [4,5,10]. These two facts together are responsible of the shape of the TSW fairness index curve.

It is important to remark the importance of a fairness value close to or over 0.8. Fig. 6 clearly shows that TSW only gets this value in a short range (35–45%), while PETER is over 0.8 in almost the entire range. This means that with PETER, all sources are getting their corresponding proportional part of the excess bandwidth. Observe that packet drops in the PETER method does not make sources to completely stop transmitting data. On the contrary, it makes sources to adapt to network conditions with the indications given by α_{ideal} and α^i_{m}.

5.3. Case 3: same contracted target rates and variation in round trip times

To evaluate the effect of having different round trip times in the network, we work in a scenario with eight TCP Reno sources where all of them have the same contracted target rate but the RTT goes from 10 to 80 ms. The influence that the RTT has on the final throughput is well known [17]. In heterogeneous scenarios, there is a bias against connections with large RTT unless this effect is alleviated in some way. Results reported in Table 3 show that targets are clearly fulfilled with PETER. We find the same problem for TSW as in previous cases, because in packets do not guarantee by themselves the contracts. It is important to remark that the diversity in RTT does not influence PETER performance.

The following figure, Fig. 7, reveals that PETER presents a fairness index above 0.8 for the entire network provisioning level (20–90%). TSW also shows good values for the fairness index but always below the former.

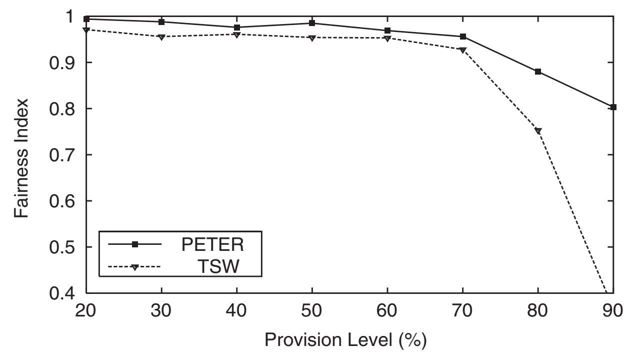

Fig. 7. Fairness index vs. provision level in case 3 with PETER and TSW.

Table 4. Throughputs obtained in case 4 with eight TCP Reno sources that have contracted target rates of 4–4–3–3–2–2–1 and 1 Mbps (provision level 60%)

s	TR	in packet throughput		Total throughput	
		PETER	TSW	PETER	TSW
1	4	3.99	3.70	4.58	4.19
2	4	3.99	3.75	5.82	5.77
3	3	2.99	2.77	4.68	5.01
4	3	2.99	2.95	4.25	4.92
5	2	1.97	1.84	2.96	3.85
6	2	1.98	1.86	3.04	3.57
7	1	0.98	0.86	1.44	2.77
8	1	0.98	0.91	1.42	2.57

The round trip time is set to 10–20–30–40–50–60–70 and 80 ms for connections s_1 to s_8, respectively.

5.4. Case 4: variation in contracted target rates and round trip times

For a more realistic environment, we study in this case the performance of PETER and TSW in a scenario with eight TCP Reno sources with different targets and different round trip times. The round trip times are set as in case 3 from 10 to 80 ms. Targets vary depending on the network load. For instance, for a 60% network provision level targets vary between 1 and 4 Mbps. From results shown in Table 4, PETER obtains a hard assurance of target rates for all connections, and so does TSW, but again with the help of out packets. Despite the heterogeneity of this scenario, Fig. 8 evidences the superiority of PETER to provide a fair excess bandwidth distribution. Meanwhile, TSW behaves worst as the heterogeneity increases.

5.5. Case 5: increment in number of sources

In this last case under consideration, we analyze the effect of incrementing the number of sources that arrive at the boundary node (E_1 in Fig. 3). We simulate a scenario with

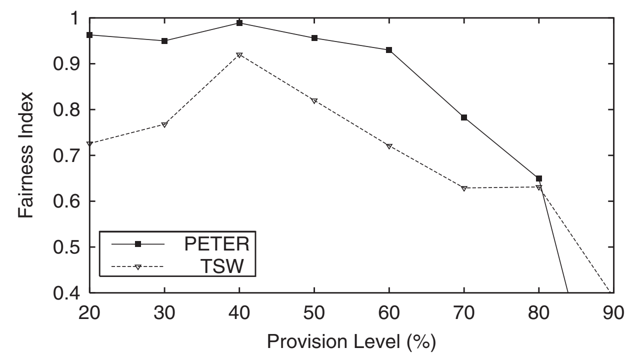

Fig. 8. Fairness index vs. provision level in case 4 with PETER and TSW.

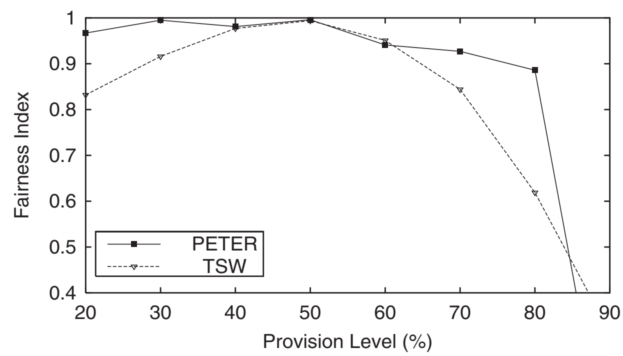

Fig. 9. Fairness index vs. provision level in case 5 with PETER and TSW.

Table 5. Throughputs obtained in case 5 with 16 TCP Reno sources that have contracted target rates of 1 Mbps (s_1 to s_4) and 1.32 Mbps (s_5 to s_{16}) with a network provision level of 60%

s	TR	in packet throughput		Total throughput	
		PETER	TSW	PETER	TSW
1	1.00	0.93	0.77	1.18	1.85
2	1.00	0.92	0.77	1.19	1.95
3	1.00	0.93	0.78	1.25	1.91
4	1.00	0.92	0.81	1.16	1.89
5	1.32	1.26	1.05	1.79	2.14
6	1.32	1.27	1.01	1.85	1.99
7	1.32	1.28	1.02	1.83	2.06
8	1.32	1.27	1.01	1.81	2.07
9	1.32	1.26	1.03	1.75	2.05
10	1.32	1.26	1.02	1.81	2.07
11	1.32	1.28	1.02	1.82	2.09
12	1.32	1.26	1.03	1.82	2.08
13	1.32	1.26	1.00	1.79	2.05
14	1.32	1.26	1.03	1.82	2.05
15	1.32	1.28	1.03	1.87	2.13
16	1.32	1.27	1.03	1.80	2.02

The round trip time is set to 50 ms for all connections.

16 TCP Reno sources, where the first four connections (s_1 to s_4) have always a contracted target of 1 Mbps. The other 12 sources (s_5 to s_{16}) contract the same target rates to fill a network provision level from 20% to 90%. For instance, sources s_5–s_{16} contract 1.32 Mbps each for a 60% provision level. Table 5 illustrates the good performance in assuring the contracts with our proposed traffic conditioner.

Fig. 9 represents the fairness index vs. the network provision level for the two schemes PETER and TSW. Although this scenario benefits both schemes (no round trip time variation), we see that PETER performs significantly better than TSW for nearly the entire range of provision level. This example shows the robustness of the DiffServ mechanism with PETER algorithm, since increasing the number of sources does not represent a degradation of the final performance.

6. Implementation concerns

Some doubts can arise about PETER at implementation time. As a first approach, the first question is, inside a DiffServ domain, how an edge node knows the link capacity and the contracted target rates to compute α_{ideal}. It is expected that the ISP is in charge of configuring all routers in its network (IP addresses, routing protocols, access lists, etc.), and this includes the QoS router configuration. In the example we are concerned with, all nodes in the domain should implement the AF PHB [1–3]. This means that they should use a proper scheduling algorithm and a buffer management mechanism such as RIO or some of its variants. Edge nodes know their serial interface (serial, isdn, frame relay, etc.) link capacity, so the only new task that the ISP should do is introducing the value of the total contracted bandwidth that this edge node serves. With this two parameters (link capacity and total contracted bandwidth), α_{ideal} is assessed.

In this paper, we do not consider the case where the edge node (the one that calculates α_{ideal}) has more than one interface to interior nodes, but the so-called stub networks (networks that have a single connection to its neighbor network). In the former case, PETER algorithm requires some modifications out of the scope of this paper.

7. Conclusions

In this paper, we propose a new control policy function for traffic conditioners that provides a strong assurance of contracted target rates and a fair distribution of spare bandwidth. We understand by fair distribution, the usage of the excess bandwidth proportionally to the contracted target rate of each source. The key to achieve a fair share is the performance of the PETER algorithm. Once packets are marked with one of two levels of precedence (in or out), PETER is the policy function applied in the traffic conditioner. The boundary node calculates the ratio excess bandwidth divided by the sum of all contracted target rates (α_{ideal}). This value is sent to traffic conditioners, placed next to TCP sources but out of the reach of the final users. Traffic conditioners measure the relation number of out packets divided by number of in packets (α_{m}^i). It is shown that for a fair share of excess bandwidth the relation $\alpha_{\mathrm{m}}^i = \alpha_{\mathrm{ideal}}$ has to be true, otherwise PETER acts moving the relation between α_{m}^i and α_{ideal} to the equality.

We extensively study the performance of our traffic conditioner for many different network conditions: variable target rates, variable round trip times, variability of both targets and delays, and increase of the number of sources that join in the boundary node. We observe that our scheme is able to guarantee contracted target rates in all cases and to offer simultaneously a proportional distribution of excess bandwidth for the network provision level in the range 20–80%, where it is supposed that most networks operate. Moreover, it performs better than the classical TSW. We conclude that it is possible to contract an AF Service satisfying Internet traffic guarantees, as shown in this paper.

Acknowledgments

This work was supported by the Spanish Research Council under Project TEC 2005-08068-C04-01/TCM.

References

[1] Blake S, Black D, Carlson M, Davies E, Wang Z, Weiss W. An architecture for Differentiated Services. RFC2475, 1998.
[2] Heinanen J, Baker F, Weiss W, Wroclawski J. Assured Forwarding PHB group. RFC2597, 1999.
[3] Grossman D. New terminology and clarifications for DiffServ. RFC3260, 2000.
[4] Cano M-D, Cerdan F, Garcia-Haro J, Malgosa-Sanahuja J. Performance evaluation of traffic conditioner mechanisms for the Internet Assured Service. Quality of Service over Next-Generation Data Networks. Proc SPIE 2001;4524:182–93.
[5] Lin W, Zheng R, Hou JC. How to make Assured Service more assured. In: Proceedings of the seventh international conference on network protocols (ICNP'99), Toronto, Canada, 1999. p. 182–91.
[6] Kusmierek E, Koodli R. Random packet marking for Differentiated Services. UMN technical report TR-00-020, Minneapolis, MN 55455. Department of Computer Science and Engineering, University of Minnesota, 2000.
[7] El-Gendy MA, Shin KG. Equation-based Packet Marking for Assured Forwarding Services. In: Proceedings of IEEE Infocom 2002, vol. 2, New York, USA, 2002. p. 845–54.
[8] Su H, Atiquzzaman M. ItswTCM: a new aggregate marker to improve fairness in Diffserv. Comput Commun 2003;26: 1018–27.
[9] Nandy B, Seddigh N, Pieda P, Ethridge J. Intelligent Traffic Conditioners for Assured Forwarding based Differentiated Services networks. In: Proceedings of Networking 2000, Paris, France, 2000. p. 540–54.
[10] Clark D, Fang W. Explicit allocation of best-effort packet delivery service. IEEE/ACM Trans Networking 1998;6: 362–73.
[11] Cano M-D, Cerdan F, Garcia-Haro J, Malgosa-Sanahuja J. Counters-based modified traffic conditioner. Lect Notes Comput Sci 2002;2511:57–67.
[12] Cerdan F, Casals O. Performance of different TCP implementations over the GFR service category. ICON J (Special Issue on QoS Management in Wired and Wireless Multimedia Communications Networks) 2000;2:273–86.
[13] Cerdan F, Casals O. Mapping an Internet Assured Service on the GFR ATM Service. Lect Notes Comput Sci 2000;1815:398–409.
[14] Bonin V, Cerdan F, Casals O. A simulation study of differential buffer allocation. In: Proceedings of third international conference on ATM, Germany, 2000. p. 365–72.
[15] Floyd S, Jacobson V. Random Early Detection gateways for congestion avoidance. IEEE/ACM Trans Networking 1993;1:397–413.
[16] Jain R. The art of computer systems performance analysis. New York, USA: Wiley; 1991.
[17] Seddigh N, Nandy B, Pieda P. Bandwidth assurance issues for TCP flows in a Differentiated Services networks. In: Proceedings of IEEE Globecom'99, vol. 3, Rio de Janeiro, Brazil, 1999. p. 1792–98.

Maria-Dolores Cano was born in Spain in 1976. She received the Degree in Telecommunications Engineering in 2000 from the Polytechnic University of Valencia (UPV), Spain, and received the Ph.D. Degree from the Polytechnic University of Cartagena, Spain, in 2004. In 2000, she joined the Polytechnic University of Cartagena, where she is an Associate Professor at the Department of Information Technologies and Communications. She has published several papers in conferences and international journals in the areas of traffic control, service integration, and QoS in high-speed networks. Her research interests include QoS provisioning in IP telecommunications networks, wireless networks, and security in telecommunications networks.

Fernando Cerdan obtained the Engineering Degree in Telecommunications in 1994 from the Polytechnic University of Catalunya and received the Ph.D. Degree from the same University in 2000. In 1996, he started working towards a Ph.D. Degree in Telecommunication supported by a National Grant. He is currently a full Associate Professor at the Polytechnic University of Cartagena. Most of his published papers are in the field of traffic control, service integration and QoS in wired IP networks, although his current interest includes security and QoS in 3G and heterogeneous networks.