ARTICLE IN PRESS

Computer Networks xxx (2008) xxx–xxx

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

On the properties of an additive increase rate accelerator

Ioannis Psaras *, Vassilis Tsaoussidis

Democritus University of Thrace, Department of Electrical and Computer Engineering, 12 Vas. Sofias Str., 67100 Xanthi, Greece

ARTICLE INFO

Article history:
Received 25 March 2008
Received in revised form 6 October 2008
Accepted 17 October 2008
Available online xxxx

Responsible Editor: J. Zhang

Keywords:
TCP
Congestion control
Transmission scheduling
Flow contention
AIMD
Adaptive AIMD
Adaptive transmission

ABSTRACT

We propose AIRA, an Additive Increase Rate Accelerator. AIRA extends AIMD functionality towards adaptive increase rates, depending on the level of network contention and bandwidth availability. In this context, acceleration grows when resource availability is detected by goodput/throughput measurements and slows down when increased throughput does not translate into increased goodput as well. Thus, the gap between throughput and goodput determines the behavior of the rate accelerator.

We study the properties of the extended model and propose, based on analysis and simulation, appropriate rate decrease and increase rules. Furthermore, we study conditional rules to guarantee operational success even in the presence of symptomatic, extra-ordinary events. We show that analytical rules can be derived for accelerating, either positively or negatively, the increase rate of AIMD in accordance with network dynamics. Indeed, we find that the "blind", fixed Additive Increase rule can become an obstacle for the performance of TCP, especially when contention increases. Instead, sophisticated, contention-aware additive increase rates may preserve system stability and reduce retransmission effort, without reducing the goodput performance of TCP.

1. Introduction

We introduce a new paradigm for the responsive behavior of flows when bandwidth availability changes due to varying network contention. We call this paradigm, Additive Increase Rate Accelerator (AIRA) [1], to reflect its operational perspective, namely to adjust the transmission rate of flows to current conditions of network contention. More precisely, we propose an algorithm, which progressively adjusts the Additive Increase factor of AIMD according to the current level of network contention. Typical systems adjust their rate, inevitably, only when contention leads to congestion; then, timeouts and multiplicative decreases force flows to reduce their windows. However, the transmission policy itself (i.e., increase/decrease rules) is not adjusted according to network contention. That is, although systems are adaptive to network dynamics, this adaptivity is limited: window size can be regulated but the window increase rate (i.e., Additive Increase factor) cannot. This is similar to a car regulating its velocity scale but with fixed acceleration.

By and large, the impact of contention on network dynamics, as well as the need for adjusting network parameters accordingly, is known to the networking community. Indeed, the conclusion is straightforward if we consider that the aggregate system rate increase is different in a system with 2 or 200 flows, which increases with a fixed rate of 1 packet per window for every successfully-acknowledged window. Consequently, our sample system with the same fixed increase/decrease parameters exhibits significantly different properties: it may reach stability or fail; it may exploit bandwidth rapidly or waste available resources; it may require major retransmission effort or minimize overhead.

In this context, we rely on two major concepts, to move beyond the confined perspective of predetermined and inflexible network parameters: (i) effort-based contention estimation and (ii) contention-oriented adaptive Additive Increase transmission. Effort, which is expressed as the ratio of throughput over goodput, reflects the efficiency of

* Corresponding author. Tel.: +30 25410 79953.
E-mail addresses: ipsaras@ee.duth.gr (I. Psaras), vtsaousi@ee.duth.gr (V. Tsaoussidis).

1389-1286/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2008.10.019

the transmission strategy; when both throughput and goodput increase, bandwidth availability is clearly indicated, otherwise, as their gap widens, transmission effort is wasted. Consequently, increased contention will be reflected by higher protocol effort. In simple terms, a protocol that monitors transmission effort could approximate the dynamics of contention. That said, responses can be triggered accordingly. We investigate precisely the responsive strategies that correspond to various contention dynamics.

The proposed paradigm requires a number of issues to be addressed, prior to deployment. How accurately can we estimate changes of contention? What is practically the gain from a hypothetical transition to AIRA paradigm? What is the adaptation scheme of choice to varying contention? Are the properties of stability and fairness violated in favor of efficiency?

We address the aforementioned issues based on analysis and simulation experiments. In particular, we show that (i) contention can be estimated coarsely; coarse estimation suffices to enhance system properties; (ii) system Fairness and Stability can be preserved if we retain the Multiplicative Decrease response to congestion; and (iii) system efficiency can be increased if we adopt an adaptive Additive Increase response to available bandwidth. Adaptive increase allows for more sophisticated utilization of bandwidth, through more aggressive increase, when contention is low and less retransmission effort when contention is high. Furthermore, we go beyond conclusive statements to investigate the particular responsive strategy that corresponds to varying contention.

During our investigation, we uncovered a number of dynamics associated with heterogeneous RTTs, as well as with long- and short-lived flows. We show that adaptive increase favors short-lived flows, which is a desirable property if we consider a utilization-oriented (i.e., time-oriented) notion of fairness. Also, flows that experience long propagation delays tend to decrease their rate slower than flows that experience short propagation delays, which is another desirable property. Therefore, our proposal demonstrates high potential for deployment.

However, a number of issues still remain open. For example, we do not discuss, here, the impact of the granularity of measurements and its association with the dynamics of network changes. Also, although we show that flows with different increase rates eventually converge to the same, approximately, contention-aware rate acceleration scheme, we do not report here the convergence properties of such scenarios. Finally, we do not present results with the whole spectrum of potential behaviors; for example, we do not investigate the impact of transmission schemes that are more aggressive than the typical TCP transmission strategy (e.g. [2–8])

We organize the remaining paper as follows: In Section 2, we discuss related studies; in Section 3, we justify our research perspective and motives; in Section 4, we define our system model, which includes definitions, observations on the dynamics of Additive Increase with diverse rates and proposed solution framework. We present AIRA in Section 5; we propose adjustments to the proposed algorithm, due to practical constraints in Section 6; we provide justification towards algorithm deployability in Section 7. Sections 5–7 include corresponding simulation results as well. In Section 8, we discuss open issues that need further investigation and, finally, in Section 9, we conclude the paper.

2. Background and related work

The *Transmission Control Protocol* (TCP) [9] is the most widely used protocol that incorporates AIMD [10] to control data transmission over the Internet. TCP operates under a closed-loop, binary-feedback policy, in order to guarantee reliable data transfer. The sending host uses the congestion window (*cwnd*) to confine the maximum number of in-flight bytes transmitted. Upon positive feedback (i.e., ACK arrival) the sender increases its *cwnd additively*:

$$cwnd \leftarrow cwnd + \frac{a}{cwnd} \quad (1)$$

while, negative feedback (i.e., duplicate ACKs), which is interpreted as network congestion, triggers *multiplicative cwnd* decrease:

$$cwnd \leftarrow cwnd - b * cwnd. \quad (2)$$

The goal of AIMD is twofold [10]:

(1) to utilize resources efficiently, which further means:
  (a) to avoid congestive collapse,
  (b) to continuously probe for available bandwidth.
(2) to converge to Fairness (i.e., allocate resources equally among participating flows).

Researchers have recently focused on the optimization of AIMD targeting either efficient or fast bandwidth exploitation (e.g. [6,7,11,4,2,5,12,8,3,13]) or convergence to fairness (e.g. [14–16]).

Moreover, the evolution of real-time Internet applications, such as audio and video streaming that require smooth transmission rate, have motivated research towards the suitability of AIMD for time-sensitive data transfer (e.g. [17,18]). A trade-off between smoothness and aggressiveness has been exploited in [17,18], since higher smoothness results in lower aggressiveness. For example, several recent TCP-friendly congestion control schemes like [19–22] achieve higher smoothness at the cost of responsiveness.

None of the above studies, however, have incorporated *the level of contention* as a decisive factor for rate adjustment. In this paper, we associate network stability with Retransmission Effort (i.e., not with congestive collapses). Our perspective is mainly justified by modern infrastructures and sophisticated protocols of today's Internet.

More precisely, we propose an adaptive Additive Increase rate, *a*, according to the level of network contention. For example, when contention increases *a* should decrease, in order to reduce the retransmission effort of the transport protocol; otherwise, when the number of flows decreases, the frequency of congestion events (per flow) decreases as well. We try to assess the cost of "blind" Additive Increase (i) on the performance of transport protocols (i.e., retransmission overhead/effort), and (ii) on network stability (i.e., frequency of congestion events or drastic transmission rate adjustments).

Please cite this article in press as: I. Psaras, V. Tsaoussidis, On the properties of an additive increase rate accelerator, Comput. Netw. (2008), doi:10.1016/j.comnet.2008.10.019

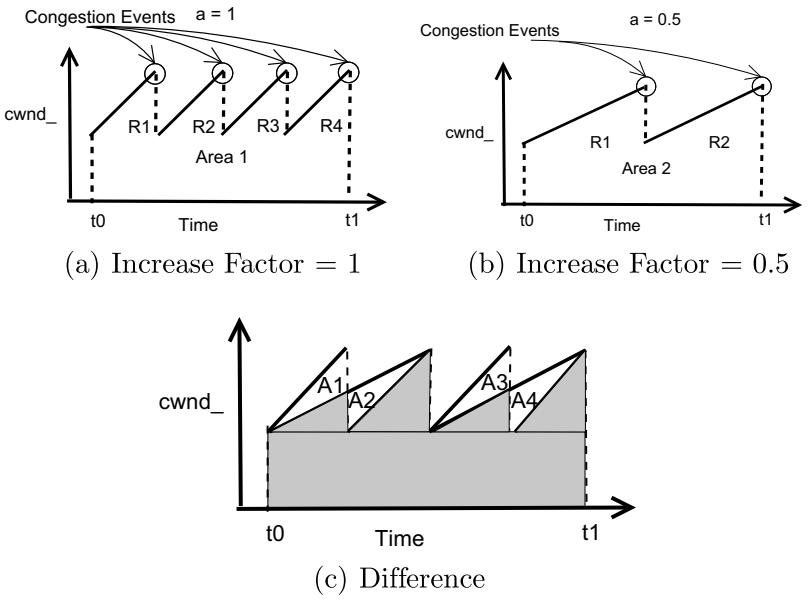


Fig. 1. Different increase factors.

3. Motivation

Deployment of AIMD is associated with two operational standards: (i) the fixed increase rate and decrease ratio and (ii) the corresponding selection of appropriate values.

Recent research has focused on altering the values for a and b (Eqs. (1) and (2)) but has not questioned really the validity and efficiency of fixed rates throughout the lifetime of participating flows. In this context, research efforts cannot address questions such as: Why do flows increase their rate by α packets instead of 2α packets, even when half users of a system leave and bandwidth becomes available?

One possible justification for not highlighting that research direction is that:

The Additive Increase factor of AIMD does not (practically) contribute to the long-term goodput performance of TCP.

In Fig. 1, we present the *cwnd* evolution for two TCP flavors: Fig. 1a, where $a = 1$ (regular TCP) and Fig. 1b, where $a = 0.5$. The area underneath the solid *cwnd* lineplot (Areas 1 and 2) represents the goodput[1] performance of the protocols. In Fig. 1c, we show that both protocols achieve the same goodput performance, since $A1 = A2$ and $A3 = A4$[2].

However,

Additive Increase affects significantly the Retransmission Effort of flows, which impacts overall system behavior, as well.

For example, TCP $a = 1$, in Fig. 1, experiences 4 congestion events, while TCP $a = 0.5$ experiences only 2. Assuming that each congestion event is associated with a fixed number of lost packets, regular TCP (i.e., $a = 1$) will retransmit twice as many packets as TCP with $a = 0.5$, without any gain in goodput performance.

We verify the above observations through simulations (using ns-2 [24]). We simulate 2 TCP-SACK [25] flows, for 200 seconds, over a single bottleneck dumbbell network topology (Fig. 2); the backbone link transmits 1 Mbps, its propagation delay is 20 ms and the Drop Tail Router has buffer capacity equal to 15 packets.

Clearly, there is a trade-off between *Aggressiveness* and *Retransmission Effort* (see Table 1). The degree of *Aggressiveness* that a transport protocol can achieve is tightly associated with its Retransmission Effort. The higher the Additive Increase factor, the more the retransmission effort of the transport protocol.

We repeat the above scenario with four participating flows to observe the goodput performance and the retransmission effort of the transport protocol (Table 1). We find that the level of contention severely impacts the retransmission overhead of the transport protocol. For example,

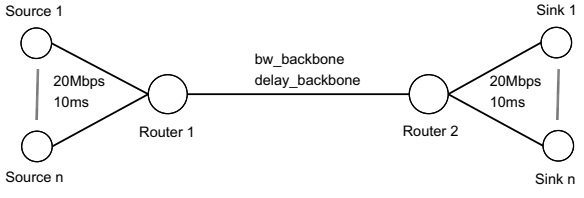


Fig. 2. Dumbbell network topology.

[1] We define the system Goodput as $\frac{Original\ Data}{Connection\ Time}$, where *Original Data* is the number of bytes delivered to the high level protocol at the receiver (i.e., excluding retransmissions and the TCP header overhead) and *Connection Time* is the amount of time required for the data delivery. Instead, system Throughput includes retransmitted packets and header overhead (i.e., $\frac{TotalData}{ConnectionTime}$).

[2] In Fig. 1c grey areas are common for both protocols; white areas are equal ($A1$ is similar to $A2$ and $A3$ is similar to $A4$). In fact, the solid *cwnd* lineplots in Fig. 1 do not represent precisely the evolution of the *cwnd* according to Eq. (1). That is, the evolution of the *cwnd*, according to Eq. (1) is not exactly linear. However, we consider the surface difference negligible and we assume for simplicity that the *cwnd* evolves linearly with time. We verify with simulations that the above assumption is indeed sound, in practice. We refer the reader to [23] for further analysis on that issue.

Please cite this article in press as: I. Psaras, V. Tsaoussidis, On the properties of an additive increase rate accelerator, Comput. Netw. (2008), doi:10.1016/j.comnet.2008.10.019

Table 1
TCP performance – different increase factors.

	Goodput	Retransmissions	
	2/4 flows	2 flows	4 flows
$a = 2$	118.9 KB/s	742 pkts	1653 pkts
$a = 1.5$	118.9 KB/s	548 pkts	1777 pkts
$a = 1$	118.8 KB/s	278 pkts	704 pkts
$a = 0.5$	118.9 KB/s	172 pkts	452 pkts
$a = 0.2$	118.9 KB/s	86 pkts	229 pkts
Delayed ACKs	118.6 KB/s	163 pkts	515 pkts

we see in Table 1 that doubling the network load (i.e., 100% increase), results approximately in 160% increase of the retransmission overhead. In all cases, we observe no goodput performance gain.

Note that further increasing the level of contention may even degrade the system goodput performance, due to timeout expirations [26–28], which are not considered in Fig. 1.

Corollary 1. *From a point onwards, when contention increases, fixed Additive Increase causes more retransmissions, with zero gains in system goodput.*

4. System model

The initial study that investigated the operational properties of AIMD is [10]. In that study, the authors assume a feedback model, where all flows become aware of congestion events synchronously. In the current study, we extend this model to reflect more realistic situations. For example, in our model, different flows may become aware of congestion events at different points in time (i.e., congestion feedback is received asynchronously). A synchronous model, inherently assumes that flows do not experience queuing delays and hence, the duration of a *Round* (which is defined here as the interval between two *cwnd* multiplicative decreases) is in-varying for all flows. Instead, we allow for the possibility of queuing delays, which further means that the duration of a *Round* may differ among flows. Finally, we also allow for the possibility of multiple packet losses at the end of a *Round*.

4.1. Definitions

We define the following terms:

(1) A *step s* is delimited by the *cwnd* update function, each time *cwnd* number of packets are successfully delivered to the receiver.
(2) A *Round* is defined as the interval between two *cwnd* multiplicative decreases.
(3) *Round Loss Rate* (p_i) is the ratio of the lost packets over the total number of sent packets, within *Round i*. The *Round Loss Rate* is calculated at the end of each *Round*.
(4) The *Throughput Slope* within a Round is defined as $\frac{a}{cwnd}$. Obviously, the *Throughput Slope* is identical to the *cwnd Slope* (see Fig. 3).

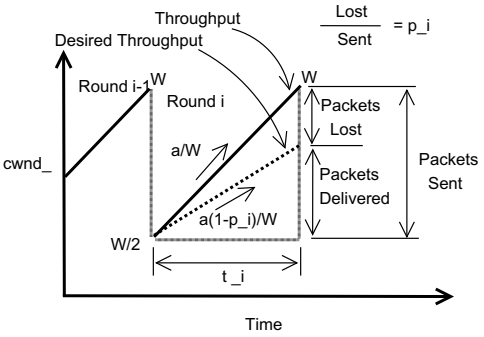


Fig. 3. Throughput/Desired Throughput Slopes at round *i*.

(5) Assuming a *Round Loss Rate* p_i and a *Round duration* t_i, the *Desired Throughput Slope* within a Round is defined as the *hypothetical cwnd Slope*, which would result in zero packet losses, within t_i, but without causing bandwidth underutilization. The *Desired Throughput Slope* is, therefore, determined by $\frac{a}{cwnd} \cdot (1 - p_i)$ (see Fig. 3).

We explicitly note that all of the above definitions apply to individual flows and that none of them is descriptive of the system as a whole.

For ease of illustration, we summarize, in Table 2, the symbols used throughout the rest of the paper.

4.2. Observations on the dynamics of additive increase with diverse rates

(1) When $p_i > p_{i-1}$ then the *Throughput Slope* exceeds the *Desired Throughput Slope*, for *Round i*.
- The greater the distance between p_i and p_{i-1}, the wider the gap between the *Throughput* and *Desired Throughput Slopes* (i.e., the protocol is too aggressive).

(2) When $p_i < p_{i-1}$ then the *Throughput Slope* is underneath the *Desired Throughput Slope*, for *Round i*.
- The greater the distance between p_i and p_{i-1}, the wider the gap between the *Desired Throughput* and *Throughput Slopes* (i.e., the protocol is too conservative).

Hence, our primary objective is to reduce the gap between the *Throughput* and *Desired Throughput Slopes*, in order to avoid extensive retransmission effort, or bandwidth under-utilization, respectively.

Table 2
Algorithm symbols.

Symbol	Meaning
i	*Round* number
p_i	*Loss rate* at *Round i*
a_i	Additive increase factor at *Round i*
s	*step*
W	*cwnd* value
$W(s)$	*cwnd* value at *step s*
t_i	Duration of *Round i*

Please cite this article in press as: I. Psaras, V. Tsaoussidis, On the properties of an additive increase rate accelerator, Comput. Netw. (2008), doi:10.1016/j.comnet.2008.10.019

4.3. Solution framework

We provide a solution framework in order to determine the primary requirements of our system model. We require from our system to:

(1) *Converge to Fairness.*
- We evaluate the fairness properties of a protocol using the Fairness Index introduced in [10]:

$$Fairness = \frac{\sum(Throughput_i)^2}{n\sum(Throughput_i^2)}, \quad (3)$$

where $Throughput_i$ is the Throughput performance of the i^{th} flow and n is the number of participating flows.
- We introduce the *ALPHA Fairness Index* of multi-rate systems. We define *ALPHA Fairness Index* as

$$ALPHA\ Fairness = \frac{\sum(a_i)^2}{n\sum(a_i^2)}, \quad (4)$$

where a_i is the Additive Increase factor of the i^{th} flow and n is the number of participating flows. The ALPHA Fairness Index has the following properties: In a homogeneous/synchronous static (contention-wise) system, increased ALPHA Fairness Index leads to increased System Fairness Index as well. In a heterogeneous/asynchronous dynamic system, however, the properties of ALPHA Fairness Index are not straightforward. For example, in a diverse-RTT system, reduced ALPHA Fairness Index may correspond to either increased or decreased system Fairness (i.e., reduced Additive Increase factor for longer-RTT flows results in reduced Fairness, while increased Additive Increase factor for longer-RTT flows results in increased system Fairness). We explore the above properties through simulations in the following sections.

(2) *Guarantee Stability.Stability* is a quality measure, which we attempt to simplify and furthermore quantify by measuring the frequency of congestion events. Therefore, a protocol is said to achieve higher *Stability*, when it minimizes retransmission overhead.
(3) *Exploit available resources efficiently*, which means:
- *faster* (i.e., aggressively) when contention is low (i.e., utilize high percentage of available bandwidth).
- *slower* (i.e., conservatively) when contention is high (i.e., minimize overhead/transmission effort).

Efficiency is achieved through high resource utilization (i.e., high goodput performance) *and* minimal retransmission overhead.

We note that the proposed algorithm is not designed for high-speed environments. That is, the operational rules of the *Additive Increase Rate Accelerator* (see Section 5) are not capable of exploiting the available resources (bandwidth) of high-speed links fastly. Instead, as we show in the following Sections, the benefits of the proposed algorithm are clear in case of conventional Internet links, where the level of flow contention is relatively high.

5. AIRA: Additive Increase Rate Accelerator

We apply a Decrease and an Increase rule to regulate the amount of data that a flow can insert into the network. The selection of the appropriate rule depends on the *Round Loss Rate* that the flow experiences (see Fig. 3). In particular, each flow adjusts its rate at the end of each *Round*, which is indicated by packet loss(es). The rate adjustment at the end of each round applies at the *next Round* (i.e., the *Round* that is about to begin). The flow at the end of *Round i* can determine the rate for round $i+1$ exploiting, recursively, the behavior of the rate during *Rounds* i and $i-1$. In particular, the Decrease rule applies when $p_i > p_{i-1}$.

5.1. The Decrease rule

Decrease Rule. In order to reduce the gap between the Throughput and Desired Throughput Slopes, the Additive Increase factor should decrease, according to equation:

$$a_{i+1} = a_i \cdot (1 - p_i). \quad (5)$$

Justification. Eq. (1) reveals that the slope of the *cwnd*, within a round, is $\frac{a}{W(s-1)}$, where s is the *step*. Considering the Round Loss Rate of the previous round, the slope of the Desired Throughput can be approximated by λ, where $\lambda < \frac{a}{cwnd}$ (see Fig. 4).

From Fig. 4 we derive that:

$$Throughput\ Slope = tan(throughput) = \frac{k}{t_i} \quad (6)$$

and

$$DesThr\ Slope = tan(DesThr) = \frac{k-m}{t_i}, \quad (7)$$

where k is the number of packets sent and m is the number of packets *lost* during *Round i*, respectively (see Fig. 3).

Dividing Eqs. (6) and (7) by parts, we get

$$\frac{tan(thr)}{tan(DesThr)} = \frac{k}{k-m}. \quad (8)$$

From Eq. (1) we know that:

$$tan(thr) = \frac{a}{cwnd} = \frac{a_i}{W(s-1)}. \quad (9)$$

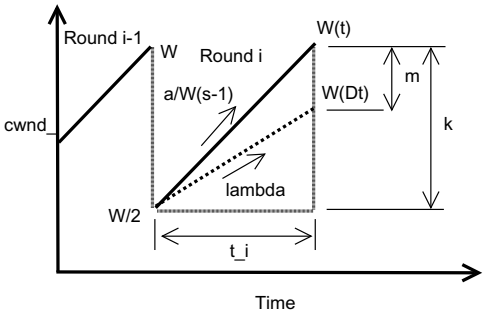


Fig. 4. Throughput/Desired Throughput (Dt) Slopes of the *cwnd* at Round *i*.

Please cite this article in press as: I. Psaras, V. Tsaoussidis, On the properties of an additive increase rate accelerator, Comput. Netw. (2008), doi:10.1016/j.comnet.2008.10.019

From Eqs. (8) and (9) we get

$$tan(DesThr) = \frac{a_i \cdot (k-m)}{W(s-1)\cdot k}. \quad (10)$$

From Fig. 4 we derive the values for k and m as follows:

$$k = \frac{W(s)}{2}, \quad m = \frac{W(s)}{2}\cdot p_i, \quad (11)$$

From Eqs. (10), and (11) we get that:

$$tan(DesThr) = \frac{a_i \cdot W(s)\cdot (1-p_i)}{W(s-1)\cdot W(s)} = \frac{a_i\cdot(1-p_i)}{W(s-1)} \quad (12)$$

or

$$\lambda_{decrease} = \frac{a_i\cdot(1-p_i)}{W(s-1)}. \quad (13)$$

Hence, the Additive Increase factor for round $i+1$, is adjusted according to Eq. (5) (see Fig. 5).

5.2. The Increase rule

According to the *Decrease rule*, the Additive Increase factor of TCP (i.e., AIRA) may get non-increasing values. In that case, however, the system may never reach equilibrium; flows already existing in the system, possibly transmitting with $a < 1$, will not have the opportunity to compete (fairly) with new, incoming flows. Therefore, we introduce an *Increase rule*, which applies when $p_i < p_{i-1}$.

Increase Rule. In order to reduce the gap between the Desired Throughput–Throughput Slopes, the Additive Increase factor should increase, according to equation:

$$a_{i+1} = a_i \cdot (1 + p_{i-1} - p_i). \quad (14)$$

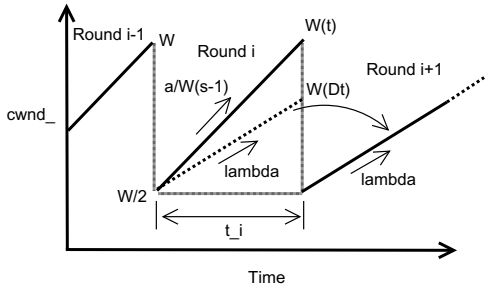


Fig. 5. *cwnd* Slope at Round $i+1$.

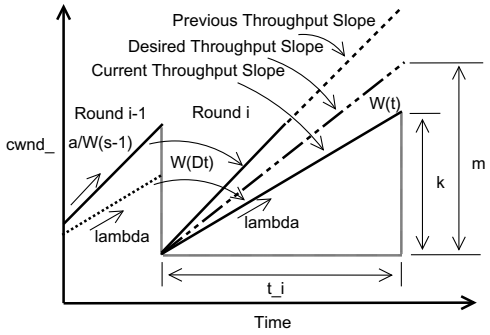


Fig. 6. Throughput/Desired Throughput Slopes of the *cwnd* at Round *i*.

Justification.

From Fig. 6 we get the *Current Throughput Slope*:

$$Throughput\ Slope = tan(throughput) = \frac{k}{t_i} \quad (15)$$

and the *Desired Throughput Slope* (DesThr):

$$DesThrSlope = tan(DesThr) = \frac{m}{t_i}. \quad (16)$$

Dividing Eqs. (15) and (16) by parts, we get

$$\frac{tan(thr)}{tan(DesThr)} = \frac{k}{m}. \quad (17)$$

From Eqs. (9) and (17) we get

$$tan(DesThr) = \frac{a_i\cdot m}{W(s-1)\cdot k}. \quad (18)$$

From Fig. 6 we derive the values for k and m as follows:

$$k = \frac{W(s)}{2}, \quad m = \frac{W(s)}{2} + \frac{W(s)}{2}\cdot\phi, \quad (19)$$

where ϕ depicts the angle difference between the *Current* and the *Desired Throughput Slopes*. From Eqs. (18) and (19) we get that

$$\lambda_{increase} = \frac{a_i\cdot(1+\phi)}{W(s-1)}. \quad (20)$$

For the purpose of the present study, we set ϕ to be the difference between the loss rate experienced in the previous round and the one experienced at the end of the current *Round*:

$$\phi = p_{i-1} - p_i. \quad (21)$$

Hence, in round $i+1$ the Additive Increase factor is adjusted according to Eq. (14) (see Fig. 7). Finally, we note that in case $p_i = p_{i-1}$, no adjustments take place (i.e., $a_i = a_{i-1}$).

5.3. Implementation details

The implementation of the proposed algorithm is a rather easy task; the implementation requires modification of two functions only, within the TCP sender's source code. Probably, the only non-trivial part of the algorithm's implementation is the *Round Loss Rate* calculation. In the present Section, we clarify this issue first and then, we pro-

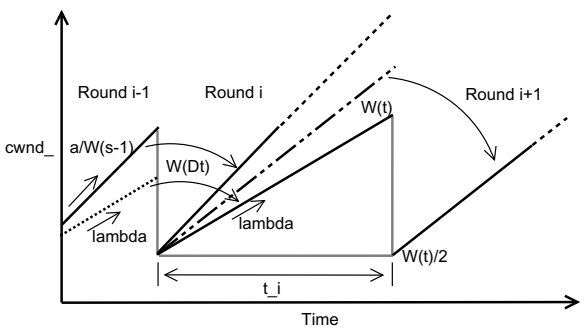


Fig. 7. *cwnd* at Round $i+1$.

Please cite this article in press as: I. Psaras, V. Tsaoussidis, On the properties of an additive increase rate accelerator, Comput. Netw. (2008), doi:10.1016/j.comnet.2008.10.019

vide the pseudocode of the proposed algorithm together with its flow diagram.

The *Round Loss Rate* calculation takes place, as already mentioned before, at the end of each *Round* and the outcome of the calculation, which triggers either the *Decrease* or the *Increase Rule*, applies to the next *Round*. The sender keeps one variable, called `snt_pkts`, which counts the total number of packets sent from the beginning of the *Round*. Upon a packet loss event, the structural properties of the *Selective ACK* version of TCP allow for identification of the packet(s) that need to be retransmitted. The sender stores the number of packets that are (probably) lost in one extra variable, called `lost_pkts`. These packets, however, are the packets lost during the current *Round only*, since otherwise (i.e., the lost packets belong to the previous *Round*), the sender would have timed-out. At that point, the sender calculates the *Round Loss Rate* as

$$Round_i LossRate = p_i = \frac{lost\_pktsatRound_i}{snt\_pktsatRound_i}. \quad (22)$$

This way, AIRA can capture more packet losses within a round, independently of the fact that TCP-SACK halves the window once, even if more losses happen within a single window of data.

We provide the pseudocode of the proposed algorithm below and its flow-diagram in Fig. 8.

```
initial a = 1
initial loss_rate = 0
initial loss_rate_l = 0// loss rate of the pre-
vious round
/* open cwnd function */
lambda_ = a_i/cwnd_
cwnd_ = cwnd_ + lambda_
/* upon loss or triple-duplicate ack */
/* close cwnd_function */
cwnd_ = cwnd_ *(1 − b)// b = 0.5
loss_rate = number of packets lost at the end of
the round/total number of packets transmitted
within this round
if (loss_rate_l<loss_rate)
then apply the decrease rule:
a_i + 1 = a_i*(1 − loss_rate)
else apply the increase rule:
a_i + 1 = a_i*(1 + [loss_rate_l - loss_rate])
```

5.4. Results: Decrease/Increase rules

We present simulation results to depict the operational properties of the Decrease/Increase rules of AIRA. We use a contention increase scenario in order to monitor rate fluctuation in case of dynamic network contention. Four flows participate in the experiment, each of which enters the system 50 s after the previous flow. The backbone link carries 2 Mbps with 20 ms propagation delay (Fig. 2); the buffer of Router 1 holds 10 packets.

In Fig. 9, we present the Additive Increase factor of the participating flows during the simulation. Initially, the

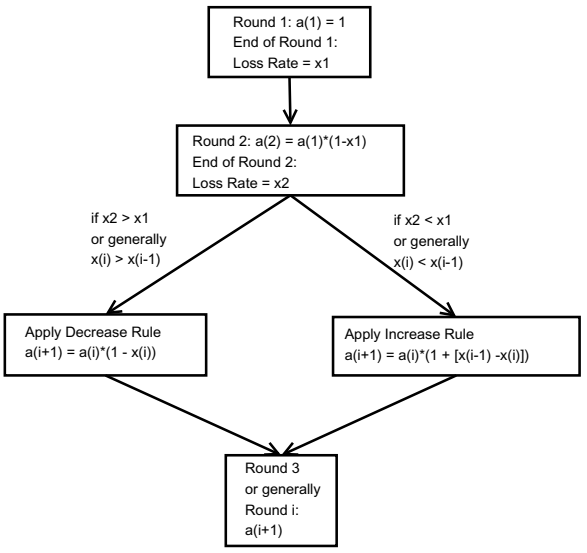


Fig. 8. AIRA flow diagram.

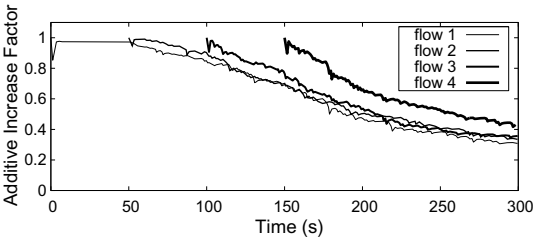
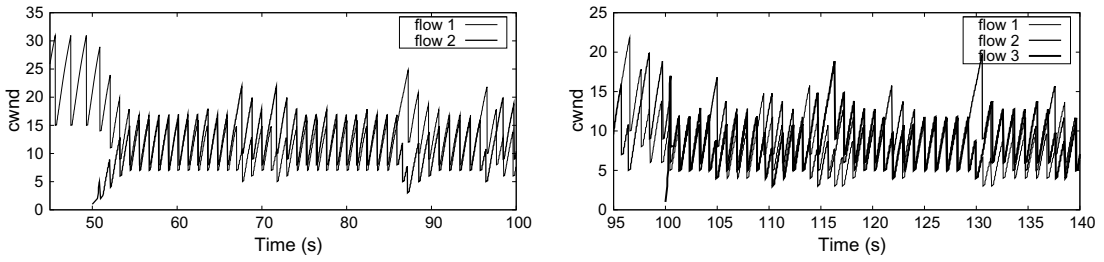


Fig. 9. Additive increase factor.

Additive Increase factor of the first flow stabilizes to a value very close to 1, since the *Loss Rate* at that time has a very low and fixed value. As contention increases, the *Loss Rate* increases as well, causing gradual reduction of the Additive Increase factor. Note, however, that the rate reduction per flow varies: new flows, which operate with greater Additive Increase factors, experience greater rate reduction. Moreover, in Figs. 10 and 11, we present the *cwnd* evolution of AIRA and AIMD when the second and third flow enter the system. We notice that: (i) the incoming AIRA flows exploit transmission opportunities as fast as regular AIMD flows do, (ii) the AIRA system may *only temporarily* experience un-fairness (e.g. 86 – –88th second in Fig. 10a) and (iii) AIRA breaks flow synchronization. Flow de-synchronization, in turn, increases system fairness, as we have also shown in [27]. This is further verified by the results presented in Table 3, where we see that AIRA increases system Goodput, Fairness and Stability. Furthermore, convergence to system Fairness is guaranteed by the Multiplicative Decrease response of AIMD. Hence, un-fairness side-effects are canceled by (i) the Multiplicative Decrease response to congestion and (ii) AIRA's inherent properties of de-synchronization.

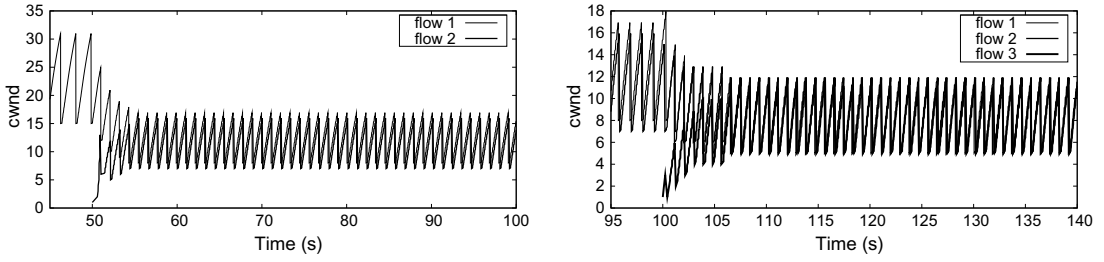
We extend the above scenario to include more flows. The bandwidth of the backbone link is now 20 Mbps and the buffer capacity at Router 1 is 100 packets. Along

Please cite this article in press as: I. Psaras, V. Tsaoussidis, On the properties of an additive increase rate accelerator, Comput. Netw. (2008), doi:10.1016/j.comnet.2008.10.019



(a) Second Flow Enters the Link (b) Third Flow Enters the Link

Fig. 10. AIRA *cwnd* in contention increase scenario.



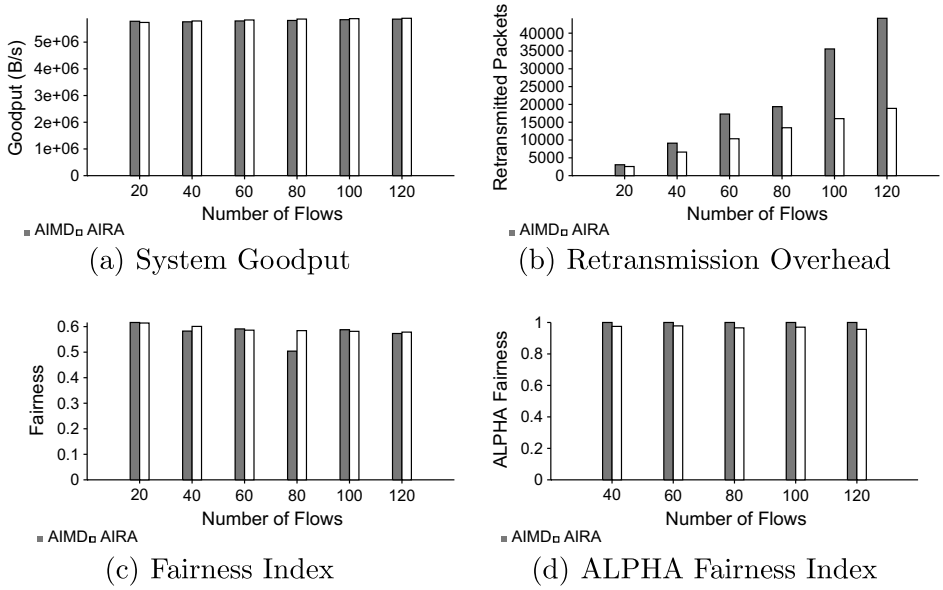
(a) Second Flow Enters the Link (b) Third Flow Enters the Link

Fig. 11. AIMD *cwnd* in contention increase scenario.

Table 3
Performance difference – contention increase scenario.

	Goodput	Retransmissions	Fairness	*a* Fairness
AIMD	222.3 KB/s	1065 pkts	0.8162	1
AIRA	227.1 KB/s	748 pkts	0.85	0.98

the lines of the previous scenario, flows are divided into quarters; each team enters the system 50 s after the previous one. The results are presented in Fig. 12. Again, we see that AIRA increases system Goodput, Stability and occasionally system Fairness. The ALPHA Fairness Index (Fig. 12d) reveals that in all cases the Additive Increase factor of all flows converges to (approximately) the same value.



(a) System Goodput (b) Retransmission Overhead

(c) Fairness Index (d) ALPHA Fairness Index

Fig. 12. Contention increase scenario.

Please cite this article in press as: I. Psaras, V. Tsaoussidis, On the properties of an additive increase rate accelerator, Comput. Netw. (2008), doi:10.1016/j.comnet.2008.10.019

6. Adjustments due to practical constraints

6.1. Lower bounds

We set lower bounds for AIRA for the following reason: Progressive reduction of the Additive Increase factor may result in transmission rate stabilization (i.e., $a = 0$). This, however, will inevitably cause system in-stability, protocol in-efficiency and flow starvation. We, therefore, bound a to 0.1, attempting to avoid the above un-desirable system property. That said, the greatest possible reduction of the Additive Increase factor is 0.9 (i.e., $a_{greatest\ reduction} = 0.9$, when $a = 0.1$).

Furthermore, we complement this absolute bound with another, dynamically-adjustable lower bound, which depends on the number of completed *Rounds* and is called *Minimum Additive Increase Limit (MAIL)*. The reason is twofold:

(1) A flow may lose several back-to-back packets, due to sudden traffic-bursts or symptomatic events. In order to limit drastic and systematic responses to symptomatic events, lower bounds need to be introduced. Otherwise, recovery from such symptomatic events may require significant effort and time or may even become impossible.
(2) When contention reaches extra-ordinary levels, AIRA lower bound (i.e., $a = 0.1$) dominates and the rest of AIRA functionality is practically suspended. In that case, MAIL will prevent short-flow starvation, adhering to a time-oriented notion of Fairness (e.g. a flow at the second *Round* operating with $a = 0.1$). That is, assuming that short[3] flows need to be favored over long, time-insensitive ones, *MAIL* is designed to provide more opportunities for data transmission to short, probably time-sensitive flows.

Minimum Additive Increase Limit (MAIL). We use the following equation to adjust MAIL:

$$MAIL = a_{min} = 1 - a_{greatest\ reduction} \cdot R_N\%, \quad (23)$$

where R_N is the number of completed *Rounds* and $a_{greatest\ reduction} = 0.9$ (see Fig. 13).

In Fig. 13, we demonstrate how MAIL evolves in time (i.e., *Rounds*). The duration of a *Round*, however, depends on the path RTT as well as on the link speed. We simulate a single AIRA flow over various link speeds and propagation delay paths, in order to observe: (i) the amount of data transferred as the number of *Rounds* increases and (ii) the number of completed *Rounds* as time elapses. Without loss of generality, we consider a 5 MBytes file size as an experimental threshold between short and long flows. We see in Fig. 14 that in all cases 5 MBytes are transferred within the first 50 *Rounds* (Fig. 14a), which correspond to less than 50 s (Fig. 14b). Hence, MAIL allows for increased transmission rates for short flows and slower transmission rates for longer ones (i.e., a may equal to 0.1 only after completion of 10 MB transfers).

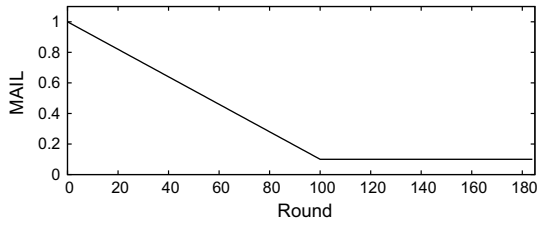


Fig. 13. Minimum Additive Increase Limit (MAIL).

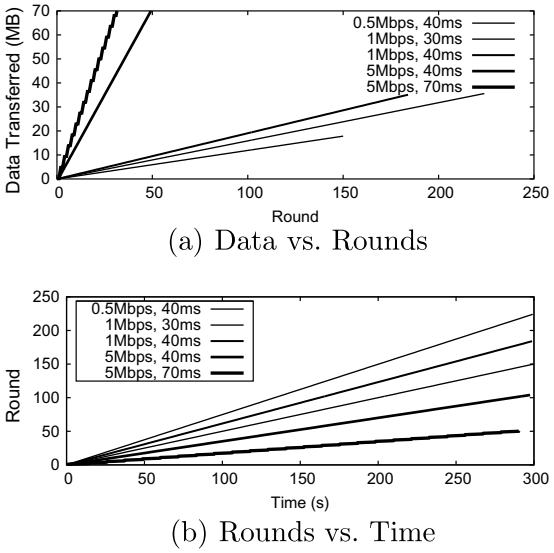


Fig. 14. Correlations between file size, number of *Rounds* and duration.

Secondary Effect. MAIL exhibits one desirable property, which was not an initial design goal: it favors long-RTT flows over shorter-RTT ones. That is, short-RTT flows complete a *Round* faster than long-RTT flows. Therefore, MAIL will get a lower value (i.e., Additive Increase factor) for a short-RTT flow faster than for a longer-RTT one. Hence, in case of moderated or high contention the short-RTT flows will operate with lower Additive Increase factors, avoiding this way starvation of long-RTT flows.

6.2. Results: Lower bounds

We verify the above hypothesis by simulation. Initially, we simulate 4 flows over the "Diverse-RTT Network Topology" (Fig. 15). We use Drop Tail routers, with buffer sizes equal to the Bandwidth-Delay Product of the outgoing links. We note that results are similar in case of Active Queue Management schemes, like RED [29], for example. The round trip propagation delay for flows 1 and 2 is 60 ms, while for flows 3 and 4 is 220 ms. Simulation time is 300 s. Fig. 16 depicts the Additive Increase factor for each

[3] We consider short flows as flows that carry files smaller than 5 MBytes (e.g. large pdf documents, presentations or software updates). Web flows, on the contrary, which are considered as short flows in the related literature, are not of interest in the present study, since such flows will finish their task either in the Slow-Start phase or within the first rounds, where the Additive Increase factor is still very close (if not equal) to 1 (i.e., a content-rich web page is rarely larger than 100KB).

Please cite this article in press as: I. Psaras, V. Tsaoussidis, On the properties of an additive increase rate accelerator, Comput. Netw. (2008), doi:10.1016/j.comnet.2008.10.019

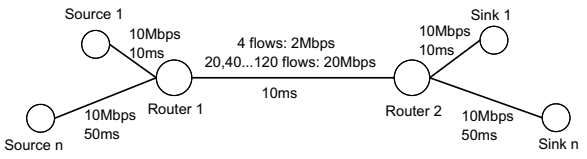


Fig. 15. Diverse-RTT network topology.

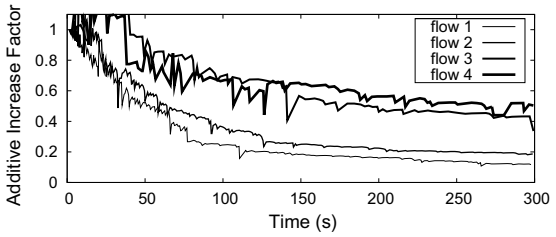


Fig. 16. Additive increase factor.

Table 4
Performance difference – RTT un-fairness

	Goodput	Retransmissions	Fairness	a Fairness
AIMD	200.6 KB/s	2009 pkts	0.6594	1.0
AIRA	209.5 KB/s	943 pkts	0.8768	0.787

flow. We see that MAIL provides more transmission opportunities to long-RTT flows (see Fairness Index in Table 4).

We extend the above scenario to include more flows. The results are presented in Fig. 17. In all cases (i.e., 4-flow scenario, Table 4 and extended scenario, Fig. 17), we see that the proposed algorithm improves *Stability* (through far less retransmissions) and *Efficiency* (through less retransmissions and slightly increased goodput). As contention increases, we notice considerable decrease of the *ALPHA Fairness Index*, which indicates more fair resource allocation among short and long-RTT flows, when AIRA is used[4].

6.3. Response to contention decrease

Although the AIRA *a Increase rule* (Section 5) proves to operate efficiently in static as well as in contention-increase scenarios, it fails to exploit extra available resources when contention decreases (i.e., ϕ is a small value, incapable of accelerating a fast enough, see Eq. (21)). Inline with our Solution Framework (Section 4.3), we apply a *Reset Condition* to AIRA, in order to prevent bandwidth wastage in case of contention decrease.

AIRA Reset Condition. Assuming that a flow's *cwnd* oscillates between $W/2$ and W before the contention decrease event, the flow will have the opportunity to expand its *cwnd* to $W+\frac{1}{2}W=\frac{3}{2}W$ after $\frac{1}{3}$ of the participating flows leave the system. When this happens, AIRA resets a to 1[5].

Analysis. A flow's window expansion speed depends on its Additive Increase factor. The greater the Additive Increase factor, the faster the bandwidth will be exploited. Given a *Round Duration* R_{Da}, for a flow with Additive Increase factor equal to a, the flow will need $0.3\cdot R_{Da}$ to expand its window to $W+\frac{1}{2}W$, according to *AIRA Reset Condition*. Provided that the fastest possible window expansion speed is achieved when $a=1$, a general rule that captures the extra window expansion time needed by a flow with smaller a is given by the following equation:

$$a_x : T_x = 0.3\cdot R_{Dx} = 0.3\cdot\frac{a_1}{a_x}\cdot R_{D1}, \tag{24}$$

where x is the Additive Increase factor (i.e., $a_{0.6}=0.6$). We plot T_x in Fig. 18a. Obviously, a flow operating with $a_x<a_1$ will complete a *Round* later than the a_1 flow (i.e., it will need $\frac{a_1}{a_x}\cdot R_{D1}$ extra *Rounds*). Hence, the total time needed by a flow with $a_x<a_1$ is

$$a_x : TotalT_x = 1.3\cdot R_{Dx} = 1.3\cdot\frac{a_1}{a_x}\cdot R_{D1}. \tag{25}$$

We plot $TotalT_x$ in Fig. 18b. Note that, since MAIL is responsible for progressively regulating the AIRA lower bound, $TotalT_x$ depends on MAIL as well. We depict the *Round Number–*$TotalT_x$ interdependence in Fig. 18c. The Number of Extra Rounds increases as the Number of *Total* Rounds increases. That is, a new flow (e.g. a flow within the first 10 *Rounds*) will become aware of the extra bandwidth after less than 2 *Rounds*. We consider this delay as acceptable, inline with our *Solution Framework* (see Section 4.3).

6.4. Results: Response to contention decrease

We repeat the simulation presented in Section 5.4; in the current setup, 2 of the 4 participating flows leave the system at the 205th second. We see in Fig. 19 that the remaining flows (i.e., flows 1 and 2) become aware of the extra available bandwidth and reset a to its initial value (i.e., 1). After the contention decrease event, both flows adjust their Additive Increase factor to a value close to 1; AIRA detects stable *Loss Rate* and hence the rate acceleration is stabilized to that value as well. The evaluation results of the current experiment are similar with the ones presented in Table 3.

7. Algorithm deployability

We consider that an algorithm can be deployed if it satisfies at least two conditions:

(1) It achieves (at least) the same goodput performance as a regular TCP-SACK flow.
(2) It allows TCP-SACK to operate as usually, with respect to system Goodput.

In this context, we attempt to assess *Deployability* of the proposed algorithm with one representative simulation

[4] This may sound as a paradox. Reader may consult justification in Section 4.3.

[5] System-wise, we assume that AIRA *should* exploit bandwidth fastly *iff* $\frac{n-x}{n}\leqslant\frac{2}{3}$, where n is the total number of participating flows and x is the number of flows who end their task and leave the system. In this case (i.e., when $x\geqslant\frac{1}{3}n$), AIRA resets a to 1.

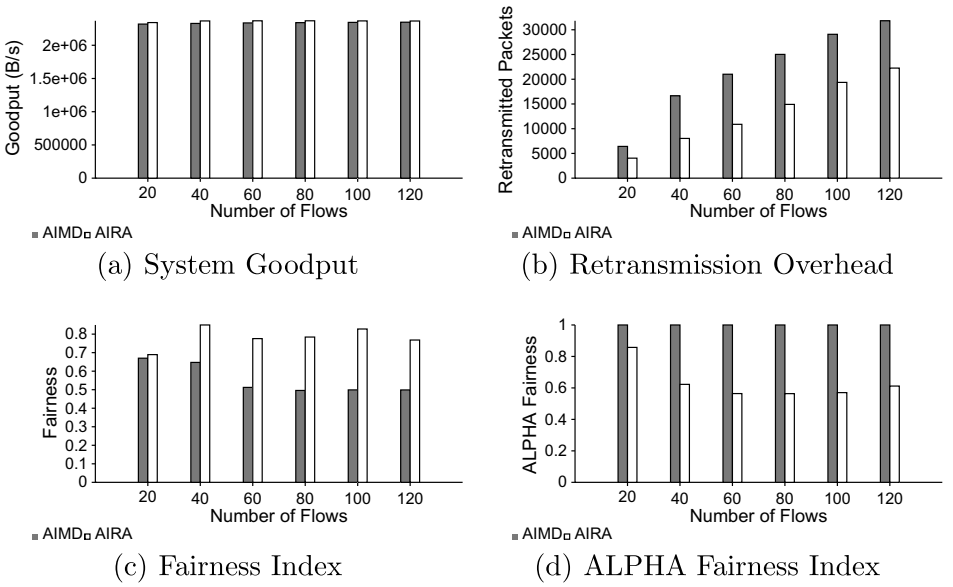


Fig. 17. RTT un-fairness scenario.

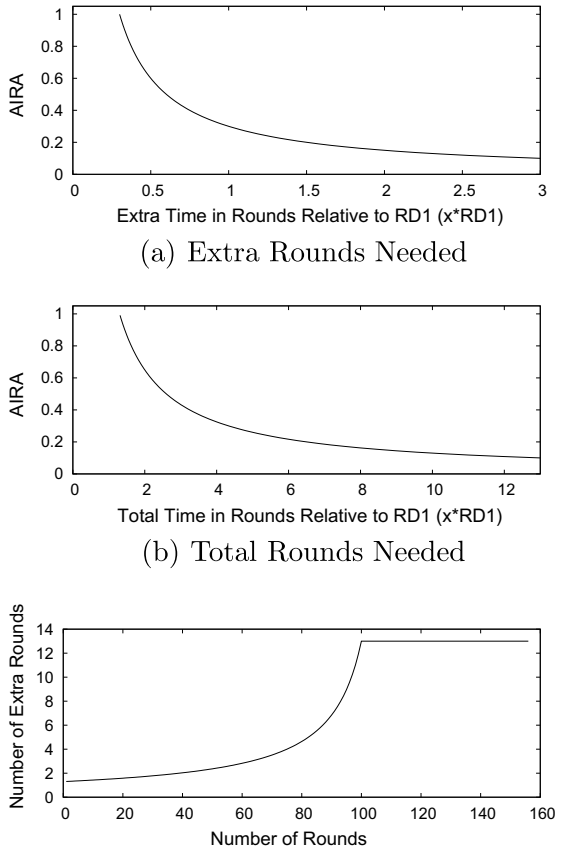


Fig. 18. AIRA bandwidth exploitation properties.

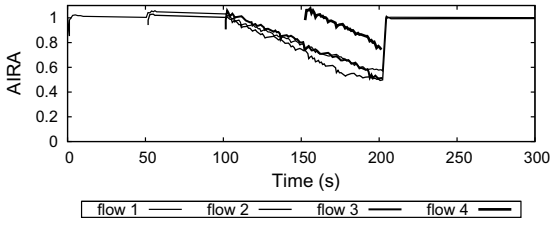


Fig. 19. AIRA reset condition.

experiment. We note that results are similar for various network conditions and simulation parameters.

We use the dumbbell network topology (Fig. 2); the speed of the backbone link is 10 Mbps, its propagation delay is 10ms and the Drop Tail buffer at Router 1 can hold 15 packets (i.e., equal to the Bandwidth-Delay Product of the backbone link). Initially, 10 regular TCP-SACK flows participate in the experiment. We see in Figs. 20a and 20d that TCP-SACK flows transmit 100 KB/s and retransmit 600 packets, in average. Next, we repeat the experiment using 10 AIRA flows. In Figs. 20b and 20e, we see that AIRA flows transmit approximately 120KB/s and retransmit 250 packets, in average. Finally, we simulate five standard TCP-SACK flows and five AIRA flows. The results are presented in Figs. 20c and 20f. We conclude that AIRA is indeed *Deployable* since it satisfies both deployability conditions: (i) AIRA flows achieve higher goodput performance than regular TCP-SACK flows, (ii) AIRA does not appear as an obstacle for TCP-SACK flows that achieve the same goodput performance as in the first experiment (Fig. 20a). AIRA takes advantage of the reduced retransmission overhead and allocates network resources to successful data transmission.

8. Open issues and future work

The Additive Increase Rate Accelerator proposed here, increases the operational complexity of networked sys-

Please cite this article in press as: I. Psaras, V. Tsaoussidis, On the properties of an additive increase rate accelerator, Comput. Netw. (2008), doi:10.1016/j.comnet.2008.10.019

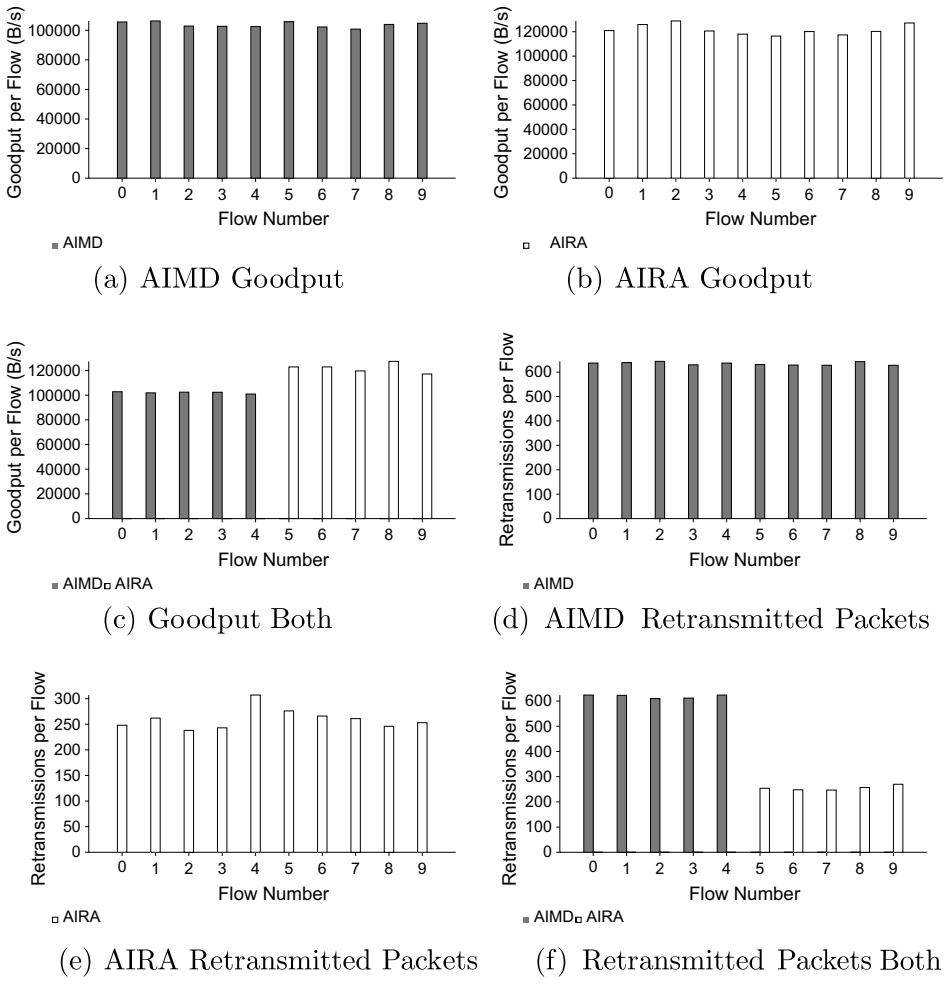


Fig. 20. Algorithm deployability.

tems. This fact alone is a negative system attribute; however, the proposed algorithm does not increase the complexity of flow engineering. That is, the flows will operate on predetermined rules, which will apply in diverse conditions. The conditions however, exhibit different properties: some are certain and precise (e.g. packet loss) but some others need to be evaluated. For instance, progressive contention is here estimated and is not explicitly communicated by some central network authority. The accuracy and precision of this estimation depends on two main factors: (i) the granularity of measurements and (ii) the accuracy of the monitoring functions. For example, the way throughput is calculated and the frequency of calculating throughput may have some impact. Higher (e.g. per packet) measurement granularity[6] will probably increase precision, but it will also increase system entropy, leading to reduced system stability. This observation calls for further investigation.

By the same token, even when contention/congestion estimation is accurate indeed, the responsive behavior of flows, does not have a sole corresponding pattern. That is, aggressiveness can be adjusted in order to balance overhead, efficiency or stability. An optimal balance has not been investigated here.

Finally, responses can be implemented rapidly or smoothly; the efficiency of each strategy depends on system dynamics. This is another issue that calls for further investigation. In conclusion, the interdependency of granularity, aggressiveness and responsiveness of AIRA has not been studied in depth.

9. Conclusions

We have shown that analytical rules can be derived for accelerating, either positively or negatively, the increase rate of AIMD in accordance with network dynamics. Indeed, we found that the "blind" Additive Increase rule can become an obstacle for the performance of TCP, espe-

[6] Although measuring throughput on a per-packet (or RTT) basis cannot capture the path loss rate and therefore, would not provide any advantage to the proposed algorithm, such an approach could potentially provide alternative information, such as the queuing delay evolution with time.

Please cite this article in press as: I. Psaras, V. Tsaoussidis, On the properties of an additive increase rate accelerator, Comput. Netw. (2008), doi:10.1016/j.comnet.2008.10.019

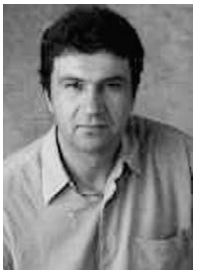
cially when contention increases. Instead, sophisticated, contention-aware additive increase rates may preserve system stability and reduce retransmission effort, without reducing the goodput performance of TCP.

Based on specific criteria, namely efficiency, fairness and stability, we proposed and evaluated an adaptive additive increase rate scheme, which we call Additive Increase Rate Accelerator (AIRA). We have shown two major results: (i) fairness is possible even in the context of varying increase rates within the same system. Furthermore, the problematic balance among short and long flows, as well as long- and short-RTT flows, can be better handled. (ii) Efficiency can be improved. Efficiency is judged not only on the basis of goodput performance but mainly against retransmission effort and overhead.

References

[1] I. Psaras, V. Tsaoussidis,AIRA: ditive Increase Rate Accelerator, in: Proceedings of IFIP Networking 2008, Singapore.
[2] L. Xu, K. Harfoush, I. Rhee, Binary Increase Congestion Control (BIC) for fast long-distance networks, in: Proceedings of INFOCOM 2004, vol. 4, 2004, pp. 2514–2524.
[3] R. King, R. Baraniuk, R. Riedi, TCP-Africa: An adaptive and fair rapid increase rule for scalable TCP, in: Proceedings of INFOCOM 2005, vol. 3, 2005, pp. 1838–1848.
[4] D.X. Wei, C. Jin, S.H. Low, S. Hegde, FAST TCP: motivation, architecture, algorithms, performance, IEEE/ACM Transactions on Networking 14 (6) (2006) 1246–1259.
[5] I. Rhee, L. Xu, CUBIC: A new TCP-Friendly high-speed TCP variant, in: Proceedings of PFLDnet, 2005.
[6] S. Floyd, Highspeed TCP for large congestion windows, RFC 3649 (December 2003).
[7] T. Kelly, Scalable TCP: improving performance in highspeed wide area networks, ACM SIGCOMM Computer Communication Review 33 (2) (2003) 83–91.
[8] K. Tan, J. Song, Q. Zhang, M. Sridharan, A Compound TCP Approach for High-Speed and Long Distance Networks INFOCOM 2006, in: Proceedings of 25th IEEE International Conference on Computer Communications, April 2006, pp. 1–12.
[9] J. Postel, Transmission Control Protocol, RFC 793, September 1981.
[10] D.-M. Chiu, R. Jain, Analysis of the increase and decrease algorithms for congestion avoidance in computer networks, Computer Networks and ISDN Systems 17 (1) (1989) 1–14.
[11] S. Jin, L. Guo, I. Matta, A. Bestavros, TCP-Friendly simd congestion control and its convergence behavior, in: Proceedings of 9th IEEE International Conference on Network Protocols (ICNP'01), Riverside, CA, November 2001. 5, 2001.
[12] G.D.J. Leith, R.N.Shorten, Experimental evaluation of CUBIC-TCP, in: Proceedings of PFLDnet, 2007.
[13] L. Brakmo, S. O'Malley, L. Peterson, TCP Vegas: New techniques for congestion detection and avoidance, in: Proceedings of SIGCOMM, 1994.
[14] A. Lahanas, V. Tsaoussidis, Exploiting the efficiency and fairness potential of AIMD-based congestion avoidance and control, Computer Networks 43 (2) (2003) 227–245.
[15] A. Lahanas, V. Tsaoussidis, τ-AIMD for asynchronous receiver feedback, in: Proceedings of the Eighth IEEE International Symposium on Computers and Communications, ISCC, 2003.
[16] G. Marfia, C. Palazzi, G. Pau, M. Gerla, M. Sanadidi, M. Roccetti, Tcp Libra: exploring rtt-fairness for tcp, UCLA Computer Science Department Technical Report TR050037.
[17] C. Zhang, V. Tsaoussidis, TCP smoothness and window adjustment strategy, IEEE Transactions on Multimedia 8 (3) (2006) 600–609.
[18] V. Tsaoussidis, C. Zhang, The dynamics of responsiveness and smoothness in heterogeneous networks, IEEE Journal on Selected Areas in Communications (JSAC) 23 (6) (2005) 1178–1189.
[19] S. Floyd, E. Kohler, TCP Friendly rate control (TFRC): the Small-Packet (SP) Variant, RFC 4828, experimental, April 2007.
[20] I. Rhee, V. Ozdemir, Y. Yi, TEAR: TCP Emulation at Receivers – flow control for multimedia streaming, NCSU Technical Report, URL <citeseer.ist.psu.edu/rhee00tear.html>, April 2000.
[21] D. Bansal, H. Balakrishnan, Binomial congestion control algorithms, in: INFOCOM 2001, URL citeseer.ist.psu.edu/bansal01binomial.html, pp. 631–640.
[22] Y.R. Yang, S.S. Lam, General AIMD congestion control, in: Proceedings of ICNP 2000, 2000.
[23] D.P. Bertsekas, Data Networks, 2nd ed., 1991., Prentice-Hall, 1987.
[24] ns 2, The network simulator – ns- 2, http://www.isi.edu/nsnam/ns/.
[25] M. Mathis, J. Mahdavi, S. Floyd, A. Romanow, TCP Selective acknowledgement options, rfc 2018, April 1996.
[26] I. Psaras, V. Tsaoussidis, L. Mamatas, CA-RTO: A Contention-Adaptive Retransmission Timeout, in: Proceedings of ICCCN, 2005.
[27] I. Psaras, V. Tsaoussidis, Why TCP timers (still) don't work well, Computer Networks 51 (8) (2007) 2033–2048.
[28] I. Psaras, V. Tsaoussidis, Wb-RTO: A Window-Based Retransmission Timeout for TCP, in: Proceedings of the 49th IEEE Global Telecommunications Conference GLOBECOM 2006, San Francisco, USA.
[29] S. Floyd, V. Jacobson, Random Early detection gateways for congestion avoidance, IEEE/ACM Transactions on Networking 1 (4) (1993) 397–413.

Ioannis Psaras received a diploma in Electrical and Computer Engineering from Demokritos University of Thrace, Greece in 2004, and the Ph.D. degree from the same institute in 2008. He won the Ericsson Award of Excellence in Telecommunications for his diploma dissertation in 2004. Ioannis has worked, as a research intern at DoCoMo Eurolabs (May–September 2005) and at Ericsson Eurolab (May–September 2006). His research interests lie in the area of transport protocols and their behavior over heterogeneous (wired, wireless, satellite, high-speed) links, deep-space communications and Delay/Disruption-Tolerant Networking. He is currently working as a post-doctoral researcher at the Center for Communications and Systems Research (CCSR) of the University of Surrey. Ioannis participates in the Technical Program Committees for IFIP NETWORKING 09 and IEEE ICCCN 09. Further information can be found at: http://www.ee.surrey.ac.uk/Personal/I.Psaras.



Vassilis Tsaoussidis received a B.Sc. In Applied Mathematics from Aristotle University, Greece; a Diploma in Statistics and Computer Science from the Hellenic Institute of Statistics; and a Ph.D. in Computer Networks from Humboldt University, Berlin, Germany (1995). Vassilis held faculty positions in Rutgers University, New Brunswick, SUNY Stony Brook and Northeastern University, Boston. In May 2003, he joined the Department of Electrical and Computer Engineering of Demokritos University, Greece. His research interests lie in the area of transport/network protocols, i.e. their design aspects and performance evaluation. Vassilis is editor in chief for the Journal of Internet Engineering and editor for the journals IEEE Transactions in Mobile Computing, Computer Networks, Wireless Communications and Mobile Computing, Mobile Multimedia and Parallel Emergent and Distributed Systems. He participated in several Technical Program Committees in his area of expertise, such as INFOCOM, NETWORKING, GLOBECOM, ICCCN, ISCC, EWCN, WLN, and several others.

Please cite this article in press as: I. Psaras, V. Tsaoussidis, On the properties of an additive increase rate accelerator, Comput. Netw. (2008), doi:10.1016/j.comnet.2008.10.019