Stochastic Delay Forecasts for Edge Traffic Engineering via Bayesian Networks

Mary Hogan
Computer Science Department
Saint Louis University, USA
mhogan26@slu.edu

Flavio Esposito
Computer Science Department
Saint Louis University, USA
flavio.esposito@slu.edu

***Abstract*—Traffic engineering at network edges is challenging given the latency-sensitive nature of all applications that need to be supported. End-to-end delay estimation and forecasts were essential traffic engineering tools even before the mobile edge computing paradigm pushed the cloud closer to the end user. In this paper, we model the path selection problem for edge traffic engineering using a risk minimization technique inspired by portfolio theory in economics, and we use machine learning to estimate path selection risks.**

In particular, using real latency time series measurements, both existing and collected with and without the GENI testbed, we compare four short-horizon latency estimation techniques, commonly used by the finance community to estimate prices of volatile financial instruments. Our results suggest that a Bayesian Network approach may lead to good latency (peak) estimation performance, as long as there are dependencies among the time series path latency measurements.

I. Introduction

Edge computing is a fairly novel computing paradigm in which much of the node processing and traffic steering takes place in a process or among paths at the edge of the network, as opposed to in the core of the network as in the Cloud Computing paradigm. This approach has been shown to improve user experience by reducing the perceived latency, and is growing in popularity because of the Internet of Things (IoT) and the vast amount of data that sensors generate. It is inefficient to transmit all the data that a bundle of sensors creates to the cloud for processing and analysis; doing so requires a great deal of bandwidth and all the back-and-forth communication between the sensors and the cloud can negatively impact performance. Traffic engineering at the edge is critical not only to support future Internet of (Medical) Things applications but perhaps more importantly, to manage data marshaling across Points of Presence, $i.e.$, servers located at the edge of the network of application provider, that nowadays generate the vast majority of Internet traffic [8].

When multiple processes compete for resources in a network, data transmission becomes inefficient, and the network can become unreliable. Given the limited resources, application providers may be unable to guarantee a level of service or a level of experience in the network. This problem is exacerbated in disaster scenarios, where connectivity is scarce or unavailable, or when latency-sensitive applications such as image pre-processing from a fleet of drones looking for survivals need to produce feedback in a timely fashion.

978-1-5386-1465-5/17/$31.00

Existing forecast-based path management solutions for mobile and delay-sensitive applications are often tailored to specific protocols or applications [3], [6], they focus on link bandwidth [3] or switch queue size estimation [6], they are not designed to dynamically steer traffic in a multi-path network [3], [4], or they focus on maximizing performance of a single flow [3], [10].

Our Contributions: In this paper, we present a model and test a path management solution that, leveraging results from portfolio theory and stochastic learning theory, helps edge traffic engineers identify end-to-end paths (routes) whose future estimated latency is minimized in a given (short) horizon. In particular, leveraging a portfolio theory formulation [7], we first introduce an analytic model for the path selection problem in edge traffic engineering, capturing the risk-return factor associated with a network path choice and its latency.[1] In our model, the problem of selecting a set of low-latency paths is equivalent to the problem of selecting a portfolio of assets, maximizing the expected return, subject to a given level of (volatility) risk. Since our model captures the risk of a path with its predicted latency value, we need a valuable short-horizon latency estimation technique. To this end, we first measured end-to-end latencies using the ICMP protocol over the GENI testbed [1], and compared the performance of (four) latency estimation techniques commonly used to predict future prices of a volatile financial instrument: a Bayesian network model [5], an autoregressive model, a moving average model, and an AutoRegressive Moving Average (ARMA) model. We then confirm our findings over traces from a real Internet measurement dataset, available at [9].

Our results show that the Bayesian network approach is the best latency (peak) predictor $i.e.$, it has the lowest relative error, as long as there are statistical dependencies among the latency measurements, and such measurements do not have latency variance too small. By dissecting the Bayesian network construction process during our learning phase, we also observe and quantify how, the dependency among subsequent latency time series samples (which vary from path to path) is highly correlated with the latency prediction error. The Bayesian network is in fact a probabilistic graphical model that represents random variables and their conditional dependencies.

The rest of the paper is organized as follows: In § II we

[1] Aside from our initial work [2], to our knowledge, this is the first attempt to apply portfolio theory to a networking problem.

present some motivating applications. In § II we introduce our model inspired by portfolio theory; in § III we describe how we applied the Bayesian network model to predict latencies and in § IV we present our initial evaluation results. Finally, in § V we present the limitations of our study, the lesson learned and our ongoing and future work.

II. Modeling Risky Paths in an Edge Network using Portfolio Theory

Aside from its applications for financial asset allocation, we argue that portfolio theory [7] is a valuable resource allocation tool also for traffic engineering problems, especially in mobile edge computing, where latency is as crucial as stock prices. In this section we first give a background on portfolio theory, applied to a standard financial portfolio selection problem and then we describe how we use it to model risky paths for an edge traffic engineering problem.

Background: Portfolio Theory. Maximizing the return is undoubtedly the first goal of every investor. The second main characteristic of an investment is the level of perceived risk to obtain such return, compared to the average over the investment period. Portfolio theory [7] formalizes the problem of selecting a portfolio $i.e.$, the set of items ($e.g.$, financial instruments) that maximizes the expected return given some level of risk. The problem can be alternatively formulated as a risk minimization problem, given an expected value of return. In general, determining what constitutes a desirable portfolio depends on many factors, including risk profile or psychology of the investor. We consider a portfolio p to be desirable if for a given expected rate of return μ_p the portfolio has the least variance σ_p^2. The classical portfolio problem considers n assets held over a period of time. Let us denote with z_i the dollar amount of asset i held throughout the investment period, at the price obtained at the beginning of the investment period. The simplest formulation does not consider obligations to buy assets at the end of the period, that would yield $z_i < 0$, so asset i always corresponds to $z_i > 0$; [2] We let p_i denote the relative price change of asset i over the period, $i.e.$, its change in price over the period divided by its price at the beginning of the period. The overall dollar return on the portfolio is hence given by $r = p^T z$, where the optimization variable is the portfolio vector $z \in \mathbb{R}^n$. A wide variety of constraints on the portfolio can be considered. Let us consider $\mathbf{1}^T z = B$, that is, the total budget to be invested is B, which is often normalized to one.

Considering a stochastic model for price (or latency) changes, we have that $p \in \mathbb{R}^n$ is a random vector, with known mean $\bar{p}$ and covariance Σ on the assets (paths) in the portfolio. Therefore, with portfolio $z \in \mathbb{R}^n$, the return r is a (scalar) random variable with mean $\bar{p}^T z$ and variance $z^T \Sigma z$. The choice of portfolio z involves a trade-off between the mean of the return and its variance. The portfolio optimization problem is the following quadratic program:

$$\begin{aligned} \underset{z}{\text{minimize}} \quad & z^T \Sigma z \\ \text{subject to} \quad & \bar{p} z \geq r_{min} \\ & \mathbf{1}^T z = 1, \\ & z_i > 0 \quad \forall i = 1, \ldots, n. \end{aligned} \tag{1}$$

[2] A more advanced model could captures also the short position investment strategy in asset i, $i.e.$, the obligation to buy the asset at the end of the period, that yields $z_i < 0$. We do not consider such "shorts" in our model.

The problem seeks the portfolio z that minimizes the return variance (associated with the risk), subject to achieving a minimum acceptable mean return r_{min} (in our case, network throughput), and satisfying the portfolio budget without shorting ($i.e.$, constraints $z_i > 0$). The risk of a small or large loss, $i.e.$, a change in portfolio values below its expected value, is directly related to the standard deviation, and increases with it. For this reason, the standard deviation (or the variance) is used as a measure of the risk associated with the portfolio.

Modeling Risky Paths in Mobile Edge Computing. We model the domain of financial instruments to be selected as physical paths, and the portfolio to be invested as virtual paths (or flows) connecting two end-points. We then model the expected return of our portfolio over a period of time (the time during which we hold the assets) as the throughput during the considered lifetime of a flow. The availability of all resources composing the portfolio (in our case physical links) fluctuates due to edge user mobility, failures, and the statistical multiplexing nature of connectionless networks. Packets corrupted or lost due to queuing delays or congestion increase throughput variance across each flow and the mobile and prone to failures nature of edge computing applications exacerbate such variations. For each path (flow) j, we model its risk (volatility) with z_{ij}, that is, as the probability of obtaining a given latency variance if we select z_{ij}. We now describe a method that we used to estimate such path latency, that can be in turn used as input of our optimization problem.

Throughput Variance Estimation via Semi-Definite Programming. In our model, we consider each virtual path to be a portfolio, where financial instruments composing such portfolios are physical nodes, and each virtual CPU capacity unit is the quantity that each virtual path requester invests on a candidate physical node. All virtual nodes to be allocated model the budget of the portfolio, usually normalized to one. In the classical portfolio optimization problem, the portfolio x is the optimization variable, and investors seek to minimize their risk subject to a minimum mean return, sometimes with other constraints as well (for example weather or not short sell is allowed). In classical portfolio optimization problems, the price statistic is known, that is, both average and variance of the price p and Σ are known parameters input to the problem. To manage our virtual paths, an alternative version of the classical portfolio optimization problem can be considered. In particular, we bound the risk (known thanks to our measurement history) and we assume that a portfolio (virtual path vector) x is known, but only partial information is available about the covariance matrix Σ. This is because Software-Defined Network (SDN) controllers are only informed about the status of a partial subset of the network status. Queuing delays in a distributed system, for example, can be at best approximated by a set of past measurement. These round-trip-time of instantaneous throughput measurement sample form the best upper and lower bound on each covariance matrix element. Therefore, we have:

$$L_{ij} \leq \Sigma_{ij} \leq U_{ij}, \tag{2}$$

where L and U are given by the past measurements on the given virtual path (portfolio). We then seek the virtual path allocation with minimum throughput variance (risk), over all covariance matrices consistent with the given bounds. We define the worst-case variance of the virtual path as:

$$\sigma_{wc}^2 = \sup\{x^T \Sigma x | L_{ij} \leq \Sigma_{ij} \leq U_{ij}, \Sigma_{ij} \geq 0 \quad i,j = 1,\ldots,n\}. \tag{3}$$

Hence, to decide how to allocate or migrate a virtual path with expected throughput variance (risk) bound, we solve the following semidefinite program:

$$\begin{aligned} \underset{\Sigma}{\text{maximize}} \quad & x^T \Sigma x \\ \text{subject to} \quad & L_{ij} \leq \Sigma_{ij} \leq U_{ij} \; i, \; j = 1,\ldots,n, \\ & \Sigma_{ij} \geq 0 \quad i,j = 1,\ldots,n. \\ & \sum_{j \in V_H} \Sigma_{ij} = 1 \qquad \forall i \end{aligned} \tag{4}$$

Candidate hosting physical paths (*i.e.*, portfolio allocations) are usually fairly short even in large networks, so the dimension of the problem is small, and the time to solution sharp. An algorithm can simultaneously solve a large number of instances of the problem with different portfolio candidate and pick the physical path with the lowest risk bound.

III. Bayesian Networks to predict Risky Paths

A Bayesian network is a probabilistic graphical model containing random variables and their conditional dependencies, expressed by a directed acyclic graph. The edge $x_j \rightarrow x_i$ represents the dependency of the random variable x_i on the random variable x_j, in which x_j is the parent of child x_i. Each random variable has a corresponding conditional probability table (CPT) that is used to determine the conditional probability P($x_i|x_j$), the probability of x_i given x_j, where $P(x_i|x_j) = \dfrac{P(x_i \cap x_j)}{P(x_j)}$ We define the set of parents for the random variable x_i to be Pa(x_i). To construct a Bayesian network from discretized data, we follow the constraint-based approach outlined by Koller and Friedman [5].

Our approach is based on the following three steps: (1) Discretize latency measurements using k-means clustering. (2) Construct a Bayesian network with discretized latency data and (3) Predict latency using the Bayesian network by maximizing conditional probability P(x_t | Pa(x_t)).

We use k-means clustering to discretize path latency measurements, and we found the optimal number of clusters is decided empirically. We then construct a Bayesian network using the discretized latency measurements. Each node in the network represents the latency at a point in time relative to time t, with the aim of predicting the latency at time t. The set of nodes in the network is: $\{x_{t-n}, x_{t-n+1}, ..., x_{t-1}, x_t\}$. The set of possible states for a node is composed of the means of the clusters. Once we construct the Bayesian network, we use it to predict path latency at time t given the latency measurements for all nodes in the set Pa(x_t). Similar approaches have been used to predict daily stock price fluctuations [11]. We obtain the discrete latency values for parent nodes of x_t. We then predict the discrete latency by computing the conditional probability P(x_t | Pa(x_t)) for each of the possible states of x_t. The predicted latency a_t is the state a_l of node x_t in which the conditional probability is maximized. To construct a portfolio, we predict the future latency of each path in a network. We then select the set of paths which minimizes overall predicted latency. Using this portfolio, we can dynamically steer traffic such that risk is minimized.

IV. Evaluation

We evaluate our method against an autoregressive model, a moving average model, and an ARMA (autoregressive moving average) model. These methods captures the time series of latency values. Unlike Bayesian networks, they do not model the dependencies within a dataset. In particular, AR(p), an autoregressive model of order p, is constructed by regressing a value within a time series against previous values in the series. Latency is then forecasted by the model, using a linear combination of previous latency values. MA(q), a moving average model of order q, is instead created from a linear combination of past white noise error terms. Finally, the ARMA model combines AR(p) and MA(q). The variable to be predicted is regressed against its past values, and its error term is then modeled from previous error values.

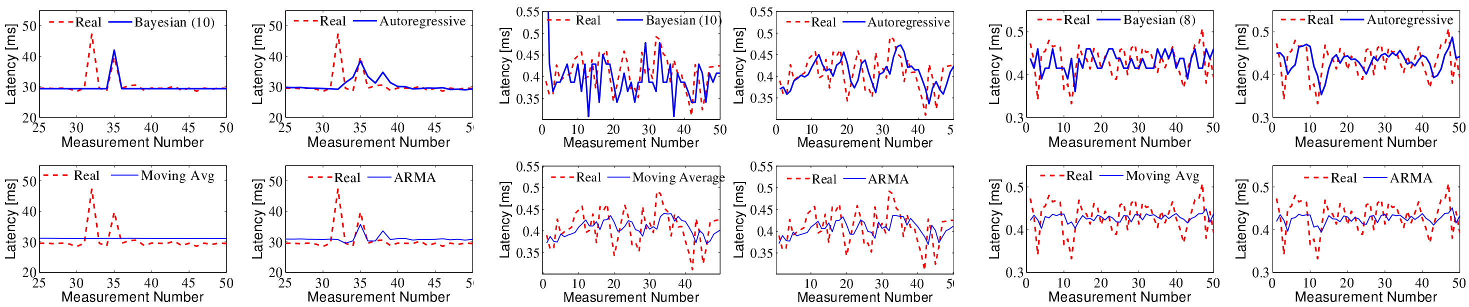
Forecasts are made by these time series models by combining past values. Consistent with the Bayesian network method, each predicted latency value is a one-step-ahead forecast.

We summarize our results in a few take home messages. *(1) Our results suggest that the Bayesian network approach predicts end-to-end latency (peak) values more accurately with respect to the benchmark approaches autoregressive, moving average and autoregressive moving average models* (Figure 1). Our evaluation include both a large dataset of traffic measurements, available at [9], as well as our own ICMP traffic measurements (*i.e.* `ping`) from a server located in the Saint Louis area towars the `Stanford.edu` web server. Moreover, we ping a couple of virtual machines within the GENI testbed [1] without reserving any virtual link bandwidth. We emulated an edge network pinging VMs in the same GENI rack as well as across two close enough East-coast GENI racks. To quantify the superior behavior of the Bayesian network approach, we show the distribution of errors across the tested methods in Figure 29.

Finding the optimal number of clusters that minimize the prediction error is still an open question, but our results suggest that, *(2) as long as there is dependency among latency measurements, 8 or 10 clusters produce the best accuracy, especially for latency peaks* (Figure $2-10$).

(3) Lower Density in the Bayesian network means lower prediction accuracy. To further dissect this observation, we observed the number of edges present in the Bayesian network. By definition, the density of such stochastic data structure is a proxy of the interdependencies among the time series. Even though determining what causes the flow time series to have loose or tight latency interdependencies requires a deeper analysis, focus of our current work, we empirically observe that lack of data dependencies means ineffectiveness of the Bayesian network predictor. This is not surprising, given the nature of the predictor, and it is further confirmed when the latency time series are sampled from a random distribution (results not shown).

(4) Our results are confirmed using across a large dataset of TCP Internet traffic traces, available at [9]. Our results on this dataset matched the results obtained from the GENI testbed, with the Bayesian network outperforming the autoregressive, moving average, and autoregressive-moving average models. We found that repeated experiments on this dataset with a consistent number of clusters produced different cluster



(*a*) SLU - Stanford (*b*) GENI: GPO Rack (*c*) GENI: GPO - University of Washington

Fig. 1. The Bayesian Network approach predicts e2e latencies more accurately w.r.t. Autoregressive, Moving Average and Autoregressive moving average: (a) ICMP traffic from Saint Louis to Stanford.edu (b) ICMP traffic across VMs in the same GENI rack. (c) ICMP traffic across two East-coast GENI racks.

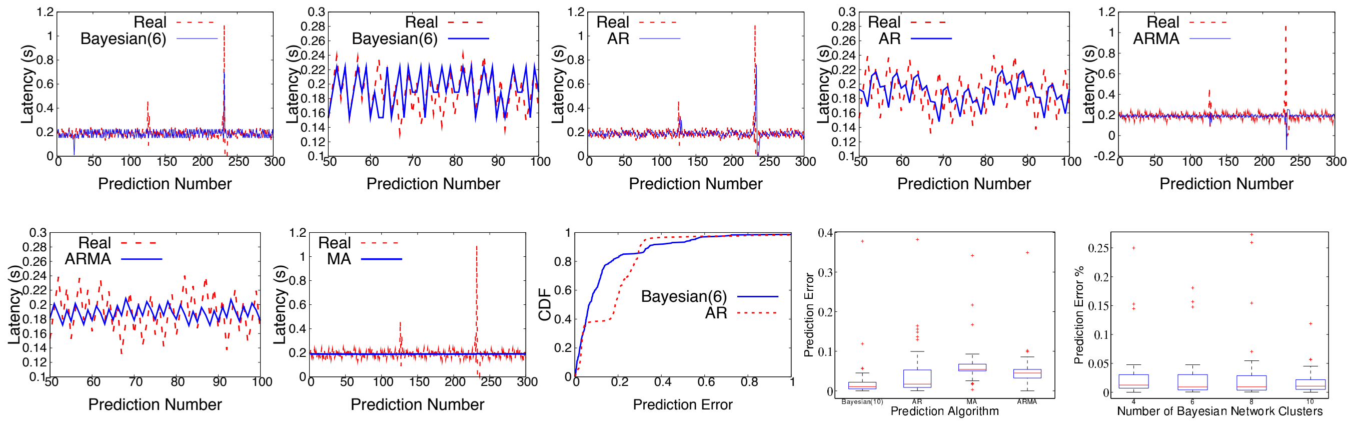


Fig. 2. (1-8) The Bayesian model produced lower errors than the autoregressive model for approximately 64% of predictions for the University of Naples Federico II dataset (first 8 plots). (9) The Bayesian network prediction with 10 clusters has the lowest latency prediction error w.r.t. Autoregressive, Moving Average and Autoregressive moving average. (10) Impact of the number of clusters in the Bayesian network on the prediction error.

centers because initial cluster centers are chosen randomly in the k-means method (used to construct the Bayesian network). This, however, surprisingly, did not have a noticeable affect on the accuracy of the Bayesian network (Figure 2).

V. Conclusions

In this paper, we modeled the path selection problem for edge traffic engineering using portfolio theory, and we compared different path prediction methods, used in real stock market time series analysis, to estimate path selection risks. Although our results suggest that a Bayesian Network approach may lead to good latency (peak) estimation performance, as long as there are dependencies among the time series path latency measurements, our finding are far from ideal for several reasons. The accuracy of the Bayesian network is limited by its ability to correctly identify and model dependencies in the data. If the variance within a dataset is small, or the dependencies between variables are weak, the Bayesian network is likely to miss the dependencies. Additionally, the clustering process is likely to hide small dependencies because the variance between two data points can be lost when they belong to the same cluster, as we use the cluster mean as input to the Bayesian network. Our results were also limited by the consistency of the latency values in our data. We found that this resulted in the cluster means having a small spread. When the Bayesian network predicted the incorrect cluster, the value of the predicted cluster was close to the actual latency value, resulting in low errors despite the erroneous prediction.

Acknowledgment

This work has been supported by the National Science Foundation award CNS-1647084.

References

[1] Global Environment for Network Innovations http://www.geni.net.

[2] Mary Hogan and Flavio Esposito. Poster: A portfolio theory approach to edge traffic engineering via bayesian networks. In *In Proc. of the 23rd ACM Annual International Conference on Mobile Computing and Networking (MOBICOM)*, page na, Snowbird, Utah, USA, 2017. ACM.

[3] Keith Winstein et al. Stochastic forecasts achieve high throughput and low delay over cellular networks. In *USENIX NSDI 13*, pages 459–471, Lombard, IL.

[4] Kok-Kiong Yap at al. Taking the Edge off with Espresso: Scale, Reliability and Programmability for Global Internet Peering. In *Proceedings of ACM SIGCOMM*, 2017.

[5] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, Cambridge, MA.

[6] M. Alizadeh et al. Data Center TCP (DCTCP). In *SIGCOMM 2010*.

[7] Harry M. Markowitz. *Mean-variance analysis in portfolio choice and capital markets*. Blackwell, Oxford [u.a.], 1987.

[8] Meinders. RIPE NCC Regional Meeting: Eurasia Network Operators Group (ENOG 11), 2016.

[9] University of Naples Federico II. Traffic Traces. Available at: http://traffic.comics.unina.it/Traces/ttraces.php.

[10] Brandon Schlinker, Hyojeong Kim, Timothy Cui, Ethan Katz-Bassett, Harsha V. Madhyastha, Italo Cunha, James Quinn, Saif Hasan, Petr Lapukhov, James, and Hongyi Zeng. engineering egress with edge fabric: Steering oceans of content to the world.

[11] Yi Zuo and Eisuke Kita. Stock price forecast using bayesian network. *Expert Systems with Applications*, 39(8):6729–6737, June 2012.