

Adaptive AIMD Congestion Control

[Extended Abstract]

Alex Kesselman
School of Computer Science
Tel-Aviv University
Tel-Aviv, Israel
alx@cs.tau.ac.il

Yishay Mansour
School of Computer Science
Tel-Aviv University
Tel-Aviv, Israel
mansour@cs.tau.ac.il

ABSTRACT

The main objectives of a congestion control algorithm are high bandwidth utilization, fairness and responsiveness in changing environment. However, these objectives are contradicting in particular situations since the algorithm has to constantly probe available bandwidth, which may affect its stability. This paper proposes a novel congestion control algorithm that achieves high bandwidth utilization providing fairness among competing connections and, on the other hand, is sufficiently responsive to changes of available bandwidth. The main idea of the algorithm is to use *adaptive* setting for the additive increase/multiplicative decrease (AIMD) congestion control scheme, where parameters may change dynamically, with respect to the current network conditions.

Categories and Subject Descriptors

C.2.2 [Network Protocols, Internetworking]: [TCP];
F.2 [Analysis of Algorithms and Problem Complexity]

General Terms

Algorithms, Performance, Design

1. INTRODUCTION

With its explosive growth, the Internet backbone faces the challenge of operating at its capacity. The main congestion points today are links connecting local Internet Service Providers (ISP's) to the backbone and intra-domain links between domains run by different large ISP's. When TCP acquires information about the current network conditions, there exists a strong tension between needing to keep a large amount of data in flight in order to fill the bandwidth delay product versus having to wait lengthy periods of time to attain feedback regarding a changing environment, especially at the onset of congestion. Moreover, in

most cases the Internet does not provide any explicit information about the network state, it is up to the transport protocol to form its own estimates, and then to use them to adapt as efficiently as possible. For these reasons congestion avoidance and control have become critical to the use of the Internet. TCP congestion control based on additive increase/multiplicative decrease (AIMD) has been introduced by Jacobson [10]. This scheme avoids the congestion collapse, as shown by Floyd and Fall [6].

In recent years, significant research efforts have been devoted to modeling such a complex protocol as TCP. Under certain assumptions of synchronized feedback, Chiu and Jain [5] have shown that an AIMD control scheme converges to a stable and fair operating point efficiently utilizing the network resources (see also [14]). Yang and Lam [23] investigate tuning of the AIMD congestion control parameters. Kelly [13] and Low et al. [17, 18, 16] consider TCP congestion control as an algorithm seeking the global optimum under a certain utility (cost) function and infer the functions implicit in the existing protocols. A number of mechanisms have been designed for providing more fine grained congestion related feedback from routers [15].

While TCP congestion control is appropriate for applications such as bulk data transfer, some real-time applications find halving the sending rate in response to a single packet loss to be unnecessarily severe because it results in drastic degradation in the user-perceived quality. A number of alternative TCP congestion control algorithms have been proposed. Bansal and Balakrishnan [3] study binomial congestion control algorithms, which are a nonlinear generalization of AIMD. Floyd et al. [7] introduce the TFRC algorithm that adjusts its sending rate as a function of the measured loss rate. Bansal et al. [4] investigate the behavior of slowly-responsive congestion control algorithms. However, to be safe for deployment in the Internet these algorithms must be stable and interact well with TCP. A recently proposed solution is the TCP-friendly paradigm [19]. A congestion control mechanism is TCP-friendly if its bandwidth usage, in the presence of a constant loss rate, is the same as that of TCP.

Our work. We suggest an *adaptive* AIMD scheme, in which the parameters may change dynamically with respect to the obtained feedback¹. In a nutshell, each positive feedback in-

¹The feedback indicates whether any packets in the prior

creases the additive increase term (while resetting the multiplicative decrease factor), and each negative feedback increases the multiplicative decrease factor (while resetting the additive increase term). Notice that basically our scheme has an inherent multiplicative increase rule of the additive increase term. This strategy, on the one hand, achieves high bandwidth utilization providing fairness among competing connections and, on the other hand, is sufficiently responsive to changes of available bandwidth in dynamic environment. In particular, we demonstrate that (1) the length of a convergence period of our algorithm depends *logarithmically* on the available bandwidth (comparing to the linear time of the existing AIMD algorithms); (2) in a steady state the bandwidth utilization is close to the optimal; (3) the time it takes for a connection to reach its fair share is proportional to the square of the logarithm of the available bandwidth. This algorithm, as we will show, is not TCP-friendly, i.e. when competing with TCP for a bottleneck resource, it will consume a larger fraction of the bandwidth. Our algorithm demonstrates good theoretical characteristics and we believe that the proposed scheme may suggest new lines for further research in TCP congestion control.

Related work. Jin et al. [11] propose a TCP-friendly congestion control algorithm called SIMD (Square-Increase/Multiplicative-Decrease), which utilizes history information. Namely, the increase in window size is proportional to the square of the time elapsed since the detection of the last loss. Our scheme is similar to SIMD in sense that the increase rate also depends on the history, but contrary to [11], the decrease rate depends on the history as well.

Karp et al. [12] and Arora and Brinkman [1] study a model in which an adversary controls the available bandwidth and present almost optimal policies for regulating the rate of a single connection using multiplicative increase/multiplicative decrease (MIMD) approach. Garg and Young [9] analyze another MIMD-based protocol and demonstrate that under certain assumptions the network throughput is near the maximum possible. However, it is well known that the MIMD scheme is unfair while our adaptive AIMD scheme is able to achieve high utilization as MIMD does without sacrificing fairness.

The congestion control algorithm has been often modeled as a system with a binary feedback function indicating congestion. Shenker [22] introduces a simple flow control scheme in which sources make synchronous rate adjustments, and applies it to a network of Poisson sources and exponential servers. Unlike [22], we make no assumptions regarding the traffic generation process. Piccolboni and Schindelhauer [21] investigate the problem of predicting the bandwidth available. In contrast to [21], we compare our algorithm with an optimal strategy rather than the best constant prediction strategy.

Bar-Noy et al. [2] study dynamic bandwidth allocation that takes into account latency and utilization and the goal is to minimize the number of bandwidth allocation changes. In this work we do not care about the number of bandwidth allocation changes, but instead we concentrate on the other objectives.

window encountered congestion in the network.

Recently, Floyd et al. [8] proposed a modification of TCP congestion control that adapts the increase strategy (makes it more aggressive) for high bandwidth links. The idea to make the sending algorithm adaptive is also reinforced by the recent work of Maor and Mansour [20]. They introduced AdaVegas, an adaptive version of TCP Vegas, that updates the increase rate dynamically, and using simulations compared it to TCP Vegas.

The rest of the paper is organized as follows. In Section 2 we present our model. Section 3 describes our congestion control algorithm. In Section 4 we analyze the convergence of our algorithm. Section 5 contains analysis of long connections. In Section 6 we study the interaction of our algorithm with TCP. We conclude with Section 7.

2. MODEL DESCRIPTION

In this section we describe our model. We consider multiple connections sharing a network that has available bandwidth R (see Figure 1). We assume that all connections run the same congestion control algorithm, and have the same round trip time (RTT).

The connections can *join* and later *leave* the system. Time is divided into successive rounds. At the start of a *round*, each connection transmits a window of packets and at the end of a round, each connection receives a feedback. We assume that there is always data pending at the source while the connection is active. Next we introduce a few useful definitions.

DEFINITION 2.1. *We denote by S_t the set of active connections in the system at time t . We denote the sets of connections joining and leaving the system at time t by J_t and L_t , respectively. Namely, $S_{t+1} = S_t \cup J_t \setminus L_t$.*

DEFINITION 2.2. *We denote by $w_t(i)$ the window of the i -th connection at time t . We denote by W_t the cumulative window at time t , that is $W_t = \sum_{i \in S_t} w_t(i)$.*

Next we relate the cumulative window size to the transmission rate. Note that since we assume identical RTT 's, the total sending rate is proportional to the cumulative window size.

DEFINITION 2.3. *We denote by R_t the total transmission rate at time t , which equals W_t/RTT .*

We also define the target cumulative window at which the available bandwidth is fully utilized and no packets are dropped.

DEFINITION 2.4. *We denote by B the target cumulative window for which $R_t = R$, i.e. $B = R \cdot RTT$.*

By the end of a round t , all connections receive a binary feedback from the network F_t ². If $W_t \leq B$ then $F_t = 0$,

²This mechanism is similar to the Explicit Congestion Notification (ECN) protocol.

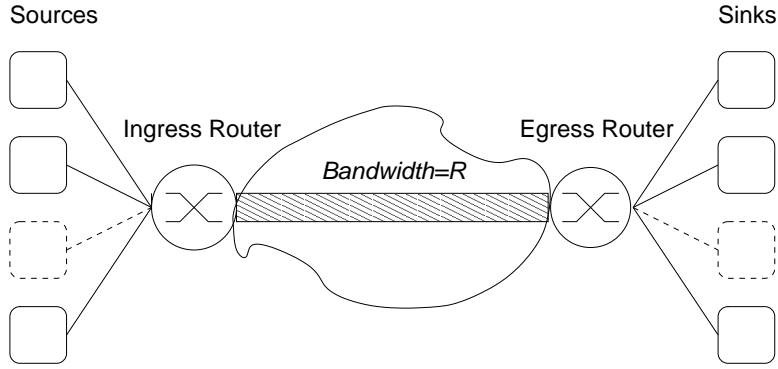


Figure 1: The network model.

which indicates a successful transmission of all packets sent. Otherwise, if $W_t > B$ then $F_t = 1$, which indicates dropping of one or more packets. There is also a cost function associated with dropped packets.

DEFINITION 2.5. In case $F_t = 1$, the overshooting cost at time t is defined as $L_t = \alpha(W_t - B)$, where $\alpha \geq 1$ is a parameter that reflects the cost of losing packets³.

Now we turn to define the objectives of a congestion control algorithm. In case the set of active connections does not change during sufficiently long period of time, we would like the congestion control algorithm to reach a steady state in which the throughput of the algorithm is close to the available bandwidth R .

DEFINITION 2.6. The congestion control algorithm A is said to stabilize if there exist constants M_R and C_R such that if the set of connections does not change during $[t_0, t']$ then $|R_{t'} - R|/R \leq C_R$, where $t' > t_0 + M_R$. We call to C_R the stability constant.

The goal of a congestion control algorithm is to maximize the number of packets transmitted and at the same time to minimize the loss incurred. Next we define the notion of so called “effective” gain.

DEFINITION 2.7. We define the effective gain of the congestion control algorithm A as:

$$G_t^A = \begin{cases} W_t & F_t = 0, \\ W_t - L_t & F_t = 1. \end{cases}$$

Notice that in a steady state the effective gain of a congestion control algorithm during every round is at least $B(1 - \alpha C_R)$. Given an input sequence σ , we define the value function of the congestion control algorithm A as:

$$V^A(\sigma) = \sum_{t=1}^{\text{length}(\sigma)} G_t^A.$$

³Our analysis can be extended to deal with non-linear cost functions. For example, a more realistic function may be a linear cost function with an additive constant term, that is $L_t = \beta + \alpha(W_t - B)$.

Notice that the value function may even be negative for some policies. However, as we will demonstrate, the value of our algorithm is always positive.

Now we define the utilization ratio of a congestion control algorithm A as the worst-case ratio between the value obtained by A and the value obtained by an optimal algorithm OPT taken over all input sequences. If the worst case ratio is negative, then we define the utilization ratio as zero. (Notice that a trivial policy that transmits no packets has utilization ratio equal to zero.)

DEFINITION 2.8. The utilization ratio U_A of the congestion control algorithm A is the minimum non-negative ratio, over all input sequences σ , between the value of A and the value obtained by an optimal algorithm OPT , that is $\max\{\min_{\sigma}(V^A(\sigma)/V^{OPT}(\sigma)), 0\}$.

Observe that in our model the optimal value of the input sequence σ equals to the sum of the bandwidth-delay product over all periods in which at least one connection in the system is active, i.e. $V^{OPT}(\sigma) = B \cdot |\{t : |S_t| > 0\}|$. That is due to the fact that even a single connection in the system may fully utilize the bandwidth.

Next we introduce a relative fairness function for a congestion control algorithm, which states that after a pre-defined period of time in the system the connection must reach at least the average window size with respect to the other active connections minus a fairness index.

DEFINITION 2.9. The congestion control algorithm is said to be fair if there exist constants M_F and C_F such that if a connection i is active during $[t_0, t']$ then $W_{t'}/|S_{t'}| - w_{t'}(i) \leq C_F$, where $t' > t_0 + M_F$. We call to C_F the fairness constant.

3. ADAPTIVE CONGESTION CONTROL

In this section we present our congestion control algorithm. The algorithm uses a TCP-like additive increase/multiplicative decrease scheme. The novel idea of the algorithm is to update the additive increase term γ and the decrease factor β with respect to the received feedback. We initialize γ to 1 and multiply it by $\phi > 1$ each time after a successful transmission. Similarly, we start with β equal to $\psi > 1$ and

multiply it by ψ after obtaining a negative feedback. The congestion control algorithm proceeds as follows.

Adaptive Congestion Control – ACC(ϕ, ψ)

```

Round 0: /* Initialization. */
    w0 = 1;
    β0 = ψ;
    γ0 = 1;

Round t:
    send(⌈wt⌉); /* Send window of ⌈wt⌉ packets. */
    If Ft = 0 Then /* No loss. */
        wt+1 = wt + γt;
        γt+1 = γt · φ;
        βt+1 = ψ;
    Else /* Loss occurred. */
        wt+1 = max(wt/βt, 1);
        βt+1 = βt · ψ;
        γt+1 = 1;

```

To complete the description of the ACC(ϕ, ψ) algorithm it remains to find proper ϕ and ψ that will ensure convergence and at the same time allow the algorithm to respond quickly to bandwidth changes. The former can be done by assigning ϕ and ψ values that are close to 1 while the latter is assured by exponential growth of γ and β . Intuitively, AIMD rule has inherent fairness characteristics and the algorithm indeed guarantees fairness, as we prove below. Now we define a property that is necessary for convergence of the ACC algorithm.

DEFINITION 3.1. *We say that ϕ and ψ are well-behaved if $\phi > \psi$ and for all $1 < k \leq D(\phi, \psi) = \lceil \sqrt{2 \log_\psi \phi} + 1/4 - 1/2 \rceil$ the following holds*

$$k > \lceil \sqrt{2 \log_\psi (\phi - (\phi - 1)/\psi^{k(k+1)/2})} + 1/4 - 1/2 \rceil.$$

We also define

$$C(\phi, \psi) = \max((1 - 1/\psi), (\phi - 1)(1 - 1/\psi)).$$

For example, the values $\phi = 1.4$ and $\psi = 1.1$ are well-behaved, and $C(1.4, 1.1) \approx 0.09$.

4. CONVERGENCE

In this section we analyze convergence of the ACC algorithm starting from an arbitrary state assuming that the set of connections in the system remains unchanged during the convergence period. We assume that there are N concurrent connections and $B \gg N$.

4.1 General Properties of the ACC Algorithm

In this section we study general properties of the ACC algorithm. It will be convenient for us to divide the run into phases, as follows.

DEFINITION 4.1. *We denote by a phase a maximal sequence of consecutive window increases followed by a sequence of decreases. We call the former sequence an increase sub-phase and the latter sequence a decrease sub-phase.*

The following claim states that the maximal overshooting of the ACC(ϕ, ψ) algorithm during an increase sub-phase is bounded by $\phi - 1$ times the difference between the cumulative window size at the end and at the beginning of the sub-phase plus the number of connections.

CLAIM 4.1. *For an increase sub-phase $\mathcal{I} = [t_1, t_2]$ the following holds:*

$$W_{t_2} \leq B + (\phi - 1)(B - W_{t_1}) + N.$$

PROOF. The claim trivially holds for $t_1 = t_2$. In what follows we assume that $t_2 > t_1$. Observe that for a round $t \in \mathcal{I}$ s.t. $t > t_1$, the increase term of a connection i is $\gamma_t(i) = \phi^{t-t_1}$. Therefore,

$$w_t(i) - w_{t_1}(i) = \sum_{j=0}^{t-1-t_1} \phi^j = \frac{\phi^{t-t_1+1} - 1}{\phi - 1}.$$

Thus, we obtain that:

$$\gamma_t(i) = (\phi - 1)(w_{t-1}(i) - w_{t_1}(i)) + 1.$$

After summing over all connections we get:

$$\begin{aligned} \sum_{i=1}^N \gamma_t(i) &\leq \sum_{i=1}^N ((\phi - 1)(w_{t-1}(i) - w_{t_1}(i)) + 1) \\ &= (\phi - 1)(W_{t-1} - W_{t_1}) + N. \end{aligned}$$

Notice that there is no loss at time t_2 , i.e. $W_{t_2} \leq B$. Hence,

$$\begin{aligned} W_{t_2+1} &= W_{t_2} + \sum_{i=1}^N \gamma_{t_2}(i) \\ &\leq B + (\phi - 1)(W_{t_2-1} - W_{t_1}) + N \\ &\leq B + (\phi - 1)(B - W_{t_1}) + N. \end{aligned}$$

□

The next corollary follows immediately from Claim 4.1.

COROLLARY 4.2. *The maximal cumulative window size is bounded from above by ϕB , for $W_{t_1} \geq N/(\phi - 1)$.*

Since the number of connections is negligible in comparison with the buffer size, in the sequel we will ignore the additive term of N in the bound of the above claim. Now we show that the length of a decrease sub-phase is bounded by a constant.

CLAIM 4.3. *The length of a decrease sub-phase $\mathcal{I} = [t_1, t_2]$ is at most $\lceil \sqrt{2 \log_\psi c + 1/4} - 1/2 \rceil$ if $W_{t_1} \leq cB$ for a constant $c > 1$.*

PROOF. Let us define $k = \text{length}(\mathcal{I})$. During \mathcal{I} the cumulative window W_t consequently decreases by a factor of $\psi(i)$ ($i = 1, \dots, k$). Notice that the total decrease factor up to time $t_2 + 1$ is $\prod_{i=1}^k \psi(i) = \psi^{k(k+1)/2}$, that is $W_{t_2+1} = \frac{W_{t_1}}{\psi^{k(k+1)/2}}$. The claim follows since $W_{t_2+1} \leq B$ and $W_{t_1} \leq cB$. \square

Claim 4.3 and Corollary 4.2 imply the next corollary.

COROLLARY 4.4. *The maximal length of a decrease sub-phase is $D(\phi, \psi)$ rounds.*

4.2 Analysis of a Single Connection

In this section we study the single connection case, i.e. $N = 1$. The next theorem shows that the $ACC(\phi, \psi)$ algorithm converges to a steady state and has a reasonably small stability constant for appropriate values of ϕ and ψ .

THEOREM 4.5. *The $ACC(\phi, \psi)$ algorithm for a single connection converges to a steady state starting from an arbitrary state after*

$$M_R = D(\phi, \psi)(\log_\phi B + D(\phi, \psi))$$

rounds and has a stability constant of

$$C_R = C(\phi, \psi),$$

assuming that ϕ and ψ are well-behaved.

PROOF. We first prove that the ACC algorithm converges to a steady state and then present a bound on the duration of the convergence period M_R . The proof of convergence is by induction on the phase number. We introduce the following notation:

$$f(n) = \begin{cases} D(\phi, \psi) - \min(n, (D(\phi, \psi) - 1)) & n \geq 0, \\ \infty & n = -1. \end{cases}$$

Let $g(x) = x(x + 1)/2$.

Induction Hypothesis. For the n -th phase the window size at the end of the increase and decrease sub-phases is at most $B(1 + (\phi - 1)(1 - \frac{1}{\psi^{g(f(n))}}))$ and at least $B/\psi^{g(f(n))}$, respectively.

Induction Basis ($n = 0$). By Corollary 4.2, the maximal window size is bounded by ϕB . The corresponding decrease sub-phase continues at most $D(\phi, \psi)$ rounds according to Corollary 4.4. Hence, the window size at the end of the first decrease sub-phase cannot drop below $B/\phi^{g(D(\phi, \psi))}$.

Induction Step. Suppose that the induction hypothesis is satisfied for all $n' \leq n$ and let us prove that it is also fulfilled for the $(n + 1)$ -th phase. By the induction hypothesis, the window size at the end of n -th phase is at least $B/\psi^{g(f(n))}$. Claim 4.1 implies that the window size at the end of the increase sub-phase is bounded by $B(1 + (\phi - 1)(1 - \frac{1}{\psi^{g(f(n))}}))$. From the Claim 4.3 and the fact that ϕ and ψ are well-behaved it follows that the number of rounds in the decrease sub-phase is at most $f(n + 1)$. Thus, the window size at the end of $(n + 1)$ -th phase is bounded from below by $B/\psi^{g(f(n+1))}$.

We have shown that after $D(\phi, \psi)$ phases the ACC algorithm converges to a steady state in which the window size changes between B/ψ and $B(1 + (\phi - 1)(1 - 1/\psi))$, which establishes the stability constant. Now we derive an upper bound on the length of the convergence period. Clearly, the duration of an increase sub-phase is bounded by $\log_\phi B$ rounds while Corollary 4.4 implies that the duration of a decrease sub-phase is at most $D(\phi, \psi)$ rounds, which yields the theorem. \square

For $\phi = 1.4$ and $\psi = 1.1$, in a steady state the connection rate will vary between $0.9 \cdot R$ and $1.04 \cdot R$, which is a fairly good bandwidth utilization.

4.3 Analysis of Multiple Connections

In this section we study the case of multiple concurrent connections. First we extend the analysis of utilization and convergence of the single connection case. Then we analyze fairness of the ACC algorithm.

OBSERVATION 4.6. *All connections have the same values of γ and β after getting the first negative feedback.*

In the sequel we assume that the connections have already received at least one negative feedback and thus are being synchronized (as we show later, it takes at most $\log_\phi(B/N)$ rounds). The following theorem establishes convergence of the ACC algorithm.

THEOREM 4.7. *The $ACC(\phi, \psi)$ algorithm for multiple connections converges to a steady state starting from an arbitrary state after*

$$M_R = D(\phi, \psi)(\log_\phi(B/N) + D(\phi, \psi))$$

rounds and has a stability constant of

$$C_R = C(\phi, \psi).$$

for well-behaved ϕ and ψ .

PROOF. (Sketch) The proof of convergence is analogous to that of Theorem 4.5. We derive a better bound on the duration of the convergence period since the duration of an increase sub-phase is bounded by $O(\log_\phi(B/N))$ rounds due to the fact that the cumulative window grows N times faster than in the single connection case. \square

Now we proceed to study fairness of the *ACC* algorithm. First we define the so called “excess” window of a connection, that is the difference between the connection window and the fair share if the connection window is above the fair share and zero otherwise. The cumulative excess window of the system is the sum of the excess windows of all individual connections.

DEFINITION 4.2. *We define the excess window of the i -th connection at time t to be $e_t(i) = \max(w_t(i) - W_t/|S_t|, 0)$. We denote by E_t the cumulative excess window at time t , that is $E_t = \sum_{i \in S_t} e_t(i)$.*

We present an upper bound on the number of rounds that is needed for a connection window to reach at least the fair share minus the fairness constant. We concentrate on the dynamics of the cumulative excess window in the system. The following observation states that the system is guaranteed to get to a fair operating point when the cumulative excess window drops below C_F .

OBSERVATION 4.8. *If $E_t \leq C_F$ then for all $i \in S_t$, $w_t(i) - W_t/|S_t| \leq C_F$.*

The next claim shows that the cumulative excess window does not change during an increase sub-phase.

CLAIM 4.9. *For any increase sub-phase $\mathcal{I} = [t_1, t_2]$, $E_{t_1} = E_{t_2}$.*

PROOF. Consider a round $t \in \mathcal{I}$. Notice that the windows of all connections are increased by the same term γ_t . We show that the cumulative excess window remains constant:

$$\begin{aligned} E_{t+1} &= \sum_{i=1}^N e_{t+1}(i) = \sum_{i=1}^N \max(w_{t+1}(i) - W_{t+1}/N, 0) \\ &= \sum_{i=1}^N \max((w_t(i) + \gamma_t) - (W_t + \gamma_t N)/N, 0) \\ &= \sum_{i=1}^N \max(w_t(i) - W_t/N, 0) \\ &= E_t. \end{aligned}$$

□

In the following claim we demonstrate that the cumulative excess window is decreased by a factor of at least ψ during a decrease sub-phase.

CLAIM 4.10. *For any decrease sub-phase $\mathcal{I} = [t_1, t_2]$, $E_{t_2} \leq E_{t_1}/\psi$.*

PROOF. Consider a round $t \in \mathcal{I}$. Observe that the windows of all connections are decreased by the same factor

$\beta_t \geq \psi$. We show that the cumulative excess window is decreased by a factor of ψ :

$$\begin{aligned} E_{t+1} &= \sum_{i=1}^N e_{t+1}(i) = \sum_{i=1}^N \max(w_{t+1}(i) - W_{t+1}/N, 0) \\ &= \sum_{i=1}^N \max((w_t(i)/\beta) - W_t/(\beta N), 0) \\ &= \frac{1}{\beta} \sum_{i=1}^N \max(w_t(i) - W_t/N, 0) \\ &= E_t/\beta. \end{aligned}$$

□

The next theorem establishes fairness of the *ACC* algorithm.

THEOREM 4.11. *The $ACC(\phi, \psi)$ algorithm will converge to the fair share starting from an arbitrary state after*

$$M_F = \log_\psi(B/C_F)(\log_\phi(B/N) + D(\phi, \psi))$$

rounds and has a fairness constant of C_F .

PROOF. By Claim 4.9 the cumulative excess window does not change during an increase sub-phase. Moreover, according to Claim 4.10 each decrease sub-phase reduces the cumulative excess window by a factor of ψ . Thus, after $\log_\psi(B/C_F)$ phases the total excess window will fall below C_F . The theorem follows by Observation 4.8 and the fact that the duration of a phase is bounded by $\log_\phi(B/N) + D(\phi, \psi)$ rounds. □

5. LONG CONNECTIONS

In this section we study the bandwidth utilization of the *ACC* algorithm when each connection remains in the system for at least $L \gg M_F$ rounds, where M_F is the convergence-to-fairness period of Theorem 4.11. We assume that the maximal number of concurrent connections is bounded by N while the set of connections in the system can change dynamically. We show that the utilization ratio of the $ACC(\phi, \psi)$ algorithm is $1 - \frac{2(\ln N + 1) \cdot M_R}{L} - \alpha C(\phi, \psi)$, where M_R is the convergence period of Theorem 4.7. (Recall that α is a parameter that reflects the cost of losing packets.)

THEOREM 5.1. *The utilization ratio of the $ACC(\phi, \psi)$ algorithm is at least*

$$U_{ACC(\phi, \psi)} = 1 - \frac{2(\ln N + 1) \cdot M_R}{L} - \alpha C(\phi, \psi)$$

assuming that all the connections have duration of at least $L > 2M_F$ rounds.

PROOF. We divide the schedule of the *ACC* algorithm into *periods*. A period starts upon arrival of a connection to an empty system and terminates when the system is empty

again. Each period is further partitioned in *phases* of length L . We assume that such a partition is possible.

By Theorem 4.7, in a steady state the *ACC* algorithm has a stability constant of at least $C(\phi, \psi)$. Thus, in a steady state the effective gain during a phase is at least $B \cdot L - \alpha C(\phi, \psi) \cdot B \cdot L$. In addition, some extra cost may be paid in case of overshooting or some bandwidth may be lost (with respect to *OPT*) in case of underutilization due to arrival or departure of connections. (Note that in every phase *OPT* sends $B \cdot L$ packets.) We call to the former the *overshooting cost* and to the latter the *opportunity cost*. We argue that during a phase the incurred overshooting cost is zero and the opportunity cost is bounded by $2B(\ln N + 1) \cdot M_R$ packets, which implies that the utilization ratio of the *ACC* algorithm is at least $1 - \frac{2(\ln N + 1) \cdot M_R}{L} - \alpha C(\phi, \psi)$.

Observe that when a new connection arrives and there are already some active connections in the system, it starts with the initial window of 1 and thus has no effect on the stability of the *ACC* algorithm, i.e. the incurred overshooting cost is zero. Moreover, a connection can enter an empty system only during the first phase of a period. By Theorem 4.7, the convergence period is bounded by M_R rounds. Hence, the incurred opportunity cost at most $B \cdot M_R$. Notice that no connection can leave the system during the first phase.

Now let us focus on departing connections. When the last connection leaves the system, the phase ends. Otherwise, if there remain some active connections, the system can become unstable. If the i -th connection leaves the system at time t then till the system stabilizes the opportunity cost is at most $w_t(i) \cdot M_R$ packets. A trivial bound on the total opportunity cost of departing connections during a phase is $B \cdot N \cdot M_R$. Next we derive a more accurate bound using the fairness properties of the *ACC* algorithm. The improved bound depends logarithmically rather than linearly on the number of connections.

Let us consider the k -th phase $\mathcal{I}_k = [t, t + L)$. Notice that the connections leaving the system throughout \mathcal{I}_k should have arrived during \mathcal{I}_{k-1} or earlier. Let $S' \in S_t$ be the set of connections that arrived before $t - M_F$ and let $N' = |S'|$. By Theorem 4.11, all connections in S' at time t have the same window size, which is at most B/N' . Observe also that all connections in S' always update their windows synchronously. Therefore, the opportunity cost incurred by the i -th departing connection from S' is at most $\frac{B}{i} \cdot M_R$. Since $N' \leq N$, the total opportunity cost incurred by connections in S' is bounded by

$$\sum_{i=1}^N \left(\frac{B}{i} \cdot M_R \right) \leq B(\ln N + 1) \cdot M_R.$$

It remains to bound the opportunity cost incurred by connections in $S'' = S_t \setminus S'$. The earliest possible departure time for these connections is $t + L - M_F$. Since $L - M_F > M_F$, their opportunity cost is also bounded by $B(\ln N + 1) \cdot M_R$, which yields the theorem. \square

6. INTERACTION WITH TCP

In this section we study the case in which the link is shared by an *ACC* connection and a regular TCP Reno connec-

tion. We analyze the relative fairness of the *ACC* algorithm. We assume that the TCP Reno connection is in *congestion avoidance phase* (steady state of TCP) having the increase term of 1 and the decrease factor of 2.

THEOREM 6.1. *The fairness constant of the system with an $ACC(\phi, \psi)$ connection and a TCP Reno connection is at least $C_F \geq \Omega(B - \log_\phi B)$.*

PROOF. (Sketch) The worst case scenario with respect to fairness is as follows. The same sequence of window sizes is repeated in all phases. Each increase sub-phase finishes when the maximum unfairness is reached and the following decrease sub-phase continues just one round. Let w_a and w_r be the window of the $ACC(\phi, \psi)$ connection and the window of the TCP Reno connection at the end of the increase sub-phase, respectively. Note that the window sizes must satisfy $\log_\phi(w_a(1 - 1/\psi)) = w_r/2$ and $w_a + w_r \leq \phi B$. It is easy to see that $\max(w_a - w_r) = \Omega(B - \log_\phi B)$. \square

Thus, an *ACC* connection could consume a larger fraction of the available bandwidth when competing with a regular TCP connection.

7. CONCLUDING REMARKS

In this paper we propose a novel end-to-end congestion control algorithm, based on *adaptive* additive increase/multiplicative decrease scheme. The proposed algorithm attains almost optimal utilization in a steady state providing fairness between competing connections and at the same time responds quickly on bandwidth changes. Even though our algorithm cannot be implemented without additional work, we believe that the proposed scheme may suggest new directions for further improvement of the current TCP congestion control.

Some challenging future directions are making the algorithm TCP-compatible, studying more complex network topologies and performing simulations with real traffic to find the optimal *ACC* parameters. Possible extensions of our model are: connections receiving distinct feedbacks, connections having different RTT's and a fluid model rather than discrete rounds.

8. REFERENCES

- [1] S. Arora and W. Brinkman, "A Randomized Online Algorithm for Bandwidth Utilization," *In Proceedings ACM-SIAM SODA*, 2002.
- [2] A. Bar-Noy, Y. Mansour and B. Schieber, "Competitive Dynamic Bandwidth Allocation," *In Proceedings of the 17th Annual ACM Symposium on Principles of Distributed Computing*, 1998.
- [3] D. Bansal and H. Balakrishnan, "Binomial Congestion Control Algorithms," *In Proceedings of IEEE INFOCOM*, Apr. 2001.
- [4] D. Bansal, H. Balakrishnan, S. Floyd and Scott Shenker, "Dynamic Behavior of Slowly Responsive Congestion Control Algorithms," *In Proceedings of ACM SIGCOMM*, Sep. 2001.

- [5] D. M. Chiu and R. Jain, "Analysis of the increase and decrease algorithms for congestion avoidance in computer networks," *Computer Networks and ISDN Systems*, 17, 1989.
- [6] S. Floyd and K. Fall, "Promoting the Use of End-to-end Congestion Control in the Internet," *IEEE/ACM Transactions on Networking*, Aug. 1999.
- [7] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-Based Congestion Control for Unicast Applications," *In Proceedings of ACM SIGCOMM*, Aug. 2000.
- [8] S. Floyd, S. Ratnasamy, and S. Shenker, "Modifying TCP's Congestion Control for High Speeds," *Internet draft*, May 2002.
- [9] N. Garg and N. E. Young, "On-Line End-to-End Congestion Control," *Proceedings IEEE Symp. Foundations of Computer Science*, 2002.
- [10] V. Jacobson, "Congestion Avoidance and Control," *In Proceedings of ACM SIGCOMM*, 1988.
- [11] S. Jin, L. Guo, I. Matta and A. Bestavros, "TCP-friendly SIMD Congestion Control and Its Convergence Behavior," *In Proceedings of The 9th IEEE International Conference on Network Protocols*, Riverside, CA, November 2001.
- [12] R. M. Karp, E. Koutsoupias, C. H. Papadimitriou and S. Shenker "Algorithmic problems in congestion control," *In Proceedings of FOCS*, 2000.
- [13] D. Katabi, M. Handley, and C. Rohrs, "Internet congestion control for future high bandwidth-delay product environments," *In Proceedings of ACM Sigcomm*, 2002.
- [14] F. P. Kelly, "Mathematical modelling of the Internet," *In Proceedings of 4th International Congress on Industrial and Applied Mathematics*, July 1999.
- [15] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan, "Rate control in communication networks: shadow prices, proportional fairness and stability," *Journal of the Operational Research Society*, 49, pp. 237-252, 1998.
- [16] S. H. Low, F. Paganini and J. C. Doyle, "Internet Congestion Control," *IEEE Control Systems Magazine*, 22(1), pp. 28-43, Feb. 2002.
- [17] S. H. Low, "A duality model of TCP flow controls," *In Proceedings of ITC Specialist Seminar on IP Traffic Measurement, Modeling and Management*, September 18-20 2000.
- [18] S. H. Low, L. Peterson, and L. Wang, "Understanding Vegas: a duality model," *In Proceedings of ACM Sigmetrics*, June 2001.
- [19] J. Mahdavi and S. Floyd, "TCP-Friendly Unicast Rate-Based Flow Control," *Available from <http://www.psc.edu/networking/papers/tcp-friendly.html>*, Jan. 1997.
- [20] A. Maor and Y. Mansour, "AdaVegas: Adaptive Control for TCP Vegas," *Manuscript*.
- [21] A. Piccolboni and C. Schindelhauer, "Discrete Prediction Games with arbitrary Feedback and Loss," *Proceedings of COLT*, 2001.
- [22] S. Shenker, "A Theoretical Analysis of Feedback Flow Control," *In Proceedings of ACM SIGCOMM*, pp. 156-165, 1990.
- [23] Y. R. Yang and S. S. Lam, "General AIMD Congestion Control," *ICNP 2000*, pp. 187-198.