Control Architecture for Tuning Intensity and Burstiness of Traffic

Qiong Li
Wireless Communications and Networking Department
Philips Research USA
345 Scarborough Road, Briarcliff Manor, NY 10510
qiong-stan.li@philips.com

David L. Mills
Department of Electrical and Computer Engineering
University of Delaware
Newark, DE 19716
mills@udel.edu

***Abstract*— In this paper we explore a control architecture that can implement active queue management and transport control. For queue management, two independent metrics — utilization factor ρ and burstiness factor η, are used to characterize queueing congestion state. Consequently, two information bits are generated to guide the sender's adaptation of traffic intensity and burstiness. Transport control is achieved through the combination of a window and a token bucket that can respond to the two-bit feedback properly. Preliminary simulation study shows that this architecture can easily achieve a specified throughput and constrain both queueing delay and delay jitter within specified bounds.**

I. Introduction

Congestion control can be done either solely by the end systems, as for TCP [1], Packet-Pair [2] and the equation-based scheme [3], or by cooperation between end-systems and network routers, such as in DECnet [4], RED + ECN-TCP [5], [6], REM [7] and ERICA [8] for ABR ATM. Traffic adjustment in a control scheme can be either window-based, as for TCP, or rated-based, as in ERICA. The performance of these schemes and similar ones has been extensively investigated by means of simulation [9], fluid models [10], control theoretic approaches [11], [2] and optimization problems [12]. A common property of these schemes is that they all use single control loops.

In this paper, we explore an architecture that can realize queue management and transport control functionalities with two decoupled control loops, so it can implement both traffic intensity and burstiness controls simultaneously and independently.

The queue management algorithm uses two parameters — the link utilization factor ρ and a so-called burstiness factor η (see Section II-B for definition) to characterize queuing state, and generates two bits of feedback information (ρ-bit and η-bit) to indicate congestion. We call this mechanism Two-Dimensional Queue Management (TDQM).

Transport control is performed through the combination of a window and a token bucket that can not only achieve reliable delivery, but also decouple receiver buffer protection from rate control, and from burstiness control.

The two loops (ρ-loop and η-loop) are designed to control different aspects of networking systems. The ρ-loop is used to maintain system stability and robustness (or first-order behavior), while the η-loop is used to control high frequency variations in either traffic intensity or system transient performance (or second-order behavior). In this paper, we present the principles behind this control architecture and test its control benefits through simulation. Preliminary results show that the system can yield a higher throughput while maintaining smaller delays and delay jitter than when only one loop, such as ρ-loop, exists.

The rest of this paper is organized as follows. Section II and III present the two components of our control architecture and their algorithms. Section IV presents ns-2 simulation results. We conclude in Section V with comments on future work.

II. TDQM

TDQM is similar in spirit to RED [5] except that it uses a two-dimensional – ρ and η – model, to characterize queueing behavior. In the following, we introduce the algorithms for calculating ρ and η.

A. Utilization Factor

To measure real-time ρ ($= \lambda/\mu$ by definition), we set an observation time window $\bar{w}$. The measured ρ is the average within the time period $\bar{w}$.

The specification of $\bar{w}$ is vital to the control of queueing behavior. Assume $b_{\bar{w}}$ is the total busy time of the queue in $\bar{w}$, and ρ_{ref} is the control reference. We can easily show that:

$$\mathbf{E} b_{\bar{w}} \leq \rho_{ref} \bar{w}. \tag{1}$$

Therefore, if $\rho_{ref} < 1$, the ρ-loop control tends to break up long busy periods into smaller ones that are of the order of $\rho_{ref}\bar{w}$. We recommend that $\bar{w}$ be chosen of the order of the queue buffer drainage time.

B. Burstiness Factor

Let $V(t)$ be the queueing delay. An instance $V(t)$ of a busy period is shown in Figure 1, in which t_i and b_i are the start time and the span time of this busy period, respectively. η is defined as:

$$\eta = \frac{\int_{t_i}^{t_i+b_i} V(t)dt}{0.5 b_i^2}. \tag{2}$$

Equation (2) shows that η (≤ 1) is the ratio between the area under the $V(t)$ and that of the dotted triangle shown in Figure 1.

Let $\overline{V_{b_i}} = \frac{1}{b_i}\int_{t_i}^{t_i+b_i} V(t)dt$ be the average of $V(t)$ in this busy period. Equation (2) can be simplified as $\eta = 2\overline{V_{b_i}}/b_i$.

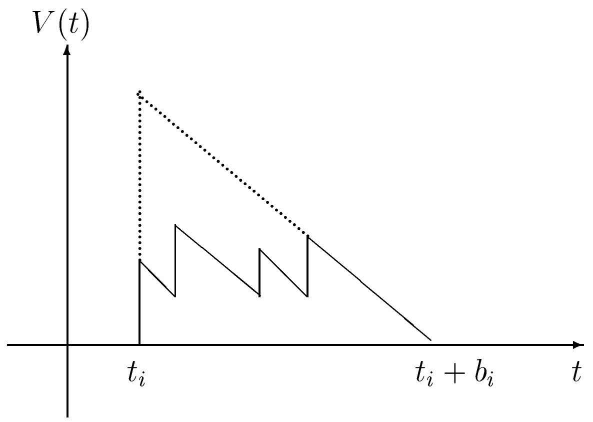


Fig. 1. An instance of queue busy period.

Therefore, another way to look at (2) is that it expresses the double of the growth rate of the average queue in this busy period.

η has the following characteristics:

- Since η characterizes the growth rate of the average queue in a busy period, it is only related to high-order traffic statistics.
- Since ρ is determined only by first-order traffic statistics, η and ρ are independent. Consequently, the two control loops are decoupled.
- Both η and ρ are dimensionless, and change in the same range of [0, 1]. We assume $V(t)$ has been normalized by the link speed.
- For the extreme bursty case, such as when all packets served in the same busy period arrive at the same time, we have $\eta \to 1$, while for the smoothest case, such as when the packets are evenly scattered in the busy period, we then have $\eta \to 0$, implying that η can somehow monotonically indicate the burstiness of arrivals.

In practice, we define a minimum value $\bar{b}$. Only when the length of a busy period is larger than $\bar{b}$ will it invoke the calculation of η. Tiny busy periods are merged together, and treated as a single artificial busy period. When the time scale of the artificial one $> \bar{b}$, a round of η calculation is then launched.

C. Packet Marking Scheme

TDQM marks the packet header with two bits (called ρ-bit and η-bit, respectively). The marking scheme is similar to RED. A general algorithm is shown in Figure 2.

As in RED, the TDQM scheme needs the support of filter algorithms to calculate $\bar{\rho}$, $\bar{\eta}$, P_ρ and P_η. These algorithms are very similar to that used in RED [5]. Due to space limitations, we do not repeat them here.

III. Transport Control

In this section, we present a skeleton design of a transport control protocol, which can adjust both the sending rate and the burstiness of traffic with the aid of TDQM.

for each packet arrival
  calculate the average utilization factor $\bar{\rho}$
  if $min_{\rho_{th}} < \bar{\rho} < max_{\rho_{th}}$
    calculate probability P_ρ
    with probability P_ρ, mark the arriving packet with a ρ-bit
  else if $max_{\rho_{th}} < \bar{\rho}$
    mark the arriving packet with a ρ-bit
  calculate the average burstiness factor $\bar{\eta}$
  if $min_{\eta_{th}} < \bar{\eta} < max_{\eta_{th}}$
    calculate probability P_η
    with probability P_η, mark the arriving packet with a η-bit
  else if $max_{\eta_{th}} < \bar{\eta}$
    mark the arriving packet with a η-bit

Fig. 2. A general algorithm of TDQM

A. Control Model

The current flow control model of TCP is a mono-window model. In the mono-window model, a single window is used to play three roles:

- tracking the sent data to achieve reliability
- receiver buffer protection
- network congestion control

There is some inefficiency with the mono-window model. First, the TCP sender uses the returned ACKs as a “clock” to step forward and open up the control window [1]. The idealized stable state scenario behind the original design of the TCP flow control is that each ACK serves as a signal to indicate that a packet has been safely delivered to the receiver, and the buffer within the network used to hold this packet is emptied (or more precisely, a unit of the network pipe is empty). Each new ACK warrants the sender to inject a new packet into the network. Therefore, the mono-window model actually accomplishes buffer resource control. With the growth in scale and usage of computer networks, network buffers are heavily shared by huge number of various flows that are statistically multiplexed together. The dynamics and the burstiness of the merged traffic have broken the idealized stable state scenario of the TCP flow control—a new ACK cannot guarantee the existence of empty buffer in the network. In this situation, it may be desirable to tune the control mechanism from the current precise ACK-clocking to a statistical ACK-clocking, meaning that each new ACK may not necessarily and immediately clock a new packet into the network, but rather certain statistical congestion measurements derived or explicitly piggybacked by ACKs may preferably be used to adapt the sender behavior.

Second, since the receiver buffer state is independent of the network congestion state, the functionalities for buffer protection and network congestion control should be decoupled from each other. An ideal flow control model should be able to respond to different state changes independently. The mono-window model lacks the flexibility to decouple the control functionalities, and jeopardizes the control efficiency under some conditions. To overcome the inefficiency of mono-window control model, a dual-window control model, which decouples receiver buffer protection from congestion control, was first introduced in [13].

Although the dual-window model decouples the receiver buffer state and network congestion state, it is still not sufficient enough to use a single window to track network congestion state changes. The reason is that, the congestion could be caused by either traffic intensity or traffic burstiness, which may not be easily distinguished by single window. To overcome this inefficiency and harmonize with TDQM, we propose a new control model as below.

Our control model is illustrated in Figure 3. This model maintains two control windows (W_b and W_p) and a token bucket (TB). W_b plays two control roles: tracking the sent data to achieve reliability and protecting the receiver buffer. W_p, a virtual window controlling the generating rate of tokens to TB, is adapted by the ρ-bit of TDQM. The TB has a capacity limit (T_c) on its stored tokens (the maximum number of tokens that can be found in the bucket at any time). When the number of stored tokens in the bucket reaches T_c, further tokens will be discarded (token overflow). T_c is adapted by the η-bit of TDQM, therefore, T_c essentially controls the burstiness of the sender. Also in this figure, TDQM is responsible for marking packets when necessary based on congestion states. If a packet is marked with either or both of the bits, the receiver will echo them back in the following ACK packet.

B. Operation and Property

The operation of the control model is briefly described below. Interested readers may refer to [14] for details.

Criteria for Packet Output: A packet is sent only if its sequence number (P_s) falls in window W_b and TB is not empty. Each packet consumes a token from TB.

W_b **Adaptation**: Always set to receiver buffer size, as explicitly communicated as being available by the receiver.

W_p **Adaptation**: Adapted by ACKs marked with a ρ-bit in a similar way to RED+ECN-TCP.

Token Generation: A variable rate Poisson process governs the token arrivals to the token packet. See Section III-C for details.

T_c **Adaptation**: Initialized to a maximum (T_{max}), and then adapted down exponentially by ACKs marked with an η-bit, or up linearly by un-marked ACKs if $T_c < T_{max}$. Always keeps $1 \le T_c \le T_{max}$. If T_c reaches 1, W_p is decreased exponentially using the η-bit instead.

This control model has the following properties.

- Decouples congestion control from buffer protection, and burstiness control from rate control.
- The randomness introduced by the token generation process can partially suppress the potential phase effect of multiple senders. The output of each sender conforms to the leaky bucket model [15] except the model parameters are adapted by TDQM and the receiver, so that the traffic intensity and burstiness can be tuned independently.
- Serves both stream-like traffic (FTP) and impulse-like transaction traffic very well.
- The token generation rate can be proportional to, but not necessarily equal to, W_p/RTT, leaving one more degree of freedom in adapting the sending rate (see the discussion in next section).

C. Token Generation

In order to mimic TCP behavior, the rate of the token generation process is designed to be adaptively proportional to W_p/RTT, which has the dimension of packet/second. Figure 4 shows a general algorithm for token generation.

```
whenever the token_generation timer expires
    if there is an uncleared timeout event
        keep only one token in the bucket
    else if the number of stored tokens does not reach bucket limit
        push one more token into the bucket
    generate an exponentially distributed random number rand_
                    with mean equal to g RTT/W_p
    reschedule the token_generation timer at rand_
```

Fig. 4. The token generation algorithm

As shown in this algorithm the rate can be adjusted by varying g. If $g > 1$, then the rate will be less than W_p/RTT, implying less aggressive than normal TCP. This feature is very useful especially when the router is so nearly full that each flow's fair share is less than one packet per round-trip time. By choosing $g > 1$, the sender can decrease the rate to less than one token per round-trip time. Conversely, if $g < 1$, the sender will be more aggressive than normal TCP. In cases where only a few flows share a huge pipe, each flow should be increased more aggressively than normal linear adaptation to exploit unused link capacity quickly.

IV. SIMULATION

In this section, we explore the performance of our control architecture using simulator ns-2 [16]. We extended ns-2 to support our control model.

A. Simulated Network

Figure 5 shows the topology of the simulated network. Node 1 is a TDQM-enabled router. All user nodes comprise modified TCP senders capable of implementing our designed controls. Node 0 is a common sink. Link A, with a bandwidth of 45 Mbps, is the bottleneck of the network. All other links have much higher capacities, such as 1 Gbps. Node 2 is comprises a ping agent for measuring path delays whenever we want.

Other common settings of the simulation: maximum values of W_b, W_p and T_c are all set to 240, packet size to 1000 bytes; delay of link A to 40 ms, queue buffer to 240 packets, and delays of other links to 1 ms.

Two types of traffic generators were used: infinite FTP sources and impulse sources. Impulse sources can be constructed by configuring an Exponential On/Off traffic generator of ns-2 with a very short On period but a relatively long Off period. We set up 30 user nodes in the simulation.

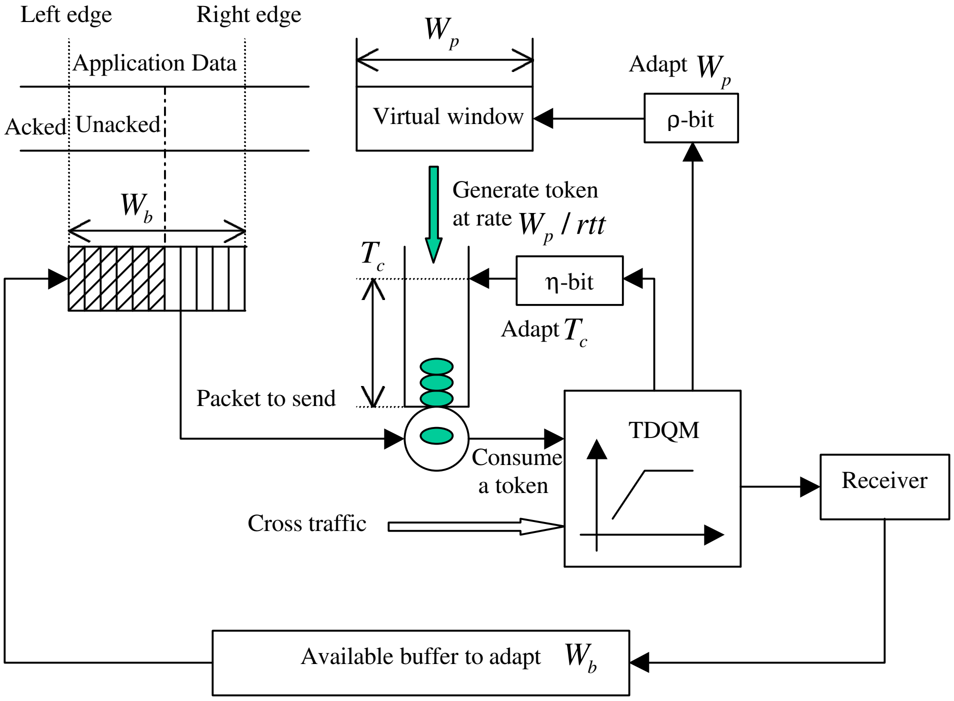


Fig. 3. The architecture of the TCP Newark control model.

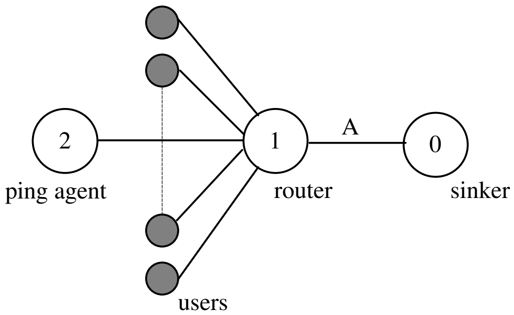


Fig. 5. The topology of the simulated network

B. Results

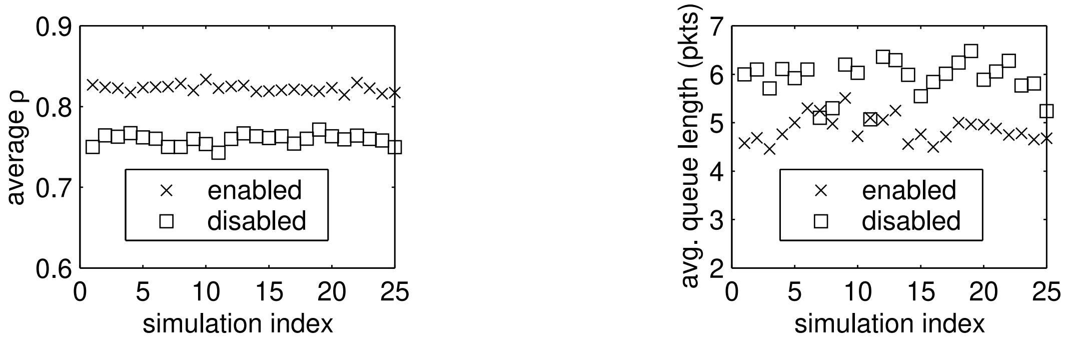
We have conducted extensive simulation studies on the dynamic performance and robust properties of our control architecture with various TDQM settings and different combinations of traffic sources. Due to space limitations, we only present one simulation case here to illustrate the obvious benefits of having two decoupled control loops. We also summarize the observations from our other simulations following this case study.

In this case, half of the user nodes comprise infinite FTP sources and the other half impulse sources. $[min_{\rho_{th}}, max_{\rho_{th}}] = [0.9, 1.0]$. $[min_{\eta_{th}}, max_{\eta_{th}}] = [0.1, 0.2]$ or $[1.0, 1.0]$. With settings of η-loop of $[1.0, 1.0]$, this loop is essentially disabled. We run the simulation 50 times; each run lasts for 100 sec of simulation time. Half of the runs have the η-loop enabled while the other half have it disabled. In all runs, we set $\bar{w}$ to half of the queue buffer drainage time and $\bar{b} = 0.025\bar{w}$ (corresponding to a busy period of three packets).

Figure 6 shows the average ρ and average queue length of each simulation. We can clearly see the benefits of having the η-loop control: under the same ρ-loop settings and statistically similar traffic conditions, when η-loop is enabled, the simulations show higher throughput, which is closer to the configured range [0.9,1.0], but shorter queue lengths than when η-loop is disabled.

Other observations of our simulations:

- The ρ-loop is essential to the stability of the network. Without this loop, periodic buffer overflow will occur. It can also effectively regulate the traffic so that resource consumption will not exceed a configured utilization factor. The η-loop can fine tune the dynamic behavior of the queue, such as maintaining a smoother queue variation, generating smaller delay jitter.
- Both loops can reach stable states within 12 round-trip times. Simulated ρ oscillates around its settings as predicted by the fluid model in [14]. η also changes around its settings in a small range.



(a)

(b)

Fig. 6. (a) the average ρ's, and (b) the average queue lengths, with η-loop enabled and disabled .

- Window W_p adapts in a way similar to normal TCP congestion control window, and token limit T_c decreases exponentially from its maximum to a stable range.
- Token accumulation is observed in senders that comprise impulse sources, implying that the transport control can sustain a certain burstiness, which is desirable for medium-sized transaction-style exchanges.

More simulation results can be found in our previous work [14].

V. Summary and Discussion

In this paper we explored an interesting control architecture that can independently tune traffic intensity and burstiness. Unlike conventional control models that use only a single metric (such as either rate/utilization factor, or smoothed queue length) to characterize queueing behavior, our control model uses two independent metrics — the utilization factor (ρ) and a newly introduced burstiness factor (η) to characterize queueing state. Consequently, two decoupled control loops (ρ-loop and η-loop) can be established: one for system robustness control (first-order control) and the other for burstiness control (second-order control).

Studies have shown that single-metric models have limitations. For example, using rate alone may lead to an unstable system. Also, since queue length is a combined result of both traffic intensity and burstiness, using smoothed queue length as control metric may not be sufficient to guide source adaptation, since it may not be clear to the source whether it should decrease sending rate, or just smooth out burstiness. It was reported that the RED queue suffers from configuration difficulties in practice; this may be related to the limitations of its single-metric model.

We realize that our architecture is not fully compatible with the current Internet control model which utilizes only single-bit feedback. However, we think it is worthwhile to think outside the current framework so we may cast new insight on the same issue and help us improve our design even within the current framework.

This paper describes only a simplistic investigation of this control architecture. More simulations are necessary to demonstrate the advantages of this architecture over others, such as RED + ECN-TCP. Also, some algorithms employed by this architecture need to be further investigated.

References

[1] V. Jacobson, "Congestion avoidance and control," in *Proceedings of ACM SIGCOMM'88*, Stanford, CA, 1988, pp. 314–329.
[2] S. Keshav, "A control-theoretic approach to flow control," in *Proceedings of ACM SIGCOMM'91*, Zurich, Switzerland, Sept. 1991.
[3] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-based congestion control for unicast applications," in *Proceedings of ACM SIGCOMM 2000*, Aug. 2000.
[4] K. K. Ramakrishnan and R. Jain, "A binary feedback scheme for congestion avoidance in computer networks with a connectionless network layer," in *Proceedings of ACM SIGCOMM'88*, Standford, CA, Aug. 1988, pp. 303–313.
[5] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, pp. 397–413, Aug. 1993.
[6] S. Floyd, "TCP and explicit congestion notification," *ACM Computer Communications Review*, vol. 21, no. 6, pp. 8–23, 1994.
[7] D. E. Lapsley and S. H. Low, "Random early marking for internet congestion control," in *Proceedings of IEEE Globecom'99*, Dec. 1999.
[8] S. Kalyanaraman, R. Jain, S. Fahmy, R. Goyal, and B. Vandalore, "The ERICA switch algorithm for ABR traffic management in ATM networks," *IEEE/ACM Transactions on Networking*, vol. 8, no. 1, pp. 87–98, Feb. 2000.
[9] D. Lin and R. Morris, "Dynamics of random early detection," in *Proceedings of ACM/SIGCOMM*, 1997.
[10] V. Misra, W. B. Gong, and D. Towsley, "Fluid-based analysis for a network of ARQ routers supporting TCP flows with an application to RED," in *Proceeding of ACM/SIGCOMM*, 2000.
[11] S. Shenker, "A theoretical analysis of feedback flow control," in *Proceedings of ACM SIGCOMM'90*, Philadelphia, PA, Sept. 1990, pp. 156–165.
[12] S. H. Low and D. E. Lapsley, "Optimization flow control, I: Basic algorithm and convergence," *IEEE/ACM Transactions on Networking*, vol. 7, no. 6, pp. 861–874, Dec. 1999.
[13] Z. Wang and J. Crowcroft, "A dual-window model for flow and congestion control," *The Distributed Computing Engineering Journal, Institute of Physics/British Computer Society/IEE*, vol. 1, no. 3, pp. 162–172, May 1994.
[14] Qiong Li, *Delay Characterization and Performance Control of Wide-Area Networks*, Ph.D Thesis, University of Delaware, 2000.
[15] J. Turner, "New directions in communications," *IEEE Communications Magazine*, vol. 24, pp. 8–15, Oct. 1986.
[16] DARPA funded VINT project, "UCB/LBNL/VINT network simulator - ns (version 2)," see http://www-mash.cs.berkeley.edu/ns, 1996.