Stability of Network Congestion Control with Asynchronous Updates

Han Cai[†] and Do Young Eun[†]

Abstract— **For network congestion control with many flows, current theoretical models predict worse performance than what the empirical study shows. In this paper, we introduce asynchronism among many flows to the traditional fluid-based model and develop an asynchronous model for network congestion control where each end-user sees slightly different snapshots of the network. Based on our model, we prove that the system is always stable when there are many flows with asynchronous updates. Both our theoretical and numerical results confirm that the asynchronism plays a dominant role in the stability of network congestion control and our asynchronous model reflects the reality far better than any other traditional fluid models with synchronous updates.**

I. Introduction

Transmission Control Protocol (TCP) and Active Queue Management (AQM) are the two major components of Internet congestion control and cooperate in a distributed manner to form a feedback control system. Implemented on each end-node, TCP specifies packet transmission rate based on the signal from the network. Routers, on the other hand, calculate the level of congestion signal from what they observe and return this feedback signal to end-nodes, which triggers the end-nodes to adjust their injection rates into the network. For these feedback networked systems, stability has been the most important measure and crucial to the overall network performance as it ensures smoothness or convergence of average 'state' of the system to its equilibrium point.

Fluid models have received most attention in analyzing the stability of network congestion control. The stability criterion of the system conveniently offers guidelines on how to design TCP/AQM systems [1], [2], [3], [4], [5] to the benefit of higher bandwidth utilization and smaller variations in delay or throughput [3]. In addition, it has been shown that the fluid model becomes accurate and a faithful representation of a system as the number of flows and the link capacity increase at the same time [4], [6]. This implies that the stability property of the system is mostly invariant with respect to the system size.

However, recent empirical studies have shown [7], [8] that the system performance critically depends on the number of flows. In particular, it is observed [7] that although there exists global synchronization among flows in a small system under drop-tail policy at the routers, the synchronization disappears and all the flows become almost independent of each other when the number of flows in the system is larger than certain value, say, few hundreds, thus resulting in higher link utilization and smaller buffer size requirement. In other words, the system tends to be more 'stable' for larger number of flows in the system. This partially supports the long-held observation that real networks tend to perform better than what the theory suggests [9], [10] and puts forward the following questions: How can we fill this gap between the theory and the reality? What is lost in the current fluid modeling approach? Can we modify the current model to reflect the reality and obtain better design guidelines?

In this paper, we show that we can fill this gap by introducing asynchronism to the current fluid model. In fact, the existence of asynchronism in a system with many flows is not new. For instance, even when many flows sharing the same link have the same round-trip-time (RTT), very small variation in router's processing time is shown to be sufficient to prevent synchronization [11]. Similarly, many other sources of randomness and imperfection in the network algorithms also contribute to the asynchronism in part. Up till now, however, only simulation and experiment results are available [7], [8] in showing the existence of asynchronism and its influence on system stability, and how to introduce it to the current fluid model is still an open problem. In this paper, we explicitly take the asynchronism among many flows into account and develop a simple fluid-based model with asynchronous updates, where each end-user sees slightly different pictures of the network status (receives slightly different congestion signal from others). We prove that under our asynchronous model, a system becomes always linearly stable *regardless of the system configuration* as the number of flows increases. Our theoretical results thus signify the importance of asynchronism in the stability analysis and design of networked systems. Our results also provide a theoretical explanation on the empirical observations in [7] and in some sense eliminate the aforementioned discrepancy between theory and practice.

Due to similar reason, the problem of how asynchronism affects the system stability has been an important consideration in many kinds of control systems other than network congestion control [12], [13], [14]. Research work in iterative methods for solving a set of linear equations have provided the basis for modeling and analyzing asynchronous systems. As will be shown later on, we point out that our approach to the asynchronous model for congestion control is very different from all those work on the asynchronous system in the literature.

The rest of the paper is organized as follows. In Section II, we provide some background on the fluid model and present related work in asynchronous iteration and its difference from our work. Section III provides stability analysis for synchronous model. We then develop our asynchronous model and prove its stability in Section IV. Section V provides simulation and numerical results, and Section VI concludes.

This work was supported in part by NSF CAREER Award CNS-0545893.

[†]Department of Electrical and Computer Engineering, North Carolina State University, Raleigh NC 27695. Email: {hcai2, dyeun}@ncsu.edu.

II. Preliminaries

A. Fluid Model for Congestion Control with Many Flows

Fluid model is widely used for TCP congestion control to describe how the average rates evolve over time and determine its convergence property or stability. Consider a link with capacity NC shared by N flows (end-users). Let each RTT be normalized to one time slot, $x_j(t)$ be the rate of flow j (or the number of packets to be transmitted) during the t^{th} time slot ($j = 1, 2, ..., N;\ t = 1, 2, ...$), and $\bar{S}(t) = \sum_{k=1}^{N} x_k(t)/N$ present the *average* rate seen at the link. Assume that congestion signal depends only on $\bar{S}(t)$. Then, the fluid model for flow j becomes

$$\begin{aligned} x_j(t{+}1) &= f\left(x_j(t)\right)\left(1{-}p\left(\bar{S}(t)\right)\right){+}g\left(x_j(t)\right)p\left(\bar{S}(t)\right) \\ &= h_j(x_1(t), x_2(t), ..., x_N(t)), \end{aligned} \tag{1}$$

where $p(x)$ is the marking function denoting the probability that the flow receives congestion signal when the average rate is x. When the current rate is x, the rate at the next time slot becomes $g(x)$ under congestion and $f(x)$ otherwise. We then impose the following assumptions on these functions:

(A1) $f(x) = x + \alpha$ for some constant $\alpha > 0$.
(A2) $0 < g(x) < x$ and $0 < g'(x) < 1$.
(A3) $p(x) : [1, \infty) \to [0, 1]$ is non-decreasing in x.

Remark 1: (A1) means that the rate increases linearly under no congestion. For instance, we have $\alpha = 1$ in case of AIMD as in the current TCP. Assumption (A2) means that in case of congestion, (i) the new rate should be smaller than the current one ($g(x) < x$) and (ii) for current rate $x_1 < x_2$, we have $g(x_1) < g(x_2)$ and also $x_1 - g(x_1) < x_2 - g(x_2)$, i.e., the sender should reduce more from a higher sending rate. This includes $g(x) = \beta x$ with $\beta \in (0, 1)$ as in AIMD and $g(x) = x - \sqrt{x}$ for $x \geq 1$.

From (1), the fluid model for one flow becomes

$$x(t{+}1){=}f(x(t))(1{-}p(x(t))){+}g(x(t))p(x(t)){=}F(x(t)). \tag{2}$$

The system (2) is linearly stable if and only if $|F'(x^*)| < 1$ where x^* is the equilibrium point of (2), i.e., $x^* = F(x^*)$.

From (A1)–(A3), after some manipulation, it follows that the condition $|F'(x^*)| < 1$ is equivalent to

$$p(x^*)\left(1 - g'(x^*)\right) + \alpha p'(x^*)/p(x^*) < 2. \tag{3}$$

B. Related Work in Asynchronous Iteration

The following model is one of the most popular models used in the study of asynchronous iteration [15]:

$$x_i(t{+}1) = \begin{cases} \sum_{j=1}^{N} a_{ij} x_j\left(k - d(i,j,t)\right), & \text{if } i \in S(t), \\ x_i(t), & \text{otherwise,} \end{cases} \tag{4}$$

where $A = \{a_{ij}\} \in \mathbb{R}^{N\times N}$ is the transition matrix, $d(i, j, t)$ are nonnegative integers denoting iteration delay, and $S(t) \in \{1, 2, ..., N\}$ is the updating set indicating which flows should be updated at time t.

Let $|A| = \{|a_{ij}|\}$ be the matrix whose elements are $|a_{ij}|$ and $\rho(A)$ be the spectral radius of matrix A, i.e.,

$$\rho(A) = \max_i |\lambda_i|, \tag{5}$$

where λ_i ($i = 1, 2, ..., N$) are eigenvalues of matrix A. The following result has received much attention in the study of asynchronous iteration [16].

Theorem 1: If $\rho(|A|) < 1$, the system in (4) is asymptotically stable under all possible iteration delays and updating sets. If $\rho(|A|) \geq 1$, there exists a sequence of iteration delays and updating sets for (4) to be not asymptotically stable. □

Other related work about the asynchronous iteration includes parallel computations, neutral networks, among others [15], [17], [18], [19]. Most of these studies focus on the case of $\rho(|A|) = 1$ and try to construct an asynchronous updating scheme for which the system is unstable, find a group of asynchronous updating schemes for which the system is stable when A is symmetric, etc. In short, they are interested in studying the effect of *different asynchronous schemes for fixed transition matrix A* as required by the application. For example, in parallel computations, one of the optimization objectives is to minimize the time to finish computations under fixed 'rules' [17], i.e., to find the optimal asynchronous scheme under a fixed transition matrix.

In contrast, our interest lies in the effect of the number of flows on network stability. For example, suppose there are N flows sharing the network. Then, the transition matrix becomes $N \times N$ matrix. At the same time, note that the randomness in the network makes any asynchronous updating scheme possible. However, since we mainly focus on the system stability as N increases, in this paper, we consider *different transition matrices under a fixed asynchronous updating scheme*.

III. Stability analysis for synchronous model

By linearizing N equations ($j = 1, 2, ..., N$) in (1) around its equilibrium point $\vec{x}^* = (x_1^*, \ldots, x_N^*)^T$, we obtain

$$\vec{x}(t+1) = F\vec{x}(t), \tag{6}$$

where $\vec{x}(t) = (x_1(t), x_2(t), ..., x_N(t))^T$ and $F = \{f_{ij}\}$ is the $N \times N$ transition matrix with

$$f_{ij} = (\partial h_i/\partial x_j)\Big|_{\vec{x}=\vec{x}^*} \tag{7}$$

where h_i is defined in (1). We then have the following.

Proposition 1: The system in (1) is linearly stable if and only if the condition in (3) is satisfied. □

Remark 2: Note that under the synchronous model, the system stability does not depend on the number of flows N, which contradicts the empirical observation in [7], [8].

Proof: We only need to consider the linearized version of (1) as in (6). It is well-known [15] that a linear system as in (6) is stable if and only if $\rho(F) < 1$ [15] where $\rho(F)$ is defined in (5).

Since all N flows are symmetric, their equilibrium points are the same, i.e., $x_1^* = x_2^* = ... = x_N^* = x^*$. Substituting this into N equations in (1) yields

$$x^* = f(x^*)\left(1 - p(x^*)\right) + g(x^*)p(x^*)$$

where $f(x)$ and $g(x)$ satisfy (A1)–(A2). Define

$$a = p(x^*)\left(1 - g'(x^*)\right), \quad c = \alpha p'(x^*)/p(x^*), \tag{8}$$

then from (A1)–(A3), we have $0 < a < 1$ and $c \geq 0$. Thus, the transition matrix can be written as $F=\{f_{ij}\}$ where

$$f_{ij}=\big(1-a-(c/N)\big)1_{\{i=j\}}+(-c/N)1_{\{i\neq j\}}. \quad (9)$$

Eigenvalues of F will be identified by the following result.

Lemma 1: [20] Suppose $N\times N$ matrix A is nonsingular. Then, for any $N\times 1$ column vectors $\mathbf{d}$ and $\mathbf{e}$, we have

$$\det(A+\mathbf{d}\mathbf{e}^T)=\det(A)(1+\mathbf{e}^TA^{-1}\mathbf{d}) \quad (10)$$

Using (10) and $A=\big[(1-a-\lambda)N/c\big]I$, $\mathbf{d}=\mathbf{e}=(1\ 1\ \ldots\ 1)^T$,

$$\det(F-\lambda I)=(1-a-\lambda)^{N-1}(1-a-c-\lambda) \quad (11)$$

Hence, the matrix F has $N-1$ identical eigenvalues at $\lambda=1-a$ and another eigenvalue at $\lambda=1-a-c$, where a and c are from (8). Since $0<a<1$, we have $|1-a|<1$. Thus, the system is linearly stable if and only if

$$|\lambda_N|=|1-a-c|<1,$$

which is identical to (3) since $0<a<1$ and $c\geq 0$. This completes the proof. ■

IV. Stability analysis with asynchronous updates

A. Asynchronous Model Description

Proposition 1 in the previous section asserts that the stability of the system in (1) does not change as the number of flows N increases. This is in fact well expected since, by appealing to the law of large numbers type arguments, the fluid model in (1) becomes more and more accurate as N increases and thus its stability property will be largely insensitive to N. However, still, this model does not explain the observed discrepancy between the model and the reality. In this section, we develop a simple, yet effective asynchronous model to fill this gap and provide better design guidelines for congestion control with many flows.

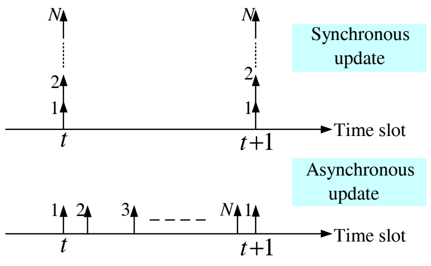


Fig. 1. Synchronous and asynchronous updates

Note that in system (1), every flow makes decision at $(t+1)^{th}$ time slot based on the same feedback signal $p\big(\sum_{k=1}^{N}x_k(t)/N\big)$. In other words, all the N flows update at the same time instance and see the same picture of network status. See Figure 1 for illustration. In reality, however, the situation is far more complicated. While drop-tail policy surely contributes to the synchronization among flows(by dropping all the incoming packets from different flows at time of congestion), there are many other factors that help make all the flows 'out-of-synch'. For example, each source updates its window size (sending rate) per each return acknowledgement (ACK) packet rather than per RTT, i.e., the real updating process is much finer than what the model shows. In addition, each of these ACK packets that carry a 'snapshot' of the network status, will share its return path with other crossing traffic and suffer small delay variations. All in all, the network is more like a chaotic system in which different flows see slightly different pictures of the network even when they have the same updating period (RTT). As in Figure 1, under asynchronous update scheme, N flows update consecutively, and each flow's change will affect the network status at the router. This in turn affects the congestion signal for other subsequent flows. In this way, all the N flows have the same update period, but they are out-of-synch.

In our asynchronous model, N flows update in order in one time slot, and the most up-to-date information of each flow is shared immediately by all the other subsequent flows. Let $y_j(t)$ be the rate of flow j during the t^{th} time slot $(j=1,2,...,N;\ \ t=1,2,...)$. Then, our asynchronous fluid model for flow j is

$$\begin{aligned} y_j(t{+}1) \ &= \ f\Big(y_j(t)\Big)\Big[1{-}p\Big(\frac{\sum_{k=1}^{j-1}y_k(t{+}1){+}\sum_{k=j}^{N}y_k(t)}{N}\Big)\Big] \\ &+ \ g\Big(y_j(t)\Big)p\Big(\frac{\sum_{k=1}^{j-1}y_k(t{+}1){+}\sum_{k=j}^{N}y_k(t)}{N}\Big) \quad (12) \end{aligned}$$

where f,g,p satisfy (A1)–(A3). Since Section (III) has already given the linearizing process, in stead of going through the similar process again, we next provide two equivalent descriptions of linearized versions of (12) based on its relationship with the synchronous model (1).

1) Asynchronous Model (Description 1): In contrast to the synchronous model, now we update step by step. Let

$$\vec{y}^{\,j}(t{+}1)=(y_1(t{+}1),\ldots,y_j(t{+}1),y_{j+1}(t),\ldots,y_N(t))^T, \quad (13)$$

and F_i $(i=1,...N)$ be transition matrix for each step, then instead of (6), now we have $\vec{y}^{\,1}(t{+}1)=F_1\vec{y}^{\,N}(t)$, $\vec{y}^{\,2}(t{+}1)=F_2\vec{y}^{\,1}(t{+}1)$, ... $\vec{y}^{\,N}(t{+}1)=F_N\vec{y}^{\,N-1}(t{+}1)$. On i^{th} step, only y_i is updated, and other flows remain the same, hence the i^{th} row of F_i is the same as F in (9), and the other rows of F_i is the same as identity matrix I.

Define $\vec{y}(t)=(y_1(t),...,y_N(t))^T$ and note that $\vec{y}(t)=\vec{y}^{\,N}(t)$ from (13). The asynchronous model can then be written as

$$\vec{y}(t+1)=(F_NF_{N-1}...F_1)\vec{y}(t), \quad (14)$$

i.e., now the transition matrix is $F_NF_{N-1}...F_1$, and the system is stable if and only if $\rho(F_NF_{N-1}...F_1)<1$.

2) Asynchronous Model (Description 2): Rewrite the original synchronous model (6) as

$$\vec{y}(t+1)=D_F\vec{y}(t)-L_F\vec{y}(t)-U_F\vec{y}(t)$$

where $D_F=(1-a-c/N)I$ is a diagonal matrix, L_F is a strictly lower triangular matrix with all elements being c/N, and U_F is a strictly upper triangular matrix.

Compare (1) and (12), the only difference of asynchronous model from the synchronous one, is that when flow j $(j=$

$1, 2, ...N$) updates its sending rate during the $(t+1)^{th}$ time slot, any flow k ($k < j$) who has already updated in this time slot will contribute its most up-to-date information to flow j's congestion signal. Hence, the asynchronous model can be written as

$$\vec{y}(t+1) = D_F\vec{y}(t) - L_F\vec{y}(t+1) - U_F\vec{y}(t),$$

which leads to

$$\vec{y}(t+1) = (I + L_F)^{-1}(D_F - U_F)\vec{y}(t) = J\vec{y}(t), \quad (15)$$

where $J = (I + L_F)^{-1}(D_F - U_F)$ is now our transition matrix for the asynchronous model.

Remark 3: For a fixed $N \in \mathbb{N}$, let $A = F$, $d(i, j, t) = 0 \times 1_{\{j \geq i\}} - 1 \times 1_{\{j<i\}}$, $S(t) = \{1, 2, ..., N\}$ in (4), we can get (15). This scheme is called Gauss-Seidel iteration [15]. But, as mentioned earlier, we study the stability of system with various transition matrices according to different N under a fixed updating scheme, which turns out to be Gauss-Seidel iteration.

B. Stability Analysis for Asynchronous Model

Based on the argument in the proof of Proposition 1, the equilibrium point $\vec{y}^* = (y_1^*, \dots, y_N^*)$ for (12) satisfies $y_1^* = \dots = y_N^* = y^*$, which again is equal to x^*, the equilibrium point for the synchronous model in (1).

We now present our main result.

Theorem 2: The system in (15) is linearly stable if

$$p(y^*)\left(1 - g'(y^*)\right) + \alpha p'(y^*)/\big(Np(y^*)\big) < 2. \quad (16)$$

□

Remark 4: The only difference between (16) and (3) is the $1/N$ factor for $\alpha p'(y^*)/p(y^*)$ in (16). Note that $p(y^*)(1 - g'(y^*))$ is always less than 1 from (A2) and (A3), so the term $c = \alpha p'(y^*)/p(y^*)$ plays a key role in deciding whether or not the system is linearly stable. (16) shows that regardless of the values of c, the system will be always stable as long as N is larger than certain value.

Proof: It suffices to show that under (16) we have $\rho(J) < 1$ [15], which means all its eigenvalues lie inside the unit circle on complex plane. For any $\lambda \in \mathbb{C}$, we have

$$\begin{aligned}&\det(\lambda I - J)\\ &= \det\left(\lambda I - (I + L_F)^{-1}(D_F - U_F)\right)\\ &= \det\left((I + L_F)^{-1}\right)\det\left(\lambda(I + L_F) - (D_F - U_F)\right)\\ &= \det\left(\lambda(I + L_F) - (D_F - U_F)\right), \quad (17)\end{aligned}$$

where (17) is from $\det\left((I+L_F)^{-1}\right) = [\det\left((I+L_F)\right)]^{-1} = 1$ since I is diagonal matrix, L_F is strictly lower triangular matrix.

Define matrix $H(\lambda) = \lambda(I+L_F) - (D_F - U_F)$ as a function of λ. If there is no λ, $|\lambda| \geq 1$ for which $\det(H(\lambda)) = 0$, then all J's eigenvalues must be inside the unit circle, hence the system (15) is linearly stable. Thus, we only need to find conditions under which all the roots of $\det(H(\lambda)) = 0$ are inside the unit circle.

Similarly to (8), define

$$a = p(y^*)\left(1 - g^{'}(y^*)\right), \quad c = \alpha p^{'}(y^*)/p(y^*) \quad (18)$$

Let $\kappa_1 = \lambda - (1-a)$, $\kappa_2 = c(\lambda - 1)/N$, $\mathbf{e} = (1, 1, \dots, 1)^T$, and $A = \{a_{ij}\}$ where $a_{ij} = \kappa_2 1_{\{i>j\}} + \kappa_1 1_{\{i=j\}}$. Then, we have

$$H(\lambda) = A + (c/N)\mathbf{e}\mathbf{e}^T.$$

In order to apply Lemma 1 to get $\det(H(\lambda))$, the matrix A must be non-singular, i.e. $\kappa_1 \neq 0$. If $\kappa_1 = 0$, we have $\lambda = 1 - a \in (0, 1)$ from (A1)–(A3), which is inside the unit circle. Thus, from now on, we can assume that $\kappa_1 \neq 0$ and the matrix A is non-singular. Moreover, when $\lambda = 1$, $H(\lambda) = aI + (c/N)\mathbf{e}\mathbf{e}^T$. From (10), $\det(H(\lambda)) = a^N\left(1 + \frac{c}{a}\right) \neq 0$, which means $\lambda = 1$ is not the root of $\det(H(\lambda)) = 0$, hence we can also safely assume that $\lambda \neq 1$, i.e., $\kappa_2 \neq 0$.

From (10), we get

$$\det(H(\lambda)) = \det(A)\left(1 + \frac{c}{N}\mathbf{e}^T A^{-1}\mathbf{e}\right) \quad (19)$$

Clearly, $\det(A) = \kappa_1^N$. As for $A^{-1} = \{a'_{ij}\}$, let $\gamma = \kappa_2/\kappa_1$, $\gamma_i = -\gamma(1-\gamma)^{i-1}$ ($i \in \mathbb{N}$), where by assumption $\kappa_{1,2} \neq 0$, γ is well defined and $\gamma \neq 0$, then

$$a'_{ij} = (\gamma_{i-j}/\kappa_1)1_{\{i>j\}} + (1/\kappa_1)1_{\{i=j\}}.$$

Hence, $\det(H(\lambda)) = \kappa_1^N\left[1 + \frac{c/N}{\kappa_1} \times \frac{1-(1-\gamma)^N}{\gamma}\right]$. Under $\kappa_1 \neq 0$, assume there exists λ' ($|\lambda'| \geq 1$) for which $\det(H(\lambda')) = 0$, i.e.

$$1 + \frac{c/N}{\kappa_1} \times \frac{1 - (1-\gamma)^N}{\gamma} = 0.$$

Substituting $\kappa_1 = \lambda' - (1-a)$, $\kappa_2 = \frac{c}{N}(\lambda' - 1)$, $\gamma = \frac{\kappa_2}{\kappa_1}$ gives

$$(\lambda')^{1/N} = 1 - \frac{(c/N)(\lambda' - 1)}{\lambda' - (1-a)},$$

and thus

$$|(\lambda')^{1/N}| = |\lambda'|^{1/N} = \left|1 - \frac{c(\lambda' - 1)/N}{\lambda' - (1-a)}\right|. \quad (20)$$

Let $\lambda' = \lambda_1 + i\lambda_2$, $\lambda_1, \lambda_2 \in R$, with $\lambda_1^2 + \lambda_2^2 \geq 1$. Since $|\lambda'| > 1$, $|\lambda'|^{1/N} > 1$ for any N. So, from (20), we have

$$[\lambda_1(1 - \frac{c}{N}) - (1 - a - \frac{c}{N})]^2 + [\lambda_2(1 - \frac{c}{N})]^2 \geq [\lambda_1 - (1-a)]^2 + \lambda_2^2,$$

which becomes

$$[\lambda_1 - (1 - \frac{a}{2 - (c/N)})]^2 + \lambda_2^2 \leq \frac{a^2}{(2 - (c/N))^2}. \quad (21)$$

Suppose

$$1 - a/(2 - c/N) < 1 \text{ and } a/(2 - c/N) < 1, \quad (22)$$

then (21) becomes a circle as shown in Figure 2.

Note that (22) holds when $a + c/N < 2$. From (18), this is equivalent to (16). Under condition (16), the only intersection of solid-line circle (21) in Figure 2 with the region $|\lambda'| \geq 1$ or $\lambda_1^2 + \lambda_2^2 \geq 1$ is $(\lambda_1, \lambda_2) = (1, 0)$. However, we already proved that $\lambda' = 1$ is not the root for $\det(H(\lambda')) = 0$. Consequently, (16) is the sufficient condition for all the roots of $\det(H(\lambda)) = 0$ to be strictly inside the unit circle. This completes the proof. ■

To see the difference between the synchronous and the asynchronous models, we take an example as follows. Set

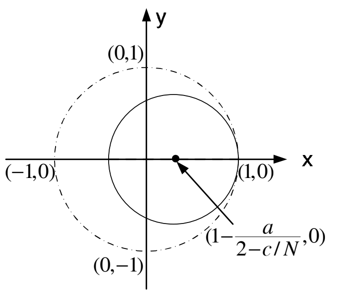


Fig. 2. (λ_1, λ_2) satisfying (21) under the condition (22).

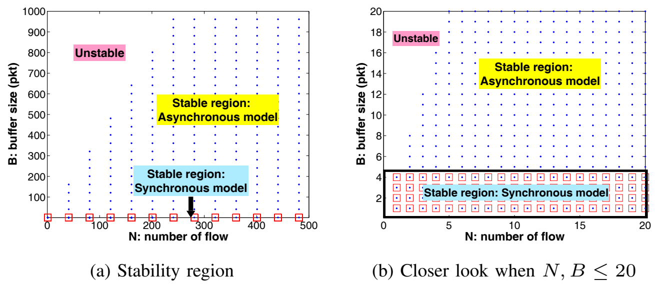


Fig. 3. Stability region for synchronous vs. asynchronous model

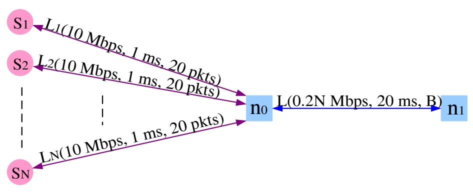

S1
S2
SN
n0
n1
L1(10 Mbps, 1 ms, 20 pkts)
L2(10 Mbps, 1 ms, 20 pkts)
LN(10 Mbps, 1 ms, 20 pkts)
L(0.2N Mbps, 20 ms, B)

Fig. 4. Network configuration for ns-2 simulation.

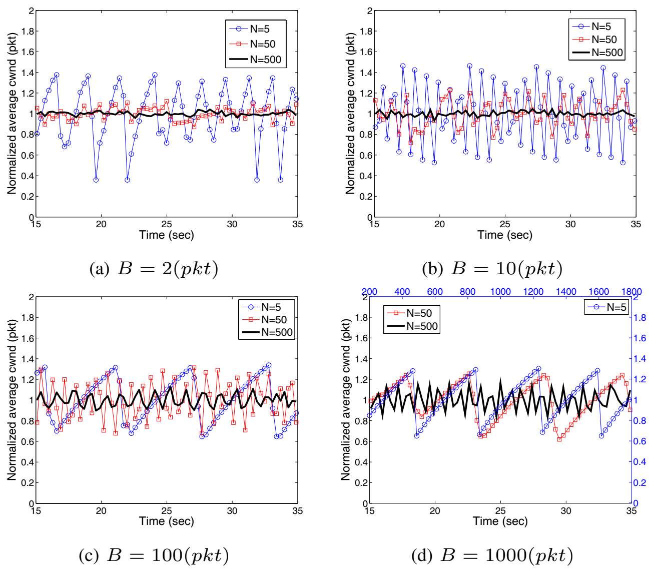


Fig. 5. Effect of the number of flows N and the buffer size B on system stability via ns-2.

the capacity per flow to $C = 0.2$ Mbps, $RTT = 42ms$, and the size of each packet to $L = 4800$ bits. Then, under buffer size B (packets), the probability of packet drop is well approximated by that of $M/M/1/B$ queue [2], [21], i.e.,

$$p(x) = \rho^B(1-\rho)/(1-\rho^{B+1}), \tag{23}$$

where $C_{\text{per-flow}} = C \times RTT/L + B/N$ and $\rho = x/C_{\text{per-flow}}$.

Figure 3 shows the set of operating points (N, B) for which the synchronous/asynchronous model is stable. Specifically, for given N and B, we first obtain the equilibrium point by solving $x^* = F(x^*)$ where $F(x)$ is from (2). As mentioned before, since N flows are symmetric, their equilibrium points are the same, independently of which model we use. Then, we decide whether this operating point is linearly stable or not: if (3) holds, then it is stable under both synchronous and asynchronous models; or if only (16) holds, then it is stable under asynchronous model but unstable under synchronous model; if neither of the conditions holds, this operating point is unstable. Figure 3 clearly shows that for a given N, the stability region of synchronous model does not change with N, whereas that of the asynchronous model is enlarged as N increases. In particular, for large N with asynchronous updates, we see that the system is mostly stable for any possible configuration, thus predicting all the good performance as empirically observed.

V. Simulation Results

In this section we present simulation results using ns-2 [22] to verify that our asynchronous model is indeed more suitable for system with many flows and much closer to reality. Figure 4 shows the network configuration for our simulation. There are N symmetric sources whose propagation delays are all identical. They all share the same bottleneck link n_0–n_1 with capacity NC ($C = 0.2$ Mbps) and the buffer size $B \in \{2, 10, 100, 1000\}$ (packets), and again each packet length is set to $L = 4800$ bits. Drop-tail buffer management scheme is used everywhere. Since each sender updates its sending rate per acknowledgement(ACK), and the packets from N senders will arrive to the bottleneck link at random time instance, we expect that the probability of getting congestion signal is still given by that of $M/M/1/B$ (or something similar) and the whole system can be regarded as a rate-based congestion controller as in (1) if synchronous or in (12) if asynchronous. In this way, different choices of B will give different stability properties of each system.

Figure 5 shows the normalized average window size over N flows via ns-2 simulation. We have simulated all possible combinations of (N, B) with $N \in \{5, 50, 500\}$ and $B \in \{2, 10, 100, 1000\}$ (packets). Note that for fixed B, the long-term means of the average sending rate for different N are different. This is because the average capacity for each flow is $C_{\text{per-flow}} = C \times RTT/L + B/N$, a decreasing function of N. For easy comparison, we normalized the average window size to $C_{\text{per-flow}}$ to ensure that it always fluctuates around 1.

As Figure 5 shows, for any given B, the system tends to be more stable with less fluctuation of the average window size as N increases. Note that this is in line with the observation from our asynchronous model and the synchronous model cannot predict this behavior as the stability property does not depend on N. Similarly, for given N (say $N = 500$), we observe that larger buffer size tends to yield slightly larger fluctuations. As we can see from Figure 3, increasing the buffer size B under the same N moves the system toward

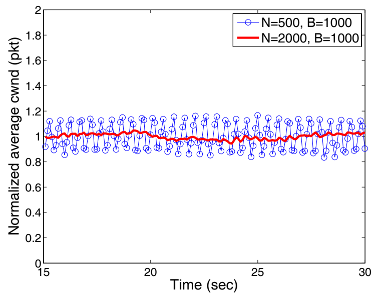
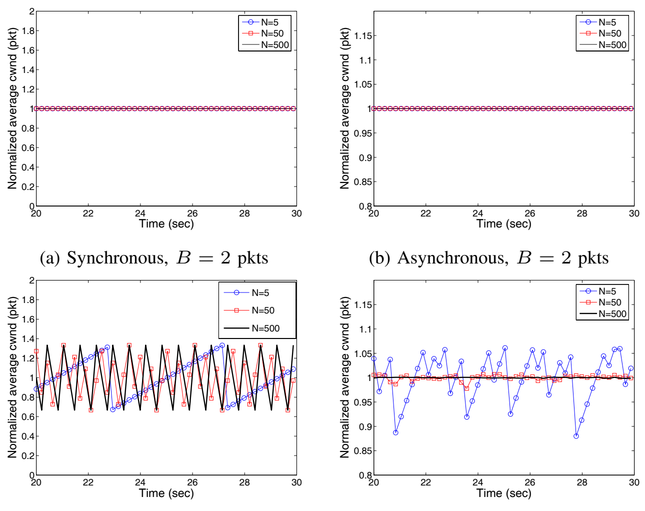


Fig. 6. Normalized average window size for $N = 500, 2000$ using ns-2 under $B = 1000$ packets. System becomes more stable as N increases.



(a) Synchronous, $B = 2$ pkts (b) Asynchronous, $B = 2$ pkts (c) Synchronous, $B = 1000$ pkts (d) Asynchronous, $B = 1000$ pkts

Fig. 7. Effect of the number of flows N and the buffer size B on system stability. We use Matlab to iterate (1) and (12).

the boundary for stability region, making the system prone to larger fluctuations. For $B = 1000$, if we increase the number of flow N even further, the system becomes more stable as shown in Figure 6. This is again well expected from Figure 3 and Theorem 2 in that the system becomes always stable for larger N under asynchronous updates, as is the case in reality.

Figure 7 shows numerical result using Matlab for the same system as in Figure 4. Specifically, we use the set of recursions in (1) and (12) with the same set of parameters as before. At each run, we randomly choose the initial window size for each flow between 0 and $C_{\text{per-flow}}$. Due to space constraint, we only report results for $B = 2, 1000$ and $N = 5, 50, 500$. (We have also tested under other scenarios and obtained the similar behavior.) When the buffer size is small ($B = 2$), both models are stable showing almost no fluctuation in the normalized average window sizes. For larger buffer size ($B = 1000$), the synchronous model becomes unstable and displays wild fluctuations regardless of N. In contrast, our asynchronous model clearly shows that even for $B = 1000$, the system gets stabilized for larger N.

VI. Conclusion

In this paper we have proposed an asynchronous model for network congestion control with many flows in which each flow sees slightly different snapshots of the network status. In contrast to traditional fluid models with synchronous updates, for which system stability remains almost independent of the number of flows N, our model predicts that the system becomes more stable as N increases and yields all the good performance, which coincides with recent empirical results. In particular, we have proven that a system becomes stable for all sufficiently large N with asynchronous updates regardless of the system configuration. Our simulation results and numerical analysis also support that our asynchronous model is simple, yet effective in predicting the realistic behavior and is a better choice for the analysis and design of the network congestion control with many flows.

References

[1] V. Misra, W. Gong, and D. Towsley, "A fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED," in *Proceedings of ACM SIGCOMM*, September 2000.
[2] F. P. Kelly, "Models for a self-managed Internet," in *Philosophical Thansactions of the Royal Society A358*, 2000, pp. 2335–2348.
[3] S. H. Low, J. Wang F. Paganini, and J. C. Doyle, "Linear stability of TCP/RED and a scalable control," in *Computer Networks*, December 2003, pp. 633–647.
[4] S. Deb, S. Shakkottai, and R. Srikant, "Stability and convergence of TCP-like congestion controllers in a Many-flow regime," in *Proceedings of IEEE INFOCOM*, April 2003.
[5] T. Alpcan and T. Basar, "Global Stability Analysis of an End-to-End Congestion Control Scheme for General Topology Networks with Delay," in *IEEE Conference on Decision and Control (CDC)*, Maui, HI, Dec. 2003.
[6] S. Shakkottai and R. Srikant, "How Good are Deterministic Fluid Models of Internet Congestion Control?," in *Proceedings of IEEE INFOCOM*, New York, NY, June 2002.
[7] G. Appenzeller, I. Keslassy, and N. McKeown, "Sizing Router Buffer," in *Proceedings of ACM SIGCOMM*, 2004.
[8] A. Dhamdhere and C. Dovrolis, "Open Issues in Router Buffer Sizing," in *ACM Computer Communication Review*, January 2006.
[9] S. Floyd and E. Kohler, "Internet Research Needs Better Models," in *First Workshop on Hot Topics in Network*, October 2003.
[10] A. Medina, M. Allman, and S. Floyd, "Measuring the Evolution of Transport Protocols in the Internet," in *ACM Computer Communication Review*, April 2005.
[11] L. Qiu, Y. Zhang, and S. Keshav, "Understanding the performance of many TCP flows," in *Comput. Networks*, 2001, pp. 277–306.
[12] V. S. Kozyakin, "Asynchronous systems: a short survey and problems," Tech. Rep., Institute for Information Transmission Problems, Russian Academy of Sciences, 2003.
[13] B. A. Huberman and N. S. Glance, "Evolutionary Games and Computer Simulations," in *Proc. Natl. Acad. Sciences*, 1993.
[14] B. Schonfisch and A. de Roos, "Synchronous and asynchronous updating in cellular automata," in *BioSystems*, 1999, vol. 51, pp. 123–143.
[15] Y. F. Su, A. Bhaya, E. Kaszkurewicz, and V. S. Kozyakin, "Further results on convergence of asynchronous linear iterations ," in *Linear Algebra Appl.*, August 1998, vol. 280.
[16] D. Chazan and W. L. Miranker, "Chaotic relaxation," in *Linear Algebra Appl.*, 1969, vol. 36, pp. 834–843.
[17] J. C. Strikwerda, "A probabilistic analysis of asynchronous iteration," in *Linear Algebra Appl.*, 2002.
[18] Y. Saad and HA van der Vorst, "Iterative solution of linear systems in the 20th century," in *J. Comput. Appl. Math*, 2000.
[19] D. B. Szyld, "The mystery of asynchronous itrations convergence when the spectral radius is one," Tech. Rep., Department of Mathematics, Temple University, 1998.
[20] M. D. Carl, *Matrix Analysis and Applied Linear Algebra*, SIAM, 2000.
[21] G. Raina and D. Wischik, "Buffer sizes for large multiplexers: TCP queueing theory and instability," in *EuroNGI*, Rome, April 2005.
[22] "The Network Simulator: ns-2," in *http://www.isi.edu/nsnam/ns/*, 2004.