Predicting Elephant Flows in Internet Exchange Point Programmable Networks

Marcus Vinicius Brito da Silva, Arthur Selle Jacobs,
Ricardo José Pfitscher, and Lisandro Zambenedetti Granville

Institute of Informatics – Federal University of Rio Grande do Sul
Av. Bento Gonçalves, 9500 – Porto Alegre, Brazil
{mvbsilva, asjacobs, rjpfitscher, granville}@inf.ufrgs.br

Abstract. Internet Exchange Points (IXPs) are high-performance networks that allow multiple autonomous systems to exchange traffic, with benefits ranging from cost reductions to performance improvements. As in any network, IXP operators face daily management challenges to promote better usage of the services provided by the network. An essential problem in IXP management concerns the identification of elephant flows, which are characterized by having traffic size and duration significantly higher than other flows. The current approaches to the identification of elephant flow in IXP networks depend that the analyzed flows exceed predefined thresholds to classify them as elephants. However, although it is not perceptible initially, elephant flows are elephant ones since their first packet. Hence, in this paper, we present a mechanism to predict flows behavior using historical observations and, by recognizing temporal patterns, identify elephant flows even before they exceed such thresholds. Our approach consists in predicting new flows size and duration through a Locally Weighted Regression (LWR) model, using the previous flows behavior and its temporal correlation with the new flow. The experimental results show that our mechanism is able to predict the volume and duration of new flows, and react to elephant flows rapidly, approximately 50.3 ms with up to 32 historical samples in the prediction model. These numbers are much smaller than the time each flow would take to exceed the thresholds to classify it as an elephant. In addition, the mechanism accurately predicts up to 80% of elephant flows in our evaluation scenarios and approximately 5% of false positives.

Keywords: Software-Defined Networking, Network Management, P4 Language

1 Introduction

Internet Exchange Points (IXPs) are high-performance networks and perform an essential role in the Internet ecosystem [1], accounting for at least 20% of all traffic exchanged between autonomous systems (ASes) [2]. As in any network, IXP operators face daily management challenges to promote a better usage of the services provided by the network. An important problem in IXP management concerns the identification of elephant flows, who are characterized by having traffic size and duration significantly higher than other flows [3]. As a consequence, elephant flows can significantly impact

on the traffic of the smaller flows that share the same path on the IXP infrastructure, thus compromising the overall perceived network Quality of Service (QoS) [4].

Because of the impacts that elephant flows have on network performance, network operators must quickly detect them to promptly perform mitigation actions [5]. As elephant flow classification depends on the flows' size and duration, it takes an amount of time t for these values (size and duration) to reach a pre-established threshold; that delays the reaction in at least t units of time. However, elephant flows are obviously elephant ones since their first packet, although detection occurs later (t). Today, there is a lack of solutions that take this situation into account, thus delaying decisions/actions, and consequently creating problems for smaller flows.

Proposed solutions such as DevoFlow [6], OpenSample [7], and SDEFIX [8] present mechanisms for identifying elephant flows in IXP networks using sFlow [9] and managing paths (as reaction to elephant flows) using SDN/OpenFlow [10]. As a first attempt to more quickly identify elephant flows in programmable IXP networks, we had proposed IDEAFIX [11], where flow duration and size were analyzed for each ingress packet immediately at the edge P4 [12] switch. Although IDEAFIX reduces the detection delay when compared to controller-based approaches, it still requires flows size and duration to reach the thresholds to be identified as elephant flow.

In this paper, to shorten the time interval until classification thresholds are surpassed, we present a mechanism to identify elephant flows in IXP programmable networks using historical observations. Our approach consists of predicting flow size and duration through a Locally Weighted Regression (LWR) model [13] [14]. The sample weights, in the LWR model, are attributed from a Gaussian distribution adjusted by the network operator according to the desired sample time window. In addition, the network operator can define a tolerance range to validate estimations according to the calculated prediction interval for each of them. If the prediction interval is less than the tolerance, then the inferred values are considered valid and immediately confronted with thresholds to characterize the flow as an elephant one or not, right in the beginning. When the prediction interval is larger than the tolerance, the inference is invalidated and the analysis proceeds to the second approach, based on the IDEAFIX identification process.

Experimental results show that our mechanism is able to predict the volume and duration of new flows and identify elephant flows rapidly, approximately 50.3ms with up to 32 samples (*i.e.*, previous flows behavior). Even when sample numbers are significantly large, *i.e.*, 2048 or 4096, our mechanism can predict and react to elephant flows at an interval of up to 126.3ms and 174.6ms, respectively. These numbers are much smaller than the time at which the flow would take to exceed the thresholds to classify it as an elephant one. In addition, the packet processing time of software-emulated P4 switches also influences the reaction time. On a hardware switch, packet processing time would be in nanoseconds. Finally, the mechanism's accuracy in well-behaved scenarios obtained at least 80% of prediction, even with conservative tolerance, and approximately 5% of false positives.

The remainder of this paper is organized as follows. In Section 2, related work is discussed. In Section 3, we describe the concept of elephant flows and the management challenges they require in an IXP network, as well as the statistical method used in our proposal. In Section 4, we describe our novel mechanism to predict and identify

elephant flows in IXP networks. In Section 5, we present the evaluation of our proposal, as well as the achieved results. Finally, in Section 6 we present the main conclusions of the study and future work perspectives.

2 Related Work

In DevoFlow [6], the OpenFlow protocol is used to keep track of elephant flows with different monitoring mechanisms, such as packet sampling and port triggering. DevoFlow changes the OpenFlow switches by adding new features and counters to the hardware in order to facilitate the identification of flows in the network. In OpenSample [7], sFlow is used to perform flow sampling on a solution to manage SDN networks. Then, flow rates are calculated by subtracting TCP sequence numbers from two samples of the same flow and dividing the value by the elapsed time between them. However, the statistics of the switches are not considered and the flows need to be sampled and sent to the control plane to be processed. This delays elephants flow identification and add a significant amount of sampling data in the network. Therefore, only when a threshold is exceeded, in terms of byte count, the flow is classified as an elephant one and it is rerouted to the least congested path between the endpoints.

In SDEFIX [8], an identification module is used to classify flows, analyzing the collected data by sFlow according to predefined rules. When size and duration thresholds are exceeded, the flow is classified as elephant and it is mitigated according to policies written by the network operator. For example, elephant flows can be routed through alternative paths in the IXP infrastructure, so as not to affect the small flows that are sharing the same paths in the network. As in the previous approaches, SDEFIX performs elephant flow analysis and identification integrally in the control plane and a reaction occurs only when thresholds are exceeded.

In our previous work [11], we proposed IDEAFIX, aiming to identify elephant flows in IXPs faster by relying on programmable data planes. To do that, IDEAFIX takes advantage of P4 features to store information about the size and the duration of network flows. This information is extracted from each packet that enters the network and compared to predefined thresholds for flow classification. If flow size and duration do not exceed such thresholds, then it is marked as not yet elephant, and the packet is routed to the default path. If thresholds are exceeded, however, then the flow is classified as an elephant one and a notification is sent to the control plane to report the detected elephant flow. Then, the network operator can define actions to mitigate the effects of that flow, in accordance with, for example, QoS policies.

Although the previous approaches allow the identification of elephant flows in IXP networks, there is a dependency on that the analyzed flows exceed the thresholds to be classified as an elephant flow. However, although it is not perceptible initially, elephant flows are elephant ones since their first packet. In this paper, to dispense with waiting dependence for the threshold to be exceeded, we present a mechanism to predict flows behavior using historical observations and, from the temporal patterns recognition, identify elephant flows even before they exceed thresholds.

3 Elephant Flows in IXP Networks and LWR

In the Internet, most of the flows have a small size and/or lifetime (*i.e.*, mice flows or small flows) [15] [16], althought there is a smaller number of flows that accounts for the majority of the traffic, also having a longer lifetime; these are the elephant flows [3] [17]. The presence of elephant flows in networks is common, and because they are flows with a long lifetime and size they may cause performance issues that demand proper management actions from network operators. For example, elephant flows can impact on smaller flows that occasionally share the same network data path. Also, elephant flows can consume memory resources of network devices, and this also can lead to undesirable delays, queuing, and packet losses [18] [3] [8]. In the case of IXP networks, the problem turns to be even more critical, because of the amount of traffic that IXP networks must deal with. Thus, the faster the elephant flows are identified, the faster their effects can be mitigated [8].

Figure 1 presents the traffic behavior in Australia IXPs [19] over the course of a week. IX-Australia offers peering in Western Australia (WA-IX), New South Wales (NSW-IX), and Victoria (VIC-IX). Although this is an aggregated traffic behavior, it is possible to recognize a periodic pattern, with discrete variations. This IXP's behavior is also seen in the Amsterdam Internet Exchange (AMS-IX), as well as on more than 30 IXPs networks in Brazil [20]. Other IXPs experience the same traffic behavior as well. Even though it is not possible to explicitly spot the elephant flows inside Figure 1, they are there. In some cases, it is observed that elephant flows can contribute with more than 59% of total traffic volume [16]. In addition, elephant flows traffic has a substantial, but not perfect, correlation with traffic in total flows [21].

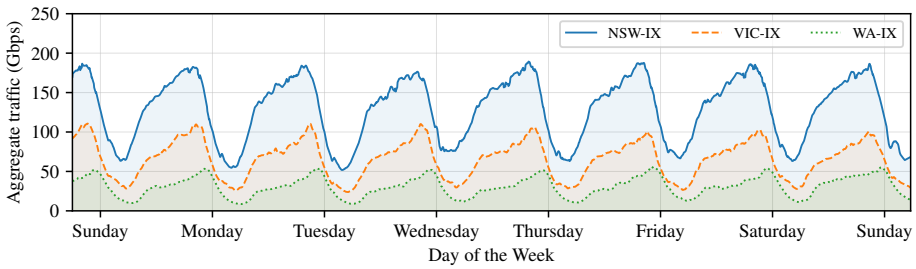


Fig. 1. Australia IXPs aggregate traffic.

Considering that there is a periodicity in the traffic of an IXP network (as shown in Figure 1), it is possible to predict the size and duration of new flow instances by observing the previous flows' temporal behavior. In other words, as the traffic pattern exhibits a periodicity, the events that previously resulted in elephant flows will later probably happen again. When taking the history of flows to predict volume and duration of new ones, we also consider that, in such a history, younger instances of flows must weight more than older instances in the prediction computation. By doing so, we can monitor flows behavior in the IXP network at run-time and predict the behavior of new flows according to their periodicity.

3.1 Locally Weighted Regression prediction method

To predict network flows behavior, we leverage a statistical method, Locally Weighted Regression (LWR) [13] [22]. LWR allows to predict the value of dependent variables from a set of independent variables, through localized weighting. LWR is based on the assumption that the neighboring values of the desired point, in a sample range, are the best indicators for the prediction [13]. In our context, the size and duration of previous flow instances (samples in the model) are dependent variables, its start timeStamp are the independent variables, and the predictions are the new flow size and duration in its start timeStamp (desired point). The neighbors are the flow instances with timeStamp closest of the new flow's timeStamp.

This method fits a surface to "local" points using distance-weighted regression [14]. LWR is derived from standard linear regression [23], as shown by Equation 1. LRW proposal consists of defining the linear model β parameters, minimizing the squared errors for each sample $(y_i - f(x_i))$, weighted locally by w_i weight.

$$F(\beta_0, \beta_1, ..., \beta_m) = \sum_{i=1}^{n} w_i (y_i - f(x_i))^2 \tag{1}$$

After the derivation [13] [22] of Equation 1, the linear model β parameters, used to predict values, are obtained by normal equations:

$$\beta = \left(X^T W^T W X\right)^{-1} X^T W^T W y \tag{2}$$

where X is an $n \times (m+1)$ matrix consisting of n data point, each represented by its m input dimensions and a "1" in the last column, y is a vector of corresponding outputs for each data point, W is a diagonal matrix of order n with the w_i weights of each data point, and β is the $m+1$ vector of unknown regression parameters. Thus, prediction of the outcome of a query point x_q becomes:

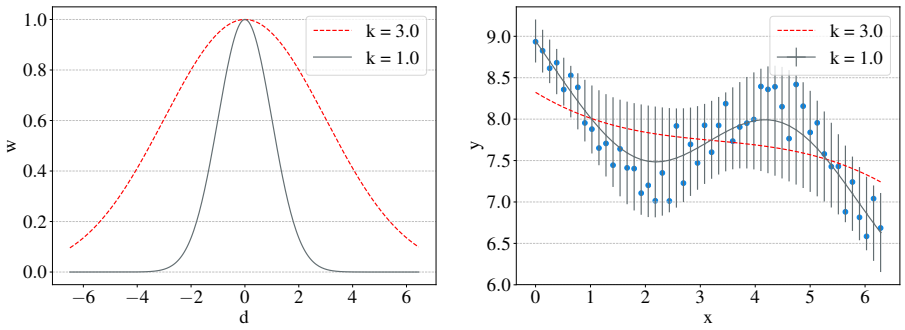
$$y_q = x_q^T \beta \tag{3}$$

The weights allow establishing the influence of the samples according to their distance (*i.e.*, age) to the desired point. Many weighting functions are proposed in literature [13] [24]. The widely used weighting function is the Gaussian kernel weighting function [25], which can be defined as follows:

$$w_i = exp\left(\frac{-d_i^2}{2k^2}\right) \tag{4}$$

From Equation 4, each sample (x_i, y_i) in the model will receive a weight w_i in the interval $[0, 1]$ (as shown in Figure 2(a)), according to its distance $d_i = \sqrt{(x_q - x_i)^2}$ to the desired point x_q. That is, how much less the distance of the (x_i, y_i) point to the (x_q, y_q) desired point (*i.e.*, $d_i \approx 0$), the greater the influence of (x_i, y_i) on the model because weight will be closer to 100% (*i.e.*, $w_i \approx 1$). The parameter k scales the size of the kernel to determine how local the regression will be. Figure 2(a) shows the Gaussian kernel magnitude with k = 1.0 and k = 3.0. That is, the greater the value of k, the larger the significant samples range in the model. However, if k is small, then the model will be influenced only by very close samples.

Figure 2(b) shows the model behavior influenced by the adjustment of the magnitude k of the Gaussian kernel. The samples were obtained by: $f(x) = -(\sin(x) + (0.4x) + 0.26\eta) + 9$; where η is a samples random noise, and $0 \leqslant x \leqslant 2\pi$. Due to small sample variation $(6.5 \leqslant y \leqslant 9.0)$ in the model, when $k = 1.0$, LWR result is well behaved and smoothly. When $k = 3.0$, the result is similar to traditional linear regression, which does not allow to follow the sample changes. The fine adjustment of parameter k is essential to use LWR properly. We propose k to be defined by the sample standard deviation of the x_i independent variables. This allows to dynamically adapt the Gaussian kernel according to changes in the behavior of the network.



(a) Gaussian kernel magnitude. (b) LWR influenced by k parameter.

Fig. 2. Gaussian kernel and LWR prediction interval.

We use prediction intervals [22] to assess the quality of fits in predictions. Prediction intervals I_q are expected bounds of the prediction error at a query point x_q, which can be defined as follows:

$$I_q = x_q^T \beta \pm t_{\alpha/2,n'-p'} s \sqrt{1 + x_q^T (X^T W^T W X)^{-1} x_q} \tag{5}$$

where $t_{\alpha/2,n'-p'}$ is Student's t-value of $n' - p'$ (Equation 6) degrees of freedom for a $100(1-\alpha)\%$ prediction bound.

$$n' = \sum_{i=1}^{n} w_i^2 \quad \text{and} \quad p' = \frac{n'}{n} m \tag{6}$$

Figure 2(b) shows the prediction interval behavior for (x_q, y_q) inferences with $k = 1.0$. The prediction interval increases when there is a large sample variation $(e.g., x \approx 2)$. When the model is well behaved $(e.g., x \approx 0)$, the forecast interval is smaller. This allows obtaining the accuracy of the inference method even with large sample variations.

4 Proposed Approach

In this section, we describe the proposed mechanism to predict and identify elephant flows using historical observations and temporal correlation. Figure 3 shows the architecture of our proposal, composed of an IXP network infrastructure abstraction with a

programmable data plane, a historical database, and a prediction module, in the control plane, attached to the network controller. When an edge switch receives the first packet of a flow, the control plane receives a notification to compute the path by which the flow will be routed in the IXP network. Then, running at the control plane, the mechanism uses the LWR model to predict the flow size and duration. When predicted values are considered valid, from the prediction interval tolerated, the mechanism characterizes the flow as an elephant or not, according to thresholds. Such a process of flows' parameters prediction and related characterization is discussed following, in four phases: sample selection, prediction, validation of predictions and flow classification.

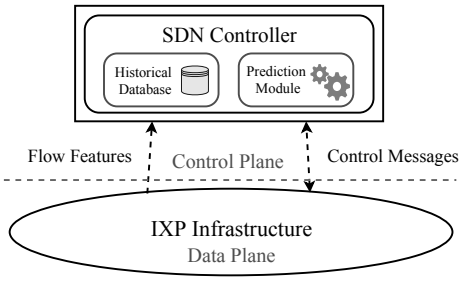


Fig. 3. Architecture.

The prediction mechanism relies on a historical database to store information about the behavior of previous flow instances. This information is extracted directly from the data plane, where an agent uses P4 to account, in each edge switch, the volume and duration of each flow that ingresses in the IXP network. To identify each flow individually, a 5-tuple (*i.e.*, source and destination IP addresses, source and destination ports, and transport protocol type) is mapped by hash functions to generate index keys. When a flow is finalized (*e.g.*, TCP FIN Flag is valid or timeout exceeded), a notification (on top of UDP) is sent to the control plane to report the flow size and duration along with its 5-tuple and start flow time (*i.e.*, *timeStamp*).

The sample selection is based on the tuple that defines the new flow. Initially, all records with the same source and destination IP address, transport protocol, and at least one of the application ports are selected. Samples are selected within a time window defined by the network operator to delimit the size of the historical database. For example, it is possible to set a sampling interval in days, weeks, months, and so on. In addition, selection can occur from hierarchical aggregation of IP addresses, based on HHH [26]. This enables more samples in the inference process, *e.g.*, to sub-nets behavior analysis.

The prediction step estimates the flow parameters according to historical information. To do that, we use the previously described LWR model (see Section 3.1). We use flow sizes and duration as dependent variables y_q in Equation 3, allowing the mechanism to predict these values individually. In both cases, the independent variables x_i are the previous flow *timeStamp* (samples), and x_q is current *timeStamp* (new flow). To establish temporal correlation and periodicity, we defined the sample temporal distance d_i concerning to the new flow time of the day (*i.e.*, within 24 hours), in according to Equation 7. That is, the largest temporal distance of a sample to the current flow will

be $24/2$ hours. Thus, it is possible to correlate periodic occurrences of elephant flows daily. For exemplification purposes, the *timeStamp* is used in hours.

$$d_i = \left|x_q - x_i\right| mod\ 24 = \begin{cases} d_i & \textbf{if } d_i \leq 24/2 \\ 24 - d_i & \textbf{if } d_i > 24/2 \end{cases} \tag{7}$$

The samples weighting in the LWR model is defined according to Equation 4. We make a modification in the Gaussian function, as shown in Equation 8. From this, the network operator can establish a minimum weight for a given range of samples. That is, distances in the range $[0,k]$ may receive a minimum weight equal to l (*e.g.*, 30% or 40%), when d_i (Equation 7) is equal to k. This allows operators to determine how conservative the inference process will be. For example, in scenarios where the network is saturated, the network operator can be more rigorous in the samples weighing to define which temporal distance will have the most significant influence on the predictions.

$$w_i = exp\left(\frac{d_i^2}{k^2} \cdot \ln(l)\right) \tag{8}$$

To adjust the magnitude of the Gaussian kernel, we propose k to be defined by the sample standard deviation of the x_i independent variables. This allows to dynamically adapt the Gaussian kernel according to changes in the behavior of the network. Finally, when sample weights are defined, it is possible to determine the β parameters of the linear model (Equation 2) and then predict the new flow volume and duration (Equation 3) according to its start *timeStamp* (*i.e.*, x_q).

After predicting flows parameters, the mechanism performs the validation of LWR inferred values. For this validation, the network operator must define a tolerance for the calculated prediction interval over each inference. The prediction interval determines the fluctuation margin for the inferred value, as shown in Figure 2(b). Thus, the network operator can determine the tolerance according to, for example, the demand of the network. That is, when the network is underutilized, the operator can set a softer tolerance; otherwise, in saturation cases, the tolerance can be more conservative.

Lastly, when the inferences are not validated (*i.e.*, $pred\_interv > tol$), it will not be possible to predict the flow behavior, and consequently, it will not be possible to classify it as an elephant one. In this case, analysis and possible identification occurs based on our previous IDEAFIX identification reference. In IDEAFIX, the packets of each flow are analyzed directly in the programmable data plane, and elephant flow identification happens whenever the volume and duration of a flow exceed the predefined thresholds.

5 Performance Evaluation

To assess the feasibility of our proposal, we focus on evaluating three main aspects: (*i*) reaction time, *i.e.*, the interval between the ingress time of the flow first packet in the IXP network, the LWR predicting time, and the identified elephant flow management time; (*ii*) excess data, *i.e.*, the number of bytes that transited in default path until the flow has been identified as an elephant one and a reaction occurs; and (*iii*) mechanism accuracy, *i.e.*, the percentage of valid elephant flows predictions in function of the prediction interval tolerance. For all metrics, lower values are better.

Figure 4 depicts the topology used in the evaluation experiments. We used a topology based on the AMS-IX [27] infrastructure, as also used in the related work [11] [8], with 8 ASes connected by edge switches to a programmable data plane IXP. Each edge switch is directly linked to a programmable switch, which, in turn, has at least two connection paths to the core of the IXP network. Switches were implemented in the language $P4_{16}$ and by using the software switch BMv2. The infrastructure was emulated using *Mininet* 2.3, with a bandwidth of 1Gbps per link and no propagation delay.

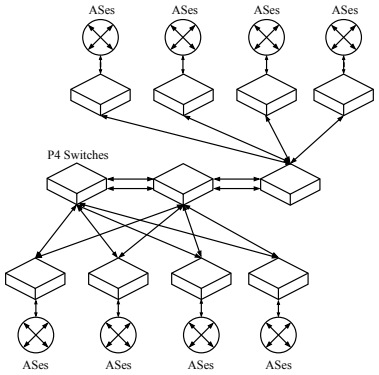
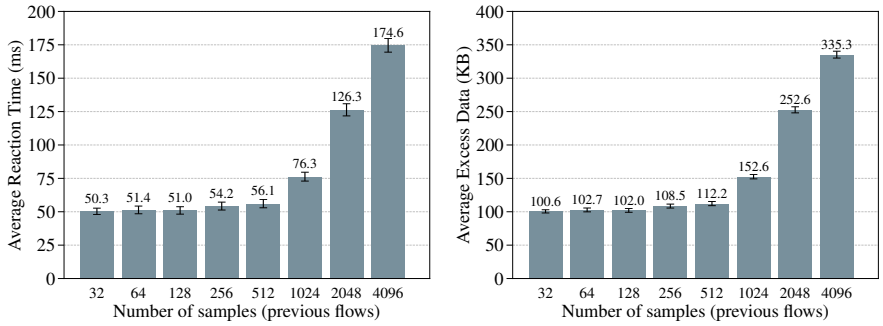


Fig. 4. IXP network topology [8].

We generated a workload with distinct sizes of TCP flows between the connected ASes, using *iPerf*. The flow bandwidth was established at 10 Mbps, and the duration was determined through a normal distribution, with a mean of 150 seconds, and a standard deviation of 20 seconds, for elephant flows. For small flows, we used a mean of 10 seconds and a standard deviation of 4 seconds [8] [11]. The IXP network traffic was distributed periodically over nine weeks (*i.e.*, $\approx$ 1,600 hours). The thresholds were defined in 10 MB and 10 seconds [8] [4]. Each experiment lasted 10 minutes, repeated 32 times, and 2,048 flows were generated, of which 12% were elephant flows [11]. We used a computer with Intel Core i7-4790 processor, 16GB of RAM, and *Ubuntu 16 LTS*.

Figure 5(a) shows the results of the reaction time to elephant flows. The reaction time was analyzed exclusively as a function of the number of samples in the prediction model. This allows us to observe the time it takes for a reaction to occur with the prediction model being generated at run-time. In the y-axis, the average reaction time (in milliseconds) has no significant difference at a 95% confidence level for a set of up to 512 samples. The most significant difference is evident when there are sets with at least 1,024, 2,048, and 4,096 samples. In these cases, the average reaction time varies between 76.3ms, 126.3ms, and 174.6ms, respectively.

These numbers are much smaller than the time at which the flow would take to exceed the thresholds to classify it as an elephant. That is, for the thresholds we used, based on the related work [8] [11], an elephant flow would take at least 10s to be identified, at best case. Thus, it is possible to notice that predicting the flows behavior early on allows one to identify the elephant flows as soon as possible. Moreover, the packet processing time of software-emulated P4 switch also influences the reaction time in our approach. On a hardware switch, packet processing time would be in nanoseconds.



(a) Reaction time. (b) Excess data.

Fig. 5. Reaction time and excess data analysis.

Figure 5(b) shows the elephant flows data volume that traveled along the standard path until a reaction occurred. In the y-axis, the average excess data (in Kilobytes) has no significant difference at a 95% confidence level for up to 512 samples. As in the previous analysis, the most significant difference one is evident when there are sets with at least 1,024, 2,048, and 4,096 previous flow samples in the prediction model. In these cases, we can consider that the elephant flows still transmit about 152.6KB, 252.6KB, and 335.3KB on the standard path in the network, respectively.

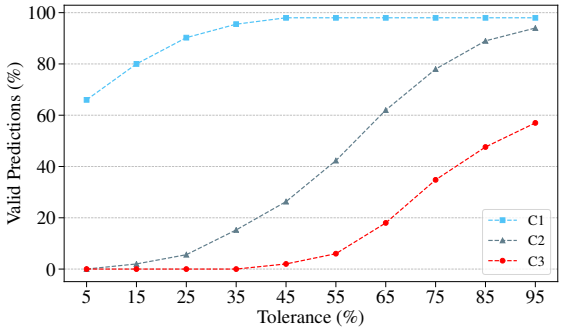


Fig. 6. Predict mechanism accuracy.

The mechanism accuracy (in Figure 6) has evaluated from the percentage of valid predictions, *i.e.*, correctly predicted and identified elephant flows. The first scenario (C1 curve) with low traffic behavior variation and regular periodicity, *i.e.*, flows follow a well-defined frequency. Scenario two (C2 curve) with more significant variation in the flows behavior and periodicity. In scenario three (C3 curve), there was a high variation in the flows behavior and low periodicity, *i.e.*, flows do not have a well-defined frequency pattern. Thus, we evaluate the percentage of valid predictions according to the tolerance defined by the network operator for the prediction interval. Prediction interval was calculated for each predicted value at a confidence level of 90%.

Figure 6 shows that in a well-behaved scenario (C1 curve), 80% of the predictions are validated even with a conservative tolerance (*e.g.*, 15%). In the worst case (C3 curve), when the traffic behavior and periodicity are not regular, approximately 20% of the predictions could be validated, with a tolerance of at least 65%. In the mean case (C2 curve), more than 60% of the predictions were validated at a tolerance of 65%. These results show that the method can predict and validate values even in non-regular scenarios. These results represent, at least, 90% of success in elephant flows identification, out of the total number of elephant flows inserted in the network at each test (*i.e.*, $\approx$ 245 flows), with approximately 5% of false positives. Lastly, the elephant flows whose predictions were not valid, was identified directly in the data plane, based on our previous IDEAFIX [11] approach, immediately after exceeding the thresholds.

6 Conclusions

In this paper, we presented a mechanism to predict and react to elephant flows in programmable IXP networks before thresholds are exceeded. Our approach consists of predicting flow size and duration through a Locally Weighted Regression (LWR) model, following the temporal correlation between the behavior of previous flows. Our experimental results show that our mechanism is able to predict the flow features, and react to elephant flows rapidly, *i.e.*, approximately 50.3ms with up to 32 samples in the model. These numbers are much smaller than the time at which the flows would take to exceed classification thresholds. In addition, the mechanism's accuracy was 80% of validated predictions even with a conservative tolerance of 15%, and approximately 5% of false positives. As future work, we will consider other methods, based on machine learning, to predict flows behavior and mechanisms to define the thresholds dynamically.

Acknowledgement

We thank CNPq for the financial support. This research has been supported by call Universal 01/2016 (CNPq), project NFV Mentor process 423275/2016-0.

References

1. B. Augustin, B. Krishnamurthy, and W. Willinger, "IXPs: Mapped?" in *ACM SIGCOMM Conference on Internet Measurement*, ser. IMC '09. NY, USA: ACM, 2009, pp. 336–349.
2. J. C. Cardona Restrepo and R. Stanojevic, "IXP traffic: a macroscopic view," in *Proceedings of the 7th Latin American Networking Conference*. ACM, 2012, pp. 1–8.
3. L. Guo and I. Matta, "The war between mice and elephants," in *Network Protocols, 2001. Ninth International Conference on*. IEEE, 2001, pp. 180–188.
4. L. A. D. Knob, R. P. Esteves, L. Z. Granville, and L. M. R. Tarouco, "Mitigating elephant flows in SDN-based IXP networks," in *Computers and Communications (ISCC), 2017 IEEE Symposium on*. IEEE, 2017, pp. 1352–1359.
5. E. Gregori, A. Improta, L. Lenzini, and C. Orsini, "The impact of IXPs on the AS-level topology structure of the Internet," in *Computer Communications*. Elsevier, 2011, pp. 68–82.
6. A. R. Curtis, J. C. Mogul, J. Tourrilhes, P. Yalagandula, P. Sharma, and S. Banerjee, "DevoFlow: Scaling flow management for high-performance networks," in *ACM SIGCOMM Conference on Internet Measurement*, vol. 41. NY, USA: ACM, 2011, pp. 254–265.

7. J. Suh, T. T. Kwon, C. Dixon, W. Felter, and J. Carter, "Opensample: A low-latency, sampling-based measurement platform for commodity sdn," in *34th IEEE International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2014, pp. 228–237.
8. L. A. D. Knob, R. P. Esteves, L. Z. Granville, and L. M. R. Tarouco, "SDEFIX—Identifying elephant flows in SDN-based IXP networks," in *IEEE/IFIP Network Operations and Management Symposium (NOMS)*. IEEE, 2016, pp. 19–26.
9. sFlow, "sFlow.org," http://www.sflow.org, 2018.
10. N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: enabling innovation in campus networks," in *ACM SIGCOMM Conference on Internet Measurement*. NY, USA: ACM, 2008, pp. 69–74.
11. M. V. B. da Silva, A. S. Jacobs, R. J. Pfitscher, and L. Z. Granville, "IDEAFIX: Identifying Elephant Flows in P4-Based IXP Networks," in *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM)*. IEEE, 2018.
12. P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese *et al.*, "P4: Programming protocol-independent packet processors," in *ACM SIGCOMM Conference on Internet Measurement*. ACM, 2014, pp. 87–95.
13. W. S. Cleveland and S. J. Devlin, "Locally weighted regression: an approach to regression analysis by local fitting," in *Journal of the American statistical association*, vol. 83, no. 403. Taylor & Francis, 1988, pp. 596–610.
14. E. E. Elattar, J. Goulermas, and Q. H. Wu, "Electric load forecasting based on locally weighted support vector regression," in *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 4. IEEE, 2010, pp. 438–447.
15. Y. Zhang, L. Breslau, V. Paxson, and S. Shenker, "On the Characteristics and Origins of Internet Flow Rates," in *ACM SIGCOMM Conference on Internet Measurement*, vol. 32, no. 4. New York, NY, USA: ACM, Aug. 2002, pp. 309–322.
16. T. Mori, R. Kawahara, S. Naito, and S. Goto, "On the characteristics of Internet Traffic variability: Spikes and Elephants," in *IEICE TRANSACTIONS on Information and Systems*, vol. 87, no. 12, 2004, pp. 2644–2653.
17. W. Fang and L. Peterson, "Inter-AS traffic patterns and their implications," in *IEEE Global Telecommunications Conference (GLOBECOM)*, vol. 3. IEEE, 1999, pp. 1859–1868.
18. T. Mori, M. Uchida, R. Kawahara, J. Pan, and S. Goto, "Identifying elephant flows through periodically sampled packets," in *ACM SIGCOMM Conference on Internet Measurement*, ser. IMC '04. ACM, 2004, pp. 115–120.
19. IX Australia, "Australia Internet Exchange Point," https://www.ix.asn.au/, 2018.
20. Internet Steering Committee in Brazil, "Brazil Internet Exchange Points," http://ix.br/trafego/agregado/rs, 2018.
21. Y. Li, H. Liu, W. Yang, D. Hu, X. Wang, and W. Xu, "Predicting inter-data-center network traffic using elephant flow and sublink information," in *IEEE Transactions on Network and Service Management*, vol. 13, no. 4. IEEE, 2016, pp. 782–792.
22. S. Schaal and C. G. Atkeson, "Robot juggling: implementation of memory-based learning," *IEEE Control Systems*, vol. 14, no. 1, pp. 57–71, 1994.
23. R. Jain, *The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling*. John Wiley & Sons, 1990.
24. J. S. Simonoff, *Smoothing methods in statistics*. Springer Science & Business Media, 2012.
25. M. P. Wand and W. R. Schucany, "Gaussian-based kernels," in *Canadian Journal of Statistics*, vol. 18, no. 3. Wiley Online Library, 1990, pp. 197–204.
26. R. Basat, G. Einziger, R. Friedman, M. Luizelli, and E. Waisbard, "Constant Time Updates in Hierarchical Heavy Hitter," in *ACM SIGCOMM Conference on Internet Measurement*, ser. SIGCOMM '17. NY, USA: ACM, 2017, pp. 127–140.
27. AMS-IX, "Amsterdam Internet Exchange Infrastructure," https://ams-ix.net/technical/ams-ix-infrastructure, 2018.