perfSONAR: Enhancing Data Collection through Adaptive Sampling

Ali Mazloum, Ali AlSabeh, Elie Kfoury, Jorge Crichigno
College of Engineering and Computing, University of South Carolina, Columbia, U.S.A.
amazloum@email.sc.edu, aalsabeh@email.sc.edu, ekfoury@email.sc.edu, jcrichigno@cec.sc.edu

***Abstract*—perfSONAR is a tool used to monitor and troubleshoot problems in high-speed networks such as Science Demilitarized Zones (DMZs). It is essential to validate that data transfers are performing as expected. However, perfSONAR suffers from the trade-off between the measurement accuracy and the overhead induced by its active tests.**

This paper presents a scheme that offloads the traffic monitoring to a programmable data plane (PDP) switch. The scheme integrates a PDP switch with perfSONAR, where the switch continuously collects network measurements (e.g., latency, throughput, packet loss rate) and periodically reports the measurements to the perfSONAR archiver. This integration significantly enhances the granularity, visibility, and troubleshooting capabilities of perfSONAR. Additionally, the scheme automates the reporting period according to the variability of the monitored measurements, which eliminates the need of human intervention observed in today's networks. In contrast to traditional schemes that report all measurements, the proposed approach uses the Linear Prediction (LP) method to only report the samples that reveal a variation on the measurements. Experimental results show that the system reduces the number of reports by five times under stable network conditions and sustains a relative mean error (RME) below 0.06.

***Index Terms*—Programmable Data Planes, P4, Linear Prediction, Adaptive Sampling, perfSONAR, Science DMZ.**

I. Introduction

perfSONAR is a monitoring tool used on campus network, Internet service providers (ISPs), research and education networks (RENs), and high-speed networks in general, including Science Demilitarized Zones (DMZs) [1]. perfSONAR uses active tests to synthetically generate end-to-end traffic and report measurements based on these tests. The measurements are used to infer potential network issues [2]. Despite its essential role and widespread deployment, perfSONAR suffers from a trade-off between the provided visibility and the induced overhead to the network [3]. Additionally, the measurements collected by perfSONAR are based on the synthetic traffic, which may not accurately represent the actual network traffic. Furthermore, the synthetic data may even interfere with the actual network traffic, causing disruptions and artificial packet bursts.

This paper presents a scheme that offloads the traffic monitoring to a programmable data plane (PDP) switch, integrated to perfSONAR. The scheme generates fine-grained measurements by passively monitoring real traffic, without incurring unnecessary overhead. Through this integration, the visibility, granularity, accuracy, and troubleshooting capabilities of perfSONAR are greatly enhanced [3].

A seamless integration between perfSONAR and the PDP switch is accomplished by satisfying two design goals: (1) using the perfSONAR archiver to store the measurements, (2) using the perfSONAR configuration daemon to configure the PDP switch at run time. To satisfy the first requirement, the control plane of the PDP switch normalizes the raw measurements extracted from the data plane and stores them directly on the archiver's database through its application programming interfaces (APIs). To satisfy the second requirement, a command is added to pSConfig, the configuration daemon of perfSONAR. The PDP switch aggregates the measurements and periodically reports them to the perfSONAR archiver. The added command is used by the administrator to configure the rate at which the PDP switch reports the measurements. Note that a high reporting rate implies less aggregation, leading to higher accuracy at the cost of increasing storage and processing overhead. On the other hand, a low reporting rate implies more aggregation, leading to lower accuracy but decreasing storage and processing overhead.

The proposed system is flexible enough and allows the network administrator to 1) adjust the reporting rate at run time through pSConfig, or 2) automate the reporting rate according to the variability of the monitored measurements. The second approach uses the Linear Prediction (LP) method [4]: instead of reporting all the measurements extracted by the data plane, the control plane only reports the samples that reveal a variation in the traffic.

The contributions of this paper are summarized as follows:

- Eliminating human intervention. The system enables the administrator to adjust the reporting rate automatically.
- Increasing the accuracy of the reports by lowering the aggregation at the data plane level. The control plane can extract the measurements from the data plane at high rates without flooding the archiver by selectively reporting samples.
- Reducing the number of reports by five times under stable network conditions and sustaining RME below 0.06 under unstable network conditions.
- Publishing the P4 code that generates fine-grained measurements on the PDP, and the Python implementation that implements the adaptive model. They can be accessed via [5].

This work was supported by the U.S. National Science Foundation, Office of Advanced Cyberinfrastructure, Award #2118311.

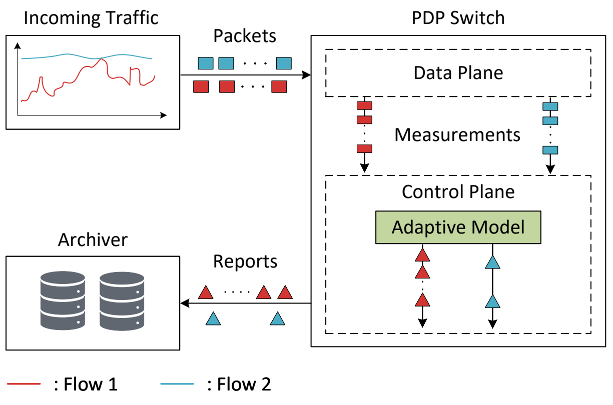

Fig. 1. System Overview.

The remainder of the paper is organized as follows. Section II provides background on programmable data planes, conventional sampling, and adaptive sampling techniques and clarifies how the proposed system differs from existing work. Section III describes the proposed approach. Section IV presents the experimental results. Section V discusses the limitations of the proposed approach and the future work in this area. Section VI concludes the paper.

II. Background and Related Work

A. Programmable Data Planes

Data plane programmability allows the developer to parse custom headers, define the processing behavior of the packets, measure the events occurring in the data plane with high precision, and report the events to the control plane with nanosecond resolution [6]. The Protocol Independent Switch Architecture (PISA) is a widely adopted data plane programming model [7]. It is composed of a programmable parser, a programmable match-action pipeline, and a programmable deparser [8]. The programmable parser operates as a state machine to parse the headers. The match-action pipeline processes the packets based on the logic defined by the P4 program. P4, or Programming Protocol-independent Packet Processors, is the de-facto language to program the data plane [9]. The programmable deparser serializes the headers and prepares them for transmission.

B. Adaptive Sampling

Adaptive sampling methods adjust the sampling rate based on the observed measurements. In general, adaptive techniques are based on fuzzy logic or LP [4]. In fuzzy logic-based systems, the sampling rate is adjusted by considering similar past experiences [10]. On the other hand, LP-based systems construct a model from the observed traffic. The model is then used to predict future measurement values. If the prediction is accurate, the sampling rate is reduced as the model has captured the current traffic pattern. On the contrary, inaccurate prediction increases the sampling rate to detect the new traffic pattern [11]. A common weakness among adaptive systems is that they ignore the unsampled measurements, and consequently, the network behavior is not monitored during the time interval between samples [4, 12].

TABLE I
Rules to define the next interval between samples in LP.

Current m	ΔT_{next}
$1-\sigma < m < 1+\sigma$	$\Delta T_{current}$
$m < 1-\sigma$	$max(\Delta T_{min}, m\Delta T_{current})$
$m > 1+\sigma$	$min(\Delta T_{max}, \Delta T_{current}+2)$

The proposed system deploys an adaptive sampling model based on the LP method. It utilizes a PDP switch to have continuous access to the measurements, such that the sampling decision is taken on each measurement extracted from the data plane. The main distinction between the existing work and the proposed system is that the system considers both sampled and unsampled measurements to adjust the adaptive model and continuously monitors the network.

III. Proposed System

A. Overview

The proposed system aims to reduce the load at the archiver through adaptive sampling. The data plane of the PDP switch collects per-packet measurements. It monitors per-flow throughput, packet losses, Round Trip Time (RTT), and the border router's queue occupancy. The control plane periodically extracts the measurements from the data plane. Instead of reporting all the measurements to the archiver, the control plane samples the measurements that reflect the behavior of the traffic. As shown in Fig. 1, the control plane deploys an adaptive sampling model. This model adjusts the measurement reporting rate based on the traffic variation.

Before discussing the adaptive model, it is essential to define some terminologies:

- Extraction rate is the rate of extracting or pulling measurements from the data plane. This rate is constant.
- Observations are the measurements extracted from the data plane.
- Samples are the observations selected by the model to be sent to the archiver for storage.
- Reports are the samples normalized by the control plane for the archiver to be able to ingest them.
- Reporting rate, also referred to as sampling rate, is the rate of sending reports to the archiver. The model dynamically adjusts this rate.

B. Adaptive Model

LP is an adaptive sampling method based on the Linear Prediction Coefficient (LPC) technique [4]. LPC is used by the Transport Control Protocol (TCP) to predict the RTT of the packets [13]. It employs a low pass filter that predicts future values based on the recent value and the average of the last observed N values, where N is the order of the filter. The low pass filter is represented by the following formula:

$$x_p = a \cdot R + (1-a) \cdot Av$$

The equation predicts the next sample x_p based on the most recent value R and the average of the previous N samples Av, using a coefficient a within the range of 0 to 1. Higher values of a give more weight for the recent value.

Unlike LPC, LP does not employ the coefficient a. Besides, LP uses the average rate of change of the last N samples instead of their average value as indicated in the equation below:

$$x_{p_{N+1}} = x_N + \frac{\Delta T_{current}}{N-1} \sum_{i=1}^{N-1} \frac{x_{i+1} - x_i}{t_{i+1} - t_i}$$

Where:

- $x_{p_{N+1}}$ is the predicted value for the next sample.
- x_N is the most recent sample.
- N is the number of samples considered.
- x_i and t_i represent the value and arrival time of the ith sample, respectively.
- $\Delta T_{current}$ is the time gap between the most recent sample and the next sample.

When a new observation arrives, the LP model calculates the error m between the predicted and observed values. m is calculated by dividing the predicted value over the expected value: $m = \frac{x_{p_{N+1}}}{x_{N+1}}$, where x_{N+1} is the observed value. If m is within σ from 1 (i.e., $1-\sigma < m < 1+\sigma$), the predicted value is considered close enough to the actual value and the sampling rate can be reduced. Otherwise, the recent observation does not follow the pattern detected by the model, and the sampling rate should increase to detect the new pattern. Table I shows how the future intervals between samples are defined based on the error. ΔT_{next} is the time gap between the next sample and the sample that follows (the value of the next $\Delta T_{current}$). ΔT_{max} and ΔT_{min} are the maximum and the minimum allowed time intervals between samples, respectively.

In the proposed system, the data extraction rate is constant, and consequently, the inter-arrival time between consecutive observations (ΔT) is constant. The model predicts the value of the next observation instead of the next sample as follows:

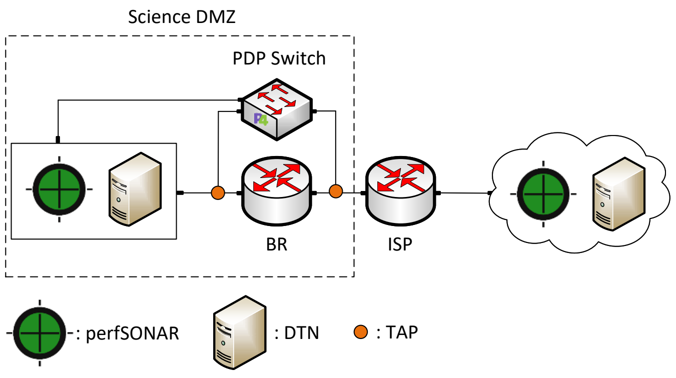

Fig. 2. Experimental Topology.

TABLE II
RULES TO DEFINE THE NEXT INTERVAL BETWEEN SAMPLES IN THE PROPOSED SYSTEM.

Current m	ΔT_{next}
$1-\sigma < m < 1+\sigma$	$min(\Delta T_{max}, \Delta T_{current} + 1)$
$m < 1-\sigma$	ΔT_{min}
$m > 1+\sigma$	ΔT_{min}

$$x_{p_{N+1}} = x_N + \frac{\Delta T}{N-1} \sum_{i=1}^{N-1} \frac{x_{i+1} - x_i}{\Delta T}$$

$$= x_N + \frac{x_N - x_1}{N-1}$$

For each flow monitored by the data plane, the proposed adaptive model maintains a vector x for the values of the last N observations, t the timestamp of the last sample, $\Delta T_{current}$ the time difference between the recent sample and the next sample, and $x_{p_{N+1}}$ the prediction of the next observation. When a new flow is detected by the PDP switch, the corresponding $\Delta T_{current}$ is set to ΔT_{min}. The error m is only calculated if $N \geq N_{min}$, where N_{min} is a constant representing the minimum number of observations required to predict future values. The vector x can hold at most N_{max} observations, where N_{max} is a variable dependent on σ and on the average change of measurements.

The time intervals between samples are adjusted based on the rules described in Table 2. If the error is within σ from unity, $\Delta T_{current}$ is incremented by 1 second. Otherwise, $\Delta T_{current}$ is set to ΔT_{min} and the vector x is emptied. The time interval between samples is minimized (i.e., the sampling rate is maximized) because a disruption in the flow behavior is detected. The vector x is emptied because the new behavior is assumed unrelated to the previous behavior (i.e., the worst-case scenario is assumed). One advantage of this assumption is that the model can adapt faster to the new behavior. Another advantage is that anomalies are automatically discarded by the model and not used to predict future values. Furthermore, this assumption bounds the maximum difference between consecutive observations (d) used to predict future values (Appendix - 1).

The adaptive model aims to minimize the number of sampled measurements while sustaining low relative mean error (RME). RME measures the difference between sampled values and the total values compared to the average of all total values. Denote by x_{N+1} the next observation to be sampled by the adaptive model. For each x_{N+1}, its RME is calculated as follows:

$$\frac{|\ x_{N+1} - \mu\ |}{\mu}$$

where $\mu = \sum_{i=1}^{N+1} \frac{x_i}{N+1}$. If the observation x_{N+1} is reported based on $\Delta T_{current}$, the upper bound on the RME is deter-

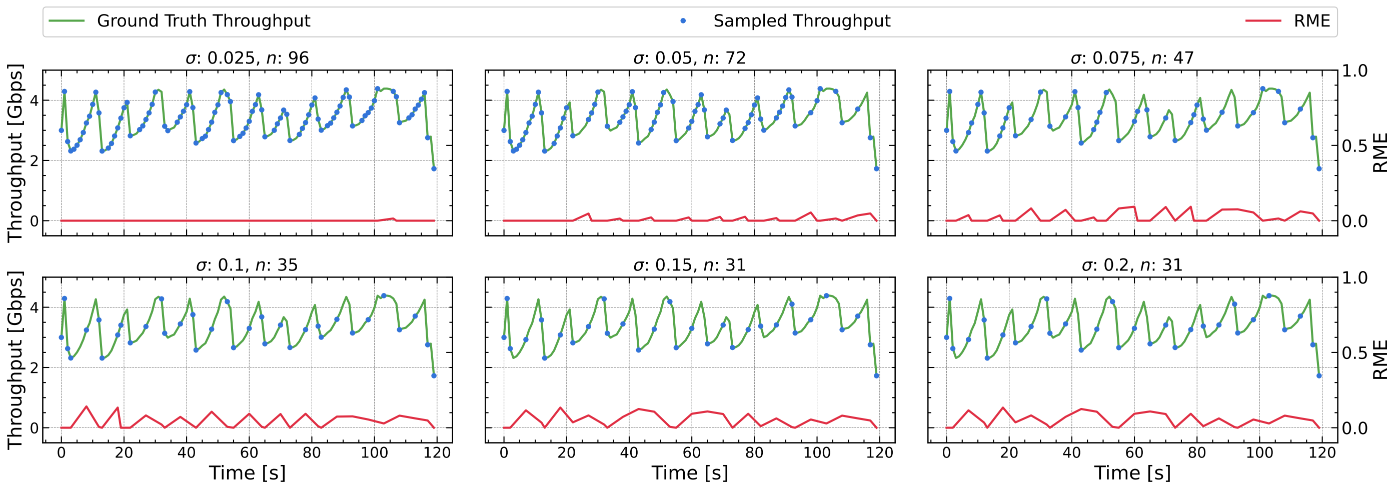

Fig. 3. The number of samples and RME with respect to different values of σ.

mined by N, σ, and x_1 (Appendix - 4). If the observation x_{N+1} is sampled because its error m is not within σ from unity, then:

$$| d_{N+1} | \nleq max(| \frac{\frac{x_N - x_1}{N} + \sigma x_N}{1-\sigma} |, | \frac{\frac{x_N - x_1}{N} - \sigma x_N}{1+\sigma} |)$$

and consequently, (Appendix - 4) does not hold.

C. *Proactive Rate Adjustment*

Reactive adaptive systems adjust the sampling rate after the pattern of traffic changes. On the other hand, proactive adaptive systems adjust the sampling rate before the deviation in the traffic pattern is observed. LP method is a reactive system as it adjusts the sampling rate based on the observed error between the predicted value and the real value.

A main feature of the proposed system is that the adaptive model has access to the recent N observations rather than the recent N samples. If observation x_{N+1} reveals a deviation in the traffic behavior, the model can send the previously unreported observations from vector x, imitating the proactive functionality. Although reporting the elements of vector x (i.e., recent N observations) is equivalent to proactively adjusting the data reporting rate to ΔT_{min} before observing the deviation in the behavior, it imposes unnecessary utilization of resources.

From any vector x, at most three elements have been reported to the archiver. In the best-case scenario, x_N is the last reported element. In such a case, no further action is required as the previous traffic pattern has already been reported, and the RME is bound from above by (Appendix - 4). In the worst-case scenario, $x_{N_{min}}$ is the last reported element (x_1 is the first element in the current traffic pattern). In such a case, reporting x_N suffices to construct the pattern of the last $N-3$ unreported measurements.

IV. Experimentation

A. *Topology Setup*

The experimental topology is depicted in Fig. 2. It consists of a Science DMZ connected to a WAN via two Juniper MX 204 routers [14] (denoted by BR and ISP in the figure). The science DMZ and the WAN have one DTN for data exchange and one perfSONAR node to monitor and archive network performance. Two optical TAPs are connected to the ingress and egress ports of the BR. The TAPs are provided by Fiber Instrument Sales that can handle speeds up to 100Gbps [15]. They duplicate the packets to the PDP switch. The PDP switch is Edgecore Wedge100BF-32X switch (Intel Tofino) [16]. It passively collects the measurements and reports them to the perfSONAR node. iPerf3 tool is utilized to generate the traffic. ΔT_{min} and ΔT_{max} are set to one and five seconds, respectively.

B. *Evaluating the Impact of* σ

A two-minute test is performed between the DTNs to evaluate how different values of σ affect the number of samples and the RME. The results of the experiment are depicted in Fig. 3. The ground truth throughput is the throughput without sampling. The upper left graph shows that the number of samples is 96, and the RME is around zero when σ is set to 0.025. The upper middle graph shows that the RME slightly increases, and the number of samples reduces by 25% with σ being 0.05 compared to 0.025. The same behavior is observed with σ being 0.075 compared to 0.05, such that the increase in the RME and the reduction in the number of samples are higher compared to the change from σ being 0.05 to 0.025.

As the value of σ increases above 0.075, the RME becomes significant. The model fails to detect the fluctuation in the throughput because σ is relatively large with respect to the average change in the throughput. For instance, when σ is set to 0.15, the number of samples is 31. The average reporting rate is $\frac{31}{120} = 0.26$ samples per second, which means that the average $\Delta T_{current}$ is 3.9 seconds. Thus, most of the throughput measurements predicted by the model were close enough to the observed values (according to the defined σ).

C. *Comparing with other Adaptive Systems*

The proposed adaptive system is compared to LP and the multiadaptive sampling technique (MuST) [12]. MuST uses the

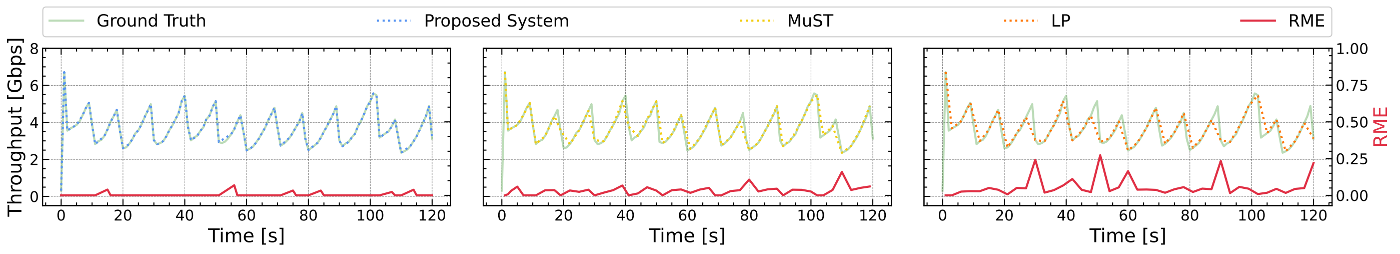

Fig. 4. The throughput and RME comparison between the proposed system, MuST, and LP.

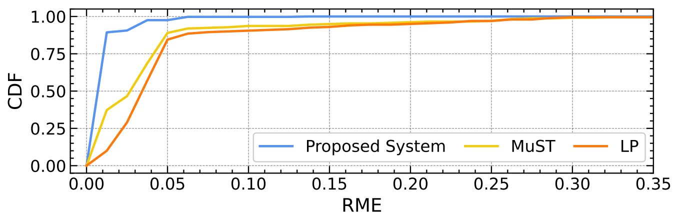

Fig. 5. CDF of the RME for the proposed system, MuST, and LP.

TABLE III
RULES TO DEFINE THE NEXT INTERVAL BETWEEN SAMPLES IN MUST.

Current m	ΔT_{next}	ΔS_{next}
$1-\sigma < m < 1+\sigma$	$\Delta T_{current}$	$\Delta S_{current}$
$m < 1-\sigma$	$m\Delta T_{current}$	$m\Delta S_{current}$
$m > 1+\sigma$	$2 * \Delta T_{current}$	$\Delta S_{current} + 0.15\Delta S_{current}$

last N observations to adjust the time interval between samples and the size of the samples. Table III shows how MuST adjusts ΔT_{next} and ΔS_{next} based on the error m, $\Delta T_{current}$, and $\Delta S_{current}$, where $\Delta S_{current}$ is the current sample size and S_{next} is the size of the next sample.

A two-minute iPerf test is performed between the two DTNs. The parameters σ and N_{max} are set to 0.05 and 5, respectively. Fig. 4 depicts the ground truth throughput, the throughput reported by each of the three systems, and the RME of each system. The proposed system outperforms LP and MuST, where the ground truth throughput and the reported throughput overlap with an RME near 0 throughout the test. MuST performs better than LP, where it missed reporting a few critical measurements, which led to an increase in the RME. LP has the highest overall RME, where it missed a higher number of critical measurements than MuST.

The cumulative distribution function (CDF) of the RME for the three systems is calculated by executing 10 tests and is depicted in Fig. 5. The maximum observed RME by the proposed system is 0.06, where 98% of the RME values are less than 0.04. The maximum observed RME by MuST is 0.28, where 75% of the RME values are less than 0.04, 85% are less than 0.05, 90% are less than 0.1, 95% are less than 0.2, and 98% are less than 0.25. The maximum observed RME by LP is 0.3, where 65% of the RME values are less than 0.04, 80% are less than 0.05, 88% are less than 0.1, 94% are less than 0.2, and 98% are less than 0.25.

V. DISCUSSION: LIMITATIONS AND FUTURE WORK

The proposed system can maintain the state (e.g., number of bytes, RTT, packet losses, etc.) of up to 2048 flows. This imposes no limitation if the system targets science DMZ networks, as they are characterized by a few elephant flows [1]. However, the system is not scalable enough to monitor the traffic on an ISP. The constraint on the number of flows that can be simultaneously monitored originates from the available resources on the PDP switches and the number of metrics maintained by the data plane [6]. Future work aims to investigate the optimal resource allocation strategy at the PDP switch to maximize the number of monitored flows without impacting the set of monitored metrics. One approach is to utilize FermatSketch, a data structure that dynamically allocates the memory resources of the PDP switch [17].

The proposed system involves the control plane in the data collection process. Because the control plane does not have enough processing power to extract per-packet measurements from the data plane [8], it periodically extracts the measurements. Thus, all measurements collected by the data plane and not extracted by the control plane are discarded. To minimize data loss, the proposed system aggregates the measurements on the data plane. The control plane then extracts the aggregated values.

In order to eliminate the loss, the data plane should directly report the measurements to the archiver. However, reporting per-packet measurements is not an option as it floods the archiver. Future work aims at deploying an adaptive sampling model directly on the data plane. The model selectively reports measurements similar to the proposed system. Nevertheless, deploying the adaptive model on the data plane results in a more accurate sampling compared to the model running on the control plane because it has access to the real measurements and not the aggregated measurements.

VI. CONCLUSION

This paper deploys an adaptive sampling model to reduce the consumption of the processing and storage resources at the perfSONAR archiver. A PDP switch is installed passively and monitors the inbound and outbound traffic to a Science DMZ. The data plane of the switch is programmed to collect and maintain per-flow per-packet measurements. The control plane then extracts the measurements and utilizes an LP-based adaptive sampling model to make a per-measurement sampling decision. The model significantly reduces the number of measurements reported to the archiver under stable network conditions and provides an accurate presentation of the state of the flows under unstable network conditions.

VII. Appendix

For any two consecutive observations x_{i-1} and x_i in vector x, their difference d_i is bounded as follows:

$$1-\sigma \le \frac{x_{p_i}}{x_i} \le 1+\sigma$$

$$1-\sigma \le \frac{x_{i-1}+\frac{x_{i-1}-x_1}{i-1}}{x_{i-1}+d_i} \le 1+\sigma$$

$$(1-\sigma)(x_{i-1}+d_i) \le x_{i-1}+\frac{x_{i-1}-x_1}{i-1} \le (1+\sigma)(x_{i-1}+d_i)$$

$$\frac{\frac{x_{i-1}-x_1}{i-1}-\sigma x_{i-1}}{1+\sigma} \le d_i \le \frac{\frac{x_{i-1}-x_1}{i-1}+\sigma x_{i-1}}{1-\sigma} \tag{1}$$

The RME is bound from above as follows:

$$\begin{aligned} x_{N+1} &= \tfrac{(N+1)x_{N+1}}{N+1} \\ &= \tfrac{x_{N+1}}{N+1} + \tfrac{x_N+d_{N+1}}{N+1} + \ldots + \tfrac{x_1+d_{N+1}+\ldots+d_2}{N+1} \\ &= \tfrac{x_{N+1}+x_{N+1}+\ldots+x_1}{N+1} + \tfrac{d_{N+1}}{N+1} + \ldots + \tfrac{d_{N+1}+\ldots+d_2}{N+1} \\ &= \mu + \sum_{i=2}^{N+1} \tfrac{(i-1)d_i}{N+1} \end{aligned}$$

$$x_{N+1}-\mu = \sum_{i=2}^{N+1} \tfrac{(i-1)d_i}{N+1}$$

by (1):

$$x_{N+1}-\mu \le \tfrac{1}{N+1} \sum_{i=2}^{N+1} (i-1) \tfrac{\frac{x_{i-1}-x_1}{i-1}+\sigma x_{i-1}}{1-\sigma}$$

$$x_{N+1}-\mu \ge \tfrac{1}{N+1} \sum_{i=2}^{N+1} (i-1) \tfrac{\frac{x_{i-1}-x_1}{i-1}-\sigma x_{i-1}}{1+\sigma}$$

if $x_{N+1}-\mu \ge 0$:

$$\mid x_{N+1}-\mu \mid \le \frac{1}{N+1} \sum_{i=2}^{N+1} (i-1) \frac{\frac{x_{i-1}-x_1}{i-1}+\sigma x_{i-1}}{1-\sigma} \tag{2.1}$$

if $x_{N+1}-\mu \le 0$:

$$\begin{aligned} \mid x_{N+1}-\mu \mid &\le \frac{-1}{N+1} \sum_{i=2}^{N+1} (i-1) \frac{\frac{x_{i-1}-x_1}{i-1}-\sigma x_{i-1}}{1+\sigma} \\ &\le \frac{1}{N+1} \sum_{i=2}^{N+1} (i-1) \frac{\frac{x_1-x_{i-1}}{i-1}+\sigma x_{i-1}}{1+\sigma} \end{aligned} \tag{2.2}$$

For any element x_i, its minimum value $x_{i_{min}}$ and its maximum value $x_{i_{max}}$ are expressed in term of x_1 as follows:

$$x_{i\_max} = \frac{ix_1}{2(1-\sigma)^{(i-1)}} - x_1(\sum_{j=1}^{i-2} \frac{i}{(i+1-j)(i-j)(1-\sigma)^j}) \tag{3.1}$$

$$x_{i\_min} = \frac{ix_1}{2(1+\sigma)^{(i-1)}} - x_1(\sum_{j=1}^{i-2} \frac{i}{(i+1-j)(i-j)(1+\sigma)^j}) \tag{3.2}$$

Substituting (3.1) and (3.2) in (2.1) and (2.2), respectively, produce the upper bound of the RME in terms of N, σ, and x_1:

$$\mid \tfrac{x_{N+1}-\mu}{\mu} \mid \le max(\tfrac{(N^2+2N)(x_{N\_max}-x_1+\sigma x_{N\_max}(N-1))}{2\mu(N-1)(1-\sigma)}, \tfrac{(N^2+2N)(x_1-x_{N\_min}-\sigma x_{N\_min}(N-1))}{2\mu(N-1)(1+\sigma)}) \tag{4}$$

References

[1] J. Crichigno, E. Bou-Harb, and N. Ghani, "A comprehensive tutorial on science DMZ," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 2041–2078, 2018.

[2] A. Hanemann, J. W. Boote, E. L. Boyd, J. Durand, L. Kudarimoti, R. Łapacz, D. M. Swany, S. Trocha, and J. Zurawski, "perfSONAR: A service oriented architecture for multi-domain network monitoring," in *Service-Oriented Computing-ICSOC 2005: Third International Conference, Amsterdam, The Netherlands, December 12-15, 2005. Proceedings 3*, pp. 241–254, Springer, 2005.

[3] A. Mazloum, J. Gomez, E. Kfoury, and J. Crichigno, "Enhancing perfSONAR Measurement Capabilities using P4 Programmable Data Planes," in *Proceedings of the SC'23 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis*, pp. 819–829, 2023.

[4] E. A. Hernandez, M. C. Chidester, and A. D. George, "Adaptive sampling for network management," *Journal of Network and Systems Management*, vol. 9, pp. 409–434, 2001.

[5] A. Mazloum, "Source codes." https://github.com/AliMazloum/perfSONAR-Adaptive-Sampling. Accessed: 11/28/2023.

[6] E. F. Kfoury, J. Crichigno, and E. Bou-Harb, "An exhaustive survey on P4 programmable data plane switches: Taxonomy, applications, challenges, and future trends," *IEEE access*, vol. 9, pp. 87094–87155, 2021.

[7] A. Mazloum, E. Kfoury, J. Gomez, and J. Crichigno, "A Survey on Rerouting Techniques with P4 Programmable Data Plane Switches," *Computer Networks*, vol. 230, p. 109795, 2023.

[8] A. AlSabeh, J. Khoury, E. Kfoury, J. Crichigno, and E. Bou-Harb, "A survey on security applications of P4 programmable switches and a STRIDE-based vulnerability assessment," *Computer Networks*, vol. 207, p. 108800, 2022.

[9] J. Gomez, E. F. Kfoury, J. Crichigno, and G. Srivastava, "A survey on TCP enhancements using P4-programmable devices," *Computer Networks*, vol. 212, p. 109030, 2022.

[10] J. Giertl, J. Baca, F. Jakab, and R. Andoga, "Adaptive sampling in measuring traffic parameters in a computer network using a fuzzy regulator and a neural network," *Cybernetics and Systems Analysis*, vol. 44, pp. 348–356, 2008.

[11] Y. Lu and C. He, "Resource allocation using adaptive linear prediction in WDM/TDM EPONs," *AEU-International Journal of Electronics and Communications*, vol. 64, no. 2, pp. 173–176, 2010.

[12] J. M. C. Silva, P. Carvalho, and S. R. Lima, "A multiadaptive sampling technique for cost-effective network measurements," *Computer Networks*, vol. 57, no. 17, pp. 3357–3369, 2013.

[13] V. Jacobson, "Congestion avoidance and control," *ACM SIGCOMM computer communication review*, vol. 18, no. 4, pp. 314–329, 1988.

[14] N. Screening, "Juniper Networks MX204," 2022. Accessed: 12/3/2023.

[15] F. I. Sale, "Optical TAP," 2022. Accessed: 11/28/2023.

[16] Intel, "Wedge 100BF-32X, 100GbE data center switch, Barefoot Networks, an Intel® company," 2023. Accessed: 11/28/2023.

[17] K. Yang, Y. Wu, R. Miao, T. Yang, Z. Liu, Z. Xu, R. Qiu, Y. Zhao, H. Lv, Z. Ji, *et al.*, "ChameleMon: Shifting Measurement Attention as Network State Changes," *arXiv preprint arXiv:2301.00615*, 2023.