Robustness issues of fluid approximations for congestion detection in best effort networks *
(extended version)

Technical report
Laboratoire Heudiasyc UMR CNRS 6599

Fabien Chatté,[†] Bertrand Ducourthial,[‡] Silviu-Iulian Niculescu [§]

Laboratoire Heudiasyc UMR CNRS 6599
Université de Technologie de Compiègne,
Centre de Recherche de Royallieu,
BP. 20529, 60205 Compiègne CEDEX, France.

March 2002

Abstract

Lots of studies have been perfomed in order to design new congestions control schemes. But lots of those works have a corrective behavior, those protocols wait that congestion appears (wait for some packets loss) for correcting it. Another strategy is to prevent congestions by reducing the sending rate, as TCP Vegas does. Fluid approximations can also help to design preventive control algorithms. The interest of fluid modeling for congestions detection lies principally on its simplicity. In this paper, we are interested on the robustness of such continuous control schemes, with respect to some of its parameters, and in particular, the *round-trip time*. The robustness analysis can give information on the congestion state of the network. It is possible to compute exact stability bounds of the RTT by using some recent results developed in delay systems theory. If the measured RTT belongs to those bounds, the network is stable (*ie.* no congestion). This exact estimator can be used to estimate the path status and to adjust the sending rate.

Keywords: network modelling, fluid modelling, continuous systems, congestion control and detection.

---

*A short vesrion has been pubilshed in Proceeding of IEEE ISCC 2002, Italy, July 2002.

†e-mail: `Fabien.Chatte@hds.utc.fr`

‡e-mail: `Bertrand.Ducourthial@hds.utc.fr`

§e-mail: `Silviu.Niculescu@hds.utc.fr`

1 Congestion control in Internet

Internet. The Internet network was born in the sixties from the connection of specific networks. Several characteristics inherits from this period, such as packet commutation, one of the first layer structured design, best effort traffic... The Internet protocol is mainly composed of the routing layer (IP [24]) and the transport layer (UDP [23] or TCP [25]). Basically, UDP is a simple unreliable transport protocol while TCP ensures a reliable data transport (*ie.* supporting loss of packets) between two distant hosts linked with an IP network.

With this design, Internet was growing until first difficulties appeared in eighties. Too much traffic had to be carrying and congestions arrived: packets were lost because routers were not able to temporarily store them. With the reliable transport protocol (TCP), these packets have to be resent, that consumes too much network resources, and decreases the throughput of network applications.

To overcome congestions, researchers refined the congestions control in TCP (see *eg.* [18, 14, 29, 2]). Thanks to these works, Internet was able to carry web traffic and to support the large related expansion in the nineties. Today, lot of works are still leading to improve the congestions control mechanisms in Internet (see *e.g.* [27, 22, 30, 8]). Indeed, more users, network applications and finally traffic needs always appear, and new network equipments are rapidly full. Hence, it is of economical interest to better use current equipments. Moreover new applications, such as multimedia (sound, video) diffusions, lead to new problems. Indeed, TCP is not really convenient for such flows because resending lost packets to complete a video sequence is less interesting than preserving the application throughput (to avoid freezing the images). In the other hand, UDP does not include congestion control. This could lead to large congestion, preventing good multimedia receptions, even if equipments could support the overall traffic (if an adequate traffic control was available).

TCP. As already mentioned, TCP is a reliable transport protocol that includes congestion control. Before detailing related works and our contribution in this field, we first recall basic behaviors of this protocol [25, 28].

TCP manages a full-duplex end-to-end connection between two symmetric hosts. On this path, data transfer is flow-oriented. The data flow that must be sent is split by the TCP sender, to be send as segments. Such segments could be fragmented into IP datagrams and could follow different routes until to reach the destination host. At the arrival, the TCP receiver merges the segments — thanks to some sequence numbers — to give a flow to the destination application. As such, TCP determines itself the size and the sending throughput of the segments. The TCP sender waits for an *acknowledgment* (ack) from the receiver. The delay between the sending of a segment and the receipt of its ack is called *round-trip-time* (RTT). If a segment is not acknowledged before the expiration of a *retransmit time-out* (RTO), it is resent; loss of segments is then supported. Duplicate segments are detected with sequence numbers, and ignored. Garbling is preventing by using checksum. Hence, TCP offers a reliable communication. Finally, the last important characteristic of TCP is its ability to adjust the sending rate. For the purpose of this paper, we are now focusing on it.

Basically, the congestion control relies on a *congestion window* (CW) that indicates to the sender how many segments can be sent without receiving the corresponding acks (*ie.* how much the sender can be optimistic). In case of congestion, the CW decreases. If no congestion is detected, the CW size increases until reaching the receive window size (temporary storage space on the TCP receiver) to prevent congestion on the destination host itself. Versions of TCP essentially differ on the way the CW size (expressed in segments in the following) varies.

To increase the congestion window, *TCP Tahoe* [14] and *TCP Reno* [28] implement the *slow start* algorithm: starting with a size of one, the CW is incremented (of one segment) at each arrival of a new ack. Hence, during a period of one RTT, the window size is doubled, which leads to an exponential growing. The slow start stops if a packet loss is detected, or if it reaches the half of the size it had when the last loss occured. At this step, the sending throughput is near the rate that could lead to new congestion. The CW size is still growing until a loss is detected, but more slowly, using the *congestion avoidance* algorithm: the CW size is approximatively incremented of one segment per RTT.

A packet loss indicates congestion between the sender and the receiver. It is detected when the RTO expires. In that case, the CW size is reduced to one and slow start is running. But the TCP receiver acknowledges always the last packet before a missing one, even if further packets are received. Hence several acks can be received for the same packet; this could be caused by a packet loss (*ie.* congestion). When three duplicate acks arrives, TCP Reno halves the CW size whereas TCP Tahoe reduces the CW to one and runs slow start. More, TCP Reno runs the *fast recovery* algorithm [2]: the lost segment is immediately resent, and the CW size is temporarily enlarges by three (for the three duplicate acks). When the ack of the lost segment is received, the CW size is decreased of three, and congestion avoidance is called. TCP Reno is currently the most widely used. Variations of this congestion control schemes are presented in Section 2.

Fluid approximation. As we recalled above, TCP adjusts its sending rate to well utilize the network resources without causing congestions. Indeed, since lost packets are resent, congestions decrease the applicative througput between the two hosts of the end-to-end connection. Informations on the network state are given by the acknowledgments that allow to compute the RTT (and to adjust the RTO), to detect segments loss, and finally to manage the CW size. Such a connection can be modeled as an input-output system with feedback, where the input is the sending rate and the output is the incoming rate of the data flow given by the TCP receiver to the destination application. Feedback is given by the acks from the TCP receiver to the TCP sender. Note that since there could have congestions on the network, the delay needed by the feedback to reach the source is variable. Hence the TCP connection is modeled by an input-output system with feedback and uncertain delay.

The increasing interest [3, 17, 12, 13] on fluid approximation models for such systems lies principally on its simplicity. More, switching networks problems — such as congestion — can be translated as feedback-control problems. In fact, the control problem here obeys the principle that the network is a *black box* unable to give any explicit information to the data sources; there is no way to construct any *closed-loop interconnection* scheme. Thus, we may interpret a packet loss and the RTT duration as *implicit feedback* information on congestions, using the feedback-control theory language (see, e.g. [15, 16]), and the congestion problem can be seen, in some sense, as a *closed-loop stability* problem.

More precisely, the fluid approximation model will include at least one integrator (the simplest model to describe a congested node) and delays (*e.g.* RTT). As indicated above, the congestion problem will be interpreted as the analysis of the asymptotic stability of the corresponding fluid model [16, 15, 12, 13].

Our contribution. This paper deals with applications to best effort traffic of some fluid approximations models encountered in the literature, and to their analysis in terms of RTT. The developements in the area of time-delay systems control [19] allow to compute optimal and suboptimal bounds guaranteeing stability of the corresponding continuous-time schemes, which define some degree of robustness in terms of delays. The idea behind is to take advantage on such robustness issues of the fluid approximation in order to improve the behavior of the actual (discrete-time) network: better congestion detection, better estimation of the RTO, better sending rate control, *etc*.

The paper is organized as follows. Some of the existing methodologies for controlling congestion in transport protocols are briefly presented in Section 2. Section 3 introduces fluid models already mentioned in the litterature for modeling data transferts in networks [12, 13]. Section 4 focus on the robustness issues of such models, using results of [19] (note that mathematical developments are not given in this short version, see appendix and [11] for more details). Section 5 interprets these results in order to derive a new congestion detection scheme for TCP-like connections. Finally, some concluding remarks end the paper in Section 6.

2 Related works on congestion control in transport protocols

In order to improve the congestion control in TCP, several options and variants have been developed around TCP. Moreover some *TCP friendly protocols* [22] have also been studied for carrying multimedia flows. We summarize here main ideas proposed for controlling congestions in Internet.

2.1 TCP variants

TCP options. The *sack* option [10] has been designed to optimize retransmissions in case of lost segments. With the sack option, the acks explicitly specify which segments have been correctly received. Hence the TCP sender can immediately resend lost segments. If the sack option is not supported, one can use the *new Reno* option [9]. This option modifies the fast recovery algorithm to take into account several segment loss during the same RTT, and to avoid unnecessary sending rate decreasing. The interpretation of the new acks by the source allows to stay in fast recovery instead of prematurely entering in congestion avoidance. Indeed, if the TCP sender enters in congestion avoidance, its CW will be reduced to one at the detection (expiration of the RTO) of the second lost segment, and the slow start algorithm would be running.

TCP Vegas. Beside options, another TCP version has been studied. TCP Vegas [4] implements a preventive behavior instead of a corrective one like others TCP versions. Instead of waiting for a data loss to adjust the sending rate, TCP Vegas estimates the available bandwidth by using the expected sending rate, and the actual sending rate. The expected rate is obtained by dividing the CW size by the smallest RTT. The actual rate is computed by dividing the number of bytes sent during the last RTT by the actual RTT. Two thresholds α and β are used to manage the CW size: the CW increases if the actual rate is smaller than α; it keeps the same size if the rate is in the interval $[\alpha, \beta]$, and it decreases if the rate is larger than β.

2.2 TCP friendly protocols

TCP Friendly Rate Control. The TFRC protocol [22] controls the rate of the source by using an equation based on the RTT, the RTO, the reception window size, and the lost segments rate. This analyse is based on stochastic and Markovian modelings of TCP [8, 22].

The loss rate is a good estimator of the congestion state of the network. It is computed with the loss events and intervals. A loss event represents a contiguous set of lost data, and a loss interval represents the interval between two loss events. The loss rate is the inverse of the weighted mean of the last eight loss intervals duration. The RTT is estimated with a weighted mean of its last computed value and the last measured value (by using timestamps included in segments). The aim of such estimations is to smooth the sending rate by avoiding important rate modifications when the loss are not critic.

TFRC uses a modified slow start algorithm that avoids to saturate the path: the new sending rate is equal to $2 \times \min(\text{old sending rate, old receiving rate})$. The receiving rate is computed by the receiver and transmited to the source in acks. At the first detected loss, slow start is stopped, and the sending rate is adjusted by using the TFRC control rate equation.

PAth STatus-based RAte control. In the PASTRA protocol, the sending rate is controlled on the base of the *status* of the path from the sender to the receiver [30]. The path status is evaluated on the base of the *relative one way trip time* (ROTT), computed by the TCP receiver. Indeed, observations of the ROTT have shown that loss are correlated to huge and short variations of the ROTT, called by the authors *spikes trains*.

PASTRA qualifies the path status by four *states*. In the first one, the receiver computes some necessary parameters to estimate the ROTT [31]. In the second state, the path is considered stable (the rate of packets in transit during spikes trains is less than 30%). The path is in the unstable state when there is too many packets in transit during spikes trains, or when PASTRA is unable to compute the parameters (used to estimate the ROTT). Finally, the last state is used when the route is changing (in that case, the receiver must compute the new reference ROTT).

These path states are used to determine some transmitting *modes*, which in turns determines the transmission rate. In the first mode, the PASTRA source estimates what rate would have a TCP connection in these conditions (*e.g.* the route is initialized or is changing). If PASTRA is unable to estimate this rate, the source enters in the second mode, where data are transmitted at a low rate. The last mode is used when the path is stable. In this (normal) case, the data are transmitted at a rate B such that $0.8 \times$ estimate TCP rate $\leq B \leq$ estimate TCP rate.

Rate Adaptative Protocol (RAP) [27]. Like TCP does, this protocol controls the sending rate on the base of packets loss. However, RAP is not a window-based protocol, but a rate-based one: to ajust the rate of the source, RAP adjusts the *inter-packets gap* (IPG) instead of the CW size. Like TCP (in congestion avoidance phase), if there is no congestion, the transmission rate periodically increases of one segment per RTT. If a congestion is detected the transmission rate is immediately decreased (the rate is halved). There is two way to detect a congestion: the RTO expires (the RAP RTO computation is different than the TCP one), or a hole in the acknowledged sequence numbers is detected. Indeed, all received segments are acknowledged by its sequence number; if the last ack acknowledges a segment sent three packets later than a non acknowledged one, the non acknowledged segment will be considered as lost. The sending rate is adjusted just one time per RTT. At each update of the rate, RAP can observe the network reactions to the last modification of the rate.

3 Fluid models for congestion control

Fluid approximation for packet-switched networks. The TCP protocol and related works on congestion detection described previously are intrinsically discrete (*i.e.* packet-based communication, states-based algorithms and so one). However some works showed that fluid modeling can be used to approximate such packet-switched network behavior [3, 26], even if some precautions must be taken [5]. Intuitively, the denser the traffic is, the smaller the inter-packets gap is, and then the more the packets traffic should be seen as a fluid one. Moreover, the smaller the packets are, the more the packet-based sampling of the flow should be negligible.

In the sequel, we shall focus on some continuous-time deterministic models derived by using such a fluid approximation method. Thanks to previous results as well as those which we develop in the following section, we believe that congestions control in TCP-like protocol can take advantages of such modeling. Indeed, robustness issues of these fluid modeling could allow some improvements in the design of actual congestion control algorithms. For instance, the computation of the RTO could be more pertinent, and the sending rate could be more precisely adjusted. Note that similar applications of control theory have already been mentionned in the litterature (see *e.g.* [17]). However, by using new developments on time-delay systems [19], we take into account the uncertain delay necessary for an ack to reach the source; the model is then more accurate, and the analysis can be more easily performed.

Fluid model. In [12, 13], two continuous-time deterministic models have been studied for data networks. New control theory results about such systems allow to envisage applications for controlling congestions in Internet. As we will see, robustness properties of these models lead to new congestions detection and sending rate control schemes, in order to keep a stable behavior of the TCP connection, and hence avoid congestions.

For the sake of simplicity, we focus on a simple continuous-time deterministic model derived from [13], modeling a single connection with a single congestion. Note that further mathematical developments could extend our methodology for more complicated cases. This model is then composed of a connection between a source (S) controlled by an access regulator and a destination node (D) through a congested node (C) served with a transmission constant capacity μ (see Figure 1). This can be described by a second-order delay-differential equation of the form:

$$\begin{cases} \dot{q}(t) = x(t-\tau_1) - \mu \\ \dot{x}(t) = -a\left(q(t-\tau_2) - q_O\right) - b\left(q(t-\tau_2-r) - q_O\right), \end{cases} \tag{1}$$

where q represents the queue size, x the current input rate, q_O the optimal queue size (target value), a and b the gains control parameters, and r an *additional delay* that we will precise later.

Using the new variable $y(t) = q(t) - q_O$, Equation 1 leads to the following second-order delay equation, where $\tau = \tau_1 + \tau_2$:

$$\ddot{y}(t) + ay(t-\tau) + by(t-\tau-r) = 0 \tag{2}$$

This equation will allow to determine pertinent relations between parameters a, b, r and τ in order to control the sending rate $x(t)$ by Equation 1. The total delay $\tau = \tau_1 + \tau_2$ represents

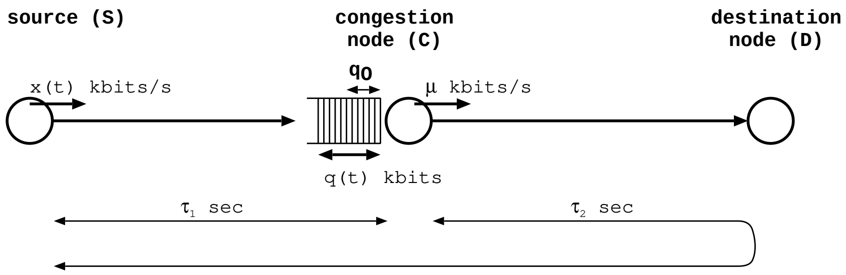

Figure 1: Modeling of a connection with one congestion.

the RTT (see Figure 1), measured using acks. The size of the queue has only been used for modeling purposes and we shall not deal with it. The parameters a, b and r are coefficients used to adjust the sending rate $x(t)$.

The additional delay r is non null in order to have one more degree of freedom. For one sending rate target, a set of tuples (a, b, r) is convenient. This is important because, as we will see, the range of RTT values for which the system is stable is also determined with (a, b, r). Hence, values of (a, b, r) will be chosen in order to ensure that the system will keep or reach a stable state, even if small variations of the RTT happened.

More complex model. A similar model with a further degree of freedom can be derived from [12, 13]:

$$\begin{cases} \dot{q}(t) = x(t-\tau_1) - \mu \\ \dot{x}(t) = -a\left(q(t-\tau_2) - q_O\right) - b\left(q(t-\tau_2-r) - q_0\right) - c\left(q(t-\tau_2-\frac{r}{2}) - q_O\right) \end{cases} \tag{3}$$

where the new parameter c has to be computed. Using the same procedure as above, the corresponding second-order delay-differential equation becomes:

$$\ddot{y}(t) + ay(t-\tau) + by(t-\tau-r) + cy\left(t-\tau-\frac{r}{2}\right) = 0 \tag{4}$$

4 Robustness analysis

Aim. Our interest is on the description of some of the delay limitations (upper/lower bounds, stability/instability characterization) of the closed-loop System 2 (and 4), that is the *robustness* of these schemes in terms of RTT (τ) and additional delay (r).

For this delay robustness analysis, both delays (τ, r) are considered as free parameters of the system, while the gains (a, b) are fixed. The problem considered here consists in analyzing the stability of Equation 2 in the *delay-parameter space $O\tau r$*, that is to find some sets of intervals of the form $[\underline{r}_i, \overline{r}_i]$ and $[\underline{\tau}_i, \overline{\tau}_i]$ function of the parameters (a, b) such that the second-order System 2 is asymptotically stable (note however that in the following and for simplicity, we will consider a single interval for τ: $[\underline{\tau}; \overline{\tau}]$). In fact, these upper and lower bounds will define some delay-insensitivity measures, or equivalently, the robustness of the control algorithm in terms of delays.

It is important to note that the System 2 free of delays $r = \tau = 0$ is an *oscilator* (characteristic equation has roots on the imaginary axis), and thus is unstable. As seen in [1] (simple second-order oscillator) or in [20] (more general problems), the use of a small delay r in the model (with $\tau = 0$) will induce the *stability* of the corresponding continuous model.

4.1 Robustness analysis of the first fluid approximation model.

Some tedious computations using the approach proposed in [19] (see also [6] for some ideas) lead to the following results. Mathematical developments can be found in [11] and in appendix.

Proposition 1 (Stability) *The System 2 with $b < 0$ and $a > |b|$ is stable for all delays r and τ satisfying the following constraints:*

$$\underline{r}_i(a,b) = \frac{2i\pi}{\sqrt{a-|b|}} < r < \frac{(2i+1)\pi}{\sqrt{a+|b|}} = \overline{r}_i(a,b), \quad i = 0,1,\ldots \tag{5}$$

$$\begin{cases} \underline{\tau} = 0 \le \tau < \tau(a,b,r) = \overline{\tau}(a,b,r) = \min_{\omega_s}\left\{\frac{1}{\omega_s}\cdot arctan\left(\frac{-b\ sin(\omega_s\ r)}{a+b\ cos(\omega_s\ r)}\right)\right\} \\ \textit{where } \omega_s \textit{ belongs to the set of positive solutions of the equation} \\ \omega^4 = a^2 + b^2 + 2ab\ cos(\omega r) \end{cases} \tag{6}$$

This proposition defines sets of intervals for τ and r that ensure the stability of System 2. Following propositions say that others values of these parameters lead to instability. The range of i guaranteeing the inequalities in Equation 5 and Equation 6 can be computed as follows:

$$i \in \mathbb{N} \quad \text{and} \quad 0 \le i < \frac{1}{2}\cdot\frac{1}{\sqrt{\frac{a+|b|}{a-|b|}}-1} \tag{7}$$

Proposition 2 (Instability due to τ) *Assume that the delay r of Equation 5 and the value $\tilde{\omega}_s$ of Equation 6 satisfy $\tilde{\omega}_s^3 > \frac{1}{2}a\cdot|b|\cdot r\cdot\sin(\tilde{\omega}_s\ r)$. Then we have:*

i) if $\tau = \overline{\tau}(a,b,r) + \varepsilon$, with $\varepsilon > 0$ sufficiently small, the System 2 is unstable.

ii) if the Equation 6 has only one positive solution, then there does not exist any positive value of $\tau > \underline{\tau}(a,b,r)$ such that the System 2 is asymptotically stable.

Proposition 3 (Instability due to r) *If for a pair (a,b) with $b < 0$ and $a > |b|$, the delay r does not satisfy the constraints of Equation 5, then the closed-loop system is unstable for any delay τ satisfying Equation 6 and the conditions* i) *and* ii) *of Proposition 2.*

4.2 Robustness analysis of a more complicated fluid approximation model

As precised in the previous sections, a second fluid approximation model using further parameters was proposed in [13], that means further degree-of-freedom, in view of improving the performances.

Stability analysis. The first step is to find conditions such that 4 is stable for sufficiently small delays τ and r:

Lemma 1 *Assume that $a+b+c > 0$ and $2b+c < 0$. Then there exists sufficiently small positive values $\varepsilon_1, \varepsilon_2 > 0$, such that Equation 4 with $\tau = \varepsilon_1$ and $r = \varepsilon_2$ is asymptotically stable. Note that the system has an oscillatory behavior if $r = \tau = 0$.*

Note that the conditions in 1 naturally extend the constraints $a > 0$, $b < 0$ and $a > |b|$ considered in the previous section. The proof is based on [19].

The stability result from the previous section rewrites as (for the sake of simplicity we have considered only the first delay interval in terms of r):

Proposition 4 (Delay bounds) *The system 4 satisfying the constraints 1 is asymptotically stable for all delays r and τ:*

$$\begin{cases} 0 < r < r_1(a,b,c) \\ 0 < \tau < \tau(r) = \mathbf{min}_{\omega_s} \left\{ \dfrac{1}{\omega_s} \cdot \mathbf{arctan} \dfrac{-bsin(\omega_s r) - csin\left(\frac{\omega_s r}{2}\right)}{a + bcos(\omega_s r) + ccos\left(\frac{\omega_s r}{2}\right)} \right\}, \end{cases} \tag{8}$$

where ω_s belongs to the set of positive solutions of the equation:

$$\omega^4 = a^2 + b^2 + c^2 + 2\left(abcos(\omega r) + (a+b)ccos\left(\frac{\omega r}{2}\right)\right). \tag{9}$$

Furthermore, assume that the chosen delay r and the solution $\tilde{\omega}_s$ defining the corresponding upper bound $\tau(r)$ in 8 satisfy the condition:

$$\tilde{\omega}_s^3 + \frac{1}{4}c(a+b)r\ sin\frac{\tilde{\omega}_s r}{2} + \frac{1}{2}abr\ sin(\omega_s r) > 0. \tag{10}$$

Then:

i) if $\tau = \tau(r) + \varepsilon$, with $\varepsilon > 0$ sufficiently small, the system 4 is unstable.

ii) if the equation 9 has only one positive solution, then there does not exist any $\tau > \tau(r)$ such that the system 4 is asymptotically stable.

We still need to compute the bound $r_1(a,b,c)$.

Introduce now the following matrix pencils $\Lambda_1 \in \mathbf{C}^{4\times 4}$ and $\Lambda_2 \in \mathbf{C}^{2\times 2}$:

$$\Lambda_1(z) = z \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & b \end{bmatrix} + \begin{bmatrix} 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \\ -b & -c & 0 & c \end{bmatrix} \tag{11}$$

$$\Lambda_2(z) = z \begin{bmatrix} 1 & 0 \\ 0 & b \end{bmatrix} + \begin{bmatrix} 0 & -1 \\ a & c \end{bmatrix}, \tag{12}$$

where $n = 2$ and $A_0 = \begin{bmatrix} 0 & 1 \\ -a & 0 \end{bmatrix}$, $A_1 = \begin{bmatrix} 0 & 0 \\ -c & 0 \end{bmatrix}$, and $A_2 = \begin{bmatrix} 0 & 0 \\ -b & 0 \end{bmatrix}$, respectively.

Consider the original system 4, and assume that $\tau = 0$. Denote $\sigma(\Lambda)$ the set of general eigenvalues of the matrix pencil Λ. The same notation will be also used for the eigenvalues of some real or complex matrix.

Let us assume that $r = \bar{r}$ be a positive value. In the sequel, one needs to introduce the following sets:

$$\begin{aligned}
\Lambda_{\bar{r},+} \;=\; & \Big\{ (r_{h_i}, \alpha_h) : \; r_{h_i} = \frac{\alpha_h}{\omega_{hi}} > \bar{r} : \\
& e^{-j\alpha_h} \in \sigma(\Lambda_1) - \sigma(\Lambda_2), \\
& j\omega_{hi} \in \sigma\left(\sum_{k=0}^{n} A_k e^{-j\frac{\alpha_h k}{n}} \right) - \{0\}, \\
& 1 \le h \le 2n, \; 1 \le i \le n \Big\}, \qquad (13) \\
\Lambda_{\bar{r},-} \;=\; & \Big\{ (r_{h_i}, \alpha_h) : \; r_{h_i} = \frac{\alpha_h}{\omega_{hi}} < \bar{r} : \\
& e^{-j\alpha_h} \in \sigma(\Lambda_1) - \sigma(\Lambda_2), \\
& j\omega_{hi} \in \sigma\left(\sum_{k=0}^{n} A_k e^{-j\frac{\alpha_h k}{n}} \right) - \{0\}, \\
& 1 \le h \le 2n, \; 1 \le i \le n \Big\}. \qquad (14)
\end{aligned}$$

As seen in [19] (general case), the set $\Lambda_{\bar{r},+}$ allows to characterize the delay values $\bar{r}_{h_i} > \bar{r}$ for which some roots of the associated characteristic equation cross the imaginary axis. The same conclusion holds for the set $\Lambda_{\bar{r},-}$, but with the difference that $\bar{r}_{h_i} < \bar{r}$.

Using the methodology in [19], we have:

Proposition 5 (First switch) *The system 4 with $\tau = 0$ satisfying the condition 1 is asymptotically stable for all delay values r:*

$$0 < r < r_1(a, b, c),$$

where

$$r_1(a, b, c) = \mathbf{inf}\left\{ \gamma : \quad (\gamma, \alpha) \in \Lambda_{0,+} \right\}. \qquad (15)$$

Furthermore, if $r = 0$ or $r = r_1(a, b, c)$, the corresponding associated characteristic equation has at least one pair of roots on the imaginary axis.

4.3 Interpretations

The interpretations presented in the previous case hold also here, although it seems that we have more degree-of-freedom in choosing the *gains*: a_0, a_1 and a_2.

In fact, improving the *performances* of the algorithm leads to *important* values of the corresponding gains. Note however some improvement of the system's behavior for *small delays* τ, r, but a *degradation* of the *performances* when the delays are increased.

Thus, a natural consequence of the results above is the following:

Corollary 1 (Stability robustness) *The control algorithm defined by the 3-tuple (a, b, c) satisfying the constraints 8 with the control-time interval $r \in (0, r_1(a, b, c))$ guarantees the stability of the fluid model if the round-trip delay τ takes any positive value in the interval $[0, \tau(r))$, where $\tau(r)$ is defined in 10.*

Remark 1 (Instability persistence) *When appro-aching the* congestion phenomenon *in actual high-speed networks, the* round-trip times *becomes larger and larger, and it is possible to belong not to the first delay-interval guaranteeing stability, but to the next ones in terms of τ for the same control-time interval r and the same gains a, b, c, till the bound $\bar{\tau}$ is not reached.*

Note that if $\tau > \bar{\tau}$, packages will be lost in the data transfer from some source to some destination using the control algorithm above for one of the congestion nodes in the path, etc.

Remark 2 *The maximal delay bound $\tau(r)$ guaranteeing stability can be* optimized *function of $r \in (0, r_1(a_0, \ldots, a_n))$, and we shall have a* 'max-min' *problem.*

5 Applications to the congestion detection

Interpretations. In previous results, variables a, b, and r are parameters while τ is the RTT measured with acks representing the implicit feedback about the network.

Parameters a, b and r can be used to control the sending rate $x(t)$ of the source, thanks to a rate control function f that can be derived from Equation 1: $x(t) = f(a, b, r)$. Such control function f is not detailed here. However, one can say that to increase the sending rate, the parameter a must be decreased, and the parameter $|b|$ must be increased. Inversely to decrease the sending rate, a must be increased and $|b|$ must be decreased.

On the other hand, Proposition 1 (and 4) says that parameters a, b and r determine the *exact* bounds of stability for τ. Hence, the variable τ can be used to characterize the stability of the network. That is, depending on parameters that control the source, the measured RTT can indicate if the TCP connection is in a stable state or not. This can be done by checking if the measured RTT belongs to some pre-computed intervals $[\underline{\tau}_i; \overline{\tau}_i]$.

It is important to note that Condition *ii)* of Proposition 2 represents a very strong result. Indeed, the fluid model approximation is unstable for all RTTs if the additional delay belongs to some pre-defined intervals. Proposition 1 says that the choice of the additional delay r becomes an important factor to induce stability or instability in the associated fluid model for small RTT. On the other hand, as indicated by Proposition 1, decreasing the duration of the additional delay is not always the best solution to improve the performances of the algorithm in terms of robustness. A different argument for such a conclusion was presented in [13].

Note that the analysis proposed in [13] (which extends the technique proposed in [12] if $r = \tau$) does not allow to conclude on such type of property. The advantage of this analysis is to give explicitly upper and lower stability/instability bounds in terms of additional delay and *round-trip times*. Furthermore, for a given triplet (a, b, r), we are able to compute the upper limit of the *round-trip time* guaranteeing the stability of the corresponding scheme, that is the behavior for small *round-trip times*.

Congestion detection. The control rate function f (derived from Equation 2) as well as Propositions 1, 2 and 3 could lead to a new transport protocol. This protocol could be a rate-based one (such as RAP: rate variations apply on the inter-packets gaps instead of on a congestion window). Its behavior should be founded on modeling equations analysis, as for TFRC. Contrary to other RTT-based protocols, a more accurate congestion detection criteria could be obtained. Here we give the main design ideas of such a source rate control algorithm (among others). Note that the following algorithm can already be used to detect congestion.

Algorithm 1: Congestion dectection algoritm

Initialization:
  $a = a_0$
  $b = b_0$
  $r = r_0$
  ▷ *The values a_0, b_0, and r_0 have been chosen in order to have a very large sending rate.*
  Compute the intervals $[\underline{\tau}_i; \overline{\tau}_i]$ by using Equation 6
  τ = minimum value of the RTT

Receiving an ack:
  $\tau =$ mesured RTT
  if $\tau \notin [\underline{\tau}_i, \overline{\tau}_i]$ **then**
    ▷ *In this case the network is unstable, the rate has to be adjusted.*
    Increase a, decrease $|b|$
  else
    ▷ *In this case, the network is stable. May be the source could increase its sending rate.*
    Decrease a, increase $|b|$
  end if
  Choose r in the interval computed by using Equation 5
  ▷ *r is chosen in order to optimize the sending rate, the RTT bounds, or the robustness of the end-to-end connection*
  ▷ *again further small variations of traffic.*
  Compute the intervals $[\underline{\tau}_i; \overline{\tau}_i]$ by using Equation 6
  ▷ *$[\underline{\tau}_i; \overline{\tau}_i]$ has to be re-computed with the new values of a, b and r.*
  $x(t) = f(a, b, r)$ ▷ *Computing the new sending rate.*

Sending a packet:
  $IPG = \frac{\text{packet size}}{x(t)}$ ▷ *Compute the Inter-Packet Gap delay.*
  send the packet respecting IPG (the inter-packet gap).

6 Concluding remarks

Transport protocols are principally based on empirical rules to detect congestion. Every presented protocol (unless TCP Vegas) uses the packet loss to detect congestion. This detection method is not preventive and it induces the creation of congestions instead of preventing them. One method to design a preventive protocol is to use the fluid modeling.

Using the fluid modeling has the advantage to permit a simplified study of the network behavior. Assuming a fluid approximation, more accurate criterias can be developed to prevent congestion. Indeed this modeling allows to apply some automatic theories about closed-loop system. Those results can give an exact and formal method to detect congestion instead of empirical ones. By studying robustness of models derived from [13], we designed a scheme for congestions detection. This model allows to derive some equations which can be used to analyze the network behavior. This analyze leads to determine pertinent relations between the RTT and the stability (the congestion state) of the network. The robustness analysis of this scheme gives exact intervals for the RTT that ensures the network has no congestion. Those exact rules can be applied to the congestion detection in a discrete packet switch protocol.

However, using a continuous framework for the study and the control of a packet-switched protocol leads to two main problems: the *accuracy* of the fluid modelling (TCP modeling is not an easy task [22]) and the *equivalence* of discrete and fluid results (depending on network and protocol parameters). Our previous studies (on the equivalence) [5] have shown that there exist some limitations on the use of fluid modeling of a discrete protocol. Indeed, both models are equivalent only under some conditions.

In future works, we plan to design the way to manage a, b, and how to choose r in function of a, b, and τ (RTT measured). We will study the function $f(a, b, r)$ used for computing the sending rate $x(t)$. And finally we will work on the design of a discrete protocol (decision function, increasing/decreasing rate algorithm, and decision frequency) in order to simulate its behavior with ns simulator [21].

A Mathematical developement

This appendix deals with mathematical development necessary for Proposition1. Assume first that $\tau = 0$, but $r \neq 0$ in Equation 2, that is we shall consider the behavior of the system on the delay axis Or, if the control-time interval is seen as a free parameter. The corresponding characteristic equation becomes transcendental:

$$s^2 + a + be^{-sr} = 0 \tag{16}$$

and is similar to the closed-loop system of an oscillator with a delayed output feedback of gain $-b > 0$ [20, 1]. Then, one can prove the following result:

Proposition 6 *The System 2 with $\tau = 0$, $b < 0$, $a > |b|$ is asymptotically stable for all the triples (a, b, r) satisfying simultaneously:*

$$\underline{r}_i(a,b) = \frac{2i\pi}{\sqrt{a-|b|}} < r < \frac{(2i+1)\pi}{\sqrt{a+|b|}} = \overline{r}_i(a,b) \tag{17}$$

for $i = 0, 1, \ldots$. If $r = \underline{r}_i(a,b)$ or $r = \overline{r}_i(a,b)$, the associated characteristic equation in closed-loop has at least one eigenvalue on the imaginary axis. Furthermore, for each pair (a,b) there exists a value $r^*(a,b)$, such that for any $r > r^*(a,b)$ the considered system is unstable.

Using a continuity argument in terms of the solutions of the characteristic equation of Equation 2 with respect to the delay τ, the result above proves that the control time-interval could have a stabilizing or destabilizing effect for small *round trip-times* τ. Indeed, for $\tau = 0$, Proposition 6 shows a sequence of stability and instability control-time intervals. Further arguments are presented below.

The next step consists in analyzing the behavior of the characteristic equation of Equation 2 in the hypothesis that r satisfies the constraints given in Proposition 6, and with τ as a free parameter, that is r and τ are decoupled variables. Since the stability is guaranteed [7] for $\tau = 0$ and, by continuity, for sufficiently small positive values $\tau = \varepsilon > 0$, we are interested to compute the optimal $\tau_i = \tau_i(a,b,r)$ for a control-time interval r given in $(\underline{r}_i(a,b), \overline{r}_i(a,b))$, and guaranteeing the above property.

For such a r, the characteristic equation of Equation 2 can be rewritten as:

$$s^2 + (a + be^{-sr})e^{-s\tau} = 0 \tag{18}$$

Using the continuity principle argument with respect to the delay τ, it follows that one only needs to characterize the solutions of the characteristic equation with respect to the imaginary axis. Thus, one can have two different situations: no solutions and at least one solution on the imaginary axis, respectively. Indeed, if there does not exist any solution of Equation 18 on $j\mathbb{R}$, then the system will be stable for any delay value (delay-independent stability, see, for instance, [19]). If there exists at least one root, then one can compute the bound on the delay τ^* for which the system will loss its stability, and such that for any $\tau \in [0, \tau^*)$, the system will be asymptotically stable (for some r appropriately chosen).

Using the stability argument presented above, it follows that the characteristic Equation 18 on the imaginary axis becomes:

$$\omega^2 = (a + be^{-j\omega r})e^{-j\omega\tau} \tag{19}$$

If one takes the modulus, we shall have:

$$\omega^4 = (a + bcos(\omega r))^2 + b^2 sin(\omega r)^2 \tag{20}$$

which is equivalent to Equation 6. Since that equation has always at least one positive solution ω_s, it follows that there exists a positive maximal delay value τ^*, such that the system will be stable for all $\tau \in [0, \tau^*)$ (see arguments in [19]). If, for example, one takes the imaginary part of Equation 19, it is easy to see that taking for τ the value:

$$\tau^* = \min_{\omega_s} \frac{1}{\omega_s} arctan \left(\frac{-bsin(\omega_s r)}{a + bcos(\omega_s\, r)} \right) > 0 \tag{21}$$

the Equation 19 is satisfied by the corresponding 'frequency' ω_s. The next step is to prove that the bound in Equation 5 is the optimal one.

Assume that there exists a smaller $0 < \tilde{\tau} < \tau^*$ such that the system is unstable. Then, based on the continuity argument (see also Proposition 3.1 in [19], p. 89), it follows that there exists some delay $\bar{\tau}$ such that $0 < \bar{\tau} < \tilde{\tau}$, and if $\tau = \bar{\tau}$, the characteristic equation has at least one solution $\bar{\omega}$ on the imaginary axis, and no roots with strictly positive real parts. This solution should satisfy the equation:

$$\bar{\omega}^2 = (a + be^{-j\bar{\omega}r})e^{-j\bar{\omega}\bar{\tau}} \tag{22}$$

In conclusion, $\bar{\omega}$ should satisfy (modulus, imaginary part constraints):

$$\begin{cases} \bar{\omega}^4 = (a + bcos(\bar{\omega}r))^2 + b^2 sin(\bar{\omega}r)^2 \\ (a + bcos(\bar{\omega}r))sin(\bar{\omega}\bar{\tau}) = -bsin(\bar{\omega}r)cos(\bar{\omega}\bar{\tau}) \end{cases} \tag{23}$$

which leads to the equality:

$$\bar{\tau} = \frac{1}{\bar{\omega}} arctan \left(\frac{-bsin(\bar{\omega}r)}{a + bcos(\bar{\omega}r)} \right) \tag{24}$$

From the definition of τ^* in Equation 6, it follows that $\bar{\tau} \geq \tau^*$, which contradicts $\bar{\tau} < \tilde{\tau} < \tau^*$. In conclusion, the bound on τ in Equation 6 is the optimal one.

The second step is to analyze the sign of the real part of the derivative of $\frac{ds}{d\tau}$ in $s = j\omega_s$, with $\omega_s > 0$. Some simple but tedious computations lead to the following:

$$Re \left[\left(\frac{ds}{d\tau} \right) \right]_{s = j\omega_s} = \frac{2\omega_s^5}{2\omega_s^3 - a|b|sin(\omega_s\, r)} \tag{25}$$

If the constraint $\tilde{\omega}_s^3 > \frac{1}{2} a \cdot |b| \cdot r \cdot \sin(\tilde{\omega}_s\, r)$ is satisfied it follows that the right-hand term is positive, that means the instability of the considered system for some delay $\tau = \tau_i(a, b, r) + \varepsilon$, with $\varepsilon > 0$ sufficiently small. If the Equation 6 has only one positive solution $\omega_s > 0$, then, due to Equation 25, it follows that at each crossing, the number of roots with positive real parts is strictly positive. Thus, the proof is complete.

References

[1] C. Abdallah, P. Dorato, Benitez-Read, and R. Byrne. Delayed positive feedback can stabilize oscillatory systems. In *Proc. American Contr. Conf.*, pages 3106–3107, 1993.

[2] M. Alman, V. Paxson, and W. Stevens. TCP congestion control. Request For Comment 2581, RFC editor, http://www.rfc-editor.org/, March 1999.

[3] J.-C. Bolot and A. U. Shakar. Analysis of a fluid approximation to flow control dynamics. In *Proc of IEEE INFOCOM'92*, pages 2398–2407, 1992.

[4] L.S. Brakmo, S.W. O'Malley, and L.L. Peterson. TCP vegas: New techniques for congestion detection and avoidance. Technical report, Departement of Computer Science The University of Arizona, 1994.

[5] F. Chatté, B. Ducourthial, D. Nace, and S.-I. Niculescu. Results about fluid approximation of a packet switched network. In *3rd IFAC Workshop on Time Delay Systems*, page To appear, December 2001.

[6] K. L. Cooke and P. van den Driessche. On zeroes of some transcendental equations. *Funkcialaj Ekvacioj*, 29:77–90, 1986.

[7] L.E. El'sgol'ts and S.B Norkin. *Introduction to the theory and applications of differential equations with deviating arguments*. Number 105 in Mathematics in Science and Eng. Academic Press, New-York (USA), 1973.

[8] S. Floyd, M. Handley, J. Padhye, and J. Widmer. Equation-based congestion control for unicast applications: the extended version. Technical Report TR-OO-003, International Computer Science Institute ICSI, Berkeley, CA, USA, March 2000.

[9] S. Floyd and T. Henderson. New reno modifications to fast recovery. Request For Comment 2582, RFC editor, http://www.rfc-editor.org/, March 1999.

[10] S. Floyd and A. Romanow. TCP selective acknowledgement options. Request For Comments 2018, RFC editor, http://www.rfc-editor.org October 1996.

[11] D. Ivanescu, R. Lozano, and S.-I. Niculescu. On closed-loop stability for mechanical systems with input delays. In *3rd IFAC Workshop on Time Delay Systems*, December 2001.

[12] R. Izmailov. Adaptive feedback control algorithms for large data transfer in high-speed networks. *IEEE Trans. Automat. Contr.*, 40:1469–1471, 1995.

[13] R. Izmailov. Analysis and optimization of feedback control algorithms for data transfers in high-speed networks. *SIAM J. Contr. Optimiz.*, 34:1767–1780, 1996.

[14] V. Jacobson. Congestion avoidance and control. In *Proc. of ACM SIGCOMM'88*, pages 314–329, 1988.

[15] S. Mascolo. Smith's principle for congestion control in high-speed data networks. In *Proc. 37th IEEE Conf. Dec. Contr.*, San Diego, 1997.

[16] S. Mascolo. Classical control theory for congestion avoidance in high speed internet. In *Proc. 38th IEEE Conf. Dec. Contr.*, pages 2709–2714, Phoenix AZ, 1999.

[17] S. Mascolo. Congestion control in high-speed communication networks using the Smith principle. *Automatica*, 35(1999):1921–1935, 1999.

[18] J. Nagle. Congestion control in IP/TCP internetworks. Request For Comment 896, RFC editor, http://www.rfc-editor.org/, January 1984.

[19] S.-I. Niculescu. *Delay effects on stability: A robust control approach*. LNCIS. Springer-Verlag, Heidelberg, May 2001. 383 pages.

[20] S. I. Niculescu and C. T. Abdallah. Delay effects on static output feedback stabilization. In *Proc. 39th IEEE Conf. Dec. Contr.*, Sydney, Australia, December 2000.

[21] The ns project, http://www.isi.edu/nsnam/ns/ns-documentation.html. *The ns manual*, August 2000.

[22] J.D. Padhye. *Towards a comprehensive congestion control framework for continuous media flows in best effort networks*. PhD thesis, University of Massachusetts Amherst, USA, 2000.

[23] J. Postel. User Datagram Protocol. Request For Comment 768, RFC editor, http://www.rfc-editor.org/, August 1980.

[24] J. Postel. Internet Protocol. Request For Comments 791, RFC editor, http://www.rfc-editor.org/, September 1981.

[25] J. Postel. Transmission Control Protocol. Request For Comment 793, RFC editor, http://www.rfc-editor.org/, September 1981.

[26] S. Ramakrishnan, S. Kalyanaraman, J. Wen, and H. Ozbay. Effect to time delay in network traffic control. In *American Control Conference*, Arlington, june 2001.

[27] R. Rejaie, J. Handely, and D. Estrin. RAP : An end-to-end rate-based congestion control mechanism for realtime streams in the internet. In *Proc. of IEEE INFOCOM'99, New-York, NY, USA*, March 1999.

[28] R.W. Stevens. *TCP/IP Illustrated*. Addison Wesley, 1994.

[29] W. Stevens. TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms. Request For Comments 2001, RFC editor, http://www.rfc-editor.org January 1997.

[30] Y. Tobe, Y. Tamura, A. Molano, S. Ghosh, and H Tokuda. Achieving moderate fairness for UDP flows by path-status classification. In *Proc. of the IEEE Annual Conf. on Local Computer Networks (LNC'2000), Tampa, FL*, November 2000.

[31] Y. Tobe, Y. Tamura, and H. Tokuda. Detection of change in one-way delay for analysing the path status. Technical report, First Passive and Active Measurement Workshop, April 2000.