Report Number: WUCSE-2006-40

2006-01-01

# A Theory of Load Adjustments and its Implications for Congestion Control

Sergey Gorinsky, Manfred Georg, Maxim Podlesny, and Christoph Jechlitschek

Multiplicative Increase (MI), Additive Increase (AI), and Multiplicative Decrease (MD) are linear adjustments used extensively in networking. However, their properties are not fully understood. We analyze responsiveness (time for the total load to reach the target load), smoothness (maximal size of the total load oscillations after reaching the target load), fairing speed (speed of convergence to equal individual loads) and scalabilities of MAIMD algorithms, which generalize AIMD algorithms via optional inclusion of MI. We prove that an MAIMD can provide faster asymptotic fairing than a less smooth AIMD. Furthermore, we discover that loads under a specific MAIMD converge from... **Read complete abstract on page 2.**

# A Theory of Load Adjustments and its Implications for Congestion Control

Sergey Gorinsky, Manfred Georg, Maxim Podlesny, and Christoph Jechlitschek

Complete Abstract:

Multiplicative Increase (MI), Additive Increase (AI), and Multiplicative Decrease (MD) are linear adjustments used extensively in networking. However, their properties are not fully understood. We analyze responsiveness (time for the total load to reach the target load), smoothness (maximal size of the total load oscillations after reaching the target load), fairing speed (speed of convergence to equal individual loads) and scalabilities of MAIMD algorithms, which generalize AIMD algorithms via optional inclusion of MI. We prove that an MAIMD can provide faster asymptotic fairing than a less smooth AIMD. Furthermore, we discover that loads under a specific MAIMD converge from any initial state to the same periodic pattern, called a canonical cycle. While imperfectly correlated with smoothness, the canonical cycle reliably predicts the asymptotic fairing speed. We also show that AIMD algorithms offer the best trade-off between smoothness and responsiveness. Then, we introduce smoothness-responsiveness diagrams to investigate MAIMD scalabilities. Finally, we discuss implications of the theory for the practice of congestion control.

2006-40

# A Theory of Load Adjustments and its Implications for Congestion Control

Authors: Sergey Gorinsky, Manfred Georg, Maxim Podlesny, Christoph Jechlitschek

Corresponding Author: gorinsky@arl.wustl.edu

Web Page: http://www.arl.wustl.edu/~gorinsky/pubs.html

Abstract: Multiplicative Increase (MI), Additive Increase (AI), and Multiplicative Decrease (MD) are linear adjustments used extensively in networking. However, their properties are not fully understood. We analyze responsiveness (time for the total load to reach the target load), smoothness (maximal size of the total load oscillations after reaching the target load), fairing speed (speed of convergence to equal individual loads) and scalabilities of MAIMD algorithms, which generalize AIMD algorithms via optional inclusion of MI. We prove that an MAIMD can provide faster asymptotic fairing than a less smooth AIMD. Furthermore, we discover that loads under a specific MAIMD converge from any initial state to the same periodic pattern, called a canonical cycle. While imperfectly correlated with smoothness, the canonical cycle reliably predicts the asymptotic fairing speed. We also show that AIMD algorithms offer the best trade-off between smoothness and responsiveness. Then, we introduce smoothness-responsiveness diagrams to investigate MAIMD scalabilities. Finally, we discuss implications of the theory for the practice of congestion control.

Type of Report: Other

# A Theory of Load Adjustments and its Implications for Congestion Control

Sergey Gorinsky, Manfred Georg, Maxim Podlesny, and Christoph Jechlitschek

Technical Report WUCSE-2006-40

Department of Computer Science and Engineering, Washington University in St. Louis
One Brookings Drive, St. Louis, MO 63130-4899, USA
{*gorinsky, mgeorg, podlesny, chrisj*}*@arl.wustl.edu*

August 2006

***Abstract*— Multiplicative Increase (MI), Additive Increase (AI), and Multiplicative Decrease (MD) are linear adjustments used extensively in networking. However, their properties are not fully understood. We analyze responsiveness (time for the total load to reach the target load), smoothness (maximal size of the total load oscillations after reaching the target load), fairing speed (speed of convergence to equal individual loads) and scalabilities of MAIMD algorithms, which generalize AIMD algorithms via optional inclusion of MI. We prove that an MAIMD can provide faster asymptotic fairing than a less smooth AIMD. Furthermore, we discover that loads under a specific MAIMD converge from any initial state to the same periodic pattern, called a canonical cycle. While imperfectly correlated with smoothness, the canonical cycle reliably predicts the asymptotic fairing speed. We also show that AIMD algorithms offer the best trade-off between smoothness and responsiveness. Then, we introduce smoothness-responsiveness diagrams to investigate MAIMD scalabilities. Finally, we discuss implications of the theory for the practice of congestion control.**

## I. Introduction

To regulate network congestion, Transmission Control Protocol (TCP) [1] and numerous other protocols rely on linear adjustments such as Multiplicative Increase (MI), Additive Increase (AI), and Multiplicative Decrease (MD) [2]. Linear adjustments are also extensively used for various networking tasks beyond traditional congestion control, e.g., for load balancing in clustered servers [3], active queue management [4], wireless media access [5], and multicast group subscription [6]. Despite the wide adoption of linear adjustments, their properties still require further understanding.

In this paper, we advance such comprehension by analyzing linear adjustment algorithms in the classical Chiu-Jain model where distributed users adjust their loads on a shared resource in response to uniform binary feedback that indicates whether the total load exceeds a target [2]. The original analysis [2] examined linear adjustment algorithms with respect to several interesting properties including responsiveness (time for the total load to reach the target), smoothness (maximal size of the total load oscillations after reaching the target), and fairing (convergence to equal individual loads). Chiu and Jain showed that MAIMD algorithms which generalize AIMD via optional inclusion of MI are stable, i.e., provide convergence to the target total load and equal individual loads. By revealing that the AIMD subclass of MAIMD algorithms offers the fastest fairing after a single adjustment, the original analysis supplied a theoretical justification for using AIMD in TCP congestion avoidance [7]. Chiu and Jain also showed that an overall optimal MAIMD algorithm does not exist due to a fundamental trade-off between responsiveness and smoothness.

Our choice of the model warrants an early discussion due to the following concerns about Chiu-Jain model: 1) The model makes oversimplifying assumptions. For instance, it assumes uniform feedback to all users while measurements at backbone routers show independent packet loss [8] and thus support an alternative assumption of non-uniform feedback; moreover, whereas MIMD control does not converge to equal individual loads in Chiu-Jain model, MIMD is fair in models with non-uniform feedback [9], [10]. Also, Chiu-Jain model implies that traffic consists of only long-lived flows, which is an obvious deviation from the Internet reality; 2) The model is not universal even in the context of congestion control. In particular, due to the assumption of binary feedback, the model does not lend itself to analysis of eXplicit Control Protocol (XCP) [11] and other designs where routers provide richer explicit feedback about congestion; 3) Chiu-Jain model is almost two decades old. Since then, more elaborate models have appeared and led to new insights and designs [12]–[14]. Nevertheless, we believe that Chiu-Jain model is appropriate for our investigation for the following reasons of increasing importance:

- **Wide applicability**. Chiu-Jain model represents many real scenarios with sufficient accuracy. The assumption of binary feedback does not interfere with analyzing TCP, which remains the dominant Internet transport protocol, or more recent proposals such as Scalable Transmission Control Protocol (STCP) [15] and Variable-structure congestion Control Protocol (VCP) [16], which does not infer congestion from losses but instead relies on explicit router feedback. Uniform feedback is generally unrealistic but does occur in real networks where congestion affects all flows at a low-multiplexing access link [17].

- **Elegance and intuitiveness**. The model is elegant in formulation and offers clear interpretation of derived results. More elaborate models become complex and lose intuitive appeal without eliminating all unrealistic assumptions. Lack of simple credible analysis [18] contributes greatly to the Internet ossification [19] because complex models fail to persuade a critical mass of stakeholders in overall goodness of advocated innovations [20]. Although Katabi, Handley, and Rohrs [11] proved XCP fairing in a more complicated model, they supported the argument with analogies between XCP and AIMD. The references to AIMD (which is widely known as stable in Chiu-Jain model) helped to alleviate concerns about XCP stability and promote the overall positive reception of XCP, even though XCP does not actually use AIMD but performs nonlinear adjustments which depend on not only the current load and fixed coefficients (as in Chiu-Jain model) but also the available network capacity (unknown to users in Chiu-Jain model).
- **Standard framework for fairing analysis**. Due to the elegance and intuitiveness, Chiu-Jain model is extensively used by textbooks to teach about fairing [21], [22] and by research papers to prove fairing properties of new protocols, including nonlinear-control protocols [23]–[25]. Thus, it is important to understand the body of knowledge induced by this standard analytical framework.

In this paper, we extend the classical theory of load adjustments and establish a number of surprising results. Section II conducts the analysis along the following three dimensions:

1) **Asymptotic fairing speed**. Our paper is the first to analyze the asymptotic fairing speed of MAIMD algorithms. We show that an MAIMD with an MI component can provide faster asymptotic fairing than a less smooth AIMD. Trying to understand this counterintuitive result, we discover that loads under a specific MAIMD converge from any initial state to the same periodic pattern, called a *canonical cycle*. While imperfectly correlated with smoothness, the canonical cycle reliably predicts the asymptotic fairing speed. We quantify the asymptotic fairness convergence with a *fairing factor* and express this metric as a function of the numbers of increases and decreases in the canonical cycle.
2) **Trade-off between responsiveness and smoothness**. We prove that AIMD guarantees the best responsiveness among linear adjustment algorithms with equal smoothness of both increase and decrease. In particular, AI rules offer a better trade-off between responsiveness and smoothness than MI.
3) **Scalabilities**. We introduce smoothness-responsiveness diagrams to investigate scalabilities of MAIMD algorithms with respect to the number of users, target load, and initial state. MI exhibits ideal population scalability. Capacity scalability of MI is the best among linear increase rules but is not ideal. AI is the best, and MI is the worst, in terms of initialization scalability.

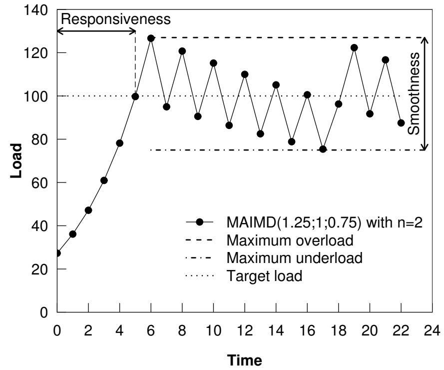



Fig. 1. Responsiveness and smoothness of MAIMD algorithms.

While the primary goal and chief contribution of this work are in extending the theory of load adjustments, Section III of our paper briefly discusses implications of the theoretical findings for the practice of congestion control. Direct practical ramifications of the asymptotic fairing analysis appear limited because the theoretical speed advantage of an MAIMD over a less smooth AIMD is only marginal and seems unrealizable in real networks. More significant for practice is our quantification of fairing speeds that reveals promising avenues for future congestion control; e.g., since reaching high fairness can take surprisingly little time, we sketch a promising protocol where load oscillations stop after all present flows discover their fair loads. Also, our analysis confirms the overall soundness of TCP design by offering theoretical rationales for using MI(2) in slow start and AIMD(1;0.5) in congestion avoidance. Finally, the theory of load adjustments exposes the performance trade-offs that explain why in trying to improve upon TCP scalability and smoothness, STCP and VCP worsen responsiveness and fairing speed.

## II. Analysis of Load Adjustments

### A. Classical Model and Propositions

In Chiu-Jain model, $n$ distributed users share a single resource that has a target load $C$. The model is synchronous and employs a discrete timescale. Every instant on the timescale represents a moment when all the users adjust their loads on the resource. At time $t$, user $i$ imposes a positive real load $l_i(t)$. Vector $\vec{l}(t) = (l_1(t), l_2(t), \ldots, l_n(t))$ captures all individual loads. The total load of the users is $L(t) = \sum_{i=1}^{n} l_i(t)$. By time $t$, the system provides all users with a uniform binary feedback

$$f(t) = \begin{cases} 0 & \text{if } L(t-1) \leq C, \\ 1 & \text{otherwise} \end{cases} \tag{1}$$

that indicates whether the total load of the users after the previous round of adjustments exceeds the target load. Except for the binary feedback $f(t)$, the system does not impart to a user any information about the resource or other users.

Chiu and Jain applied their model to analyze behavior of linear adjustment algorithms that change the load of each user $i$ as follows:

$$l_i(t) = \begin{cases} x + pl_i(t-1) & \text{if } f(t) = 0, \\ d + yl_i(t-1) & \text{otherwise} \end{cases} \tag{2}$$

where coefficients $x$, $p$, $d$, and $y$ are constant real numbers. The search for stable algorithms that converge from any initial state toward the efficient fair state where the load of every user is $\frac{C}{n}$ identified the following stability conditions: $d = 0$, $0 \leq y < 1$, $x > 0$, and $p \geq 1$, i.e., the decrease rule is purely multiplicative, and the increase rule is additive with an optional multiplicative component.

Our investigation refers to this class of stable algorithms as MAIMD (Multiplicative Additive Increase Multiplicative Decrease) and to a specific algorithm within the MAIMD class as MAIMD($p$;$x$;$y$) where $p$, $x$, $y$ are respectively the MI, AI, and MD coefficients of the algorithm. Since an MI component is optional, AIMD forms an interesting subclass of MAIMD. We denote MAIMD(1;$x$;$y$) as AIMD($x$;$y$).

The total load under MAIMD($p$;$x$;$y$) converges not to a single value but into oscillating within a finite range around the target load (see Figure 1). The size of this range represents *smoothness* $S$ of the algorithm: $S = \frac{L_{max} - L_{inf}}{C} = p - y + \frac{nx}{C}$ where $L_{max}$ and $L_{inf}$ are respectively the maximum and infimum of the total load after it reaches the target load from any initial state.

*Responsiveness* $R$ of the MAIMD($p$;$x$;$y$) algorithm is the amount of time taken by the total load to reach the target load from the initial state: $R = \left\lceil \log_p \left( \frac{(p-1)C+nx}{(p-1)L(0)+nx} \right) \right\rceil$ if $L(0) \leq C$ and $p > 1$; $R = \left\lceil \frac{C - L(0)}{nx} \right\rceil$ if $L(0) \leq C$ and $p = 1$; $R = \left\lceil \log_y \left( \frac{C}{L(0)} \right) \right\rceil$ if $L(0) > C$.

Individual loads oscillate similarly to the total load but converge infinitesimally close to each other. *Fairness* $F(t) = \min_{i=1}^{n} l_i(t) / \max_{i=1}^{n} l_i(t)$ quantifies this process of fairing. Fairness takes its values from range $[0, 1]$ and converges to 1.

An ideal algorithm would minimize both $S$ and $R$ as well as maximize the fairing speed. However, a fundamental trade-off exists between smoothness and responsiveness: values of coefficients $p$, $x$, or $y$ that improve responsiveness worsen smoothness. It is impossible to narrow down the MAIMD class to a specific algorithm with optimal responsiveness and smoothness.

With respect to the speed of fairing, Chiu and Jain observed that whereas a single decrease does not affect fairness, a single increase under MAIMD($p$;$x$;$y$) improves fairness the most when $p$ is reduced to 1. The observation led to a proposition that the AIMD subclass of MAIMD offers optimal fairing.

### *B. Asymptotic Fairing Speed*

The classical assertion of the fastest fairing under AIMD is important because it serves as the only theoretical justification for favoring AIMD over MAIMD in TCP and other prominent protocols. Gorinsky and Vin cast doubt on optimality of fairing under AIMD by showing that MAIMD($p$;$x$;$y$) with $p > 1$ can raise fairness significantly higher than AIMD($x$;$y$) after the same number of multiple adjustments [26]. However, this observation neglects two aspects of the fairing problem. First, the slower fairing after a fixed number of steps does not mean that AIMD($x$;$y$) fails to overcome the lag eventually and then provide consistently better fairness than MAIMD($p$;$x$;$y$). Second, AIMD($x$;$y$) is smoother than MAIMD($p$;$x$;$y$). Since there is the fundamental trade-off between smoothness and responsiveness, a similar trade-off might exist between smoothness and fairing speed. Then, the lag of AIMD($x$;$y$) might be due to its smoother parameter settings, rather than the absence of an MI component.

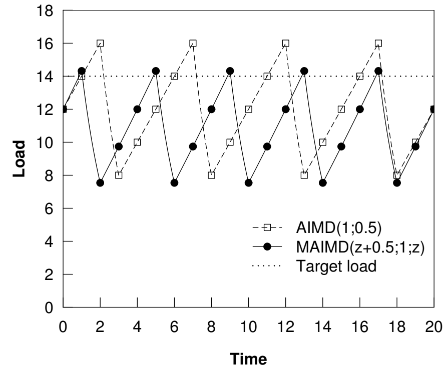



Fig. 2. Total loads during the first common 20-step period of adjustments under AIMD(1;0.5) and MAIMD($z + 0.5$;1;$z$): $z \approx 0.5266$, $C = 14$, $n = 2$, and $\vec{l}(0) = (11, 1)$.

Hence, we start our analysis by comparing asymptotic fairing of MAIMD and AIMD when the compared algorithms have equal smoothness. To reason about speeds of asymptotic fairing, we define the following notion:

***Definition 1:*** *Algorithm $X$ provides faster asymptotic fairing from initial state $\vec{l}(0)$ than algorithm $Y$ if*

$$\exists\ \tau\ \ \forall\ t > \tau \qquad F^X(t) > F^Y(t) \tag{3}$$

*where $F^X(t)$ and $F^Y(t)$ represent fairness provided at time $t$ by algorithms $X$ and $Y$ respectively.*

As fairness asymptoticly approaches 1, the traditional representation of fairness becomes inconvenient due to accumulation of nines after the decimal point. To facilitate comparison of fairness levels during asymptotic fairing, we define a new representation that remaps fairness from its traditional range $[0, 1]$ into interval $[0, \infty]$:

***Definition 2:*** *Nines-representation $N(t)$ of fairness $F(t)$ at time $t$ is*

$$N(t) = -\log_{10}(1 - F(t)). \tag{4}$$

This auxiliary representation has a simple interpretation: the integer portion of nines-representation shows the number of consecutive nines right after the decimal point in fairness. For example, the integer portion in the nines-representation of fairness 0.99925 is equal to 3, revealing that fairness has three consecutive nines right after its decimal point.

There are different ways to provide MAIMD and AIMD with equal smoothness. We begin in conformity with the classical single-step analysis where the compared algorithms have the same AI component. To compensate the smoother

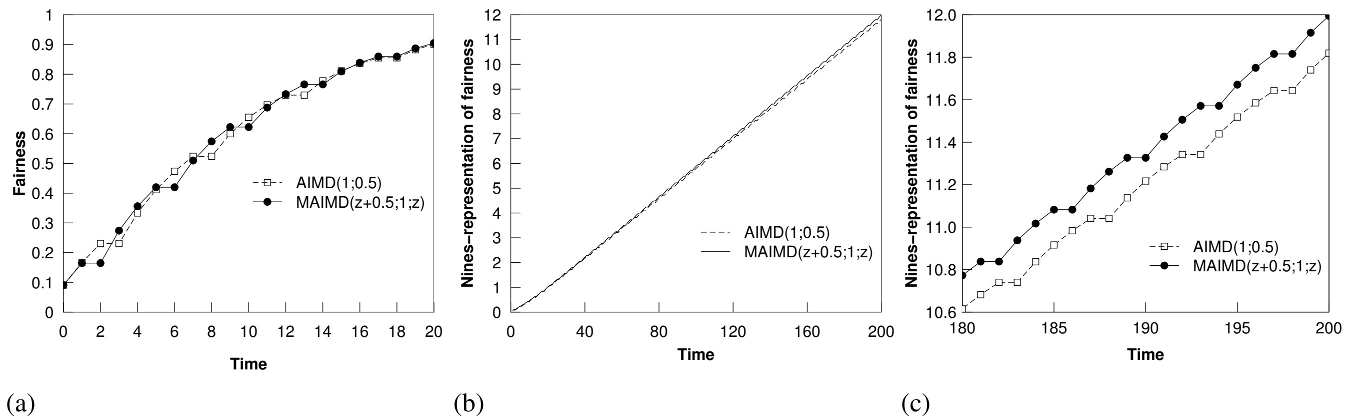



Fig. 3. Faster fairing of MAIMD($z+0.5$;1;$z$) over AIMD(1;0.5) when $z \approx 0.5266$, $C = 14$, $n = 2$, and $\vec{l}(0) = (11, 1)$: (a) slight lead after the first 20 steps, (b) first ten common periods of adjustments, (c) stable lead during the tenth common period.

increase of AIMD with a smoother decrease of MAIMD, we compare AIMD($x$;$y$) with MAIMD($p$;$x$;$z$) where $p = 1+z-y$ and $0 < y < z < 1$. Each of the two algorithms has smoothness $1 - y + \frac{nx}{C}$. Despite the assurance of a larger fairness improvement under AIMD($x$;$y$) after a single increase from the same state, MAIMD($p$;$x$;$z$) might offer faster fairing in the long run:

***Theorem 1:*** *AIMD does not guarantee the fastest asymptotic fairing among MAIMD algorithms of equal smoothness.*

To improve readability, the main body of our paper includes only summaries of the proofs for this and subsequent theorems. We relegate the proof details and supporting lemmata to the Appendix.

*Proof summary:* Consider the system with two users, target load 14, and initial state $(11, 1)$ and compare AIMD(1;0.5) with MAIMD($z+0.5$;1;$z$) where $z \approx 0.5266$ is a root of the quartic equation $12z^4 + 20z^3 + 11z^2 + 4z = 9$. The Appendix reports the exact value of $z$ in Figure 9. The algorithms have equal smoothness $\frac{9}{14}$. The total load under AIMD(1;0.5) follows a five-step cycle of four increases and one decrease: $L(5k) = 12$, $L(5k+1) = 14$, $L(5k+2) = 16$, $L(5k+3) = 8$, and $L(5k+4) = 10$ where $k$ is a nonnegative integer. The total load under MAIMD($z+0.5$;1;$z$) changes with a four-step cycle of three increases and one decrease: $L(4k) = 12$, $L(4k+1) \approx 14.319$, $L(4k+2) \approx 7.5404$, and $L(4k+3) \approx 9.7409$. Hence, the total loads under AIMD(1;0.5) and MAIMD($z+0.5$;1;$z$) have a common period of 20 steps. Referring to AIMD(1;0.5) and MAIMD($z+0.5$;1;$z$) with superscripts "A" and "M" respectively, we derive that $F^M(20k+20) > F^A(20k+20)$ if $F^M(20k) > F^A(20k)$. Since $F^M(20) > F^A(20)$, we prove by induction that $F^M(20n) > F^A(20n)$ for any positive integer $n$. Then, by Definition 1, AIMD does not guarantee the fastest asymptotic fairing among MAIMD with equal smoothness. ■

To illustrate the above proof, Figure 2 shows the first common 20-step period of the total loads under the compared algorithms. Figure 3a shows that MAIMD($z+0.5$;1;$z$) acquires a slight fairness advantage over AIMD(1;0.5) after the first common period. Although the algorithms take turns in providing better fairness early on, MAIMD($z+0.5$;1;$z$) overtakes AIMD(1;0.5) for good at time 67 and then unfailingly yields higher fairness, as Figure 3b indicates. Figure 3c shows the persistent lag of fairness under the AIMD algorithm during the tenth common period of the total loads.

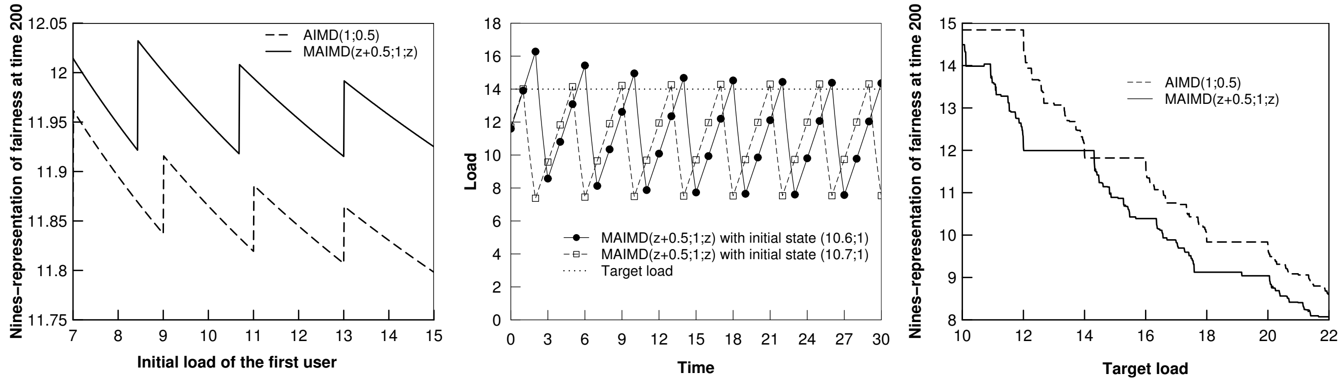
To understand reasons for the counterintuitive Theorem 1, we examine sensitivities of the fairing speed under AIMD(1;0.5) and MAIMD($z+0.5$;1;$z$) to the system configuration. Figure 4a shows that MAIMD($z+0.5$;1;$z$) outpaces AIMD(1;0.5) after 200 steps from all examined initial loads. Hence, MAIMD($z+0.5$;1;$z$) excels not because of starting in a special state that forces the total loads under the algorithms to oscillate with the common period disadvantageous for AIMD(1;0.5). Figure 4b hints at a likely reason for the resilience to the choice if the initial load: from different initial states, the total load under MAIMD converges to the same periodic pattern of oscillations. We represent this periodic pattern with a *canonical cycle*:

***Definition 3:*** *A canonical cycle of an adjustment algorithm is the shortest finite repeating sequence of total loads under the algorithm, starting at the smallest value.*

For example, the canonical cycle of AIMD(1;0.5) in the above system with $C = 14$ and $n = 2$ is (8,10,12,14,16). Convergence to a canonical cycle is a property of all MAIMD algorithms:

***Theorem 2:*** *The total load under an MAIMD algorithm converges to a unique canonical cycle.*

*Proof summary:* Consider a system with target load $C$, $n$ users, and MAIMD($p$;$x$;$z$) control. *Case 1*: If $pzC + nx > C$, then exactly one increase follows each decrease sequence, and lengths of all decrease sequences that follow an increase differ by at most one step. Let $m$ denote $\left\lceil \log_z \frac{C}{pzC+nx} \right\rceil$. If $m \geq \log_z \frac{C}{pC+nx}$, then the total load converges to the periodic pattern of one increase and $m$ decreases. If $m < \log_z \frac{C}{pC+nx}$, then each decrease sequence contains either $m$ or $m+1$ steps, depending on whether the underload after the previous decrease sequence is at most $\frac{z^{-m}C-nx}{p}$ or above. In this range of $m$, load oscillations also converge from any initial state to




(a) (b) (c)

Fig. 4. Sensitivities of the fairing speed under AIMD(1;0.5) and MAIMD($z + 0.5$;1;$z$) to the system configuration, $z \approx 0.5266$, $n = 2$, and $l_2(0) = 1$: (a) dependence on the initial state when $C = 14$; (b) convergence to the canonical cycle; (c) sensitivity to the target load when $l_1(0) = 11$.

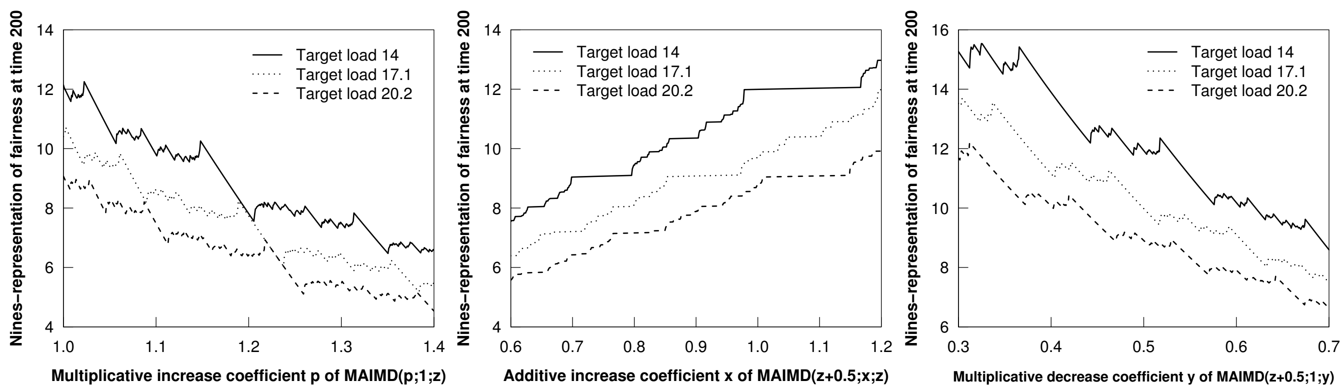



Fig. 5. Sensitivities of the fairing speed to MAIMD coefficients: $z \approx 0.5266$, $n = 2$, and $\vec{l}(0) = (11, 1)$.

a unique periodic pattern. *Case 2* (which covers all settings that Case 1 does not): If $z(pC + nx) \leq C$, then exactly one decrease follows each increase sequence, and the proof mirrors the reasoning in Case 1.

As per the above, load oscillations converge from any initial state to the same periodic pattern. With $T$ denoting the period duration, we express the total load at time $t = \tau + kT$ as

$$L(t) = q^k L(\tau) + \frac{1 - q^k}{1 - q} r \tag{5}$$

where $\tau$ represents an initial transient (which depends on the initial state), $k$ is the number of subsequent periods, and values of $q$ and $r$ depend on the phase within the period but not on the initial load. Since $q^k \to 0$ and $L(t) \to \frac{r}{1-q}$ when $t \to \infty$, the total load converges from any initial state to values $\frac{r}{1-q}$ that form a unique canonical cycle. ■

Theorem 2 suggests that the asymptotic fairing speed of an MAIMD is determined by the canonical cycle, rather than smoothness. This leads us to the insight that MAIMD($p$;$x$;$y$) with $p > 1$ might converge to fairness faster than an AIMD with worse smoothness in terms of both increase and decrease:

***Theorem 3:*** *An MAIMD algorithm with an MI component can provide faster asymptotic fairing than an AIMD algorithm with worse smoothness of both increase and decrease.*

*Proof summary:* For the same system configuration as in the proof of Theorem 1, we compare the asymptotic fairing speeds of AIMD(1;0.5) and MAIMD(1.00014;0.999;$y$) where $y = \frac{14001-(100.014^3-10^6)\cdot\frac{999}{70}}{1.00014^3\cdot 14001} \approx 0.5716$. The two proofs are similar in general but this proof overcomes a new subtle challenge that arises because MAIMD(1.00014;0.999;$y$) converges to the values of its canonical cycle asymptotically instead of adhering to them exactly. ■

The presence of the MI component in the MAIMD is not essential for the above proof. For instance, AIMD($x$;$y$) with $x = 0.9999$ and $y = \frac{13336}{23335} \approx 0.5715$ also provides faster asymptotic fairing than AIMD(1;0.5) despite being smoother in terms of both increase and decrease.

***Corollary 1:*** *There exists no fundamental trade-off between the asymptotic fairing speed and smoothness.*

Unlike the initial state, the target load does affect the qualitative outcome of comparing the fairing speeds of the algorithms from the proof of Theorem 1. Figure 4c shows that MAIMD($z$+0.5;1;$z$) supplies faster fairing than AIMD(1;0.5) only in a narrow region around the target load 14. Furthermore, raising the target load slightly beyond this region makes the canonical cycle of MAIMD($z$+0.5;1;$z$) change frequently. The canonical cycle of AIMD(1;0.5) exhibits similar sensitivities to the target load. For instance, when the target load reduces from 14 to 13.99, the canonical cycle of AIMD(1;0.5) changes from (8,10,12,14,16) to a 39-entry sequence that starts at its minimum value 6.996, peaks 8 times, and contains the maximum value 15.984 in the 35-th entry.

We now study sensitivities of the fairing speed to MAIMD coefficients. Figure 5 agrees with Corollary 1 that no fundamental trade-off exists between smoothness and asymptotic fairing speed. The graphs also show definite, though non-monotonic, dependencies of the fairing speed on the MAIMD coefficients that determine smoothness. In general, the fairing speed improves as the MI or MD coefficient decreases, or the AI coefficient increases.

Figure 3b indicates that after an initial transient, average improvement of fairness in its nines-representation occurs at a stable rate. Hence, to characterize the asymptotic fairing speed of MAIMD algorithms, we introduce the following metric called a *fairing factor*:

***Definition 4:*** *Fairing factor $G$ of an MAIMD algorithm is*

$$G = \lim_{t \to \infty} \frac{N(t) - N(0)}{t}. \quad (6)$$

Then, we express the fairing factor of an MAIMD in terms of properties of the canonical cycle:

***Theorem 4:*** *The fairing factor of MAIMD(p;x;z) equals*

$$G = -\frac{I \log_{10} p + D \log_{10} z}{I + D} \quad (7)$$

*where $I$ and $D$ are respectively the number of increases and decreases in the canonical cycle of the algorithm.*

*Proof summary:* Denoting the maximum individual load at time $t$ as $l_{max}(t)$, we express improvement of fairness after one increase as

$$N(t) - N(t-1) = \log_{10} \frac{l_{max}(t)}{l_{max}(t-1)} - \log_{10} p \quad (8)$$

and lack of change in fairness after one decrease as

$$N(t) - N(t-1) = \log_{10} \frac{l_{max}(t)}{l_{max}(t-1)} - \log_{10} z. \quad (9)$$

Then, average improvement of fairness from time 0 to $t$ is

$$\frac{N(t) - N(0)}{t} = \frac{\log_{10} \frac{l_{max}(t)}{l_{max}(0)} - i(t) \log_{10} p - d(t) \log_{10} z}{t}$$

where $i(t)$ and $d(t)$ are respectively the overall number of increases and decreases from time 0 to $t$. Since load oscillations converge to the canonical cycle, we apply Definition 4 to derive Equation 7. ■

It is interesting that the AI coefficient $x$ affects the fairing factor of MAIMD($p$;$x$;$z$) only indirectly through the numbers of increases and decreases in the canonical cycle. Figure 6 illustrates the accuracy of the fairing factor as a predictor of average fairness improvement under MAIMD($p$;1;$z$) where $z \approx 0.5266$ and $p$ varies from 1 to 1.1. For time interval [0,50], the prediction is significantly off due to the initial transient. Extension of the averaging interval to [0,200] subdues the contribution from the initial transient and improves the prediction dramatically. Finally, shifting the averaging interval

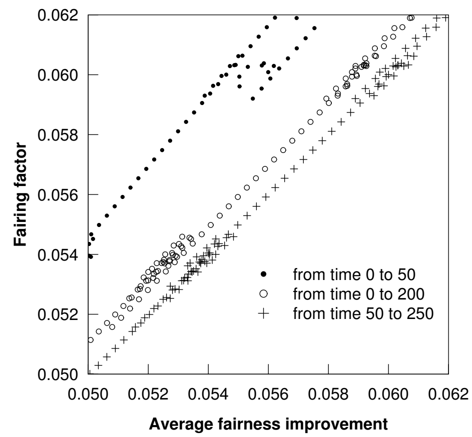



Fig. 6. Accuracy of the fairing factor as a predictor of average improvement in the nines-representation of fairness under MAIMD($p$;1;$z$) where $p$ varies from 1 to 1.1, $z \approx 0.5266$, $n = 2$, $C = 14$, and $\vec{l}(0) = (11, 1)$.

to [50,250] yields an almost perfect prediction, indicating that the fairing factor is an appropriate metric for representing the asymptotic fairing speed of an MAIMD algorithm.

### *C. Trade-Off between Responsiveness and Smoothness*

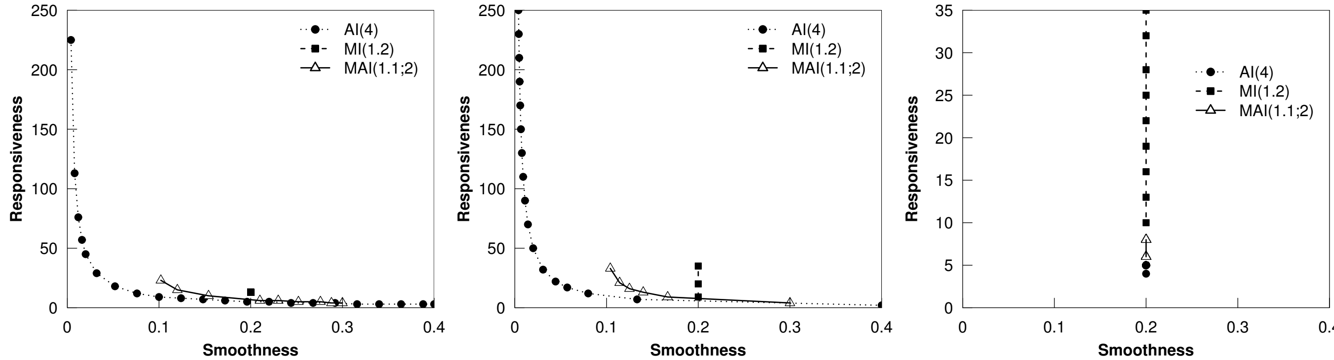
In this section, we examine MAIMD and AIMD algorithms when they have equal smoothness of both increase and decrease, i.e., we compare AIMD($x$;$y$) and MAIMD($p$;$v$;$y$) where $x = v + \frac{(p-1)C}{n}$ and $1 < p < 1 + \frac{nx}{C}$. Both algorithms overshoot the target load by at most $nx$.

First, we compare increase rules AI($x$) and MAI($p$;$v$). Focusing on the increase components preserves the fundamental trade-off between responsiveness and smoothness: values of coefficients $x$, $p$, and $v$ that improve responsiveness of increase worsen smoothness of increase. Comparison of AI($x$) and MAI($p$;$v$) under equal smoothness produces the following surprising theorem, the formulation of which also covers MI rules:

***Theorem 5:*** *AI guarantees the best responsiveness among linear increase rules of equal smoothness.*

*Proof summary:* Let $M$ be an increase rule MAI($p$;$v$) or MI($p$); in this proof, MI($p$) is equivalent to MAI($p$;0). Consider the increase rule $A$ that uses AI($x$) where $x = v + \frac{(p-1)C}{n}$. Rules $M$ and $A$ have equal smoothness $p - 1 + \frac{nv}{C}$. Let us denote responsiveness of rule $A$ as $R$. Supposing $L^M(t-1) \leq L^A(t-1) < C$ for any $t$ from 1 to $R-1$, we derive $L^M(t) \leq L^A(t) < C$. Since $L^M(0) = L^A(0)$, we establish by induction that $L^M(t) \leq L^A(t) < C$ for any time $t$ from 0 through $R-1$. Thus, responsiveness of rule $M$ is at least $R$. Hence, AI guarantees the best responsiveness among MAI and MI with equal smoothness. ■

We called Theorem 5 above a surprising result because MI is often perceived as more aggressive in acquiring available capacity, i.e., more responsive. However, responsiveness is subject to the fundamental trade-off with smoothness. Theorem 5 states that with smoothness being equal, AI is at least as responsive as any linear increase rule with an MI component.




(a) (b) (c)

Fig. 7. Scalabilities of AI, MAI, and MI rules: (a) population scalability when $L(0) = 100$, $C = 1000$, and $n$ changes from 1 to 100; (b) capacity scalability when $L(0) = 100$, $n = 50$, and $C$ changes from 500 to 50000; (c) initialization scalability when $n = 50$, $C = 1000$, and $L(0)$ changes from 2 to 200.

The result has an intuitive explanation: while AI($x$) always responds to underload with increases of constant size, per-increase gains under MAI($p$;$v$) or MI($p$) are smaller when applied to smaller loads; thus, having the same overshoot after reaching the target load implies that AI($x$) needs the smallest number of increase steps to reach the target from any underload.

Since AIMD($x$;$y$) and MAIMD($p$;$v$;$y$) use the same decrease rule MD($y$), their decrease behaviors are identical. Nevertheless, striving for comprehensiveness, we examine MD in the broader context of MAD (Multiplicative Additive Decrease) and AD (Additive Decrease) rules and derive:

***Theorem 6:*** *MD guarantees the best responsiveness among linear decrease rules of equal smoothness.*

*Proof summary:* Let $A$ denote a decrease rule MAD($q$;$d$) or AD($d$); in this proof, AD($d$) is equivalent to MAD(1;$d$). To avoid undershoot to negative values, additive decrease is subject to truncation to zero [2]. Hence, smoothness of rule $A$ is $\min\{1-q-\frac{nd}{C};1\}$. Then, consider the decrease rule $M$ that uses MD($y$) with $y = \max\{q+\frac{nd}{C};0\}$ and has equal smoothness. Let us denote responsiveness of rule $M$ as $R$. Supposing $C < L^M(t-1) \leq L^A(t-1)$ for any $t$ from 1 to $R-1$, we derive $C < L^M(t) \leq L^A(t)$. Since $L^M(0) = L^A(0)$, we establish by induction that $C < L^M(t) \leq L^A(t)$ for any time $t$ from 0 through $R-1$. Thus, responsiveness of rule $A$ is at least $R$. Hence, MD guarantees the best responsiveness among MAD and AD rules with equal smoothness. ■

The duo of Theorems 5 and 6 establishes that AIMD provides the best trade-off between responsiveness and smoothness:

***Corollary 2:*** *AIMD guarantees the best responsiveness among linear adjustment algorithms with equal smoothness of both increase and decrease.*

### *D. Scalabilities*

According to Theorem 5, AI guarantees the best responsiveness among MAI and MI with equal smoothness. However, setting the coefficient $x$ for AI($x$) to achieve equal smoothness with an MAI($p$;$v$) or MI($p$) rule requires knowledge of $\frac{C}{n}$. Since the users do not know this value, we now examine sensitivities of responsiveness and smoothness to the following three system parameters: (1) number $n$ of users, (2) target load $C$, and (3) initial load $L(0)$. We evaluate scalabilities of increase rules along each of these three dimensions separately and refer to respective scalability properties as *population scalability*, *capacity scalability*, and *initialization scalability*.

A parameter change can affect both smoothness and responsiveness. Instead of combining these two aspects of efficiency into a single metric, we propose to study the scalabilities with scatter diagrams of smoothness and responsiveness. Subsequently, we refer to these diagrams as *smoothness-responsiveness (SR) diagrams*.

Figure 7a presents an SR diagram for AI(4), MAI(1.1;2), and MI(1.2) when $L(0) = 100$, $C = 1000$, and $n$ changes from 1 to 100. First of all, the graph confirms that AI offers the best trade-off between smoothness and responsiveness. When $n = 50$, the three rules have equal smoothness 20% but responsiveness of AI(4) is 5, of MAI(1.1;2) is 7, and of MI(1.2) is 13. Similarly, smoothness of AI(4) is 7.2%, of MAI(1.1;2) is 12.8%, and of MI(1.2) is 20% when the rules have equal responsiveness 13 [this occurs when AI(4) serves 18 users, and MAI(1.1;2) serves 14 users]. The SR diagram also illustrates that MI rules provide the best population scalability. MI(1.2) has constant smoothness 20% and responsiveness 13 regardless of $n$. Moreover, MI(1.2) is smoother than AI(4) with $n \geq 51$ and more responsive than AI(4) with $n \leq 17$. By displaying a lesser dispersion of points for MAI(1.1;2) than for AI(4), the diagram reveals that population scalability of MAI rules is better than with AI but is still worse than the ideal population scalability of MI.

Figure 7b examines capacity scalability by plotting an SR diagram for the same three rules when $L(0) = 100$, $n = 50$, and $C$ changes from 500 to 50000. The outcome is qualitatively similar to the observed for population scalability. Capacity scalability is the worst with AI, intermediate with MAI, and best with MI. However, Figure 7b also reveals an important difference: while MI(1.2) does support constant smoothness 20% regardless of $C$, responsiveness of MI(1.2) worsens from 9 to 35 as the target load increases. Hence, capacity scalability is not ideal even under MI.

Finally, Figure 7c reports an SR diagram to capture initialization scalability of AI(4), MAI(1.1;2), and MI(1.2) when $n = 50$, $C = 1000$, and $L(0)$ changes from 2 to 200. In these settings, the rules maintain equal smoothness 20%. However, MI(1.2) shows a much greater sensitivity to lowering the initial load: responsiveness worsens from 9 to 35 for MI(1.2), from 6 to 8 for MAI(1.1;2), and only from 4 to 5 for AI(4). At the first glance, the results might seem surprising because responsiveness of MI is logarithmic whereas responsiveness of AI is linear. The explanation is simple: responsiveness of MI($p$) depends on ratio $\frac{C}{L(0)}$ while responsiveness of AI($x$) is a function of difference $C - L(0)$; hence, as the initial load approaches 0, responsiveness of MI($p$) worsens without bound but responsiveness of AI is limited from above by $\left\lceil \frac{C}{nx} \right\rceil$.

## III. Implications for Congestion Control

While extending the theory of load adjustments constitutes the primary focus and chief contribution of our paper, we now briefly discuss implications of our theoretical findings for the practice of congestion control. Sections III-A, III-B, and III-C review existing protocols and suggest future avenues for congestion control along three respective dimensions of our analysis: asymptotic fairing speed, trade-off between responsiveness and smoothness, and scalabilities of MAIMD algorithms.

### A. Asymptotic Fairing Speed

The asymptotic fairing analysis established the most counterintuitive result that an MAIMD can provide faster asymptotic fairing than a less smooth AIMD. Also, the analysis was by far the most challenging intellectually, e.g., in the context of proving the existence of the unique canonical cycle and predicting the fairing speed with the fairing factor. On the other hand, direct practical ramifications of the asymptotic analysis appear limited. The fairness advantage acquired by an MAIMD over a less smooth AIMD is only marginal and arises due to differences in their canonical cycles, rather than from a fundamental necessity of the MI component. Besides, since the canonical cycle depends on the target load and number of users, choosing an MAIMD to optimize the asymptotic fairing speed is practically infeasible, while any change in $C$ or $n$ is likely to make the chosen algorithm suboptimal. Even modest random noise in a real network is capable of subverting such optimization. Hence, AIMD remains the most prudent practical choice among MAIMD algorithms from the fairing-speed perspective.

Indirect but more significant practical benefits of our asymptotic analysis lie in the quantification of the fairing speed. In many realistic scenarios, an MAIMD or AIMD algorithm converges to high fairness surprisingly quickly. Further oscillations of the loads yield no meaningful improvement in fairness but keep causing such undesirable effects as long queuing at the bottleneck link or low utilization of the link capacity. This observation led us to design Multimodal Control Protocol (MCP) where a flow maintains steady transmission after discovering its fair efficient rate [27]. By allocating few bits per packet to communicate control information between routers and hosts, MCP keeps queuing low, avoids packet losses, and utilizes the bottleneck link efficiently in the stable mode. To ensure convergence to fairness, MCP incorporates an innovative mechanism that enables a flow to urge all flows sharing its bottleneck links to operate in a fairing mode, which is dedicated to fairness improvement.

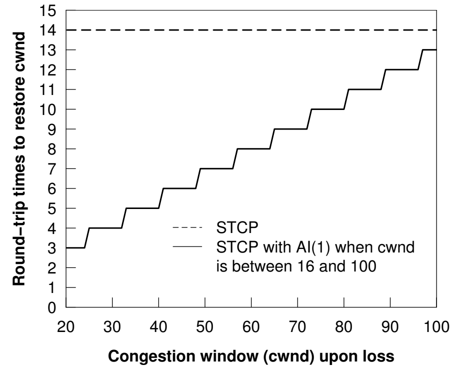


Fig. 8. Slow responsiveness of STCP in the congestion-avoidance mode when the congestion window is under 100.

Our quantification of the fairing speed also suggests that if a congestion control protocol prefers to employ a single simple algorithm for load adjustments, then an MAIMD with an MI component might constitute a reasonable choice. While likely to offer slower asymptotic fairing than an AIMD, the MAIMD might take comparable time to supply high fairness and be a better overall pick due to strengths in other properties such as population and capacity scalabilities.

By exposing the definite correlation between better smoothness and slower fairing, the fairing analysis also sheds light on performance of existing protocols. For instance, while TCP in congestion avoidance relies on AIMD(1;0.5), VCP opts for AIMD(1;0.875) to improve on TCP smoothness. In agreement with our theory, the larger MD coefficient gives VCP not only smoother load oscillations but also slower convergence to fairness.

### B. Trade-Off between Responsiveness and Smoothness

Trying to understand deeper the fundamental trade-off between responsiveness and smoothness, we showed that AIMD guarantees the best responsiveness among linear adjustment algorithms with equal smoothness of both increase and decrease. This surprising result represents an additional theoretical argument for using AIMD in TCP congestion avoidance.

More interestingly, the theorem reveals that MI rules are not inherently more responsive than AI rules. Figure 8 illustrates this for STCP which strives to improve on TCP scalability by using MI(1.01) instead of AI(1) in congestion avoidance. Setting the MI coefficient to 1.01 hampers responsiveness of increases when the load is under 100. STCP takes up to eleven extra round-trip times to restore the congestion window after a loss than an STCP modification with AI(1) would when the congestion window is between 16 and 100 [STCP uses MI(2) for increases from lower loads].

Again, VCP is an example of recent proposals that improve upon one property of TCP congestion control at the expense of another property. While TCP uses MI(2) in slow start, VCP employs MI(1.0625) whenever the bottleneck link utilization is below 80%. Due to the smaller MI coefficient, VCP exercises smoother control but exhibits worse responsiveness than TCP in slow start.

### C. Scalabilities

With respect to responsiveness and fairing speed under equal smoothness, AIMD offers the best basis for practical congestion control among linear adjustment algorithms. However, our analysis showed that AI is inferior to MI in terms of population and capacity scalabilities. This finding explains the multimodal design of TCP: MI(2) in slow start is for scalable acquisition of available capacity while AIMD(1;0.5) in congestion avoidance supplies fast fairing with increases that are at least as smooth as in slow start.

When the available capacity surges due to an increase in the bottleneck link capacity or dramatic decline of competing traffic, remaining TCP flows keep operating in congestion avoidance and suffer from the poor scalabilities of AI. This deficiency of TCP design is captured by the theory of load adjustments and constitutes the main reason for continued research on alternative controls.

## IV. Conclusion

This paper extended the classical Chiu-Jain theory of load adjustments by analyzing the asymptotic fairing speed, responsiveness, smoothness, and scalabilities of linear adjustment algorithms. We proved that an MAIMD can provide faster asymptotic fairing than a less smooth AIMD. Furthermore, we discovered that loads under a specific MAIMD converge from any initial state to a unique canonical cycle. While imperfectly correlated with smoothness, the canonical cycle predicts the asymptotic fairing speed reliably. We quantified the asymptotic fairing speed with a fairing factor and expressed this metric as a function of the numbers of increases and decreases in the canonical cycle. Besides, we proved that AIMD guarantees the best responsiveness among linear adjustment algorithms with equal smoothness of both increase and decrease. Then, we introduced smoothness-responsiveness diagrams to investigate population, capacity, and initialization scalabilities of linear adjustments. Our analysis showed that MI exhibits the best capacity scalability and ideal population scalability while AI offers the best initialization scalability.

Due to the general nature of the theory, the findings of our analysis are potentially applicable to a great variety of problems in networking and distributed systems. At the end of this paper, we just briefly discussed some implications for the practice of congestion control. In particular, we reviewed TCP, STCP, and VCP in the light of the theory and discussed promising directions for future congestion control.

## References


[1] M. Allman, V. Paxson, and W. Stevens, "TCP Congestion Control," RFC 2581, April 1999.

[2] D. Chiu and R. Jain, "Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks," *Journal of Computer Networks and ISDN*, vol. 17, no. 1, pp. 1–14, June 1989.

[3] M. Welsh and D. Culler, "Adaptive Overload Control for Busy Internet Servers," in *Proceedings USENIX Conference on Internet Technologies and Systems*, March 2003.

[4] S. Floyd, R. Gummadi, and S. Shenker, "Adaptive RED: An Algorithm for Increasing the Robustness of RED," *www.icir.org/floyd/papers/adaptiveRed.pdf*, August 2001.

[5] M. Heusse, F. Rousseau, R. Guillier, and A. Duda, "Idle Sense: An Optimal Access Method for High Throughput and Fairness in Rate Diverse Wireless LANs," in *Proceedings ACM SIGCOMM 2005*, August 2005.

[6] J. Byers, G. Horn, M. Luby, M. Mitzenmacher, and W. Shaver, "FLID-DL: Congestion Control for Layered Multicast," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 8, pp. 1558 – 1570, October 2002.

[7] V. Jacobson, "Congestion Avoidance and Control," in *Proceedings ACM SIGCOMM 1988*, August 1988.

[8] G. Appenzeller, I. Keslassy, and N. McKeown, "Sizing Router Buffers," in *Proceedings ACM SIGCOMM 2004*, September 2004.

[9] E. Altman, K. Avrachenkov, and B. Prabhu, "Fairness in MIMD Congestion Control Algorithms," *Telecommunication Systems*, vol. 30, no. 4, pp. 387–415, December 2005.

[10] S. Gorinsky, "Feedback Modeling in Internet Congestion Control," in *Proceedings Next Generation Teletraffic and Wired/Wireless Advanced Networking (NEW2AN 2004)*, February 2004.

[11] D. Katabi, M. Handley, and C. Rohrs, "Congestion Control for High Bandwidth-Delay Product Networks," in *Proceedings ACM SIGCOMM 2002*, August 2002.

[12] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-Based Congestion Control for Unicast Applications," in *Proceedings ACM SIGCOMM 2000*, August 2000.

[13] S. Low, "A Duality Model of TCP and Queue Management Algorithms," *IEEE/ACM Transactions on Networking*, vol. 11, no. 4, pp. 525–536, August 2003.

[14] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP Throughput: A Simple Model and its Empirical Validation," in *Proceedings ACM SIGCOMM 1998*, September 1998.

[15] T. Kelly, "Scalable TCP: Improving Performance in Highspeed Wide Area Networks," *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 2, pp. 83–91, April 2003.

[16] Y. Xia, L. Subramanian, I. Stoica, and S. Kalyanaraman, "One More Bit Is Enough," in *Proceedings ACM SIGCOMM 2005*, August 2005.

[17] L. Qiu, Y. Zhang, and S. Keshav, "Understanding the Performance of Many TCP Flows," *Computer Networks*, vol. 37, no. 3-4, pp. 277–306, November 2001.

[18] S. Floyd and E. Kohler, "Internet Research Needs Better Models," in *Proceedings HotNets-I*, October 2002.

[19] T. Anderson, L. Peterson, S. Shenker, and J. Turner, "Overcoming the Internet Impasse through Virtualization," *IEEE Computer*, vol. 38, no. 4, pp. 34–41, April 2005.

[20] R. Jain, *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*, 1st ed. John Wiley & Sons, April 1991.

[21] J. Kurose and K. Ross, *Computer Networking: A Top-Down Approach Featuring the Internet*, 3rd ed. Addison-Wesley, May 2004.

[22] L. Peterson and B. Davie, *Computer Networks: A Systems Approach*, 3rd ed. Morgan Kaufmann, October 2003.

[23] D. Bansal and H. Balakrishnan, "Binomial Congestion Control Algorithms," in *Proceedings IEEE INFOCOM 2001*, April 2001.

[24] S. Jin, L. Guo, I. Matta, and A. Bestavros, "TCP-friendly SIMD Congestion Control and Its Convergence Behavior," in *Proceedings IEEE ICNP 2001*, November 2001.

[25] D. Loguinov and H. Radha, "Increase-Decrease Congestion Control for Real-time Streaming: Scalability," in *Proceedings IEEE INFOCOM 2002*, June 2002.

[26] S. Gorinsky and H. Vin, "Additive Increase Appears Inferior," *www.arl.wustl.edu/~gorinsky/pdf/tr2000-18.pdf*, Department of Com-

$$z = \frac{1}{2}\left(\sqrt{\frac{1}{6} + \frac{1415}{36\sqrt[3]{24\sqrt{4978461}-5869}} - \frac{\sqrt[3]{24\sqrt{4978461}-5869}}{36} + \frac{8}{27\sqrt{\frac{1}{12} - \frac{1415}{36\sqrt[3]{24\sqrt{4978461}-5869}} + \frac{\sqrt[3]{24\sqrt{4978461}-5869}}{36}}}} - \sqrt{\frac{1}{12} - \frac{1415}{36\sqrt[3]{24\sqrt{4978461}-5869}} + \frac{\sqrt[3]{24\sqrt{4978461}-5869}}{36}} - \frac{5}{6}\right)$$

Fig. 9. Multiplicative decrease coefficient $z$ for MAIMD($z+0.5$;1;$z$) in the proof of Theorem 1 ($z \approx 0.5266$).

puter Sciences, University of Texas at Austin, Tech. Rep. TR2000-18, May 2000.

[27] M. Podlesny and S. Gorinsky, "Multimodal Congestion Control for Low Stable-State Queuing," *www.arl.wustl.edu/~gorinsky/pdf/WUCSE-TR-2006-41.pdf*, Department of Computer Science and Engineering, Washington University in St. Louis, Tech. Rep. WUCSE-2006-41, August 2006.

## APPENDIX

***Lemma 1:*** *Fairness $F(t)$ under MAIMD equals $l_k(t)/l_j(t)$ where $k$ and $j$ identify users with the minimum and maximum initial loads respectively.*

*Proof:* By definition, $F(t) = \min_{i=1}^{n} l_i(t) / \max_{i=1}^{n} l_i(t)$. Since $p \geq 1$ and $y \geq 0$ in MAIMD($p$;$x$;$y$), adjustments do not change the order of individual loads: $l_i \geq l_m$ implies $x+pl_i \geq x+pl_m$ and $yl_i \geq yl_m$. Hence, $F(t) = l_k(t)/l_j(t)$ where $k$ and $j$ identify users with the minimum and maximum initial loads respectively, i.e., $l_k(0) = \min_{i=1}^{n} l_i(0)$ and $l_j(0) = \max_{i=1}^{n} l_i(0)$. ■

***Proof details for Theorem 1:*** Since fairness $F(t)$ equals $l_2(t)/l_1(t)$ according to Lemma 1, and the total load at time $t = 20k$ is $L(t) = l_1(t) + l_2(t) = 12$ under both algorithms, we can express the user loads as

$$l_1(20k) = \frac{12}{F(20k)+1} \quad \text{and} \quad l_2(20k) = \frac{12F(20k)}{F(20k)+1}. \quad (10)$$

From time $20k$ to time $20k+20$, the twenty adjustments transform any load $l(20k)$ under AIMD(1;0.5) to

$$l(20k+20) = pl(20k) + r \quad (11)$$

where

$$p = 0.0625 \quad \text{and} \quad r = 5.625 \quad (12)$$

and under MAIMD($z+0.5$;1;$z$) to

$$l(20k+20) = ql(20k) + s \quad (13)$$

where

$$0.06003 < q < 0.060031 \quad \text{and} \quad 5.6398 < s < 5.6399. \quad (14)$$

Supposing that

$$F^M(20k) = G \quad \text{and} \quad F^A(20k) = H \quad (15)$$

where

$$0 \leq H \leq G < 1, \quad (16)$$

we derive:

$$\begin{aligned}
& F^M(20k+20) - F^A(20k+20) \\
= \quad & \{\text{ Lemma 1 }\} \\
& \frac{l_2^M(20k+20)}{l_1^M(20k+20)} - \frac{l_2^A(20k+20)}{l_1^A(20k+20)} \\
= \quad & \{\text{ Equations 11 and 13 }\} \\
& \frac{ql_2^M(20k)+s}{ql_1^M(20k)+s} - \frac{pl_2^A(20k)+r}{pl_1^A(20k)+r} \\
= \quad & \{\text{ Equations 10 and 15 }\} \\
& \frac{q\frac{12G}{G+1}+s}{q\frac{12}{G+1}+s} - \frac{p\frac{12H}{H+1}+r}{p\frac{12}{H+1}+r} \\
= \quad & \\
& \frac{12\big((ps-qr)(1-GH) + (12pq+ps+qr)(G-H)\big)}{(12q+s+sG)(12p+r+rH)} \\
> \quad & \{\text{ Equations 12 and Inequalities 14 }\} \\
& \frac{0.17775(1-GH) + 8.8221(G-H)}{(6.3603+5.6399G)(6.375+5.625H)} \\
> \quad & \{\text{ Inequalities 16 }\} \\
& 0.
\end{aligned}$$

Therefore, we establish $F^M(20k+20) > F^A(20k+20)$ if $F^M(20k) > F^A(20k)$. ■

***Proof details for Theorem 2:*** In Part I of the proof, we show that oscillations of the total load converge from any initial state to a unique periodic pattern of increases and decreases. Then, Part II proves that the total load under this periodic pattern of adjustments converges to values forming a canonical cycle.

*Part I:* First, we consider the settings where an increase after reaching the target load is at least as potent as a decrease.

*Case 1:* If $pzC + nx > C$, then exactly one increase follows each decrease sequence. In a decrease sequence between two increases, the number of steps belongs to interval $\left[\log_z \frac{C}{pzC+nx}; 1 + \log_z \frac{C}{pC+nx}\right)$. The length of this interval is between 1 and 2. Hence, lengths of all such decrease sequences differ by at most one step. Let $m$ denote $\left\lceil \log_z \frac{C}{pzC+nx} \right\rceil$.

*Case 1.1:* If $m \geq \log_z \frac{C}{pC+nx}$, then each decrease sequence contains exactly $m$ steps. Hence, oscillations of the total load converge from any initial state to the periodic pattern of one increase and $m$ decreases.

*Case 1.2:* If $m < \log_z \frac{C}{pC+nx}$, then each decrease sequence contains either $m$ or $m+1$ steps, depending on whether the underload after the previous decrease sequence is at most $L = \frac{z^{-m}C-nx}{p}$ or above. When the current total load belongs to $(zC;L]$, the subsequent increase is followed by $m$ decreases, but when the load lies in $(L;C]$, the increase is followed by $m+1$ decreases.

Interval $(zC;C]$ consists of underloads that are possible after the total load reaches the target. A single increase-decrease oscillation splits $(zC;C]$ into two intervals of underloads: oscillating with $m$ decreases lifts $(zC;L]$ into interval $\left(z^{m+1}pC + z^m nx; C\right]$, while oscillating with $m+1$ decreases lowers $(L;C]$ into interval $\left(zC; z^{m+1}pC + z^{m+1}nx\right]$. A gap with width $(1-z)z^m nx$ separates the two created intervals. Hence, the cumulative coverage by possible underloads reduces after the oscillation.

Subsequent oscillations make similar impacts on intervals of possible underloads. Let $(f;g]$ denote an interval of underloads before an oscillation. Then, the oscillation transforms the interval as follows:

- If $L \in (f;g)$, the oscillation splits interval $(f;g]$ into two intervals: $(z^m(pf+nx);C]$ at the very top and $\left(zC; z^{m+1}(pg+nx)\right]$ at the very bottom of the original interval $(zC;C]$ of possible underloads.
- If $L \geq g$, the oscillation lifts interval $(f;g]$ into interval $(z^m(pf+nx); z^m(pg+nx)]$.
- If $L \leq f$, then the oscillation lowers $(f;g]$ into interval $\left(z^{m+1}(pf+nx); z^{m+1}(pg+nx)\right]$.

Each oscillation reduces the cumulative length of possible underload intervals. A finite number of oscillations places $L$ inside a gap between two intervals or at the upper border $g$ of some interval $(f;g]$. At this point, the process of splitting stops, and subsequent oscillations do not change the number of the intervals. Furthermore, since a gap forms under an interval only when an oscillation lifts the interval from the very bottom of $(zC;C]$, each of the stabilized number of intervals is reachable from every other interval through a finite sequence of oscillations. Hence, the total load converges from any initial state to the same periodic pattern of adjustments.

*Case 2* (which covers all settings that Case 1 does not): If $z(pC+nx) \leq C$, then exactly one decrease follows each increase sequence. The proof mirrors the argument in Case 1. After the total load reaches the target, lengths of all increase sequences differ by at most one step. In Case 2.1, each increase sequence contains exactly $j$ steps. In Case 2.2, each increase sequence contains either $j$ or $j+1$ steps, but load oscillations converge from any initial state to a unique periodic pattern in this case as well.

*Part II:* With $T$ denoting the period duration, we express the total load at any time $t = \tau + kT$ as

$$L(t) = qL(t-T) + r \tag{17}$$

where $\tau$ represents an initial transient (which depends on the initial state and includes the phase within the period), $k$ is the number of subsequent periods, and values of $q$ and $r$ depend on the phase within the period but not on the initial load. A series of $k$ periods transforms the total load into:

$$L(t) = q^k L(\tau) + \frac{1-q^k}{1-q} r. \tag{18}$$

As time advances, the contribution from $L(\tau)$ into the current load diminishes, and $L(t)$ becomes shaped by the cumulative impact of intermediate additive increases, i.e., $q^k \to 0$ and $L(t) \to \frac{r}{1-q}$ when $t \to \infty$. Hence, the total load converges from any initial state to values $\frac{r}{1-q}$ that form a unique canonical cycle. ■

***Proof details for Theorem 3:*** According to Theorem 2, the total load under MAIMD(1.00014;0.999;$y$) converges to a unique canonical cycle, which is: $L(4k) \approx 10.0019$, $L(4k+1) \approx 12.0013$, $L(4k+2) = 14.001$, and $L(4k+3) \approx 8.0028$.

From time $20k+5$ to time $20k+25$, the twenty adjustments transform any load $l(20k+5)$ under AIMD(1;0.5) to

$$l(20k+25) = pl(20k+5) + r \tag{19}$$

where

$$p = 0.0625 \quad \text{and} \quad r = 5.625 \tag{20}$$

and under MAIMD(1.00014;0.999;$y$) to

$$l(20k+25) = ql(20k+5) + s \tag{21}$$

where

$$0.06114 < q < 0.061141 \quad \text{and} \quad 5.6337 < s < 5.6338. \tag{22}$$

Since fairness $F(20k+5)$ is $l_2(20k+5)/l_1(20k+5)$ according to Lemma 1, and total load $L(20k+5)$ equals $l_1(20k+5) + l_2(20k+5)$, we express the user loads at time $t = 20k+5$ under each of the algorithms as:

$$l_1(t) = \frac{L(t)}{F(t)+1} \quad \text{and} \quad l_2(t) = \frac{L(t)F(t)}{F(t)+1}. \tag{23}$$

The total load under AIMD(1;0.5) is equal to

$$L^A(20k+5) = 12. \tag{24}$$

We prove by induction for any integer $k \geq 7$ that the total load under MAIMD(1.00014;0.999;$y$) is bounded from above as:

$$L^M(20k+5) < 12.0014. \tag{25}$$

We also observe that $F^M(145) > F^A(145)$. Supposing that

$$F^M(20k+5) = G \quad \text{and} \quad F^A(20k+5) = H \tag{26}$$

where

$$0 \leq H \leq G < 1, \tag{27}$$

we derive:

$$
\begin{aligned}
& F^M(20k+25) - F^A(20k+25) \\
= & \quad \{ \text{ Lemma 1 } \} \\
& \frac{l_2^M(20k+25)}{l_1^M(20k+25)} - \frac{l_2^A(20k+25)}{l_1^A(20k+25)} \\
= & \quad \{ \text{ Equations 19 and 21 } \} \\
& \frac{ql_2^M(20k+5)+s}{ql_1^M(20k+5)+s} - \frac{pl_2^A(20k+5)+r}{pl_1^A(20k+5)+r} \\
= & \quad \{ \text{ Equations 23, 26, and 24 } \} \\
& \frac{q\frac{L^M(20k+5)G}{G+1}+s}{q\frac{L^M(20k+5)}{G+1}+s} - \frac{p\frac{12H}{H+1}+r}{p\frac{12}{H+1}+r} \\
> & \quad \{ \text{ Inequalities 25 and 27 } \} \\
& \frac{q\frac{12.0014G}{G+1}+s}{q\frac{12.0014}{G+1}+s} - \frac{p\frac{12H}{H+1}+r}{p\frac{12}{H+1}+r} \\
> & \{ \text{ Equations 20 and Inequalities 22 } \} \\
& \frac{0.09777(1-GH)+8.903(G-H)}{(6.3676+5.6338G)(6.375+5.625H)} \\
> & \{ \text{ Inequalities 27 } \} \\
& 0.
\end{aligned}
$$

Therefore, we establish by induction that $F^M(20k+5) > F^A(20k+5)$ for any integer $k \geq 7$. MAIMD(1.00014;0.999;$y$) provides faster asymptotic fairing than AIMD(1;0.5). ■

***Proof details for Theorem 4:*** An increase at time $t$ improves the nines-representation of fairness under MAIMD($p$;$x$;$z$) by:

$$
\begin{aligned}
& N(t) - N(t-1) \\
= & \quad \{ \text{ Definition 2 and Lemma 1 } \} \\
& \log_{10}\left(1 - \frac{l_{min}(t-1)}{l_{max}(t-1)}\right) - \log_{10}\left(1 - \frac{l_{min}(t)}{l_{max}(t)}\right) \\
= & \\
& \log_{10}\left(\frac{l_{max}(t-1) - l_{min}(t-1)}{l_{max}(t-1)} \cdot \frac{l_{max}(t)}{l_{max}(t) - l_{min}(t)}\right) \\
= & \quad \{ \ l_i(t) = x + pl_i(t-1) \text{ for both } i \in \{max, min\} \ \} \\
& \log_{10} \frac{l_{max}(t)}{pl_{max}(t-1)},
\end{aligned}
$$

i.e.,

$$
N(t) - N(t-1) = \log_{10} \frac{l_{max}(t)}{l_{max}(t-1)} - \log_{10} p. \tag{28}
$$

A decrease at time $t$ does not affect fairness but with derivations similar to the above, we express the lack of change as:

$$
N(t) - N(t-1) = \log_{10} \frac{l_{max}(t)}{l_{max}(t-1)} - \log_{10} z. \tag{29}
$$

To compute the average improvement in the nines-representation of fairness from time 0 to $t$, we represent the overall number of increases and decreases during this time interval as $i(t)$ and $d(t)$ respectively and derive:

$$
\begin{aligned}
& N(t) - N(0) \\
= & \\
& \sum_{\tau=1}^{t} (N(\tau) - N(\tau-1)) \\
= & \quad \{ \text{ Equations 28 and 29 } \} \\
& \sum_{\tau=1}^{t} \log_{10} \frac{l_{max}(\tau)}{l_{max}(\tau-1)} - i(t)\log_{10} p - d(t)\log_{10} z \\
= & \\
& \log_{10} \prod_{\tau=1}^{t} \frac{l_{max}(\tau)}{l_{max}(\tau-1)} - i(t)\log_{10} p - d(t)\log_{10} z \\
= & \\
& \log_{10} \frac{l_{max}(t)}{l_{max}(0)} - i(t)\log_{10} p - d(t)\log_{10} z.
\end{aligned}
$$

Then, we express the fairing factor of MAIMD($p$;$x$;$z$) algorithm as follows:

$$
\begin{aligned}
& G \\
= & \quad \{ \text{ Definition 4 } \} \\
& \lim_{t\to\infty} \frac{N(t) - N(0)}{t} \\
= & \quad \{ \text{ previous derivation } \} \\
& \lim_{t\to\infty} \frac{\log_{10}\frac{l_{max}(t)}{l_{max}(0)} - i(t)\log_{10} p - d(t)\log_{10} z}{t} \\
= & \quad \{ \ l_{max}(t) \text{ is bounded from above and below } \} \\
& -\left(\lim_{t\to\infty}\frac{i(t)}{t}\right)\log_{10} p - \left(\lim_{t\to\infty}\frac{d(t)}{t}\right)\log_{10} z \\
= & \quad \{ \text{ load oscillations converge to the canonical cycle } \} \\
& -\frac{I}{I+D}\log_{10} p - \frac{D}{I+D}\log_{10} z
\end{aligned}
$$

where $I$ and $D$ are respectively the number of increases and decreases in the canonical cycle. Hence, the fairing factor equals:

$$
G = -\frac{I\log_{10} p + D\log_{10} z}{I+D}. \tag{30}
$$

■

***Proof details for Theorem 5:***

$$
\begin{aligned}
& L^M(t) \\
= & \quad \{ \text{ Increase under rule } M \ \} \\
& pL^M(t-1) + nv \\
\leq & \quad \{ \text{ Hypothesis } L^M(t-1) \leq L^A(t-1) < C \ \} \\
& pL^A(t-1) + nv \\
= & \quad \{ \text{ Increase under rule } A \ \} \\
& p(L^A(t) - nv - (p-1)C) + nv \\
= & \\
& L^A(t) - (p-1)(pC - L^A(t) + nv)
\end{aligned}
$$

$$
\begin{aligned}
< \quad & \{\ L^A(t) < C\ \} \\
& L^A(t) - (p-1)((p-1)C + nv) \\
< \quad & \{\ p > 1 \text{ and } v \geq 0\ \} \\
& L^A(t) \\
< \quad & \{\ L^A(t) < C\ \} \\
& C.
\end{aligned}
$$

■

***Proof details for Theorem 6:*** If $q + \frac{nd}{C} > 0$, then $y = 0$, and rule $M$ reaches $C$ from overload in one step. Otherwise, $y = q + \frac{nd}{C}$ and

$$
\begin{aligned}
& L^M(t) \\
= \quad & \{\ \text{Decrease under rule } M\ \} \\
& yL^M(t-1) \\
\leq \quad & \{\ \text{Hypothesis } C < L^M(t-1) \leq L^A(t-1)\ \} \\
& yL^A(t-1) \\
= \quad & \{\ \text{Decrease under rule } A \text{ without truncation to } 0\ \} \\
& \frac{y(L^A(t) - nd)}{q} \\
= \quad & \{\ y = q + \tfrac{nd}{C}\ \} \\
& L^A(t) + \frac{nd}{qC}(L^A(t) - yC) \\
< \quad & \{\ L^A(t) > C \text{ and } nd < 0\ \} \\
& L^A(t) \\
< \quad & \{\ L^A(t) < C\ \} \\
& C.
\end{aligned}
$$

■