Theoretical Analysis of Performances of TCP/IP Congestion Control Algorithm with Different Distances

Tsuyoshi Ito and Mary Inaba

Department of Computer Science, The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan

Abstract. According to current TCP/IP implementations, the acceleration in additive-increase phase depends on the distance of connection. In this paper, the performance of Additive Increase and Multiplicative Decrease (AIMD) congestion control algorithm in TCP is analyzed in two ways, both focusing on effects of the *heterogeneity* or the mixture of different accelerations caused by different distances. First, we analyze flow time minimization, extending the competitive analysis by Edmonds et al. to heterogeneous case. We show (a) the performance loss of TCP/IP in Long Fat Pipe Networks (LFNs) is caused by the heterogeneity rather than long distance itself. Next, we step forward to more realistic *single-drop model*, where upon each congestion only one, instead of all, connection drops rate, and analyze asymptotic total and per-connection bandwidth utilizations. We show (b) increasing the number of connections makes total utilization better as opposed to common model, (c) in homogeneous environments, *victim policies* or choice of which connection drops do not affect total utilization, and (d) in heterogeneous two-connection environments, maximum total utilization is achieved by certain victim policy which leads to unfair share, whereas fair utilization is achieved by certain random victim policy.

1 Introduction

The Transmission Control Protocol (TCP) is used by most data transfer in the Internet. It has been widely known [1, 2] that the current implementations of TCP do not perform well in long-distance high-bandwidth networks, or Long Fat Pipe Networks (LFNs). These days, the backbone network over gigabits per second such as Abilene and GÉANT is rapidly constructed, and the bandwidth of the links in the Internet, especially of the long-distance ones, is increasing. As a result, the Internet has become an LFN. However, the exact reason why the performance of TCP suffers in LFNs is not known. It is observed [3] that the negative impact of using faster network interface than the bottleneck capacity is more severe in long-latency connections than in short-latency connections.

At the same time, there are more and more needs for the transfer of various kinds of large data. For example, people will send e-mails with video images of tens or hundreds of megabytes length in near future. As an example where huge

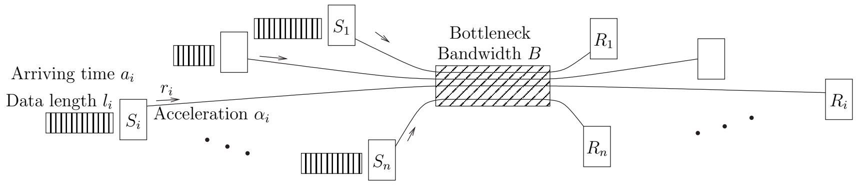

Fig. 1. A single-bottleneck network consisting of a bottleneck with bandwidth B and n connections with different distances. Each connection C_i has the acceleration α_i which is inversely proportional to the square of its RTT.

data is concerned, some research institutes currently receive data of terabytes produced by scientific measurement instruments by the physical transportation of Digital Linear Tapes (DLTs), but they can receive them online if the LFN problem is resolved [4]. This indicates the necessity of the analysis of the performance of the long-time transfer of very large data.

The congestion control in TCP makes a guess on the appropriate transmission rate by only using the data exchanged between the endpoints of the connection. The current congestion control algorithm increases the transmission rate by α in unit time while the transmission succeeds, and decreases it by multiplying $1-\beta$ to the current rate. This algorithm is called Additive Increase and Multiplicative Decrease (AIMD) [5].

In this paper, in quest of the exact reason the current TCP does not perform well on LFNs, theoretical analyses are performed from various viewpoints on the most fundamental network model with a single bottleneck, as depicted in Figure 1. The performance is analyzed in the case that each of the connections with different distances transfers large data. As we focus on the transfer of large data, we consider only the AIMD congestion avoidance phase of TCP of sufficiently long period, ignoring the effect of the slow start phase which is relatively short period of time.

In the real world, the distance of a connection affects the behavior of the AIMD mainly in three ways. (1) Acceleration: In the AIMD algorithm, the transmission rate of a connection increases by $\alpha = c/T^2$ per unit time while the transmission succeeds, where c is Sender Maximum Segment Size (SMSS), which is a constant for usual case, and T is Round Trip Time (RTT), which reflects the distance of the connection. This α is called the *acceleration* of the connection. (2) Response time: After a node transmits its data, it takes the time amount of RTT to know whether the transmission has succeeded or failed. (3) The number of congestion points: Long-distance connections pass more congestion points such as routers and switches than short-distance connections.— We focus on the difference of (1) to isolate the effects of different distances of connections. We say the environment is *homogeneous* if all the connections have an equal acceleration, and *heterogeneous* otherwise.

Edmonds et al. [6] consider the single-bottleneck network and prove by theoretical analysis that the AIMD algorithm performs well when all the connections

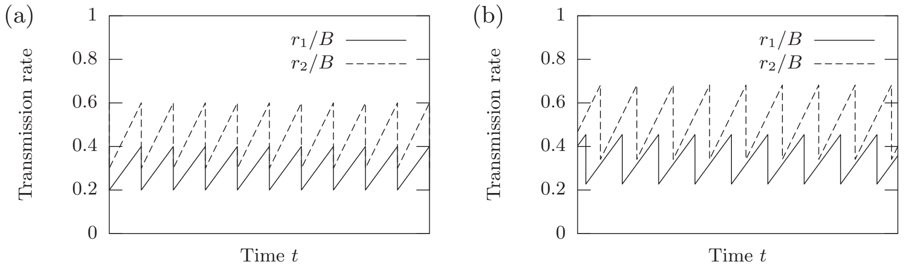

Fig. 2. Time evolution of the transmission rates r_1 and r_2 of the two connections with the accelerations $\alpha_1 : \alpha_2 = 2 : 3$ and the drop factor $\beta = 1/2$. (a) uses the all-drop model, and (b) uses the single-drop model and Periodic victim policy.

have a common acceleration, that is, in the homogeneous case. In section 4, we extend their result to the heterogeneous case and show a result that suggests the AIMD does not perform well when connections have different accelerations, thus explaining the low throughputs under the coexistence of short- and very-long-distance communication.

In section 5, we further analyze the total bandwidth utilization and the share of the available bandwidth in the stationary state. Many existing results, including the result by Edmonds et al. and our extension to it, assume that when congestion occurs at the bottleneck, all the connections drop their transmission rate at the same time as depicted in Figure 2 (a). With this assumption, it is shown that the total utilization does not depend on the number of connections. To fill a gap between this assumption and the reality, we consider another model of the drop as shown in Figure 2 (b). In the new model, when congestion occurs, one connection is chosen as *victim* and only the victim drops its transmission rate and the transmission rate of the other connections does not change. We call this model the *single-drop model* and refer to the previous model as the *all-drop model*. In the single-drop model, we have several choices of the order in which the connections are chosen as a victim. We refer to these choices as *victim policies*. Victim policy is an abstraction of the algorithm executed by the bottleneck router to choose which packet to discard when the network is congested. We investigate which victim policy maximizes the total utilization. In addition, we prove that in the all-drop model and some single-drop victim policies, the bandwidth is shared among the connections in proportion to their accelerations, hence unfairly in proportion to the inverse of the square of their RTTs, whereas one of the randomized victim policies results in fair per-connection utilization.

2 Related works

TCP congestion control is an algorithm which works without knowledge about the bandwidth of links or information about other communication sharing the network. There are two approaches to the theoretical analysis of the performance of such incomplete-information algorithms. Probabilistic analysis is the analysis of the average case after assuming some probabilistic distribution of the unknown information, and competitive analysis is the analysis of the worst *competitive ratio* of the performance to the fictional case where the complete information were available to an algorithm.

Probabilistic analysis. Several papers [7–9] analyze how the throughput of homogeneous TCP connections is affected by random packet losses under the assumption that every packet is dropped independently with a constant probability. De Vendictis et al. [10] consider the environment with two connections where one connection uses the current TCP and the other uses a different congestion control algorithm called TCP Vegas, and analyze the throughputs of the connections in the stationary state.

Competitive analysis. At the top of our knowledge, the application of the competitive analysis to the performance evaluation of TCP congestion control was first proposed and performed by Karp et al. [11]. They formalized the congestion control as the algorithm to guess a secret available bandwidth which changes little by little over time. Edmonds et al. [6] consider the setting where multiple homogeneous connection jobs arrive and complete over time. They regard TCP as an online and distributed algorithm to share the available bandwidth among ongoing connections and compare it to scheduling algorithms which share the available processors among ongoing jobs in the centralized manner. They show that TCP achieves a constant competitive ratio independent of the number of connections by the competitive analysis against the optimal offline scheduling algorithm. However, the result holds only for the homogeneous case.

3 Definitions

Figure 1 illustrates the *single-bottleneck network* we consider. The network consists of one bottleneck with bandwidth B, n *senders* $S_1, \ldots, S_n$ on one side of the bottleneck, and n corresponding *receivers* $R_1, \ldots, R_n$ on the other side. Sender S_i sends its data to receiver R_i, together making a *connection* C_i. S_i sends data at the rate of r_i per unit time, where r_i, called the *transmission rate* of C_i, changes as time goes on. Any algorithm must control the transmission rates so that their sum $\sum_{i=1}^{n} r_i$ never exceed B. Here we use *fluid model*: r_i can be any nonnegative real value and the data can be sent as if it does not have the minimum unit such as a packet, an octet or even a bit.

Each connection C_i is associated with three constants: the *arriving time* a_i, the *data length* $l_i > 0$, and the *acceleration* $\alpha_i > 0$. The connection C_i starts at time a_i to send l_i amount of data. We consider both the case of $l_i < \infty$ and the case of $l_i = \infty$. The acceleration α_i is used by the AIMD algorithm as described later.

In this paper, the behavior of the AIMD congestion control algorithm is formalized as follows. A constant $0 < \beta \leq 1$ fixed. β is called *drop factor* and common to all the connections. Each C_i maintains its transmission rate $r_i \geq 0$ as follows. While $\sum_{i=1}^{n} r_i < B$, in other words, the sum of transmission rates of the n connections is less than the bottleneck bandwidth, each C_i transmits an infinitesimally small amount $r_i\,dt$ of data for an infinitesimally short time dt and increases r_i by $\alpha_i\,dt$. When $\sum_{i=1}^{n} r_i = B$, meaning that the sum of transmission rates hits the bandwidth, what happens depends on which *drop model* we adopt. (1) *All-drop model*: All the r_i's are multiplied by $(1-\beta)$ instantly at the same time, as shown in Figure 2 (a). (2) *Single-drop model*: One connection C_i is chosen as *victim* and its transmission rate r_i is multiplied by $(1-\beta)$ instantly. Note that in the single-drop model, the choice of victim is not unique, and we will discuss about *victim policies* in section 5. For example, Periodic victim policy defined in section 5.3 chooses every connection as victim in turn as shown in Figure 2 (b).

When $l_i < \infty$ and $\int_{a_i}^{t} r_i\,dt = l_i$, meaning that connection C_i has sent all of its data, then connection C_i terminates. In this case, the time elapsed since the arriving time a_i until the termination of C_i is called the *flow time* f_i of connection C_i, and the sum $F = \sum_{i=1}^{n} f_i$ is just called the *flow time*.

In the current TCP congestion control algorithm, β is fixed to $1/2$, and α_i is inversely proportional to the square of RTT of connection C_i. The case that α_i's are equal for all the connections is called *homogeneous* case, and the other case *heterogeneous* case.

4 Competitive analysis of flow time in heterogeneous environments

In this section, we assume the all-drop model and we consider the case that $l_i < \infty$ for all i, that is, each sender sends a finite amount of data. In this setting, we consider the optimization problem of minimizing the flow time.

Now consider the arriving time a_i is not known until the request of data transfer of C_i arrives at time a_i. Similarly, consider the data length l_i is not known until the sender sends l_i amount of data, reaching the end of data. This situation is common, because it corresponds to the case that the congestion control algorithm is implemented as a protocol stack independent of the application which decides when and which data to send. The AIMD algorithm works without any problem in this situation, because it does not use any information given in future to work. In this sense, the AIMD algorithm is called an *online algorithm.*

Besides, the AIMD is a *distributed algorithm* in the following sense. Each connection C_i only requires the information about its own parameters, a_i, α_i and l_i, and does not need to know the bottleneck bandwidth B or the parameters of the other connections, provided the sender knows whether $\sum r_i < B$ or $\sum r_i = B$. In TCP, this last additional information is supplied by the presence or the absence of acknowledgment from the receiver.

In contrast to the online and distributed AIMD algorithm, we can consider fictional *offline* and *centralized* algorithms. This kind of algorithms know B, and a_i and l_i of all the n connections before any request arrives, and controls all the r_i's simultaneously. Because offline and centralized algorithms have more access to knowledge than online and distributed algorithms like the AIMD, the optimal offline and centralized algorithm achieves no longer flow time than the AIMD.

For the homogeneous case where $\alpha_1 = \cdots = \alpha_n = \alpha$, Edmonds et al. prove the following.

Theorem 1 ([6]). *The AIMD is competitive to the optimal offline and centralized algorithm with a limited bottleneck bandwidth in the following sense. Let $\varepsilon > 0$ and*

$$s = 2(2+\varepsilon) \cdot \frac{1}{\beta} \cdot \frac{2}{2-\beta}, \tag{1}$$

and suppose we compare the flow time $F(\mathcal{C})$ of the set $\mathcal{C} = \{C_1, \ldots, C_n\}$ of connections achieved by the AIMD with bottleneck bandwidth B and that achieved $F_{\mathrm{OPT}}(\mathcal{C})$ by the optimal offline and centralized algorithm with bottleneck bandwidth B/s. Then, for $D = 4n\beta B/(s\alpha)$, it holds that $\frac{F(\mathcal{C})}{F_{\mathrm{OPT}}(\mathcal{C})+D} \leq 2 + \frac{4}{\varepsilon}$.

Now we consider the heterogeneous case. As we mentioned in section 3, the flow time $F(\mathcal{C})$ can be written by using the flow time f_i of individual connection as $F(\mathcal{C}) = \sum_{i=1}^{n} f_i$. In a similar way, we define *modified flow time* as: $F'(\mathcal{C}) = \sum_{i=1}^{n} \alpha_i f_i$, that is, the sum of the flow times of the connections weighted by the accelerations of the connections.

Then Theorem 1 is extended as follows.

Theorem 2. *Let $\mathcal{C} = \{C_1, \ldots, C_n\}$ be a set of connections. Suppose $\alpha_{\min} \leq \alpha_i \leq \alpha_{\max}$ for all i, and α_i be a multiple of α_{unit}. Let $\varepsilon > 0$, and define s is in equation (1). Let $F(\mathcal{C})$ be the flow time achieved by the AIMD with bottleneck bandwidth B and $F_{\mathrm{OPT}}(\mathcal{C})$ be that achieved by the optimal offline and centralized algorithm with bottleneck bandwidth B/s. Then, it holds that*

$$\frac{F'(\mathcal{C})}{F'_{\mathrm{OPT}}(\mathcal{C}) + \frac{\alpha_{\max}}{\alpha_{\mathrm{unit}}} D} \leq 2 + \frac{4}{\varepsilon} \quad \textit{and} \quad \frac{F(\mathcal{C})}{F_{\mathrm{OPT}}(\mathcal{C}) + D} \leq \frac{\alpha_{\max}}{\alpha_{\min}} \left(2 + \frac{4}{\varepsilon}\right)$$

where $D = n \cdot \frac{\beta^2(2-\beta)}{2+\varepsilon} \cdot \frac{B}{\alpha_{\mathrm{unit}}}$.

Proof (sketch). We make a new set $\mathcal{C}'$ of connections from the given set $\mathcal{C}$ so that all the connections in $\mathcal{C}'$ have the acceleration of α_{unit}. For each connection C_i in $\mathcal{C}$, let $n_i = \alpha_i/\alpha_{\mathrm{unit}}$ and divide C_i into n_i equal connections with arriving time a_i, data length l_i/n_i and acceleration α_{unit}. Then it holds $F'(\mathcal{C}') = F'(\mathcal{C})$. Because $\mathcal{C}'$ is made just by dividing the connections of $\mathcal{C}$ to smaller ones, it holds $F'_{\mathrm{OPT}}(\mathcal{C}') \leq F'_{\mathrm{OPT}}(\mathcal{C})$. The theorem is obtained by applying Theorem 1 to $\mathcal{C}'$.

This gives the same competitive ratio as the homogeneous case for the modified flow time, and $\alpha_{\max}/\alpha_{\min}$ times as worse competitive ratio as the homogeneous case for normal flow time.

Because the modified flow time attaches importance to the flow time of connections with large acceleration, or short-distance connections, the fact proven above that the modified flow time is near optimal explains that in heterogeneous case long-distance connections get less bandwidth, resulting worse competitive ratio of the normal flow time.

5 Analysis of asymptotic bandwidth utilization

In this section, we consider the case that $l_i = \infty$ for all i, that is, all the senders have infinite data to transmit and the connections never terminate. As discussed in the introduction, this is an approximation of the case that all the connections continue for a long time. Under this assumption, we analyze the asymptotic bandwidth utilization.

Let us introduce some notations. Let $A = \sum_{i=1}^{n} \alpha_i$. The transmission rate at time t is denoted by $r_i[t]$. Let $\boldsymbol{r}[t] = (r_1[t], \ldots, r_n[t])^{\mathrm{T}}$.

For $t_1 \le t_2$, the amount $W_i[t_1, t_2]$ of data transmitted in connection C_i between time t_1 and t_2 is $W_i[t_1, t_2] = \int_{t_1}^{t_2} r_i[t]\, dt$, and we let $W[t_1, t_2]$ be the total amount of data transmitted in n connections between the same period,

$$W[t_1, t_2] = \sum_{i=1}^{n} W_i[t_1, t_2] = \int_{t_1}^{t_2} (r_1[t] + \cdots + r_n[t])\, dt.$$

The (asymptotic) bandwidth utilization U_i of connection C_i and the (asymptotic) total utilization U are defined as the limit of time average of the proportion of transmission rate in available bandwidth[1]:

$$U_i = \frac{1}{B} \lim_{T \to \infty} \frac{W_i[0, T]}{T} \quad \text{and} \quad U = \frac{1}{B} \lim_{T \to \infty} \frac{W[0, T]}{T}.$$

A larger total utilization means the algorithm makes use of much bandwidth and that it is efficient. Besides, a small variation in the values of U_i means the algorithm is fair.

Most of the proofs are omitted due to space limitation.

5.1 Total and per-connection utilizations in all-drop case

Theorem 3. *In the all-drop model, the total and per-connection utilizations are* $U = 1 - \frac{\beta}{2}$ *and* $U_i = \frac{\alpha_i}{A} U$.

The proof of Theorem 3 uses the idea of "adjusted" and "unadjusted" bandwidths used in [6]. Theorem 3 says that in the all-drop model, the total utilization does not depend on the number of connections. This is different than the empirical fact. In the following sections, we consider the single-drop model.

5.2 Total utilization in homogeneous single-drop case

In this section we consider the homogeneous single-drop case where $\alpha_1 = \cdots = \alpha_n = \alpha$.

Theorem 4. *In the homogeneous single-drop model, total bandwidth utilization* U *is* $U = \frac{(2-\beta)n}{(2-\beta)n+\beta}$ *regardless of how we choose victim of each drop.*

[1] U_i and U may not have limit values depending on the choice of victims. In such cases, U_i and U are not defined.

Proof (sketch). By using the potential function

$$\varphi(\boldsymbol{r}) = \frac{1}{2\alpha} \cdot \frac{(2-\beta)\{B^2 - (B - \sum r_i)^2\} - \beta \sum r_i^2}{(2-\beta)n + \beta}.$$

it can be proven that for any $t_1 \leq t_2$,

$$W[t_1, t_2] + \varphi(\boldsymbol{r}[t_2]) - \varphi(\boldsymbol{r}[t_1]) = B\frac{(2-\beta)n}{(2-\beta)n + \beta}(t_2 - t_1),$$

which proves the claim.

Theorem 4 shows that in the single-drop model, the total utilization U increases as n increases, which means dividing data into multiple streams gives better total throughput. This is different from the case of the all-drop model.

Here is an intuitive interpretation of Theorem 4

Suppose we want to achieve high total utilization by choosing appropriate victim. When the sum $\sum r_i$ of transmission rates hits the bandwidth B, we are forced to choose a victim C_v and decrease the sum $\sum r_i$ by βr_v. One choice is to choose a connection with small r_v as victim to keep $\sum r_i$ relatively high and achieve a high throughput for a moment. But this way the other r_i's will increase a little, meaning that when a connection $C_{v'}$ other than C_v is eventually chosen as victim, $\sum r_i$ will decrease by much. Because we cannot continue choosing C_v as victim for an arbitrarily long time, sooner or later we have to pay for the increased $C_{v'}$, canceling the short-term gain of total utilization.

5.3 Total and per-connection utilizations under Periodic victim policy

In this section, we consider Periodic victim policy as a typical example of a deterministic policy. This policy is similar to the all-drop model in that it chooses every connection C_i equal times.

Definition 1. Periodic victim policy *is the policy where connection C_1 is chosen as victim of the first drop, C_2 of the next drop, then $C_3, \ldots, C_n$, and this process is repeated infinitely. An example is shown in Figure 2 (b).*

Theorem 5. *Under Periodic victim policy, it holds*

$$U = \frac{2-\beta}{2 - \beta(1 - \sum_{i=1}^{n}(\alpha_i/A)^2)}, \quad U_i = U \cdot \frac{\alpha_i}{A}.$$

The proof of Theorem 5 is based on the fact that the operation on vector $\boldsymbol{r}$ in every period is represented as the multiplication of a matrix. The theorem is obtained by computing the eigenvector of the matrix.

Theorem 5 implies that under Periodic victim policy, the bandwidth is shared in proportion to α_i like the all-drop model, and α_i's with small deviation give better total utilization.

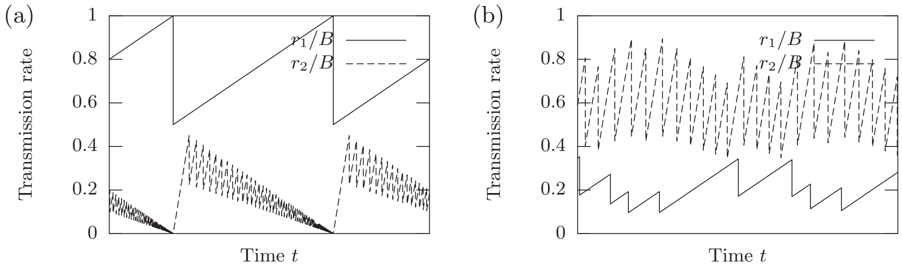

Fig. 3. Time evolution of the transmission rates r_1 and r_2 of the two connections with the accelerations $\alpha_1 : \alpha_2 = 1 : 9$ and the drop factor $\beta = 1/2$, under (a) Priority victim policy and (b) Share-Random victim policy.

5.4 Upper and lower bounds of total utilization in heterogeneous single-drop case

In this section, we consider Priority victim policy, which is the most unfair policy in some sense. Figure 3 (a) illustrates this policy. Intuitively, Priority victim policy chooses the connection C_i with the largest i that has nonzero transmission rate as victim. However, this informal definition is not accurate because the transmission rates are always nonzero. Instead, we define Priority victim policy as follows.

Definition 2. *Let* $0 < \varepsilon < 1/n$. ε-Priority victim policy *is the policy where on every drop, connection* C_i *with the largest* i *that satisfies* $r_i \geq \varepsilon B$ *is chosen as victim.* Priority victim policy *is the limit of* ε*-Priority policy as* $\varepsilon \to 0$.

Theorem 6. *Let* $A_0 = 0$ *and* $A_i = \alpha_1 + \cdots + \alpha_i$. *Under Priority victim policy,*

$$U = 1 - \prod_{i=1}^{n} \frac{(2-\beta)A_{i-1} + \beta\alpha_i}{(2-\beta)A_{i-1} + 2\alpha_i}, \; U_i = \frac{(2-\beta)\alpha_i}{(2-\beta)A_{i-1} + 2\alpha_i} \prod_{j=1}^{i-1} \frac{(2-\beta)A_{j-1} + \beta\alpha_j}{(2-\beta)A_{j-1} + 2\alpha_j}.$$

When $n = 2$, Priority victim policy gives the maximum and the minimum of the total utilization as the following theorem implies.

Theorem 7. *Let* $n = 2$ *and* $\alpha_1 \leq \alpha_2$. *If the total utilization* U *converges to some value, it holds*

$$1 - \frac{\beta}{2} \cdot \frac{\beta\alpha_1 + (2-\beta)\alpha_2}{2\alpha_1 + (2-\beta)\alpha_2} \leq U \leq 1 - \frac{\beta}{2} \cdot \frac{(2-\beta)\alpha_1 + \beta\alpha_2}{(2-\beta)\alpha_1 + 2\alpha_2}.$$

The proof of Theorem 7 uses potential function method with the potential function

$$\varphi(\boldsymbol{r}) = \frac{((2-\beta)\alpha_1 + \beta\alpha_2)r_1((2-\beta)B - r_1) + 2(2-\beta)\alpha_1(B - r_1)r_2 - 2\alpha_1 r_2^2}{2\alpha_1((2-\beta)\alpha_1 + 2\alpha_2)}.$$

This theorem indicates an interesting fact that as long as the total utilization is concerned, the router should discard the packet from the connection with the higher acceleration upon congestion. This strategy may also be useful to discourage the use of high acceleration by selfish connection, thus achieving high total utilization and penalty to selfish connection at the same time.

5.5 Total utilization with two heterogeneous connections under (p_1, p_2)-Random victim policy

In this section, we assume $n = 2$ and consider the following (p_1, p_2)-Random policy.

Definition 3. *Let* $p_1, p_2 > 0$ *and* $p_1 + p_2 = 1$. (p_1, p_2)-Random victim policy *is the policy where on every drop,* C_1 *is chosen as victim with probability* p_1 *and* C_2 *with* p_2*.*

When a randomized victim policy is used, the value of $W[t_1, t_2]$ varies depending on random choices. Therefore, we consider the expected total utilization $\mathrm{E}[U]$ and $\mathrm{E}[U_i]$.

Theorem 8. *Let* $p_1 = \alpha_1/A$ *and* $p_2 = \alpha_2/A$*. For any* $\boldsymbol{s} = (s_1, s_2)^{\mathrm{T}}$ *with* $s_1 + s_2 \leq B$*, the expected total and per-connection utilization under the condition* $\boldsymbol{r}[t_0] = \boldsymbol{s}$ *are given by* $\mathrm{E}[U] = \frac{2(2-\beta)}{4-\beta}$ *and* $\mathrm{E}[U_1] = \mathrm{E}[U_2] = \frac{2-\beta}{4-\beta}$*.*

The proof uses the potential function which is quadratic in r_1 and r_2.

This means that by choosing victim with probabilities proportional to α_i, the total utilization is equal to that in the two-connection homogeneous case and the two connections share the bandwidth in a fair manner, avoiding the inefficiency and unfairness caused by the heterogeneity.

5.6 Simulation of two heterogeneous connections under Share-Random victim policy

Definition 4. Share-Random victim policy *is the policy where on every drop, each* C_i *is chosen as victim with probability* r_i/B*, as shown in Figure 3 (b).*

Share-Random victim policy is the policy which is most easily implemented by a router placed at the bottleneck. Provided all the packets are infinitesimally short and the same length, the number of packets received by the router for each connection C_i at some moment is in proportion to the transmission rate r_i. When the sum $\sum r_i$ exceeds the capacity B of the router, the router will discard one packet received at the moment, which is for the connection C_i with the probability r_i/B. This scenario assumed the drop-tail behavior of the router, but the same thing happens if the router uses the Random Early Detection (RED) [12] given the buffer in the router is small enough.

We performed the numerical simulation of the utilizations by two connections under Share-Random policy, with B and $\beta = 1/2$ fixed and α_1 and α_2 altered

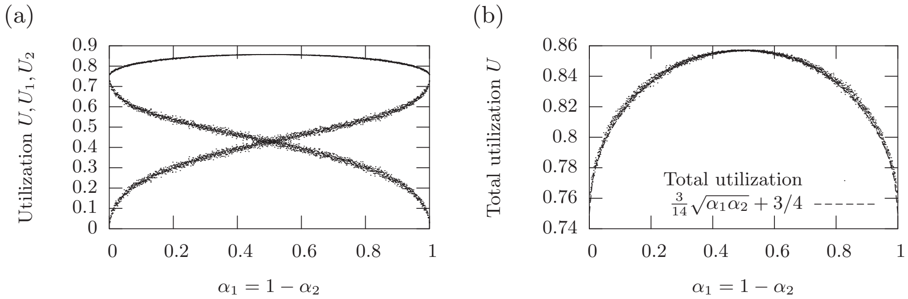

Fig. 4. Total utilization U and the utilizations U_i by each connection C_i of two connections under Share-Random victim policy with $\beta = 1/2$, $B = 1$ and different values for α_1 and α_2, while keeping $\alpha_1 + \alpha_2 = 1$.

while maintaining $A = \alpha_1 + \alpha_2 = 1$. Figure 4 shows the total and per-connection utilizations in this case. From Figure 4 (b) and the results with other values of β, we conjecture the following.

Conjecture. *In heterogeneous two-connection case under Share-Random victim policy, it holds*

$$\mathrm{E}[U] = \left(1 - \frac{\beta}{2}\right)\left(1 + \frac{2\beta}{4-\beta} \cdot \frac{\sqrt{\alpha_1\alpha_2}}{A}\right).$$

In addition, Figure 4 (a) suggests that in Share-Random case, the sharing of bandwidth among the connections is closer to the fair sharing than the all-drop case and the single-drop Periodic case. It is nearly proportional to the square root of the acceleration, or inversely proportional to RTT. This can be interpreted that the Share-Random victim policy mitigates the unfairness caused by different accelerations by choosing the connection with higher throughput more often than the other connection.

6 Conclusion and future works

We performed analyses of the performance of AIMD congestion control algorithm focusing on the heterogeneity of accelerations α_i of connections. The competitive analysis of the total flow time showed that the performance loss was caused by the heterogeneity, suggesting that one of the causes of performance problem of TCP in LFNs is the mixture of connections with different distances, rather than just the long distances of connections. The analysis of bandwidth utilization in stationary state revealed that the victim policy greatly affected the performance. When two connections with different accelerations exist, the maximum total utilization is achieved by Priority victim policy and the fair utilization is achieved by (p_1, p_2)-Random policy.

To tackle the LFN problem, many alternative congestion control algorithms for TCP have been proposed [13–15]. The extension of our analyses to these new congestion control algorithms will be useful to compare them.

References

1. Jacobson, V., Braden, R.: TCP extensions for long-delay paths. RFC 1072 (1988) Obsoleted.
2. Jacobson, V., Braden, B., Borman, D.: TCP extensions for high performance. RFC 1323 (1992)
3. Nakamura, M., Inaba, M., Hiraki, K.: Fast Ethernet is sometimes faster than Gigabit Ethernet on LFN — observation of congestion control of TCP streams. In: Proceedings of the 15th IASTED International Conference on Parallel and Distributed Computing and Systems (PDCS), ACTA Press (2003) To appear.
4. Hiraki, K., Inaba, M., Tamatsukuri, J., Kurusu, R., Ikuta, Y., Koga, H., Zinzaki, A.: Data Reservoir: Utilization of multi-gigabit backbone network for data-intensive research. In: Proceedings of the IEEE/ACM SC2002 Conference. (2002)
5. Chiu, D.M., Jain, R.: Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. Computer Networks and ISDN Systems **17** (1989) 1–14
6. Edmonds, J., Datta, S., Dymond, P.W.: TCP is competitive against a limited adversary. In: Proceedings of the Fifteenth Annual ACM Symposium on Parallel Algorithms and Architectures. (2003) 174–183
7. Mathis, M., Semke, J., Mahdavi, J., Ott, T.: The macroscopic behavior of the TCP congestion avoidance algorithm. Computer Communication Review **27** (1997)
8. Lakshman, T.V., Madhow, U.: The performance of networks with high bandwidth-delay products and random loss. IEEE/ACM Transactions on Networking **5** (1997) 336–350
9. Padhye, J., Firoiu, V., Towsley, D., Kurose, J.: Modeling TCP throughput: A simple model and its empirical validation. In: Proceedings of the ACM SIGCOMM '98. (1988) 303–314
10. De Vendictis, A., Baiocchi, A.: Modeling a mixed TCP Vegas and TCP Reno scenario. In: Networking 2002: Proceedings of 2nd International IFIP-TC6 Networking Conference. Volume 2345 of Lecture Notes in Computer Science. (2002) 612–623
11. Karp, R.M., Koutsoupias, E., Papadimitriou, C.H., Shenker, S.: Optimization problems in congestion control. In: 41st Annual Symposium on Foundations of Computer Science (FOCS), IEEE Computer Society (2000) 66–74
12. Floyd, S., Jacobson, V.: Random early detection gateways for congestion avoidance. IEEE/ACM Transactions on Networking **1** (1993) 397–413
13. Floyd, S.: HighSpeed TCP for large congestion windows. Internet Draft (work in progress) (2003) `http://www.ietf.org/internet-drafts/draft-ietf-tsvwghighspeed-01.txt`.
14. Kelly, T.: Scalable TCP: Improving performance in highspeed wide area networks. First International Workshop on Protocols for Fast Long-Distance Networks (PFLDnet 2003) (2003) `http://datatag.web.cern.ch/datatag/pfldnet2003/papers/kelly.pdf`.
15. Jin, C., Wei, D.X., Low, S.H.: FAST TCP for high-speed long-distance networks. Internet Draft (work in progress) (2003) `http://netlab.caltech.edu/pub/papers/draft-jwl-tcp-fast-01.txt`.