A Markovian Model for TCP Analysis in a Differentiated Services Network [⋆]

Chadi Barakat and Eitan Altman
INRIA, 2004 route des Lucioles, 06902 Sophia Antipolis, France
{cbarakat,altman}@sophia.inria.fr

Abstract. Many schemes have been proposed to support TCP traffic in a Differentiated Services network. We present in this paper an analytical model to study the performance of these schemes. The model is based on a Markovian fluid approach. We provide first a general version of the model, then we specify it to the different proposed schemes. For each scheme, we calculate the throughput achieved by a TCP connection. We compare then their service differentiation capacity under different subscription levels, different reservations, and different round-trip times.

1 Introduction

There has been an increasing interest these last years in enhancing the best effort service of the Internet to provide new applications with some guarantees in terms of bandwidth, losses, and end-to-end delay. Differentiated Services architecture (DiffServ) is considered as the most promising approach in this field for reasons of scalability and incremental deployment [3, 14]. Flows are monitored at the edge of the network. Their parameters (rate, burst size) are compared to the contract signed between the user and the service provider. Compliant packets are marked with a high priority. Non-compliant packets are shaped, rejected, or injected into the network with a low priority. In the core of the network, priority packets are privileged over non-priority ones. This privilege can be in the form of a better scheduling (e.g., priority scheduling) as in the Premium service architecture [11, 14], or in the form of a lower drop probability as in the Assured service architecture [3, 8]. The main advantage of the DiffServ framework is that packets in the network are treated as a function of their priority not as a function of the flow they are belonging to. This makes the framework scalable, flexible, and easy to introduce into the Internet.

The utility of such framework to applications using UDP, the best effort transport protocol of the Internet, is evident. An example of such applications is a real time video or audio communication tool. If the network is well dimensioned, these applications are able to realize the throughput they desire. The problem appears with applications using TCP [10], the connection-oriented transport protocol of the Internet. An application using TCP may ask the network for a better service (i.e., more throughput) by reserving a certain bandwidth. If at the edge of the network non-compliant packets are rejected, TCP will reduce its

[⋆] An extended version of this paper is available upon request from the authors.

rate when it reaches the reserved bandwidth. The application fails in this case to use the bandwidth it is paying for as well as any unreserved bandwidth in the network. The solution to this problem is to let non-compliant packets get into the network as low priority packets. This improves TCP performance since the rate can now reach larger values. But, TCP is not aware of the reservation. The loss of a low priority packet is not distinguished from the loss of a high priority packet and the rate is reduced in the same manner. This has been shown to result in an unfairness in the distribution of network resources [2, 3, 5, 6, 17, 18]. The main result of these studies is that TCP is unable to realize its target throughput in a DiffServ network. The target throughput is defined as the reserved bandwidth plus a fair share of any unreserved bandwidth. A connection with a small reservation has been shown to achieve better performance than a connection with a large reservation. These works show also the well known problem of TCP unfairness in presence of different round-trip times (RTT). A connection with small RTT achieves better performance than a connection with long RTT. Some solutions have been proposed to alleviate these problems. They consist in either changing TCP sources, or marking TCP flows differently at the edge of the network, or changing the behavior of network routers. The performance of these solutions has been often evaluated via simulations in [3, 5, 6, 17]. In [18], a mathematical model has been proposed to calculate the throughput of a connection as a function of the drop probability of its packets. Three schemes have been compared. But, this model is not able to study the impact of the parameters of the other connections (e.g., RTT, reserved bandwidth). Also, it makes the simplistic assumption that TCP window varies in a cyclic manner with all the cycles having the same duration. Our experimentations over the Internet have shown that this assumption is not so realistic [1].

In this paper we present a general Markovian model able to a) calculate the performance of all the connections sharing a bottleneck b) account for the different solutions already proposed, or to be proposed. Using this model, we compare the performance of some schemes proposed to support TCP in a DiffServ network. In the next section we outline the different schemes we are considering in this paper. In Section 3 we explain our model. In Section 4 we calculate the throughput of a TCP connection as a function of two parameters. By appropriately setting these two parameters, we are able to specify our model to any one of the proposed schemes. Section 5 explains how these two parameters must be set. In Section 6 we present some numerical results.

2 TCP in a DiffServ Network

The main objectives of a DiffServ scheme supporting TCP traffic are:

- The available bandwidth must be efficiently utilized.
- In the case where the sum of the reservations is less than the total throughput (the under-subscription case), each connection must realize its reservation. The difference between the total throughput and the total reservation must be shared equally between the different connections.

– In the case where the sum of the reservations is greater than the total throughput (the over-subscription case), the total throughput must be distributed between the different connections proportionally to their reservations.

The original proposition to support TCP in a DiffServ network is due to Clark [3]. Two priority levels have been proposed for packet marking. A packet that arrives at the edge of the network and finds the rate of the connection smaller than the reservation, is marked with a high priority and is called an IN packet, otherwise it is marked with a low priority and it is called an OUT packet. The service differentiation comes from the different probabilities network routers drop these two types of packets at the onset of congestion. A variant of RED (Random Early Detection) buffers [7] is proposed to implement this difference in the drop probability. This variant, called RIO (RED IN/OUT) [3], has two minimum thresholds instead of one. When the average queue length exceeds the lower minimum threshold, OUT packets are probabilistically dropped in order to signal congestion to TCP sources. The buffer starts to drop probabilistically IN packets when the average queue length (sometimes the average number of IN packets in the buffer) exceeds the upper minimum threshold. This scheme has however some problems to satisfy the above objectives. Due to the saw tooth window variation of TCP (Figure 1), a connection is obliged to transmit a certain amount of OUT packets in order to realize its reservation. Since OUT packets are very likely to be dropped, the reservation may not be realized. Moreover, a connection with a large reservation is obliged to transmit more OUT packets than a connection with a small reservation. Also, a connection with a large reservation has in general larger window than that of a connection with a small reservation which makes it more affected by the loss of an OUT packet. This explains the bias of the scheme proposed in [3] against connections with large reservations.

The first and the most intuitive solution to this problem is to change TCP in a way that the source reduces differently its window when OUT or IN packets are lost [5, 17]. The source is supposed to know the priority of the lost packet. Also, it is supposed to know the bandwidth reserved by the connection. The loss of an IN packet is an indication that the congestion window must be divided by two as in standard TCP. The loss of an OUT packet is an indication that the unreserved bandwidth in the network is congested. The source reduces then its window by half the number of OUT packets it has in the network. The main problem with this solution is that it requires a change at the source and a knowledge of the priority of the lost packet which are two difficult tasks to implement.

The other solutions try to help connections with large reservations to send more OUT packets than connections with small reservations, and this is without changing TCP algorithms. The first solution proposed in [3] is based on the saw tooth variation of TCP window. To be able to realize its reservation, a TCP connection must be protected from the other connections until it reaches 4/3 of its reservation (see Figure 1). The idea [3] is to change the marker so that it marks packets as OUT when the rate of the connection exceeds 4/3 of the reserved bandwidth. We call this proposition the *Saw Tooth Marking* scheme. It has the drawback of injecting into the network during some periods more IN packets than we promised it.

The second solution [17] proposes to drop OUT packets in network routers according to the reserved bandwidth. The authors [17] show that dropping OUT packets with a probability inversely proportional to the bandwidth reserved by the connection improves the performance. We call this solution the *Inverse Drop Probability* scheme. Its main drawback is that it requires that network routers know the bandwidth reserved by every connection.

The last scheme we consider is the one that proposes to mark the packets with three priority levels instead of two [8, 9, 17]. A RED buffer with three thresholds is used. The idea is to protect the OUT packets of a connection transmitting at less than its fair share from the OUT packets of another connection exceeding its fair share, by giving them some medium priority while giving low priority to those of the latter connection. The difference from Saw Tooth Marking is that we are not injecting into the network more high priority packets than what we promised.

3 The Markovian Fluid Model

Consider N long life TCP connections sharing a bottleneck of bandwidth μ. Let $X^i(t)$ be the transmission rate of connection i at time t. It is equal to the window size divided by the RTT of the connection. The connections increase their rates until the network gets congested. The congested router starts to drop packets in order to signal the congestion to TCP sources. A source receiving a congestion signal (i.e. by detecting the loss of packets) reduces its rate, then it resumes increasing it. Let t_n denote the time at which the nth congestion event occurs and let $D_n = t_{n+1} - t_n$. Denote by X^i_n the transmission rate of connection i at time t_n, and by X^i_{n+} its transmission rate just after t_n (after the disappearance of the congestion). X^i_{n+} is equal to X^i_n if connection i did not reduce its rate at t_n and to $R_i(X^i_n)$ otherwise. $R_i(X^i_n)$ is a function of X^i_n usually equal to $X^i_n/2$ [10]. We introduce now some assumptions:
Assumption 1: We assume first that queueing times in network nodes are small compared to the propagation delay. This holds with the active buffer management techniques (e.g., RED [7]) that keep the queue length at small values. The RTT of a connection say i is then approximately constant equal to T_i and the rate of the connection can be considered to increase linearly with time between congestion events (by one packet every bT_i, with b equals the number of data packets covered by an ACK). We can write $X^i_{n+1} = X^i_{n+} + \alpha_i D_n$. This is ture if the slow start phases and timeout events are rare which is the case of new versions of TCP in case of long transfers [4]. We can also consider that the congestion appears when the sum of the transmission rates reach μ. Thus, instants t_n are given by $\sum_{i=1}^{N} X^i(t_n) = \mu$.
Assumption 2: The second assumption we made is that only one connection reduces its rate upon a congestion and that the probability that a connection reduces its rate is a function of its rate and the rates of the other connections

at the moment of congestion. This is again the aim of the new buffer management techniques (e.g. RED [7]) that implement random drop in order to sent congestion signals to connections consuming more than their fair share of the bottleneck bandwidth while protecting connections consuming less than their fair share [7]. We assume that the reaction of the connection receiving the first congestion signal is quick so that the congestion in the network disappears before other packets from other connections are dropped.

Let U_n^i be a random variable equal to 1 if source i reduces its rate at time t_n and to 0 otherwise. Using Assumption 2, we have always $\sum_{i=1}^N U_n^i = 1$. The probability that U_n^i is equal to one is a function of the connection rates at time t_n. Let $p_i(X_n^1, X_n^2, \ldots, X_n^N)$ denote this probability. It represents the probability that the dropped packet upon congestion belongs to connection i. This probability together with $R_i(X_n^i)$ form the two parameters that need to be appropriately chosen in order to cover all the proposed schemes.

Theorem 1. *The process $\{X_n^1, X_n^2, \ldots, X_n^N\}$ can be described as a homogeneous Markov process of dimension $N-1$.*

Proof. The transmission rates at time t_n are related by $\sum_{i=1}^N X^i(t_n) = \mu$. Thus, the problem can be analyzed by considering only the rates of $N-1$ connections. In the particular case of $N=2$ we get a one-dimensional problem. Concerning the Markovian property of the model, it is easy to show that the state of the process at time t_{n+1} depends only on its state at time t_n. Indeed, for any i we have,

$$X_{n+1}^i = X_n^i + U_n^i(R_i(X_n^i) - X_n^i) + \alpha_i D_n. \tag{1}$$

Summing over all the i, we get,

$$D_n = \frac{\sum_{i=1}^N U_n^i(X_n^i - R_i(X_n^i))}{\sum_{i=1}^N \alpha_i}. \tag{2}$$

Given that $R_i(X_n^i)$ and the value taken by U_n^i are only a function of the process state at time t_n, X_{n+1}^i is only a function of the process state at time t_n and the Markovian property holds. The process is homogeneous since its state at time t_{n+1} depends only on its state at time t_n and not on n. $\diamond$

The transmission rate of a connection takes a finite number of values between 0 and μ. This number depends on the RTT of the connection and the packet size. Denote by $\mathcal{X}$ the state space of our chain. For each state $X = (x_1, \ldots, x_N) \in \mathcal{X}$, the chain can jump to N different states at the next congestion event. This depends on which source reduces its rate. Denote by $F^i(X) = (f_1^i, \ldots, f_N^i)$ the next state when the system is in state X and source i reduces its rate. Using (1),

$$f_j^i = \begin{cases} x_j + (x_i - R_i(x_i))\alpha_j / \sum_{m=1}^N \alpha_m & \text{if } j \neq i \\ R_i(x_i) + (x_i - R_i(x_i))\alpha_i / \sum_{m=1}^N \alpha_m & \text{if } j = i \end{cases}$$

Let $\mathcal{P} = (P_{XY})_{X,Y \in \mathcal{X}}$ denote the transition matrix of the chain. We have,

$$P_{XY} = \begin{cases} p_i(X) & \text{if there is an } i \text{ such that } Y = F^i(X) \\ 0 & \text{otherwise} \end{cases}$$

Denote by $\Pi = (\pi_X)_{X \in \mathcal{X}}$ the stationary distribution of our chain. We solved the system numerically for different scenarios and we found that Π always exists and is unique. Let X^i, U^i, and D represent the processes X_n^i, U_n^i, and D_n in the unique stationary regime.

4 Calculation of the Throughput

The throughput of a connection say i (or the time average of its transmission rate) is equal to

$$\begin{aligned}\bar{X}^i &= \lim_{t\to\infty} \frac{1}{t}\int_0^t X^i(u)du = \lim_{n\to\infty} \frac{\sum_{m=0}^{n-1}\int_{t_m}^{t_{m+1}} X^i(u)du}{\sum_{m=0}^{n-1} D_m} \\ &= \lim_{n\to\infty} \frac{\frac{1}{n}\sum_{m=0}^{n-1}(X_m^i + U_m^i(R_i(X_m^i) - X_m^i))D_m + \alpha_i(D_m)^2/2}{\frac{1}{n}\sum_{m=0}^{n-1} D_m} \\ &= \frac{E[(X^i + U^i(R_i(X^i) - X^i))D + \alpha_i(D)^2/2]}{E[D]} \qquad (3)\end{aligned}$$

Let $D^j(X)$ denote the time until the next congestion event when the system is in state $X \in \mathcal{X}$ and source j reduces its rate. Using (2) we have, $D^j(X) = (x_j - R_j(x_j))/\sum_{m=1}^{N}\alpha_m$. Thus, $\bar{X}^i =$

$$\frac{\sum_{X\in\mathcal{X}} \pi_X \left(\sum_{j=1}^{N} p_j(X)\left(x_i D^j(X) + \alpha_i(D^j(X))^2/2\right) + p_i(X)(R_i(x_i) - x_i)D^i(X)\right)}{\sum_{X\in\mathcal{X}} \pi_X \sum_{j=1}^{N} p_j(X)D^j(X)} \qquad (4)$$

5 Application of the Model to Real Schemes

Suppose that the system is in state $X = (x_1, \ldots, x_N) \in \mathcal{X}$ in the stationary regime. In order to calculate the throughput, we find in this section the expressions of the two functions $p_i(X)$ and $R_i(x_i)$ for every scheme.

5.1 Standard TCP with RIO

A source i asks the network for a bandwidth μ_i. When the congestion appears at the bottleneck, the router starts to drop OUT packets with a certain probability. If there is no OUT packets in the network, congestion remains and the router starts to drop probabilistically IN packets. When an OUT or IN packet is dropped, the corresponding connection divides its rate by two. Thus, $R_i(x_i) = x_i/2$ in this case and in all the subsequent cases where standard TCP is used.

The probability that a connection reduces its rate upon a congestion is equal to 0 when it is transmitting only IN packets and there is at least one connection transmitting OUT packets. It is equal to 1 if it is the sole connection transmitting OUT packets. Now, we study the case where the connection is transmitting OUT packets together with other connections. The other case, that of all the connections transmitting only IN packets, will be directly deduced.

Let q be the probability that an OUT packet is dropped at the bottleneck upon congestion. Let V be the result of the probabilistic drop applied to a packet. It is equal to 1 if the packet is really dropped and to zero otherwise. Denote by Y the number of the connection to which the dropped OUT packet belongs. In the following we denote by $P_X(A)$ the probability that event A happens given that the system is in state $X \in \mathcal{X}$ upon congestion. We have,

$$p_i(X) = P_X(Y = i|V = 1) = \frac{P_X(Y = i).P_X(V = 1|Y = i)}{\sum_{m=1}^{N} P_X(Y = m).P_X(V = 1|Y = m)}$$

$P_X(V = 1|Y = m)$ is no other than q. Thus, $p_i(X)$ is equal to $P_X(Y = i)$ which is equal to the ratio of the rate at which connection i is sending OUT packets and the total rate at which OUT packets are sent. Thus,

$$p_i(X) = P_X(Y = i) = \frac{x_i - \mu_i}{\sum_{m=1}^{N}(x_m - \mu_m)1\{x_m > \mu_m\}},$$

where $1\{\}$ is the indicator function.

Similarly, when all the connections are transmitting only IN packets, $p_i(X)$ is equal to $P_X(Y = i)$ which is equal to the ratio of the rate at which connection i is sending IN packets and the total rate at which IN packets are sent. It follows,

$$p_i(X) = \begin{cases} x_i/\mu & \text{if } \sum_{m=1}^{N} 1\{x_m > \mu_m\} = 0 \\ ((x_i - \mu_i)1\{x_i > \mu_i\}) \,/\, \left(\sum_{m=1}^{N}(x_m - \mu_m)1\{x_m > \mu_m\}\right) & \text{otherwise} \end{cases}$$

5.2 Modified TCP with RIO

$p_i(X)$ is the same as in the previous section. The difference is in the function $R_i(x_i)$. If an IN packet is lost, the source divides its rate by two as with standard TCP. If the dropped packet is an OUT packet, only the rate of OUT packets is divided by two. We consider in our model that the dropped packet from connection i is an IN packet if at the moment of congestion source i is transmitting at less than its reservation, otherwise it is an OUT packet. Thus,

$$R_i(x_i) = \begin{cases} x_i/2 & \text{if } x_i < \mu_i \\ \mu_i + (x_i - \mu_i)/2 & \text{otherwise} \end{cases}$$

5.3 Inverse Drop Probability Scheme

Standard TCP is used, therefore $R_i(x_i)$ is equal to $x_i/2$. The difference in this case is that packets of different connections (IN or OUT) are not treated in the same manner in the core of the network. The idea proposed in [17] is to drop OUT packets from a connection with a probability that varies as the inverse of its reservation. However, the drop probability of IN packets is not specified. IN packets are actually dropped when all the connections are transmitting at less than their reservations. In this case and according to our objectives (Section 2), the throughput of a connection must be proportional to its reservation. It is known that the throughput of a TCP connection varies as the square root of the packet drop probability [1, 13, 15]. Thus, we propose to drop IN packets with a probability that varies as the inverse of the square of the reservation. We add this new feature to the proposed scheme.

As in the case of RIO with standard TCP, a connection reduces its rate with probability 1 if its the sole connection exceeding its reservation and with probability 0 if it is transmitting only IN packets and if there is at least one connection transmitting OUT packets. For the remaining two cases, we consider first the case when the connection is transmitting OUT packets together with other connections. The other case will be directly deduced.

Suppose that the bottleneck router drops OUT packets of source m with a probability q/μ_m, q is a constant. Then,

$$\begin{aligned} p_i(X) &= \frac{P_X(Y=i).P_X(V=1|Y=i)}{\sum_{m=1}^N P_X(Y=m).P_X(V=1|Y=m)} = \frac{P_X(Y=i)/\mu_i}{\sum_{m=1}^N P_X(Y=m)/\mu_m} \\ &= \frac{x_i/\mu_i - 1}{\sum_{m=1}^N (x_m/\mu_m - 1)1\{x_m > \mu_m\}} \end{aligned}$$

The general expression of $p_i(X)$ for this scheme is given by

$$p_i(X) = \begin{cases} \left(x_i/\mu_i^2\right) / \left(\sum_{m=1}^N x_m/\mu_m^2\right) & \text{if } \sum_{m=1}^N 1\{x_m > \mu_m\} = 0 \\ ((x_i/\mu_i - 1)1\{\mu_i > x_i\}) / \left(\sum_{m=1}^N (x_m/\mu_m - 1)1\{x_m > \mu_m\}\right) & \text{otherwise} \end{cases}$$

5.4 Saw Tooth Marking Scheme

Standard TCP, two priority levels, and RIO buffers are used. Thus, $R_i(x_i) = x_i/2$. The difference here is in the marker operation. The flow of connection i contains OUT packets when its rate x_i exceeds $4\mu_i/3$. The rate of its OUT packets at the moment of congestion is equal to $x_i - 4\mu_i/3$ rather than $x_i - \mu_i$. The new expression of $p_i(X)$ is then

$$p_i(X) = \begin{cases} x_i/\mu & \text{if } \sum_{m=1}^N 1\{x_m > 4\mu_m/3\} = 0 \\ ((x_i - 4\mu_i/3)1\{x_i > 4\mu_i/3\}) / \left(\sum_{m=1}^N (x_m - 4\mu_m/3)1\{x_m > 4\mu_m/3\}\right) & \\ & \text{otherwise} \end{cases}$$

5.5 Standard TCP with Three Drop Priorities

In this scheme the source makes two reservations instead of one. Denote these reservations by μ_i^1 and μ_i^2 with $\mu_i^1 < \mu_i^2$. Standard TCP is used at the source, then $R_i(x_i) = x_i/2$. Packets are marked with three priority levels or three colors. Packets exceeding μ_i^2 are marked with low priority (red color). Those exceeding μ_i^1 but not μ_i^2 are marked with medium priority (yellow color). Packets sent at a rate slower than μ_i^1 are marked with a high priority (green color).

As in the RIO case, the network starts first to drop low priority packets. This happens when one of the sources, say i, is exceeding its upper reservation μ_i^2. If those packets don't exist, medium priority packets are dropped. Medium priority packets exist in the network when one of the sources say i is exceeding its lower

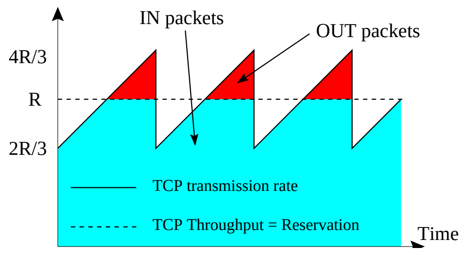

Fig. 1. The saw tooth variation of TCP rate

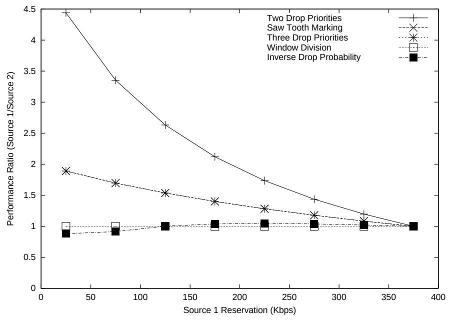

Fig. 2. Performance comparison for a 50% total reservation

reservation μ_i^1. If it is not the case, the network drops high priority packets. A connection reduces its rate with probability one if it is transmitting alone above a certain level. It reduces its rate with probability 0 if it is transmitting below a level and there is another connection transmitting above the same level. In the other cases, the probability that a connection reduces its rate is equal to the probability that the dropped packet belongs to this connection. Similarly to the RIO case we write,

$$p_i(X) = \begin{cases} x_i/\mu & \text{if } \sum_{m=1}^{N} 1\{x_m > \mu_m^1\} = 0 \\ \left((x_i - \mu_i^1)1\{x_i > \mu_i^1\}\right) / \left(\sum_{m=1}^{N}(x_m - \mu_m^1)1\{x_m > \mu_m^1\}\right) & \\ \text{if } \sum_{m=1}^{N} 1\{x_m > \mu_m^1\} > 0 \text{ and } \sum_{m=1}^{N} 1\{x_m > \mu_m^2\} = 0 & \\ \left((x_i - \mu_i^2)1\{x_i > \mu_i^2\}\right) / \left(\sum_{m=1}^{N}(x_m - \mu_m^2)1\{x_m > \mu_m^2\}\right) & \text{otherwise} \end{cases}$$

To compare this scheme to previous ones, μ_i^1 and μ_i^2 must be set as a function of the desired throughput μ_i. We looked at the saw tooth variation of TCP rate (Figure 1). On average and in order to realize a throughput μ_i, the connection rate should vary between $2\mu_i/3$ and $4\mu_i/3$. Thus, we give packets below $2\mu_i/3$ the highest priority, packets between $2\mu_i/3$ and $4\mu_i/3$ the medium priority, and packets above $4\mu_i/3$ the lowest priority. This corresponds to $\mu_i^1 = 2\mu_i/3$ and $\mu_i^2 = 4\mu_i/3$. This scheme is compared in the sequel to the other schemes with these particular values of the two reservations. Other values could be however used.

6 Some Numerical Results

We solve numerically our model for the case of two concurrent TCP connections. We take μ =1.5 Mbps and we set TCP segments to 512 bytes. Reservations are expressed in kbps. The receivers are supposed to acknowledge every data packet ($b = 1$). First, we give the two connections the same RTT (100ms) and we study the performance of the different schemes under different reservations and different subscription levels. Second, we study the impact of a difference in RTT on the service differentiation provided by the different schemes.

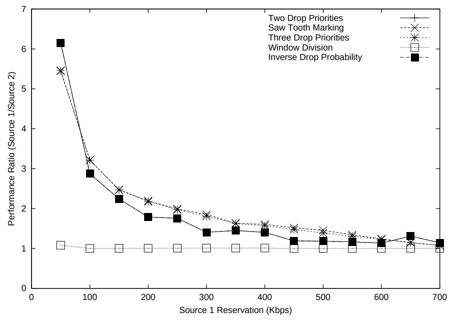

Fig. 3. Performance comparison for a 100% total reservation

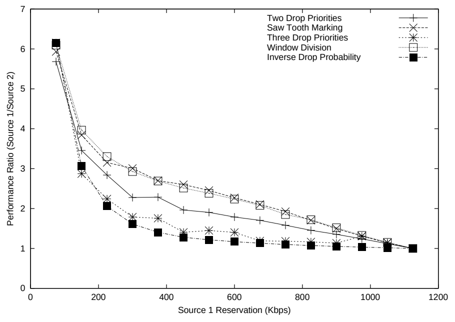

Fig. 4. Performance comparison for a 150% total reservation

6.1 Impact of the Reservation

We change the reservations of the two sources in a way that their sum is constant and equal to $\rho\mu$; ρ indicates the level of subscription. We consider three values of ρ: 0.5, 1 and 1.5. For each ρ and according to the objectives in Section 2, we define a factor F that characterizes how much connection 1 is favored with respect to connection 2. For $\rho < 1$, the network is under-subscribed and the two sources must share fairly the excess bandwidth. We define F as the ratio of $\bar{X}^1 - \mu_1$ and $\bar{X}^2 - \mu_2$. The optimum scheme is the one that gives the closest F to one. An $F > 1$ means that the scheme is in favor of connection 1. For $\rho \geq 1$, the network is over-subscribed. The bandwidth must be shared proportionally to the reservation. We define F in this case as being the ratio of $\bar{X}^1/\mu_1$ and $\bar{X}^2/\mu_2$. Again, the optimum scheme is the one that gives the closest F to one, and an $F > 1$ means that the scheme is in favor of connection 1.

In Figures 2, 3, and 4, we plot F for the three cases $\rho = 0.5$, 1, and 1.5. The X-axis shows the reservation of source 1. For all the schemes and as one must predict, F converges to 1 when the the reservation of source 1 moves to that of source 2. In the under-subscription case, the original RIO scheme gives the worst service. The connection with the small reservation achieves better performance than that with the large reservation. The other schemes improve the service. They give connection 2 more chance to increase its rate above its reservation which improves its throughput. In the over-subscription case the situation changes. This is more depicted in Figure 4. In this case, the original RIO scheme gives better performance than the proposed schemes (except for the Three Color scheme). The problem here is that the source with the large reservation is transmitting almost always IN packets and cannot profit from the high priority we give to OUT packets. The increase in the priority of OUT packets helps the source with the small reservation which achieves better throughput.

6.2 Impact of Round-Trip Time

We study in this section how well the proposed schemes resist to a difference in RTT. We suppose that the two connections are asking for the same bandwidth ($\mu_1 = \mu_2$). We set T_2 to 50ms and we vary T_1 between 50ms and 500ms. Ideally,

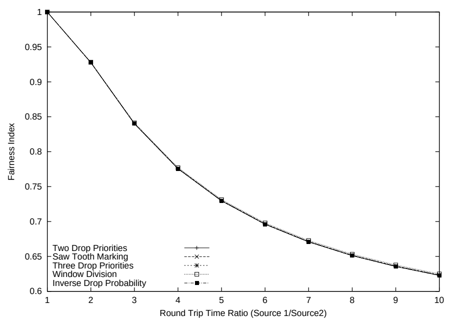

Fig. 5. Fairness index for a 0% total reservation

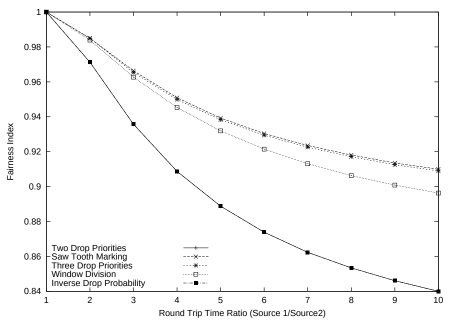

Fig. 6. Fairness index for a 50% total reservation

the two connections must achieve the same throughput independently of their RTT. To quantify the impact of T_1, we use the Fairness Index defined in [12]: $FI = (\bar{X}^1 + \bar{X}^2)^2 / \left(2((\bar{X}^1)^2 + (\bar{X}^2)^2)\right)$. This is an increasing function of fairness. It varies between 1/2 when one of the two connections is shut down, and 1 when the two connections realize the same throughput. We plot in Figures 5, 6, 7, and 8, FI as a function of T_1/T_2 for four values of ρ: 0, 0.5, 1 and 1.5. The zero reservation case corresponds to a best effort network. All the schemes achieve the same performance (Figure 5). The fairness deteriorates as T_1 increases. The small RTT connection (i.e., 2) gets better and better performance. A small reservation as for $\rho = 0.5$ protects the long RTT connection and improves the service. Indeed, as T_1 starts to increase, the throughput of connection 1 drops, but at a certain point, it fells below its reservation and the connection starts to send only high priority packets. It becomes then protected from the other connection. The schemes other than RIO with standard TCP improve further the service. With these schemes, the long RTT connection has more chances to stay above its reservation. In the over-subscription case the situation again changes. RIO with standard TCP gives better performance than that of the other schemes. The reason is that giving a connection more chances to exceed its reservation helps the connection with small RTT rather than the connection with long RTT.

6.3 Discussion of the Results

We summarize our results as follows: In the under-subscription case, the RIO scheme is more biased against large reservation connections and long RTT ones than the other schemes. This is not the case in the over-subscription case where it provides a better performance. Except for the Three Color scheme, the other schemes are useful in the first case. The Three Color scheme is useful in all the cases since the priority it gives to OUT packets is less than that of IN packets.

References

1. E. Altman, K. Avratchenkov, and C. Barakat, "A stochastic model for TCP/IP with stationary random losses", *ACM SIGCOMM*, Sep 2000.
2. A. Basu and Z. Wang," A Comparative Study of Schemes for Differentiated Services", *Bell labs Technical Report*, Aug 1998.

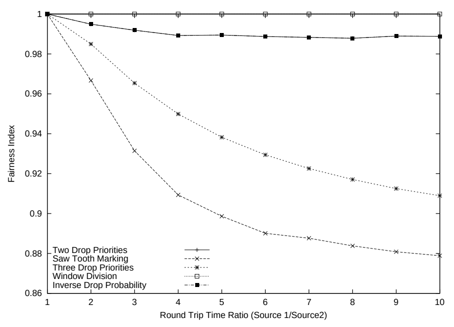

Fig. 7. Fairness index for a 100% total reservation

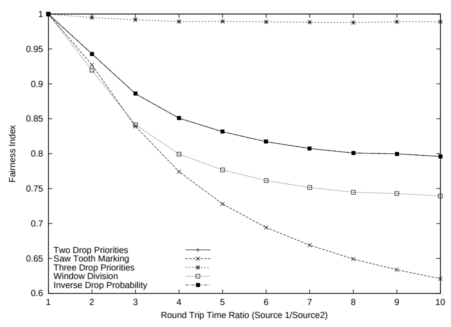

Fig. 8. Fairness index for a 150% total reservation

3. D. Clark and W. Fang, "Explicit Allocation of Best Effort Packet Delivery Service", *IEEE/ACM Transactions on Networking*, Aug 1998.
4. K. Fall and S. Floyd, "Simulation-based Comparisons of Tahoe, Reno, and SACK TCP", *Computer Communication Review*, Jul 1996.
5. W. Feng, D. Kandlur, D. Saha, and K. Shin, "Understanding TCP Dynamics in a Differentiated Services Internet", *IEEE/ACM Transactions on Networking*, Apr 1998.
6. W. Feng, D. Kandlur, D. Saha, and K. Shin, "Adaptive Packet Marking for Providing Differentiated Services in the Internet", *International Conference on Network Protocols*, Oct 1998.
7. S. Floyd and V. Jacobson, "Random Early Detection gateways for Congestion Avoidance", *IEEE/ACM Transactions on Networking*, Aug 1993.
8. J. Heinanen, T. Finland, F. Baker, W. Weiss, and J. Wroclawski, "Assured Forwarding PHB Group", *RFC 2597*, Jun 1999.
9. J. Heinanen, T. Finland, and R. Guerin, "A Two Rate Three Color Marker", *Internet Draft*, May 1999.
10. V. Jacobson, "Congestion avoidance and control", *ACM SIGCOMM*, Aug 1988.
11. V. Jacobson, K. Nichols, and K. Poduri, "An Expedited Forwarding PHB", *RFC 2598*, Jun 1999.
12. R. Jain, D. Chiu, and W. Hawe, "A Quantitative Measure Of Fairness And Discrimination For Resource Allocation In Shared Computer Systems", DEC Research Report TR-301, Sep 1984.
13. T.V. Lakshman and U. Madhow, "The performance of TCP/IP for networks with high bandwidth-delay products and random loss", *IEEE/ACM Transactions on Networking*, Jun 1997.
14. K. Nichols, V. Jacobson, and L. Zhang, "A Two-bit Differentiated Services Architecture for the Internet", *Internet Draft*,¡draft-nichols-diff-svc-arch-00.txt¿, Nov 1997.
15. J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP Throughput: a Simple Model and its Empirical Validation", *ACM SIGCOMM*, Sep 1998.
16. W. Stevens, "TCP Slow-Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms", *RFC 2001*, Jan 1997.
17. I. Yeom and A. Reddy, "Realizing throughput guarantees in Differentiated Services Networks", *TAMU-ECE-9806*, 1998.
18. I. Yeom and A. Reddy, "Modeling TCP behavior in a Differentiated-Services Network", *TAMU ECE Technical Report*, May 1999.