

# Some Observations on Individual TCP Flows Behavior in Network Traffic Traces

Lei Qian

Department of Computer Science  
The University of Auckland  
Auckland, New Zealand  
lqia012@aucklanduni.ac.nz

Brian E. Carpenter

Department of Computer Science  
The University of Auckland  
Auckland, New Zealand  
brian@cs.auckland.ac.nz

**Abstract—** We propose a modified Transmission Control Protocol (TCP) flow classification method and Round Trip Time (RTT) computation method which is more precise and more dynamic than the traditional fixed timeout method, especially for long, sparse TCP flows and long RTTs. Then we present passive measurement results for TCP RTT and RTT variation in network traffic traces. Analysis shows several interesting behavior patterns in individual flows. We observe TCP flows with regular patterns of RTT distribution. Individual TCP flows may also apparently have self-similar RTT distributions. Most long-active but sparse TCP flows use port 80 with a relatively short RTT. Lastly, we show and discuss plots of TCP flows with very long RTT values.

**Keywords;** TCP Flows; Flow Separation; RTT Computation; RTT Variation; Time Series Plot; CDF Distribution.

## I. INTRODUCTION

A network flow is typically identified by its IP header 5-tuple [12]. TCP is the dominant protocol in the Internet [1, 2, and 13], and passive TCP flow measurement is of interest for commercial and research purposes.

W. Leland et al first observed self-similarity in the time distribution of Ethernet traffic [14]. N. Wisipongphan et al measured TCP self-similarity using simulation [15] and stated that “if one or more streams passing through the bottleneck is self-similar and the aggregate flow does not exceed the capacity, traffic observed at the bottleneck will also be self-similar [15]”. Our original motivation for the present work was to find whether individual TCP flows in the trace files available to us have self-similar properties.

Many analyses of individual TCP flows are based on active measurements [15]. In this work, we attempt to study traffic characteristics by analyzing passive network traces and extracting individual flows from them.

In Section II, we introduce our measurement data sets; and discuss our measurement metrics, methodologies and framework. In Section III, we illustrate four different types of TCP flow with different characteristics, with some discussion of possible causes. In Section IV we conclude our work and propose some topics for future study.

## II. MEASUREMENT METHODOLOGY AND FRAMEWORK

**Measurement Trace Files:** We used eight existing trace files captured in various years at the Internet gateways

outside the firewalls at the University of Auckland and the University of Waikato (WITS) [4]. Over 3 TB of traffic and 4.7 billion packets were assessed in our measurement work. Table I shows basic information on the trace files [1].

### Measurement Metrics:

- **TCP Flow Classification:** TCP flows are typically classified by a 5-tuple: Source IP Address, Destination IP Address, Source Port, Destination Port and Protocol Number [3]. We use a hash table to store these attributes as a 5-tuple key, to decide whether a TCP Packet belongs to certain TCP flow. TCP connections are bidirectional, so we aggregate two simultaneous flows with IP addresses and port numbers swapped as one two-way flow [3].

A TCP connection may be open for a very long time. Researchers normally set a fixed timeout value; if a TCP flow transmits no data within that time, the analysis program will deem the flow closed and remove its entry from the hash table [5]. This technique avoids memory overflow, but its disadvantage is that it is likely to separate one lengthy single flow into several shorter flows.

Our flow classification method has three checking criteria. First, we use TCP’s FIN bit to signal that a TCP connection is terminated. Second, like other workers, we use a timeout of 30s without data to deem a flow closed. Thirdly, however, we do not check this value unless the hash table size reaches a large value, specifically 100000 entries. Thus, we allow the hash table to grow to 95% of its predefined maximum size. Only then do we apply the 30 second timeout. Our results show that this modification typically allows us to retain a TCP flow’s entry for more than 10 minutes ‘silence’. We believe this is a more accurate method than the traditional fixed timeout method, especially for detecting long-active but sparse TCP flows.

- **Selection of TCP Flows:** Since each traffic trace file contains at least 4 million TCP flows, it is impossible to analyse all of them. Our recent work [1] shows that over 90% of TCP flows have less than 350s lifetime. We chose to analyse individual flows with long lifetimes, so that time series plots are more meaningful. Thus, we selected the 100

longest lifetime flows from each trace file for analysis.

- **Round Trip Time (RTT) Computation:** RTT is defined [7, 8] as the time between sending a TCP packet and receiving its acknowledgement (ACK). We calculate RTTs from sequence numbers and the time stamps in the trace files. (This measures RTT at the measurement point, not the host, a negligible difference compared to wide-area Internet RTT.)

We applied the same logic used for flow classification. We use 5s as RTT timeout value but check the timeout only when the hash table reaches 15000 entries, and thus detect some TCP packets with up to 10s RTT.

- **Computing RTT Variations (RTTV):** IP delay Variation (IPDV) is defined as the difference between the one-way delays of successive packets [9]. We define RTT Variation as the time difference between RTT values of the two successive TCP packets in the same flow.

**Measurement Framework:** We used an Intel i7 Q740 processor with 8GB memory hardware to analyse our trace files using Java software. Our measurement framework required us to run each trace file twice. We first group all TCP packets into corresponding flows, then sort all TCP flows in descending order based on TCP flow lifetime. Secondly, we store the details of the longest 100 TCP flow in each trace file into our database. Lastly we go through each trace file again to work out instantaneous RTT and RTT Variation for those longest TCP flows and output them for further plotting. Time series distribution and cumulative distribution function (CDF) distribution plots were made for both RTT and RTT Variation.

### III. RESULTS AND DISCUSSION

In this section, we present results showing four different phenomena in the RTT distributions and RTT Variation distributions of TCP flows. We only provide one typical example for each behavior type due to space constraints. More examples and details can be found in [11].

Figures 1, 3, 4, 6 and 7 each consist of four subplots; the first shows the instantaneous RTTs of an individual TCP flow. The second shows the CDF distribution of the RTT of this flow. The third subplot shows the time series of RTT Variation and the fourth shows its CDF Distribution. Figures 6 and 7 use a different y-axis scale, because they show an example of a TCP flow with extremely long RTT. Figures 2 and 5 show six subplots of RTTs. Each represents half the duration of the previous one, chosen at random.

#### A. TCP Flows with a Regular Patterned Feature

Figure 1 shows a TCP flow example we have named “regular patterned feature”. The time series RTT plot shows the RTT value regularly goes up and down; most TCP packets were acknowledged within ~1000ms. The RTT CDF distribution shows that ~50% of TCP packets are

acknowledged between 800 and 1000ms. The RTTV time series and RTTV CDF plots show RTTV centred around zero, suggesting that the path between the two ends of this TCP flow is relatively stable.

Figure 2 zooms in on the RTT time distribution. The time series seems to have a regular pattern at each level of zoom. The RTT appears to be reasonably stable for short intervals followed by a brief increase or decrease. A possible explanation is the impact of TCP congestion control. TCP will attempt to increase the window size steadily, increasing the data rate until congestion delays ACKs and increases RTT; then TCP will decrease its window size which will reverse the effect.

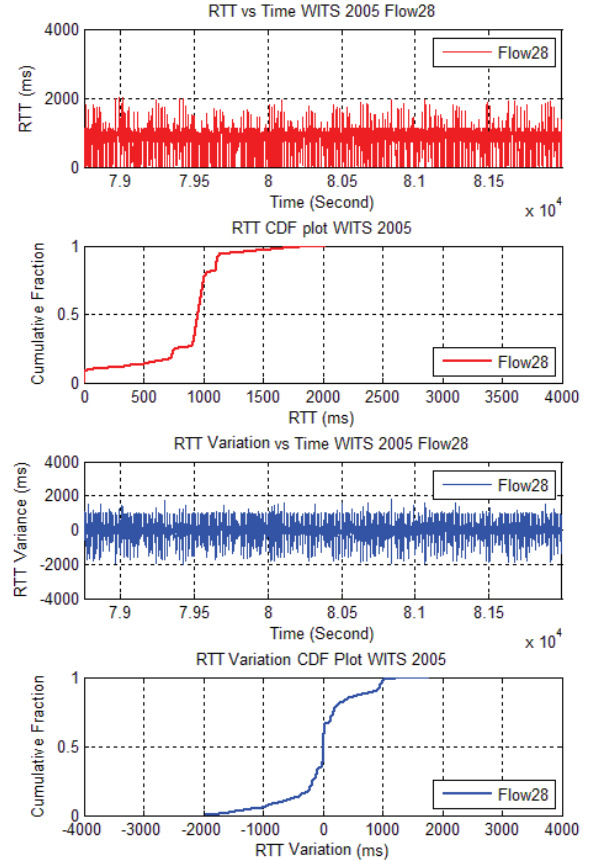


Figure 1. Time Series and CDF Distribution of RTT and RTT Variation for Flow 28 in WITS 2005.

#### B. Long-Active TCP Flows with Sparse Traffic Volume

The TCP flow in Figure 3 has 144.4 minutes (8664.67s) lifetime but only 12470 packets, were sent, including ACKs. Fewer than two TCP packets or ACKs were sent each second on average. In total, only ~2.3MB of data were transmitted in 2.3 hours. This kind of TCP flow has a long lifetime but comparatively few packets transmitted.

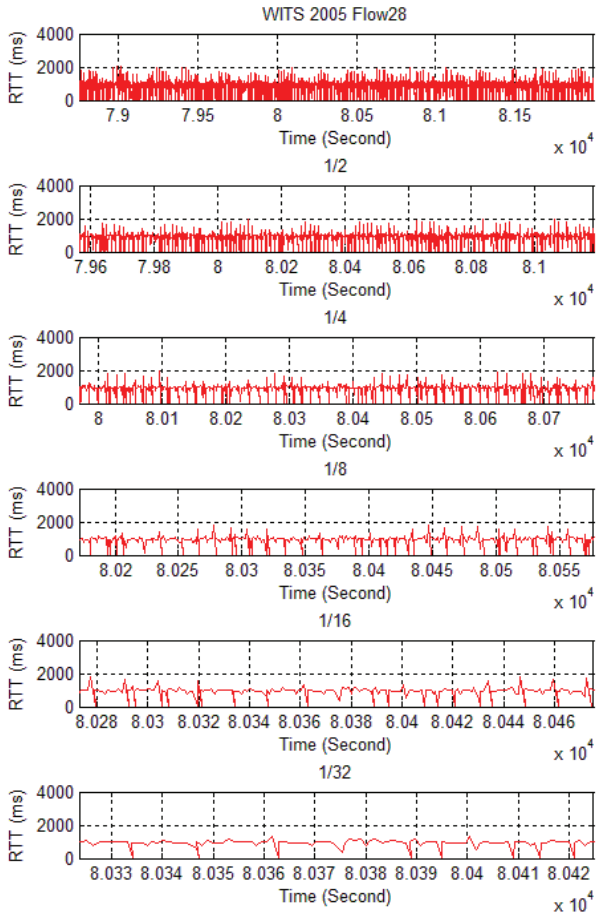


Figure 2. Zoomed In time series versus RTT plots for Figure 1.

The RTT CDF plot in Figure 3 shows that over 80% of packets have been acknowledged within 800ms. The stepwise CDF distributions are because only a few packets are included in this flow. The RTTV CDF plot in Figure 3 shows RTTV between -800ms and 1000ms. Most RTTV values are within  $\pm 500$ ms.

We had expected that long lifetime, low traffic TCP flows would typically be SSH flows using TCP port 22. However, most long lifetime, low traffic flows are persistent HTTP flows using TCP port 80.

Two possible reasons for HTTP flows to have such a characteristic are HTTP Long Polling and HTTP Streaming. In HTTP Long Polling, waiting for a server's response may cause long-lived HTTP flows without much traffic [10]. In HTTP Streaming, when an HTTP session is established, "the server keeps a request open indefinitely even after it pushes data to the client [10]."

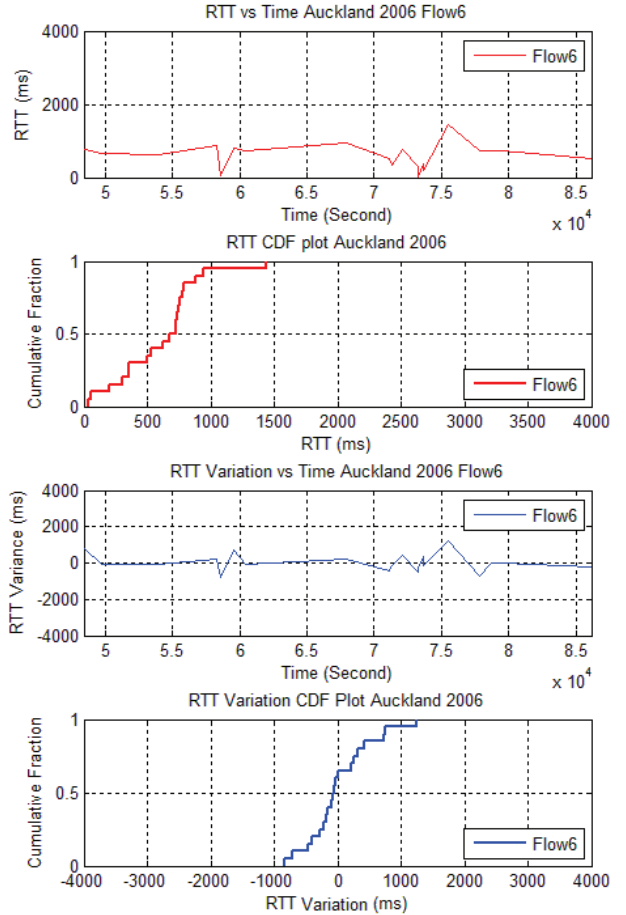


Figure 3. Time Series and CDF Distribution of RTT and RTT Variation for Flow 6 in Auckland 2006.

### C. TCP Flows with Self-Similar Appearance

Figure 4 shows the instantaneous time series and CDF distributions of RTT and RTTV of the longest TCP flow in the AKL-2007 trace. The RTT distribution shows that the RTT continuously increases and decreases between a few milliseconds to around 3000ms. The RTT CDF distribution shows that  $\sim 20\%$  of TCP packets are acknowledged within 200ms. There is a vertical increase at  $\sim 200$ ms where about 40% of TCP packets are acknowledged. The instantaneous time series RTTV distribution shows that RTTV is spread within  $\pm 3000$ ms. In the RTTV CDF plot, almost all RTTV is between  $\pm 2000$ ms. We observe that the RTTV CDF distribution appears to be symmetrical about the middle point (0, 0.5) of the plot.

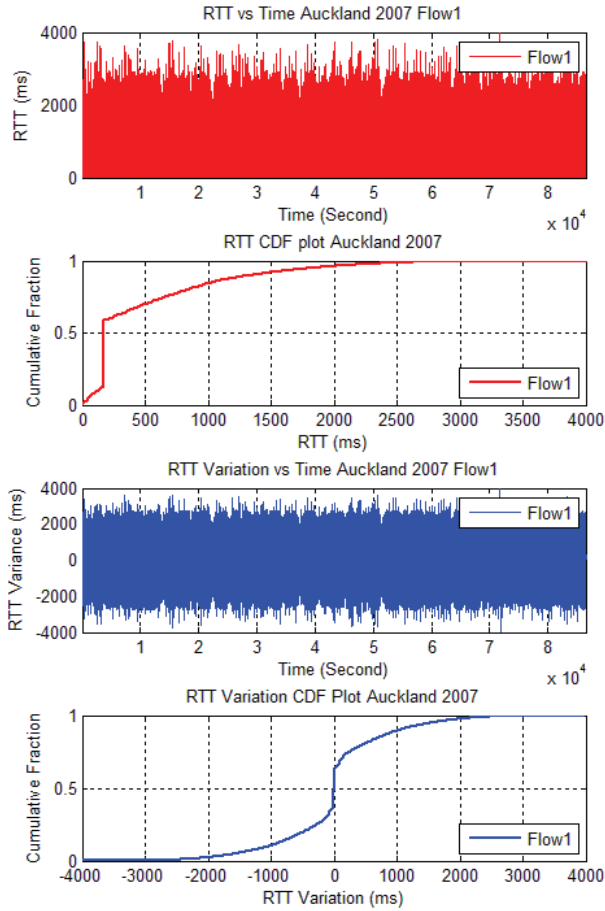


Figure 4. Time Series and CDF Distribution of RTT and RTT Variation for Flow 1 in Auckland 2007.

Figure 5 shows the same original plot and five zoomed-in subplots of the instantaneous time series RTT distributions for the longest TCP flow in the AKL-2007 trace. Each subsequent plot is a randomly chosen half of the previous one. All six plots appear similar to each other despite having different timescales, typical of self-similar distributions. Thus, a self-similarity property seemingly exists in some of the observed flows. The symmetrical RTT Variation CDF distribution of this flow also suggests that the two ends of this TCP connection switched between congested and uncongested frequently, which may be related to self-similarity.

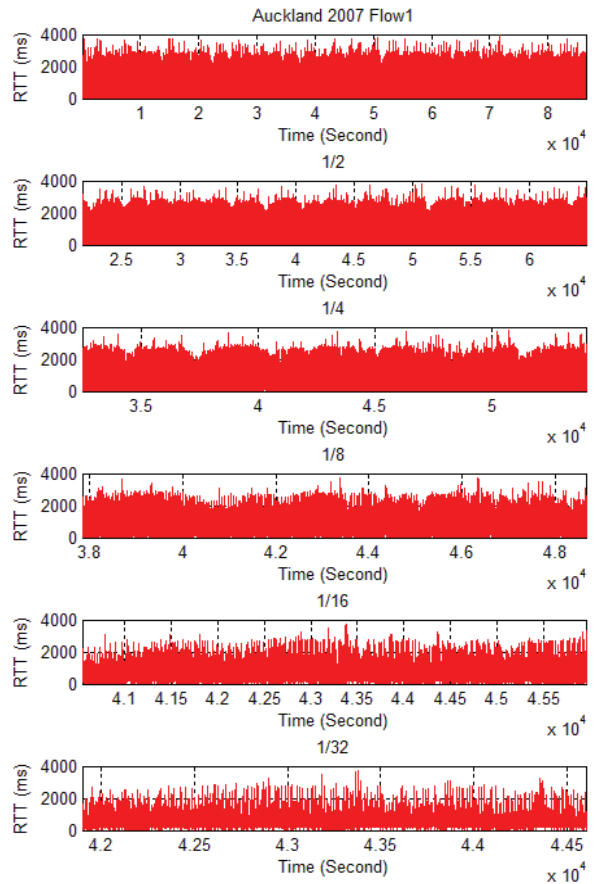


Figure 5. Zoomed In time series versus RTT plots for Flow 1 AUCKLAND 2007

#### D. TCP Flows with Long RTT Value

Figure 6 shows an example of a long RTT TCP flow. The RTT time series shows that most TCP packets are acknowledged with RTTs between  $\sim 3000$ ms and  $5000$ ms. The RTT CDF plot shows that few TCP packets are acknowledged between  $3000$ ms and  $4000$ ms. Nearly 70% of TCP packets in the flow are acknowledged between  $4000$ ms and  $5000$ ms. The RTTV distribution shows that the range of RTTV is mainly within  $\pm 1000$ ms, and about 40% of TCP packets in this flow have an RTTV around zero.

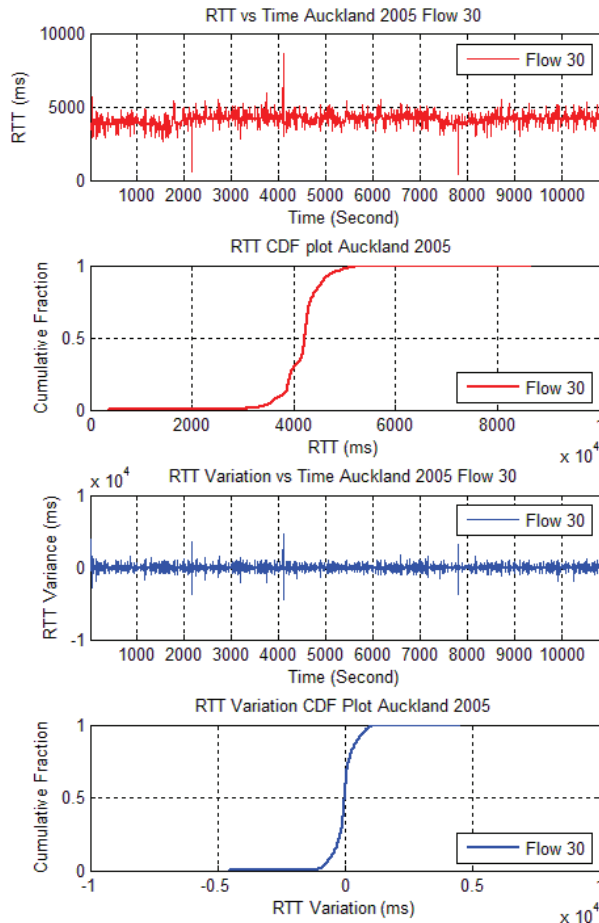


Figure 6. Time Series and CDF Distribution of RTT and RTT Variation for Flow 30 in Auckland 2005.

Based on the RTT distribution and RTTV distributions in Figure 6, we believe that this long average RTT is not caused by congestion, since the RTT CDF plot shows RTTs consistently longer than 3000ms and the RTTV CDF plot shows relatively modest RTTV. There is no evidence of more than one or two packets with a shorter RTT. When long RTTs are caused by network congestion, we expect to see high RTTV and a reasonable number of shorter RTTs, as in the previous examples. Thus we believe Figure 6 reflects a relatively stable TCP session despite its large RTT. Perhaps the path includes multiple satellite hops or a low capacity, or stable and congested bottleneck, or one of the end systems is slow.

Figure 7 shows the 37<sup>th</sup> longest TCP flow in the WITS-2004 trace. The RTT frequently increases and decreases during the flow, between a few milliseconds and 10000ms. The RTT CDF plot shows that only about 20% of TCP packets are acknowledged within a few milliseconds and

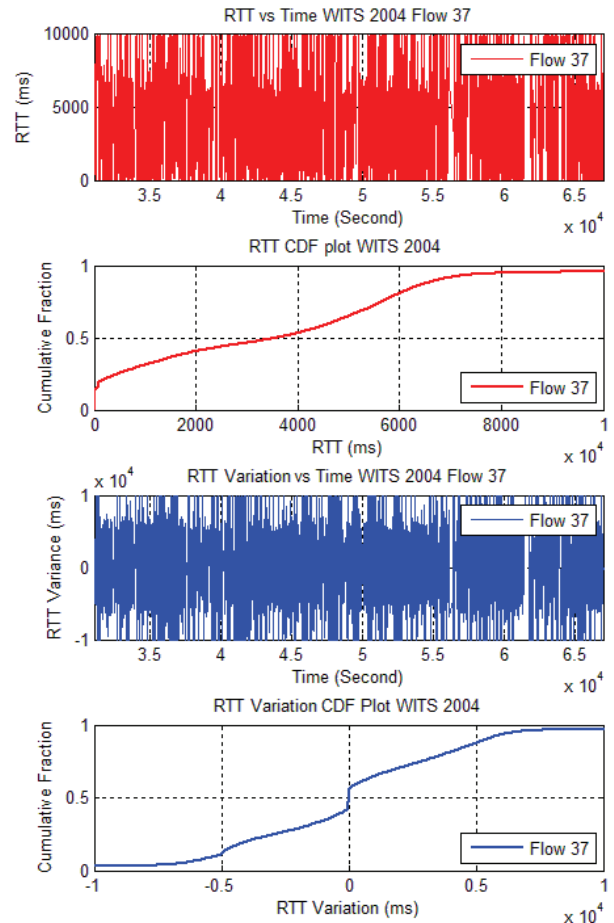


Figure 7. Time Series and CDF Distribution of RTT and RTT Variation for Flow 37 in WITS 2004.

only ~50% are acknowledged within 4000ms. Another ~30% are acknowledged between 4000ms and 6000ms. The RTT CDF curve appears to be loosely linear in the second subplot, which implies the TCP performance is unstable. The RTTV changes frequently within  $\pm 10000$ ms, a very wide range.

The most likely explanation for such instability is congestion. When the network experiences heavy load, each flow's RTT will be strongly affected by TCP congestion control mechanisms interacting with router queues. A recent presentation [17] also proposed that "buffer bloat" may explain such results, since more physical buffer installed in network devices will greatly increase queuing delays.

#### IV. CONCLUSION AND FUTURE WORK

##### A. Conclusion

We have described a modified method of identifying TCP flows in passive traces, and a similar method of

calculating TCP RTT values, by maximising memory usage. We use hash table size to control our flow timeout decision. Thus, incomplete TCP flows and long RTT values have more chance to survive in the hash table, compared to methods based on a fixed timeout value. Only infinite memory would guarantee that all flows are complete.

We observed TCP flow types in various network traces:

1. Flows with a regular patterned RTT distribution.
2. Flows that remain active for a long time with few packets transferred and relatively short RTT.
3. Flows with apparent self-similarity in their RTT distributions.
4. Flows with extremely high RTT values, but stable end-to-end behavior, that do not necessarily reflect unstable network conditions.

### B. Future Work

These results are a starting point and further work could be done to passively measure individual TCP flows, for example:

- The method used in this paper can identify sparse TCP flows in network traces and calculate long RTTs. However, this required tuning of the hash table size to identify relevant flows without memory overflow. A more stable technique for long duration, low volume flows would be beneficial.
- We observed that some TCP flows show apparent self-similarity in their RTT plots. It would be interesting to build a statistical model to detect such self-similarity objectively in individual TCP flows.
- Our selection of TCP flows was based on flow lifetimes. Alternatively, the selection could be based on some other attributes such as the packet count of each flow or the total traffic volume transferred.

### ACKNOWLEDGMENT

We are grateful to Associate Professor Nevil Brownlee and Dr Dongjin Lee (The University of Auckland) for constructive advice and valuable comments. We thank Richard Nelson and the WAND research group (The University of Waikato) for providing us resources and useful information about network trace files.

### REFERENCES

- [1] L. Qian and B. Carpenter, "A Flow-Based Performance Analysis of TCP and TCP Applications", Third International Conference on Computer and Network Technology (ICCNT2011), Taiyuan, China,.
- [2] D. Lee, B. Carpenter and N. Brownlee, "Observations of UDP to TCP Ratio and Port Numbers", Fifth International conference on Internet Monitoring and Protection (ICIMP 2010), Barcelona, May 2010.
- [3] D. Lee and N. Brownlee, "Passive Measurement of One-way and Two-way Flow Lifetimes", ACM SIGCOMM Computer Communication Review, Volume 37, Number 3, July 2007.
- [4] Waikato Internet Traces Storage <http://wand.cs.waikato.ac.nz/wand/wits/index.html>.
- [5] N. Brownlee, "Some Observations of Internet Stream Lifetimes", PAM 2005, pp 265-277, 2005.
- [6] P. Sessini and A. Mahanti, "Observations on Round-Trip Times of TCP Connections", .
- [7] J. Aikat, J. Kaur, F. Donelson Smith and K. Jeffay, "Variability in TCP Round-trip Times", Proceedings of the 3<sup>rd</sup> ACM SIGCOMM conference on Internet measurement. October 2003, May 1998..
- [8] G. Almes, S. Kalidindi, and M. Zekauskas, "A Round-trip Delay Metric for IPPM," *Internet RFC* 2681
- [9] C. Demichelis and P. Chimento, "IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)", *Internet RFC* 3393
- [10] S. Loreto, P. Saint-andre, S. Salsano and G. Wilkins, "*Known Issues and Best Practices for the Use of Long Polling and Streaming in Bidirectional HTTP*", *draft-loreto-http-bidirectional-07*, January, 2011. Available at: <http://tools.ietf.org/id/draft-loreto-http-bidirectional-07.txt>
- [11] L. Qian, Analysis of TCP Flows Based on Applications and Round-Trip Time, M.Sc. Thesis, The University of Auckland, April, 2011.
- [12] K.C Claffy, H.W Braun, and G.C. Polyzos, "A parameterizable methodology for Internet traffic flow profiling," *Selected areas in Communications, IEEE Journal on*, vol. 13, pp. 1481-1494, 1995.
- [13] R. Caceres, P.B. Danzig, S. Jamin and D.J. Mitzel, "Characteristics of Wide-Area TCP/IP Conversations," *Proceedings of ACM SIGCOMM*, 1991.
- [14] W.E. Leland, M.S. Taqqu, W. Willinger and D.V. Wilson, "On the Self-Similar nature of Ethernet Traffic," *Computer Communication Review, ACM SIGCOMM*, 2002.
- [15] N. Wisitpongphan and J.M. Peha, "Effect of TCP on Self-Similarity of Network Traffic," Proceedings of 12<sup>th</sup> IEEE International Conference on Computer Communications and Networks (ICCCN), 2003
- [16] H. Goto, Y. Hasegawa, and M. Tanaka, "Efficient Scheduling Focusing on the Duality of MPL Representatives," Proc. IEEE Symp. Computational Intelligence in Scheduling (SCIS 07), IEEE Press, Dec. 2007, pp. 57-64, doi:10.1109/SCIS.2007.357670.
- [17] Jim Gettys, "Bufferbloat – dark Buffers in the Internet", Bell Labs, 24th March 2011, Available at: <http://www.ietf.org/proceedings/80/slides/tsvarea-1.pdf>

| Trace File | Start Time           | Duration | Volume (Gigabytes) | Total TCP Flows (Million) | Number of Packets(Million) |
|------------|----------------------|----------|--------------------|---------------------------|----------------------------|
| AKL-2003   | 2003-Dec-04 [00:00]  | 24.00hr  | 68.23              | 103.07                    | 150.23                     |
| AKL-2005   | 2005-Aug-16 [14:00]  | 03.00hr  | 48.42              | 3.55                      | 89.09                      |
| AKL-2006   | 2006-July-27 [13:00] | 24.00hr  | 294.30             | 130.22                    | 486.95                     |
| AKL-2007   | 2007-Nov-01 [13:00]  | 24.00hr  | 652.41             | 200.05                    | 1077.48                    |
| AKL-2009   | 2009-Aug-03 [08:00]  | 11.00hr  | 1860.74            | 40.58                     | 2534.84                    |
| WITS-2004  | 2004-Mar-01 [00:00]  | 24.00hr  | 37.29              | 4.05                      | 91.42                      |
| WITS-2005  | 2005-May-12 [00:00]  | 24.00hr  | 58.40              | 5.59                      | 138.82                     |
| WITS-2006  | 2006-Oct-30 [00:00]  | 24.00hr  | 79.25              | 4.89                      | 173.12                     |

TABLE I. Overview of Network Traffic Trace Files for Measurement [1]