Estimating Retransmission Timeouts in IP-Based Transport Protocols

Stan McClellan
Ingram School of Engineering
Texas State University
San Marcos, TX USA
stan.mcclellan@txstate.edu

Wuxu Peng
Dept. of Computer Science
Texas State University
San Marcos, TX USA
wuxu@txstate.edu

***Abstract*—This paper analyzes the algorithm used for estimating retransmission timeouts in connection-oriented IP-based transport protocols, such as the Transmission Control Protocol (TCP) and the Stream Control Transmission Protocol (SCTP). The estimation algorithm uses historical values of the round-trip time to estimate future round-trip delays, and so creates a maximum waiting time before triggering retransmission attempts. The purpose of the analysis is to question / validate some of the fundamental assumptions used in the estimation algorithm. The conclusion of the analysis is that the algorithm is somewhat mismatched to the application area. Alternative algorithms are discussed, and potential modifications are presented.**

***Keywords*-SCTP; retransmission timeout; round-trip time; RTT; RTO; Jacobson algorithm; Chebyshev approximation; parameter estimation; upper bound.**

I. INTRODUCTION

IP-based transport protocols such as the Transmission Control Protocol (TCP) [1] and the Stream Control Transmission Protocol (SCTP) [2], [3] estimate maximum round-trip times using data from prior successful transmissions. The purpose of this estimation process is to create a triggering mechanism for retransmission procedures when transmissions are lost or seriously delayed. Estimation of the maximum round-trip time is performed via the Jacobson algorithm [4], which is codified in several IETF RFC's, including RFC 6298 [5]. The Jacobson algorithm has an interesting basis in fundamental theory, but suffers from some performance issues due to a mismatch between the theory and the application area. Performance issues related to the Jacobson algorithm and other retransmission procedures have been noted and addressed in several alternative approaches, including [6]–[11]. This paper discusses the Jacobson algorithm, the theory which motivates it, and several alternative algorithms including a new approach which is a modified form of Jacobson. Section II describes the estimation process and its use in establishing timeouts for retransmission procedures. Related work in retransmission optimization and timeout boundaries is also summarized. In Section III, the parameters of the existing algorithm are analyzed, and in Section IV an alternative approach is presented based on similar theoretical concepts, and achieving improved results. Performance results based on implementation and simulation are summarized in Section V, and Section VI concludes the paper. Note that much of this work is presented in the context of SCTP, but is also applicable to TCP since the timeout estimation processes are identical.

II. RETRANSMISSION MECHANISMS

When an SCTP sender transmits a unit of data, called a *chunk*, it also initializes a *retransmission timer* with an estimated value of the round-trip time (RTT). The value of this timer is the *retransmission time-out* (RTO). When an acknowledgment arrives, the timer is cancelled. If the timer expires before an acknowledgment arrives, the chunk may be retransmitted. The value of RTO is calculated from observed / actual values of RTT using the *Jacobson Algorithm*, which is detailed in Section III. A too optimistic retransmission timer may expire prematurely, producing *spurious timeouts* and *spurious retransmissions*, reducing a connection's effective throughput. On the contrary, a retransmission timer that is too conservative may cause long idle times before lost packets are detected and retransmitted. This can also degrade performance [6]. So, the difficulty lies in finding an algorithm which has a solid theoretical basis, is not computationally expensive, and can predict RTT with enough bias to minimize retransmission events and waiting time *simultaneously*.

Performance issues related to retransmission procedures, including alternatives to the Jacobson Algorithm, have been noted and addressed several times in the literature. Much work has been focused on late retransmission and other optimizations of the overall retransmission scenarios [12], [13]. Many authors approach this problem with a "holistic" or overall perspective on the retransmission procedures where RTT estimation contributes to triggering these procedures. Other authors specifically address the estimation of RTT, and propose completely new algorithms. However, the Jacobson Algorithm is deeply rooted in the fabric of connection-oriented IP transport protocols, and its basis in fundamental theory is well-established. In following subsections, we briefly summarize several important approaches to retransmission and RTO estimation.

 ISBN: 978-1-61208-262-2

A. Holistic approaches

Holistic or overall approaches to improving retransmission performance typically address the state machines surrounding the retransmission process, including the estimation algorithm which may be used to trigger these procedures. Examples of holistic approaches include *Early Fast Retransmit* (EFR) [10], *Early Retransmit* (ER) [11], and *Window-Based Retransmission* (WB-RTO) [14] as well as protocol-specific techniques such as *Thin Streams* [9], [10], [15].

Early Fast Retransmit (EFR) is an optional mechanism in FreeBSD which is active whenever the congestion window is larger than the number of unacknowledged packets, and packets remain to be sent. When the RTO timer expires and the entire congestion window is not used, EFR retransmits all packets that could have been acknowledged [10]. The *Early Retransmit* (ER) algorithm [11] suggests that a mechanism should be in place to recover lost segments when there are too few unacknowledged packets to trigger Fast Retransmit. The Early Retransmit algorithm reduces waiting time in four specific situations [10]. The *Window-Based Retransmission Timeout* (WB-RTO) [14] asserts that timeout mechanisms based solely on RTT estimates lead to unnecessary retransmissions and unfair resource allocation, and proposes to schedule flows on the basis of their contribution to congestion. *Thin Streams* [9], [10], [15] optimizes throughput for "thin streams" which are often used in control applications, and often depend on SCTP for transport. When stream characterization is accurate, throughput can be improved by adapting specific sections of the retransmission procedures to match flow characteristics.

B. Alternative estimation algorithms

Alternative estimation algorithms address specific performance issues which have been noted in the Jacobson Algorithm. These issues may be related to overshoot in the estimated value, spurious behaviors for certain traffic characteristics, or inefficient bounding computations. In some cases, heuristic state-machine approaches are also included because of complexities associated with the retransmission process. Examples of alternative or modified RTT estimation algorithms include *Peak-Hopper* [7], *Eifel* [6], and *Weighted Recursive Median* (WRM) [8].

The *Eifel* approach [6] notes a particular style of erroneous performance in the Jacobson algorithm, and adapts the algorithm in several ways to compensate for this performance oddity. As a result, Eifel eliminates unnecessary retransmissions which can result from spurious RTO violations. Similar to Eifel, the *Peak-Hopper* algorithm [7] also observes that the Jacobson algorithm responds inappropriately to certain fluctuations in RTT. This behavior produces "spikes" in RTO values because the algorithm does not distinguish between positive and negative variations. The modification proposed in [7] reduces this effect for a wide range of cases, and the findings in [9] concur. However,

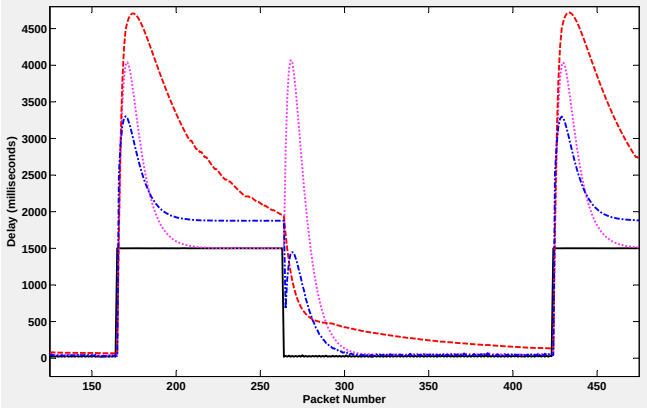

Figure 1. Relative performance of RTO estimators to large discontinuities in RTT. Estimators shown are the Jacobson Algorithm (dotted line), the Eifel Algorithm (dashed line), and the Modified Jacobson Algorithm (dash-dot line). The common RTT sequence driving all algorithms is shown as a solid line.

this solution results in higher average RTO values than the RFC approach [5], which can be a problem [10]. The *Weighted Recursive Median* (WRM) algorithm [8] redefines RTT estimation from a signal processing standpoint. WRM is effective, but tends to be computationally expensive even in a recursive form, which is a problem for per-packet operations.

C. Other considerations

The remainder of this paper addresses the estimation process for the maximum RTT, or the value which establishes the RTO timer. When the RTO timer has expired, retransmission procedures commence, and may include various conditionally executed processes. We do not address those processes, or the overall retransmission procedure. In most cases, we use the Jacobson and Eifel estimation algorithms for comparison because they are widely accepted or implemented.

For reference, the performance of the Jacobson and Eifel algorithms are shown in Figure 1 along with the modified version of the Jacobson Algorithm which is discussed in detail in Section IV. The traces in the figure are all driven by a common RTT sequence created using our testbed of systems with modified networking stacks, as described in more detail in Section V. In the figure, the quiescent sections of the RTT sequence ($t < 170$ms, $260\text{ms} < t < 430$ms) have a fairly low mean, with similarly low standard deviation. This is typical of modern, high-speed networks. Also note that the RTT sequence has abrupt increases ($t \approx 170$ms, $t \approx 430$ms) followed by a stable period ($170\text{ms} < t < 260$ms), and then an abrupt decrease ($t \approx 260$ms). Note that Jacobson and Eifel both "overshoot" after the abrupt positive discontinuity in RTT. However, at the second discontinuity which is abrupt but negative, Eifel corrects downward, whereas Jacobson again corrects upward. This is a primary beneficial feature

 ISBN: 978-1-61208-262-2

of the Eifel approach. Unfortunately, the tuning of the algorithm creates a larger deviation or "upper bound" for the RTT sequence than is available through the Jacobson Algorithm.

Abrupt changes in RTT such as those shown in Figure 1 cause the RTO timer to expire, resulting in binary backoff of the timer (BEBO) and retransmission procedures. The RTO values associated with BEBO and retransmissions are not shown, because they do not affect future estimated values of RTO. Post-processing of the RTT estimate to enforce minimum values for RTO is also not considered here. In the RFCs for TCP and SCTP, a hard-coded minimum value for RTO is specified as 1000 milliseconds [1]–[3], [5], whereas the minimum value of the Eifel algorithm is defined as "RTT + 2 $\times$ ticks" (or 200 msec) [6]. The definition of the minimum RTO in TCP-Lite is typically 2 times the clock granularity, which is often taken as 500 msec [3], [6]. This minimum procedure is secondary to the estimation algorithm, and often completely replaces the RTT estimate. For example, in a broadband network with large minimum RTO, the RTT estimate can be orders of magnitude below the minimum, resulting in long wait-times for triggering retransmission. So, in all comparisons here we disable the enforcement of a minimum RTO and focus on the performance of the estimation algorithms.

III. The Jacobson Algorithm

The Jacobson Algorithm, originally proposed in [4], uses the Chebyshev Bound [16] to produce a reasonable value of RTO, or the maximum time the sender will wait for an acknowledgement. After exceeding RTO, a transmission is declared lost and retransmission procedures commence. Interestingly, Jacobson noted the poor performance of the algorithm, since loads higher than 30% resulted in retransmission of packets that were only delayed in transit (i.e., not lost) [4]. This behavior is also noted in later literature, including [6], [7], [15].

The specific computations for the bounding procedure are driven by estimates of the mean (μ) and the standard deviation (σ) of an assumed RTT distribution. The estimates of μ and σ are not conventional parameter estimates, as iid samples from a population. Rather, $\hat{\mu}$ and $\hat{\sigma}$ are *predicted* using prior RTT values. Jacobson estimates $\hat{\mu}$ and $\hat{\sigma}$ from observed values of RTT, and then computes an "overbound" for RTO using Chebyshev. This is the same as saying "we waited a reasonable time ($\hat{\mu}$) and *then* some ($K\hat{\sigma}$), but the ACK didn't come back, so the packet must have gotten lost". This RTO calculation is invoked by the sender for each unique transmission. As such, it is optimized for integer arithmetic and all coefficients are diadic.

Regardless of the specific assumptions or optimizations, Jacobson computes the RTO threshold as

$$x_{thr} = \hat{\mu}_{n+1|n} + K \cdot \hat{\sigma}_{n+1|n} \tag{1}$$

where x_{thr} is the RTO threshold or "overbound" for RTT, K is the number of "standard deviations past the mean," and $\hat{\mu}_{n+1|n}$ and $\hat{\sigma}_{n+1|n}$ are the estimates (predictions) of the mean and standard deviation of the RTT distribution for the next iteration (subscript $n+1$) given some data up to the current time (subscript n).

A. Jacobson and Chebyshev

The Chebyshev bound (2) is a universal bound applicable even for unknown distributions. Chebyshev shows that the probability of the random variable occurring outside a range around the mean (μ) depends on the standard deviation (σ).

$$\Pr\left[|X-\mu| \geq \epsilon\right] \leq \frac{\sigma^2}{\epsilon^2} \tag{2}$$

For RTT estimation, the Jacobson Algorithm uses a fixed offset from the mean $(K\sigma)$ [2], [3]. This simplifies the bound and allows a direct computation of the "violation probability". With $\epsilon = K\sigma$ and $K = 4$ in (2),

$$\Pr\left[|X-\mu| \geq K\sigma\right] \leq \frac{1}{K^2} \tag{3}$$

and the fixed, double-sided "violation probability" is $\frac{1}{16} = 0.0625$. However, the RTO timeout is single-sided because the timeout algorithm is only concerned with the case where $X > \mu$. So for a symmetric distribution and $K = 4$, the RTO timeout will be exceeded roughly 3% of the time, with the assumption that the RTT values are reasonably iid.

B. Predicting μ and σ

Rather than using conventional parameter estimation algorithms requiring storage of historical values and more complex calculation, Jacobson estimates μ and σ using simple *prediction algorithms* as in (4). These algorithms rely on current values of the quantities (subscript n) as well as the measured RTT value (x_n). Interestingly, the use of *filtered* (predicted) values for μ and σ implicitly contradicts the assumption of a valid distribution in (2). However, this approach gains computational efficiency and reflects the nonstationary nature of RTT values.

$$\begin{aligned} \hat{\mu}_{n+1|n} &= \hat{\mu}_n + \alpha(x_n - \hat{\mu}_n), \text{ and} \\ \hat{\sigma}_{n+1|n} &= \hat{\sigma}_n + \beta\left(|\hat{\mu}_n - x_n| - \hat{\sigma}_n\right). \end{aligned} \tag{4}$$

When viewed as a time-series prediction or filtering algorithm, it is clear that the relations in (4) are single-pole, lowpass IIR filters (predictors).

IV. The Modified Jacobson Algorithm

The use of Chebyshev to bound the retransmission timeout is reasonable, since it provides a "target probability" for the timeout calculation. However, the Markov bound [16] is also applicable for RTT estimation since it explicitly uses knowledge of the positivity of the random variable, as in (5) which is valid when $f_X(x) = 0$ for $x < 0$ and $\alpha > 0$.

$$\Pr\left[X \geq \alpha\right] \leq \frac{\mu}{\alpha}. \tag{5}$$

 ISBN: 978-1-61208-262-2

Using $\alpha = \mu + K\sigma$ in (5) produces an expression similar to (2). However, the Markov formulation results in a *variable* probability for RTO violation. This concept is particularly unappealing for small, relatively stable values of RTT, since the overbound RTO might have a high probability of violation, which would create spurious retransmissions and a large amount of unwanted network traffic. Recall that in the Chebyshev case, the choice of $\epsilon = K\sigma$ produced a fixed violation probability around 3%. The violation probability of the Markov bound can also be fixed as in (3) if $\alpha = 32\mu$:

$$\Pr[X \geq 32\mu] \leq \frac{\mu}{32\mu} = \frac{1}{32} = \frac{1}{2K^2}. \quad (6)$$

Unfortunately, a bias of 32 times the mean would not produce a viable estimate of RTT, and the overbound RTO would be extremely loose. Therefore, the Markov bound *alone* is not be a reasonable choice to estimate the RTO timeout. However, a *combination* of Markov and Chebyshev approaches seems to produce an effective estimator.

Combining an estimate of σ as in the Jacobson Algorithm with a *biased* estimate of μ as in the Markov bound results in a formulation that retains the Chebyshev structure but improves certain performance aspects. So, we use a slightly revised version of (1) and call this approach the *Modified Jacobson Algorithm*,

$$x_{thr} = \mathcal{A} \cdot \hat{\mu}_{n+1|n} + \mathcal{B} \cdot \hat{\sigma}_{n+1|n}. \quad (7)$$

In (7), the standard deviation estimator $\hat{\sigma}_{n+1|n}$ is identical to the estimator (4) used in the Jacobson Algorithm. However, the multiplier for $\hat{\sigma}_{n+1|n}$ is *reduced* (i.e. $\mathcal{B} = 2$ whereas $K = 4$). Further, the mean estimator $\hat{\mu}_{n+1|n}$ is replaced with the current value of RTT (x_n), and is *biased* in the spirit of a Markov estimator. In this case, we choose $\mathcal{A} = 1.25$ as a reasonable bias term, whereas Jacobson uses $\mathcal{A} = 1.0$.

Note that the bias term $\mathcal{A}$ for $\hat{\mu}_{n+1|n}$ in (7) is *not equivalent* to the use of gain in the prediction of $\hat{\mu}_{n+1|n}$ in (4). The structure of the prediction filter for $\hat{\mu}_{n+1|n}$ causes delay in the formulation of the overbound, which is problematic. There are no coefficients for the prediction filter which will simultaneously improve delay and maintain stability in the estimation of μ, and incorporating gain in the prediction does not improve the estimate. These undesirable effects are completely eliminated in the Modified approach with the use of x_n as the estimator of μ. This adjustment allows for the use of a Markov-like bias term $\mathcal{A}$ and significantly enhances the performance of the Modified algorithm.

Several factors must be specifically noted for the Modified Algorithm. First, dependence on the variance of the RTT sequence is preserved via $\hat{\sigma}_{n+1|n}$ and the Chebyshev-like formulation. Some dependence on σ must be maintained in the estimation procedure for cases where the RTT values exhibit significant variability. However, the multiplier $\mathcal{B}$ can be different (smaller) than in Jacobson. This reduced dependence on σ mediates undesirable "overshoot" which is problematic in Jacobson, and has been addressed heuristically in Eifel. Refer to abrupt changes in RTT as shown in Figure 1 for examples.

Secondly, dependence on the mean of the RTT sequence is preserved via the use of x_n for $\hat{\mu}_{n+1|n}$, and a bias is incorporated via the Markov-like formulation for cases where $\sigma \to 0$. Some dependence on μ is important, since this allows isolation of the variability. However, in cases where $\sigma \to 0$, Jacobson tends to "settle" directly onto RTT, leading to heuristic modifications including static minimum values which override the Jacobson estimates. This undesirable behavior of Jacobson is clearly evident in Figure 1.

Thirdly, explicit dependence on both μ and σ is retained via the hybrid Markov/Chebyshev formulation which *biases* the estimate higher and reduces the need for secondary minimum computations. Also, the prediction structure for $\hat{\mu}$ and $\hat{\sigma}$ is preserved, which is an important consideration.

Finally, the computational complexity of the Modified Algorithm is essentially the same as the original Jacobson Algorithm. Elimination of the prediction structure for $\hat{\mu}$ and the use of a bias term along with a simplified multiplier for $\hat{\sigma}$ results in an algorithm with the same operational complexity and no heuristic conditional logic steps.

V. Performance results

To validate algorithm performance, we constructed a "real-world" test environment which pairs client and server computers with modified network stacks via a controllable network infrastructure. In the network testbed, a client system transmits data to a server system using a specially-constructed user-space application which can vary overall payload length and SCTP chunk size, as well as other parameters such as the number of test iterations. The network stack of the client systems also implement user-selectable timeout estimation algorithms and record important parameters for each transmission. Parameters recorded by the client's network stack include the timestamped, per-chunk values of the actual (measured) RTT, estimated RTO, $\hat{\mu}$, $\hat{\sigma}$, and so on. As described in Section II-C, post-processing of the RTO estimate to enforce minimum values for RTO is disabled since we are investigating the effect of estimation algorithms.

Additionally, the network devices and SCTP server are modified to introduce algorithmically controllable delays in acknowledgements of chunks and delays in delivery of various classes of network traffic. This feature results in an ability to introduce specific "delay profiles" which duplicate other known results (as in Figure 2 [6]) or randomize the round-trip time of the network. Using trace data collected directly from the network stacks of the client & server computers, we were also able to create simulations of system performance which have been cross-checked for accuracy against the delay and estimation performance of the actual systems. All performance data described in this paper was produced using our "real-world" testbed, and has

 ISBN: 978-1-61208-262-2

Table I
ESTIMATION PERFORMANCE GATHERED USING NETWORK TESTBED AND SIMULATION. PERFORMANCE IS MEASURED AS MEAN ABSOLUTE ERROR (MAE) IN MILLISECONDS.

Delay profile	Figure	Jacobson	Eifel	Modified
Eifel ramp	Figure 2	1731	2091	1577
Quiet/Spikes	Figure 3	39.81	140.7	43.29
Delay burst	Figure 4	11.31	41.79	2.00

Table II
ESTIMATION PERFORMANCE GATHERED USING NETWORK TESTBED AND SIMULATION. PERFORMANCE IS MEASURED IN TERMS OF TIMEOUT EVENTS (TO) PER 10,000 PACKETS TRANSMITTED.

Delay profile	Figure	Jacobson	Eifel	Modified
Eifel ramp	Figure 2	1	1	1
Quiet/Spikes	Figure 3	99	51	79
Delay burst	Figure 4	378	82	12

been incorporated into accurate simulations of the estimation algorithms.

The performance of the Modified Jacobson Algorithm is shown in the context of various RTT sequence characteristics or "delay profiles" in Figure 2, Figure 3, and Figure 4. Figure 2 reproduces an important delay profile from literature describing the Eifel algorithm (Fig. 6 of [6]). Figure 3 contains the delay profile for a real, quiescent network with 100 msec average delay and short, artificially induced delay spikes. Figure 4 contains the delay profile for a long-term delay burst on an otherwise quiescent network with 80 msec average delay.

Quantitative assessment of algorithm performance gathered from a large number of packet transmissions is shown in Table I and Table II for each of the delay profiles described by Figure 2, Figure 3, and Figure 4. The data in Table I is presented in terms of Mean Absolute Error (MAE) in milliseconds between the RTO estimate and the actual RTT value for the same packet transmission, according to (8). Table II shows the number of induced timeout events (TO) for each estimation algorithm, and is measured in number of events per 10,000 packets transmitted.

$$\text{MAE} = \frac{1}{N} \sum_{N} |x_{thr} - x_n| \qquad (8)$$

Using our network testbed, we duplicated the RTT sequence in Figure 2 from [6]. This "delay profile" clearly shows the improvement of Eifel over Jacobson, particularly at the termination of the "ramp" sequences. Note that Jacobson overshoots *upward* even though the RTT sequence has rapidly declining values. Eifel compensates for overshoot, but at the expense of slightly higher bias from the RTT sequence. Note that the Modified approach mimics Jacobson during ramp ascension, and also compensates for the overshoot at ramp termination. Modified has a smaller

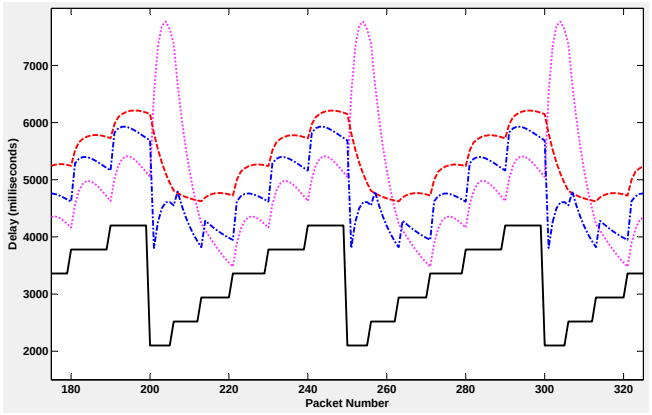

Figure 2. Performance of RTO estimators for the "ramp" RTT sequence described in [6]. Jacobson (dotted), Eifel (dashed), and Modified (dash-dot). The common RTT sequence driving all algorithms (solid).

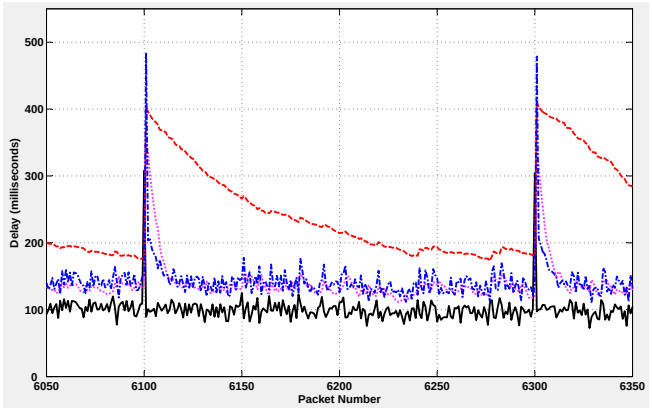

Figure 3. Performance of RTO estimators for a quiescent network with short, artificially induced delay spikes. Jacobson (dotted), Eifel (dashed), and Modified (dash-dot). The common RTT sequence driving all estimation algorithms (solid).

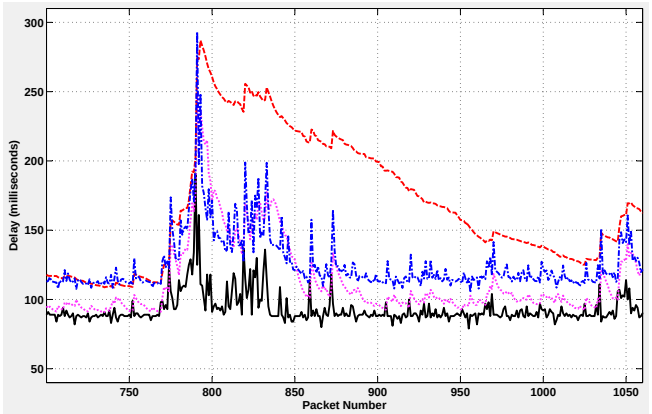

Figure 4. Performance of RTO estimators for a quiescent network with a large, naturally-occurring multi-packet delay burst. Jacobson (dotted), Eifel (dashed), and Modified (dash-dot). The common RTT sequence driving all algorithms (solid).

 ISBN: 978-1-61208-262-2

bias than Eifel. Interestingly, each algorithm induces a single timeout event, but the Modified Jacobson algorithm does so with an MAE 154 msec smaller than Jacobson, and 514 msec smaller than Eifel. Such a significant difference in MAE can translate to a smaller task-completion time in cases where a small differential in timeouts is encountered.

A delay profile from a relatively quiescent network with artificially induced delay spikes is shown in Figure 3. The network exhibits an average delay of around 100msec, and has artificially induced delay spikes which are typical of an unstable link. Note that with this network profile, delay spikes occur every 200 packets, forcing a timeout. After each delay spike, the estimation algorithms recover in very different manners: Modified and Jacobson fall quickly toward the quiescent RTT sequence, while Eifel decays very slowly, creating a relatively large wait-time for over 100 subsequent packets. As a result, the MAE for the Eifel algorithm with this delay profile is more than 3 times larger than the other estimation algorithms while still creating only 35% fewer timeout events. Regardless, each estimation algorithm creates a very small proportion of timeout events relative to the number of packets transmitted.

A delay profile from another network test is shown in Figure 4. In this figure, the network exhibits an average delay during quiescent periods of around 80 msec. However, between packets 750 and 850 a large, naturally occurring, correlated delay burst is observed which disrupts the estimation algorithms. Note that Modified and Jacobson both fall quickly after individual, large delay spikes. However, Modified maintains a larger offset during the quiescent period between packets 900 and 1000 due to the specific bias for $\hat{\mu}$. The failure of Jacobson to maintain a bias during periods of low RTT variance ($\sigma \rightarrow 0$) is responsible for many timeout events, with Jacobson inducing *30 times more timeouts* than Modified, and *almost 5 times more* than Eifel. Eifel again exhibits a bias which is significantly larger than Jacobson or Modified, with an MAE *20 times larger* than Modified, and *4 times larger* than Jacobson. Also note that Jacobson tracks RTT fairly well, but has large, positive overshoot when RTT drops suddenly. Modified compensates for the Jacobson "overshoot" problem in cases where the RTT sequence drops suddenly (cf. packet 800 & 840).

VI. Conclusion and Future Work

This paper analyzes the methods for computing and using RTT and RTO estimates in IP-based transport protocols such as SCTP and TCP. The theoretical basis of the Jacobson Algorithm is discussed, and an alternative approach is presented which retains the fundamentally sound theoretical basis and operational structure of the algorithm, but improves the performance over other well-known techniques without introducing heuristic modifications. Future work involves the continued optimization of the modified algorithm as well as investigation into the effects of variable minimum bound for the RTO timer.

References

[1] J. Postel, "Transmission control protocol," RFC793, Sep. 1981.

[2] R. Stewart *et al.*, "Stream control transmission protocol," RFC 2960, Oct. 2000.

[3] R. Stewart, "Stream control transmission protocol," RFC4960, Sep. 2007.

[4] V. Jacobson, "Congestion avoidance and control," in *Proc. Comput. Commun. Review*, ser. SIGCOMM'88. New York, NY: ACM, Aug. 1988, pp. 314–329.

[5] V. Paxson, M. Allman, J. Chu, and M. Sargent, "Computing TCP's retransmission timer," RFC6298, June 2011.

[6] R. Ludwig and K. Sklower, "The Eifel retransmission timer," *Comput. Commun. Review (SIGCOMM)*, vol. 30, pp. 17–27, July 2000.

[7] H. Ekstrom and R. Ludwig, "The Peak-Hopper: A new end-to-end retransmission timer for reliable unicast transport," in *Proc. 23rd Joint Conf. of the IEEE Computer & Communications Societies (INFOCOM 2004)*, vol. 4, Nov. 2004, pp. 2502–2513.

[8] L. Ma, G. Arce, and K. Barner, "TCP retransmission timeout algorithm using weighted medians," *IEEE Sig. Proc. Letters*, vol. 11, pp. 569–572, June 2004.

[9] J. Pedersen, C. Griwodz, and P. Halvorsen, "Considerations of SCTP retransmission delays for thin streams," in *Proc. 31st IEEE Conf. on Local Computer Networks*, Tampa, FL, Nov. 2006, pp. 135–142.

[10] A. Petlund, P. Beskow, J. Pedersen, E. S. Paaby, C. Griwodz, and P. Halvorsen, "Improving SCTP retransmission delays for time-dependent thin streams," *Multimedia Tools and Applications*, vol. 45, pp. 33–60, Oct. 2009.

[11] M. Allman, K. Avrachenkov, U. Ayesta, J. Blanton, and P. Hurtig, "Early retransmit for TCP and stream control transmission protocol (SCTP)," RFC5827, Apr. 2010.

[12] M. Allman and V. Paxson, "On estimating end-to-end network path properties," in *Proc. Conf. Appl., technol., arch., and protocols for comput. commun.*, ser. SIGCOMM '99. New York, NY, USA: ACM, 1999, pp. 263–274.

[13] L. Coene and J. Pastor-Balbas, "Telephony signaling transport over stream control transmission protocol (SCTP) applicability statement," RFC4166, Feb. 2006.

[14] I. Psaras and V. Tsaoussidis, "Why TCP timers (still) don't work well," *Computer Networks*, vol. 51, pp. 2033–2048, Nov. 2007.

[15] J. Pedersen, "Evaluation of SCTP retransmission delays," Master's thesis, University of Oslo Department of Informatics, May 2006.

[16] A. Papoulis, *Probability, Random Variables, and Stochastic Processes*, 3rd ed. New York, NY: McGraw-Hill, 1991.

 ISBN: 978-1-61208-262-2