On the Impact of Bursty Cross-Traffic on Distributed Real-Time Process Control

Brian Robinson
ABB Inc.
29801 Euclid Ave.
Wickliffe, OH 44092, U.S.A.
Brian.P.Robinson@us.abb.com

Vincenzo Liberatore
Division of Computer Science
Case Western Reserve University
10900 Euclid Ave, Cleveland, OH 44106, U.S.A.
vincenzo.liberatore@case.edu

Abstract

Cross traffic is present in all parts of current networks, both local and wide-area. Experiments are presented which show the effect that different types of cross traffic have on a distributed real-time control application. This cross-traffic takes the following forms: no traffic, continuous high volume traffic, small-burst traffic, large-burst traffic, and Internet traffic. It is concluded that bursty cross traffic has adverse effects on distributed process control even when the average network utilization is low; showing that over provisioning by itself can still lead to poor quality of service.

1. Introduction

Distributed real-time process control relies on network Quality-of-Service to achieve stability and high performance, but the presence of cross-traffic in the network can cause congestion and degrade Quality-of-Service. Process control involves physical dynamics, and so it requires a network with low loss rates and low and predictable delays. For example, late or missing data can jeopardize the stability and performance of a controlled unit [1]. Meanwhile, process control uses a network infrastructure that is shared among multiple flows. The same link can be crossed by multiple sensing and actuation flows, periodic bulk data transfers that implement logging and redundancy, and third-party wide-area traffic. Most networks do not separate different types of traffic and therefore these networks are potentially plagued by poor Quality-of-Service.

Cross-traffic can follow different patterns of behavior, but one frequent trait is that cross-traffic alternates bursts of activity with quiescent periods. For example, operation logs are periodically backed up on a server for redundancy and fault-tolerance, so that bursts of traffic are followed by a period of inactivity ([4, 13], as well as our own experience). In wide-area networks, the aggregate traffic is fundamentally the superposition of a large number of bursty sources, each of which alternates between on and off periods that follow a heavy-tailed distribution. The resulting aggregate traffic is bursty in the sense that it can be modeled as a fractal and self-similar stochastic process with Hurst parameter $H > ½$ [11].

The objective of this paper is to examine the impact of various types of bursty cross-traffic on real-time process control. This paper presents a set of emulations that use commercial off-the-shelf equipment (ABB AC800M) and protocols (MMS [2, 3]) on a local-area network. Additional experiments were run on the wide-area Internet. Our main conclusion is that bursty traffic can degrade process control performance vastly more than continuous traffic even when network utilization is much lower.

This paper is organized as follows. In section 2 a set of experiments are proposed and described. Section 3 presents the experimental results for the paper. Section 4 summarizes related work. Section 5 contains the conclusions of the paper.

2. Experimental Design

A discrete-time PI controller was configured to control a scalar plant. The plant was simulated according to the equation $y(k+1) = \alpha u(k+1) + (1-\alpha) y(k)$, where $y(k)$ is the plant output at step k, $u(k)$ is the plant input, and α is the plant gain. The experiments were run over a range of values for α from 0.1 to 0.3 in 0.05 increments. We measured the *settling time* as the time it takes for the output y to approximately reach and stay at the set point 50 (±1%) from an initial value $y(0)=0$. In addition to settling time, the *overshoot* was measured as the maximum value over the set point that the process attains when converging to the set point, normalized as a percentage of the set point. The PI gain β will take the values 1.0, 2.0, and 3.0, and the integral gain was left at the default values of 20. For each choice of the α and β parameters, experiments are run twenty times. Then, the average and standard deviations of the settling times and overshoot are calculated and recorded.

The controller and simulated plant run on ABB's *AC800M Ethernet* units. These units are the state-of-the-art distributed process controllers available from ABB. The AC800M uses the *Manufacturing Message Specification (MMS)* [2, 3] to pass data values between controllers. MMS is an international standard protocol, defined by ISO 9506 [2, 3]. The AC800M implementation of MMS uses TCP as the transport protocol and IP as the network protocol, which is a standard for use with Ethernet [6, 7]. These controllers communicate at a data rate of 10Mbps. The two controllers are configured to execute on a 100 ms *cycle time*, which is the period for the controller to schedule its pending tasks. For example, the plant simulator schedules the computation of $y(k+1)$ after 100ms since the time when the computation of $y(k)$ was scheduled. The only exception to this rule is that I/O and network communication are scheduled more frequently on a 50ms *scan cycle*, even when a scan cycle could have no data to send or receive. A ratio of 2:1 is an ABB recommended ratio for these units.

Another component of our network is two Linksys 10/100 switches that have the capability of monitoring the total amount of traffic crossing a link and report it for every one second interval. All switches were configured to operate at 10Mbps on all links. The last component of the network is two PCs that generate cross-traffic. The resulting topology is shown in Figure 1.

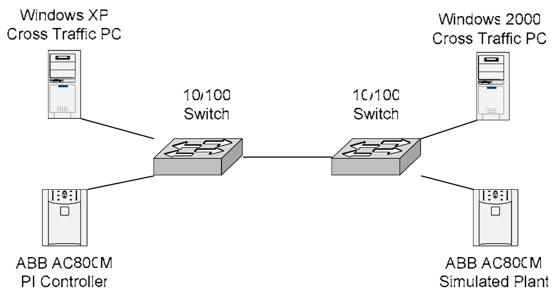

Figure 1. Network Topology.

Cross-traffic takes the following forms: no traffic, continuous high volume traffic, bursty traffic, and random wide-area Internet cross-traffic. The no-traffic experiment serves as a control experiment to show how the process runs when minimal jitter is present on the network. The continuous high volume traffic experiment gauges process convergence with cross-traffic generated by one continuous TCP flow. The bursty traffic experiment involves running the process in the presence of varying traffic that sends data in bursts across the network.

Moreover, a set of experiments was conducted over various wide-area distances, with the PI controller being placed remotely. The first experiment involved placing the units on separate subnets, which were at a distance of 2 hops. The next experiment involved placing the PI controller in Rochester, NY and the simulated plant in Wickliffe, OH, which lead to a distance of 5 hops between the units. The final experiment involved moving the PI controller to Vasteras, Sweden, which was at a distance of 9 hops. These routes remained consistent throughout the test, as verified by running traceroute continually during the experiments. At each location, all α and β values were tried. A fundamental difference between the local and the remote set of experiments is that remote experiments are subject to Internet cross-traffic that was generated by third-parties independently of the experiments being run.

Round-trip times (RTT) were calculated continuously throughout the run of each experiment, and summarized as the average and standard deviation. The average gives a nominal RTT and the standard deviation is a measure of delay jitter.

3. Experimental Results

3.1 Local Experiments

In the first set of experiments, the switches (Figure 1) were directly connected. The only traffic on the link between the switches is the cross traffic that is intentionally injected as part of the experiments below. No third-party traffic was present.

3.1.1 No Traffic

The first experiment served as the baseline for our control set-up. This experiment was conducted with the controllers directly connected through the two switches, and with no cross traffic being generated from the PCs. Because of the lack of cross traffic, there was no jitter, so that any variability in the settling time and overshoot is due to the local processes. Figures 2 and 3 show the average and standard deviation of the settling times and overshoot for each α and β value. The results are shown for the α and β values that had all twenty data runs converge within the 10 minute duration of the experiments.

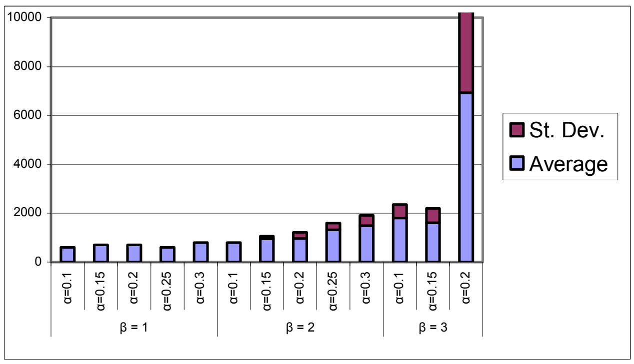

Figure 2. No Cross Traffic Settling Times

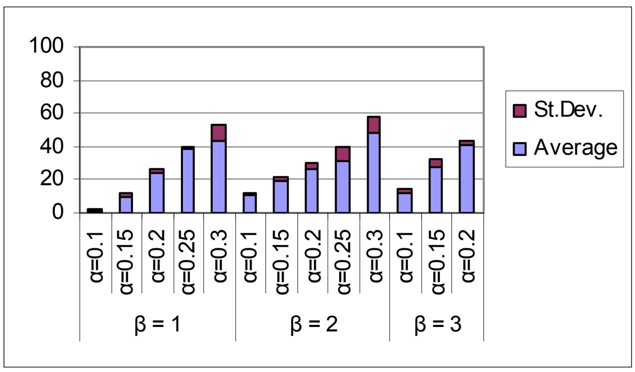

Figure 3. No Cross Traffic Overshoot

In these baseline experiments, the average round trip time was 3 milliseconds and the standard deviation was 0. When β was set to 1.0, the process was very repeatable and the standard deviation of the settling times was 0 and the standard deviation of the overshoot was small, except in the case where α = 0.3. When β was set to 2.0, the data shows that there is some variation in both settling times and overshoot, and that it increases with α. For a β setting of 3.0, the standard deviation for the settling time is larger, while the standard deviation for overshoot is small. In particular, if β=3.0 and α>0.2, the process never converged to the set point within the first 10 minutes. We will say that a process is *out of control* if its output *y* oscillates widely around the set point within every few controller scan cycles without converging to the set point. When β=3.0 and α>0.2, the process was out of control.

3.1.2 Continuous FTP Traffic

The second experiment dealt with additional cross-traffic on the link between the switches. This cross traffic was created by a large, continuous FTP transfer between the two PCs. The continuous cross-traffic caused the network utilization to reach more than 80%, as measured by the switches. For comparison, traditional wisdom suggests that Ethernet should not be utilized over 30% [8].

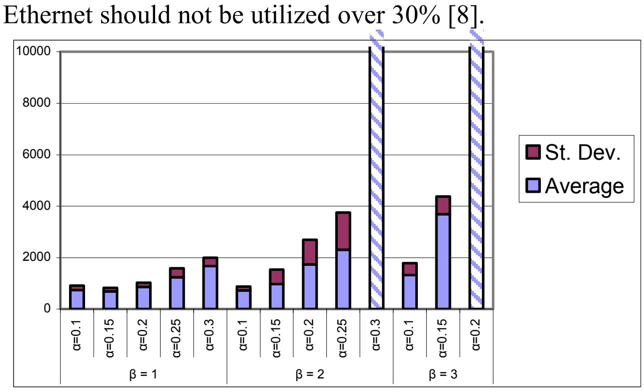

Figure 4. Constant Cross Traffic Settling Times

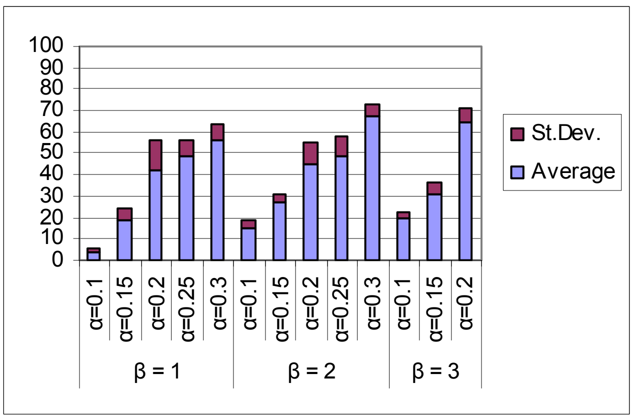

Figure 5. Constant Cross Traffic Overshoot

Figure 4 shows the average settling times and their standard deviation, and Figure 5 shows the average overshoot and their standard deviation. The striped bars on the graph represent experiments in which some trials were out of control. The RTTs for this data set have an average of 16.45 milliseconds and a standard deviation of 18.1 milliseconds (i.e., the standard deviation was larger than the average). With β=1.0 there is some variation in the settling times, which can be compared with the no-traffic case (Section 3.1.1 No Traffic) were no variability was present. The overshoot also had more variation within the experiments compared to the no-traffic case. This data set shows a standard deviation of between 130 and 330 milliseconds. Once β was set to 2.0, the process settling time became even more variable, and the average overshoot increased, compared to the no-traffic case. The process with α > 0.2 had a high standard deviation in settling time. In the case β=3.0, all of the α≥0.2 processes were out of control.

3.1.3 Small Burst Traffic

The third experiment involved subjecting the simulation to bursty cross traffic. This type of traffic is characterized by large amounts of data sent out in short intervals, which alternate with no-traffic intervals. Bursts were created by a small program that generates TCP traffic, which is then sent out in bursts at periodic intervals set by the user. The first experiment of this type (denoted as *small bursts*) involved sending out 50 kilobyte bursts every 150 ms. These bursts of traffic led to an average network utilization between 12% and 25%, as measured by the switches.

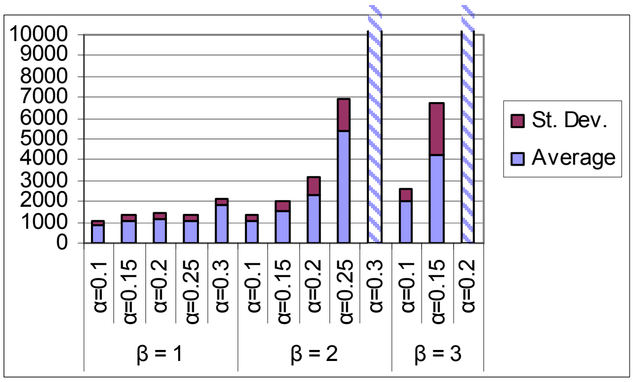

Figure 6. Small-Burst Cross Traffic Settling Times

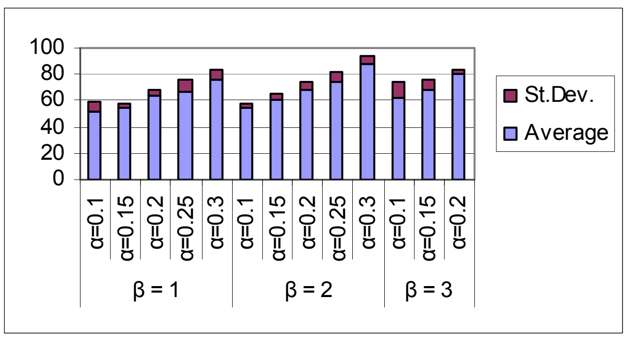

Figure 7. Small-Burst Cross Traffic Overshoot

Figure 6 shows the average settling times and their standard deviation, and Figure 7 shows the average overshoot and their standard deviation. For this experiment, the average round trip time was 24.6 milliseconds and the standard deviation was 18.46 milliseconds. Just like with continuous traffic, it is evident that small-burst traffic has a large effect on the settling times and overshoot of the distributed process. This can be seen by comparing Figures 6 and 7 with Figures 2 and 3.

An important comparison is to look at this bursty traffic against the constant FTP traffic. By comparing Figures 6 and 7 with Figures 4 and 5, the result shows that, in almost every case, the bursty traffic had at least as large impact on settling times as the constant FTP traffic. In addition, the overshoot was close to, and in some cases larger, than for the constant traffic. In general, settling times are longer, more variable, and containing as much overshoot with small-burst traffic than with constant FTP traffic. We conclude that average network utilization can be a poor predictor of control performance. In contrast to widespread claims (e.g., [9]), pure network over-provisioning can lead to unacceptable Quality-of-Service.

3.1.4 Large Burst Traffic

Another experiment was conducted with a different type of bursty traffic. In this experiment (denoted as *large bursts*), the burst size was increased from 50Kbytes to 100Kbytes on the same 150 ms interval. The larger burst size lead to an average network utilization of 25% to 50% during the experiment.

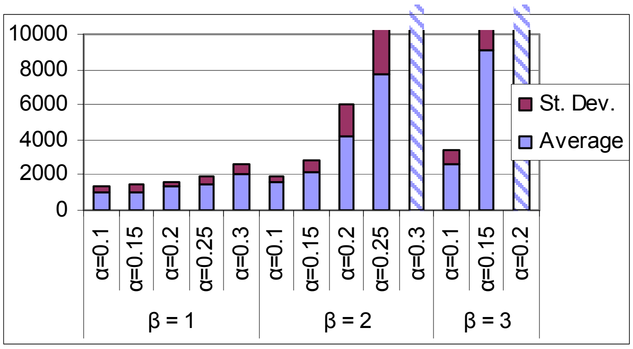

Figure 8. Large-Burst Traffic Settling Time

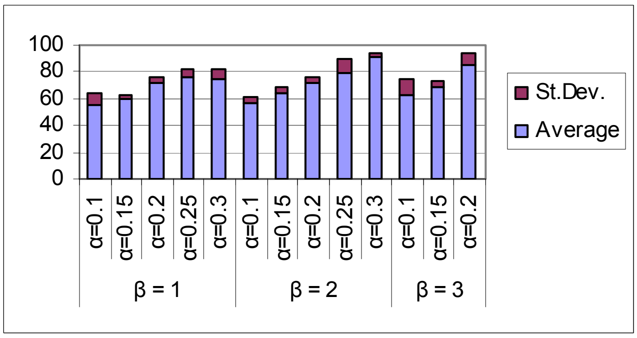

Figure 9. Large-Burst Traffic Overshoot

This experiment had an average RTT of 40.7 milliseconds and a standard deviation of 27.42 milliseconds. The experiments with large and small bursts can be compared as follows. First, large-burst experiments took longer on average to converge to the set point and contained more overshoot than the low-burst experiment. Moreover, the standard deviation in settling times and overshoot is larger in the presence of the additional burst traffic. For the large-burst traffic, almost all of the control processes had a larger standard deviation for settling time than for both the low-burst traffic and the continuous FTP traffic, in many cases by a factor of 2.

3.2 Multiple Hop Distance Experiments

The next set of experiments deals with Internet cross-traffic. All of the previous experiments were conducted on a local area context, where the hop distance was 1. For these next experiments, hop distances of 2, 5, and 9 were adopted. The controllers were also subjected to uncontrolled traffic, as the controllers were placed directly on the ABB corporate network and the end-to-end paths included non-ABB links. In particular, the 9-hop path traverses an intercontinental link.

Extensive Internet measurements over the course of a decade have proved that wide-area traffic is indeed self-similar [5], and so it can be safely assumed that all of the multi-hop experiments encounter bursty cross-traffic. On the other hand, the nature of these wide-area experiments prevents us from explicitly setting the burstiness level of the cross-traffic and from measuring bandwidth utilization on the bottleneck link in the end-to-end path. Finally, the experimental set-up does not allow us to distinguish between the effects of larger nominal RTTs, delay jitter, and drop rates. As a consequence, the results of this section are to be viewed primarily as a descriptive overview of control performance over wide-area networks with bursty cross-traffic.

3.2.1 2-hop Distance

The first multi-hop experiment was conducted over 2 hops. The outcome from this experiment is shown in Figures 10 and 11.

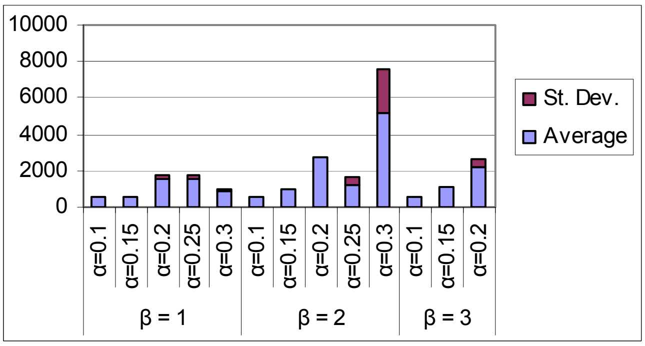

Figure 10. 2-Hop Distance, Internet Cross-Traffic Settling Time

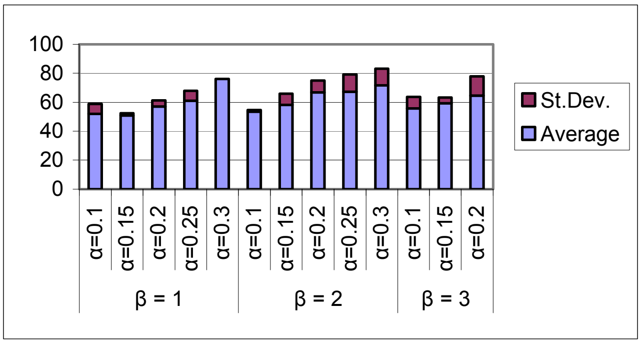

Figure 11. 2-Hop Distance, Internet Cross-Traffic Overshoot

The average round trip time was 10.3 milliseconds and the standard deviation was 1.342 milliseconds. These results show that for lower α values, no variation in settling time exists, while a large variation in overshoot exits. In addition, some variation is evident at higher α values, but it is not consistent between β values. This is due to the traffic characteristics of this 2-hop network. While no traffic statistics were available, the end-to-end path involved only two lab subnets with no active experiments on them at the time the experiment was run. Therefore, this network did not have enough users on it to exhibit large traffic volumes or large traffic bursts. It is only on occasion that there are cases of heavy traffic, as users sign on and request data. Therefore, we believe that the variability of settling times and overshoot is due to sporadic cross-traffic more than to any specific aspect of these experiments.

3.2.2 5-hop Distance

The next multi-hop data set used a distance of 5 hops between controllers. The remote controller was located in Rochester, NY and the local controller was in Wickliffe, OH. The data for this experiment is shown in Figures 12 and 13.

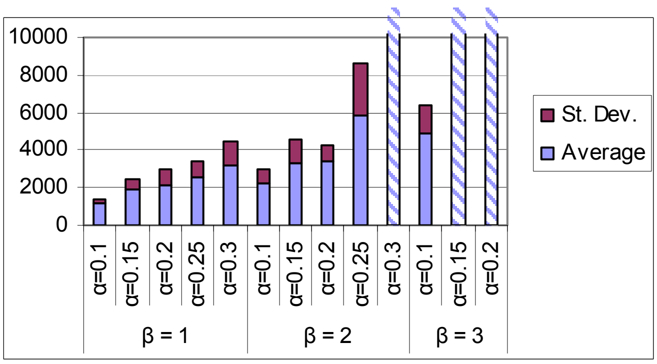

Figure 12. 5-Hop Distance, Internet Cross Traffic Settling Time

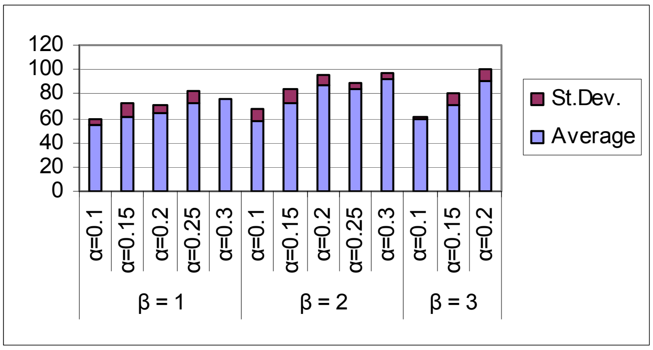

Figure 13. 5-Hop Distance, Internet Cross Traffic Overshoot

The average round trip time for this experiment was 39.95 milliseconds and the standard deviation was 8 milliseconds. Comparing the data shown in Figure 7 with the control experiment, it is evident that this experiment exhibits much more variation than the baseline. When β=1.0, the control experiment had no variation whereas the 5-hop data shows a large amount of variation in settling times. In addition, some processes were in control under no-traffic but are now out of control. For example, with β=2.0 and α=0.25, 0.3, the control experiment showed standard deviations in settling time of 200 and 400 ms. Using the same β and α values with the 5-hop distance between controllers, the variation in settling time increases to 2.7 seconds and 77 seconds respectively and these processes are out of control. In total, there are 3 processes that are out of control in the 5-hop data set that were in control in the no traffic case.

3.2.3 9-hop Distance Experiment

The final experiment run was conducted over a 9-hop distance. This involved locating the remote controller in Vasteras, Sweden and keeping the local controller in Wickliffe, OH. The data is shown in Figure 14.

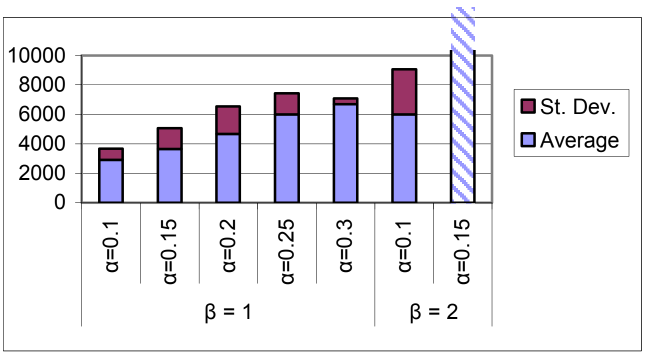

Figure 14. 9-Hop Distance, Internet Cross Traffic Settling Time

For this experiment, the average round trip time was 145.3 milliseconds and the standard deviation was 15 milliseconds. (Note that there is no overshoot data for the 9-Hop distance experiments. The controller in Sweden could not be used a second time to collect overshoot data.) The 9-hop experiment showed the most dramatic variation of all the experiments conducted. By comparing Figure 14 with both the 5-hop data and the control experiment show that the 9-hop data is by far the worst data set of the group. This data includes 5 new processes that did not ever converge in the 9-hop tests that converged previously. In addition, one process (β=2.0, α= 0.15) that was in control went out of control. Again, both the average settling times and their standard deviation increased between the 5 and 9-hop data sets.

4.) Related Work

The evidence for traffic burstiness is well-known, both on the factory floor [4, 13] and on the wide-area Internet [11]. Bursty cross-traffic implies longer queuing delays than Poisson traffic and justifies either higher levels of bandwidth provisioning or network QoS, but it cannot be solved by increasing buffers [14]. However, the wide-area context sees large-scale flow aggregation, which can be paired with bandwidth over-provisioning to improve network QoS [11]. The implications for Internet Service Providers are discussed in [10].

Poor network QoS can impact real-time process control [11, 12]. In particular, the effects of constant FTP traffic on process control have been discussed in [1].

5.) Conclusions

The goal of this paper was to show the effects that different forms of bursty cross-traffic have on distributed process control. It was concluded that cross-traffic can have a large detrimental effect on the process control algorithm even when average network utilization is low. The finding puts in perspective various claims that pure over provisioning can lead to high levels of Quality-of-Service.

Future work can expand these results in several directions. First, additional experiments can be run with additional forms of cross-traffic. Second, additional experiments could use real process control applications, which would lead to a comprehensive characterization of process-level impact. Next, an analysis could investigate the relationship of the controller scan cycle, the delay jitter, and the round trip time. Finally, novel sampling and control algorithms could be developed to understand and handle the jitter encountered in distributed process control.

Acknowledgments

This work was supported in part under NSF grant CCR-0329910.

References

[1] M.S. Branicky, V. Liberatore, and S.M. Phillips. “Networked Control System Co-Simulation for Co-Design”, *American Control Conference*, 2003.

[2] ISO 9506-1. “Industrial Automation Systems – Manufacturing Message Specification – Part 1: Service Definition”. 2003.

[3] ISO 9506-2. “Industrial Automation Systems – Manufacturing Message Specification – Part 2: Protocol Specification”. 2003.

[4] W. Feng, D. Kandlur, D. Saha, and K. G. Shin. “Understanding and Improving TCP Performances Over Networks with Minimum Rate Guarantees”, IEEE Transactions on Networking, April, 1999.

[5] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, “On the selfsimilar nature of Ethernet traffic (extended version),” IEEE/ACM transactions on Networking, 1994.

[6] E. Gressier-Soudan, M. Lefebvre, M. Rozier, “Protocol of Transport MMS on Chorus: Various Approaches”, *RTS’96,* Paris, January 1996.

[7] Weis, F. “An approach for the Integration of Applications MMS”, *CNAM Doctoral Thesis,* Paris, 1996.

[8] Spurgeon, Charles. “Ethernet: The Definitive Guide”, O’Reilly Publishing, February 2000.

[9] C. Fraleigh, F. Tobagi and C. Diot, “Provisioning IP backbone networks to support latency sensitive traffic”, Proceedings IEEE Infocom 2003, San Francisco CA, USA, 2003.

[10] Leer, C. “Provisioning for Bursty Internet Traffic: Implications for Industry and Internet Structure”, *citeseer.ist.psu.edu/444266.html*, December 1999.

[11] W. Zhang, M.S. Branicky, and S.M. Phillips. “Stability of networked control systems”, *IEEE Control Systems Magazine*, 21 (1), pp 84-99, 2001.

[12] M.S. Branicky, S.M. Phillips, and W. Zhang. “Scheduling and Feedback co-design for networked control systems”, *Proceedings of the IEEE CDC*, Las Vegas, 2002.

[13] S. Kweon, K. G. Shin, and G. Workman. Achieving Real-Time Communication over Ethernet with Adaptive Traffic Smoothing. *Proceedings of the Sixth IEEE Real Time Technology and Application Symposium (RTAS)*, 2000.

[14] I. Norros. On the Use of Fractional Brownian Motion in the Theory of Connectionless Network. *IEEE Journal on Selected Areas in Communications*, 6(13), 953-962, August 1995.