Designing a Delay-based Adaptive Congestion Control Mechanism using Control Theory and System Identification for TCP/IP Networks

Mitsushige Morita[†], Hiroyuki Ohsaki[‡] and Masayuki Murata[§]

[†] Sun Microsystems K.K
Setagaya, Tokyo 158-8633, Japan
E-mail: mitsushige.morita@sun.com

[‡] Graduate School of Information Science and Technology, Osaka University
Toyonaka, Osaka 560-8531, Japan
E-mail: oosaki@ist.osaka-u.ac.jp

[§] Cybermedia Center, Osaka University
Toyonaka, Osaka 560-0043, Japan
E-mail: murata@cmc.osaka-u.ac.jp

ABSTRACT

In the Internet, TCP (Transmission Control Protocol) has been used as an end-to-end congestion control mechanism. Of all several TCP implementations, TCP Reno is the most popular implementation. TCP Reno uses a loss-based approach since it estimates the severity of congestion by detecting packet losses in the network. On the contrary, another implementation called TCP Vegas uses a delay-based approach. The main advantage of a delay-based approach is, if it is properly designed, packet losses can be prevented by anticipating impending congestion from increasing packet delays. However, TCP Vegas was designed using not a theoretical approach but an ad hock one. In this paper, we therefore design a delay-based congestion control mechanism by utilizing the classical control theory. Our rate-based congestion control mechanism dynamically adjusts the packet transmission rate to stabilize the round-trip time for utilizing the network resources and also for preventing packet losses in the network. Presenting several simulation results in two network configurations, we quantitatively show the robustness and the effectiveness of our delay-based congestion control mechanism.

Keywords: Delay-based Congestion Control Mechanism, Packet Delay Dynamics, Classical Control Theory

1. INTRODUCTION

In the Internet, TCP (Transmission Control Protocol) has been used as an end-to-end congestion control mechanism.[1] Of all several TCP implementations, TCP Reno is the most popular implementation. TCP Reno is a loss-based approach since it estimates the severity of congestion by detecting packet losses in the network. On the contrary, another implementation called TCP Vegas[2, 3] uses a delay-based approach, which utilizes the variation of the round-trip time for congestion control. In such a delay-based approach, if it is properly designed, packet losses in the network can be prevented by performing congestion control from the variation of the round-trip time and therefore the throughput can be improved.

In our past studies,[4, 5] we have proposed an approach for modeling the packet delay dynamics using system identification.[6] We have regarded the network seen by a specific source host as a dynamic SISO (Single-Input and Single Output) system. We have modeled the round-trip time dynamics using the ARX (Auto-Regressive

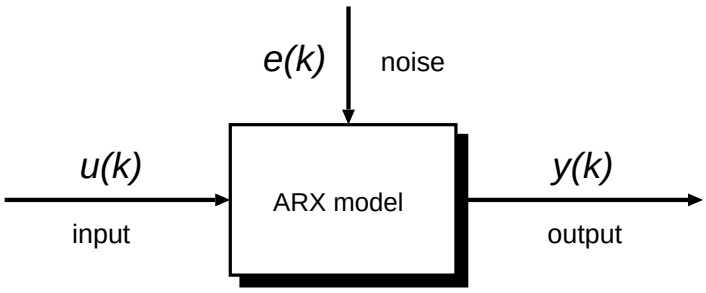


Figure 1: Block diagram of ARX model.

eXogenous) model. For example, the input to the network was the packet transmission rate from the source host, and the output from the network was the round-trip time measured at the source host.[5] By investigating the accuracy of the model built from measured data in LAN and WAN environments, we have found that the ARX model can well capture the round-trip time dynamics when the bottleneck link is shared by a small number of users.

In this paper, we design a rate-based congestion control mechanism using the modeling approach that utilizes the system identification. Specifically, applying the classical control theory[7] to the model describing the round-trip time dynamics, we design a rate-based congestion control mechanism, which uses the packet delay for congestion detection. We use two well-known techniques from the classical control theory: the pole placement technique using the state feedback and the state observer. We regard the network seen by a specific source host as the controlled system, and regard to achieve a constant round-trip time as the control objective. Presenting simulation results in two network models, we show that our proposed rate-based congestion control mechanism operates efficiently.

This paper is organized as follows. In Section 2, we shortly explain the system identification, the pole placement technique using the state feedback and the state observer. In Section 3, we explain the algorithm of our rate-based congestion control mechanism. In Section 4, we perform simulation experiments, and show the efficiency of our rate-based congestion control mechanism. Section 5 concludes this paper and discussthe future works.

2. SYSTEM IDENTIFICATION, POLE PLACEMENT USING STATE FEEDBACK, AND STATE OBSERVER

In what follows, we briefly explain three techniques, which will be used to design a rate-based congestion control mechanism: the system identification,[6] the pole placement using the state feedback,[7] and the state observer.[7]

2.1. System Identification

A system identification is the technique to estimate the parameters of the model describing the target system from the measured input and output data. In this paper, the ARX model[6] is used for modeling the round-trip time dynamics. First, we will explain the ARX model. The ARX model is a linear and time-invariant parametric model and widely used for various system identification problems. A single-input and single-output system can be represented by the ARX model. Figure 1 illustrates the block diagram of the ARX model.

Let $u(k)$ and $y(k)$ be the input and output data measured at kth constant sampling interval. The ARX model is defined as

$$\begin{aligned} A(q)\,y(k) &= B(q)\,u(k-n_d)+e(k) \\ A(q) &= 1+a_1q^{-1}+\ldots+a_{n_a}q^{-n_a} \\ B(q) &= b_1+b_2q^{-1}+\ldots+b_{n_b}q^{-n_b+1} \end{aligned}$$

where q^{-1} is the delay operator; i.e., $q^{-1}u(k) \equiv u(k-1)$. The numbers n_a and n_b are the orders of the ARX model. The number n_d corresponds to the delay from the input to the output. The white noise $e(k)$ represents the unmeasurable disturbance in the output data. All coefficients of the polynomials, a_i and b_i are parameters of the ARX model.

The system identification is the technique to build a mathematical model of the target system from measured input and output data. In the followings, we explain how the parameters of the ARX model are estimated using system identification. For compact notation, the parameters of the ARX model are represented by

$$\theta = [a_1,\ldots,a_{n_a},b_1,\ldots b_{n_b}]^T \tag{1}$$

We define $\hat{y}(k|\theta)$ as a *one-step-ahead prediction* of the ARX model; i.e., $\hat{y}(k|\theta)$ is the output of the ARX model using the input and output data before slot $k-1$ and parameters of the ARX model θ. More specifically, $\hat{y}(k|\theta)$ is defined as

$$\hat{y}(k|\theta) \equiv \psi(k)\,\theta^T \tag{2}$$

where

$$\psi(k) = [-y(k-1),\ldots,-y(k-n_a),u(k-n_d-1),\ldots,u(k-n_d-n_b)]^T$$

We also define $\varepsilon(k|\theta)$ as a *prediction error* at slot k, which is the difference between measured output $y(k)$ and the one-step-ahead prediction $\hat{y}(k|\theta)$.

$$\varepsilon(k|\theta) \equiv y(k)-\hat{y}(k|\theta) \tag{3}$$

In the system identification, the parameters of the ARX model θ are determined so that the prediction error is minimized. In particular, the least-squares method has been widely used, which determines θ so that the *loss function* is minimized. The loss function $J_N(\theta)$ is defined by

$$J_N(\theta) \equiv \frac{\sum_{k=1}^{N}\varepsilon(k|\theta)^2}{N} \tag{4}$$

where N is the number of the input and output data (i.e., the number of samples) used for system identification.

There are two types of system identification methods: batch (off-line) method and recursive (on-line) method.[6] The batch method determines parameters off-line using measured input and output data. On the contrary, the recursive method determines parameters on-line while measuring the input and output data. When using the least-squares method, the batch least-squares method needs inverse matrix operations, although the number of required samples is small. On the contrary, the recursive least-squares method does not need an inverse matrix operation, but the number of required samples is larger than that of the batch least-squares method.

In what follows, we explain the recursive least-squares method, which will be used in our rate-based congestion control mechanism. Let $\hat{\theta}(k)$ be the estimated parameters at slot k and $\mathbf{P}(k)$ as the covariance matrix at slot k.

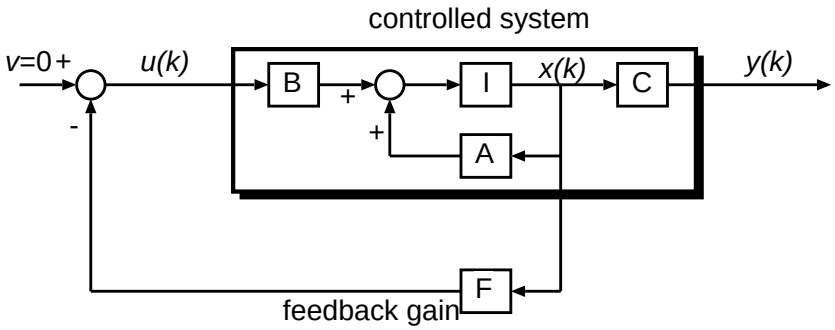


Figure 2: Block diagram of plant and state feedback.

Initialization

$$\begin{aligned} \hat{\theta}(0) &= 0 \\ \mathbf{P}(0) &= \gamma \mathbf{I} \ \ (\gamma > 0, \mathbf{I} \text{ is the identity matrix}) \end{aligned}$$

Update every measurement of input and output data

$$\begin{aligned} \hat{\theta}(k) &= \hat{\theta}(k-1) + \mathbf{P}(k)\,\psi(k)\,[y(k) - \psi^T(k)\,\hat{\theta}(k-1)] \\ \mathbf{P}(k) &= \mathbf{P}(k-1) - \frac{\mathbf{P}(k-1)\,\psi(k)\,\psi^T(k)\,\mathbf{P}(k-1)}{1 + \psi^T(k)\,\mathbf{P}(k-1)\,\psi(k)} \end{aligned}$$

In general, γ are set around 10^3–10^4.

2.2. Pole Placement using State Feedback

In this paper, we assume the controlled system is the single-input and single-output system. The pole placement using the state feedback is a technique to arbitrarly control the stability and the transient behavior of the target system by feeding back state variables of the target system[7] (see Fig. 2). In Fig. 2, the input $u(k)$ is determined based on state variables $\mathbf{x}(k)$. In a closed-loop system, the problem of controlling the stability and the transient behavior is equivalent to the problem of placing the poles, which represent the characteristic of the target system. Namely, if the poles of the target system can be placed at an arbitrarily position by choosing the feedback gain $\mathbf{F}$ in Fig. 2 appropriately, the stability and the transient behavior of the system can be arbitrarily controlled.

Using the state space representation, the single-input and single output system in Fig. 2 can be given by

$$\left\{ \begin{array}{lcl} \mathbf{x}(k+1) &=& \mathbf{A}\,\mathbf{x}(k) + \mathbf{B}\,u(k) \\ y(k) &=& \mathbf{C}\,\mathbf{x}(k) \end{array} \right. \tag{5}$$

where $u(k)$ is the input to the system and $y(k)$ is the output from the system at slot k. The vector $\mathbf{x}(k)$ consists of state variables at slot k, and $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ are constant matrices.

By assuming that all state variables $\mathbf{x}(k)$ are accurately measurable, the input to the system is given by the following state feedback.

$$u(k) = -\mathbf{F}\mathbf{x}(k) \tag{6}$$

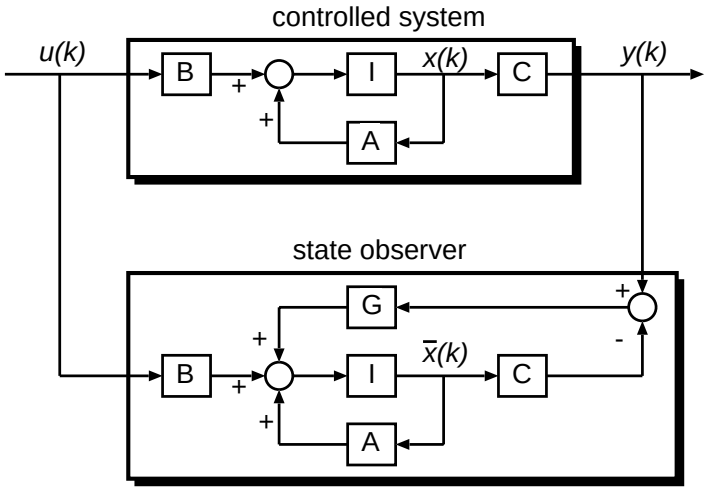


Figure 3: Block diagram of state observer.

Substituting Eq. (6) into Eq. (5) gives

$$\begin{cases} \mathbf{x}(k+1) &= (\mathbf{A} - \mathbf{BF})\mathbf{x}(k) \\ y(k) &= \mathbf{C}\mathbf{x}(k) \end{cases} \tag{7}$$

In the system given by Eq. (7), the poles of the system can be placed at any desired location by choosing the feedback gain $\mathbf{F}$ properly. Thus, the stability and the transient behavior of the system given by Eq. (7) can be determined arbitrarily.

However, for the pole placement using the state feedback, it is necessary that the target system is *controllable*.[7] A system is called *controllable* if all state variables $\mathbf{x}(k)$ can be controlled by the input $u(k)$. More specifically, the target system is controllable if the following equation is satisfied.

$$rank\mathbf{V} = n$$

where $\mathbf{V}$ is the controllability matrix defined by

$$\mathbf{V} \quad = \quad [\mathbf{B}, \mathbf{AB}, \mathbf{A}^2\mathbf{B}, \ldots, \mathbf{A}^{n-1}\mathbf{B}]$$

Note that the number n indicates the order of state variables $\mathbf{x}(k)$.

2.3. State Observer

The state observer is a mechanism to estimate state variables of the system from measured input and output data (Fig. 3). In the previous section, we assumed that all state variables $\mathbf{x}(k)$ are measurable. In practice, however, those state variables $\mathbf{x}(k)$ are unavailable or difficult to obtain. So the state observer is used to estimate state variables $\mathbf{x}(k)$ of the target system from the measured input $u(k)$ and output $y(k)$. Letting $\bar{\mathbf{x}}(k)$ be estimated state variables of the system, the following equation is obtained from Fig. 3.

$$\bar{\mathbf{x}}(k+1) \quad = \quad (\mathbf{A} - \mathbf{GC})\,\bar{\mathbf{x}}(k) + \mathbf{G}\,y(k) + \mathbf{B}\,u(k) \tag{8}$$

Let $q(k)$ be the difference between $\mathbf{x}(k)$ and $\bar{\mathbf{x}}(k)$, the following equation is obtained from Eqs. (5) and (8).

$$q(k+1) \quad = \quad (\mathbf{A} - \mathbf{G}\,\mathbf{C})\,q(k) \tag{9}$$

This equation indicates that the poles of the system given by Eq. (9) can be placed at any desired location by choosing $\mathbf{G}$ properly. Thus, when $k \to \infty$, the state observer can realize $q(k) \to 0$.

Note that for realizing the state observer, it is necessary that the system is *observable*.[7] A system is called *observable* if all state variables $\mathbf{x}(k)$ can be uniquely determined by observing the output $y(k)$. More specifically, the target system is observable if the following equation is satisfied.

$$rank\mathbf{N} = n$$

where $\mathbf{N}$ is the observability matrix defined by

$$\mathbf{N} \quad = \quad [\mathbf{C}^T, \mathbf{A}^T\,\mathbf{C}^T, (\mathbf{A}^T)^2\,\mathbf{C}^T, \ldots, (\mathbf{A}^T)^{n-1}\,\mathbf{C}^T]^T$$

3. DESIGNING A RATE-BASED CONGESTION CONTROL MECHANISM

3.1. Overview

Figure 4 shows the block diagram of our rate-based congestion control mechanism. In this paper, we regard the network seen by a specific source host as a controlled system. We regard the packet transmission rate as the output from the source host (i.e., the input to the network). We also regard the round-trip time as the input to the source host (i.e., the output from the network). We assume that the destination host performs some procedure, such as returning the ACK packet to the source host for each arriving packet, to enable the round-trip time measurement at the source host. In this paper, the control objective is to achieve a constant round-trip time for realizing a delay-based congestion control mechanism.

As depicted in Fig. 4, the source host is composed of two blocks: the system identification block and the control block. In the system identification block, the model represenging the round-trip time dynamics is obtained by measuring the input and output data at the source host. In the control block, a packet transmission rate is determined by applying the classical control theory to the model obtained in the system identification block.

For designing a rate-based congestion control mechanism, the control objective should be chosen carefully according to several performance criteria. There are several performance metrics of a rate-based congestion control mechanism.

- End-to-end throughput
- End-to-end packet loss probability
- End-to-end packet delay and its variance (i.e., jitter)

Our rate-based congestion control mechanism performs congestion control based on the variation of the round-trip time. Therefore, we define the control objective to achieve a constant round-trip time measured at the source host. If the round-trip time (i.e., the control objective) is chosen properly, it is possible to reduce the packet delay and its variance. Also, it is expected that the end-to-end throughput can be improved and packet loss probability can be kept small.[2]

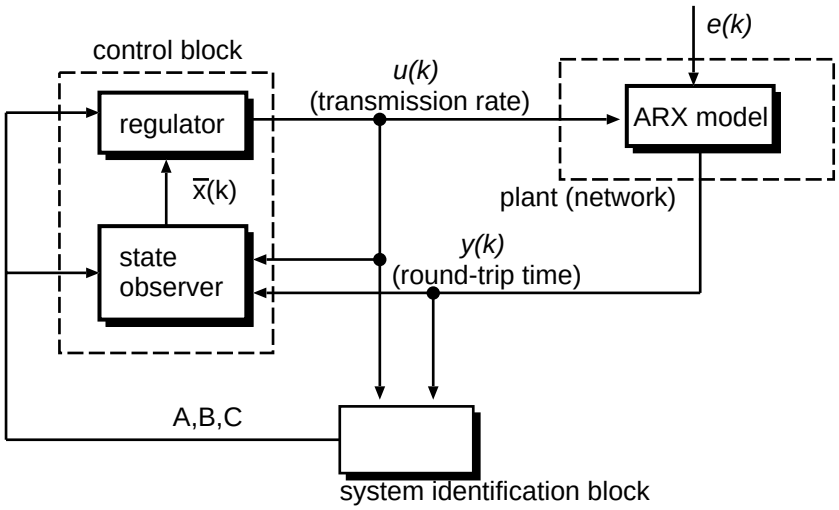


Figure 4: Block diagram of a rate-based congestion control mechanism.

3.2. System Identification Block

In the system identification block, the round-trip time dynamics are modeled from the measured input and output data using the system identification. In this paper, we use the ARX model to model the network seen by a specific source host. The input to the network, $u(k)$, is the packet transmission rate, and the output from the network, $y(k)$, is the round-trip time. By measuring the input and output data at the source host, the parameters of the ARX model are determined using the recursive least-squares method explained in Section 2.

Let us specifically define the input to the network and the output from the network. For system identification purposes, it is desirable for average values of input and output to be zero. In this paper, we therefore define the input to the network as the difference of the packet transmission rate from its average $\overline{u}$. We also define the output from the network as the difference of the round-trip time from the control objective y^*. Letting the sampling interval be T, $u(k)$ and $y(k)$ are defined as

$$\begin{aligned} u(k) &= \frac{\sum_{i\in\phi_s(k)} l(i)}{T} - \overline{u} \\ y(k) &= \frac{\sum_{i\in\phi_r(k)} (t_r(i) - t_s(i))}{|\phi_r(k)|} - y^* \end{aligned}$$

where $t_s(i)$ is the time at which the ith packet is injected into the network, and $t_r(i)$ is the time at which the ith ACK packet is received by the source host. We further introduce $l(i)$ as the size of the ith packet. We also introduce $\phi_s(k)$ (or $\phi_r(k)$) as the set of packet numbers sent (or received) during kth sampling interval; i.e.,

$$\begin{aligned} \phi_s(k) &\equiv \{n : k\,T \le t_s(n) < (k+1)\,T\} \\ \phi_r(k) &\equiv \{n : k\,T \le t_r(n) < (k+1)\,T\} \end{aligned}$$

where $\overline{u}$ is the average packet transmission rate. $\overline{u}$ is estimated using the low pass filter

$$\overline{u} \leftarrow \alpha\,\overline{u} + (1-\alpha)\,u(k) \tag{10}$$

where α is the weight of the low pass filter.

In our rate-based congestion control mechanism, it is probable for the source host to receive no packets during a sampling interval when the packet transmission rate is low. In such a case, average values of past input and output data will be used instead.

3.3. Control Block

Using the state observer, the control block estimates state variables of the target system from the input and output data measured at the source host. Next, it utilizes the pole placement technique using the estimated state variables. Consequently, it tries to reduce the difference between the round-trip time and the control objective y^* by controlling its packet transmission rate.

The state observer estimates state variables of the ARX model obtained in the system identification block. First, by converting the obtained ARX model into the state space representation, $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ are obtained. Next, using Eq. (8), it estimates state variables $\mathbf{x}(k)$ of the target system. Then, $\mathbf{G}$ is determined so that the poles of the system described in Eq. (9) are placed at $\lambda_i (1 \leq i \leq max(n_a, n_b) + n_d - 1)$. It should be noted that the parameters of the ARX model are changed every sampling interval since the recursive least-squares method is used in the system identification block. Hence, the state observer updates $\mathbf{G}$ every N_O sampling intervals.

Using state variables $\bar{\mathbf{x}}(k)$ of the ARX model estimated by the state observer, the pole placement using the state feedback is applied. The pole placement realizes a rate-based congestion control mechanism that converges the round-trip time to the control objective. More specifically, the pole placement using the state feedback is performed according to Eq. (7). Namely, $\mathbf{F}$ is determined so that the poles of the system given by Eq. (7) are placed at $\mu_i (1 \leq i \leq max(n_a, n_b) + n_d - 1)$. It should be noted that the parameters of the ARX model are changed in every sampling interval since the recursive parameter estimation is performed by the system identification block. Hence, $\mathbf{F}$ is updated every N_R sampling intervals.

At the source host, the packet transmission rate is determined according to the feedback control using the state feedback. Note that the packet transmission rate is changed every N_U sampling intervals. The source host sends packets using UDP protocol. For simple implementation, we assume that the upper layer protocol performs error recovery caused by packet losses, so that our rate-based congestion control mechanism does not react to packet losses.

For efficient operation, the control objective y^* must be chosen carefully. In this paper, using the same technique with TCP Vegas,[3] y^* is chosen as

$$y^* = \tau + \frac{\nu}{\overline{u}} \tag{11}$$

where τ is the minimum round-trip time measured at the source host and ν is a control parameter.

4. SIMULATION

4.1. Network Model

In this section, we evaluate the effectiveness of our rate-based congestion control mechanism. Using simulation experiments, we investigate how the actual round-trip time converges to the control objective. We also investigate how the throughput is improved by introducing our rate-based congestion control mechanism. In the followings, using the ns2 simulator,[8] we evaluate the performance of our rate-based congestion control mechanism in two network configurations. Table 1 summarizes parameters used in our simulation experiments. Note that during the first N_C slots after starting the packet transmission, the system identification block models the round-trip time dynamics although the control block does not perform any procedure. During this period, the packet inter-departure time from the source host is varied according to the exponential distribution, where the average packet transmission rate is set to 10 [Mbit/s].

Table 1. Simulation parameters.

T	sampling interval	30	ms
n_a, n_b	orders of ARX model	3, 3	
n_d	delay of ARX model	3	
N_R	$\mathbf{F}$ update interval	10	slot
N_O	$\mathbf{G}$ update interval	10	slot
N_U	$u(k)$ update interval	1	slot
N_C	modeling interval	500	slots
λ_i	desired poles of $\mathbf{F}$	0.5	
μ_i	desired poles of $\mathbf{G}$	0.5	
α	weigth of the low pass filter	0.95	
ν	control objective	100	packets
γ	initial vaule of least-squares method	10,000	

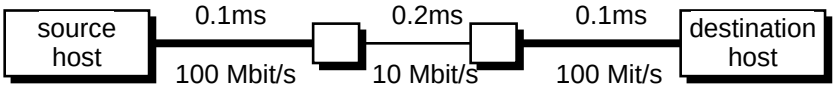


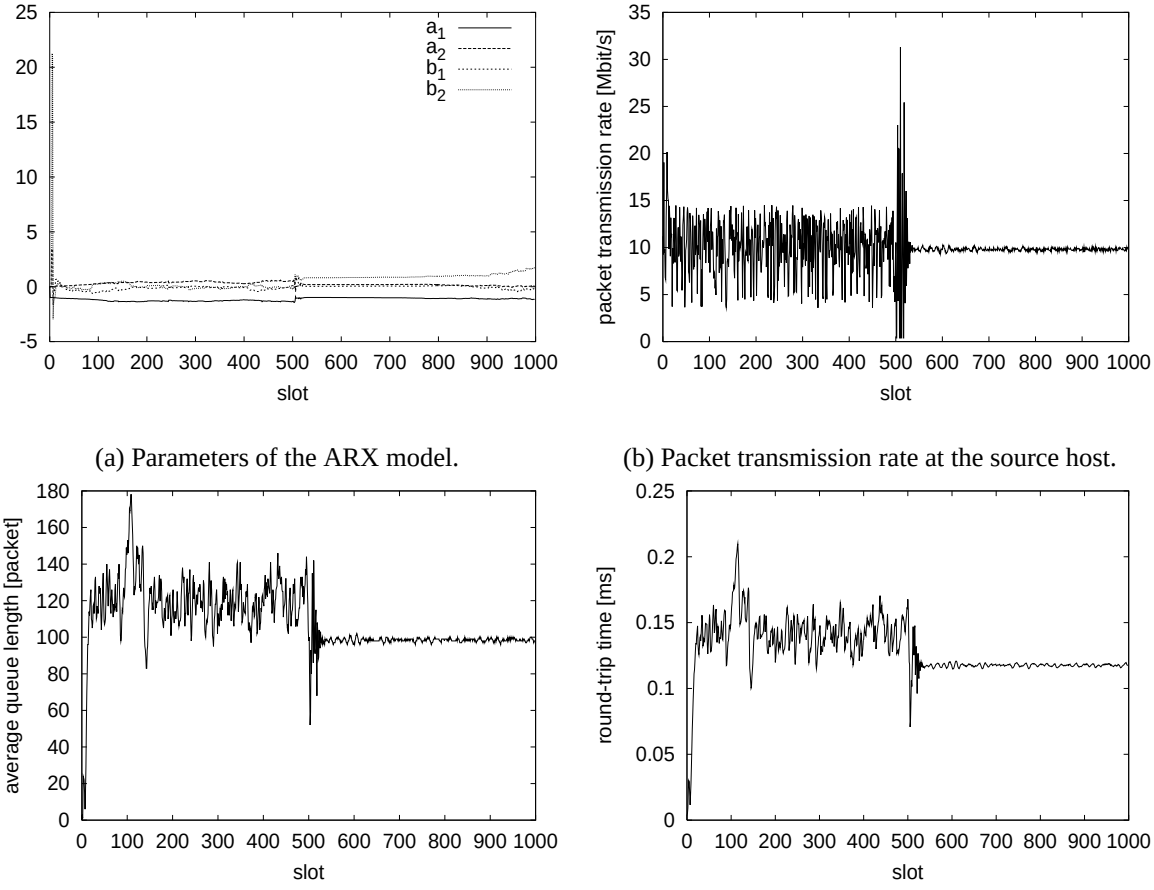
Figure 5: Network model $\mathbf{N1}$.

4.2. NetWork Model $\mathbf{N1}$

As depicted in Fig. 5, the network model $\mathbf{N1}$ is simple, where no background traffic exists. There are two routers between the source and the destination hosts. The source and destination hosts are connected to the router via 100 [Mbit/s] link. Each router is connected to 10 [Mbit/s] link, which is the bottleneck in the network. The propagation delay of the bottleneck link is 0.2 [ms], and the buffer size of routers is equally set to 200 [packets].

Figure 6 shows simulation results in the network model $\mathbf{N1}$. Figure 6 (a) shows the parameters of the ARX model estimated by the system identification block at each slot. This figure indicates that the parameters of the ARX model converge after about 10 slots. Figure 6 (b) shows evolution of the packet transmission rate at the source host. After the rate control is started at 500 slot, the packet transmission rate is controlled by the control block. One can find that the packet transmission rate is stabilized at 10 [Mbit/s] soon after the control block is initiated. Recalling that the bandwidth of the bottleneck link is 10 [Mbit/s], one can also find that the rate-based congestion control mechanism utilizes the network resources efficiently.

Figure 6(c) shows evolution of the queue length, which is the average of instantaneous queue length of the bottleneck router in each slot. This figure indicates that the queue length is stabilized around 50 slots (i.e., about 1,500 [msec]) after the rate control is started at 500 slot. At this time, the queue length is equal to the control objective $\nu = 100$ [packet]. Therefore, we can conclude that in this network model, our rate-based congestion control mechanism perfectly achieves the control objective. Figure 6(d) shows evolution of the round-trip time, which is the average of measured round-trip times at the source host in each slot. One can find that our rate-based congestion control mechanism can stabilize the round-trip time at the desired value y^*. From these observations, we conclude that our rate-based congestion control mechanism can operate very efficiently in the network model $\mathbf{N1}$.



(a) Parameters of the ARX model.

(b) Packet transmission rate at the source host.

(c) Queue length at the bottleneck router.

(d) Round-trip time measured at the source host.

Figure 6: Simulation results in network model **N1**.

4.3. Network Model $\mathbf{N2}$

In the network model $\mathbf{N2}$ (see Fig. 7), the bottleneck link is shared by several users. Except this point, the network model $\mathbf{N2}$ is the same with the network model $\mathbf{N1}$. In the network model $\mathbf{N2}$, two types of background traffic are generated; i.e., UDP packets of 3 [Mbit/s] are generated from source hosts $S1$ and $S2$ to destination hosts $R1$ and $R2$, respectively. The packet transmission from $S1$ to $R1$ is initiated at 1,000 slot, while from $S2$ to $R2$ is at 1,500 slot.

Figure 8 shows simulation results in the network model $\mathbf{N2}$. We can find from Fig. 8(a) that parameters of the ARX model slowly converge after the background traffic is injected into the network (i.e., at 1,000 slot and 1,500 slot). Figure 8(b) shows that around 100 slots the packet transmission rate is stabilized after the amount of background traffic has changed. We can find that our rate-based congestion control mechanism can utilize the network resource unused by non-controlled background traffic very efficiently. Figure 8(c) shows that the queue length quickly converges to the control objective $\nu = 100$ [packets], but it temporarily increases after the amount of background traffic is increased. Similarly, Fig.8 (d) indicates that our rate-based congestion control

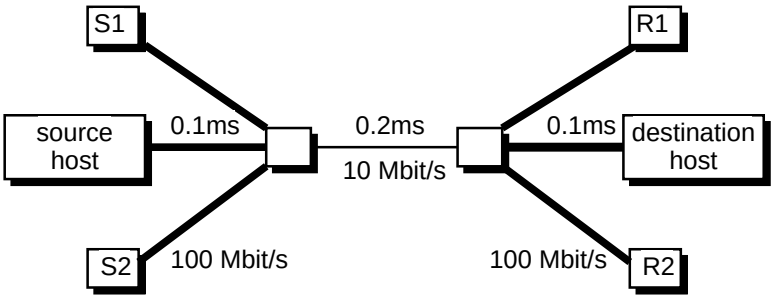


Figure 7: Network model $\mathbf{N2}$.

mechanism can realize the constant round-trip time. From these observations, we conclude that our rate-based congestion control mechanism can operate efficiently in the network model $\mathbf{N2}$ as well as in the network model $\mathbf{N1}$.

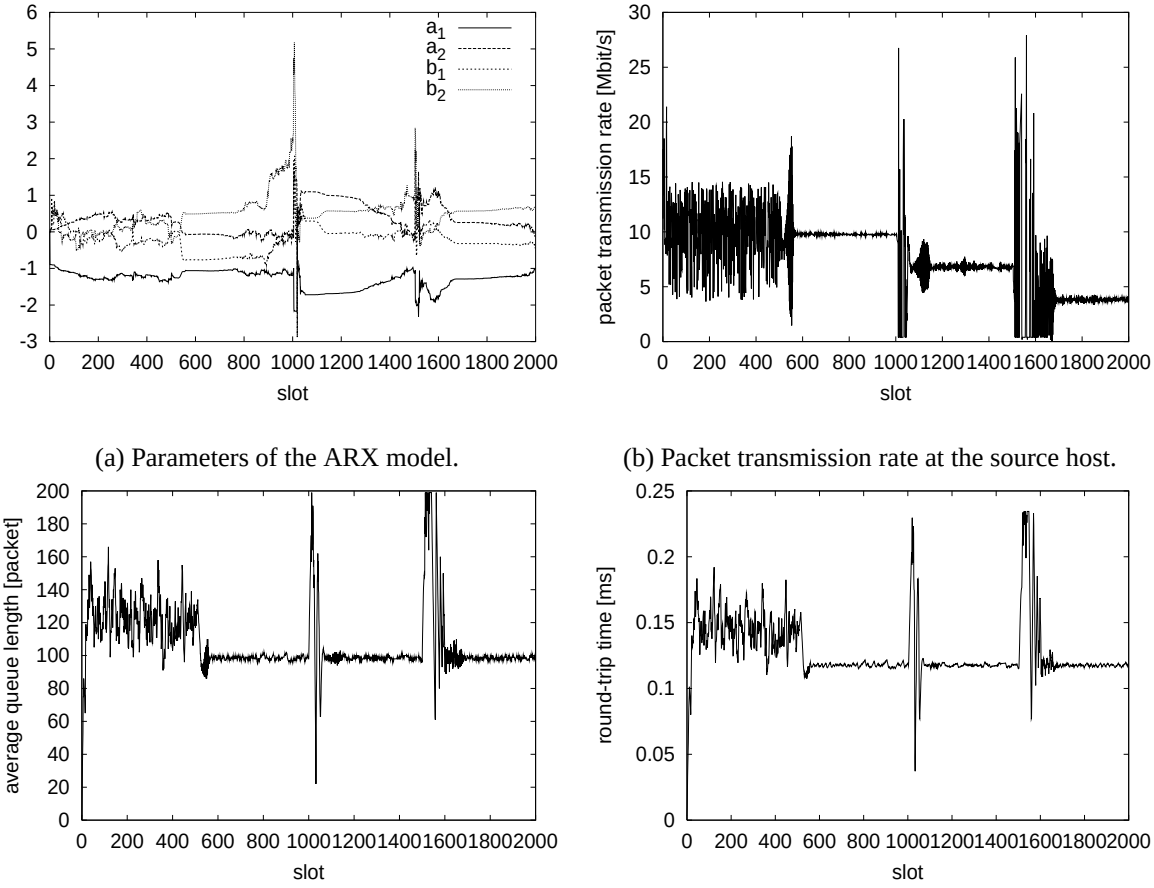
5. CONCLUSION

In this paper, by using the system identification and the classical control theory, we have proposed a rate-based congestion control mechanism, which utilizes measured packet delays for detecting congestion in the network. The key idea was to regard the network seen by a specific source host as a controlled system, and to stabilize the round-trip time as a control objective. Our rate-based congestion control mechanism was composed of two blocks: the system identification block and the control block. The system identification block models the round-trip time dynamics at the source host from the measured input and output data. The control block determines the packet transmission rate by combining two techniques: the pole placement using the state feedback and the state observer. Through simulation experiments, we have shown that our rate-based congestion control mechanism rapidly and accurately achieves the control objective of stabilizing the round-trip time. We have also shown that it utilizes the network resources quite efficiently. We believe that our rate-based congestion control mechanism can be applied to realtime applications as well as non-realtime ones since it controls the packet transmission rate to stabilize the measured round-trip time. In addition, it can be applied in various network environments since it models the round-trip time from measurement results using system identification.

However, in general, an accurate model is difficult to obtain in rather complex networks where the noise (e.g., background traffic) is non-negligible. Therefore, our rate-based congestion control mechanism is more effective especially in rather simpler networks where the bottleneck link is shared by a small number of users. As summarized in Tab. 1, our rate-based congestion control mechanism has many control parameters. In particular, parameters used in the system identification block (i.e., parameters for modeling the round-trip time dynamics) should be determined carefully. If these parameters are inappropriate, the ARX model fails to capture the round-trip time dynamics so that the rate-based congestion control will not work correctly. It is also important to carefully choose poles of the control block. These control parameters are directly related to the stability and the transient behavior of the rate-based congestion control mechanism. However, it is well known that there is a trade-off between the transient behavior and the stability; i.e., when these poles are close to the origin in the complex plane, the transient behavior is good whereas the stability may not be good. As a future work, we will investigate how to determine these control parameters for practical purposes.

REFERENCES

1. J. Postel, "Transmission control protocol," *Request for Comments (RFC) 793* , Sept. 1981.



(a) Parameters of the ARX model.

(b) Packet transmission rate at the source host.

(c) Queue length at the bottleneck router.

(d) Round-trip time measured at the source host.

Figure 8: Simulation results in network model **N2**.

2. R. Jain, "A delay-based approach for congestion avoidance in interconnected heterogeneous computer networks," *ACM Computer Communication Review* **19**, pp. 56–71, Oct. 1989.
3. L. S. Brakmo, S. W. O'Malley, and L. L. Peterson, "TCP Vegas: New techniques for congestion detection and avoidance," in *Proceedings of ACM SIGCOMM '94*, pp. 24–35, Oct. 1994.
4. H. Ohsaki, M. Murata, and H. Miyahara, "Modeling end-to-end packet delay dynamics of the Internet using system identification," in *Proceedings of Seventeenth International Teletraffic Congress*, pp. 1027–1038, Dec. 2001.
5. H. Ohsaki, M. Morita, and M. Murata, "Measurement-based modeling of Internet round-trip time dynamics using system identification," to be presented at *the Second IFIP-TC6 Networking Conference (NETWORKING 2002)* , May 2002.
6. L. Ljung, *System identification — theory for the user*, Prentice Hall, Englewood Cliffs, N.J., 1987.
7. R. Isermann, *Digital control systems, Volume 1: fundamentails, deterministic control*, Springer-Verlag Berlin Heidelberg, 1989.
8. "The network simulator – ns2." available at `http://www.isi.edu/nsnam/ns/`.