A Sample-Mode Packet Delay Variation Filter for IEEE 1588 Synchronization

(Invited Paper)

Margaret Anyaegbu, Cheng-Xiang Wang
Joint Research Institute for Signal and Image Processing
School of EPS, Heriot-Watt University
Edinburgh, UK
mua4@hw.ac.uk, cheng-xiang.wang@hw.ac.uk

William Berrie
TES Electronic Solutions Ltd.
Edinburgh, UK
william.berrie@tes-dst.com

***Abstract*—Recent studies have shown that the delay distribution of IEEE 1588 synchronization traffic varies with the characteristics and amount of the background traffic in the network, and influences the choice of packet selection filters. In this paper, we characterize the delay profile of an Ethernet cross-traffic network statically loaded with one of the ITU-T network models and a larger Ethernet inline traffic loaded with uniformly-sized packets. We also propose an iterative sample-mode packet delay variation (PDV) filter and use numerical simulations in OPNET to illustrate the performance of the filter in the networks. We compare the performance of the proposed PDV filter with those of the existing sample minimum, mean, and maximum filters and observe that the sample-mode filtering algorithm is able to match or outperform other types of filters, at different levels of network load.**

Keywords – Synchronization, IEEE 1588 Precision Time Protocol (PTP), packet delay variation filtering.

I. INTRODUCTION

Recently, there has been a migration from traditional circuit-switched networks based on Time Division Multiplexing (TDM) towards packet-switched networks such as Ethernet. This migration has been mainly driven by the desire to reduce costs and increase bandwidth for new types of services. For instance, some mobile telecommunications operators have started upgrading their backhaul networks to Carrier Ethernet, using new standards such as the Long Term Evolution (LTE) technology framework [1]-[3]. Other promising post-LTE technologies, including Cooperative Multiple Input Multiple Output (MIMO) and cognitive radio, are also being considered [4]-[7]. Some Digital Enhanced Cordless Telecommunications (DECT) system providers have also started using Ethernet as the backbone for connecting base stations in office environments, as well as for distributing high quality audio in auditoria and conference centers. There has also been consideration for using DECT video systems connected over an Ethernet backbone in transportation applications such as train carriages, in order to provide better visibility for train operators.

When migrating towards Ethernet, synchronization is an important requirement that must be provided. Frequency synchronization is necessary for facilitating seamless handover and preserving connection integrity in cellular wireless systems, while time synchronization is required for reducing interference or improving capacity in systems that employ Time Division Duplexing (TDD) or Time Division Multiple Access (TDMA) techniques. For instance, the authors of [8] and [9] have studied time synchronization in base stations for mobile backhaul applications. There has also been interest in providing frame and multi-frame time synchronization for DECT base stations [7]. For both mobile backhaul and DECT applications, the required synchronization accuracy is in the microseconds range. Existing TDM technologies, such as Synchronous Optical Networking (SONET), Synchronous Digital Hierarchy (SDH), and Plesiochronous Digital Hierarchy (PDH), are capable of providing synchronization using a timing reference carried at the physical layer; however Ethernet was not designed for the transport of synchronization.

Two basic methods exist for distributing precision synchronization over packet networks: via the physical layer e.g., Synchronous Ethernet (SyncE), or by exchanging timestamps using a packet protocol. SyncE [11] is a standard for the distribution of frequency over Ethernet links. While it can deliver a high level of frequency accuracy without susceptibility to packet delay variation (PDV), SyncE cannot provide time synchronization [12]. The most popular packet-based synchronization protocols are Network Time Protocol (NTP) [13] and IEEE 1588 Precision Time Protocol (PTP) [14]. As NTP was primarily designed for synchronization over the Internet, the achievable accuracy is in the milliseconds range, which is unsuitable for the DECT and mobile backhaul applications under consideration. PTP, on the other hand, is capable of providing sub-microsecond synchronization accuracy; therefore it is generating a lot of interest as a potential solution for such applications. The new Ethernet Audio Video Bridging (AVB) standard [15], which is based on PTP, can also transport synchronous information; however it requires a special network with reservation protocols to ensure that it meets all the timing constraints required by multimedia applications. Similarly, a reserved timing network can also achieve synchronization. Our goal, however, is to provide time synchronization on a legacy network without reservation.

978-1-4673-3070-1/12/$31.00 ©2012 IEEE

When PTP is deployed in an existing Ethernet network, the synchronization traffic must contend for the network path and share existing network elements with the non-synchronization traffic. This contention causes PDV, especially at the output queue buffers of network switches. If left unchecked, this PDV will adversely affect the synchronization accuracy of the protocol. The PTP standard recognizes this issue, and offers some recommendations for guarding against PDV. One solution is to utilize PTP transparent clocks or boundary clocks at intermediate nodes in the network, in order to measure and correct for the time a packet spends in the queue buffers of output ports. This is likely to be the preferred solution for new network installations, as the specialized switches can be factored into the design and costing of the network. For existing installations however, it is often cumbersome and costly to replace the existing network devices. The standard therefore recommends that other techniques such as traffic design, priority tagging of synchronization traffic, and PDV filtering can be employed. In some cases, more than one technique may be required.

In general, the goal of PDV filtering is to select at least one "good" packet out of several packets within the received synchronization traffic and then use the good packet or packets to achieve synchronization. For most of the PDV filtering algorithms in the literature [16]-[18], a "good" packet is defined as one with the shortest transit time through the network. Hence these algorithms tend to group the arriving synchronization traffic into non-overlapping windows, select the ones that were least impacted by queuing delay, and discard the remaining packets. The PDV filters which employ such algorithms are referred to as sample-minimum or earliest arrival packet filters.

Sample minimum filters can work effectively, as long as packets with minimal queuing delay are delivered at appropriate intervals. In [19] and [20], the authors showed that this is true for cross-traffic networks with moderate levels of background traffic (typically less than 45% utilization). However, they observed that for a heavily-loaded cross-traffic network and moderately-loaded high-hop in-line traffic network, the probability of finding a minimum-delay packet was significantly reduced. They thus proposed sample-mean and sample-maximum filters, respectively, based on the observed packet delay distributions in the networks.

In this paper, first of all we characterize the delay distributions of a cross-traffic network at different network load levels and illustrate how the mean and variance of the observed delays vary with the network load. We further consider the delay distribution of a large in-line traffic network. From the observed delay distributions, we suggest the most likely type of existing filter that would perform best. Then, we propose an iterative sample-mode PDV filtering technique that selects "good" packets from a mode bin to achieve synchronization. Finally, we compare the performance of the sample mode filter with the performance of the existing types of filters for some of the network scenarios whose delay distributions were profiled.

The rest of the paper is organized as follows. In Section II, we describe how the delay profile of a network can impact the performance of a PDV filtering algorithm, and characterize the profile of a cross-traffic network and an in-line traffic network. Section III summarizes existing approaches to PDV filtering, and describes the proposed sample-mode filtering algorithm. Simulation results and analysis are presented in Section IV, and then conclusions are drawn in Section V.

II. PACKET DELAY PROFILING

In order to characterize the delay profile and appreciate the effects of PDV on the synchronization performance of the network, we simulated a 5 hop network with data-centric background traffic based on the ITU-T G.8261 Network Traffic Model 2 [21], as illustrated in Fig. 1. Each node generates background traffic that follows the same path as the SYNC packets. The next node extracts the background traffic and injects new (independent) traffic along the synchronization path. In the literature, this type of traffic pattern is referred to as cross-traffic [20]. Quality of Service (QoS) was implemented in the model so that the SYNC packets were transmitted with strict priority queuing at the switch output ports. Simulations were carried out for 20%, 40%, 60% and 80% background traffic load levels. Other simulation parameters are provided in Table 1.

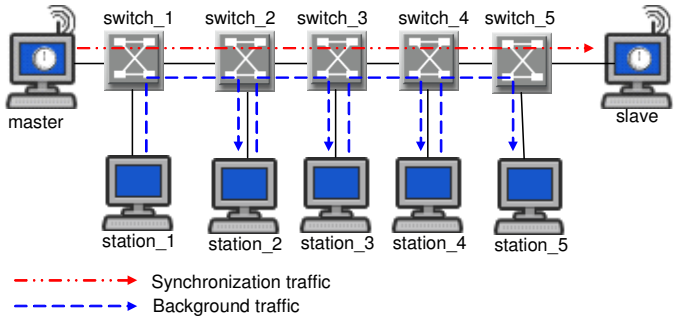

Figure 1. 5-hop network topology for cross traffic.

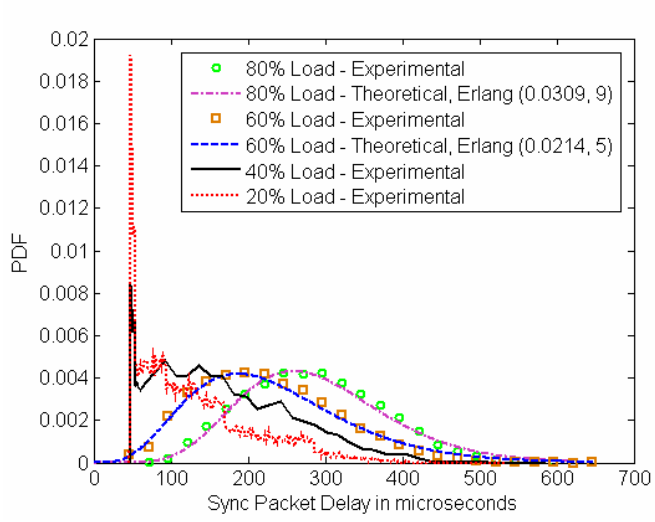

Figure 2. Packet delay distribution for a 5 hop cross traffic network.

TABLE 1: SIMULATION PARAMETERS FOR PACKET DELAY PROFILING.

Distance between nodes	50 metres
Synchronization Interval	3.90625 milliseconds
Fractional frequency offset of slave	2 parts per million
Line Rate	100 Mbps

For each SYNC packet received at the slave, the end-to-end delay was measured and used to generate an experimental probability density function (PDF) of the packet delay, as shown in Figure 2. At low loads (typically less than 45%), most of the packets experienced no queuing in the network, as evidenced by the very strong modes at the minimum delay value. As the load increases, the probability of finding a packet with minimum delay decreases. For example, there is no packet at the minimum delay value at 80% load, as all the packets experience queuing in the network. In fact, the PDFs at higher loads have well-defined shapes and can be fitted to Erlang density distributions.

Inspection of the shapes of the delay profiles in Fig. 2 suggests that the low load scenarios would be amenable to sample-minimum filtering, while the higher load scenarios might benefit from sample-mean filtering.

We also consider the network scenario depicted by Fig. 3, where the background traffic follows the same path as the synchronization traffic. This type of traffic, termed in-line traffic, has practical significance in access aggregation networks such as in mobile backhaul systems where a network controller provides both timing traffic and background traffic (such as voice and data) to base stations. We modelled a 16-hop network in which the size of the packets was uniformly distributed between 64 bytes and 1500 bytes, which are the minimum and maximum packet sizes for Ethernet, respectively. The other simulation parameters were unchanged.

We observed that at 20% load level, the delay PDFs in this network can again be fitted to a theoretical Erlang PDFs, as depicted in Fig. 4; however, none of the packets experienced the minimum delay due to the long chain of queuing elements in the network. As the load increases, most of the packets tend to experience the maximum amount of delay. The resulting PDFs for these increased-congestion scenarios do not fit any popular distributions, but rather resemble mirror-images of the Erlang density distribution.

Recall that an Erlang density function with rate and shape parameters given by λ and k, respectively, can be expressed by:

$$f(x) = \frac{\lambda^k x^{k-1} \exp(-\lambda x)}{(k-1)!};\ x, \lambda \geq 0. \qquad (1)$$

A mirrored-Erlang density can be obtained from (1) by extending the function definition to include negative values of the shape parameter, and shifting the resultant function so that only physically-realistic positive delay values are displayed [20].

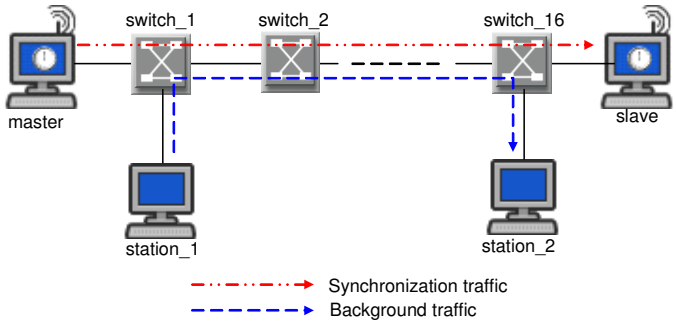

Figure 3. 16-hop network topology for in-line traffic.

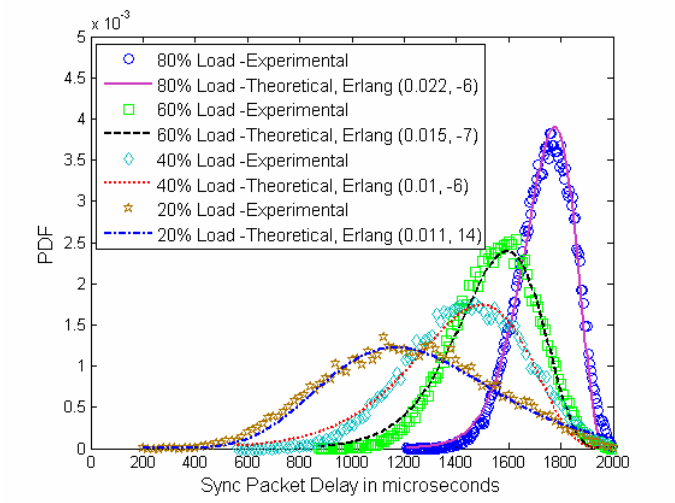

Figure 4. Packet delay distribution for a 16-hop inline traffic network.

Since the delay distributions at higher loads and for higher hop counts tend to follow an Erlang distribution, then in theory, an optimal PDV filter can be designed based on the Erlang distribution parameters. However, the expected computational overhead of such a filter would make it unsuitable for real-time use.

Inspection of the shapes of the delay profiles in Fig. 4 suggests that none of the scenarios would also be amenable to sample-minimum filtering. Sample-maximum filtering might benefit the 80% load scenario, while the other load scenarios might benefit from sample-mean filtering.

As a further analysis of the impact of the network load on the packet delay, we present the mean and variance of the delays in Fig. 5 and Fig. 6, respectively. As expected, the mean packet delay increases with the level of network load for both scenarios; however the cross-traffic scenario exhibits a more obvious linear relationship.

A more interesting result is observed for the variance. For smaller (low hop count) networks at low loads, the variance is low because most of the packets experience little or no queuing. As the load increases, the variance increases slightly to reflect the increased amount of queuing. The converse is true for larger networks. In these networks the probability of finding a packet that experiences no queuing is statistically

small, even at low load levels; hence the variance is high. However, as the load increases most packets tend to experience the maximum delay. Hence, the variance has an inverse relationship with the network load level.

III. PACKET FILTERING TECHNIQUES

As previously mentioned, the performance of a PDV filter depends on the probability of finding "good" packets among a sequence of arriving packets within a window. Thus the aim of PDV filtering is to select packets that experienced similar amount of delay through the network. Three main types of filters have been considered for PDV filtering, namely – sample minimum, sample maximum and sample mean. If the distribution of the delay is known a-priori, then the best filter to use is the one which matches the delay distribution, thus maximizing the chances of finding good packets. However, if the delay distribution does not quite match up with any of the filters, then the residual clock offset after synchronization might be greater than desired.

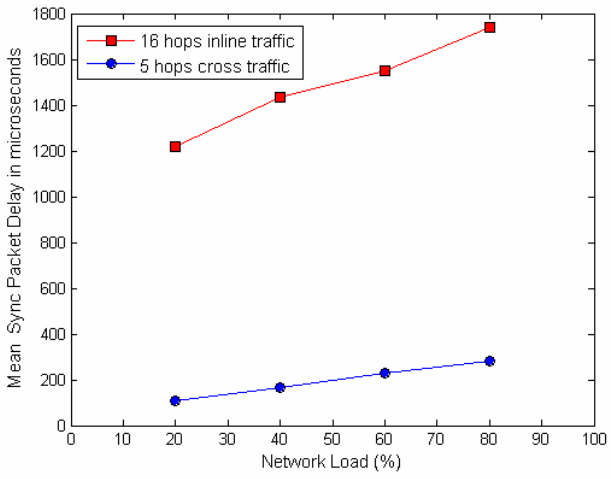

Figure 5. Mean of packet delays for different network load levels

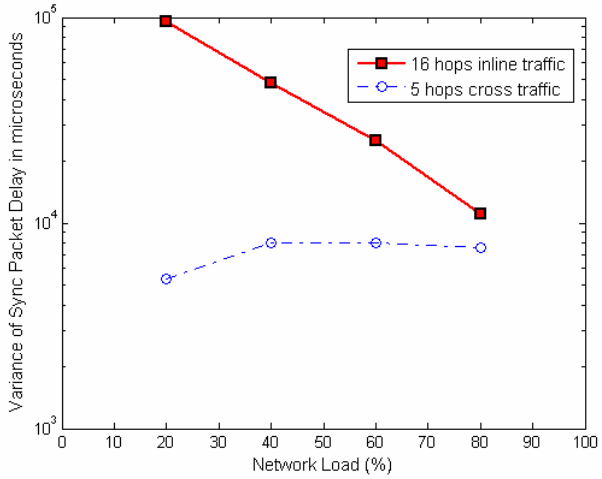

Figure 6. Variance of packet delays for different network load levels

Having characterized the delay profiles of two network topologies, we now review the three existing PDV filters, and propose a new type of filter based on the sample mode.

A. *Existing Approaches*

Given a window with W SYNC packets, master origin timestamp M_i, slave reception timestamp S_i, and delay $\delta_i = S_i - M_i$ for the ith packet ($0 < i \leq W$), a sample-minimum filter defines a good packet as one which satisfies the following:

$$\delta_i \leq \delta_{min} + \alpha. \quad (2)$$

Similarly, a good packet for a sample-maximum filter is one with:

$$\delta_i \geq \delta_{max} - \alpha. \quad (3)$$

For a sample-mean filter, a good packet would satisfy:

$$(\delta_{mean} - \alpha/2) \leq \delta_i \leq (\delta_{mean} + \alpha/2) \quad (4)$$

where δ_{min}, δ_{max}, and δ_{mean} are the minimum, maximum, and mean, respectively, of all W δ_i samples in the window, and the threshold value α depends on the desired accuracy.

B. *Proposed Approach*

The goal of our proposed PDV filter is to maximize the chances of finding "good" packets by selecting packets from within the sample mode bin. The rationale behind this approach is that when the delay distribution does not quite match up with any of the existing filters, the sample mode will yield the largest number of "good" packets. Thus the sample mode filter should give a good fit for all traffic distributions, with a relatively low computational overhead.

For each SYNC packet received at the slave, $\delta = S - M$ is computed from the master origin and slave reception timestamps and stored. After W samples have been obtained, a histogram is created to approximate the delay distribution of the received packets. If two or more bins have the same number of samples, the mode bin is selected as the bin with the minimum delay value. Good packets are defined as packets selected from within the mode bin, i.e.

$$(\delta_{mode} - \alpha/2) \leq \delta_i \leq (\delta_{mode} + \alpha/2) \quad (5)$$

where δ_{mode} is the sample mode and α represents the width of each histogram bin.

The first step in our sample mode PDV filter algorithm is the computation of the rate compensation factor (RCF). The RCF is calculated using from the largest origin timestamp M_{max}, least origin timestamp M_{min}, largest reception timestamp S_{max}, and least reception timestamp S_{min} within the mode bin as follows:

$$RCF = \frac{S_{max} - S_{min}}{M_{max} - M_{min}}. \quad (6)$$

This RCF is used to correct for the mismatched clock rate at the slave with respect to the master clock. After the first computation of the RCF, subsequent computations are done for every frequency synchronization interval, rather than for every iteration. This is due to the well-known fact that changes

in clock rates are usually caused by aging of the oscillators or temperature changes, and as such they tend to occur extremely slowly. In fact, if the maximum clock drift rate ρ is known, then the frequency synchronization interval f can be estimated from the target synchronization accuracy η as follows:

$$f = \frac{\eta}{2\rho}. \quad (7)$$

From (6), it is evident that at least two good packets are required for computing the RCF.

Once the RCF has been computed, subsequent iterations of the algorithm aim to reduce the clock offset between the master and the slave. At least one good packet is required for the offset computation. The packet selection rate is computed as the quotient of the number of packets in the mode bin and the window size W, and then compared with a packet selection threshold. If the selection rate exceeds the threshold level, the window size can be reduced. On the other hand, if no packet is found within the mode bin after increasing the window size to the maximum, the slave sends a management message to halve the synchronization interval at the master.

IV. Simulation and Results

To illustrate our approach and compare its performance with those of the existing filters, we used OPNET Modeler 16.0 [22] to simulate the networks depicted in Fig. 1 and Fig. 3. In order to allow a like-for-like comparison of the filters, we fixed the window size at 500 samples for all filters, the threshold value α was set at 0.2 μs, the synchronization interval was 976.5625 μs and the residual fractional frequency offset at the slave after the initial RCF estimation was 30 parts per billion.

In Fig. 7, we compare the results obtained from the four different filters after synchronization in a 40% 5-hop cross-traffic scenario. As observed in the corresponding delay profile of Fig. 2, most of the packets experience the minimum delay; hence the sample-minimum filter is a good match and performs optimally. Since the mode corresponds to the minimum delay point, the performance of our sample mode filter is also optimal, and is in fact identical to that of the sample-minimum. As expected, the sample-mean and sample-maximum filters perform poorly, because they struggle to find "good" packets.

In Fig. 8, we compare the results obtained after synchronization in an 80% 16-hop inline-traffic scenario. As observed in the corresponding delay profile of Fig. 4, none of the packets experience the minimum delay; hence the sample-minimum filter has the worst performance. The delay profile does not match well with the sample-mean or sample-maximum filters either, hence their performance is also sub-optimal. Thus, the sample-mode filter gives the best performance.

As a final analysis, we can observe the performance of a specific filter in both of these scenarios and see how the performance is influenced by the corresponding delay profile. The sample-maximum filter, for example, yields the worst performance in the first scenario, as most packets experience the minimum delay. On the other hand, it performs better than both the sample-minimum and sample-mean filters in the larger heavier-loaded network scenario which has higher levels of queuing. The converse is true for the sample-minimum filter. The performance of the sample-mean filter is only marginally better in the first scenario, and this marginal improvement can be explained by the fact that the variance of the delay is slightly lower in the first scenario, as illustrated in Fig. 6.

V. Conclusion

We have characterized the IEEE 1588 PTP synchronization packet delay profiles for a small cross-traffic network and a large in-line traffic network with different levels of background traffic and observed from the shape of the distributions that the existing sample-minimum, sample-maximum, and sample-mean filters perform sub-optimally for some of the load levels in these scenarios. A low-computation sample-mode filtering algorithm has been proposed, which selects packets from the mode bin and uses these "good" packets to achieve synchronization. Numerical simulations have shown that when the delay profile is a good match for an existing filter, the sample-mode filter performs as well as the existing filter. When the delay profile is not a good match for an existing filter, the sample-mode filter outperforms the existing filters. Future work will conduct similar experiments for more congested networks, as well as using background traffic characteristics obtained from real networks.

Acknowledgment

The authors gratefully acknowledge EPSRC, TES Electronic Solutions Ltd, and the Industrial Doctorate Centre in Optics and Photonics Technologies at Heriot-Watt University for funding this research. We also thank OPNET for providing the network simulation software free of charge under the University License agreement. C.-X. Wang would also like to acknowledge the support from the RCUK for the UK-China Science Bridges Project: R&D on (B)4G Wireless Mobile Communications and the support of the Opening Project of the Key Laboratory of Cognitive Radio and Information Processing (Guilin University of Electronic Technology), Ministry of Education (Grant No.: 2011KF01).

References

[1] A. Sutton, "Building better backhaul," *Engineering and Technology Magazine,* vol. 6, no. 5, pp. 72 – 75, Jun. 2011.

[2] M. Howard, "Using Carrier Ethernet to Backhaul LTE," Infonetics Research White Paper, Feb. 2011.

[3] P. Briggs, R. Chundury, and J. Olsson, "Carrier Ethernet for Mobile Backhaul," *IEEE Communications Magazine,* vol. 48, no. 10, pp. 94 – 100, Oct. 2010.

[4] X. Cheng et al, "Cooperative MIMO channel modeling and multi-link spatial correlation properties," *IEEE Journal on Selected Areas in Communications,* vol. 30, no. 2, pp. 388 – 396, Feb. 2012.

[5] C.-X. Wang, X. Hong, H.-H. Chen, and J. S. Thompson, "On capacity of cognitive radio networks with average interference power constraints," *IEEE Transactions on Wireless Communications,* vol. 8, no. 4, pp. 1620 – 1625, Apr. 2009.

[6] C.-X. Wang et al, "Cooperative MIMO channel models: a survey," *IEEE Communications Magazine,* vol. 48, no. 2, pp. 80 – 87, Feb. 2010.

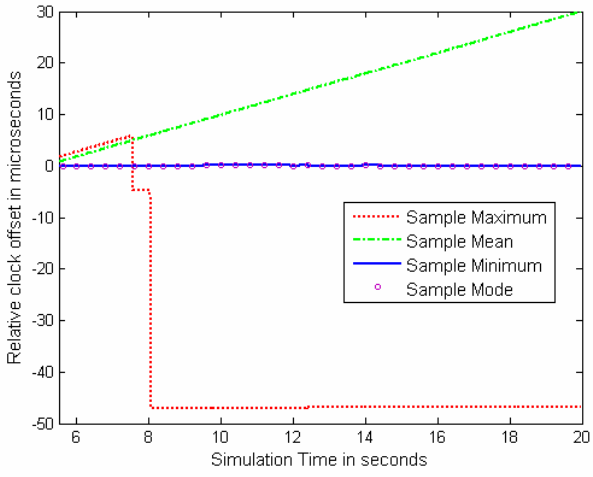

Figure 7. Relative clock offset for 5 hop cross traffic network at 40% load.

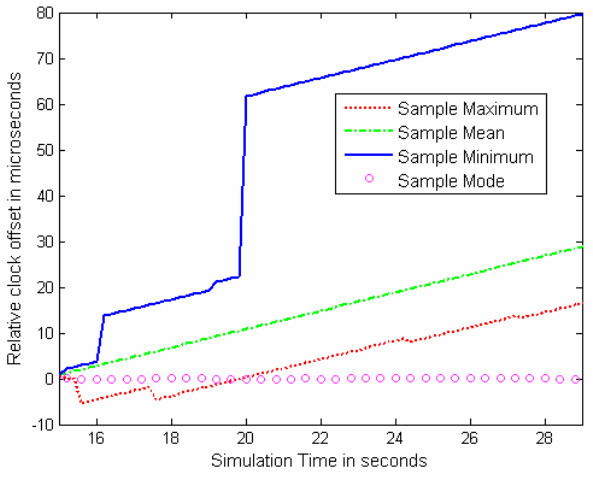

Figure 8. Relative clock offset for 16 hop inline traffic network at 80% load.

[7] X. Hong, C.-X. Wang, H.-H. Chen, and Y. Zhang, "Secondary spectrum access networks: recent developments on the spatial models," *IEEE Vehicular Technology Magazine,* vol. 4, no. 2, pp. 36 – 43, Jun. 2009.

[8] A. Magee, "Synchronization in next-generation mobile backhaul networks," *IEEE Communications Magazine,* pp. 110 – 116, Oct. 2010.

[9] Z. Ghebretensae, J. Harmatos, and K. Gustafsson, "Mobile broadband backhaul network migration from TDM to carrier Ethernet," *IEEE Communications Magazine,* pp. 102 – 109, Oct. 2010.

[10] C. Na, D. Obradovic, and R. Scheiterer, "Method for Synchronizing Clocks in a Communication Network". U.S. Patent Application US 2011/0161524 A1, Jun. 30, 2011.

[11] Timing characteristics of a synchronous Ethernet equipment slave clock (EEC), ITU-T Recommendation G.8262/Y.1362, June 2010.

[12] J.-L. Ferrant et al, "Synchronous ethernet: a method to transport synchronization," *IEEE Communications Magazine,* pp. 126 – 134, Sept. 2008.

[13] D. Mills, "Network Time Protocol (version 4) Protocol and Algorithms Specification", RFC 5905, University of Delaware, June 2010.

[14] IEEE 1588: IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems, 2nd Edition, 2008.

[15] IEEE 802.1AS Standard: Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks, 1st Edition, March 2011.

[16] D. T. Bui, A. Dupas, and M. Le Pallec, "Packet Delay Variation Management for a better IEEE 1588v2 Performance," *ISPCS 2009 IEEE International Symp. on Precision Clock Synchronization for Measurement, Control and Communication*, pp. 75 - 80, Oct. 2009.

[17] T. Murakami and Y. Horiuchi, "Improvement of Synchronization Accuracy in IEEE 1588 Using a Queuing Estimation Method," *ISPCS 2009 IEEE International Symp. on Precision Clock Synchronization for Measurement, Control and Communication*, pp. 12 - 16, Oct. 2009.

[18] C S. Johannessen, "Time Synchronization in a Local Area Network," *IEEE Control Systems Magazine*, vol. 24, no. 2, pp. 61 – 69, Apr. 2004.

[19] I. Hadzic and D. R. Morgan, "On Packet Selection Criteria for Clock Recovery," *ISPCS 2009 IEEE International Symp. on Precision Clock Synchronization for Measurement, Control and Communication*, pp. 35 - 40, Oct. 2009.

[20] I. Hadzic and D. R. Morgan, "Adaptive Packet Selection for Clock Recovery," *ISPCS 2010 IEEE International Symp. on Precision Clock Synchronization for Measurement, Control and Communication*, pp. 42 - 47, Sep. 2010.

[21] Timing and Synchronization Aspects in Packet Networks, ITU-T Recommendation G.8261, April 2008.

[22] OPNET Product Documentation 16.0, www.opnet.com.