Why Rely on Blind AIMDs?

Ioannis Psaras, Mehrdad Dianati and Rahim Tafazolli
Center for Communication Systems Research (CCSR)
Dept. of El. Eng., University of Surrey
Guildford, GU2 7XH, Surrey, UK
EDAS Paper ID:1569146346
(i.psaras, m.dianati, r.tafazolli)@surrey.ac.uk

ABSTRACT

We are motivated by the fact that fixed Increase rates and Decrease ratios for AIMD cannot adjust TCP's performance to the Internet's diverse networking conditions. Indeed, we find that fixed values for the increase/decrease factors of AIMD restrain flexibility, which is a fundamental property of transport protocols in order to guarantee utilisation and fairness in Modern and Future Internetworks. We propose a new paradigm for AIMD designs that has the potential to adjust TCP's behaviour according to network conditions. The proposed *Multi-Rate AIMD* increases additively the Additive Increase factor of AIMD when the level of network contention is low and decreases multiplicatively (the AI factor) on the face of triple duplicate ACKs or timeout expirations. The Multiplicative Decrease factor of AIMD remains constant in order to guarantee fairness and stability. We show that MR-AIMD reduces retransmission effort significantly, when contention is high, becomes aggressive when contention decreases and tolerates against random, transient errors due to fading channels.

1. INTRODUCTION

The *Transmission Scheduling* rules for transport layer protocols have been traditionally designed on the basis of *demand and supply*. *Demand* refers to the amount of user data that need to be transferred over the network, while *supply* refers to the capacity of the network itself. The success of the *Traditional Internet* relies mainly on its ability to provide system utilisation and fairness. In turn, system utilisation and fairness rely heavily on the transmission scheduling strategy of the transport layer protocol.

The Internet is characterised by *application diversity* and *infrastructure heterogeneity*. Applications download data and leave the system, causing drastic load fluctuations. That said, demand may only temporarily exceed supply. On the other hand, generous supply may be provided permanently (e.g., high-speed paths). Finally, demand may exceed supply for long time periods, during rush hours for example. In all cases, transport protocols inevitably adjust their transmission and retransmission rates according to feedback provided either by the receiver side or by the network itself. Feedback, however, arrives at the sender side, typically, in binary form. That is, the sender is not informed about the intensity of contention and therefore, responses are identical in all cases. Moreover, link errors, which result in packet losses, are falsely interpreted as congestion events and trigger similar responses as well. We argue that the growth of the Internet and its subsequent effects on the characteristics of Internet traffic have not been incorporated into the *Transmission and Retransmission Scheduling* rules of transport layer protocols.

Indeed, the dominant transport layer congestion control algorithm, AIMD, remains untouched. Although AIMD has proved to work perfectly until now (i.e., congestive collapses are effectively avoided), we argue that it is about time to reconsider the inherent properties of AIMD in conjunction with today's Internet application and infrastructure heterogeneity. For example, we show that 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.

We attempt to assess the cost of the "blind" Additive Increase rule of AIMD on the performance of the *Transmission Control Protocol* (TCP), within the context of the Modern and the Future Internet. In Section 2, we present our motives and show that a fixed value for the Additive Increase factor of AIMD fails to provide system utilisation and fairness in Modern Internetworks. In Section 3, we introduce a new paradigm for the responsive behaviour of flows (i) when bandwidth availability changes due to varying network contention or (ii) when network conditions cause system in-stability, un-fairness and inefficient resource utilisation.

More precisely, we investigate the properties of a *Multi-Rate*, AIMD-based, Additive Increase factor. Briefly, the algorithm operates as follows: upon successful delivery of *cwnd* number of packets to the receiver side, not only the *cwnd*, but also the *the Additive Increase factor* increases Additively (i.e., $a \leftarrow a + \frac{a'}{cwnd}$), while on the face of loss, *the Additive Increase factor* is Multi-

plicatively Decreased (i.e., $a \leftarrow a - b \cdot a$). Decisions as to whether the AI factor should be increased or decreased and by how are based on AQM techniques, namely ECN [3], although several alternative approaches may also apply. In Section 4, we present our initial, but still very promising simulation results. We discuss some issues that need to be further investigated before final conclusions can be drawn and conclude the paper in Section 5.

2. MOTIVATION: BLIND AIMD

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 the *Additive Increase*, a, and *Multiplicative Decrease*, b, factors, in order to achieve fast bandwidth exploitation (e.g., [4], [7], [1], [13], [11], [2], [8]) or faster convergence to fairness (e.g., [6], [9] and references therein), 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 a packets instead of* $2a$ *packets, even when half users of a system leave and bandwidth becomes available?*

2.1 Congested Wired Network

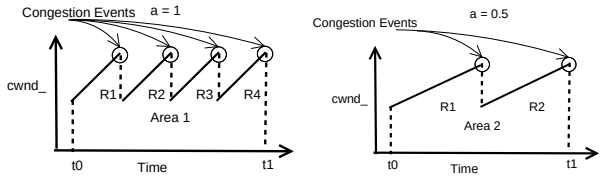
One possible justification for not highlighting the above research direction is that:

The Additive Increase factor of AIMD does not contribute to the long-term Goodput performance of TCP.

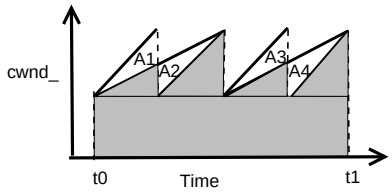
In Figure 1, we present the *cwnd* evolution for two TCP flavors: Figure 1(a), where $a = 1$ (regular TCP) and Figure 1(b), where $a = 0.5$. The area underneath the solid *cwnd* lineplot (Area 1 and 2) represents the Goodput[1] performance of the protocols. In Figure 1(c), we show that both protocols 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 behaviour, as well.

For example, TCP $a = 1$, in Figure 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.



(a) Increase Factor = 1 (b) Increase Factor = 0.5



(c) Difference

Figure 1: Different Increase Factors

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

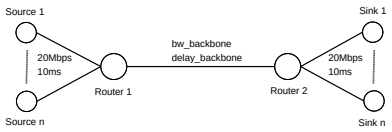


Figure 2: Dumbbell Network Topology

Table 1: TCP Performance I - 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

Clearly, there is a tradeoff 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 (see for example, the 4 flow scenario in Table 1). Note that further increasing the level of contention may even degrade the system Goodut performance, due to timeout expirations [10], which are not considered in Figure 1.

[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., $\frac{Total\ Data}{Connection\ Time}$).

[2]In Figure 1(c) grey areas are common for both protocols; white areas are equal ($A1$ is similar to $A2$ and $A3$ is similar to $A4$).

In high contention scenarios, the higher the Additive Increase factor, the more retransmissions it causes, with zero gains in system Goodput.

2.2 Wireless, Mobile Computing

On the contrary, losses due to congestion may not always be the case. The evolution of mobile, wireless networking calls for further investigation and adjustment of transport layer algorithms to deal with losses due to wireless, fading channels as well. In this context,

The Additive Increase factor of AIMD may very well impact TCP's Goodput performance, when contention is low and losses are due to wireless errors.

We repeat the previous simulation to verify the above statement. The backbone link can now transfer 10Mbps (instead of 1Mbps) and we additionally insert 0.3 Packet Error Rate (PER) to emulate losses due to fading, wireless channels. The results are presented in Table 2. We observe that in case of low contention and transient errors due to fading channels, higher values for the Additive Increase factor of AIMD can boost TCP's performance significantly.

In low contention scenarios, where transient losses happen due to fading channels, the higher the Additive Increase factor, the more the Goodput gains for TCP.

Table 2: TCP Performance II - Different Increase Factors

	Goodput (KB/s)	
	2 flows	4 flows
$a = 5$	114.8	238.4
$a = 2$	81.3	179.7
$a = 1.5$	73.5	150.5
$a = 1$	63.1	127.9
$a = 0.5$	44.2	89

2.3 Bandwidth Exploitation Properties

Today's Internet application and infrastructure heterogeneity demands for responsive transport protocols, which are able to exploit extra available bandwidth rapidly, in case of contention decrease; at the same time the transport layer protocol should be able to adjust its transmission rate downwards in case of incoming flows, in order to (i) leave space for the new flows and (ii) not overflow the network. That said, fixed Additive Increase provides fixed transmission rate acceleration both in case of contention decrease and in case of extra bandwidth constraints. We argue that such behaviour is undesirable indeed, since it leads to slow bandwidth exploitation, when bandwidth becomes available, while it requires great retransmission effort when bandwidth constraints prevail (see Section 2.1).

In case of contention decrease / increase scenarios, fixed acceleration leads to either slow resource exploitation or high retransmission effort, respectively.

3. MR-AIMD: MULTI-RATE AIMD

3.1 The Algorithm

Regular TCP increases its congestion window by 1, upon successful transmission of *cwnd* number of packets (i.e., $cwnd \leftarrow cwnd + \frac{a}{cwnd}$), while negative feedback (i.e., three duplicate ACKs), which is interpreted as network congestion, triggers *multiplicative cwnd* decrease (i.e., $cwnd \leftarrow cwnd - b \cdot cwnd$), where $a = 1$ and $b = 0.5$, according to [5].

As an initial approach to a "non-blind", dynamically adjustable increase factor, we attempt to graft the basic AIMD functionality *into the Additive Increase factor of TCP*. More precisely, the *Multi-Rate AIMD* algorithm increases the *cwnd* value according to:

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

The proposed algorithm makes use of Active Queue Management (AQM) techniques in order to set the Additive Increase rate, a', of MR-AIMD. In particular, the algorithm uses the Explicit Congestion Notification (ECN) bit [3]. Once set, by the intermediate router, the ECN bit may trigger one of three possible responses: (i) Additive Increase, (ii) Multiplicative Decrease or (iii) stabilisation of the AI factor. Upon arrival of an ECN marked packet MR-AIMD increases additively the AI factor of AIMD (i.e., choice (i)). The rationale behind our choice is as follows: according to measurements, multiplicative decrease of the AI factor results in very low values and therefore, conservative behaviour. On the other hand, rate stabilisation, through choice (iii), may result in system instability and flow unfairness, in the long term. Due to space limitations, we do not elaborate further on this issue, but we report that initial results verify our decisions for increased system performance. Since an ECN marked packet signals for incipient network congestion, we decelerate MR-AIMD's rate upon arrival of a marked packet. That is, we set the initial value for a to 1 and for a' to 0.5, while an ECN marked packet signals for reduction of a' to 0.005 (see Equation 1).

At this point, we explicitly note that we do not use the standard ECN algorithm, but rather a simple modification, which reduces the Additive Increase factor *only* (i.e., the flow's *cwnd* is *not* reduced). This modification exhibits a number of desirable properties: i) it smooths TCP's transmission rate and ii) it avoids (to an extend) TCP's drastic rate fluctuations, whenever deemed appropriate according to the proposed algorithm.

The Additive Increase factor, a, decreases multiplicatively, according to Equation 2, upon a triple duplicate

ACK event:

$$a \leftarrow a - b \cdot a. \quad (2)$$

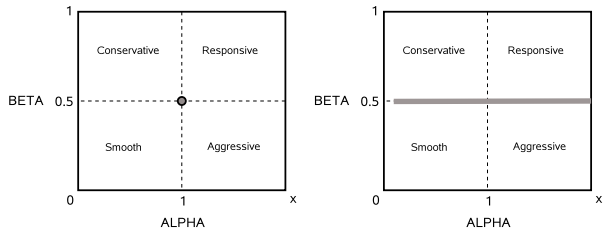
The Multiplicative Decrease factor of MR-AIMD, b, is set to 0.5 similarly to TCP-AIMD (Equation 2), in order to guarantee fairness and stability [5]. Furthermore, upon a timeout event, MR-AIMD reduces the Additive Increase factor to its initial value, 0.5, in order to account for increased levels of network contention.

3.2 Discussion

We assume that TCP's operational space, with regard to the Additive Increase and Multiplicative Decrease factors ALPHA and BETA, is represented by four basic domains: i) conservative, ii) aggressive, iii) smooth and iv) responsive (see Figure 3). The current, blind TCP-AIMD implementation covers a single point, only, within TCP's operational space (see Figure 3(a)). Clearly, the fixed increase/decrease parameters deal with none of the four operational domains, efficiency-wise and moreover, *any* pair of *fixed* increase/decrease parameters can deal with *one* operational domain *only*. We argue that such settings form an inflexible, conservative, worst-case approach to TCP's operational properties. For instance, a sophisticated transport layer algorithm should adjust according to network conditions: it should become conservative when contention is high, aggressive in case of transient wireless errors, responsive in case of contention increase/ decrease and smooth in case of (relatively) static network load.

The proposed scheme extends TCP's functionality to operate along the x-axis of TCP's operational space. This way, several desirable properties are added to TCP's inherent functionality. That said, careful design can lead to more aggressive transmission when contention is low and losses happen on the wireless portion of the network, while conservative transmission, when contention increases, can account for reduced retransmission overhead. The current proposal constitutes a first step on the further extension of TCP's functionality, in order to exploit the whole spectrum of possible behaviours (i.e., utilisation of the y-axis as well).

We note that MR-AIMD does not target high-speed environments. Although the MR-AIMD's transmission rate may increase compared to regular TCP, its operational properties are not intended to exploit high-speed links. Instead, the proposed algorithm attempts to deal with the application diversity and infrastructure heterogeneity of present and future internetworks. In any case, most algorithms for high-speed environments are triggered from a point onwards, depending on the value of the *cwnd* (e.g., [4], [7]). By the same token, similar settings may also apply to MR-AIMD as well.



(a) TCP-AIMD (b) MR-AIMD

Figure 3: x-AIMD Operational Space

4. PRELIMINARY RESULTS

We evaluate the performance of the proposed algorithm through simulations. We use the SACK version of TCP with the timestamps option enabled. The simulation scenarios are similar to the ones presented in Section 2. That is, we use the dumbbell network topology, the queueing policy is RED and the buffer size is set according to the bandwidth-delay product of the outgoing link.

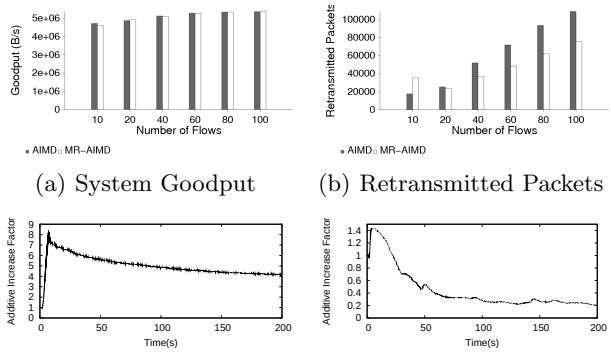
4.1 Congested Wired Networks

Initially, we simulate a wired network where the backbone link transfers 48 Mbps and induces propagation delay of 40ms. We repeat the simulation for increasing number of participating flows (from 10 to 100), to capture the performance of the proposed algorithm relatively with the level of contention.

We observe that when contention is low (e.g., 10 flows over 48Mbps) the proposed algorithm achieves the same Goodput performance as regular AIMD (see Figure 4(a)); the retransmission effort graph (Figure 4(b)) reveals that for low contention environments, the proposed AIMD operates aggressively. In Figure 4(c), we graph the average Additive Increase factor for a random flow, when the total number of participating flows is 10. This Figure verifies the aggressive behaviour of MR-AIMD, when contention is low.

As contention increases, however, losses due to buffer overflow become more frequent, leading to multiplicative decreases of both the *cwnd* and the Additive Increase factor. In turn, smaller increase rates lead to reduced retransmission effort (see Figure 4(b)). In Figure 4(d), we present the average Additive Increase factor of MR-AIMD, for a random flow, when the total number of participating flows is 100. Indeed, we see that the average value of MR-AIMD's Additive Increase factor is below 1, allowing for less aggressive transmission, since the level of network contention so permits.

Overall, we see that the proposed algorithm adjusts efficiently to the level of network contention, taking advantage of its dynamic increase/decrease acceleration properties to utilise resources accordingly. In particular, when contention is low the algorithm is aggres-



(a) System Goodput (b) Retransmitted Packets

(c) 10 Flows Experiment (d) 100 Flows Experiment

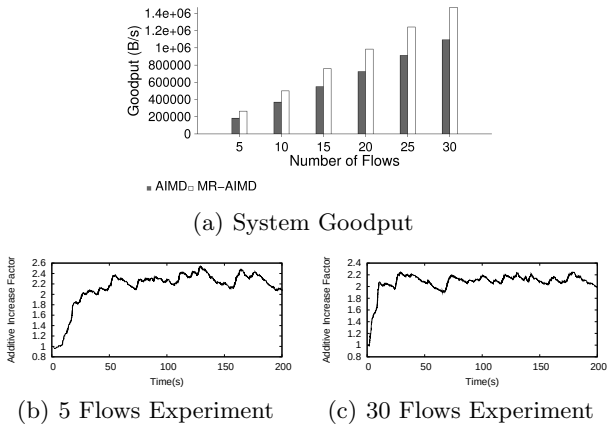
Figure 4: Performance over Congested Wired Networks

sive, ready to utilise rapidly extra available bandwidth, while when contention increases the algorithm lowers the transmission rate to reduce retransmission effort.

4.2 Wireless Networks

We repeat the simulation presented in Section 2.2 to observe the performance of the proposed algorithm over wireless, lossy links. In the current setup the backbone link transfers 48Mbps with 40ms propagation delay, while the PER is 0.3.

Figure 5(a) depicts the outcome of the simulation. We see that the proposed algorithm is tolerant against random, transient errors caused by wireless, fading channels. MR-AIMD accelerates transmission faster than conventional AIMD, becoming more aggressive, when conditions permit, which is another desirable property in case of wireless errors [12]. Indeed, we see in Figures 5(b) and 5(c) that the Additive Increase factor is far above 1, allowing for speedy transmission and up to, approximately, 30% higher Goodput performance (Figure 5(a)) in case of errors due to wireless, lossy links.



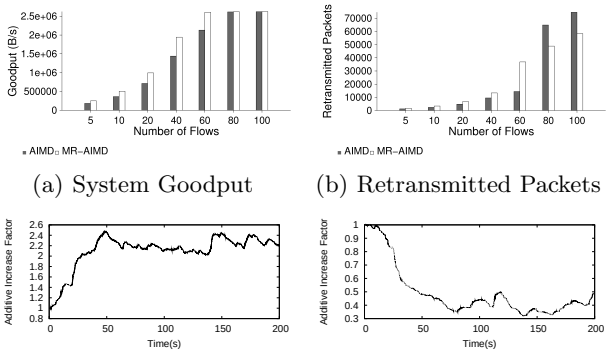
(a) System Goodput

(b) 5 Flows Experiment (c) 30 Flows Experiment

Figure 5: Performance over Wireless Networks

4.3 Mixed Environments

We perform one more experiment in order to verify that MR-AIMD adjusts correctly in mixed wired-wireless environments, where the level of contention may vary. The simulation environment is the same as previously, but the backbone link can now transfer 24Mbps. We repeat the simulation for increasing number of participating flows, from 5 to 100. Indeed, we see in Figure 6(a) that when contention is low MR-AIMD exploits the available resources, tolerates against random link errors and accelerates transmission (see Figure 6(c)), increasing the overall system Goodput. In contrast, when contention increases (i.e., 80 and 100 participating flows), MR-AIMD reduces its transmission rate, through lower Additive Increase factors (Figure 6(d)), although some errors may still be due to fading channels. By doing so, MR-AIMD achieves the same Goodput performance as conventional AIMD, but reduces the retransmission effort of the transport protocol (Figure 6(b)). The present experiment verifies the hybrid behaviour of MR-AIMD, which based on ECN signals adapts appropriately to the network conditions. Although ECN is not famous as an error discriminator, our initial results show that ECN-capable transports may be benefitted, at least to an extend, from its operation as such. Further experimentation is needed in order to uncover ECN's capabilities regarding its accuracy on that direction.



(a) System Goodput (b) Retransmitted Packets

(c) 5 Flows Experiment (d) 80 Flows Experiment

Figure 6: Performance over Mixed Wired-Wireless Networks

4.4 Bandwidth Exploitation Properties

We attempt to briefly assess the bandwidth exploitation properties of the proposed algorithm. The Additive Increase factor of MR-AIMD progresses in time according to:

$$a_n = a_{n-1} + a' \cdot n, \ where\ n \geq 1. \qquad (3)$$

In Equation 3, n stands for the number of RTTs, and a' is the acceleration factor of MR-AIMD (i.e., ei-

ther 0.5 or 0.005). The initial value for a_n, for a new connection for example, is 1. Otherwise, for an existing connection, the initial value of a_n depends on the algorithm's state (i.e., AI through Equation 1, or MD through Equation 2).

In turn, MR-AIMD's *cwnd* after n RTTs is given by:

$$W_{fin} = W_{init} + \sum a_n, \tag{4}$$

TCP's *cwnd* after n RTTs is given by:

$$W_{fin} = W_{init} + a \cdot n, \tag{5}$$

where W_{init} is the initial *cwnd* and W_{fin} is the *cwnd* after n RTTs. Obviously, for TCP-AIMD $a = 1$, while for MR-AIMD a' is either equal to 0.5 or 0.005.

We assume a contention decrease event, where a number of participating flows leave the system when $W_{init} = 20$. From that point onwards, the rest of the flows have to exploit the extra bandwidth as fast as possible. We assume that since contention has decreased there are no ECN signals to the TCP sender (at least up to a certain point where contention becomes high due to the increased congestion windows of the participating flows).

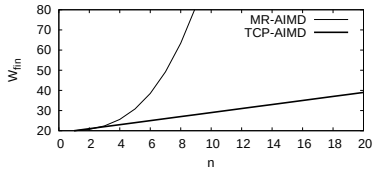


Figure 7: Extra Bandwidth Exploitation Properties

As expected, we see in Figure 7 that MR-AIMD has the potential to exploit extra available network resources rapidly, without threatening the system's stability. That is, although initially the algorithm appears aggressive (MR-AIMD doubles its window within 6 RTTs, while TCP-AIMD needs 20 RTTs), it will immediately slow down, when ECN marked packets indicate incipient congestion. However, due to limited space and time, we do not elaborate further on that issue here.

5. CONCLUSIONS

We argue that a blind Additive Increase factor for AIMD limits TCP's performance in terms of efficient resource utilisation. We proposed a rather simple but novel approach towards a new design space for transport layer internetworking. Although the proposed settings are chosen based on experimental evaluations only, they seem to boost TCP's performance significantly. Moreover, additional modifications can easily be incorporated. For example, we did not evaluate here the properties of MR-AIMD with regard to the RTT-unfairness problem of TCP. Although one may argue that the proposed algorithm, in its current form, may extend TCP's inability to treat diverse RTT flows fairly, simple modifications can improve TCP's performance on that direction as well. For instance, the MR-AIMD's Additive Increase factor may be complemented with a fraction of the flow's measured RTT sample (i.e., $a \leftarrow a + \frac{c \cdot a'}{cwnd}$, where c is the flow's latest measured RTT sample).

6. ACKNOWLEDGMENTS

This work is supported in part by the European Union through the 4WARD project (http://www.4ward-project.eu/) in the 7th Framework Programme.

7. REFERENCES

[1] D. X. W. et al. FAST TCP: motivation, architecture, algorithms, performance. *IEEE/ACM Transactions on Networking*, 2007.

[2] K. T. et al. A Compound TCP Approach for High-Speed and Long Distance Networks.

[3] S. Floyd. TCP and Explicit Congestion Notification. *SIGCOMM CCR*, 24(5):10–23, 1994.

[4] S. Floyd. HighSpeed TCP for Large Congestion Windows, RFC 3649, December 2003.

[5] V. Jacobson. Congestion avoidance and control. In *Proc. of SIGCOMM*, pages 314–329, 1988.

[6] S. Jin, L. Guo, I. Matta, and A. Bestavros. TCP-friendly SIMD Congestion Control and Its Convergence Behavior. In *Proc. of 9th IEEE ICNP, Riverside, CA*, 2001.

[7] T. Kelly. Scalable TCP: Improving Performance in Highspeed Wide Area Networks. *ACM SIGCOMM CCR*, 33(2):83–91, 2003.

[8] R. King, R. Baraniuk, and R. Riedi. TCP-Africa: An Adaptive and Fair Rapid Increase Rule for Scalable TCP. In *Proceedings of INFOCOM 2005*, volume 3, pages 1838–1848, 2005.

[9] A. Lahanas and V. Tsaoussidis. Exploiting the Efficiency and Fairness Potential of AIMD-based Congestion Avoidance and Control. *Computer Networks*, 43(2):227–245, 2003.

[10] I. Psaras and V. Tsaoussidis. Why TCP timers (still) don't work well. *Computer Networks*, 51(8):2033–2048, 2007.

[11] I. Rhee and L. Xu. CUBIC: A New TCP-Friendly High-Speed TCP Variant. In *Proceedings of PFLDnet*, 2005.

[12] V. Tsaoussidis and I. Matta. Open issues on TCP for Mobile Computing. *Wiley, WCMC*, 2(1), Feb. 2002.

[13] L. Xu, K. Harfoush, and I. Rhee. Binary Increase Congestion Control (BIC) for fast long-distance networks. In *Proceedings of INFOCOM 2004*, volume 4, pages 2514–2524, March 2004.