TCP Retransmission Timer Adjustment Mechanism Using System Identification

M. Haeri, *Member, IEEE* and A. H. Mohsenian Rad

Abstract— **In this paper, the TCP retransmission timer adjustment mechanism is approached from the system's theoretic point of view. TCP uses a simple recursive system identification algorithm to capture both time variant and time invariant Internet behavior and provides a dynamic Round-Trip Time (RTT) predictor. Based on real Internet data collection, it is observed that when the retransmission timer is adjusted by using the proposed predictor instead of the traditional RTT estimator, its performance increases significantly.**

I. Introduction

RECENT studies have shown that 83%-95% of all IP traffic is managed by TCP [1]. It is also reported that about 13% of the TCP packets are to be retransmitted [2]. Receiving duplicate acknowledgements and TCP Retransmission Timeouts (RTO) are two events, which cause packet retransmissions. However, most of the retransmissions are triggered by timeout, which is announced by TCP retransmission timer [2, 3]. The timer is started when a segment is sent. If the segment is acknowledged before the timer expires, the timer is stopped. If on the other hand, the timer goes off before the acknowledgement comes in, TCP assumes that the transmitted packet is lost along its traveling in the Internet and the segment is retransmitted [4]. In this policy the main question is: *How long should the timeout interval be?*
In comparison with other similar timers like such used in Data Link Layer (DLL), this question is much more difficult to answer for TCP. RTT has high variance and it is not possible to have a fixed timeout interval [4]. Short timeout makes *useless retransmission*. It causes unnecessary traffic, reduce connection's *effective throughput*, and also trigger congestion control [4, 5] that may additionally reduce the end-to-end throughput. On the other hand, if the timeout is too conservative, it may cause long *idle time* before the lost packet is retransmitted.

Dr. M. Haeri is with the Electrical Engineering Department, Sharif University of Technology, Tehran, Iran. He is also the head of Advanced Control Systems Laboratory. Tel: +98 (21) 616 59 64, Fax: +98 (21) 602 32 61, E-mail: haeri@sina.sharif.edu

A. H. Mohsenian Rad is a graduate student of electrical engineering in Sharif University of Technology and a member of Advanced Control Systems Laboratory. E-mail: mohsenianrad@mehr.sharif.edu

In 1988, V. Jacobson [6] proposed a *dynamic* timeout interval adjustment mechanism in which, TCP applies a discrete event low pass filter to previous RTT measurements and estimate the future one. Most of the existing TCP implementations use his algorithm. Recent investigations have indicated that the usage of the algorithm results in an unsatisfactory performance [2, 7, and 8]. In an example test that was carried out in reference [2], 857142 TCP packet retransmissions have been observed. Among them, only 436747 packets were really lost. It means almost half of the retransmitted packets were unnecessary.

There are two complementary approaches to reduce the degrading effect of RTO in TCP's performance: First, modifying the TCP flow control scheme so that the duplicate acknowledgements are triggered more. This will decrease the reliance on RTO to recover from packet loss. Second approach deals with improving the TCP RTO adjustment mechanism [7, 9]. Therefore, less time will be spent waiting for the timer to expire and fewer faults will be happened to make useless retransmission.

Most of the papers like [7, 9] are dealing with the problem with computer science point of view. In this paper we employ system's theoretic approach and use the system identification tools to provide an improved RTT predictor. To increase the performance we have modified pervious works on the modeling of the Internet [10-13]. In contrary to the existing works, we applied recursive prediction error system identification algorithms. To investigate the performance, actual Internet data has been collected during different days of a week and also different hours of each day using specially developed software.

The paper is organized as follows; In Section II, the traditionally implemented RTO adjustment mechanism is described. Section III explains the proposed approach to the RTO adjustment mechanism, which is based on recursive system identification. In Section IV, results obtained from application of both methods to real Internet data are compared and finally, conclusion is discussed in Section V.

II. Traditional RTO adjustment Mechanism

In an ordinary TCP implementation, timeout is adjusted dynamically based on an estimate of RTT [4]. By starting a connection, TCP initialize a variable, which is considered

as the best estimate of the round-trip-time. When a segment is sent, a timer is started, both to see how long the acknowledgement takes and to trigger a retransmission if it takes too long. If the acknowledgement gets back before the timer expires, TCP measures the acknowledgement time (M), resets the timer, and updates the RTT according to the following formula.

$$RTT = \alpha RTT + (1-\alpha)M \quad (1)$$

Where α is a smoothing factor that determines how much weight is given to the previous value of the RTT . Usually timeout is determined as $\delta.RTT$.

In his famous paper about congestion control [6] in 1988, Jacobson proposed making δ roughly proportional to the standard deviation of the acknowledgement time probability density function. In particular, he suggested using the *mean deviation* as a cheap estimator of the standard deviation. His algorithm requires keeping track of another smoothed variable D , the mean deviation. Whenever an acknowledgement comes in, the difference between the expected and observed values ($|RTT - M|$) is computed and then D is updated as follows.

$$D = \beta D + (1-\beta)|RTT - M| \quad (2)$$

A typical implementation of TCP uses $\alpha = 0.875$ and $\beta = 0.75$ [14]. Most of the existing TCP implementations employ Jacobson's algorithm to determine the timeout.

$$Timeout = RTT + 4 \times D \quad (3)$$

Jacobson initially used 2 as the multiplier of D , but later works have shown that 4 gives better performance. The choice of the factor 4 minimizes unnecessary timeouts and retransmission but makes more idle time to expire the timer when a packet is lost.

III. TIMEOUT ADJUSTMENT USING SYSTEM IDENTIFICATION

A. System Approach

Consider a conventional packet transmission process over the Internet. It can be defined by three basic components: 1) the sender end-host (the source) which sends information packets one-by-one by *known rates* and receive the acknowledgement of each acknowledged packet after a measurable round-trip-time. 2) The receiver end-host (the destination) which receives the mentioned information packets and send the acknowledgement for each received packet. 3) The Internet network which is the carrier of these packet transmission with an unknown internal and dynamic behavior [15]. These components are illustrated in Fig. 1. Every thing outside the source computer is considered as our system, which is called the *Network*. Inside the source computer, TCP is dealing with packet transmission. Interaction between TCP and Network can be shown as in Fig. 2.

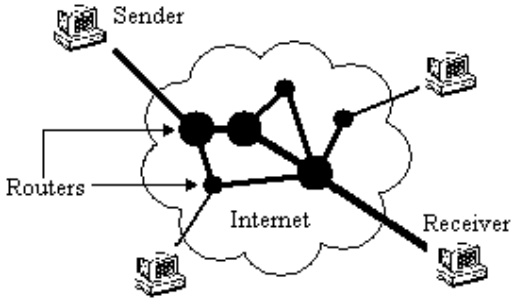

Fig. 1: Basic components of a conventional Internet transmission process.

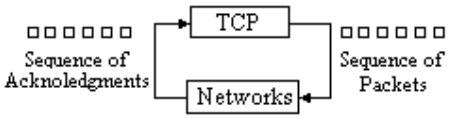

Fig. 2: Block diagram of a packet transmission process.

To determine the closed-loop model describing the interaction, the following conditions are assumed to be satisfied. 1) All the packet losses are detected by timeout. 2) All the packets are acknowledged separately. 3) All the packets have the same size.

By assumption 3, network block in Fig. 2 is excited by the sequence of packets that have fixed size but receive in different time intervals. Therefore, the time interval between two successive transmitted packets, which is called Inter-Departure Time (IDT), can be used as the network's input. When a packet is transmitted to the network it would be lost or acknowledged after its round-trip time. Now if we assume that the lost packet has an infinite round-trip time, then RTT will be the single output of the network block.

To describe the TCP behavior, type of its implementation should be known. In this work, Tahoe version [5] that is one of the predominant TCP implementations is considered. Based on Tahoe implementation and from system's theoretic point of view, a TCP transmission rate control includes the following three components.

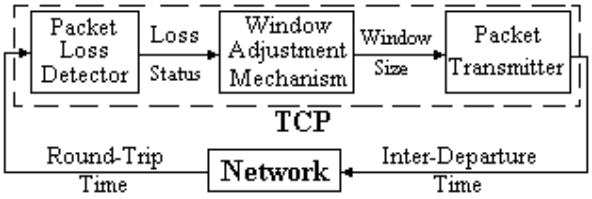

Fig. 3: Block diagram of an Internet transmission process (TCP in detail).

1) Packet Loss Detector.
2) Congestion Window Adjustment Mechanism.
3) Packet Transmitter.

Fig. 3 shows the proposed closed-loop model of described Internet packet transmission process. The most important functions of the packet loss detector block is described by equations (1), (2) and (3). Equation (1) represents the RTT predictor that works based on the previous RTT measurements. Performance of the predictor can be improved considerably if a more accurate model of the network is employed. This is discussed in the following subsection.

B. RTO Adjustment mechanism with Recursive System Identification

In [10-13], the authors tried to capture the Internet delay dynamic by a *fixed model*. They used *offline* system identification methods. In [10], a Box and Jenkins model was applied to a network structure with two source host–destination host pairs and a single bottleneck link, which was simulated on network simulator (*ns-2*). The result was good. In [11] an ARX model was applied to an *ns-2* based network simulator with ten source host–destination host pairs and a single bottleneck link. The authors used an ARX model with 31 parameters ($n_a = 18$, $n_b = 13$) to get an acceptable result. Also in [12, 13] an ARX model was applied to some different *ns-2* based simulated LANs (Local Area Network) and the result was acceptable for small structures but had flaws for more complicated ones. In [13], in addition to mentioned simulated LAN structures, the author applied an ARX model to actually collected Internet data with a data collection setup that was similar to the well-known *ping* program. The author reported that the ARX model fails to capture the round-trip time dynamics. He argued that the inefficiency is caused because of the network behavior. For a rather complex network topology with bottleneck link shared by many users, the packet transmission rate from the source to host has little impact on the round-trip time.

Generally the network's dynamic behavior is affected by two factors: First) time-invariant factors like routers' queue dynamics and links' propagation delays. Second) time-varying factors like the connection numbers that use each shared router during the routing path. Although the complicated behavior of time-invariant factors can be described by a fixed (high order) model, but these fixed models are not able to regard the second type factors. The *ns-2* based simulated structures were not complicated enough to show these aspects of the Internet behavior but the actual test in [13] was faced to this phenomena and failed.

In present study, it is assumed that the time-varying factors like the average connection establishments and disestablishments do not change with high frequency. Then it would be possible to converge to new situations with time varying model parameters. Therefore we proposed recursive prediction error system identification method with presence of the forgetting factor.
The system to be identified is the network, which is shown in Fig. 4.

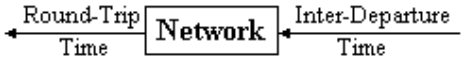

Fig. 4: Network as a system to be identified.

In this structure, each round-traveled packet (which is not dropped) can produce a data pair. Assume that each packet has a number between 1 to NP (Number of Packets) and we show the k^{th} packet with $packet(k)$. Using the following definitions;

$$\begin{aligned} s(k) &= \text{send time of } packet(k) \\ a(k) &= \text{acknowledged receiption time of } packet(k) \end{aligned} \tag{4}$$

The mentioned data pair can be calculated as follows.

$$\begin{aligned} u(k) &= s(k) - s(k-1) \\ y(k) &= a(k) - s(k) \end{aligned} \tag{5}$$

Where $u(k)$ is the inter-departure time of $packet(k)$ and $y(k)$ is its round-trip time. Therefore n pairs of $(u(k), y(k))$ are available when n packets are acknowledged. These data pairs can be used to identify the system dynamics. To take out the mean values of the variables, $\Delta u(k)$ and $\Delta y(k)$ are used in the estimation process.

$$\Delta y(k) = y(k) - y(k-1), \quad \Delta u(k) = u(k) - u(k-1) \tag{6}$$

To capture the networks dynamical behavior based on the above data pairs, a fixed ARX model structure with time-varying parameters is used. Input/output relation in this model is given as:

$$\begin{aligned} \Delta y(k) + a_1 \Delta y(k-1) + \cdots + a_{n_a} \Delta y(k-n_a) = b_1 \Delta u(k-1) \\ + \cdots + b_{n_b} \Delta u(k-n_b) \end{aligned} \tag{7}$$

Where n_a and n_b are orders of the numerator and denominator polynomials of the transfer function. Adjustable parameters in this case are:

$$\theta(k) = \left[a_1 \cdots a_{n_a} \;\; b_1 \cdots b_{n_b} \right]^T \tag{8}$$

$\theta(k)$ is determined to minimize the least squares criterion with the exponential forgetting factor. It is updated at time k as follows.

$$\theta(k) = \theta(k-1) + L(k)\left[\Delta y(k) - \varphi^T(k)\theta(k-1)\right] \quad (9a)$$

$$L(k) = \frac{P(k-1)\varphi(k)}{\lambda + \varphi^T(k)P(k-1)\varphi(k)} \quad (9b)$$

$$P(k) = \frac{1}{\lambda}\left[P(k-1) - \frac{P(k-1)\varphi^T(k)\varphi(k)P(k-1)}{\lambda + \varphi^T(k)P(k-1)\varphi(k)}\right] \quad (9c)$$

$$\varphi(k) = [-\Delta y(k-1)\ -\Delta y(k-2)\cdots -\Delta y(k-n_a)\ \ \Delta u(k-1)\ \Delta u(k-2)\cdots \Delta u(k-n_b)]^T \quad (9d)$$

λ is the forgetting factor. The most recent data is given unit weight, but data that is n time units old is weighted by λ^n. For higher λ, less weight is given on the previous measurements. Generally λ is selected between 0.95 and 0.999. $\lambda = 0.98$ was chosen in the present work. Recursive prediction error method is introduced in detail in [16, chapter 11].

Based on the estimated parameters, output variation can be predicted as follows.

$$\Delta\hat{y}(k+1) = -\hat{a}_1\Delta y(k) - \cdots - \hat{a}_{n_a}\Delta y(k-n_a+1) + \hat{b}_1\Delta u(k) + \cdots + \hat{b}_{n_b}\Delta u(k-n_b+1) \quad (10)$$

The network output is then determined using Equation (6).

$$\hat{y}(k+1) = y(k) + \Delta\hat{y}(k+1) \quad (11)$$

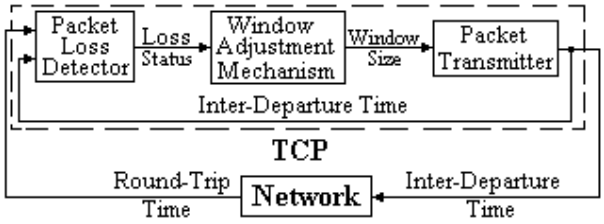

Fig. 5: Proposed closed loop structure.

Now if we substitute the proposed RTT predictor with conventional predictor in Equation (1), a new RTO adjustment mechanism is introduced. The new closed loop structure is illustrated in Fig. 5.

IV. Comparisons Based on Actual Internet Data

A. Data Collection

The well-known *ping* program uses ICMP (Internet Control Message Protocol) *Echo Request* and *Echo Reply* messages. Some network devices limits the rate of ICMP packets, therefore to collect actual Internet data pairs, we developed specific data-collector software. The software comprises two parts, one in the source computer and another in the destination computer. These two parts, which are in the application layer, manage all the transmissions. We transmitted User Datagram (UDP) packets. Unlike the TCP packets, the flow control process of UDP packets is not affected by the operating system. Therefore in our test, we were able to observe exact RTT and IDT values and amount of lost packets directly. Source and destination computers were connected with dial up links to two separate famous private Internet Service Providers (ISP) in Tehran. Both of them use satellite links. Therefore there was a complicated routing path with 1357 msec as the average packet RTT. In the source part, a very simplified two-phase window-based transmission rate adjustment mechanism was implemented. There was no retransmission mechanism. Our simplified window-based algorithm enables us to collect the RTT and the IDT as well as the window-size. We used this software to collect 10 data sets during different days from August 7, 2003 to August 11, 2003 and different hours at each day. Each data set was collected by 5000 packet transmissions therefore totally 50000 data were collected.

B. Comparison Criteria

In section I, it was stated that an ideal RTO minimizes the useless retransmissions and idle times as much as possible. Therefore we used *overall useless retransmissions* and *average retransmission idle time* as comparison criteria. When we applied each RTO adjustment algorithm to a data set, these two metrics were calculated and used as RTO performance comparison criteria. Another important comparison is related to the RTT prediction capabilities. Therefore *mean absolute prediction error* was used as the third criterion.

C. Result

We applied the recursive parameter estimation approach in Equation (9) to each data set and determined the parameters of an ARX model structure. Table 1 shows the results of the proposed approach when the model orders are $n_a = 1$, $n_b = 1$, $n_k = 1$.

Our observations are summarized in the following remarks:

Remark 1: If we use congestion-window size as the model input, system identification will come up to an inaccurate result therefore it is important to use IDT as input signal.

Remark 2: The network has strong time varying property and therefore simple model (model with less parameters) indicate better performance in coping with the network dynamics. Another benefit of the simple models is that they require less time to be updated, which is an important issue in a transmission related algorithm.

Remark 3: Recursive system identification algorithms have transient behaviors to capture the dynamics of the network. Therefore, all the comparison's metrics were calculated after 500 steps (from packet (501) to packet (5000)).

Remark 4: Useless retransmission and *idle time* criteria are in opposite directions. It is possible to violate one of them

to get better performance in another one. Therefore they should be considered together to show the performance. The balanced improvement in both of them is possible by improving the RTT prediction.

Table 1: These improvements were observed for each data set.

(1)	(2)	(3)	(4)	(5)	(6)
1	1342	115	27.9%	39.5%	27.5%
2	1339	58	23.8%	46.9%	23.6%
3	1607	420	50.9%	27.5%	50.3%
4	1434	272	18.8%	67.3%	18.1%
5	1248	160	32.5%	0%	32.7%
6	1326	78	19.4%	55.6%	19.0%
7	1358	84	25.1%	29.7%	26.0%
8	1370	787	1.6%	12.9%	1.8%
9	1196	467	25.0%	29.5%	24.8%
10	1352	150	26.5%	28.1%	25.9%
Mean	1357	259	25.2%	33.7%	25%

(1) Data set, **(2)** Average RTT (msec), **(3)** Standard deviation of RTT changes (msec), **(4)** Improvement in RTT prediction, **(5)** Avoided useless retransmission, **(6)** Reduce retransmission idle time

V. Conclusion

In this paper we employed the system approach to TCP retransmission timer adjustment mechanism. Recursive system identification algorithm used to cope with the time varying behavior of the Internet. A simple system identification based RTT predictor was substituted for the traditional predictor in the RTO adjustment mechanism. This includes a new closed-loop model to illustrate the TCP-Network interaction. The proposed RTO adjustment algorithm was applied to several actual Internet data. It was observed that the proposed method could improve the RTT prediction quality by 25.2 %, reduce the useless retransmissions by 33.7 % and the retransmission idle times by 25%.

Acknowledgment

This work was supported financially by Sharif University of Technology and Iranian Telecommunication Research Center (ITRC). Authors hereby sincerely appreciate their support.

References

[1] S. McCreary, K. Claffy, "Trends in Wide-Area IP Modeling, Measurement and Management", ITC Specialist Seminar on IP Traffic Modeling, Measurement and Management, Monterrey, CA, September 2000.

[2] H. Balakrishnan, V. Padmanabhan, S. Seshan, M. Stemm, M. Katz, "TCP Behavior of a busy Internet Server: Analysis and Improvements", Proc. IEEE INFOCOM 1998.

[3] D. Lin, H.T. Kung, "TCP Fast Recovery Strategies: Analysis and Implementations", Proc. IEEE INFOCOM 1998.

[4] A.S. Tanenbaum, *Computer Networks*, 4th Edition, Prentice Hall, NJ, 2003.

[5] S. H. Low, F. Paganini, J. C. Doyle, "Internet Congestion Control", IEEE Control Systems Magazine, pp. 28-43, Feb. 2002.

[6] V. Jacobson, "Congestion Avoidance and Control", Proc. ACM SIGCOMM, pp. 314-329, 1988.

[7] R. Ludwig, K. Sklower, "The Eifel Retransmission Timer", ACM Computer Communication Review, Vol. 30, Issue 3, Pages 17-27, July 2000.

[8] N. Seddigh, "Performance Analysis of TCP's Retransmission Timeout Mechanism", Master Thesis, Department of Systems and Computer Engineering, Carlton University, Canada, Dec 2000.

[9] M. Allman, V. Paxon, "On Estimating End-to-end Network path properties", ACM Computer Communication Review, Vol. 31, Issue 2-Suppliment, April 2001.

[10] L. B. White, "Internet Transport Layer System Identification", Proc. IEEE Int. Symp. Statistical Signal Processing, Singapore, August 2001.

[11] H. Ohsaki, M. Murata, H. Miyahara, "Modeling End-to-End Packet Delay Dynamics of the Internet Using System Identification", Proc. International Teletraffic Cong. 2001.

[12] H. Ohsaki, M. Morita, M. Murata, "On Modeling Round-Trip-Time Dynamics of the Internet Using System Identification", IEICE Transactions on Communications, Vol. E85-B, No. 1, Jan. 2002.

[13] M. Morita, "Studies on Modeling Packet Delay Dynamics of the Internet Using System Identification and Its Application for Designing a Rate-Based Congestion Control Mechanism", Master Thesis, Department of Informatics and Mathematical Science, Osaka University, Japan, Feb. 2002.

[14] M. Allman, V. Paxon, "On Estimating End-to-end Network Path Properties" Proc. ACM SIGCOMM, pp 263-274, 1999.

[15] V. Paxon, "End-to-End Internet Packet Dynamics", IEEE/ACM Tran. Networking, Vol. 7, No. 3, pp. 277-292, 1999.

[16] L. Ljung, *System Identification: Theory for the User*, Prentice Hall, Upper Saddle River, N.J. 2nd edition, 1999.