Reducing Packet-Loss by Taking Long-Range Dependences into Account

J. Ignacio Alvarez-Hamelin* and Pierre Fraigniaud

Laboratoire de Recherche en Informatique, Bât 490 Université Paris Sud, 91405 Orsay CEDEX, France, {ihameli,pierre}@lri.fr

Abstract. We show that the "fractal" behavior of Internet traffic can be efficiently and practically employed to significantly reduce packet-loss. Thanks to recent advances in the theory of self-similar processes, we define the *probabilistic congestion* of a link, based on an estimated computation of the packet-loss probability over that link. This congestion parameter allows valid predictions on the future behavior of the network, on which one can base efficient routing strategies. We show how to implement the computation of the probabilistic congestion, and we illustrate several applications for improving unicast and multicast protocols.

Keywords: Self-Similar Traffic, Routing, Multicast, IP Networks.

1 Introduction

Starting in the early 90's, there has been a number of empirical studies that provide evidence of the prevalence of self-similar traffic patterns in packet networks such as the Internet (see, e.g., [9,17]). This "fractal" behavior is very different both from conventional telephone traffic, and from standard models for packet traffic (e.g., Poisson). In particular, significant traffic variance (burstiness) is present on a wide range of time scales in self-similar traffic, and hence such traffic exhibits long-range dependencies (i.e., values at any instant are correlated with values at future instants). This paper aims to explore situations for which routing can take advantage of the long-range dependence nature of the traffic, and to provide effective solutions for such situations.

Self-similarity was observed in Local Area Networks [9] as well as in Wide Area Networks [17]. In [19,22] it is shown that the self-similarity at the LAN level can result from the superposition of on/off sources with strictly alternating on- and off-periods, and whose on-periods or off-periods have high variability. User related traffic, as World-Wide-Web (WWW) traffic [5,4], WWW workload [18], and Variable-Bit-Rate (VBR) video data [2,7] present characteristics that are consistent with self similarity. On the other hand, [16] shows that transport

* Also from the Facultad de Ingeniería of Universidad de Buenos Aires. Supported by the ALPHA program CORDIAL.

mechanisms are important factors in translating self-similarity from the application layer to the link layer. In particular, the flow control mechanisms of TCP seem to maintain the long-range dependency structure induced by heavy-tailed file size distributions, a phenomenon which is not observed to the same extent when using the non-flow-controlled UDP transport protocol (see [15]). In fact, it is shown in [21] that TCP congestion control "propagates" self-similarity. Roughly speaking, if a TCP stream meets another TCP stream which exhibits large time-scale fluctuations, then TCP will react in a way that causes the former traffic to inherit the self-similarity nature from the latter. In addition to that, [20] concludes that the congestion control of TCP itself, as a deterministic process, creates chaos, which generates self-similarity. There are hence many reasons why traffic in IP exhibits a self-similar behavior. This paper shows that this traffic property can be effectively employed to reduce packet-loss.

Packet-loss is the cause of important performance degradations in packet networks, for it is intrinsically related to all "standard" Quality of Services (QoS) measures: latency, bandwidth, jitter, etc. Using TCP, it is necessary to resends packets, with time-consuming effects. Using UDP, packet-loss has a significant impact on the user-perception for certain applications. IPv4 allows the use of a Type-of-Service (ToS) field in the IP-headers, for choosing the route with minimum packet-loss. However, this facility has not been implemented, or rarely used, because of the uncertainty of foreseeing the future behavior of a system with short-range dependencies, as Internet was assumed to be. On the other hand, recent advances in the theory of self-similar variation allow the anticipation of the future behavior of systems with long-range dependencies, as the Internet really is. Anticipating the traffic behavior allows efficient routing strategies to be developed, while preserving the best effort nature of IP, and in particular no resource reservation is required.

Various routing protocols include QoS in their definition, in the sense that they are able to set up the best routes according to pre-defined parameters such as number of hops, maximum delay, or available bandwidth. The route is chosen among a set of alternative paths proposed by the protocol. This is typically the case of the QoSMIC [6] and YAM [3] multicast routing protocols, as well as of any Dijkstra-like unicast routing protocol. Basically, we propose to perform these choices according to the probability of packet-loss.

We define the *probabilistic congestion* c of a link as a logarithmic transformation of the packet-loss probability p along that link. (The probabilistic congestion of a route is thus simply the sum of the probabilistic congestion of each of its links.) The probability p is estimated using the theory developed in [11,12,13], which proposes an explicit formula for the probability that a given buffer contains more than x bytes, assuming a plausible model of self-similar traffic. The computation requires the values of three parameters: the mean input rate m, the "variance coefficient" a, and the Hurst parameter H. The latter two parameters characterize the "quality" of the traffic, in contrast to the long run mean rate m which characterizes its "quantity" alone [12]. We show that these three parameters can be computed on-line at every router, using locally accessible traffic variables. They are estimated at time t based on the traffic observed during the last time period $[t - \Delta, t]$, for some Δ. Assuming a traffic with large time-scale fluctuations, we expect that the traffic will offer the same characteristics during the time interval $[t, t + \delta]$ as it did during $[t - \Delta, t]$, at least for a sufficiently small $\delta < \Delta$. In particular, the packet-loss probability computed based on parameters specific to the traffic observed during $[t - \Delta, t]$ is expected to not vary too much in the near future. We have grounds to believe in such a behavior of the packet-loss probability since we observed, based on real traffic observations on Abilene's backbone, that long-range dependences imply smooth variations of the probabilistic congestion c, as opposed to, e.g., the available bandwidth, which experiences abrupt variations.

In this paper, we demonstrate the interest of the probabilistic congestion by improving QoS-sensitive multicast routing protocols. More precisely, we compare the performance (in terms of packet-loss) of QoSMIC using probabilistic congestion vs. QoSMIC using standard dynamic criteria. Simulations were performed on Abilene's topology, as well as on UUNET's topology, for several group configurations. The simulated traffic was based on real traffic samples from Abilene, and hence offers realistic characteristics. In all cases, we observed a significant improvement for QoSMIC using probabilistic congestion compared to QoSMIC using any other dynamic criteria. Note that the multicast protocol itself is not modified, only the information collected by the nodes for the construction of the multicast tree differ: they collect probabilistic congestion rather than current delays or available bandwidth. The remaining of the protocol is exactly the same.

We also demonstrate the interest of the probabilistic congestion for improving performances of unicast routing. In particular, in a network supporting different classes of services, the routing paths of the priority classes can be chosen as the ones with minimum probabilistic congestion. Simulations on the Abilene's backbone demonstrate that this approach yields significant improvements in term of packet losses.

2 Probabilistic Congestion

We follow the model extensively explored by I. Norros in a series of papers (see, e.g., [11,12,13]). The self-similar variation of a packet-network traffic is modeled by a Gaussian self-similar process. A normalized fractional Brownian motion with self-similar parameter $H \in [\frac{1}{2}, 1)$ is a stochastic process Z_t, $t \in \mathbb{R}$, characterized by the following properties: (1) Z_t is Gaussian, (2) with probability 1, $t \mapsto Z_t$ is continuous, (3) $\mathbf{E}Z_t^2 = |t|^{2H}$ for all t, (4) $Z_0 = 0$, and $\mathbf{E}Z_t = 0$ for all t, and (5) for all t, and $s_1 < s_2 < \ldots < s_k$, the distribution of $(Z_{t+s_2} - Z_{t+s_1}, \ldots, Z_{t+s_k} - Z_{t+s_{k-1}})$ is independent of t. Note that, for $H = \frac{1}{2}$, Z_t is the standard Brownian motion [10]. The parameter H is the Hurst parameter. Let us denote by A_t the amount of traffic (in bytes, say) offered to a link of the network during time interval $[0, t)$. Norros models A_t as a so called fractional

Brownian *traffic*, that is

$$A_t = mt + \sqrt{am}Z_t \ , \tag{1}$$

where $m > 0$ is the mean input rate, and $a > 0$ is a variance coefficient. The self-similarity is captured by the Brownian scaling relation stating that, for any $t > 0$, $\{Z_{st}, s \geq 0\}$ and $\{t^H Z_s, s \geq 0\}$ have the same finite-dimensional distributions. The traffic fluctuations buffering of a fractional Brownian traffic of parameters m, a, and H offered to a link of capacity $C > m$ with infinite queue length is then defined as

$$X_t = \sup_{s \leq t} \Big(A_t - A_s - C(t-s) \Big).$$

Norros [12] derived a lower bound of the probability that the local storage exceeds a certain limit:

$$\mathbf{Pr}(X_t > x) \geq \bar{\Phi}\Big(\frac{(C-m)^H x^{1-H}}{H^H (1-H)^{1-H}\sqrt{am}}\Big) \ , \tag{2}$$

where $\bar{\Phi}(y) = \mathbf{Pr}(Z_1 > y)$, that is the residual distribution function of the standard Gaussian distribution. Since $\bar{\Phi}(y) \sim e^{-y^2/2}$, $\mathbf{Pr}(X_t > x)$ can hence be lower bounded by a Weibull distribution, in particular with regards to the tail behavior. Norros performed simulation to check the accuracy of the bound in Eq. (2). The queue length process was generated by the usual formula

$$X_{k\tau} = \Big(X_{(k-1)\tau} - C\tau + A_{k\tau} - A_{(k-1)\tau} \Big)^+ \ , \tag{3}$$

where τ is the resolution of the experimental sample, and $x^+ = x$ if $x > 0$, and 0 otherwise. Interestingly, one observation coming out from the simulations is that the Weibull approximation $\mathbf{Pr}(X_t > x) \sim Y$ where Y is the right side of Eq. 2 is a satisfactory accurate approximation. This motivates us to define the following parameter:

Definition. The *probabilistic congestion* c of a link with buffer size b supporting self-similar traffic of parameters m, a and H, is defined by $c =$

$$-\ln\left(1 - \exp\Big(- \frac{(C-m)^{2H}}{2H^{2H}(1-H)^{2-2H}am}\ b^{2-2H}\Big)\right) \ . \tag{4}$$

According to the Weibull approximation, we have $c \sim -\ln(1 - \mathbf{Pr}(X_t > b))$, and hence, up to the logarithmic rescaling, the probabilistic congestion follows the same behavior as the packet-loss probability. In particular, given a path $P = \{e_1, \ldots, e_k\}$, with probabilistic congestion of e_i equal to c_i, the probabilistic congestion of P is naturally defined as $\sum_i c_i$.

The probabilistic congestion c of any link can be approximated by its tail router. For each of its outgoing link e, every router counts the number of bytes that were sent through e during specific time-intervals of length t_u. At time t, let A_i, $1 \leq i \leq N$, be the number of bytes offered to link e during time-interval

$I_i = (t - it_u, t - (i-1)t_u]$. Then let $B_i = A_i/t_u$ be the average throughput of link e during I_i. The mean rate m is simply estimated by $\overline{B} = \frac{1}{N}\sum_{i=1}^{N} B_i$, and the variance coefficient a is estimated by $\sigma^2(B)/m$. The auto-covariance of the B_i's is $\dot{B}_k = \sum_{i=1}^{N}(B_i - \overline{B})(B_{i-k} - \overline{B})$, for $k = -N+1, \ldots, N-1$. The $\dot{B}_k$'s, $k \geq 0$, plot is a second order self-similar process $x \mapsto \alpha e^{-\beta x}$, and the Hurst parameter characterizing the traffic observed through e between times $t - Nt_u$ and t is $H = 1 - \beta/2$. (Here $\Delta = N t_u$.) $\dot{B}$ can be computed rapidly thanks to a constant number of Fast Fourier Transforms (FFT): $\dot{B} = FFT^{-1}(\widehat{B} \cdot \overline{\widehat{B}})$ where $\widehat{B} = FFT(B)$, and $\overline{x}$ is the conjugate of x. Then H can be easily obtained by a linear regression applied on $\log \dot{B}$. Once m, a, and H have been estimated, the probabilistic congestion c of link e is computed according to its definition in Eq. (4). This probabilistic congestion will be used during the next time-interval $(t, t+\delta]$. It is worth mentioning that although the N-point FFT requires $O(N \log N)$ arithmetic operations, it can be performed in logarithmic time in parallel, and the sequential computation of the FFT is certainly doable for reasonably large N without overloading current routers. Moreover, the estimation of m, a, and H is performed every δ time units. In practice, one can set δ equal to a couple of minutes (say 10mn), for $\Delta \simeq$ 1h and $t_u \simeq$ 3.5s. This yields the computation of FFT's on $\Delta/t_u = 1024$ points, which is reasonably small. Still the estimation of the long-scale behavior of the traffic will remain good enough, as shown in the next sections.

3 Application to Multicast

Several protocols have been proposed to support group communications. Among them, YAM [3], QoSMIC [6] and MORF [24] take into account QoS for the construction of the multicast tree. A common feature of these three protocols is to offer multiple routes between a new member and the current tree. This allows them to select the "best" route with regards to some QoS requirements. MORF and YAM use static parameters (e.g., link capacity, link delay, or reliability), but QoSMIC uses dynamic parameters (e.g., available bandwidth, current delay). Hence, we focussed our attention to this latter protocol, and show that using probabilistic congestion offers better performances than using the available bandwidth or the current delay, as far as packet-loss is concerned. QoSMIC can create shared trees or source-based trees. In this paper, we assume a behavior in the shared tree mode. A new router, aiming to join the group, searches for a router in the tree where to connect. There are two search procedures in QoSMIC: *local search* and *multicast tree search*. The local search is similar to the one proposed in YAM: the new router performs exploration of its neighborhood at successive distance $1, 2, \ldots,$ until the Time To Live field of the IP header is exhausted. The multicast tree search is performed via a call to a Manager router, which starts exploration of the tree to find appropriate candidates in the tree for the new router to connect with. How much the simultaneous use of these two types of search allows to reduce the complexity of joining is discussed in [6]. In our experiments, we consider all possible paths constructed by the unicast routing from the new router to all routers in the current tree. Therefore, the search performed from a router x aiming to join the group results in a set $y_1, \ldots, y_k$ of candidate routers in the tree. Then x selects the appropriate candidate by comparing the characteristics of the routes between itself and every y_i. It is underlined in [6] that, although the routes are restricted by static information in the routing information base, the new router selects among these routes using dynamic routing information. The use of the probabilistic congestion ideally fits with this setting.

In order to compare the QoSMIC selection using available bandwidth vs. the selection using probabilistic congestion, we also use the Minimal λ-Tree multicast protocol (MλT) briefly described as follows [1]. MλT is built upon a multiple-path routing protocol which maintains a table D_v at each router v, such that $D_v[i,x]$ is the length of the shortest path connecting node v to node x when leaving v through the i-th interface. MλT explores, for every node x of the group, all paths from the source s to x that are of length at most $\varrho \cdot d(s,x) + r$ where ϱ and r are constants fixed *a priori*, and $d(s,x)$ is the distance from s to x (measured in #hops). Then MλT selects, for every router x, the path from s to x whose QoS is the best. Finally, it constructs a multicast tree by merging all selected paths. Therefore, MλT constructs a tree T rooted at the source s such that, for every node x of the group, the route from s to x in T has the best QoS among all routes from s to x of length at most $\varrho \cdot d(s,x) + r$. Obviously, there is tradeoff between the control overhead, and the efficiency of MλT. The construction of the tree by MλT is indeed costly if ϱ or r is large. On the other hand, the efficiency (in terms of packet-loss) of the tree is expected to be good if many routes are considered for the selection, i.e., if ϱ or r is large. In some sense, the performances of MλT are those of an ideal multicast protocol that would be allowed to explore a large portion of the network to select the routes.

In total, we have considered four tree-construction protocols: Reverse Shortest Path (RSP), Greedy, QoSMIC, and MλT. Recall that RSP sets up the tree simply as the union of the (unicast driven) paths from the source to every destination. This technique is used in CBT and PIM-SM. In Greedy, a new node aiming to join the group connects to the closest node in the current tree. This technique is used in YAM. We compare the efficiency of using available bandwidth vs. probabilistic congestion for the selection of the routes in QoSMIC. MλT was implemented using probabilistic congestion only. On one hand RSP and Greedy allows to compare the two versions of QoSMIC with (sort of) basic approaches for tree-construction. On the other hand, MλT allows to compare the same two versions of QoSMIC with a (sort of) ideal tree-construction protocol.

3.1 Experimental Results

We have performed simulations based on realistic traffic patterns obtained from the traffic samples observed on Abilene's backbone. Actually, since Abilene is underutilized, we assumed links with 15% capacities of those of the real Abilene network. This rescaling yields a load of up to 98% for the most occupied links.

We have divided the time in 389 non-overlapping windows of 4h10 each. Since the network is observed at the period of 5mn, every window contains 50 samples. For each window, we computed the three corresponding characteristic parameters m, a, and H. Therefore, we based our simulations on 389 sets $\{(m_e, a_e, H_e), e \in E\}$, where E is the set of the 28 links of Abilene's backbone, and each triple is computed from a real IP traffic. Therefore, we obtained more than 10,000 experimental triples in total. Each of them is rescaled for a link of unit capacity (i.e., m and a are rescaled to m/C and a/C where C is the link capacity). We have performed experiments on the UUNET backbone topology. UUNET is a world-wide private network with 129 nodes. We did not had access to the buffer size b of the links of the network, therefore we set $b = 4800$ bytes for all links, which is a bit more than the maximum size of an FDDI packet. We used UUNET's backbone with all its original link capacities, which vary from few tens Mb/s for some links, to roughly 10 Gb/s for others.

Each experiment is performed according to the following protocol. For each link e of the network, we choose one triple (m, a, H) among the 10,000 experimental triples, and re-scale m and a to fit with the capacity C of link e (i.e., m and a are rescaled to $m\ C$ and $a\ C$). Then the probabilistic congestion of each link is computed according to Eq. 4. The probabilistic congestion of a path is the sum of the probabilistic congestion of its links. Similarly, the available bandwidth $C - w$ of each link is computed, where C is the physical capacity of the link, and w is the used capacity. The used capacity is computed as the average number of bytes crossing the link during some predefined interval of time. The available bandwidth of a path is the minimum of the available bandwidth of its links.

For a considered multicast group $\{x_1, \dots, x_k\}$, and a source s, we simulate the execution of RSP, Greedy, and the two versions of QoSMIC, assuming members arrive successively in the order $x_1, \dots, x_k$. Hence, we get four trees T_{RSP}, T_{Greedy}, $T_{\text{QoSMIC}}^{\text{prob cong}}$, and $T_{\text{QoSMIC}}^{\text{avail bdw}}$. For the ideal protocol MλT, a tree $T_{\text{M}\lambda\text{T}}$ spanning all group members, with minimum probabilistic congestion, is computed globally (although MλT can also be implemented in a distributed greedy manner). Actually, to make MλT more realistic, we didn't explore all paths, but only those of bounded length. More precisely, we have run MλT with $\varrho = 1$ and $r = 2$.

Then, for each link, we have performed simulations by injection of a self-similar traffic A_t through the link during 512s (roughly 8mn30), where A_t is described in Eq. (1), and where m, a, and H are the parameters chosen for that link. That is, for every link with buffer size b, we run the iterative process

$$Y_{k\tau} = (m - C)\tau + \sqrt{am}\,(Z_{k\tau} - Z_{(k-1)\tau})$$

and

$$X_{k\tau} = \begin{cases} 0 & \text{if } X_{(k-1)\tau} + Y_{k\tau} \leq 0; \\ b & \text{if } X_{(k-1)\tau} + Y_{k\tau} \geq b; \\ X_{(k-1)\tau} + Y_{k\tau} & \text{otherwise.} \end{cases}$$

The resolution of the simulation is $\tau = 1/512$s. The Z_t's were generated using the simulator of Norros [14]. For each link e, we computed the ratio between the

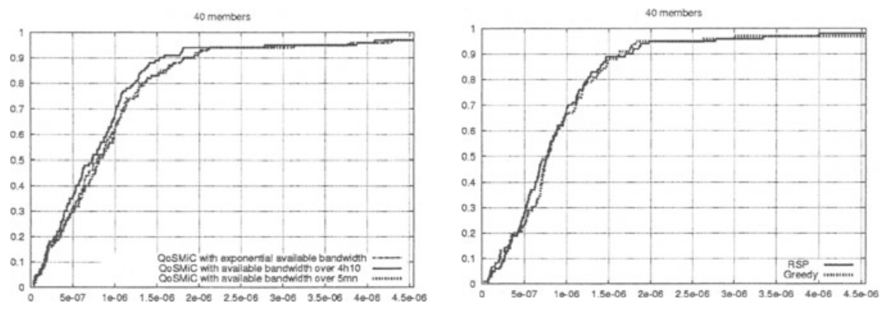


Fig. 1. Cumulative distributions of the packet-loss probability

total number of bytes lost by link e during the 8mn30 of the simulation, and the total number of bytes sent over link e during the same period of time. This ratio is therefore the probability p_e for an application using link e to lose packets. For each tree-construction protocol, we then computed the probability for a multicast application using the resulting tree $T = (V, E)$ to not lose packets, estimated by $\mathbf{Pr}(T) = \Pi_{e \in E}(1 - p_e)$. For the same multicast group, we repeated the experiment 100 times (i.e., with 100 different triples (m, a, H) for each link). We performed experiments for different multicast groups, of different sizes. Results are displayed using the cumulative distributions F of the packet-loss probability. That is, the vertical axis is a percentage, and the horizontal axis is a probability. Given a probability p, $F(p)$ is the percentage of experiments which returned a tree for which the packet-loss probability was $\leq p$.

Fig. 1 (left) displays the performances of three versions of QoSMIC, all using predictions based on the available bandwidth, for a group of size 40. All versions average the used capacity w of each link. The first version uses exponential averaging [8]. The last two versions differ on the interval of time during which the used capacity w of each link was averaged. We considered either average over the last 4h10, i.e., $w = m$ where m is the mean input rate of the link, or average over 5mn, i.e., $w = B$ where B is the average throughput of the link during the last 5mn. Fig. 1 (left) shows that, compared to the two others type of averaging, predictions based on an average over a long period of time (here 4h10) offer slightly better results. For instance, using an average over 4h10, 35% of our experiments return a tree for which the packet-loss probability is $\leq 5 \cdot 10^{-7}$. Using an average over 5mn, only 30% of our experiments return a tree for which the packet-loss probability is $\leq 5 \cdot 10^{-7}$. This latter packet-loss probability is roughly the same for exponential averaging. Indeed, exponential averaging is similar to short time averaging because the exponential decays very quickly. All our experiments have confirmed this behavior. Therefore, for comparisons with tree-constructions based on other methods, we considered only the version of QoSMIC averaging the available bandwidth over 4h10.

Fig. 1 (right) shows the performances of RSP and Greedy for a group of size 40. As far as packet-loss is concerned, the two strategies perform roughly

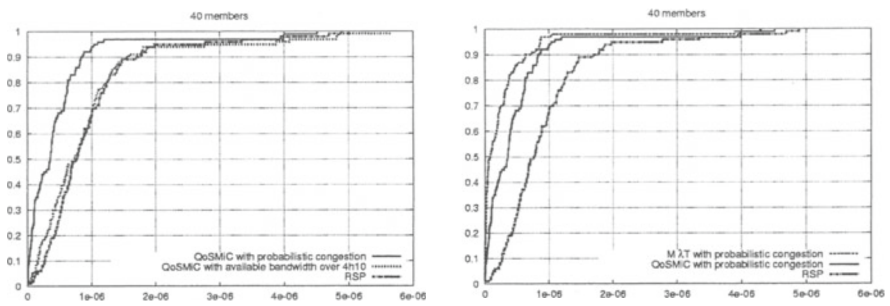
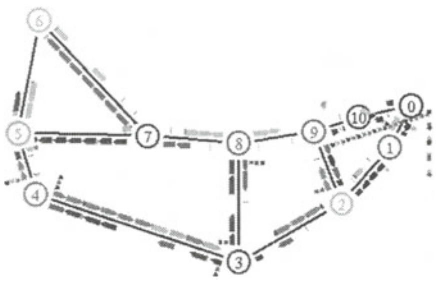


Fig. 2. Cumulative distributions of the packet-loss probability (continued)

the same, with perhaps a little advantage to RSP. All our other experiments show the same relative behavior of the two protocols. In the following, we will hence consider RSP only, as a representative of popular protocols such as PIM-SM or CBT.

Fig. 2 (left) presents a first set of significant results. It displays the performances of RSP, QoSMIC with available bandwidth , and QoSMIC with probabilistic congestion, for a group of size 40. QoSMIC with available bandwidth performs better than RSP in general, and hence better than the Greedy protocol. That is, although the traffic is subject to bursts at all time scales, which makes average bandwidth a poor predictor for the future behavior of the system, QoSMIC takes in fact some benefits from the selection of the routes according to the available bandwidth. More interestingly, QoSMIC with probabilistic congestion performs significantly better than QoSMIC with available bandwidth, bringing supports to the main claim of this paper. For instance, almost 70% of our experiments on QoSMIC with probabilistic congestion return a tree for which the packet-loss probability is $\leq 5 \cdot 10^{-7}$ for the 40-node group. In comparison, only 35% of our experiments on QoSMIC with available bandwidth return a tree with the same QoS. In 90% of our experiments, QoSMIC with probabilistic congestion has packet-loss probability $\leq 8.6 \cdot 10^{-5}$, and QoSMIC with available bandwidth has packet-loss probability $\leq 16 \cdot 10^{-5}$. Actually, the improvement of using QoSMIC with probabilistic congestion compared to QoSMIC with available bandwidth is by far bigger than the improvement of using QoSMIC with available bandwidth compared to RSP.

Finally, Fig. 2 (right) displays comparisons between QoSMIC using probabilistic congestion with, on one hand, the standard protocol RSP, and on the other hand, the ideal protocol MλT. Of course, MλT outperforms the two other protocols. The gain obtained from using MλT compared to QoSMIC with probabilistic congestion is roughly the same as the gain of using the latter compared to RSP. In particular, in terms of average packet-loss, QoSMIC with probabilistic congestion performs almost 2 times better than RSP.



# flows	% l. P_1	# t.l.P_1	% l. P_2	# t.l.P_2
1	2.87	12930	0	4014
2	2.94	14207	0	3462
3	2.93	13819	0	3590
5	3.06	13371	0	3284
10	2.92	11925	0	2625
20	4.83	14218	0.01	3866
30	6.27	15068	0.05	4668
40	6.96	14968	0.14	3927
50	7.98	14968	0.27	3920
60	9.37	15829	0.64	5145

Fig. 3.

4 Application to Unicast

Probabilistic congestion could also be used for unicast routing. Consider for instance a network supporting two classes of services: a priority class (i.e., voice transmission), and a non-priority class. Assume moreover that the amount of priority class traffic is small in front of the non-priority class traffic. In this context, one could use probabilistic congestion to construct and maintain the unicast tables of the priority class. A straightforward approach consists of setting up the weight of each link as its probabilistic congestion, and using a standard Dijkstra-like unicast routing protocols. Then the priority class traffic would benefit from a significant decrease in packet-loss. We illustrate this approach by using the following scenario[1]. We used again the backbone of the Abilene network (c.f. Fig. 3 left), in which we injected exponential on-off TCP traffic between (almost) every pair of nodes. More precisely, we took the pairs {3,4}, {3,5}, {3,6}, {4,5}, {4,6}, {5,6}, {9,2}, {9,3}, {9,4}, {10,2}, {10,3}, {10,4}, {10,9}, {6,9}, {5,7}, {3,8}, {0,2}, and {0,10}, and connect a source and a receptor at each node or router of each pair. We used the NS2 [23] simulator to inject and route the traffic. In this simulation we observed a self-similar traffic traversing every link, because of TCP. Figure 3 (left) shows a screen capture of NAM [23], the network animator of NS2. In this figure, it is possible to observe the traffic on all links, and the buffers occupation. We selected two nodes, $x = 2$ and $y = 10$ arbitrarily. The path set up by the routing tables of Abilene between x and y is denoted by $P_1 = \{2, 9, 10\}$, through one intermediate node. There is an alternative longer path, through two intermediate nodes, denoted by $P_2 = \{2, 1, 0, 10\}$. We have simulated 60 seconds of traffic, and we have computed the probabilistic congestion (using the natural logarithm) of P_1 and P_2 ($c_1 = 48.8 \cdot 10^{-2}$ and $c_2 = 26.5 \cdot 10^{-2}$) as well as the available bandwidth of these two paths ($b_1 = 4.07$Mb/s and $b_2 = 0.579$Mb/s). Clearly, the choice between P_1 and P_2 depends on whether one uses probabilistic congestion or available bandwidth. We show that probabilistic congestion is the best criterion. For that purpose,

[1] The complete description of the our scenario is available on request to the authors.

we again performed simulations during 60 seconds of TCP traffic, with several CBR voice flows from node x to node y, at 23Kb/s. The table in Figure 3 (right) displays the number of voice flows from x to y, the percentage of voice-packet lost on path P_1, the total number of packets lost on P_1 (during the 60 s), the percentage of voice-packet lost on path P_2, and the total number of packets lost on P_2 (during the 60 s). Using path P_2, the total number of packets losses is approximately 1/3 of the number of packet losses when using path P_1, independently of the number of flows. Moreover, looking at the voice flows packets only, we observed that the percentage of packet losses in these flows is significantly smaller using path P_2 than using path P_1. For instance, if there are 60 aggregated voice flows, then the percentage of packet losses in these flows is 9.37% for path P_1, whereas it is only 0.64% for path P_2. Since voice is very sensitive to packet losses, the benefit in term of QoS of using a prediction criteria such as probabilistic congestion would be significant in comparison with other criteria such as available bandwidth.

5 Conclusion and Future Works

A challenging problem consists of checking whether the probabilistic congestion could be efficiently used for the whole unicast traffic. We have shown that probabilistic congestion captures simultaneously qualitative and quantitative traffic characteristics, and, from that perspective, it offers better properties than standard QoS parameters such as delays or bandwidth. However, it is unclear *how much* the global traffic would improve in terms of packet-loss. Indeed, the behavior of the network traffic depends heavily on TCP which dynamically adapts to the congestion of the routes. We leave this question open for future investigations.

References

[1] Alvarez-Hamelin, J.I., Fraigniaud, P.: MλT: A Multicast Protocol with QoS Support. In: The 12th International Conference on Computer Communications and Networks, ICCCN 2003, IEEE (2003) 264–269

[2] Beran, J., Sherman, R., Taqqu, M., Willinger, W.: Long-range dependence in Variable-Bit-Rate video traffic. IEEE Transaction on Communications **43** (1995) 1566–1579

[3] Carlberg, K., Crowcroft, J.: Building Shared Trees using a one-to-many joining mechanism. In: ACM SIGCOMM Computer Communication Review. (1997) 5–11

[4] Crovella, M., Taqqu, M., Bestavros, A.: Heavy-tailed probability distributions in the World Wide Web. chapter 1, pages 3-26, Chapman & Hall, New York (1998)

[5] Crovella, M.E., Bestavros, A.: Self-similarity in World Wide Web traffic: evidence and possible causes. IEEE/ACM Transactions on Networking **5** (1997) 835–846

[6] Faloutsos, M., Banerjea, A., Pankaj, R.: QoSMIC: Quality of Service sensitive Multicast Internet protoCol. In: SIGCOMM. (1998) Vancouver BC.

[7] Garrett, M., Willinger, W.: Analysis, modeling and generation of self-similar vbr video traffic. In: ACM SIGCOMM. (1994) 269–280

[8] Jain, R.: Congestion control and traffic management in ATM networks: Recent advances and A survey. Computer Networks and ISDN Systems **28** (1996) 1723–1738

[9] Leland, W., Taqqu, M., Willinger, W., Wilson, D.: On the self-similar nature of the Ethernet traffic. IEEE/ACM Transaction on Networking **2** (1994) 1–15

[10] Mandelbrot, B., Ness, J.V.: Fractional Brownian motion, fractional noises and applications. SIAM Review **10** (1968) 422–437

[11] Norros, I.: A storage model with self-similar input. Queueing Systems **16** (1994) 387–396

[12] Norros, I.: On the Use of Fractional Brownian Motion in the Theory of Connectionless Networks. IEEE Journal of Selected Areas in Communications **13** (1995) 953–962

[13] Norros, I.: Busy periods of fractional Brownian storage: a large deviations approach. Adv. Perf. Anal. **2** (1999) 1–19

[14] Norros, I., Mannersalo, P., Wang, J.: Simulation of fractional Brownian motion with conditionalized random midpoint displacement. Adv. Perf. Anal. **2** (1999) 77–101

[15] Park, K., Kim, G., Crovella, M.: On the relationship between file sizes, transport protocols, and self-similar network traffic. In: 4th Int. Conference on Network Protocols (ICNP '96). (1996) 171–180

[16] Park, K., Kim, G., Crovella, M.: On the Effect and Control of Self-Similar Network Performance. In: SPIE Conf. on Performance and Control of Network Systems. (1997)

[17] Paxson, V., Floyd, S.: Wide-Area Traffic: The Failure of Poisson Modeling. IEEE/ACM Transaction on Networking **3** (1995) 226–244

[18] Richard, O., Cappello, F.: Sur la nature auto-similaire de l'activité de stations de travail et de serveurs HTTP. Technique et Science informatiques **17** (1998) 635–658

[19] Taqqu, M.S., Willinger, W., Sherman, R.: Proof of a fundamental result in self-similar traffic modeling. ACMCCR: Computer Communication Review **27** (1997) 5–23

[20] Veres, A., Boda, M.: The chaotic nature of TCP congestion control. In: IEEE INFOCOM. (2000) 1715–1723

[21] Veres, A., Kenesi, Z., Molnár, S., Vattay, G.: On the propagation of Long-Range Dependence in the Internet. In: ACM SIGCOMM. (2000) 243–254

[22] Willinger, W., Taqqu, M.S., Sherman, R., Wilson, D.V.: Self-similarity through high-variability: statistical analysis of Ethernet LAN traffic at the source level. IEEE/ACM Transactions on Networking **5** (1997) 71–86

[23] : `http://www.isi.edu/nsnam/ns/`.

[24] Zappala, D., Estrin, D., Shenker, S.: Alternate path routing and pinning for inter-domain multicast routing. Technical Report USC-CS-TR-97-655, University of Southern California (1997)