Parallel TCP Sockets: Simple Model, Throughput and Validation

Eitan Altman, Dhiman Barman
INRIA
{altman,dbarman}@sophia.inria.fr

Bruno Tuffin
IRISA/INRIA
bruno.tuffin@irisa.fr

Milan Vojnović
Microsoft Research Cambridge
milanv@microsoft.com

***Abstract*— We found a formula for aggregate throughput of arbitrarily given number of competing additive-increase, multiplicative-decrease connections (TCP congestion avoidance mode) for a bottleneck, under assumption that loss events over connections are non synchronized. The formula captures throughput-deficiency due to the additive-increase and multiplicative-decrease adaptation. The formula suggests that already a few connections are sufficient to almost entirely eliminate this throughput deficiency. The result reveals the aggregate throughput insensitivity on the way losses are assigned over competing connections over time, for any given number of competing connections. The result is validated by simulations and Internet measurements. The latter validates the model in cases when analysis assumptions are met, but also encounters cases of the throughput deficiency due to synchronization of loss events and the receiver window constraint. The results would inform on the throughput efficiency of parallel TCP transfers, an approach used widely for bulk data transfer.**

I. Introduction

Parallel TCP sockets is a generic "hack" to improve throughput attained by TCP for bulk data transfers by opening several TCP connections and striping the data file over them. There are two factors that cause TCP throughput deficiency (F1) TCP window synchronization and (F2) window adaptation in TCP congestion avoidance. The former is well known and fixes have been proposed a while ago (e.g. RED [12]). The throughput deficiency due to (F2) is intrinsic to congestion window control in TCP congestion avoidance and is also well known. It is perhaps best illustrated by admitting a simple model of a single TCP connection over a bottleneck whose steady-state window dynamics is deterministic and periodic described as follows. In a period, the window W increases linearly in time with rate 1 packet per round-trip time and is reduced to one half of its current level when it encounters congestion. It is readily computed that such idealized TCP connection attains 75% link utilization. The simple model is the underlying model of a particular TCP square-root loss-throughput formula, as found in [11], [18]. The concept of parallel TCP sockets is widely used by applications such as for example GridFTP*. In practice, it is often unclear how many sockets one needs to open in order to achieve satisfactory throughput since opening too many connections may be undesirable for various reasons. Relation of TCP throughput of a single connection to loss event rate and mean round-trip time is to date fairly understood (e.g. [4], [18], [19]), but the results do not readily extend to parallel TCP connections. These formulae only leave us with two other unknowns. The problem is that it is not well understood how these two unknown parameters depend on the number of TCP connections competing in a bottleneck. The approach leveraging on known TCP loss-throughput formulae was pursued by Hacker et al [13], which provides very informative measurement results, but no throughput formula for parallel TCP connections.

It is important to distinguish the factors (F1) and (F2) as they are different throughput-deficiency causes. Our analysis is concerned with factor (F2). It is a good news as it suggests that already a few connections fixes the problem, in operational regimes when there is no synchronization (i.e (F1) not true) or there is some but is weak.

In this paper, the send rate of each connection is assumed to increase linearly in time in absence of congestion indication to this connection, and otherwise decrease to a fraction β of the send rate just before a congestion indication (multiplicative decrease). We adopt a model of bottleneck originally introduced by Baccelli and Hong [9] where link congestion events occur whenever the aggregate rate of connections hits the link capacity. Strictly speaking, the model assumes zero-buffer; see [9] for discussion how this can be relaxed to accommodate a finite buffer. The zero-buffer assumption is made for tractability and would be justified in scenarios when propagation delay dominates the queueing delay. We provide validation of this in Section V. We assume

(ONE) At each congestion event, a single connection is signaled to reduce its send rate by one multiplicative decrease.

Our main result is a formula for aggregate throughput $\bar{x}(N)$ of N connections over a link of fixed capacity c,

$$\bar{x}(N) = c\left(1 - \frac{1}{1+\frac{1+\beta}{1-\beta}N}\right) \tag{1}$$

where $0 < \beta < 1$; for TCP connections $\beta = 1/2$. The result is *exact* under a mild "stability" condition (see Section II, Theorem 1). Note that for the special case of one connection, $N = 1$, the result reads $\bar{x}(N) = c(1+\beta)/2$. Hence, for TCP-like setting $\bar{x}(N) = (3/4)c$, which recovers the 75% utilization pointed out earlier. Note that nothing is said on

*www.globus.org/datagrid/gridftp.html

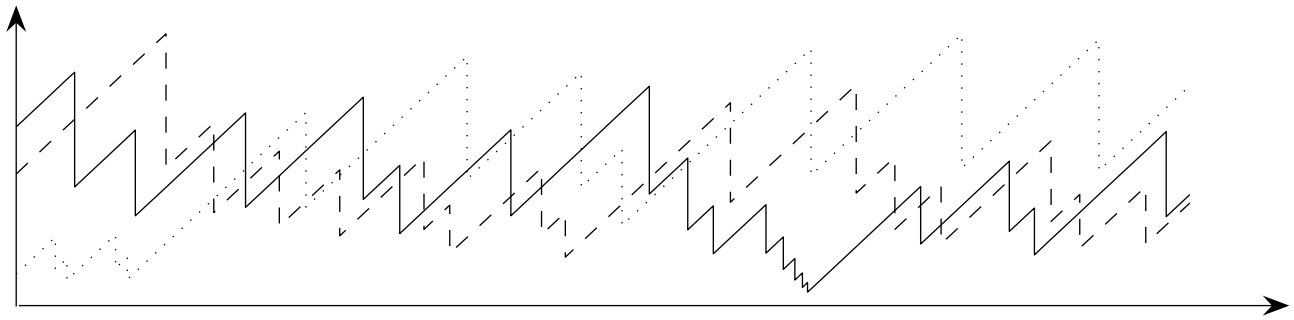

Fig. 1. Sample paths of three AIMD connections competing for a link.

which connection is chosen at link congestion events. Figure 1 illustrates several system assumptions that we made so far: linear increase and multiplicative decrease, link congestion times, and assumption (ONE). The assumption (ONE) was made in some other works, e.g. [10]. It would be more general to assume that at each link congestion event a subset of connection is chosen to undergo a multiplicative decrease. On the other extreme, we may assume that at each link congestion event, ALL connections undergo multiplicative decrease. This corresponds to TCP window synchronization. Baccelli and Hong [9], in turn, assume each connection is chosen with some fixed probability, so that at each congestion event a fraction of connections is signaled a congestion indication. The reader should bear in mind that (ONE) and similar assumptions are mathematical abstractions. In practice, though the system dynamics is blurred by various factors such as packetization and feedback delays and there will always be some positive time between any two connections undergoing multiplicative decrease, so it may not be justified to aim at validating (ONE) in some strict sense. The definition of link congestion times would in practice accommodate TCP connections over a network interface with finite buffer and FIFO queueing discipline (DropTail), with buffer length sufficiently smaller than bandwidth-delay product. In practice, most network interfaces are DropTail. The linear increase of send rate would approximate well TCP [14] in regime where round-trip time is dominated by propagation delay and queueing delay can be neglected [9]. It is well known that with substantial queueing delay, TCP window growth over time is sub-linear for large windows [7]. The assumption that send rate increase is linear over time, may be satisfied by transport control protocols other than standard TCP, e.g. [16] shows this to be true for Scalable TCP [15].

In our analysis we also assume:

(R) TCP connections are not constrained by their receiver windows.

With TCP receiver window in effect, the throughput would increase roughly linearly with the number of parallel TCP sockets for a range from 1 to some number of TCP connections. TCP receiver window constraint is not further considered in this paper. In analysis we assume (R) holds and in experimental validations we set TCP receiver window larger than the bandwidth-delay-product whenever possible.

We argue that it is of interest to understand throughput efficiency of parallel TCP sockets for a small number of parallel sockets. Opening too many sockets may not be of practical interest as this imposes processing overhead on end-host operating systems and also leaves smaller bandwidth to other flows that may sporadically share the bottleneck during a parallel-socket data transfer. The result (1) suggests that the throughput gain of N parallel TCP sockets monotonically increases with N and this increase is such that already 3 sockets is sufficient to attain 90% link utilization. Setting $N = 6$ would yield almost 95%. This suggests that a few connections is already enough to compensate for the throughput deficiency of TCP. This is a good news as it would provide incentive to rational throughput-greedy users not to use an overly large number of parallel sockets per transfer, as this would yield only a marginal throughput gain and as argued earlier opening too many sockets may not be desirable.

There has been a recent surge of interest on TCP and small buffer sizes [8]. While results in this paper do have some connection to this line of the work, as our analysis would be accurate for appropriately chosen small buffer size, there are notable differences. The results in [8] suggest that buffers in routers can be set to smaller values than bandwidth-delay product (BDP) for a large number of TCP connections and still have full link utilization. More specifically, it is suggested that for a link shared by N connections and link capacity scaled linearly with N, it is sufficient to have buffer length of the order BDP/$\sqrt{N}$, asymptotically for large N. The throughput formula (1) is for any finite N and as argued earlier it is our particular interest to address the case when the number of connections N is not too large. In our context, it may well happen that bottleneck link is in access network, not a backbone router, and it may simply turn out to be small relative to propagation round-trip time of a network path between end-points of a parallel transfer. This would hold in practice for many end-to-end paths. Consider a bottleneck link with buffer size L packets, packets of length MTU B, and link capacity c MB/s. Then, we may regard the buffer to be small for network paths of propagation round-trip time RTT such that RTT $\gg$ L MTU/c. For example, consider in practice typical values $c = 100$ Mb/s, MTU=1500 B, and L=100 packets. Then, the

condition is RTT $\gg$ 12 ms, which would be true for many wide-area data transfers.

The result (1) tells us more. It reveals that throughput is insensitive to a choice of loss policy that decides which connection is signaled per congestion event as long as the loss policy verifies (ONE) and stability conditions in Theorem 1. Note that the set of loss policies that verify (ONE) is large and includes many natural candidates for loss policies. This insensitivity result may appear counter-intuitive to some as one's intuition may be that loss policies that favor picking connections with larger rate at congestion times would yield smaller aggregate throughput. The result says that this is not true. The throughput insensitivity in the present form was suggested by a result in [5], which is under more restrictive assumptions of only 2 connections and special loss policies that render send rates a Markov process. One may wonder whether steady-state probability distribution, and in particular, higher-order moments, of aggregate send rate are insensitive to a choice of loss policy. We show that this is not true in Section III. The result in Section III is not new as it was already found in [5], however, Section III provides corrected versions with complete proofs. The results in Section III and Section IV are particular choices of loss policy: (i) rate-proportional (= a connection is chosen with probability proportional to its send rate just before the congestion event), (ii) largest-rate (= a connection with largest rate just before the congestion event is chosen), (iii) rate-independent (= a connection is chosen with fixed probability), and (iv) beatdown (= a connection is chosen as long as permitted by link capacity). The rate-proportional loss policy may be deemed a natural candidate and may be seen as randomized version of the largest-rate policy. We consider the rate-independent policy in view of past work [9]. The fancy beatdown loss policy is considered only to demonstrate that throughput insensitivity found in this paper holds for a versatile set of loss policies. Note that in the Internet either of these loss policies or some other policies may emerge as a result of the dynamics of TCP congestion window control and particular queue discipline such as for example DropTail. More radical approach would be to enforce assumption (ONE) and particular loss policy by an intelligent queue discipline. Analysis of benefits and performance of such intelligent queue disciplines is beyond the scope of this paper.

A. Organization of the paper

Section II contains our main result. In that section, we provide a sufficient set of assumptions and notations and then state the main result, along with discussion of its implications (see [6] for proofs). Section III shows that steady-state distribution of the aggregate send rate of connections is not insensitive to loss policy of the bottleneck link. Several auxiliary results are provided in Section IV. Section V contains experimental validation, both simulations and internet measurements. The last section concludes the paper.

II. Main Result

A. Model and result

We consider N connections that compete for a link of capacity c by adapting their send rates as follows. Let $X_i(t)$ be the send rate of a connection i at time t. A time t' is said to be a link congestion time whenever it holds $\sum_{i=1}^N X_i(t') = c$, i.e. the aggregate send rate hits the link capacity. Let T_n be the n-th link congestion time enumerated such that $T_0 \le 0 < T_1$, for some time instant labeled with 0. A connection i increases its send rate linearly in time with rate $\eta_i > 0$ and it is reduced at some link congestion times to a fraction $0 < \beta_i < 1$ of the current send rate. We call such an adaptive connection, additive-increase and multiplicative-decrease (AIMD) connection with additive-increase parameter η_i and multiplicative-decrease parameter β_i.

The foregoing verbal definitions can be formally rephrased as, for $i = 1, 2, \ldots, N$,

$$X_i(t) = X_i(T_n) + \eta_i(t - T_n), \;\; T_n \le t < T_{n+1} \tag{2}$$

and

$$X_i(T_{n+1}) = (1 - (1-\beta_i)z_i(n))X_i(T_n) + \eta_i S_n \tag{3}$$

with

$$S_n = \frac{1}{\overline{\eta}} \left(c - \sum_{j=1}^N (1-\beta_j) z_j(n) X_j(T_n) \right), \tag{4}$$

where $S_n := T_{n+1} - T_n$ is obtained from $\sum_{i=1}^N X_i(T_{n+1}) = c$, $\overline{\eta} := \sum_{j=1}^N \eta_j$, and $z_j(n) = 1$ if the connection j is signaled a congestion indication at the n-th link congestion event, else, $z_j(n) = 0$. Under (ONE) $\sum_{j=1}^N z_j(n) = 1$. We are now ready to state our main result.

Theorem 1: Consider the prevailing system of N AIMD connections with identical AIMD parameters, competing for a link of capacity c. Assume that ONE holds, i.e. at each link congestion event, exactly one connection is signaled a congestion indication. Assume in addition that the following stability conditions hold: the system has a unique time-stationary distribution and times between successive link congestion events, sampled at congestion events, have a strictly positive mean. Then, the throughput X^{ONE} has the time-stationary expected value

$$\frac{\mathbb{E}(X^{\mathsf{ONE}}(0))}{c} = 1 - \frac{1}{1 + \frac{1+\beta_1}{1-\beta_1} N}. \tag{5}$$

Proof is given in [6].

B. Discussion of the result and its implications

We discuss several implications of Theorem 1.

1) Throughput of parallel TCP sockets: The throughput formula (5) suggests explicit dependence of the throughput on the number of parallel TCP sockets N. Indeed, the throughput monotonically increases from $(1+\beta_1)/2c$ for $N = 1$ to 1 as N tends to infinity. The former case of one connection is precisely the known result $(3/4)c$ thats hold for TCP-like

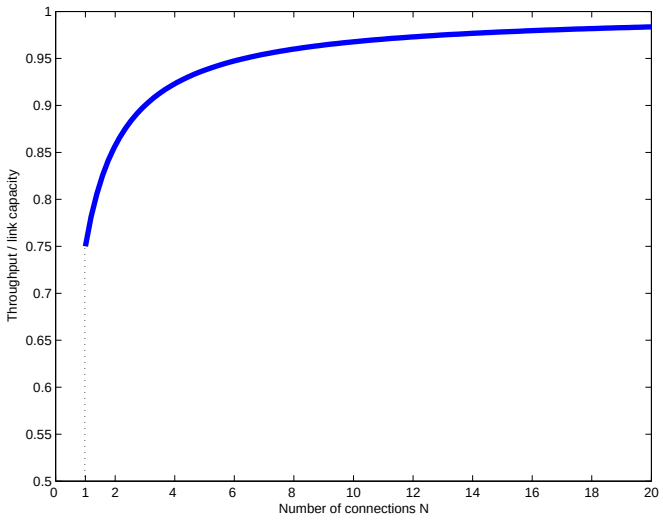

Fig. 2. Throughput of N identical AIMD connections competing for a link of unit capacity with TCP-like setting ($\beta_1 = 1/2$).

setting $\beta_1 = 1/2$. The dependence on the number of sockets given by (5) is showed in Figure 2 for $\beta_1 = 1/2$. The result suggests that opening as few as 3 connections would already yield 90% link utilization and 6 sockets, almost 95% link utilization. Hence, opening a number of sockets beyond some relatively small value would yield only a marginal throughput gain. Equation (5) gives explicit dependence of the throughput on the number of sockets N, and thus provides a guideline to choose the magic configuration parameter of the number of sockets for a data transfer. Choosing this configuration parameter in practice was a puzzle as throughput dependence on the number of sockets was largely not understood. The result also suggests incentive to throughput-greedy users not to open too many sockets, as opening a few is enough.

2) Throughput is insensitive: No matter which connection is signaled a congestion indication per link congestion event, the mean throughput remains the same, as long as the mild stability conditions of the theorem hold. This throughput insensitivity finding may appear counter-intuitive as one may expect that loss policies that favor signaling congestion indication to large-rate connections would have detrimental effect on the mean aggregate throughput. The result reveals this is not true. We show in Section III that the aggregate send rate higher-order statistics is not insensitive on the loss policy.

The throughput insensitivity in the general form found in this paper was motivated by original finding in [5]. Note that we generalize the finding of [5] in the following directions: (i) to any finite number of connections N ([5] is for $N = 2$) and (ii) a larger set of loss policies. We discuss (ii) in some more detail now. The loss policies in [5] are restricted to those that base decision which connection to signal a congestion indication only on the send rates just before a link congestion event. As an aside, note that this renders the send rates a Markov process. The result of Theorem 1 tells us that the throughput insensitivity holds for a much broader set of loss policies that can depend on any past system state as long as the mild stability conditions of the theorem hold.

Note that the throughput insensitivity allows to bias signaling congestion indication to some connections at link congestion times. The throughputs achieved by these special connections would indeed deteriorate by this bias, however, the aggregate throughput over all connections remains the same. This is illustrated by the following example.

Example 1: Consider 4 AIMD connections competing for a link of unit capacity. The connections have identical AIMD parameters, set as $\eta_i = 1$ and $\beta_i = 1/2$. The loss policy is rate-independent with the probability that a connection i is selected equal to $\sigma/2$, for $i = 1,2$, and $(1-\sigma)/2$, for $i = 3,4$, for some $0 < \sigma < 1$. The parameter σ is the bias parameter that for $\sigma < 1/2$ favors picking either connection 1 or 2, $\sigma = 1/2$ favors no connection, and $\sigma > 1/2$ favors picking connections 3 or 4. Hence, as σ ranges from 0 to 1, the connections 1 and 2 (resp. 3 and 4) will receive equal throughput that will decrease (resp. increase) with σ. In the prevailing example, the system does have a unique time-stationary distribution, hence the hypotheses of Theorem 1 are satisfied. Theorem 1 implies that the mean aggregate throughput is insensitive to the preferential treatment of connections. The simulation results in Figures 3 and 4 indeed validate the claim.

3) Comparison with the synchronized case: Consider now the loss policy we call ALL, which signals all connections a congestion indication at each link congestion event. Note that this loss policy satisfies (ONE) only if $N = 1$. The loss policy ALL models the special regime of synchronized TCP connections. A natural question is to compare the throughputs under loss policy ALL and loss policies that verify (ONE). The throughput under loss policy ALL is readily obtained to be[†]

$$\frac{\mathbb{E}(X^{\mathsf{ALL}}(0))}{c} = \frac{1+\beta_1}{2}. \tag{6}$$

It is easy to check that

$$\mathbb{E}(X^{\mathsf{ONE}}(0)) \geq \mathbb{E}(X^{\mathsf{ALL}}(0))$$

with equality only if $N = 1$. Note that, indeed, the throughput achieved under the loss policy ALL for $N > 1$ is exactly that achieved by a single connection under either ALL or ONE. Thus, the ratio of the throughputs achieved under a loss policy that satisfies (ONE) and under the loss policy ALL corresponds to the throughput gain achieved under the loss policy that satisfies (ONE) for N connections with respect to the throughput for $N = 1$. The loss policies under (ONE) provide no gain with respect to the loss policy ALL for $N = 1$, but they provide larger throughput for a factor which is exactly that given by (5) divided by $(1+\beta_1)/2$. The last means that loss policies under (ONE) provide throughput gain with respect the loss policy ALL equal to $2/(1+\beta_1)$, as

[†]This follows by the fact that under ALL there is a unique limit point x^* of the send rates just before a link congestion event such that $x_i^* = c/N$, $i = 1,2,\ldots,N$.

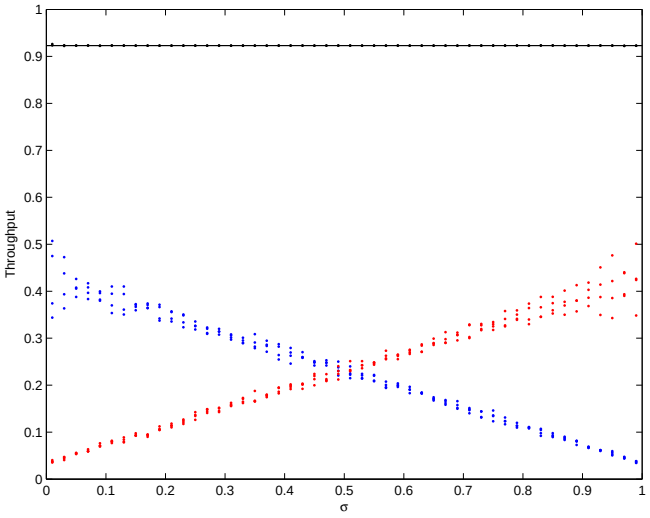

Fig. 3. Throughputs of 4 AIMD connections in Example 1. There are two classes of connections, each class containing 2 connections. A connection selected at a link congestion time is from the first class with probability σ. The flows of the two classes attain throughputs that depend on the parameter σ. However, the aggregate throughput is *insensitive* (dots aligned with the analytical prediction shown by the solid line). This validates Theorem 1.

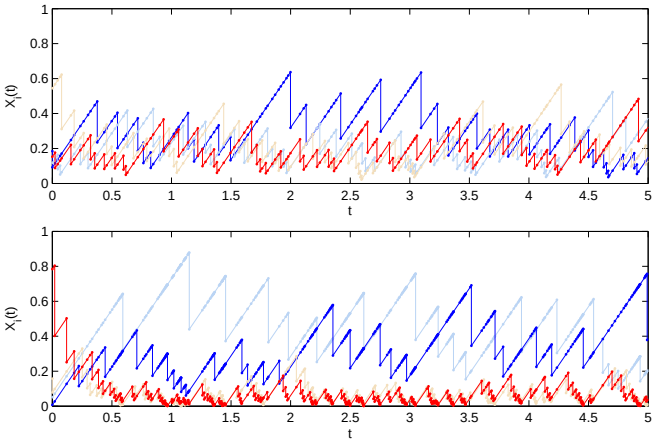

Fig. 4. Sample-paths of the rates of the 4 AIMD connections "orchestrated" by the rate-independent policy (Example 1). A connection is chosen from the probability distribution $[\sigma,\sigma,1-\sigma,1-\sigma]/2$. (Left) $\sigma = 1/2$ so that each connection has equal chance to be selected. (Right) $\sigma = 1/8$ so that two connections are selected more often than the other two connections.

the number of connections N tends to infinity. For TCP-like setting, this amounts to $4/3$, which is precisely compensating for the throughput deficiency of a single TCP connection.

4) What this tells us about MultTCP?: MultTCP, proposed by Oechlin and Crowcroft [10], is a window control protocol that aims at emulating a given number of N TCP connections. The design rationale is to scale the linear increase of the control with N as $N\eta_1$ and set the multiplicative decrease parameter to $1-(1-\beta_1)/N$. MultTCP can thus be seen as a single AIMD connection with appropriately set additive-increase and multiplicative-decrease parameters as a function of N. The choice of the multiplicative decrease parameter is based on an approximation by considering a virtual set of N parallel connections. It is assumed that at each congestion indication to these connections, only one connection undergoes a multiplicative decrease. Thus, it admits loss policies verifying (ONE). The send rate of this selected connection at a time just before a congestion indication is x/N, where x is the aggregate rate over all connections just before the congestion indication. The last is an approximation as, indeed, the selected connection may have the send rate other than the even share x/N. In the sequel, we consider MultTCP for TCP-like setting $\beta_1 = 1/2$. We already observed that MultTCP can be treated as a single AIMD connection with additive-increase parameter $N\eta_1$ and multiplicative-decrease parameter $1-N/2$, hence, the throughput follows immediately from (6)

$$\frac{\mathbb{E}(X^{\mathsf{MultTCP}}(0))}{c} = 1 - \frac{1}{4N}.$$

From (5), the throughput of N TCP-like connections is:

$$\frac{\mathbb{E}(X^{\mathsf{ONE}}(0))}{c} = 1 - \frac{1}{1+3N}.$$

We note that $\mathbb{E}(X^{\mathsf{MultTCP}}(0)) \geq \mathbb{E}(X^{\mathsf{ONE}}(0))$, for all $N \geq 1$, with equality only if $N = 1$. It is readily computed that the mean throughput of a MultTCP connection is less than 1.025 of the mean aggregate throughput of N TCP-like connections under a loss policy in ONE. It is hence, for any practical purpose, a good approximation. The last observation does not provide much motivation to obtain a correction for MultTCP so that its throughput is exactly that of N TCP-like connections under a loss policy that verifies (ONE). Note, however, redefining only β_1 to a function of N provides no solution, except for the trivial case $N = 1$. Indeed, by equating (5) and (6) and a simple calculation one obtains $\beta_1(N-1) = N-1$, hence, a $0 < \beta_1 < 1$ exists only if $N = 1$.

5) Round-trip time insensitivity: The result (1) suggests insensitivity of the mean throughput to the round-trip time of the connections as (5) does not depend on η_1. With TCP-like connections, the parameter η_1 will be defined as reciprocal of a fixed round-trip time to the power of 2.

III. Steady-state distribution of the aggregate send rate is NOT insensitive

The insensitivity property found in the earlier section is for the first-order moment of the time-stationary aggregate send rate. The time-stationary distribution is *not* insensitive to the loss policy. This was already showed in [5] by computation of the time-stationary second moments for particular policies. We provide a complete proof for the second moments, which yields corrected versions of the corresponding expressions in

[5]. A new information is for the beatdown policy.

Theorem 2: Consider two identical AIMD connections with $\beta_1 = 1/2$. The time-stationary second moments for either flow are given by

1) **Beatdown**: $\frac{\mathbb{E}(X^{(1)}(0)^2)}{c^2} = \frac{1}{3} \approx 0.33333$
2) **Rate-independent**: $\frac{\mathbb{E}(X^{(1)}(0)^2)}{c^2} = \frac{5}{24} \approx 0.20833$
3) **Rate-proportional**: $\frac{\mathbb{E}(X^{(1)}(0)^2)}{c^2} = \frac{679}{3396} \approx 0.19994$
4) **Largest-rate**: $\frac{\mathbb{E}(X^{(1)}(0)^2)}{c^2} = \frac{4}{21} \approx 0.1905$.

The proof is a tedious exercise of elementary and Palm calculus and is deferred to [6].

Note that the order of the loss policies with respect to the time-stationary second-order moment of the throughput is rather natural. Intuitively, one would expect that the largest-rate policy attains smallest second-order moment as it *balances* the rates at congestion times by picking a connection with largest rate. One also would expect that the rate-proportional policy has a larger second-order moment of the throughput, but not too much as it also favors selecting connections with larger rates at link congestion times. The rate-independent policy, with each connection selected equally likely, has a larger second-order moment of the throughput than both the largest-rate and rate-proportional. The beatdown policy attains the largest second-order moment of the throughput within the given set of policies.

IV. Additional analysis for largest-rate and beatdown policies

This section shows some auxiliary results on the dynamics for particular loss policies, the largest-rate and beatdown. The analysis provides some insight on the dynamics of the send rates and serve in the proof of the second-order moment results in the earlier section. It may be however skipped by a reader non interested in technical details, with no sacrifice of continuity.

A. Largest-rate policy

From now, we use the short-cut notation $\mathsf{x}_i(n) := X_i(T_n)$. We show that for a homogeneous population of N AIMD connections, there exists a unique stationary point for the rate vector $\mathsf{x}(n) = (\mathsf{x}_1(n), \cdots, \mathsf{x}_N(n))$, under the largest-rate policy and identify this stationary point.

In view of (3) and (4), dynamics of the rates under the largest-rate policy is fully described by the following *deterministic* dynamical system:

$$\mathsf{x}_i(n+1) = \mathsf{x}_i(n) + \left(\frac{\eta_i}{\overline{\eta}} - 1_{\{M_n=i\}}\right)(1-\beta_{M_n}) \vee_{j=1}^{N} \{\mathsf{x}_j(n)\}$$

where $M_n \in \operatorname{argmax}_j \mathsf{x}_j(n)$. Throughout, we use the operators $x \vee y := \max\{x,y\}$ and $x \wedge y := \min\{x,y\}$, for $x, y \in \mathbb{R}$.

This can be further rewritten as follows. Let $\tilde{\mathsf{x}}(n)$ be the vector $\mathsf{x}(n)$ with coordinates ordered in decreasing order, for any given n. Thus, $\tilde{\mathsf{x}}_1(n) \geq \tilde{\mathsf{x}}_2(n) \geq \cdots \geq \tilde{\mathsf{x}}_N(n)$, for all n. We can write:

$$\mathsf{x}_i(n+1) = \begin{cases} a_i \tilde{\mathsf{x}}_1(n), & M_n = i \\ \mathsf{x}_i(n) + b_i \tilde{\mathsf{x}}_1(n), & M_n \neq i, \end{cases} \tag{7}$$

where $a_i := \beta_i + \eta_i(1-\beta_i)$ and $b_i := \eta_i(1-\beta_i)$, $i = 1, 2, \ldots, N$.

In the remainder of this section, we confine to homogeneous flows, thus $a_i = a_1$ and $b_i = b_1$, for all $i = 2, 3, \ldots, N$. Without loss of generality, we assume $c = 1$. Thus, for any n, $\mathsf{x}(n)$ takes value on a $N-1$ dimensional simplex $\mathcal{S}_{N-1} = \{x \in \mathbb{R}_+^N : \sum_{i=1}^N x_i = 1\}$ and $\tilde{\mathsf{x}}(n)$ on the subset of $\mathcal{S}_{N-1}$, $\mathcal{S}_{N-1}^0 = \{x \in \mathbb{R}_+^N : \sum_{i=1}^N x_i = 1,\ x_i \geq x_j,\ i \leq j\}$.

Lemma 1 (representation): Consider a homogeneous population of N AIMD users competing for a link under the largest-rate loss policy. The dynamics of the ordered vector of rates $\tilde{\mathsf{x}}(n)$ is given by the recurrence

$$\tilde{\mathsf{x}}(n+1) = g(\tilde{\mathsf{x}}(n)), \tag{8}$$

$$\begin{array}{rccc} \text{where} & \mathcal{S}_{N-1}^0 & \longmapsto & \mathcal{S}_{N-1}^0 \\ & x & \to & g(x) \end{array}$$

with $g(\cdot) = (g_1(\cdot), g_2(\cdot), \ldots, g_N(\cdot))$, and

$$g_i(x) = \begin{cases} a_1 x_1 \vee (x_2 + b_1 x_1), & i = 1 \\ (a_1 x_1 \vee (x_{i+1} + b_1 x_1)) \wedge (x_i + b_1 x_1), & \text{else} \\ a_1 x_1 \wedge (x_N + b_1 x_1), & i = N. \end{cases}$$

This lemma is proved in [6].

Proposition 1: The map $g : \mathcal{S}_{N-1} \to \mathcal{S}_{N-1}$ of the recurrence (8) has a unique fixed point x^0 in the interior of $\mathcal{S}_{N-1}$ (all coordinates strictly positive) given by

$$x_i^0 = \begin{cases} \frac{2}{(1+\beta_1)N + (1-\beta_1)} & i = 1 \\ x_1^0 \left(1 - \frac{1-\beta_1}{N}(i-1)\right) & i = 2, 3, \ldots, N. \end{cases} \tag{9}$$

The proof is deferred to [6].

Remark 1: The fixed point uniquely identifies a N-periodic sequence $\mathsf{x}^0(n)$ in the simplex $\mathcal{S}_{N-1}$. We readily construct a continuous-time function $\mathsf{x}^0(t)$ with values in $\mathcal{S}_{N-1}$ associated to the sequence $\mathsf{x}^0(n)$ by the definition of the inter-congestion times (4), with $T_0 = 0$. As customary, a time-stationary sample-path is constructed by shifting the T-periodic function $\mathsf{x}^0(t)$ for a time uniformly at random on the interval $[0, T]$.

Remark 2: The proposition says that the stationary point of the rates just before link congestion event is such that that connections are signaled a congestion indication in a round-robin fashion.

B. Beatdown

It is particular to the beatdown loss policy that congestion times accumulate, i.e. the number of congestion events on some finite time intervals is infinite (see Figure 5). This implies that $\mathbb{E}^0(T_1) = 0$. This prevents us from applying Theorem 1. In this section we circumvent this problem by looking at some special link congestion times, and show that indeed the throughput-insensitivity property holds.

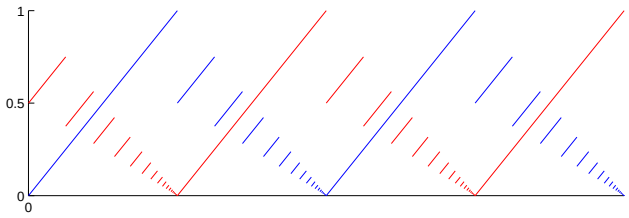

Fig. 5. Sample-paths of the rates of two AIMD flows "orchestrated" by the beatdown policy.

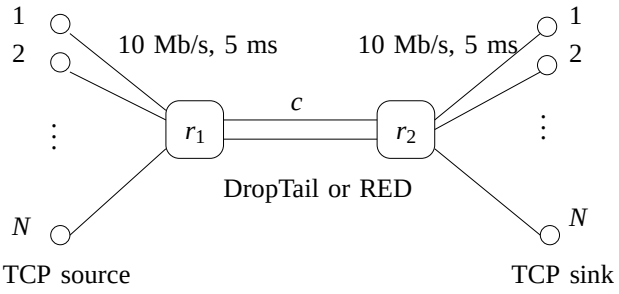

Fig. 6. ns-2 network configuration.

Define τ_n as a link congestion time at which a connection becomes tagged according to the beatdown policy. Let $y_i(n)$ be the rate of a flow i at time τ_n (subsequence of $x_i(n)$). With an abuse of notation, let $z_i(n)$ indicates whether a flow i was selected at time τ_n. The rates embedded at the selected link congestion times evolve as follows:

$$y(n+1) = (1 - z_i(n))(y_i(n) + \eta_i(\tau_{n+1} - \tau_n)). \qquad (10)$$

Indeed, if a flow i is tagged at τ_n, i.e. $z_i(n) = 1$, then $y_i(n+1) = 0$. Else, $z_i(n) = 0$, and thus the flow continues increasing its rate with rate η_i from the initial rate $y_i(n)$. Summing $\sum_{j=1}^{N} y_j(n) = c$, yields

$$\tau_{n+1} - \tau_n = \frac{\sum_{j=1}^{N} z_j(n) y_j(n)}{\sum_{j=1}^{N} (1 - z_j(n)) \eta_j}. \qquad (11)$$

Finally, we have:

$$y_i(n+1) = (1 - z_i(n)) \left(y_i(n) + \frac{\eta_i}{\overline{\eta} - \eta_{M_n}} y_{M_n}(n) \right),$$

where, recall, $M_n = i$ for i such that $z_i(n) = 1$.

We have the following result whose proof is in [6].

Theorem 3: For a finite and homogeneous population of AIMD connections, any time-stationary beatdown loss policy is throughput-insensitive.

In the sequel of this section, we give the stationary point for the flow rates at selected congestion times, for a homogeneous population of AIMD connections. Let $\tilde{y}(n)$ be the vector of the rates $y(n)$ sorted in decreasing order. The dynamics is as follows:

$$y_i(n+1) = \begin{cases} 0 & M_n = i \\ y_i(n) + \frac{1}{N-1} \tilde{y}_1(n) & \text{otherwise.} \end{cases}$$

The stationary point satisfies the following identities: $\tilde{y}_k = \tilde{y}_{k+1} + \frac{1}{N-1} \tilde{y}_1$, $k = 1, 2, \ldots, N-1$. It follows

$$\tilde{y}_k = \frac{N-k}{N-1} \tilde{y}_1.$$

Now, $\sum_{i=1}^{N} \tilde{y}_i = c$, from which it follows $\tilde{y}_1 = 2c/N$. In summary,

$$\frac{\tilde{y}_i}{c} = \begin{cases} \frac{2}{N} & i = 1 \\ \frac{1}{N-1} \left(1 - \frac{i}{N}\right) & i = 2, 3, \ldots, N-1 \\ 0 & i = N. \end{cases}$$

V. Experimental Results

The results in earlier sections are exact for AIMD connections and the prevailing assumptions therein. In this section, we provide experimental validation of our analysis by simulations and internet measurements. Simulation and measurements scripts and data are available on-line at [1].

We performed ns-2 simulations as this allows us to easily configure queueing disciplines in order to conduct controlled experiments. Our internet measurements provide validation in real-world scenarios.

A. Setup of ns-2 experiments

We describe first our ns-2 experiments. We consider a system of N identical TCP-SACK connections that traverse a bottleneck as showed in Figure 6. The bottleneck link implements either DropTail or RED queueing discipline. Payload of a TCP packet is set to 1000 bytes. Maximum window size is set to 20000 packets. Each simulation is run for 2000 seconds and initial 1000 seconds are truncated to reduce the effect of initial transient. Start times of connections are randomized by uniformly drawing the starting times within interval $[0, 4]$ seconds.

In some simulations we use RED bottleneck. Original design goal of RED was to mitigate TCP window synchronization [12]. RED parameters are set as follows. Minimum and maximum threshold sizes are set to 0.6 and 0.8 factor of the buffer length. The RED packet drop probability function is chosen with the value equal to 0.1 at the maximum buffer threshold. The queue size averaging factor is set to 0.001. The RED option `wait` was enabled; this would enforce some waiting between successive packet drops.

Theorem 1 suggests the following claim: (H) a system of N identical TCP connections that compete for a bottleneck with a small buffer attains link utilization $1 - 1/(1+3N)$.

B. DropTail

The results are shown in Figure 7. We observe that typically, for small buffer lengths, the achieved throughput is smaller than predicted by the analysis. Moreover, we note that the larger the buffer is, the larger the throughput is. If buffer length is sufficiently large, throughput is larger than predicted by the analysis. We verified by inspection that in many cases in Figure 7, TCP windows are synchronized. This is exhibited for $N = 5$ and 10 in Figure 8. The figure shows sum of

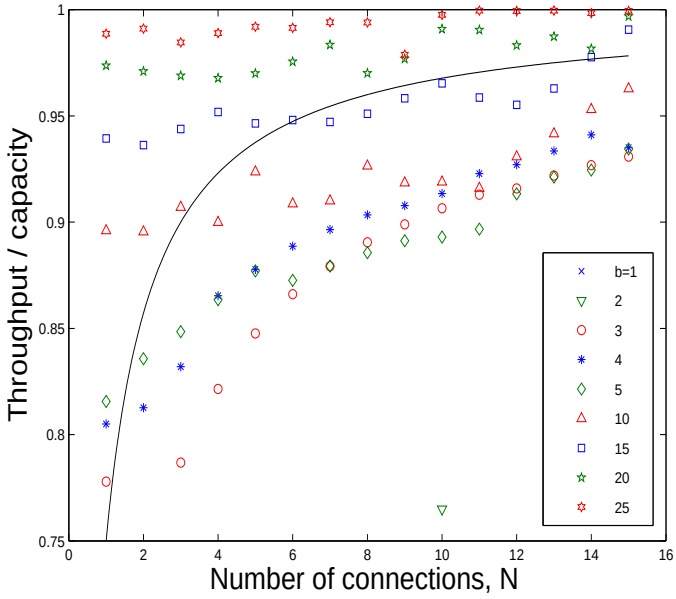

Fig. 7. Time-average send rate normalized with link capacity versus the number of TCP connections. Results are obtained in ns-2 for DropTail with varying buffer length b as indicated in the figure.

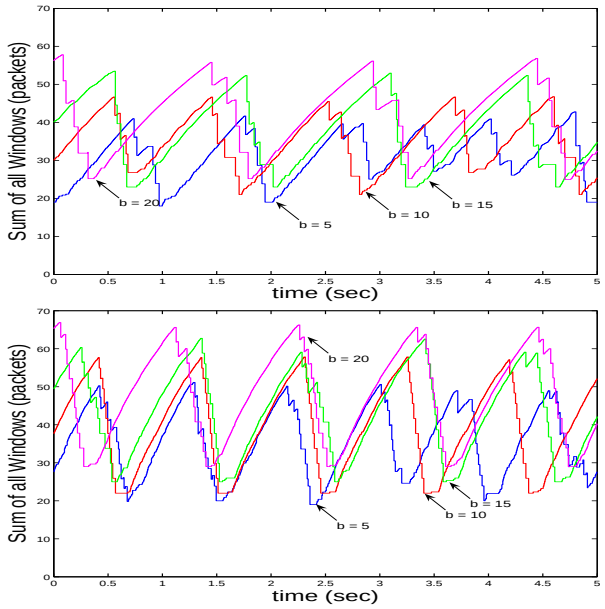

Fig. 8. Sum of congestion windows over all TCP connections for (top) $N = 5$ and (bottom) $N = 10$, competing for bottleneck link with propagation round-trip time 120 ms and buffer lengths as indicated in the figure. The results clearly indicate synchronization of TCP connections.

congestion windows over all connections versus time. The sum of congestion windows essentially evolves over time as congestion window of a single TCP connection, which indicates synchronization. This is further reflected in Figure 9 that shows congestion windows of individual connections.

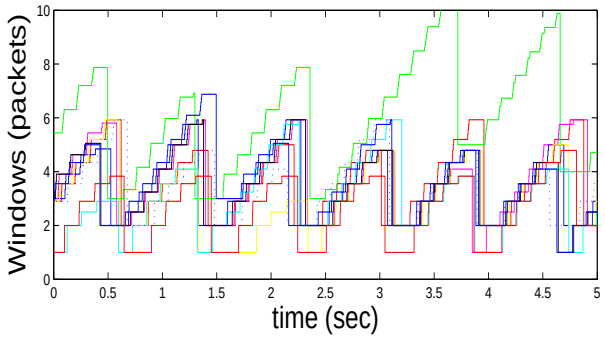

Fig. 9. Synchronization in DropTail bottleneck with $N = 10$ and $b = 5$.

C. RED

We show results in Figure 10. The results exhibit reasonable conformance to Theorem 1.

D. Loss polices that conform to ONE

We observed TCP window synchronization in experiments with DropTail. In order to validate Theorem 1, our aim is to design experiments in which assumption ONE holds, i.e. at each congestion event exactly one connection is signaled congestion. To that end, we implement a loss-policy—threshold-dropper—described as follows. The queue buffer length is set to virtually infinite value. The dropping is based on a finite positive threshold T_h and loss policy described as follows. Consider the system from a time at which there are at most T_h queued packets. The first packet arrival that exceeds threshold is dropped and the corresponding connection is declared selected. The selected connection remains selected for an interval of τ time units, where τ is set to a factor of the mean round-trip time. This process repeats. No packets are dropped during this period. It is clear from the description that threshold-dropper enforces assumption ONE. It also permits implementation of various loss polices.

Figure 11–left shows throughput achieved with threshold-dropper that implements rate-independent loss policy. We note a good qualitative conformance with Theorem 1, in particular when the number of connections is not too large. For sufficiently large number of connections, the throughput is essentially equal to the link capacity. In these case, the queue does not deplete, which explains larger throughput than predicted by the analysis. Analogous results for largest-rate loss policy are showed in Figure 11–right. The experimental results obtained for rate-independent and largest-rate policy match well, which indicates throughput insensitivity. This is in conformance to the analysis.

We further provide results that indicate threshold-dropper mitigates TCP window synchronization. Figure 12–top shows the congestion window evolution of 10 individual TCP connections. This is substantiated by looking at the sum of congestion windows in Figure 12–bottom. The alert reader will compare this with Figure 8.

1) What loss polices emerge with DropTail and RED?: It is often assumed that a connection observes loss events with intensity in time proportional to send rate. This corresponds

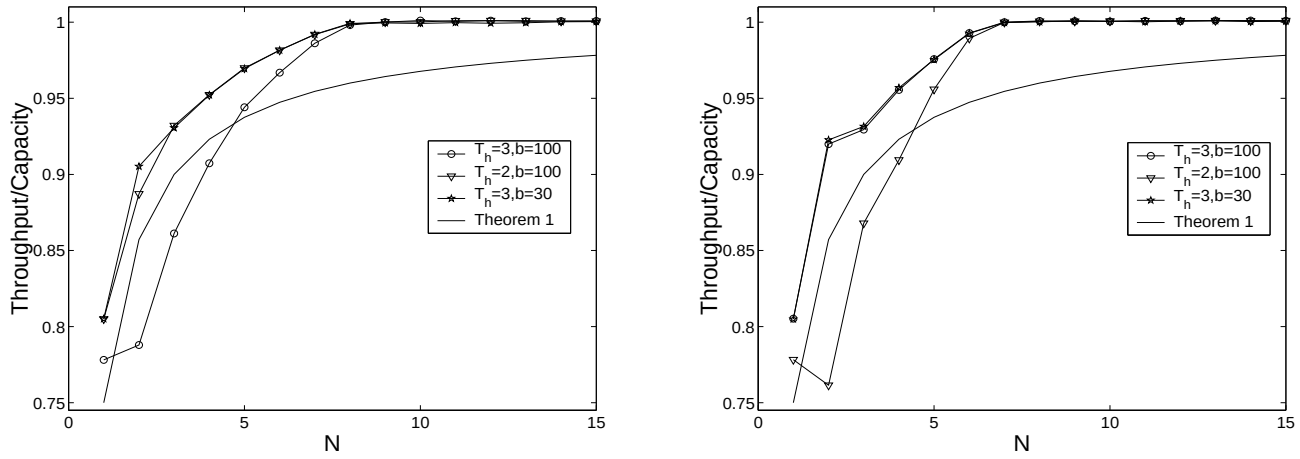

Fig. 11. The ratio of the throughput and link capacity versus the number of connections N with threshold-dropper: (left) rate-independent, (right) largest-rate. The results conform well with Theorem 1 (solid line).

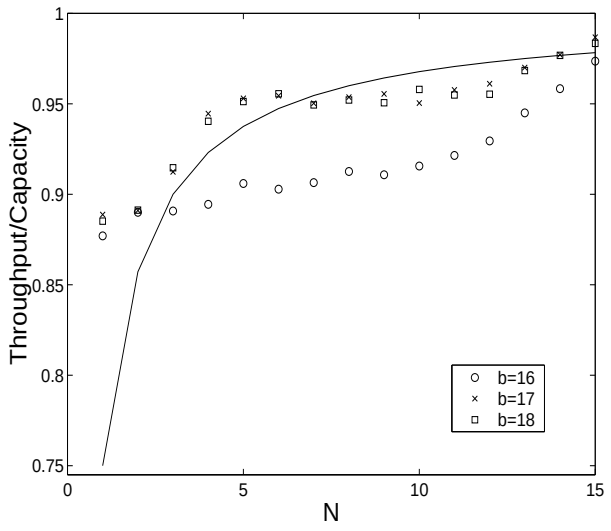

Fig. 10. Time-average send rate normalized with link capacity versus the number of TCP connections. Results are obtained in ns-2 for RED with varying buffer length as indicated in the figure.

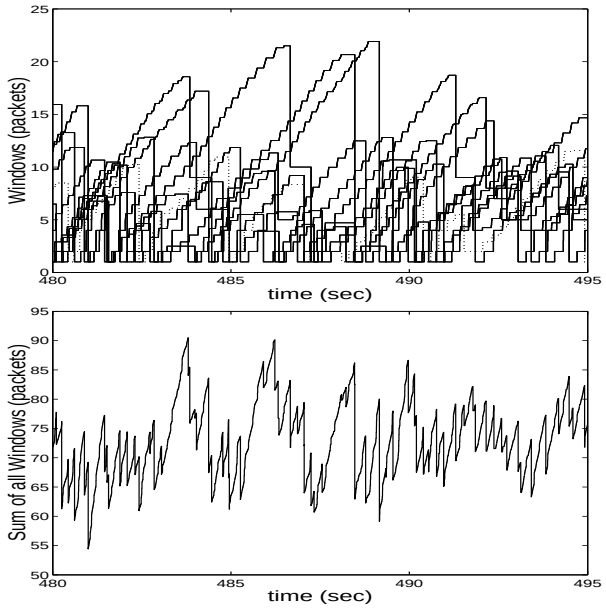

Fig. 12. (Top) Congestion windows of $N = 10$ TCP connections, (bottom) sum of all congestion windows. The loss policy is threshold-dropper with threshold $T_h = 3$. The results suggest absence of synchronization.

to rate-proportional loss policy introduced earlier. We investigate over a limited set of experiments whether or not rate-proportional loss policy emerges with DropTail and RED. To that end, we proceed with the following test. Time is binned into consecutive intervals of length T time units. A connection is assumed to receive a congestion indication in a bin n, if there is at least one packet drop for this connection in the nth bin. The window size of the connection i in bin n is denoted by $w_i(n)$. Our goal is to estimate $\mathbb{P}(z_i(n) = 1 | w_i(n) = w)$, where $z_i(n) = 1$ if the connection i is indicated congestion in bin n, else $z_i(n) = 0$. We estimate this conditional probability by the empirical mean

$$\overline{z}(w) := \frac{\sum_n \sum_{i=1}^{N} 1_{\{z_i(n)=1, w_i(n)=w\}}}{\sum_n \sum_{i=1}^{N} 1_{\{w_i(n)=w\}}}$$

which under stationary and ergodicity assumptions converges with probability 1 to $\mathbb{P}(z_i(n) = 1 | w_i(n) = w)$, as number of time bins tends to infinity. In Figure 13, we show the empirical mean $\overline{z}(w)$ versus w for systems of $N = 10$ TCP connections with propagation round-trip time of 120 ms and DropTail link for buffer sizes = 5, 10, 15, 20 packets. Same is showed in Figure 14, but for RED. Some of the results suggest the larger the congestion window of a connection just before a link loss event is, the more likely the connection is signaled a congestion indication.

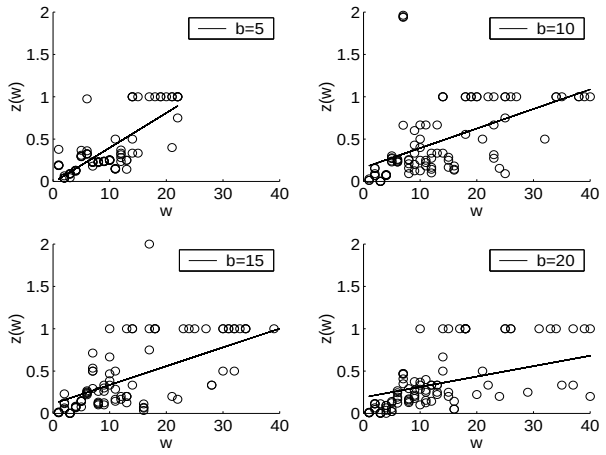

Fig. 13. Empirical estimate of $\mathbb{P}(z_i(n)|w_i(n)=w)$ versus the window size w for DropTail link and 10 TCP connections, for buffer length as indicated in the graphs.

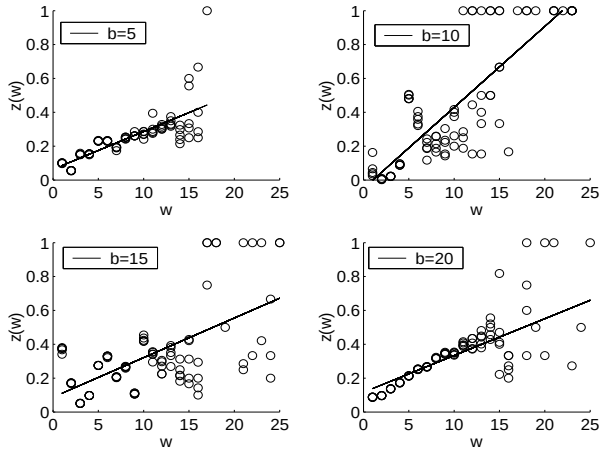

Fig. 14. Same as in Figure 13, but for RED.

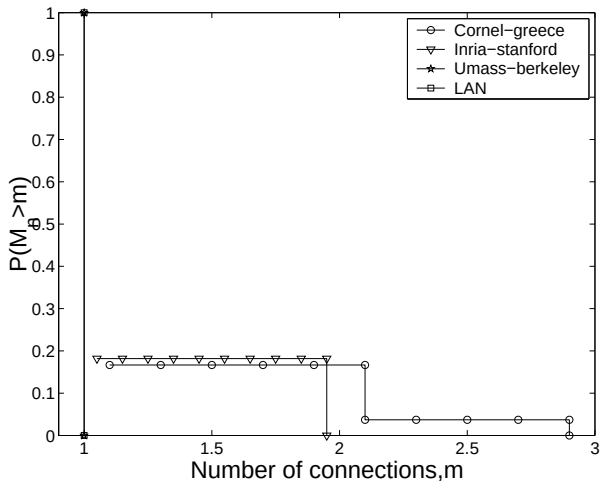

Fig. 15. Complementary distribution function of the number of connection assigned a packet drop in an interval length of const times RTT for 10 parallel TCP connections.

E. Internet measurements

We now proceed with more realistic experiments by measurements on the internet planet-scale testbed, PlanetLab [2]. We choose randomly a set of distinct end-to-ends path from different continents such as Europe and North America. For each path, we conduct 10 repeated experiments. Each set of measurement starts with parallel TCP transfers with up to 20 or 50 parallel TCP connections. We measure the aggregate throughput obtained using the tool `iperf` [3] and round-trip time using `ping`. Some basic information about end-to-end paths is showed in Table I.

The measured throughputs are showed in Figure 16 in which each run is for 10 sec duration (i.e., default in iperf and corresponds to at least 50 RTT samples). We set c to be the highest throughput ever observed among all runs. In Figure 16, we note a very good match of measurements `inria-stanford` with Theorem 1. The measurements `LAN` suggests buffer saturation. All results exhibit qualitative conformance with the throughput increase with the number of connections predicted by the analysis. In Figure 16, the receiver window sizes are set as given in Table I. In order to ensure that TCP connections are not receiver window constrained, we increase the receiver window to 200 MBytes on a linux host, `frumious.bu.edu`. In Figure 18, we show measured throughputs on planetlab hosts at MIT, UCSD and Cornell along with result of Theorem 1.

We next investigate whether connections are synchronized. On distinct end-to-end paths, we run `iperf` tool with 10 parallel TCP connections. We collect packets transmitted in both directions of a connection with tcpdump from which we infer loss events. We clump loss events to congestion events based on a time threshold (a factor of the round-trip time) and count the number of distinct connections, M_n, that suffer at least one loss event in a time bin n. We plot the empirical complementary distribution function of M_n in Figure 15. We observe that M_n is a small fraction of $N = 10$ on almost all paths. Furthermore, we plot loss events over time in Figure 17 and observe that losses do not appear clustered together, which suggests absence or weak TCP window synchronization. We further plot congestion window evolution of 10 connections transferring data for 20 mins on the path `Cornell-BU`–see Figure 19. We found this path to be lossy and congestion window evolutions exhibit some synchronization. In Figure 18 (middle), we observe a linear growth in throughput for $N \leq 5$ but we ensure there is no receiver window limit by setting it a sufficiently large value. Moreover, bandwidth-delay product of the path, `UCSD-BU` is around 1MByte (assuming $c =$ 100Mbps). It could be the reason that the connections are application limited (i.e., iperf) and $N \leq 5$ connections may never saturate the link. We observe 5 loss-events in `MIT-BU` path, 2 on `UCSD-BU` path and 501 on `Cornell-BU` path in one run of each experiment whereas the measured throughputs are averaged over 10 runs.

TABLE I

SET OF END-TO-END PATHS USED IN MEASUREMENTS

path	Source host	Destination Host	RTT	Receiver Buffer Size
umass-berkeley	planetlab1.cs.umass.edu	planetlab2.millennium.berkeley.edu	97ms	256 KBytes
inria-stanford	planetlab3.inria.fr	planet2.scs.stanford.edu	194.7ms	256 KBytes
cornell-greece	planetlab4-dsl.cs.cornell.edu	planet2.ics.forth.gr	337.9ms	256 KBytes
LAN	einat.inria.fr	titeuf.inria.fr	1.6 ms	128 KBytes
MIT-BU	planetlab1.csail.mit.edu	frumious.bu.edu	0.993ms	200 MBytes
UCSD-BU	planetlab1.ucsd.edu	frumious.bu.edu	84.2ms	200 MBytes
Cornell-BU	planetlab4-dsl.cs.cornell.edu	frumious.bu.edu	66.7ms	200 MBytes

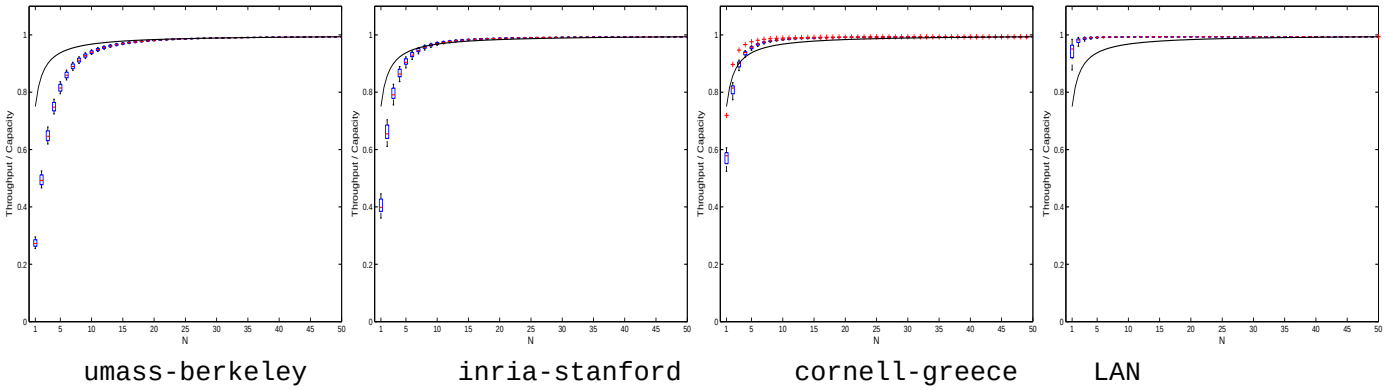

Fig. 16. Throughputs measurements of N parallel TCP connections. For each N, 10 distinct measurements were conducted and aggregate throughput was measured using [17]. The figures show boxplots that indicate the measurement samples are concentrated around corresponding average values. Solid line is the result of Theorem 1 interpolated by fitting the measured value for $N = 50$ with the corresponding analytical value. The measurement results exhibit qualitative growth of the throughput with N as predicted by the analysis. The latter shows particularly good conformance with Theorem 1.

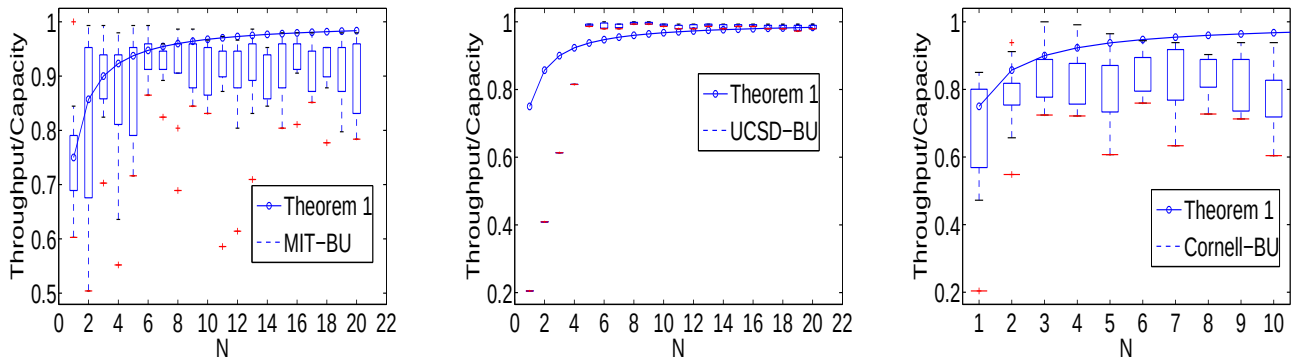

Fig. 18. Same as in Figure 16 but for paths `MIT-BU`, `UCSD-BU` and `Cornell-BU`. Measurements were done directly using `iperf` and aggregate throughput is averaged over 10 runs where each run was for 50sec

VI. CONCLUSION

We found a formula for aggregate throughput of parallel connections that individually adapt their send rates according to additive-increase, multiplicative-decrease connections, as in TCP congestion avoidance mode. The result is obtained under assumption that loss events over connections are non synchronized and queueing is negligible. Other than that, the result holds under much generality; in particular, it allows for various loss assignments over connections over distinct loss events in time.

The implications of the result are: (i) it suggests an explicit dependence of the aggregate throughput on the number of parallel TCP sockets; (ii) it tells the throughput deficiency due to additive-increase, multiplicative-decrease adaptation is almost entirely eliminated already with a few connections ($\geq 90\%$ utilization for as few as 3 sockets); (iii) it reveals aggregate throughput insensitivity on the way losses are assigned over competing connections [a property found in [5], but under more restrictive assumptions].

Our Internet measurements identify cases when the model assumptions seem to be verified and there is a good conformance between the theory and measurements. On the other hand, the measurements also discover cases when throughput-deficiency is due to synchronization of losses (window syn-

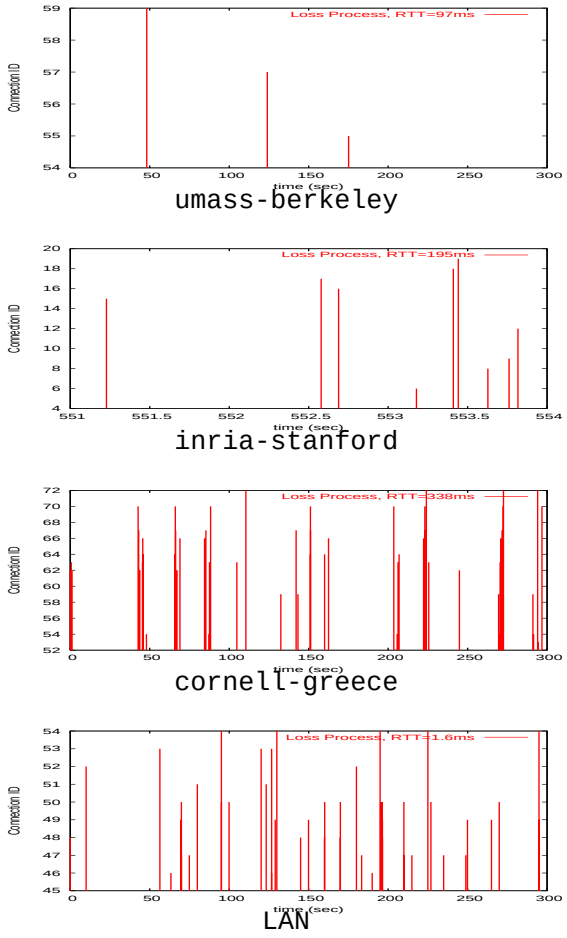

Fig. 17. Connections incurring loss events versus time These experiments show loss events when 10 parallel connections run for 5 mins subsequently to experiments in Figure 16. In some cases loss events are rare.

chronization) and receiver window constraint.

The result suggests that over Internet paths with neither synchronized losses nor receiver window limitation, already a few parallel TCP sockets would suffice to eliminate the throughput-deficiency of the additive-increase, multiplicative-decrease adaptation.

ACKNOWLEDGMENTS

E. A and B. T were in part supported by EuroNGI NoE and INRIA TCP cooperative research action. M. V. is grateful to Laurent Massoulié and Don Towsley for discussions of some technical points. He also thanks Dinan Gunawardena for his valuable suggestions on practical implications of the throughput-insensitivity found in this paper. Jon Crowcroft and Karl Jeacle pointed to the Grid application. We also thank Paul Stauffer for his help in configuring TCP parameter values in `frumious.bu.edu`.

REFERENCES

[1] `http://research.microsoft.com/~milanv/socks.htm`.
[2] `http://www.planet-lab.org`.
[3] `http://dast.nlanr.net/Projects/Iperf/`.

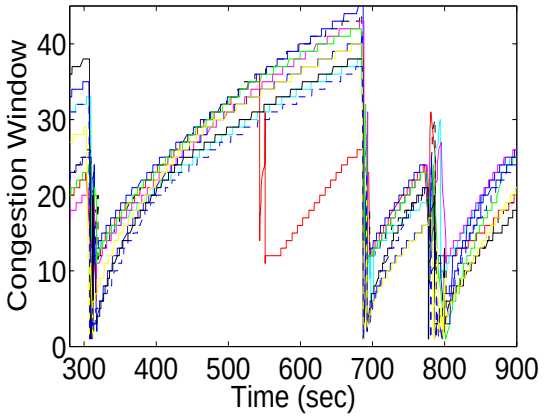

Fig. 19. Congestion window process of 10 connections running for 20 mins between `planetlab4-dsl.cs.cornell.edu` and `frumious.bu.edu` measured subsequently to experiment in Figure 18 (right).

[4] E. Altman, K. Avrachenkov, and C. Barakat. A stochastic model of tcp/ip with stationary random losses. In *Proc. of the Sigcomm'00*, pages 231–242, 2000.
[5] E. Altman, R. El Azouzi, D. Ros, and B. Tuffi n. Loss Strategies for Competing TCP/IP Connections. In *Proc. of the Networking 2004*, Athens, Greece, May 2004.
[6] E. Altman, D. Barman, B. Tuffi n, and M. Vojnović. Parallel tcp sockets: Simple model, throughput and validation. Microsoft Research Technical Report,`http://research.microsoft.com/~milanv/TR-2005-130.pdf`, 2005.
[7] E. Altman, F. Boccara, J. Bolot, P. Nain, P. Brown, D. Collange, and C. Fenzy. Analysis of the tcp/ip flow control mechanism in high-speed wide-area networks. In *34th IEEE Conference on Decision and Control*, pages 368–373, New Orleans, Louisiana, Dec 1995.
[8] G. Appenzeller, I. Keslassy, and N. McKeown. Sizing Router Buffers. In *Proc. of ACM Sigcomm 2004*, Portland, Oregon, USA, Aug/Sept 2004.
[9] F. Baccelli and D. Hong. AIMD, Fairness and Fractal Scaling of TCP Traffi c. In *Proc. of IEEE Infocom 2002*, New York, NY, http://www.di.ens.fr/∼trec/aimd, 2002.
[10] J. Crowcroft and P. Oechslin. Differentiated End-to-End Internet Services using a Weighted Proportional Fair Sharing TCP. *ACM Computer Communications Review*, 47(4):275–303, 2004.
[11] S. Floyd and K. Fall. Promoting the use of end-to-end congestion control in the Internet. *IEEE/ACM Trans. on Networking*, 7(4):458–472, August 1999.
[12] S. Floyd and Van Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Trans. on Networking*, 1(4):397–413, August 1993.
[13] T. Hacker, B. Athey, and B. Noble. The end-to-end performance effects of parallel TCP sockets on a lossy wide-area network. In *16th IEEE-CS/ACM International Parallel and Distributed Processing Symposium*, Ft. Lauderdale, FL, April 2002.
[14] Van Jacobson and M.J. Karels. Congestion avoidance and control. In *Proc. of the ACM SIGCOMM'88*, pages 314–329, Stanford, August 1988.
[15] T. Kelly. Scalable TCP Improving Performance in HighSpeed Wide Area Networks. In *PFLDNet'03*, Geneve, Switzerland, February 2003.
[16] R. El Khoury and E. Altman. Analysis of Scalable TCP. In *In Proc. of Het-Nets'04*, West Yorkshire, United Kingdom, July 2004.
[17] Dong Lu, Yi Qiao, Peter Dinda, and Fabian Bustamante. Modeling and Taming Parallel TCP on the Wide Area Network. In *In Proceedings of the 19th IEEE IPDPS'05*, Denver, Colorado, April 2005.
[18] J. Mahdavi and S. Floyd. TCP-Friendly Unicast Rate-Based Flow Control. Technical note sent to end-2-end interest mailing list, http://www.psc.edu/networking/papers/tcp_friendly.html, January 1997.
[19] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling TCP Reno Performance: A Simple Model and its Empirical Validation. *IEEE/ACM Trans. on Networking*, 8(2):133–145, 2000.