Delayed Stability and Performance of Distributed Congestion Control

Yueping Zhang
Texas A&M University
College Station, TX 77843
yueping@cs.tamu.edu

Seong-Ryong Kang
Texas A&M University
College Station, TX 77843
skang@cs.tamu.edu

Dmitri Loguinov*
Texas A&M University
College Station, TX 77843
dmitri@cs.tamu.edu

ABSTRACT

Recent research efforts to design better Internet transport protocols combined with scalable Active Queue Management (AQM) have led to significant advances in congestion control. One of the hottest topics in this area is the design of discrete congestion control algorithms that are asymptotically stable under heterogeneous feedback delay and whose control equations do not explicitly depend on the RTTs of end-flows. In this paper, we show that max-min fair congestion control methods with a stable *symmetric* Jacobian remain stable under arbitrary feedback delay (including heterogeneous directional delays) and that the stability condition of such methods does *not* involve any of the delays. To demonstrate the practicality of the obtained result, we change the original controller in Kelly's work [14] to become robust under random feedback delay and fixed constants of the control equation. We call the resulting framework *Max-min Kelly Control* (MKC) and show that it offers smooth sending rate, exponential convergence to efficiency, and fast convergence to fairness, all of which make it appealing for future high-speed networks.

Categories and Subject Descriptors

C.2.2 [**Communication Networks**]: Network Protocols

General Terms

Algorithms, Performance, Theory

Keywords

Discrete Congestion Control, Heterogenous Delay, Stability

1. INTRODUCTION

Over the last fifteen years, Internet congestion control has evolved from binary-feedback methods of AIMD/TCP [2], [29] to the more exciting developments based on optimization theory [22], [23], game theory [10], [18], and control theory [9], [10], [24], [25]. It is widely recognized that TCP's congestion control in its current shape is inadequate for very high-speed networks and fluctuation-sensitive real-time multimedia. Thus, a significant research effort is currently under way (e.g., [5], [6], [8], [11], [14], [15], [18], [27]) to better understand the desirable properties of congestion control and develop new algorithms that can be deployed in future AQM (Active Queue Management) networks.

One of the most important factors in the design of congestion control is its *asymptotic stability*, which is the capacity of the protocol to avoid oscillations in the steady-state and properly respond to external perturbations caused by the arrival/departure of flows, variation in feedback, and other transient effects. Stability proofs for distributed congestion control become progressively more complicated as feedback delays are taken into account, which is especially true for the case of *heterogeneous* delays where each user i receives its network feedback delayed by a random amount of time D_i. Many existing papers (e.g., [4], [9], [10], [11], [16], [17], [18], [23]) model all users with homogeneous delay $D_i = D$ and do not take into account the fact that end-users in real networks are rarely (if ever) synchronized. Several recent studies [19], [24], [26] successfully deal with heterogeneous delays; however, they model D_i as a deterministic metric and require that end-flows (and sometimes routers) dynamically adapt their equations based on feedback delays, which leads to RTT-unfairness, increased overhead, and other side-effects (such as probabilistic stability).

In this paper, we set our goal to build a discrete congestion control system that maintains both stability and fairness under heterogeneously delayed feedback, allows users to use fixed parameters of the control equation, and admits a low-overhead implementation inside routers. We solve this problem by showing that any max-min fair system with a stable symmetric Jacobian remains asymptotically stable under arbitrary directional delays and apply this result to the original controller proposed by Kelly *et al.* [14]. We call the result of these efforts *Max-min Kelly Control* (MKC) and demonstrate that its stability and fairness do not depend on any parameters of the network (such as delay, path length, or the routing matrix of end-users). We also show that with a proper choice of AQM feedback, MKC converges to efficiency exponentially fast, exhibits stability and fairness under *random* delays, converges to fairness almost as quickly as AIMD, and does not require routers to estimate any parameters of individual flows.

*This work was supported in part by NSF grants CCR-0306246, ANI-0312461.

SIGCOMM'04, Aug. 30–Sept. 3, 2004, Portland, Oregon, USA.

By isolating bottlenecks along each path and responding only to the most-congested resource, the MKC framework allows for very simple stability proofs, which we hope will lead to a better understanding of Kelly's framework in the systems community and eventually result in an actual implementation of these methods in real networks. Our initial thrust in this direction includes `ns2` simulations of MKC, which show that finite time-averaging of flow rates inside each router coupled with a naive implementation of end-user functions leads to undesirable transient oscillations, which become more pronounced when directional delays $D_i^{\rightarrow}$ and $D_i^{\leftarrow}$ to/from each router increase. We overcome this drawback with simple changes at each end-user and confirm that the theoretically predicted monotonic convergence of MKC is achievable in real networks, even when the routers do not know the exact combined rate of end-flows at any time instant n. We also show that our algorithms inside the router incur low overhead (which is less than that in XCP [11] or RED [7]) and require only one addition per arriving packet and two variables per router queue.

The rest of this paper is organized as follows. In Section 2, we review related work. In Section 3, we study delayed stability and steady-state resource allocation of the classic Kelly controls. In Section 4, we present MKC and prove its delay-independent stability. In Section 5, we evaluate convergence properties and packet loss of MKC. In Section 6, we implement MKC in `ns2` and simulate its performance under heterogeneous delays. In Section 7, we conclude our work and suggest directions for future research.

2. BACKGROUND

A large amount of theoretical and experimental work is being conducted to design stable congestion controls for high-speed networks. Such examples include FAST TCP [8], HSTCP [5], Scalable TCP [15], BIC-TCP [28], and XCP [11], all of which aim to achieve quick convergence to efficiency, stable rate trajectories, fair bandwidth sharing, and low packet loss. An entirely different direction in congestion control is to model the network from an optimization or game-theoretic point of view [10], [16], [17], [18], [23]. The original work by Kelly *et al.* [13], [14] offers an economic interpretation of the resource-user model, in which the entire system achieves its optimal performance by maximizing the individual utility of each end-user. To implement this model in a decentralized network, Kelly *et al.* describe two algorithms (*primal* and *dual*) and prove their global stability in the absence of feedback delay. However, if feedback delay is present in the control loop, stability analysis of Kelly controls is non-trivial and currently forms an active research area [4], [9], [19], [24], [26], [27].

Recall that in Kelly's framework [14], [24], each user $i \in [1, N]$ is given a unique route r_i that consists of one or more network resources (routers). Feedback delays in the network are heterogeneous and directional. The forward and backward delays between user i and resource j are denoted by $D_{ij}^{\rightarrow}$ and $D_{ij}^{\leftarrow}$, respectively. Thus, the round-trip delay of user i is the summation of its forward and backward delays with respect to any router $j \in r_i$: $D_i = D_{ij}^{\rightarrow} + D_{ij}^{\leftarrow}$. Under this framework, Johari *et al.* discretize Kelly's primal algorithm as follows [9]:

$$x_i(n) = x_i(n-1)+\kappa_i\Big(\omega_i-x_i(n-D_i)\sum_{j\in r_i}\mu_j(n-D_{ij}^{\leftarrow})\Big), \quad (1)$$

where κ_i is a strictly positive gain parameter, ω_i can be interpreted as the willingness of user i to pay the price for using the network, and $\mu_j(n)$ is the congestion indication function of resource j:

$$\mu_j(n) = p_j\Big(\sum_{u\in s_j} x_u(n-D_{uj}^{\rightarrow})\Big), \quad (2)$$

where s_j denotes the set of users sharing resource j and $p_j(\cdot)$ is the price charged by resource j. Note that we use a notation in which $D_i = 1$ means immediate (i.e., most recent) feedback and $D_i \geq 2$ implies delayed feedback.

Next, recall that for a homogeneous delay D, system (1)-(2) is locally stable if [9]:

$$\kappa_i \sum_{j\in r_i}\Big((p_j + p_j'\sum_{u\in s_j} x_u)\Big|_{x_u^*}\Big) < 2\sin\left(\frac{\pi}{2(2D-1)}\right), \quad (3)$$

where x_u^* is the stationary point of user u and $p_j(\cdot)$ is assumed to be differentiable at x_u^*.

For heterogeneous delays, a combination of conjectures made by Johari *et al.* [9], derivations in Massoulié [24], and the proofs of Vinnicombe [26] suggest that delay D in (3) can be simply replaced with individual delays D_i to form a system of N stability equations; however, the proof exists only for the *continuous* version of (1) and leads to the following necessary stability equation [26]:

$$\kappa_i \sum_{j\in r_i}\Big((p_j + p_j'\sum_{u\in s_j} x_u)\Big|_{x_u^*}\Big) < \frac{\pi}{2D_i}. \quad (4)$$

We should also note that Ying *et al.* [30] recently established delay-independent stability conditions for a family of utility functions and a generalized controller (1). Their work is similar in spirit to ours; however, the analysis and proposed methods are different.

3. CLASSIC KELLY CONTROL

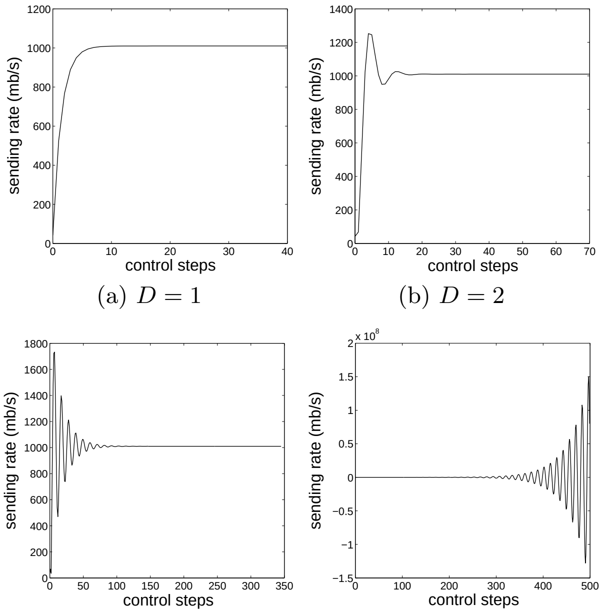
In this section, we first discuss intuitive examples that explain the cryptic formulas in the previous section and demonstrate in simulation how delays affect stability of Kelly controls (1)-(2). We then show that Kelly's proportional fairness [14], or any mechanism that relies on the *sum* of feedback functions from individual routers, always exhibits linear convergence to efficiency. Note that due to limited space, we omit certain proofs and refer the reader to the technical report [31] for more information.

3.1 Delayed Stability Example

The following example illustrates stability problems of (1) when feedback delays are large. We assume a single-source, single-link configuration and utilize a standard congestion indication function, which computes the estimated packet loss using instantaneous arrival rates:

$$p(n) = \frac{x(n) - C}{x(n)}, \quad (5)$$

where C is the link capacity and $x(n)$ is the flow rate at discrete step n. We remark that under AQM feedback assumed throughout the paper, we allow *negative* packet loss in (5), which signals the flows to increase their sending rates when $x(n) < C$. In section 5.1, we show that the negative component of packet-loss (5) improves convergence to efficiency from linear to exponential.



(a) $D = 1$ (b) $D = 2$

(c) $D = 3$ (d) $D = 4$

Figure 1: Stability of Kelly control under different feedback delays ($\kappa = 1/2$, $\omega = 10$ mb/s, and $C = 1,000$ mb/s).

Applying (5) to Kelly control (1) yields a linear end-flow equation:

$$x(n) = x(n-1) + \kappa\omega - \kappa\big(x(n-D) - C\big). \tag{6}$$

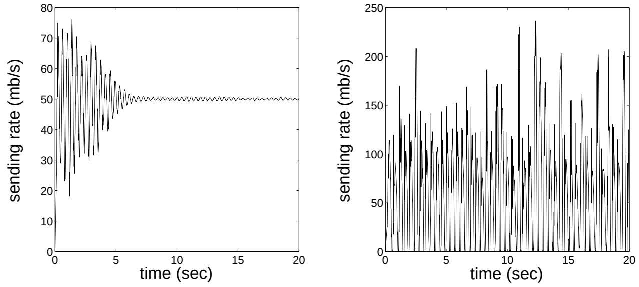
Next, assume a particular set of parameters: $\kappa = 1/2$, $\omega = 10$ mb/s, and $C = 1,000$ mb/s. Solving the condition in (3), we have that the system is stable if and only if delay D is less than four time units. As illustrated in Figure 1(a), delay $D = 1$ keeps the system stable and monotonically convergent to its stationary point. Under larger delays $D = 2$ and $D = 3$ in Figures 1(b) and (c), the flow exhibits progressively increasing oscillations before entering the steady state. Eventually, as soon as D becomes equal to four time units, the system diverges as shown in Figure 1(d).

Using the same parameter κ and reducing ω to 20 kb/s, we examine (6) via `ns2` simulations, in which a single flow passes through a link of capacity 50 mb/s. We run the flow in two network configurations with the round-trip delay equal to 90 ms and 120 ms, respectively. As seen in Figure 2, the first flow reaches its steady state after decaying oscillations, while the second flow exhibits no convergence and periodically overshoots capacity C by 200%.

Since Kelly controls are unstable unless condition (3) is satisfied [9], a natural strategy to maintain stability is for each end-user i to adaptively adjusts its gain parameter $\kappa_i \sim 1/D_i$ such that (3) is not violated. However, this method depends on reliable estimation of round-trip delays D_i and leads to unfairness between the flows with different RTTs.

3.2 Stationary Rate Allocation

In this section, we examine how packet-loss function (5) affects the resource allocation of Kelly's proportional fairness (1). Consider a network of M resources and N homogeneous users (i.e., with the same parameters κ and ω). Further assume that resource j has capacity C_j, user i utilizes route r_i of length M_i (i.e., $M_i = |r_i|$), and packet-loss $\eta_i(n)$ fed back to user i is the *aggregate* feedback from all resources in path r_i. We further assume that there is no redundancy in the network (i.e., each user sends its packets through at least one resource and all resources are utilized by at least one user). Thus, we can define utilization matrix $A_{N\times M}$ such that $A_{ij} = 1$ if user i passes through resource j (i.e., $j \in r_i$) and $A_{ij} = 0$ otherwise. Further denote the j-th column of A by vector $\mathbf{V}_j$. Clearly, $\mathbf{V}_j$ identifies the set s_j of flows passing through router j.



(a) $D = 90$ ms (b) $D = 120$ ms

Figure 2: Simulation results of the classic Kelly control under different delays ($\kappa = 1/2$, $\omega = 20$ kb/s, $C = 50$ mb/s).

Let $\mathbf{x}_j(n) = \langle x_1(n - D_{\overrightarrow{1j}}), x_2(n - D_{\overrightarrow{2j}}), \cdots, x_N(n - D_{\overrightarrow{Nj}})\rangle$ be the vector of sending rates of individual users observed at router j at time instant n. In the spirit of (5), the packet loss of resource j at instant n can be expressed as:

$$p_j(n) = \frac{\mathbf{x}_j(n)\cdot\mathbf{V}_j - C_j}{\mathbf{x}_j(n)\cdot\mathbf{V}_j}, \tag{7}$$

where the dot operator represents vector multiplication. Accordingly, the end-to-end feedback $\eta_i(n)$ of user i is:

$$\eta_i(n) = \sum_{j\in r_i} p_j(n - D_{\overleftarrow{ij}}), \tag{8}$$

and the control equation assumes the following shape:

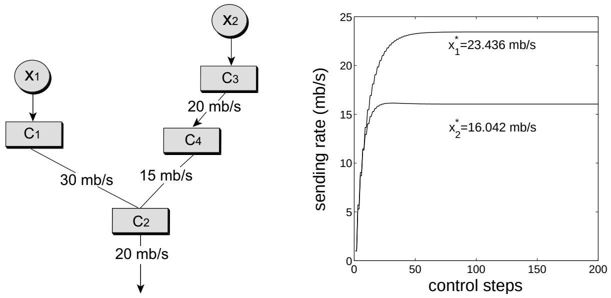
$$x_i(n) = x_i(n-1) + \kappa_i\Big(\omega_i - x_i(n - D_i)\eta_i(n)\Big). \tag{9}$$

Then, we have the following result.

LEMMA 1. *Let $\mathbf{x}^* = \langle x_1^*, x_2^*, \cdots, x_N^*\rangle$ be the stationary rate allocation of Kelly control (9) with packet-loss function (7)-(8). Then $\mathbf{x}^*$ satisfies:*

$$\sum_{i=1}^{N} M_i x_i^* = \sum_{j=1}^{M} C_j + N\omega. \tag{10}$$

Lemma 1 provides a connection between the stationary resource allocation and the path length of each flow. Note that according to (10), the stationary rates x_i^* are constrained by the capacity of *all* resources instead of by that of individual bottlenecks. In fact, this observation shows an important difference between the real network paths, which are limited by the *slowest* resource, and the model of proportional fairness, which takes into account the capacity of *all*



(a) simulation setup (b) simulation results

Figure 3: Rate allocation of proportional fairness ($\kappa = 0.1$ and $\omega = 5$ mb/s).

resources in the network. This difference leads to severe under/over-utilization of individual routers as illustrated in the following simulation.

Figure 3(a) shows the topology of the simulation, in which there are two flows, x_1 and x_2, and four resources, C_1-C_4. Notice that resource $C_2 = 20$ mb/s is the bottleneck of x_1 and $C_4 = 15$ mb/s is the bottleneck of x_2. The path lengths of the two flows are, respectively, $M_1 = 2$ and $M_2 = 3$. Setting $\omega = 5$ mb/s and $\kappa = 0.1$, let $\langle x_1^*, x_2^* \rangle$ be the stationary rate allocation of the system, which according to (10) must satisfy:

$$2x_1^* + 3x_2^* = \sum_{j=1}^{4} C_i + 2 \times 5 = 95. \tag{11}$$

Simulation results for this setup are depicted in Figure 3(b). As seen in the figure, the steady-state rate assignment is $\langle 23.436, 16.042 \rangle$ mb/s, which indeed satisfies prediction (11); however, notice that the combined stationary rate of both flows is 39.5 mb/s, which exceeds C_2 by 97%. As a result, the users overshoot network capacity in the steady state and persistently suffer from significant packet loss.

This problem is easy to understand. Observe that uncongested routers C_1 and C_3 encourage end-flows to increase their rates through negative feedback, while congested resources C_2 and C_4 signal the opposite and encourage the flows to reduce their rates. Combining this conflicting feedback into summation (8), each user settles in some middle ground that keeps neither their slowest, nor their fastest resources in r_i fully "satisfied." Even for a network with a single flow, (11) shows that the stationary rate of the flow is simply the average capacity of all resources on its path: $x^* = (\sum_{j=1}^{M} C_j + \omega)/M$. For the example in Figure 3, x_1 would converge to 27.5 mb/s, which is well in excess of its bottleneck capacity C_2.

In general, for proportional fairness (8) and similar methods that rely on the combined pricing function of all resources to remain viable, no price should be charged at routers that are not suffering any packet loss. Under these circumstances, notice in (1) that the flows increase their rates by $\kappa_i \omega_i$ at each discrete time-step before they reach full link utilization at the slowest router. This results in linear AIMD-like probing for new bandwidth, which is generally considered "too slow" for high-speed networks.

In the next section, we overcome both drawbacks of controller (1) (i.e., instability under delay and undesirable link utilization) by abandoning proportional fairness and focusing on its max-min counterpart.

4. STABLE CONGESTION CONTROL

In this section, we propose a new version of discrete Kelly controls, which allows negative packet-loss feedback and maintains stability under heterogeneous delays.

4.1 Max-min Kelly Control

We start our discussion with the following observations. First, we notice that in the classic Kelly control (1), the end-user decides its current rate $x_i(n)$ based on the most recent rate $x_i(n-1)$ and delayed feedback $\mu_j(n - D_{\overleftarrow{ij}})$. Since the latter carries information about $x_i(n - D_i)$, which was in effect *RTT time units earlier*, the controller in (1) has no reason to involve $x_i(n-1)$ in its control loop. Thus, the sender quickly becomes unstable as the discrepancy between $x_i(n-1)$ and $x_i(n - D_i)$ increases. One natural remedy to this problem is to retard the reference rate to become $x_i(n - D_i)$ instead of $x_i(n-1)$ and allow the feedback to accurately reflect network conditions with respect to the first term of (1).

Second, to avoid unfairness[1] between flows, one must fix the control parameters of all end-users and establish a uniform set of equations that govern the system. Thus, we create a new notation in which $\kappa_i \omega_i = \alpha$, $\kappa_i = \beta$ and discretize the Kelly control as following:

$$x_i(n) = x_i(n - D_i) + \alpha - \beta \eta_i(n) x_i(n - D_i), \tag{12}$$

where $\eta_i(n)$ is the congestion indication function of user i.

Next, to overcome the problems of proportional fairness demonstrated in the previous section and utilize negative network feedback, we combine (12) with max-min fairness (this idea is not new [11]), under which the routers only feed back the packet loss of the *most-congested* resource instead of the combined packet loss (8):

$$\eta_i(n) = \max_{j \in r_i} p_j(n - D_{\overleftarrow{ij}}), \tag{13}$$

where $p_j(\cdot)$ is the congestion indication function of individual routers that depends only on the aggregate arrival rates of end-users:

$$p_j(n) = p_j\Big(\sum_{u \in s_j} x_u(n - D_{\overrightarrow{uj}})\Big). \tag{14}$$

We call the resulting controller (12)-(14) *Max-min Kelly Control* (MKC) and emphasize that the flows congested by the same bottleneck receive the *same* feedback and behave independently of the flows congested by the other links. Therefore, in the rest of this paper, we study the single-bottleneck case since each MKC flow is always congested by only *one* router. Implementation details of how routers should feed back function (13) and how end-flows track the changes in the most-congested resource are presented in the simulation section.

4.2 Delay-Independent Stability

Before restricting our analysis to MKC, we examine a wide class of delayed control systems, whose stability directly follows from that of the corresponding undelayed systems. We

[1] While "fairness" is surely a broad term, we assume its max-min version in this paper.

subsequently show that MKC belongs to this category and obtain a very simple proof of its stability. First consider the following theorem.

Theorem 1. *Assume an undelayed linear system $\mathcal{L}$ with N flows:*

$$x_i(n) = \sum_{j=1}^{N} a_{ij} x_j(n-1). \tag{15}$$

If coefficient matrix $A = (a_{ij})$ is real-valued and symmetric, *then system $\mathcal{L}_D$ with arbitrary directional delays:*

$$x_i(n) = \sum_{j=1}^{N} a_{ij} x_j(n - D_j^{\rightarrow} - D_i^{\leftarrow}), \tag{16}$$

is asymptotically stable if and only if $\mathcal{L}$ is stable.

Proof. We first show the sufficient condition. Assume that $\mathcal{L}$ is asymptotically stable. Applying the z-transform to system (16), we obtain:

$$\mathbf{H}(z) = Z_1 A Z_2 \mathbf{H}(z), \tag{17}$$

where $Z_1 = \text{diag}(z^{-D_i^{\leftarrow}})$ and $Z_2 = \text{diag}(z^{-D_i^{\rightarrow}})$ are the diagonal matrices of directional delays, and $\mathbf{H}(z)$ is the vector of z-transforms of each flow rate x_i:

$$\mathbf{H}(z) = \big\langle H_1(z), H_2(z), \cdots, H_N(z) \big\rangle^T. \tag{18}$$

Notice that linear system (16) is stable if and only if all poles of its z-transform $\mathbf{H}(z)$ are within the unit circle in the z-plane [12]. To examine this condition, re-organize the terms in (17):

$$(Z_1 A Z_2 - I)\,\mathbf{H}(z) = 0. \tag{19}$$

Next notice that the poles of $\mathbf{H}(z)$ are simply the roots of:

$$\det(Z_1 A Z_2 - I) = 0. \tag{20}$$

Thus, ensuring that all roots of (20) are inside the unit circle will be both sufficient and necessary for $\mathcal{L}_D$ to be stable. Rewriting (20):

$$\begin{aligned} \det(Z_1 A Z_2 - I) &= \det(Z_1[A - Z_1^{-1} I Z_2^{-1}] Z_2) \qquad (21)\\ &= \det(Z_1)\det(A - Z_1^{-1} Z_2^{-1})\det(Z_2). \end{aligned}$$

Since $\det(Z_1)$ and $\det(Z_2)$ are strictly non-zero for non-trivial z, (20) reduces to:

$$\det(A - Z_1^{-1} Z_2^{-1}) = \det\big(A + B(z)\big) = 0, \tag{22}$$

where $B(z) = -Z_1^{-1} \cdot Z_2^{-1} = -\text{diag}(z^{D_i})$ is the diagonal matrix of round-trip delays. Thus, it remains to examine whether the roots of (22) are inside the unit circle.

To bound the roots of (22), we first need the following theorem from [20].

Theorem 2 (Li-Mathias [20]). *Given N-dimensional square matrices Q_1 and Q_2, whose singular values are $\alpha_1 \geq \alpha_2 \geq \cdots \geq \alpha_N$ and $\beta_1 \geq \beta_2 \geq \cdots \geq \beta_N$, respectively, the following holds:*

$$|\det(Q_1 + Q_2)| \geq \begin{cases} 0 & \text{if } [\alpha_N, \alpha_1] \cap [\beta_N, \beta_1] \neq \emptyset \\ \Big|\prod_{j=1}^{N} (\alpha_j - \beta_{N-j+1})\Big| & \text{otherwise} \end{cases}. \tag{23}$$

We next apply the lower bounds given in the above theorem to (22). Recall that singular values of a square matrix X are the non-negative square roots of the eigenvalues of the product of X and its adjoint (or equivalently, conjugate transpose) matrix X^* [1]. In (22) both matrices A and $B(z)$ are symmetric and real-valued, which means that their singular values are the absolute values of their eigenvalues. Let $\{\lambda_i\}$ be the eigenvalues of A. Then the singular values of A are $\{\alpha_i | \alpha_i = |\lambda_i|\}$. Similarly, we get that the singular values of a diagonal matrix $B(z)$ are $\{\beta_i | \beta_i = |z^{D_i}|\}$. Without loss of generality, we assume that $\{\alpha_i\}$ and $\{\beta_i\}$ are ordered by their magnitude, i.e., $\alpha_1 \geq \alpha_2 \geq \cdots \geq \alpha_N$ and $\beta_1 \geq \beta_2 \geq \cdots \geq \beta_N$.

Let z_0 be a root of (22). Then there are two possibilities:

1) The intervals defined by $\{\alpha_i\}$ and $\{\beta_i\}$ intersect, i.e., $[\alpha_N, \alpha_1] \cap [\beta_N, \beta_1] \neq \emptyset$. This means that there must exist at least one singular value β_j such that $\beta_j \in [\alpha_N, \alpha_1]$. According to the assumption that undelayed system $\mathcal{L}$ in (15) is stable, each eigenvalue λ_i of matrix A must satisfy $|\lambda_i| = \alpha_i < 1$, which leads to $0 \leq \beta_j < 1$. This translates into $|z_0^{D_j}| < 1$, for some j, and directly leads to $|z_0| < 1$ since all delays D_j are discrete and no less than 1.

2) The two segments do not overlap, i.e., $[\alpha_N, \alpha_1] \cap [\beta_N, \beta_1] = \emptyset$. Then, combining (22) and (23), we have:

$$|\det\big(A + B(z_0)\big)| = 0 \geq \Big|\prod_{j=1}^{N} (\alpha_j - \beta_{N-j+1})\Big|, \tag{24}$$

which implies that the product in (24) must equal zero:

$$\Big|\prod_{j=1}^{N} (\alpha_j - \beta_{N-j+1})\Big| = 0. \tag{25}$$

This means that there exists an index k ($1 \leq k \leq N$) such that $\alpha_k - \beta_{N-k+1} = 0$, which contradicts the assumption that intervals $[\alpha_N, \alpha_1]$ and $[\beta_N, \beta_1]$ are disjoint.

Repeating steps 1) and 2) for all roots $\{z_i\}$ of (22), we obtain that they all must lie within the unit circle, which leads to the asymptotic stability of $\mathcal{L}_D$ in (16).

Since $\mathcal{L}$ is a special case of $\mathcal{L}_{\mathcal{D}}$ (i.e., all delays are 1 time unit), stability of the latter implies that of the former and leads to the necessary condition of the theorem. □

Theorem 1 opens an avenue for inferring stability of delayed linear systems based on the coefficient matrices of the corresponding undelayed systems. Moreover, it is easy to see that Theorem 1 applies to nonlinear systems as stated in the following corollary.

Corollary 1. *Assume an undelayed N-dimensional nonlinear system $\mathcal{N}$:*

$$x_i(n) = f_i\big(x_1(n-1), x_2(n-1), \cdots, x_N(n-1)\big), \tag{26}$$

where $\{f_i | f_i : \mathbb{R}^N \rightarrow \mathbb{R}\}$ is the set of nonlinear functions defining the system. If the Jacobian matrix J of this system is symmetric and real-valued, system $\mathcal{N}_D$ with arbitrary delay:

$$\begin{aligned} x_i(n) = f_i\big(&x_1(n - D_1^{\rightarrow} - D_i^{\leftarrow}), x_2(n - D_2^{\rightarrow} - D_i^{\leftarrow}),\\ &\cdots, x_N(n - D_N^{\rightarrow} - D_i^{\leftarrow})\big) \qquad (27) \end{aligned}$$

is locally asymptotically stable in the stationary point $\mathbf{x}^*$ if and only if $\mathcal{N}$ is stable in $\mathbf{x}^*$.

Based on the above principles, we next prove local stability of MKC under heterogeneous feedback delays.

4.3 Stability of MKC

We first consider an MKC system with a generic feedback function $\eta_i(n)$ in the form of (13), which we assume is differentiable in the stationary point and has the same first-order partial derivative for all end-users. Our goal is to derive sufficient and necessary conditions for the stability of (12)-(13) under arbitrarily delayed feedback.

We approach this problem by partitioning all users into non-overlapping sets based on their corresponding bottleneck routers. We assume that each set of users $\mathcal{S}$ is fairly stable and that the bottlenecks do not change for the duration of this analysis. Suppose that $\mathcal{S}$ contains users $\{x_1, \dots, x_N\}$ and that the corresponding delays to/from their bottleneck router are given by $D_i^{\rightarrow}$ and $D_i^{\leftarrow}$. Then, we can simplify (12)-(13) by dropping index j of the bottleneck resource and expanding $\eta_i(n)$ in (12):

$$x_i(n) = x_i(n - D_i) + \alpha - \beta p(n - D_i^{\leftarrow}) x_i(n - D_i), \qquad (28)$$

where

$$p(n) = p\Big(\sum_{u=1}^{N} x_u(n - D_u^{\rightarrow})\Big) \qquad (29)$$

is the packet-loss function of the bottleneck router for set $\mathcal{S}$. Notice that $x_i(n - D_i)$ in (28) can be represented as $x_i(n - D_i^{\rightarrow} - D_i^{\leftarrow})$ and that controller (28)-(29) has the same shape as that in (27).

To invoke Theorem 1, our first step is to show stability of the following undelayed version of (28)-(29):

$$\left\{ \begin{array}{lcl} x_i(n) & = & \big(1 - \beta p(n-1)\big) x_i(n-1) + \alpha \\ p(n) & = & p\Big(\sum_{u=1}^{N} x_u(n)\Big) \end{array} \right. . \qquad (30)$$

THEOREM 3. *Undelayed N-dimensional system (30) with feedback $p(n)$ that is common to all users has a symmetric Jacobian and is locally asymptotically stable if and only if:*

$$0 < \beta p^* < 2, \qquad (31)$$

$$0 < \beta p^* + \beta N x^* \left.\frac{\partial p}{\partial x_i}\right|_{\mathbf{x}^*} < 2, \qquad (32)$$

where x^* is the fixed point of each individual user, vector $\mathbf{x}^* = \langle x^*, x^*, \cdots, x^* \rangle$ is the fixed point of the entire system, and p^* is the steady-state packet loss.

PROOF. We first derive the stationary point x^* of each individual user. Since all end-users receive the same feedback and activate the same response to it, all flows share the bottleneck resource fairly in the steady state, i.e., $x_i(n) = x^*$ for all i. Using simple manipulations in (30), we get the stationary individual rate x^* as following:

$$x^* = \frac{\alpha}{\beta p^*}. \qquad (33)$$

Linearizing the system in $\mathbf{x}^*$:

$$\left.\frac{\partial f_i}{\partial x_i}\right|_{\mathbf{x}^*} = \left.\left(1 - \beta p - \beta x_i \frac{\partial p}{\partial x_i}\right)\right|_{\mathbf{x}^*}, \qquad (34)$$

$$\left.\frac{\partial f_i}{\partial x_k}\right|_{\mathbf{x}^*} = \left.\left(-\beta x_i \frac{\partial p}{\partial x_k}\right)\right|_{\mathbf{x}^*}, \quad k \neq i, \qquad (35)$$

where $f_i(\mathbf{x}) = (1 - \beta p(\mathbf{x}))x_i + \alpha$. Since packet loss depends on the *aggregate* rate of all users, $p(n)$ has the same first partial derivative evaluated in the fixed point for all users, which implies that for any users i and k, we have:

$$\left.\frac{\partial p}{\partial x_i}\right|_{\mathbf{x}^*} = \left.\frac{\partial p}{\partial x_k}\right|_{\mathbf{x}^*}. \qquad (36)$$

This observation leads to a simple Jacobian matrix for MKC:

$$J = \begin{pmatrix} a & b & \cdots & b \\ b & a & \cdots & b \\ \vdots & \vdots & \ddots & \vdots \\ b & b & \cdots & a \end{pmatrix}, \qquad (37)$$

where:

$$a = 1 - \beta p^* - \beta x^* \left.\frac{\partial p}{\partial x_i}\right|_{\mathbf{x}^*}, \quad b = -\beta x^* \left.\frac{\partial p}{\partial x_i}\right|_{\mathbf{x}^*}. \qquad (38)$$

Clearly Jacobian matrix J is circulant[2] and thus its k-th eigenvalue λ_k is given by [1]:

$$\lambda_k = a + b(\zeta_k + \zeta_k^2 + \zeta_k^3 + \cdots + \zeta_k^{N-1}), \qquad (39)$$

where $\zeta_k = e^{i2\pi k/N}$ $(k = 0, 1, \dots, N-1)$ is one of the N-th roots of unity. We only consider the case of $N \geq 2$, otherwise the only eigenvalue is simply a. Then, it is not difficult to get the following result:

$$\lambda_k = \begin{cases} a + (N-1)b & \zeta_k = 1 \\ a + b\dfrac{\zeta_k - \zeta_k^N}{1 - \zeta_k} = a - b & \zeta_k \neq 1 \end{cases}, \qquad (40)$$

where the last transition holds since $\zeta_k^N = 1$ for all k.

Next, recall that nonlinear system (30) is locally stable if and only if all eigenvalues of its Jacobian matrix J are within the unit circle [12]. Therefore, we get the following necessary and sufficient local stability conditions:

$$\left\{ \begin{array}{c} |a - b| < 1 \\ |a + (N-1)b| < 1 \end{array} \right. . \qquad (41)$$

To ensure that each λ_i lies in the unit circle, we examine the two conditions in (41) separately. First, notice that $|a - b| = |1 - \beta p^*|$, which immediately leads to the following:

$$0 < \beta p^* < 2. \qquad (42)$$

Applying the same substitution to the second inequality in (41), we obtain:

$$0 < \beta p^* + \beta N x^* \left.\frac{\partial p}{\partial x_i}\right|_{\mathbf{x}^*} < 2. \qquad (43)$$

Thus, system (30) is locally stable if and only if both (42) and (43) are satisfied. □

According to the proof of Theorem 3, Jacobian J of the undelayed system (30) is real-valued and symmetric. Combining this property with Corollary 1, we obtain the following result.

[2] A matrix is called *circulant* if it is square and each of its rows can be obtained by shifting (with wrap-around) the previous row one column right [1].

Corollary 2. *Heterogeneously delayed MKC (28)–(29) is locally asymptotically stable if and only if (31)-(32) are satisfied.*

Corollary 2 is a generic result that is applicable to MKC (12) with a wide class of congestion-indicator functions $\eta_i(n)$. Further note that for a given bottleneck resource with pricing function $p(n)$ and its set of users $\mathcal{S}$, conditions (31)-(32) are easy to verify and do *not* depend on feedback delays, the number of hops in each path, or the routing matrix of all users. This is in contrast to many current studies [9], [24], [26], [27], whose results are dependent on individual feedback delays D_i and the topology of the network.

4.4 Exponential MKC

To understand the practical implications of the derivations above, consider a particular packet-loss function $p(n)$ in (29):

$$p(n) = \frac{\sum_{u=1}^{N} x_u(n - D_u^{\rightarrow}) - C}{\sum_{u=1}^{N} x_u(n - D_u^{\rightarrow})}, \tag{44}$$

where we again assume a set $\mathcal{S}$ of N users congested by a common router of capacity C. This is a rather standard packet-loss function with the exception that we allow it to become negative when the link is under-utilized. As we show in the next section, (44) achieves exponential convergence to efficiency, which explains why we call the combination of (28),(44) *Exponential MKC* (EMKC).

Theorem 4. *Heterogeneously delayed EMKC (28),(44) is locally asymptotically stable if and only if* $0 < \beta < 2$.

Proof. We first derive the fixed point of EMKC. Notice that in the proof of Theorem 3, we established the existence of a unique stationary point $x_i^* = x^*$ for each flow. Then assuming EMKC packet-loss function (44), we have:

$$p^* = \frac{Nx^* - C}{Nx^*}. \tag{45}$$

Combining (45) and (33), we get the stationary point x^* of each end-user:

$$x^* = \frac{C}{N} + \frac{\alpha}{\beta}. \tag{46}$$

Denoting by $X(n) = \sum_{i=1}^{N} x_i(n)$ the combined rate of all N end-users at time n, the corresponding combined stationary rate X^* is:

$$X^* = Nx^* = C + N\frac{\alpha}{\beta}. \tag{47}$$

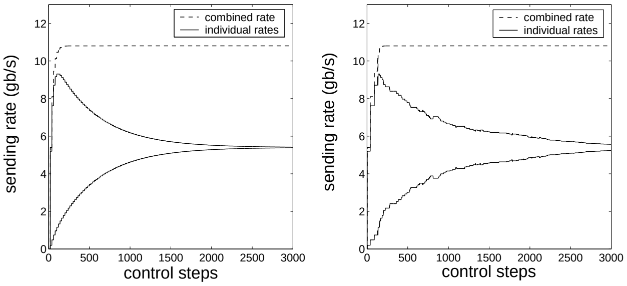
Next, recall from Theorem 3 that stability conditions (31)-(32) must hold for the delayed system to be stable. Consequently, we substitute pricing function (44) into (32) and obtain with the help of (47):

$$\beta p^* + \beta N x^* \left.\frac{\partial p(n)}{\partial x(n)}\right|_{\mathbf{x}^*} = \beta p^* + \frac{\beta N x^* C}{N^2 x^{*2}} = \beta. \tag{48}$$

Thus, condition (32) becomes:

$$0 < \beta < 2. \tag{49}$$

Notice that in the steady state, packet loss probability p^* is no larger than one. Thus, condition (49) is more conservative than (31), which allows us to conclude that when $0 < \beta < 2$, all eigenvalues of Jacobian matrix J are inside the unit circle. Applying Corollary 2, heterogeneously delayed EMKC in (28),(44) is also locally asymptotically stable if and only if $0 < \beta < 2$. □



(a) $D = 20$ (b) random D_i

Figure 4: Two EMKC flows ($\alpha = 200$ mb/s and $\beta = 0.5$) share a single link of capacity 10 gb/s: (a) constant (homogeneous) delay $D = 20$ time units; (b) heterogeneous delays randomly distributed between 1 and 100 time units.

To better understand the implication of this result, consider an illustration in Figure 4, in which two EMKC flows ($\alpha = 200$ mb/s and $\beta = 0.5$) share a bottleneck link of capacity 10 gb/s. Recall that for the same setup ($\beta = 0.5$), Kelly controls are unstable for any delay $D \geq 4$ time units (see Figure 1). In both cases shown in Figure 4, EMKC flows approach full link utilization without oscillations and eventually share the resource fairly. These simulation results support our earlier conclusion that MKC is a stable and fair controller under *random* delays, which is a requirement for any practical method in the current Internet.

5. PERFORMANCE OF EMKC

5.1 Convergence to Efficiency

In this section, we show that EMKC converges to efficiency exponentially fast.

Lemma 2. *For $0 < \beta < 2$ and constant delay D, the combined rate $X(n)$ of EMKC is globally asymptotically stable and converges to $X^* = C + N\alpha/\beta$ at an exponential rate.*

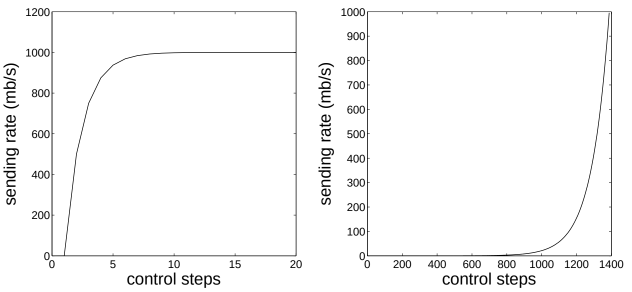
Proof. Since delays do not affect stability of EMKC, assume a constant feedback delay D and re-write (28):

$$x_i(n) = (1 - \beta p(n - D))x_i(n - D) + \alpha, \tag{50}$$

where $p(n)$ is the undelayed version of (44). Taking the summation of (50) for all N flows, we get that EMKC's combined rate $X(n) = \sum_{i=1}^{N} x_i(n)$ forms a linear system:

$$\begin{aligned} X(n) &= \left(1 - \beta\frac{X(n-D) - C}{X(n-D)}\right) X(n-D) + N\alpha \\ &= (1-\beta)X(n-D) + \beta C + N\alpha. \end{aligned} \tag{51}$$

It is clear that the above linear system is stable if and only if $0 < \beta < 2$. Since convergence of linear systems implies global asymptotic stability, we conclude that $X(n)$



(a) EMKC (b) Scalable TCP

Figure 5: The speed of saturating a bottleneck link of capacity 1 gb/s: (a) EMKC ($\beta = 0.5$ and $\alpha = 10$ kb/s); (b) Scalable TCP.

is globally stable regardless of individual flow trajectories $x_i(n)$.

We next show the convergence speed of $X(n)$. Recursively expanding the last equation, we have:

$$X(n) = (1-\beta)^{\frac{n}{D}}(X_0 - X^*) + X^*, \qquad (52)$$

where X_0 is the initial combined rate of all flows and $X^* = C + N\alpha/\beta$ is the combined stationary rate. Notice that for $0 < \beta < 2$, the first term in (52) approaches zero exponentially fast and $X(n)$ indeed converges to X^*. $\square$

This result is illustrated in Figure 5(a) for $\beta = 0.5$ and $\alpha = 10$ kb/s, where EMKC saturates a 1 gb/s link in only 16 steps. In Figure 5(b), we show the convergence rate of Scalable TCP [15], which is a recent method proposed for high-speed networks. Although Scalable TCP also claims bandwidth exponentially fast, its increase rate 1.01^n is much slower than that of EMKC. This is illustrated in the figure where it takes Scalable TCP approximately 1,200 steps to reach full link capacity from the same initial rate.

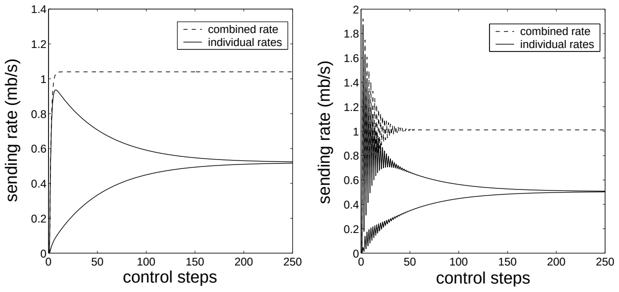
Additionally notice how the value of β affects the behavior of EMKC. For $0 < \beta \leq 1$, the system monotonically converges to the stationary point; however, for $1 < \beta < 2$, the system experiences decaying oscillations before reaching the stationary point, which are caused by the oscillating term $(1-\beta)^{n/D}$ in (52). This phenomenon is illustrated in Figure 6 for two values of β. Thus, in practical settings, β should be chosen in the interval $(0, 1]$, where values closer to 1 result in faster convergence to efficiency.

5.2 Convergence to Fairness

We next investigate the convergence rate of EMKC to fairness. To better understand how many steps EMKC requires to reach a certain level of max-min fairness, we utilize a simple metric that we call *ε-fairness.* For a given small positive constant ε, a rate allocation $\langle x_1, x_2, \cdots, x_N\rangle$ is ε-fair, if:

$$f = \frac{\min_{i=1}^N x_i}{\max_{j=1}^N x_j} \geq 1 - \varepsilon. \qquad (53)$$

Generally speaking, ε-fairness assesses max-min fairness by measuring the worst-case ratio between the rates of any pair of flows. Given the definition in (53), we have the following result.



(a) $\beta = 0.5$ (b) $\beta = 1.9$

Figure 6: Behavior of EMKC for different β ($C = 1$ mb/s and $\alpha = 10$ kb/s).

THEOREM 5. *Consider an EMKC network with N users and a bottleneck link of capacity C. Assuming that the system is started in the maximally unfair state, ε-fairness is reached in θ_M steps, where:*

$$\theta_M = \frac{\big(C + N\frac{\alpha}{\beta}\big)\big(\log N - \log\varepsilon\big)}{N\alpha} + \Theta\Big(\frac{N\alpha}{C}\Big). \qquad (54)$$

A comparison of model (54) to simulation results is shown in Figure 7(a) (note that in the figure, the model is drawn as a solid line and simulation results are plotted as isolated triangles). In this example, we use a bottleneck link of capacity $C = 1$ mb/s shared by two EMKC flows, which are initially separated by the maximum distance, i.e., $x_1(0) = 0, x_2(0) = C$. As seen from the figure, the number of steps predicted by (54) agrees with simulation results for a wide range of ε.

As noted in the previous section, parameter β is responsible for the convergence speed to efficiency; however, as seen in (54), it has little effect on the convergence rate to fairness (since typically $N\alpha \ll C$). In contrast, parameter α has no effect on convergence to efficiency in (52), but instead determines the convergence rate to fairness in the denominator of (54). Also observe the following interesting fact about (54) and the suitability of EMKC for high-speed networks. As C increases, the behavior of θ_M changes depending on whether N remains fixed or not. For a constant N, (54) scales linearly with C; however, if the network provider increases the number of flows as a function of C and keeps $N = \Theta(C)$, ε-fairness is reached in $\Theta(\log C)$ steps. This implies exponential convergence to fairness and very good scaling properties of EMKC in future high-speed networks. Both types of convergence are demonstrated in Figure 7(b) for constant $N = 2$ and variable $N = \lceil C/500\rceil$ (for the latter case, C is taken to be in kb/s). As the figure shows, both linear and logarithmic models obtained from (54) match simulations well.

We next compare EMKC's convergence speed to that of rate-based AIMD. Recall that rate-based AIMD(α, β) adjusts its sending rate according to the following rules assuming $\alpha > 0$ and $0 < \beta < 1$:

$$x(t) = \begin{cases} x(t - RTT) + \alpha & \text{per RTT} \\ (1-\beta)x(t - RTT) & \text{per loss} \end{cases}. \qquad (55)$$

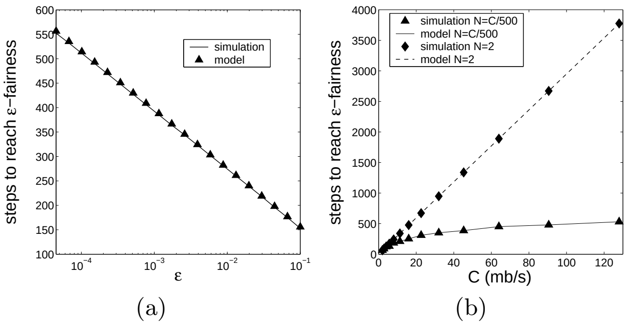


Figure 7: (a) Verification of model (54) against EMKC simulations ($C = 1$ mb/s, $\alpha = 10$ kb/s, and $\beta = 0.5$). (b) Exponential and linear rates of convergence to fairness for EMKC ($\varepsilon = 0.1$).

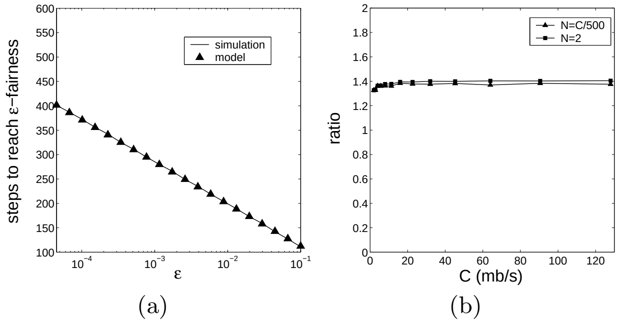


Figure 8: (a) Verification of model (56) against AIMD simulations ($C = 1$ mb/s, $\alpha = 10$ kb/s, and $\beta = 0.5$). (b) Ratio θ_M/θ_A for fixed and variable N.

Theorem 6. *Under the assumptions of Theorem 5, rate-based AIMD reaches ε-fairness in θ_A steps, where:*

$$\theta_A = \frac{(C + N\frac{\alpha}{\beta})(\log N - \log\varepsilon)}{-N\alpha\log(1-\beta)/\beta} + \Theta\Big(\frac{N\alpha}{C}\Big). \tag{56}$$

Figure 8(a) verifies that model (56) is also very accurate for a range of different ε. Notice from (54) and (56) that the speed of convergence to fairness between AIMD and EMKC differs by a certain constant coefficient. The following corollary formalizes this observation.

Corollary 3. *For the same parameters N, α, β such that $N\alpha \ll C$, AIMD reaches ε-fairness $\theta_M/\theta_A = -\log(1-\beta)/\beta$ times faster than EMKC.*

For TCP and $\beta = 0.5$, this difference is by a factor of $2\log 2 \approx 1.39$, which holds regardless of whether N is fixed or not as demonstrated in Figure 8(b). We should finally note that as term $\Theta(N\alpha/C)$ becomes large, MKC's performance improves and converges to that of AIMD.

5.3 Packet Loss

As seen in previous sections, EMKC converges to the combined stationary point $X^* = C + N\alpha/\beta$, which is above capacity C. This leads to constant (albeit usually small) packet loss in the steady state. However, the advantage of this framework is that EMKC does not oscillate or react to individual packet losses, but instead adjusts its rate in response to a *gradual* increase in $p(n)$. Thus, a small amount of FEC can provide a smooth channel to fluctuation-sensitive applications such as video telephony and various types of real-time streaming. Besides being a stable framework, EMKC is also expected to work well in wireless networks where congestion-unrelated losses will not cause sudden reductions in the rates of end-flows.

Also notice that EMKC's steady-state packet loss $p^* = N\alpha/(C\beta + N\alpha)$ increases linearly with the number of competing flows, which causes problems in scalability to a large number of flows. However, it still outperforms AIMD, whose increase in packet loss is quadratic as a function of N [21]. Furthermore, if the network provider keeps $N = \Theta(C)$, EMKC achieves *constant* packet loss in addition to exponential convergence to fairness.

Finally, observe that if the router is able to count the number of flows, zero packet loss can be obtained by adding a constant $\Delta = N\alpha/(\beta C)$ to the congestion indication function [3]. However, this method is impractical, since it needs non-scalable estimation of the number of flows N inside each router. Hence, it is desirable for the router to adaptively tune $p(n)$ so that the system is free from packet loss. One such method is AVQ (Adaptive Virtual Queue) proposed in [16], [19]. We leave the analysis of this approach under heterogeneous delays and further improvements of EMKC for future work.

6. SIMULATIONS

We next examine how to implement scalable AQM functions inside routers to provide proper feedback to MKC flows. This is a non-trivial design issue since the ideal packet loss in (44) relies on the sum of *instantaneous* rates $x_i(n)$, which are never known to the router. In such cases, a common approach is to approximate model (44) with some time-average function computed inside the router. However, as mentioned in the introduction, this does not directly lead to an oscillation-free framework since directional delays of real networks introduce various inconsistencies in the feedback loop and mislead the router to produce incorrect estimates of $X(n) = \sum_i x_i(n)$.

In what follows in this section, we provide a detailed description of various AQM implementation issues and simulate EMKC in `ns2` under heterogeneous feedback delays.

6.1 Packet Header

As shown in Figure 9, the MKC packet header consists of two parts – a 16-byte *router* header and a 4-byte *user* header. The router header encapsulates information that is necessary for the router to generate precise AQM feedback and subsequently for the end-user to adjust its sending rate. The *rid* field is a unique label that identifies the router that generated the feedback (e.g., its IP address). This field is used by the flows to detect changes in bottlenecks, in which case they wait for an extra RTT before responding to congestion signals of the new router. The *seq* field is a local variable incremented by the router each time it produces a new value of packet loss p (see below for more). Finally, the Δ field carries the length of the averaging interval used by the router in its computation of feedback.

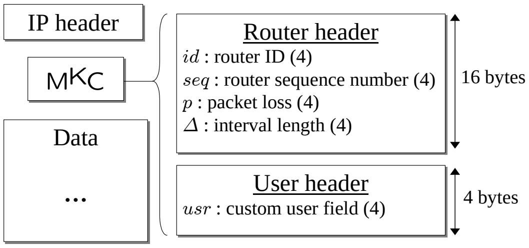


Figure 9: Packet format of MKC.

The *user* header is necessary for the end-flows to determine the rate $x_i(n - D_i)$ that was in effect RTT time units earlier. The simplest way to implement this functionality is to inject the value of $x_i(n)$ into each outgoing packet and then ask the receiver to return this field in its acknowledgments. A slightly more sophisticated usage of this field is discussed later in this section.

6.2 The Router

Recall that MKC decouples the operations of users and routers, allowing for a scalable decentralized implementation. The major task of the router is to generate its AQM feedback and insert it in the headers of all passing packets. However, notice that the router never knows the exact combined rate of incoming flows. Thus, to approximate the ideal computation of packet loss, the router conducts its calculation of $p(n)$ on a discrete time scale of Δ time units. For each packet arriving within the *current* interval Δ, the router inserts in the packet header the feedback information computed during the *previous* interval Δ. As a consequence, the feedback is retarded by Δ time units inside the router in addition to any backward directional delays $D_i^{\leftarrow}$. Since MKC is robust to feedback delay, this extra Δ time units does not affect stability of the system. We provide more implementation details below.

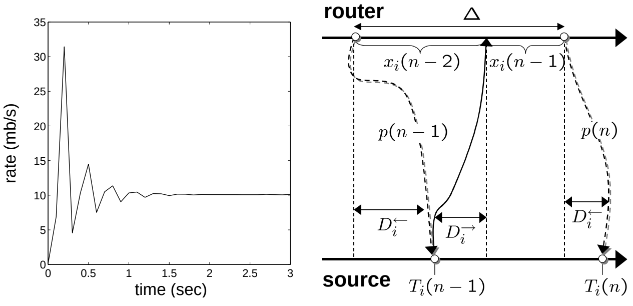
During interval Δ, the router keeps a local variable S, which tracks the total amount of data that has arrived into the queue (counting any dropped packets as well) since the beginning of the interval. Specifically, for each incoming packet k from flow i, the router increments S by the size of the packet: $S = S + s_i(k)$. In addition, the router examines whether its locally recorded estimate $\tilde{p}$ of packet loss (which was calculated in the previous interval Δ) is larger than the one carried in the packet. If so, the router overrides the corresponding entries in the packet and places its own router ID, packet loss, and sequence number into the header. In this manner, after traversing the whole path, each packet records information from the most congested link.[3]

At the end of interval Δ, the router approximates the combined arriving rate $X(n) = \sum_{i=1}^{N} x_i(n - D_i^{\rightarrow})$ by averaging S over time Δ:

$$\tilde{X} = \frac{S}{\Delta}. \tag{57}$$

Based on this information, the router computes an estimate of packet loss $p(n)$ as following:

$$\tilde{p} = \frac{\tilde{X} - C}{\tilde{X}}, \tag{58}$$



(a) (b)

Figure 10: Naive EMKC implementation: (a) one ns2 flow ($\alpha = 100$ kb/s, $\beta = 0.9$, and $\Delta = 50$ ms) passes through a bottleneck link of capacity 10 mb/s; (b) inconsistent feedback and reference rate.

where C is the capacity of the outgoing link known to the router (these functions are performed on a per-queue basis).

After computing $\tilde{p}$, the router increments its packet-loss sequence number (i.e., $seq = seq + 1$) and resets variable S to zero. Newly computed values seq and $\tilde{p}$ are then inserted into qualified packets arriving during the next interval Δ and are subsequently fed back by the receiver to the sender. The latter adjusts its sending rate as we discuss in the next section.

6.3 The User

MKC employs the primal algorithm (12)-(13) at the end-users who adjust their sending rates based on the packet loss generated by the most congested resources of their paths. However, to properly implement MKC, we need to address the following issues.

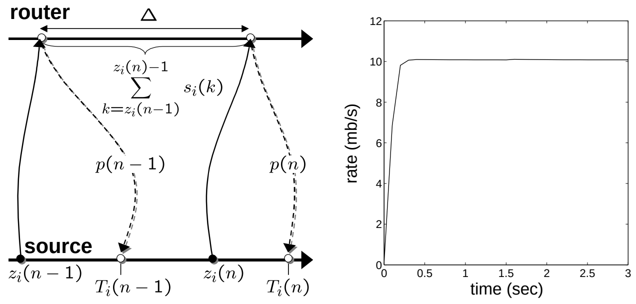
First, notice that ACKs carrying feedback information continuously arrive at the end-user and for the most part contain duplicate feedback (assuming Δ is sufficiently large). To prevent the user from responding to redundant or sometimes obsolete feedback caused by reordering, each packet carries a sequence number seq, which is modified by the bottleneck router and is echoed by the receiver to the sender. At the same time, each end-user i maintains a local variable seq_i, which records the largest value of seq observed by the user so far. Thus, for each incoming ACK with sequence seq, the user responds to it if and only if $seq > seq_i$. This allows MKC senders to pace their control actions such that their rate adjustments and the router's feedback occur on the same timescale.

Second, recall from (12)-(13) that MKC requires both the delayed feedback $\eta_i(n)$ and the delayed reference rate $x_i(n - D_i)$ when deciding the next sending rate. Thus, the next problem to address is how to correctly implement the control equation (12). We develop two strategies for this problem below.

6.3.1 Naive Implementation

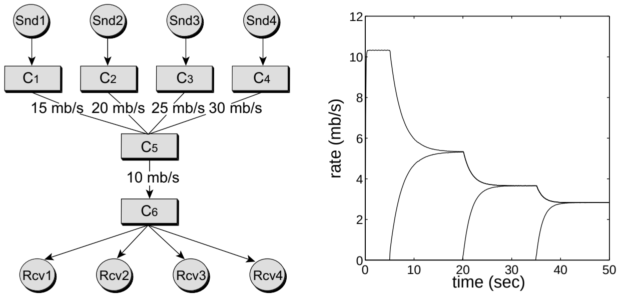
One straightforward option is to directly follow (12) based on the rate that was in effect exactly D_i time units earlier. Since round-trip delays fluctuate, the most reliable way to determine $x_i(n - D_i)$ is to carry this information in the *usr* field of each packet (see Figure 9). When the receiver echoes the *router* field to the sender, it also copies the *user* field

[3] Note that multi-path routing is clearly a problem for this algorithm; however, *all* AQM congestion control methods fail when packets are routed in parallel over several paths.



(a) (b)

Figure 11: Proper EMKC implementation: (a) graphical explanation of the algorithm; (b) result of ns2 simulations with one EMKC flow ($\alpha = 100$ kb/s, $\beta = 0.9$, and $\Delta = 50$ ms) over a link of capacity 10 mb/s.



(a) ns2 topology (b) simulation results

Figure 12: Four EMKC flows ($\alpha = 100$ kb/s and $\beta = 0.9$) with heterogeneous feedback delays share a bottleneck link of capacity 10 mb/s.

into the acknowledgment. We show the performance of this strategy via ns2 simulations in Figure 10(a), in which a single MKC flow passes through a bottleneck link of capacity 10 mb/s. We set α to 100 kb/s, β to 0.9, packet size to 200 bytes, and router sampling interval Δ to 50 ms. As seen from Figure 10(a), the sending rate converges to its stationary point in less than 2 seconds and does not exhibit oscillations in the steady state; however, the flow exhibits transient oscillations and overshoots C by over 200% in the first quarter of a second. Although this transient behavior does not affect stability of the system, it is greatly undesirable from the practical standpoint.

6.3.2 Proper Implementation

To remove the transient oscillations, we first need to understand how they are created. Notice from (57)-(58) that since the router calculates the packet loss based on the average incoming rate over interval Δ, it is possible that packets of *different* sending rates $x_i(n_1)$ and $x_i(n_2)$ arrive to the router during the same interval Δ. Denote by $T_i(n)$ the time when user i receives the n-th non-duplicate feedback $p(n)$. Since the user responds to each feedback only once, it computes new sending rates $x_i(n)$ at time instances $T_i(n)$. To better understand the dynamics of a typical AQM control loop, consider the illustration in Figure 10(b). In the figure, the router generates feedback $p(n-1)$ and $p(n)$ exactly Δ units apart. This feedback is randomly delayed by $D_i^{\leftarrow}$ time units and arrives to the user at instances $T_i(n-1)$ and $T_i(n)$, respectively. In response to the first feedback, the user changes its rate from $x_i(n-2)$ to $x_i(n-1)$; however, the router observes the second rate only at time $T_i(n-1)+D_i^{\rightarrow}$. At the end of the n-th interval Δ, the router averages both rates $x_i(n-2)$ and $x_i(n-1)$ to produce its feedback $p(n)$ as shown in the figure.

When the control loop is completed, the user is misled to believe that feedback $p(n)$ refers to a single rate $x(n-1)$ and is forced to incorrectly compute $x(n)$. This inconsistency is especially pronounced in the first few control steps during which the flows increase their rates exponentially and the amount of error between the actual rate and the reference rate is large.

Instead of changing the router, we modify the end-users to become more sophisticated in their processing of network feedback. The key is to allow end-users to accurately estimate their own contribution to $\tilde{X}$ and determine their *average* rates seen by the router during interval Δ. For each outgoing packet k, MKC sender i places the packet's sequence number k in the *usr* field and records in local memory the size of the packet $s_i(k)$ and its sequence number k. Upon arrival of the n-th non-duplicate feedback at time $T_i(n)$, the end-flow extracts the *usr* field from the acknowledgment and records its value in variable $z_i(n)$, which is the sequence number of the packet that generated feedback $p(n)$. To compute the new rate $x_i(n)$, the user calculates the amount of data that it has transmitted between packets $z_i(n-1)$ and $z_i(n)-1$ and normalizes the sum by Δ, which is exactly the average rate used by the router in generation of $p(n)$.

To visualize this description, consider Figure 11(a), in which the end-flow is about to decide its sending rate $x_i(n)$ at time $T_i(n)$. Notice in the figure that feedback $p(n)$ is based on all packets of flow i with sequence numbers between $z_i(n-1)$ and $z_i(n)-1$. Through the use of $z_i(n)$, we obtain a projection of the time-interval used by the router in its computation of $p(n)$ onto the sequence-number axis of the end user.[4] Given the above discussion, the user computes its average rate as:

$$\bar{x}_i(n) = \frac{1}{\Delta} \sum_{k=z_i(n-1)}^{z_i(n)-1} s_i(k), \qquad (59)$$

and utilizes it in its control equation:

$$x_i(n) = \bar{x}_i(n) + \alpha - \beta\eta_i(n)\bar{x}_i(n). \qquad (60)$$

Next, we turn our attention to the ns2 simulation in Figure 11(b) and examine the performance of this strategy with a single flow. The figure shows that (59)-(60) successfully eliminates transient oscillations and offers fast, monotonic convergence to the steady state. Our next example shows the performance of the new method (59)-(60) with multiple flows. The simulation topology of this example is illustrated in Figure 12(a): four EMKC flows identical to that in Figure 10(a) share the same bottleneck link of capacity 10 mb/s. The round trip delays of the four flows are 50 ms, 60 ms, 70 ms, and 80 ms, respectively, and the sampling intervals Δ

[4] Note that this approach is robust to random delays, but may be impeded by severe packet loss at the router.

of routers C1-C5 are 120 ms, 140 ms, 160 ms, 180 ms, and 100 ms, respectively. At time 0, the first flow starts at 125 kb/s and monotonically converges to bottleneck capacity in less than 0.4 seconds as seen in Figure 12(b). Five seconds later, the second flow joins at initial rate 150 kb/s. The figure shows that the system is immediately re-stabilized in the new stationary point and the individual flows quickly converge to fairness without oscillations. This behavior is repeated when the other two flows join the network and the system regains stability and fairness with ideal performance (i.e., monotonically).

7. CONCLUSION

This paper investigated the properties of Internet congestion controls under non-negligible directional feedback delays. We focused on the class of control methods with symmetric Jacobians and showed that all such systems are stable under heterogeneous delays. To construct a practical congestion control system with a symmetric Jacobian, we made two changes to the classic discrete Kelly control and created a max-min version we call MKC. Combining the latter with a negative packet-loss feedback, we developed a new controller EMKC and showed in theory and simulations that it offers smooth sending rate and fast convergence to efficiency. Furthermore, we demonstrated that EMKC's convergence rate to fairness is exponential when the network provider scales the number of flows N as $\Theta(C)$ and linear otherwise. From the implementation standpoint, EMKC places very little burden on routers, requires only two local variables per queue and one addition per arriving packet, and allows for an easy implementation both in end-to-end environments and under AQM support. Our future work involves improvement of the convergence speed to fairness and design of pricing schemes for EMKC to achieve loss-free performance regardless of the number of flows N.

8. REFERENCES

[1] R. Bronson. *Schaum's Outline of Theory and Problems of Matrix Operations*. McGraw-Hill, 1988.

[2] D.-M. Chiu and R. Jain, "Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks," *Computer Networks and ISDN Systems*, 17(1):1–14, June 1989.

[3] M. Dai and D. Loguinov, "Analysis of Rate-Distortion Functions and Congestion Control in Scalable Internet Video Streaming," *ACM NOSSDAV*, June 2003.

[4] S. Deb and R. Srikant, "Global Stability of Congestion Controllers for the Internet," *IEEE Transactions on Automatic Control*, 48(6):1055 – 1060, June 2003.

[5] S. Floyd, "High-speed TCP for Large Congestion Windows," *RFC 3649*, December 2003.

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

[7] S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance," *IEEE/ACM Transactions on Networking*, 1(4):397–413, January 1993.

[8] C. Jin, D. Wei, and S. H. Low, "FAST TCP: Motivation, Architecture, Algorithms, Performance," *IEEE INFOCOM*, March 2004.

[9] R. Johari and D. K. H. Tan, "End-to-End Congestion Control for the Internet: Delays and Stability," *IEEE/ACM Transactions on Networking*, 9(6):818–832, December 2001.

[10] K. Kar, S. Sarkar, and L. Tassiulas, "A Simple Rate Control Algorithm for Maximizing Total User Utility," *IEEE INFOCOM*, April 2001.

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

[12] W. G. Kelley and A. C. Peterson. *Difference Equations*. Harcourt / Academic Press, 2001.

[13] F. P. Kelly, "Charging and Rate Control for Elastic Traffic," *European Transactions on Telecommunications*, 8(1):33–37, January 1997.

[14] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan, "Rate Control for Communication Networks: Shadow Prices, Proportional Fairness and Stability," *Journal of the Operational Research Society*, 49(3):237–252, March 1998.

[15] T. Kelly, "Scalable TCP: Improving Performance in High-speed Wide Area Networks," *First International Workshop on Protocols for Fast Long-Distance Networks*, February 2003.

[16] S. Kunniyur and R. Srikant, "Analysis and Design of an Adaptive Virtual Queue (AVQ) Algorithm for Active Queue Management," *ACM SIGCOMM*, August 2001.

[17] S. Kunniyur and R. Srikant, "A Time-Scale Decomposition Approach to Adaptive Explicit Congestion Notification (ECN) Marking," *IEEE Transactions on Automatic Control*, 47(6):882 – 894, June 2002.

[18] S. Kunniyur and R. Srikant, "End-to-End Congestion Control Schemes: Utility Functions, Random Losses and ECN Marks," *IEEE/ACM Transactions on Networking*, 11(5):689 – 702, October 2003.

[19] S. Kunniyur and R. Srikant, "Stable, Scalable, Fair Congestion Control and AQM Schemes that Achieve High Utilization in the Internet," *IEEE Transactions on Automatic Control*, 48(11):2024–2029, November 2003.

[20] C.-K. Li and R. Mathias, "The Determinant of the Sum of Two Matrices," *Bull. Australian Math. Soc.*, 52(3):425–429, 1995.

[21] D. Loguinov and H. Radha, "End-to-End Rate-Based Congestion Control: Convergence Properties and Salability Analysis," *IEEE/ACM Transactions on Networking*, 11(5):564–577, August 2003.

[22] S. H. Low, "A Duality Model of TCP and Queue Management Algorithms," *IEEE/ACM Transactions on Networking*, 11(4):525–536, August 2003.

[23] S. H. Low and D. E. Lapsley, "Optimization Flow Control I: Basic Algorithm and Convergence," *IEEE/ACM Transactions on Networking*, 7(6):861–874, December 1999.

[24] L. Massoulié, "Stability of Distributed Congestion Control with Heterogeneous Feedback Delays," *IEEE/ACM Transactions on Networking*, 47(6):895–902, June 2002.

[25] F. Paganini, J. Doyle, and S. H. Low, "A Control Theoretical Look at Internet Congestion Control," *The Mohammed Dahleh Symposium*, 2002.

[26] G. Vinnicombe, "On the Stability of End-to-End Congestion Control for the Internet," Technical Report CUED/F-INFENG/TR.398, University of Cambridge, December 2000.

[27] G. Vinnicombe, "Robust congestion control for the Internet," Technical report, University of Cambridge, 2002.

[28] L. Xu, K. Harfoush, and I. Rhee, "Binary Increase Congestion Control for Fast, Long Distance Networks," *IEEE INFOCOM*, March 2004.

[29] Y. R. Yang and S. S. Lam, "General AIMD Congestion Control," *IEEE ICNP*, November 2000.

[30] L. Ying, G. E. Dullerud, and R. Srikant, "Global Stability of Internet Congestion Control with Heterogeneous Delays," *American Control Conference*, June 2004.

[31] Y. Zhang, S.-R. Kang, and D. Loguinov, "Delayed Stability and Performance of Distributed Congestion Control (extended version)," *Texas A&M Technical Report*, August 2004.