TCP Retransmission Timeout Algorithm Using Weighted Medians

Liangping Ma, *Student Member, IEEE*, Gonzalo R. Arce, *Fellow, IEEE*, and Kenneth E. Barner, *Senior Member, IEEE*

***Abstract*—This letter presents a new retransmission timeout (RTO) algorithm based on recursive weighted median (RWM) filters for the transmission control protocol (TCP). The RTO algorithm utilized in current TCP implementations is Jacobson's algorithm [1], which is based on recursive linear filtering. While linear filters are adequate for estimation in Gaussian signal environments, the round trip time (RTT) signals filtered to determine the RTOs are often impulsive. Thus, Jacobson's algorithm is not effective in many cases. The proposed algorithm employs RWM filters that yield improved performance when operating on RTT signals with heavy tailed statistics. Simulation results show that the proposed method yields significantly tighter RTT bounds than Jacobson's method over Internet traffic with heavy tailed statistics.**

***Index Terms*—Recursive weighted median filters, retransmission timeout (RTO), round trip time (RTT), transmission control protocol (TCP).**

I. Introduction

TRANSMISSION control protocol (TCP) provides reliable logical communications for applications (e.g., FTP, telnet, and HTTP) running on different hosts on the Internet [2, p. 82]. It is critical for TCP to have an accurate retransmission timeout (RTO) algorithm, which greatly affects data transfer delay and congestion control [3, p. 467]. An RTO algorithm assigns a timeout to each outgoing packet based on the past RTT observations, where the RTT is the time it takes for a packet to travel from client to server and then back to the client [2, p. 88]. The desired RTO should be slightly larger than the corresponding RTT [1], [2, p. 224].

The RTO algorithm used in current TCP implementations was proposed by Jacobson [1]. More recently developed RTO algorithms include the Eifel Retransmission and the jitter-based algorithms [4], [5]. The Eifel algorithm was heuristically designed based on an assumed set of RTT characteristics, which may not hold for all network configurations. The jitter-based algorithm utilizes the fractal-like structure of the RTT processes, but a table has to be pre-configured for each Internet path making it unsuitable for real time applications.

This letter proposes a new RTO algorithm based on recursive weighted median (RWM) filtering. The RWM algorithm yields better performance than Jacobson's algorithm and is simple enough to allow real time applications. The RWM approach is motivated by the following facts. First, Jacobson's algorithm is not optimal. This algorithm is based on linear filtering, which provides adequate performance when the underlying signals exhibit Gaussian statistics. However, the RTT statistics are often impulsive [6], and cannot be accurately modeled as a Gaussian process, thus leading to poor RTO performance of Jacobson's algorithm. The impulsive characteristic of the RTT is clearly shown in Fig. 1. Second, weighted median (WM) filters are optimal for signals obeying the heavy tailed Laplacian distribution [7], [8]. Accordingly, WM filters yield improved performance when filtering signals with heavy tailed distributions. A rich theory is available for the design and optimization of WM filters [7], [8]. Utilizing the robust characteristics of the WM filtering, the proposed RWM algorithm yields RTOs that are less susceptible to rapid variations on the data.

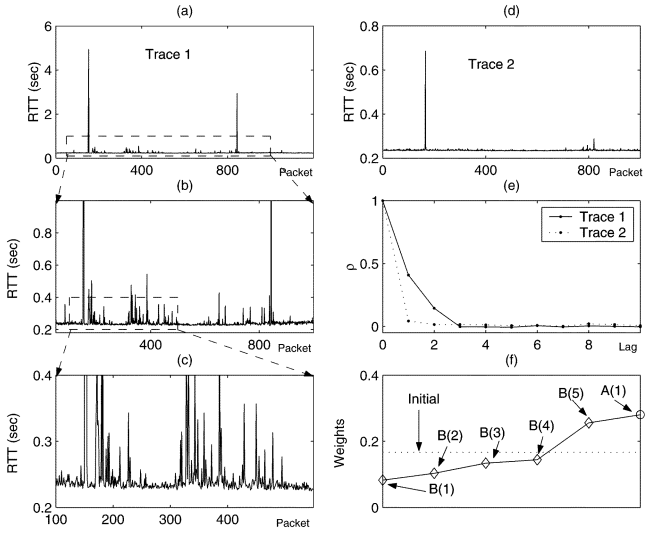

Fig. 1. Examples of RTT traces. Trace 1 (a) is for the connection pogo.udel.edu—clock.nc.fukuoka.jp, and trace 2 (d) is for the connection pogo.udel.edu—tictoc.tip.csir.au, both measured on April 18, 2001. A probe packet was sent every 60 s. (b) Zoom-in of (a), and (c) zoom-in of (b). Sample autocorrelation coefficients for both traces are depicted in (e), and the optimized weights for the RWM filter for trace 2 are illustrated in (f).

The remainder of this letter is organized as follows. In Section II, Jacobson's algorithm is discussed and the RWM algorithm is proposed in Section III. Simulation results and some comments are given in Section IV.

II. Jacobson's RTO Algorithm

In Jabobson's algorithm, the RTT estimate is obtained by linearly filtering the previous RTTs and the RTO is then set as the

Manuscript received July 10, 2003; revised October 10, 2003. This work was supported in part by the National Science Foundation ITR-ANI under Grant 0312851. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Dimitris A. Pados.

The authors are with the Department of Electrical and Computer Engineering, University of Delaware, Newark, DE 19716 USA (e-mail: lma@ece.udel.edu; arce@ece.udel.edu; barner@ece.udel.edu).

Digital Object Identifier 10.1109/LSP.2004.827957

RTT estimate plus a scaled mean absolute deviation [1]. Specifically, if all the RTTs up to that of packet $i-1$ are observed, the RTT estimate for packet i is calculated by

$$a_i = (1-g_1)a_{i-1} + g_1 m_{i-1} \tag{1}$$

where m_{i-1} is the RTT observation for packet $i-1$, a_{i-1} is the RTT estimate for packet $i-1$, and the constant $g_1 = 1/8$ is generally used. The mean absolute deviation of the RTT estimation is then calculated by

$$v_i = (1-g_2)v_{i-1} + g_2|a_{i-1} - m_{i-1}| \tag{2}$$

where v_{i-1} is the mean absolute deviation for packet $i-1$, and the constant $g_2 = 1/4$ is generally used. Finally, the RTO for packet i is set as the RTT estimate a_i plus a scaled mean absolute deviation of the RTT,

$$r_i^J = a_i + kv_i \tag{3}$$

where the superscript J indicates that the RTO is formed by Jacobson's algorithm, and the scale factor $k = 4$ is generally used. This development shows that the RTO is the result of a sequence of linear operations. The inefficiency of linear filters in non-Gaussian signal estimation is well known [7]. The resulting RTO is thus severely degraded by outlier data in the RTT sequence, as is illustrated in Fig. 2(a), (c), and (d) and Fig. 3(a) and (c).

III. Proposed RTO Algorithm

The proposed algorithm operates in a fashion conceptually similar to Jacobson's: the RTT is estimated and the RTO is set as a scaled version of the estimated RTT. The details of the RWM algorithm are presented below along with a discussion on implementation issues.

A. Estimation of RTT

Suppose again that the RTTs of all the packets up to packet $i-1$ are observed, then the RTT estimate for packet i is given by

$$\begin{aligned} a_i &= \text{MEDIAN}(A_1 \diamond a_{i-1}, \ldots, A_N \diamond a_{i-N}, \\ &\qquad B_1 \diamond m_{i-1}, \ldots, B_M \diamond m_{i-M}) \qquad (4) \\ &= \text{WM}\left(\left[a_{i-\ell}|_{\ell=1}^{N}, m_{i-j}|_{j=1}^{M}\right], \mathbf{W}^T\right) \qquad (5) \end{aligned}$$

where m_{i-j} and $a_{i-\ell}$ are defined in (1), N is the number of previous RTT estimates considered, M is the number of previous RTT observations considered, $\mathbf{W}$ is the weight vector defined as $\mathbf{W} = [A_1, A_2, \ldots, A_N, \underbrace{B_1, B_2, \ldots, B_M}_{W_j \text{times}}]^T$, and $\diamond$ is the replication operator, $W_j \diamond X_j = \overbrace{X_j, \ldots, X_j}$. A detailed description and analysis of RWM filters and their computation for real-valued weights can be found in [7], [8].

To implement this method, the window size and the weights must be determined. First, note that utilizing even a single recursive term allows information on past outputs to be captured. In simulations, increasing the number of recursive terms beyond one does not significantly improve performance, and we therefore set $N = 1$. Similarly, simulations show that utilizing five feed-forward terms is sufficient to capture the correlations between RTT samples, and we set $M = 5$.

Consider next the design of filter weights. The optimal weights can be obtained through adaptive optimization procedures [8]. For example, the optimal weights for trace 2, depicted in Fig. 1(d), are shown in Fig. 1(f), where it can be seen that the weight decreases exponentially as a function of lag. However, optimizing the weights is not feasible in real time applications. First, the need of many RTT observations in the iterative optimization procedure leads to intolerable delay. Second, if the signal is highly nonstationary, which is often the case with Internet traffic, convergence may not be obtained. Thus, we opt to utilize a fixed set of weights that provide good robust performance across a wide range of RTT statistics. As

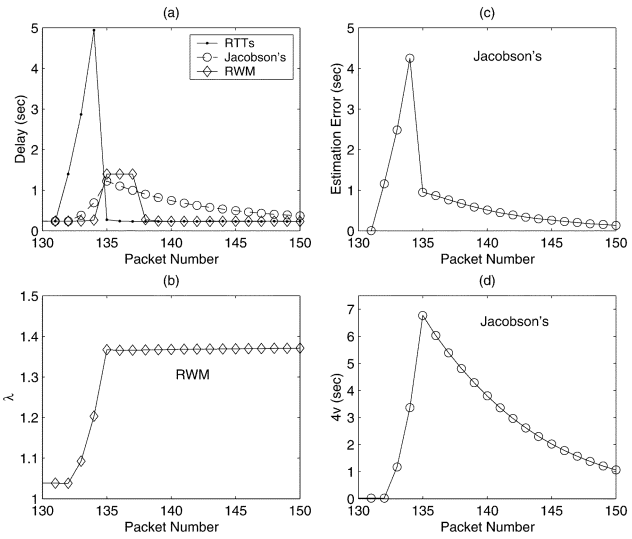

Fig. 2. Illustration of the two algorithms applied to a segment of trace 1 containing an outlier: (a) RTT estimates, (b) scale factor λ in the RWM algorithm, (c) mean absolute error in RTT estimation in Jacobson's algorithm, and (d) scaled mean absolute deviation in Jacobson's algorithm.

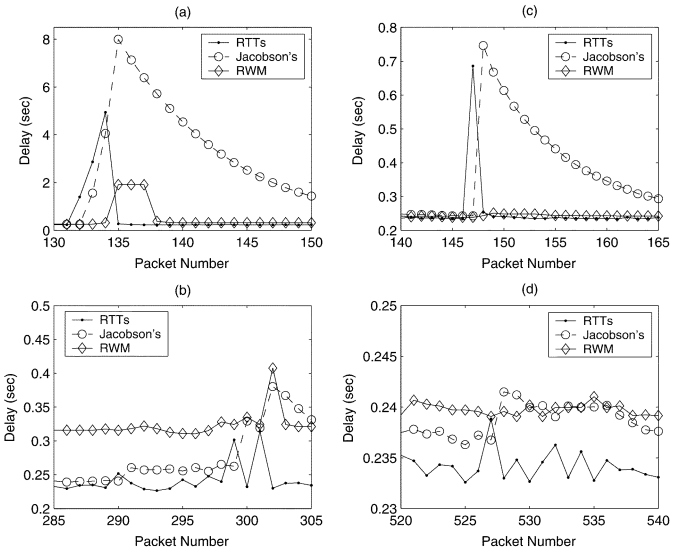

Fig. 3. Experimental results: the RTOs for (a) an impulsive region and (b) a nonimpulsive region in trace 1, and the RTOs for (c) an impulsive region and (d) a nonimpulsive region in trace 2. Although Jacobson's algorithm yields tighter RTOs for (b) and (d), it generates failed RTOs (estimates are lower than the RTTs) for some packets.

experiments indicate that the correlation between RTT samples, as shown in Fig. 1(e), drops rapidly with lag, we utilize weights that emphasize the relative importance of samples with small lag. An exponential function may be used for this purpose. Thus, the weights can be expressed as

$$A(1) = \beta, \quad B(j) = \alpha^{j-1} \tag{6}$$

for $j = 1, \ldots, M$, where $0 < \alpha < 1$, $\beta > 0$. A necessary requirement imposed on the choice of the parameters α and β is that no single weight accounts for more than one half of the sum of all the weights. Otherwise, the output will be always the over weighted sample [7], [8]. These parameters can be adjusted using the fact that increasing β allows more information on past outputs to be captured while increases in α result in more smoothing on RTTs. Experimentation shows that the values $\beta = 1/2$ and $\alpha = 7/8$ provide good performance for a wide range of network conditions.

B. Determination of RTO

Once an estimate of the RTT is established, there are two common methods for forming the RTO value. One method adds a bias to the estimated RTT, where this bias represents the uncertainty or variability in the RTT. This is the approach adopted by Jacobson. Alternatively, the estimated RTT can be scaled, where the scale factor reflects the RTT variability. We have found that the second approach, when utilized in conjunction with the recursive WM RTT estimates, produces the best results. Let the scale factor be denoted as

$$\lambda = 1 + \gamma \tag{7}$$

where $\gamma > 0$. To measure the variability, we can utilize the ratio of the standard deviation to the mean [9]. Considering that the standard deviation for impulsive signals leads to unreliable statistics, we replace the standard deviation with the mean absolute deviation, which is less affected by RTT outliers. The mean absolute deviation to the mean is thus defined as

$$\zeta = \frac{E[|m - E[m]|]}{E[m]}. \tag{8}$$

The ratio ζ is estimated from the accumulated statistics, which may be updated as the network conditions change. To be proportional to the variability in the RTT, the scale factor λ can be expressed as

$$\lambda = 1 + \mu\zeta \tag{9}$$

where μ is a scale parameter that can be determined empirically.

Finally, we multiply the RTT estimate a_i by λ and set the product as the RTO,

$$r_i^M = \lambda a_i \tag{10}$$

where the superscript M indicates that the RTO is formed through the proposed median algorithm. In order to fix the value of the scale parameter μ in (9), we set $E\left[r_i^M\right] = E\left[r_i^J\right]$ in simulations, and find that μ varies in the range $[4.3, 4.7]$. This small range indicates that a fixed μ (e.g., 4.5) can be used in practice.

C. Implementation Issues

The memory requirements and computational complexity of any new RTO algorithm must be considered. The RWM operation requires the storage of the input samples, their ranks, associated weights, and the weight-based threshold [8]. Additionally, $E[m]$, $E[|m - E[m]|]$, μ, λ, a_i, r_i^M must be stored. This results in a total of $(3M + 3N + 7)$ values that must be stored, or 25 terms for the suggested parameters $N = 1$ and $M = 5$. In contrast, Jacobson's algorithm must store 7 terms. Thus, the additional memory required by the RWM algorithm is minimal.

The proposed algorithm utilizes a running recursive WM filter. In this case one new sample and one recursive sample enter the window at each update. This reduces the required computations to, at most, $(\lceil \log(M+N-2) \rceil + \lceil \log(M+N-1) \rceil)$ comparisons and $(M + N - 1)$ additions and threshold operations. Thus, for the typical case $N = 1$ and $M = 5$, each RTO update will require at most, 6 comparisons, and 5 additions/threshold operations. Thus the computational load of the proposed algorithm is modest and in line with that required by Jacobson's algorithm.

It should be also noted that the RWM algorithm is suitable for short TCP connections. For $M = 5$, $N = 1$, only 5 RTT samples are needed to produce the RTO, assuming that the previous output is set to an initial value. If the number of RTT samples is less than 5, the RTO estimate can be set heuristically. The difficulty of producing reliable RTOs with insufficient RTT samples also exists in Jacobson's algorithm.

Several practical factors affect the performance of the RWM algorithm. In current TCP implementations, an RTO is set to the maximum of the computed RTO value and a minimum RTO, where the unit of RTO is in clicks and the click is referred to as the timer granularity [10].[1] It has been noted that the performance of Jacobson's algorithm is dominated by the minimum RTO and, to a lesser extent the timer granularity. Factors such as parameter setting have less impact on performance [10]. This might also hold for the RWM algorithm. In addition, if the RTOs produced by the RWM and Jacobson's algorithm are below the minimum RTO, then the resulting RTOs will be the same, eliminating the advantage of the RWM algorithm. The same result follows if two different RTOs are rounded to the same coarse timer granularity. But for TCP implementations with a small minimum RTO and a finer timer granularity, the RWM algorithm has clear advantages.

IV. Experimental Results and Discussions

There are two approaches to testing RTO algorithms that can be adopted: simulations using softwares like *ns* [11], and experiments based on Internet RTT traces. Due to the immensely heterogeneous nature of the Internet [12], it is difficult to set up a simulation such that the resulting RTT traces exhibit the statistical properties observed in the Internet RTT data. Thus, we adopt the second approach for the testing.

For performance comparisons, one can either fix the mean RTO and compare the correct prediction probabilities (the probability that an RTO is larger than the corresponding RTT)

[1]The timer granularity of BSD is 0.5 s.

produced by the algorithms, or fix the correct prediction probabilities and compare the resultant mean RTOs. The algorithm producing a higher correct prediction probability or a lower mean RTO is considered better [6]. In the experiments presented here, we utilize the results of Jacobson's algorithm as a benchmark and set μ so that the two algorithms produce the same correct prediction probabilities. The parameters $\alpha = 7/8$ and $\beta = 1/2$ are used for the RWM algorithm and the data used in the experiments are available online [13]. A sampling period of 60 s is used, which is a reasonable period for two reasons. First, it is of the same order as the median (about 15 s) of the *user think time* (the duration between two clicks on a Web page) for Web traffic, which is the leading Internet traffic [14]. TCP is largely idle between clicks and does not take RTT samples. Second, TCP implementations in general take a RTT sample at most once per RTT rather than once per packet [3, p. 468]. This sampling strategy makes the Web traffic TCP sample the RTT at each click, after which there is a pause until the next click. Thus, the Web traffic TCP takes RTT samples about once per *user think time*.

The experimental results for the two RTT sequences depicted in Fig. 1 are shown in Fig. 3, which illustrates that the RTO produced by Jacobson's algorithm is susceptible to RTT outliers, as the RTOs following each RTT outlier decay very slowly. If there is a packet loss immediately after an RTT outlier, the timeout will occur much later than it should. In contrast, the RWM algorithm is more robust and yet quickly follows the trend of the RTT. As a result, the RWM algorithm outperforms Jacobson's algorithm in impulsive regions, as is shown in Fig. 3(a) and (c). The advantages of the proposed algorithm over Jacobson's in impulsive regions are due to the optimality of the WM filter for samples with such heavy tailed distributions. The two algorithms do not differ significantly in nonimpulsive regions, as is shown in Fig. 3(b) and (d), where Jacobson's algorithm yields tighter RTOs but fails in some locations. Therefore, the RWM algorithm gives overall performance that is superior to Jacobson's. This can be seen in the numerical simulation results presented in Table I, which show that the RWM algorithm achieves a 19.3% reduction in the mean absolute error (MAE) in the estimation of the RTT and an 80.5% reduction in the mean RTO, when applied to the RTT series depicted in Fig. 3(a). Experiments over a total of 10 traces show that an average reduction of 16.8% in the MAE in the estimation of the RTT, and an average reduction of 7.9% in the mean RTO, are obtained by using the RWM algorithm.

TABLE I
COMPARISON OF THE RWM ALGORITHM WITH JACOBSON'S ALGORITHM UNDEREQUAL CORRECT PREDICTION PROBABILITIES

Traces	Regions	MAE in Estimation of RTT (sec)			Mean RTO (sec)		
		Jacobson's	RWM	Reduction (%)	Jacobson's	RWM	Reduction (%)
trace 1	Fig. 3 (a)	0.7091	0.5720	19.3	2.8693	0.5583	80.5
	Fig. 3 (b)	0.0135	0.0149	-9.4	0.2686	0.3373	-25.6
	overall[2]	0.0264	0.0212	19.7	0.3551	0.3047	14.2
trace 2	Fig. 3 (c)	0.0395	0.0240	39.2	0.3716	0.2447	34.2
	Fig. 3 (d)	0.0011	0.0013	-18.2	0.2383	0.2406	-0.97
	overall[2]	0.0021	0.0019	9.5	0.2442	0.2400	1.8

[2] The RTT data in Fig. 3(a) and (b) constitutes only part of trace 1, and similarly the data in Fig. 3(c) and (d) makes up only part of trace 2.

In summary, the RWM timeout algorithm outperforms Jacobson's timeout algorithm, especially for RTTs with heavy tailed statistics.

REFERENCES

[1] V. Jacobson, "Congestion avoidance and control," in *Proc. SIGCOMM'88*, Stanford, CA, Aug. 1988, pp. 314–329.

[2] J. Kurose and K. Ross, *Computer Networking: A Top-Down Approach Featuring the Internet*. Boston, MA: Addison-Wesley, 2001.

[3] L. L. Peterson and B. S. Davie, *Computer Networks: A System Approach*, 2nd ed. San Francisco, CA: Morgan Kaufmann, 2000.

[4] R. Ludwig and K. Sklower, "The Eifel retransmission timer," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 30, pp. 17–27, July 2000.

[5] Q. Li and D. L. Mills, "Jitter-based delay-boundary prediction of wide-area networks," *IEEE/ACM Trans. Networking*, vol. 9, pp. 578–590, Oct. 2001.

[6] Q. Li, "Delay Characterization and Performance Control of Wide-Area Networks," Ph.D. dissertation, Univ. Delaware, Newark, 2000.

[7] L. Yin, R. Yang, M. Gabbouj, and Y. Neuvo, "Weighted median filters: A tutorial," *IEEE Trans. Circuits Syst.*, vol. 43, pp. 157–191, Mar. 1996.

[8] G. R. Arce and J. Paredes, "Recursive weighted median filters admitting negative weights and their optimization," *IEEE Trans. Signal Processing*, vol. 48, pp. 768–779, Mar. 2000.

[9] J. Lee and I. Jurkevich, "Speckle filtering of synthetic aperture radar images: A review," in *Remote Sensing Review*. New York: Harwood, 1994, vol. 8, pp. 313–340.

[10] M. Allman and V. Paxson, "On estimating end-to-end network path properties," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 29, pp. 263–274, Oct. 1999.

[11] [Online] Available: http://www.isi.edu/nsnam/ns.

[12] V. Paxson and S. Floyd, "Why we don't know how to simulate the internet," in *Proc. Winter Simulation Conf.*, 1997, pp. 1037–1044.

[13] [Online] Available: ftp://ftp.udel.edu/pub/ntp/ntpstats/pogo/primary/rawstats.20010417.

[14] B. A. Mah, "An empirical model of HTTP network traffic," in *Proc. INFOCOM*, 1997, pp. 592–600.