Dynamic Resource Scaling for VNF over Nonstationary Traffic: A Learning Approach

Kaige Qu, *Student Member, IEEE,* Weihua Zhuang, *Fellow, IEEE*, Xuemin (Sherman) Shen, *Fellow, IEEE*, Xu Li, and Jaya Rao

***Abstract*—Software defined networking (SDN) and network function virtualization (NFV) are key enablers for service-level customized network slicing in fifth generation (5G) core networks. Network slices are required to be isolated from each other in terms of service performance with traffic load fluctuations. In this paper, the virtual network function (VNF) scalability issue is studied to meet the quality-of-service (QoS) requirement in the presence of nonstationary traffic, through joint VNF migration and resource scaling. A traffic parameter learning method based on change point detection and Gaussian process regression (GPR) is proposed, to learn traffic parameters in a fractional Brownian motion (fBm) traffic model for each stationary traffic segment within a nonstationary traffic trace. Then, the time-varying VNF resource demand is predicted from the learned traffic parameters based on an fBm resource provisioning model. With the detected change points and predicted resource demands, a VNF migration problem is formulated as a Markov decision process (MDP) with variable-length decision epochs, to maximize the long-term reward integrating load balancing, migration cost, and resource overloading penalty. A penalty-aware deep Q-learning algorithm is proposed to incorporate awareness of resource overloading penalty, with improved performance over benchmarks in terms of training loss reduction and cumulative reward maximization.**

***Index Terms*—Virtual network function (VNF), resource scaling, migration, nonstationary traffic, change point detection, traffic parameter learning, Gaussian process regression, resource demand prediction, reinforcement learning**

I. Introduction

The service-oriented fifth-generation (5G) core networks are featured by customized services with differentiated multi-dimensional performance requirements, which can be provisioned through network slicing enabled by the promising software defined networking (SDN) and network function virtualization (NFV) paradigms [1]–[4]. A network slice supports a composite service via virtual network function (VNF) chaining, with dedicated packet processing functionality at each VNF, such as intrusion detection system (IDS) and firewall. Packets processed by a VNF are transmitted to next VNF in the same VNF chain for further processing, generating traffic between consecutive VNFs, i.e., inter-VNF subflows. During the planning of network slices, VNFs are placed at NFV-enabled commodity servers or data centers, referred to as NFV nodes, and inter-VNF subflows are routed over physical paths between the locations of corresponding VNFs [5]. The VNFs and subflows are allocated with static amounts of processing and transmission resources, respectively, according to the estimation of long-term resource demands [5], [6]. During the operation of network slices, traffic arrivals of each service fluctuate over time, possibly leading to a mismatch between traffic load and resource availability, which is detrimental to service performance and resource utilization. With the flexibility provided by SDN and NFV, it is possible to migrate VNFs among several candidate NFV nodes in a neighborhood, with elastic processing resource allocation for consistent quality-of-service (QoS) guarantee in the presence of traffic fluctuations [7], [8]. Accordingly, the subflows are rerouted over alternative physical paths to the new VNF locations.

With time-varying traffic, the processing resource demand of a VNF is dependent on both the statistics of traffic arrivals and the QoS requirement. We consider delay-sensitive VNF chains with a stringent end-to-end (E2E) delay requirement [9], [10]. Suppose that the E2E delay requirement is decomposed into per-hop delay requirement at each VNF. For example, the probability of packet processing (including queueing) delay at a VNF exceeding a certain delay bound should not be beyond an upper limit. With changes in traffic statistics, the processing resource demand of a VNF varies for a certain QoS requirement. Existing studies usually assume prior knowledge about the time-varying resource demands or predict the future resource demands based on historical resource demand information [11]–[14]. The average traffic rate in a certain time duration is usually considered as the resource demand [13], [14]. However, resource allocation/scaling according to the average traffic rate is not sufficient to satisfy a stringent delay requirement. In reality, a resource demand trace with inherent QoS guarantee is difficult to obtain. Instead, a traffic trace with packet arrival information is usually available [15]. Therefore, a resource demand prediction scheme is required, to predict the time-varying QoS-aware resource demands following the traffic statistical changes detected in an available packet arrival traffic trace. Then, VNF scaling decisions can be made, to scale up/down the amount of resources allocated to the VNFs according to the predicted resource demands and update the placement of VNFs among several candidate NFV nodes. There can be overlapping among the sets of candidate NFV nodes for different VNFs. Here we consider one VNF in a neighborhood with several candidate NFV nodes, and treat

This work was financially supported by research grants from Huawei Technologies Canada and from the Natural Sciences and Engineering Research Council (NSERC) of Canada.

Kaige Qu, Weihua Zhuang, and Xuemin (Sherman) Shen are with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, N2L 3G1 (emails: {k2qu, wzhuang, sshen}@uwaterloo.ca).

Xu Li and Jaya Rao are with Huawei Technologies Canada Inc., Ottawa, ON, Canada, K2K 3J1 (emails: {Xu.LiCA, jaya.rao}@huawei.com).

the dynamics of other VNFs as background traffic at the NFV nodes. The dynamics of other VNFs are attributed to dynamics in both their traffic arrivals and scaling decisions.

There are several existing studies on dynamic VNF placement and traffic routing, based on decisions made in a proactive or reactive manner at consecutive non-overlapping decision epochs of equal length, e.g., 30 minutes [11], [13], [14]. The selection of epoch length is difficult and usually based on experience. If the decision epoch is too long, traffic burstiness in different time granularities within an epoch cannot be captured, resulting in challenges for continuous QoS guarantee; if the decision epoch is too short, decisions are made frequently, possibly resulting in unnecessary expensive VNF migrations for temporal short traffic bursts. A better way is to adopt adaptive epoch length according to changes in traffic statistics (e.g., mean and variance) and resource demands. Several change point detection algorithms, either retrospective or online, have been developed for detecting structural breaks in a nonstationary time series [16]–[18]. Online algorithms provide inference about change points as each data sample arrives, which is more appropriate for detecting traffic statistical changes, based on which VNF scaling decisions can be made reactively without a significant latency [17], [18]. Under the assumption that a nonstationary traffic trace can be partitioned into consecutive stationary traffic segments with unknown change points, the decision epochs with variable lengths are to be identified based on change point detection. Each stationary traffic segment corresponds to one decision epoch. Traffic arrivals of a VNF are from a service-level flow which is an aggregation of traffic flows of different users. In core and backbone networks, the aggregation level is high, which makes Gaussian traffic approximation work well beyond a timescale of around 100 ms [19]. Gaussianity of a certain distribution can be checked by quantile-quantile (QQ) plot versus a standard Gaussian distribution [20]. Fractional Brownian motion (fBm) is a Gaussian process with properties such as self-similarity and long-range dependence (LRD) which comply with the properties of real-world network traffic [19], [21]. Hence, we adopt the fBm traffic model, based on which the characteristic traffic parameters of each stationary traffic segment are learned, and the corresponding resource demands are predicted.

At each decision epoch, a VNF scaling decision is made, which possibly requires VNF migrations. We use VNF scaling decision and VNF migration decision interchangeably. A VNF migration incurs migration cost, such as signaling overhead to reroute traffic and resource overhead to transfer VNF states associated with the VNF, which should be minimized [22]. On the other hand, a balanced load distribution among the candidate NFV nodes makes the network more robust to future traffic variations, which is beneficial for long-term efficient resource utilization [22], [23]. There is a trade-off between the two goals. For example, a pure load balancing solution may result in frequent and expensive VNF migrations. Therefore, we jointly consider the two objectives, by jointly minimizing the migration cost and the maximum resource loading factor among all candidate NFV nodes. Moreover, there is another trade-off between cost minimizations in the short term and in the long run. When a VNF migration is required, the VNF is migrated to the current most lightly loaded NFV node for cost minimization in the current decision epoch. However, a lightly loaded NFV node can become heavily loaded in the future due to increasing background resource usage, resulting in further migrations to avoid performance degradation. In contrast, for cost minimization in the long run, the VNF should be migrated to an NFV node which is expected not to be heavily loaded in the current and successive decision epochs. Reinforcement learning (RL) provides an approach for long-run cost minimization, with the ability to capture inherent patterns in network dynamics and to make intelligent decisions accordingly [14], [24]–[29].

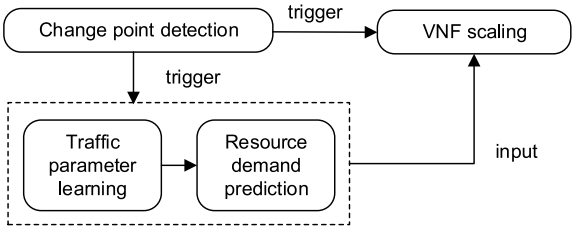


Fig. 1: Workflow of dynamic VNF scaling for nonstationary traffic.

In this paper, a dynamic VNF scaling problem is studied, to meet the delay requirement in the presence of nonstationary traffic. The new contributions are summarized as follows, with a workflow given in Fig. 1.

- To provide QoS guarantee for the VNF with nonstationary traffic input, a change-point-driven traffic parameter learning and resource demand prediction scheme is proposed. First, the prior-unknown change points of the nonstationary traffic are detected online, using a Bayesian online change point detection (BOCPD) algorithm with post-processing, which identifies the boundaries between consecutive stationary traffic segments. Then, the fBm traffic parameters are learned for the upcoming stationary traffic segment after each newly detected change point, using Gaussian process regression (GPR) with an fBm kernel (covariance) function. Afterwards, the resource demand of the upcoming stationary traffic segment is predicted based on an fBm resource provisioning model;
- With the detected change points and predicted resource demands, a VNF migration problem is formulated as a Markov decision process (MDP) with variable-length decision epochs, to minimize the overall cost integrating imbalanced loading, migration cost, and resource overloading penalty in the long run. A deep Q-learning algorithm with penalty-aware prioritized experience replay is proposed to solve the MDP, with performance gains in terms of both cost and training loss reduction compared with benchmark algorithms.

The rest of this paper is organized as follows. The system model is presented in Section II, and the change-point-driven resource demand prediction scheme is proposed in Section III. Section IV presents the MDP formulation and the penalty-aware deep Q-learning algorithm. Performance evaluation is given in Section V, and conclusions are drawn in Section VI.

II. System Model

A. Network Scenario

A service request is represented as a VNF chain, originating from a source node and traversing through a number of VNFs in sequence towards a destination node. The source and destination nodes are seen as dummy VNFs. The aggregate traffic stream between two consecutive VNFs is referred to as an inter-VNF subflow. We consider one VNF in the VNF chain, with an incoming subflow from its upstream VNF, and an outgoing subflow towards its downstream VNF. For packet processing at the VNF, it is required that the delay violation probability should not exceed an upper limit, i.e., $\Pr(d > D) \leq \varepsilon$, where d is a random variable denoting the experienced VNF packet processing (including queueing) delay, D is the delay bound, and ε is the maximum delay violation probability. The VNF can be placed at an NFV node in a candidate set $\mathcal{N}_C$. The considered VNF is initially placed at NFV node $n_0 \in \mathcal{N}_C$.

B. Nonstationary Traffic Model

Multi-Timescale Time Series: Traffic arrivals at the VNF can be represented as a time series, with each traffic sample being the number of packet arrivals in non-overlapping, successive time intervals. We consider traffic arrivals in different timescales, including a medium timescale with interval length (in second) equal T_M (e.g., 20 s), and a small timescale with interval length (in second) equal T_S (e.g., 0.1 s). Let $\boldsymbol{x}_M$ denote the time series in medium timescale, given by

$$\boldsymbol{x}_M = [x_M(0), x_M(1), \cdots, x_M(m), \cdots] \tag{1}$$

where m (≥ 0) is an index for the medium time interval and $x_M(m)$ is the m-th traffic sample in medium timescale, representing the number of packet arrivals in the m-th medium time interval. A series of traffic samples between medium time intervals m and m' (inclusive) is given by

$$\begin{aligned}\boldsymbol{x}_M[m:m'] = [&x_M(m), x_M(m+1), \cdots, \\ &x_M(m'-1), x_M(m')], \quad m' > m.\end{aligned} \tag{2}$$

Similarly, a small-timescale time series is represented as

$$\boldsymbol{x}_S = [x_S(0), x_S(1), \cdots, x_S(t), \cdots] \tag{3}$$

where t (≥ 0) is an index for the small time interval and $x_S(t)$ is the t-th traffic sample in small timescale. Let $\boldsymbol{x}_S[t:t']$ denote a series of traffic samples between small time intervals t and t' (inclusive), given by

$$\boldsymbol{x}_S[t:t'] = [x_S(t), x_S(t+1), \cdots, x_S(t'-1), x_S(t')], \ t' > t. \tag{4}$$

Let $A(t)$ denote the cumulative number of packet arrivals before small time interval t, given by

$$A(t) = \begin{cases} \sum_{t'=1}^{t} x_S(t'-1), & t \geq 1 \\ 0, & t = 0. \end{cases} \tag{5}$$

Letting Λ be the long-term average traffic rate in packet/s, the following relationship holds:

$$\Lambda = \lim_{m' \to \infty} \frac{1}{m'} \sum_{m=0}^{m'} \frac{x_M(m)}{T_M} = \lim_{t' \to \infty} \frac{1}{t'} \sum_{t=0}^{t'} \frac{x_S(t)}{T_S} \tag{6}$$

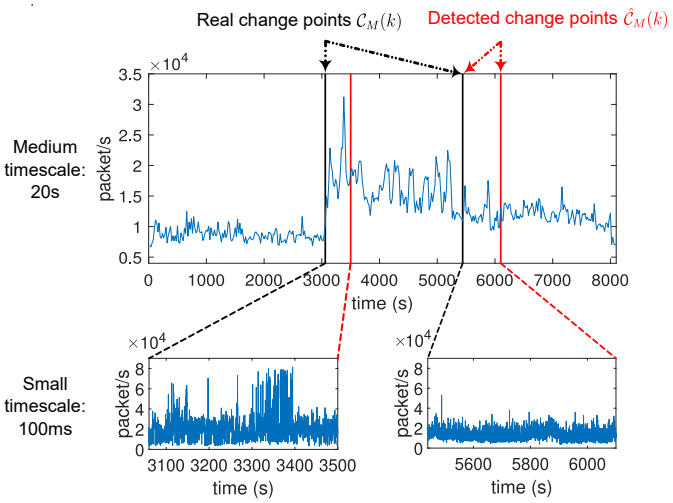


Fig. 2: An illustration of nonstationary traffic model in different timescales.

where $\frac{x_M(m)}{T_M}$ and $\frac{x_S(t)}{T_S}$ are average traffic rates (in packet/s) in the m-th medium time interval and the t-th small time interval, respectively. Assume that T_M is multiples of T_S. As illustrated in Fig. 2, a medium-timescale time series can be mapped to a small-timescale time series within the same time duration, represented as

$$\boldsymbol{x}_M[m:m'] \Rightarrow \boldsymbol{x}_S\left[\frac{mT_M}{T_S} : \left(\frac{(m'+1)T_M}{T_S} - 1\right)\right]. \tag{7}$$

Stationary Traffic Segments with Unknown Change Points: The real-world network traffic usually exhibits nonstationarity [30]. Here, we consider nonstationary traffic arrivals for the VNF, as illustrated in Fig. 2. Assume that the nonstationary traffic time series can be partitioned into non-overlapping stationary traffic segments with unknown change points in time. Between two neighboring change points, traffic statistics such as mean and variance do not change. Let integer k (≥ 0) indicate the k-th stationary traffic segment. Consider that the change points can be located by a change point detection algorithm based on traffic statistical changes in the medium-timescale time series. Let $\mathcal{C}_M(k)$ be the index of the k-th change point in medium timescale, i.e., $x_M(\mathcal{C}_M(k))$ is the first traffic sample (in medium-timescale) of the k-th stationary traffic segment. We have $\mathcal{C}_M(0) = 0$ to indicate the beginning of the timeline. Correspondingly, the k-th change point in small timescale, $\mathcal{C}_S(k)$, is given by

$$\mathcal{C}_S(k) = \frac{\mathcal{C}_M(k) T_M}{T_S}. \tag{8}$$

Factional Brownian Motion for a Stationary Traffic Segment: A standard fBm process $\{Z_s(t), t = 0, 1, \cdots\}$ is a centered Gaussian process with $Z_s(0) = 0$ and covariance function

$$\psi_{Z_s}(t_1, t_2) = \frac{1}{2}\left(t_1^{2H} + t_2^{2H} - |t_1 - t_2|^{2H}\right) \tag{9}$$

where $H \in (0, 1)$ is Hurst parameter [21]. For $H \in [0.5, 1)$, the fBm process is both self-similar and LRD. A general fBm process $\{Z(t), t = 0, 1, \cdots\}$, denoting the the cumulative number of packet arrivals before the t-th time unit in a

stationary traffic time series, is represented by

$$Z(t) = \lambda t + \sigma Z_s(t) \tag{10}$$

where $\lambda = \mathbb{E}(\frac{Z(t)}{t})$ is the mean of packet arrivals in a time unit, σ is the standard deviation of packet arrivals in a time unit [21]. Here, a time unit corresponds to a small time interval. The covariance function of $Z(t)$ is given by

$$\psi_Z(t_1, t_2) = \frac{\sigma^2}{2}\left(t_1^{2H} + t_2^{2H} - |t_1 - t_2|^{2H}\right). \tag{11}$$

The fBm traffic model is adopted for a stationary traffic segment. For the k-th stationary traffic segment, we consider a shifted discrete timeline in small timescale, $\dot{t}$, starting at the beginning of the k-th stationary traffic segment, with $\dot{t} = t - \mathcal{C}_S(k)$. Then, we have $\dot{x}_S(\dot{t}) = x_S(t - \mathcal{C}_S(k))$, representing the number of packets arrived in the $\dot{t}$-th shifted small time interval. The cumulative number of packet arrivals in the k-th stationary traffic segment before $\dot{t}$ is modeled as an fBm process with traffic parameters $\{\lambda(k), \sigma(k), H(k)\}$, given by

$$\dot{A}_k(\dot{t}) = \begin{cases} \sum_{\dot{t}'=1}^{\dot{t}} \dot{x}_S(\dot{t}'-1), & 1 \le \dot{t} \le C_S(k+1) - C_S(k) - 1 \\ 0, & \dot{t} = 0. \end{cases} \tag{12}$$

III. Traffic Parameter Learning and Resource Demand Prediction

Since traffic statistics change across different stationary traffic segments, the amount of processing resources allocated to the VNF for probabilistic QoS guarantee, i.e., $\Pr(d > D) \le \varepsilon$, should be dynamically adjusted. Here, a change-point-driven traffic parameter learning and resource demand prediction scheme is proposed, to predict resource demands from learned fBm traffic parameters of stationary traffic segments between detected change points. It provides a triggering signal for dynamic VNF migration to be discussed in Section IV.

A. Bayesian Online Change Point Detection

The Bayesian online change point detection (BOCPD) algorithm was first introduced in [17]. Central to the BOCPD algorithm is the run length denoted by L. A run is defined as a traffic segment with the same statistics. Online inference about the run length is performed at every time step, given a conditional prior distribution over the run length and an underlying predictive model. We use the BOCPD algorithm to detect statistical changes in mean and variance of the nonstationary medium-timescale time series $\boldsymbol{x}_M$, under the assumption that the medium-timescale traffic samples are from i.i.d Gaussian distribution $\mathcal{N}(\mu, \nu^2)$, with unknown (and perhaps changing) mean μ and variance ν^2. A time step in the BOCPD algorithm corresponds to a medium time interval. Note that the i.i.d Gaussian assumption is used to detect change points. For traffic parameter learning, we do not rely on such an assumption.

The run length at the m-th time step, denoted by L_m, represents the number of traffic samples before the m-th traffic sample, $x_M(m)$, within the same run. The run length L_m is a random variable taking values from $\{0, 1, \cdots, m\}$, as illustrated in Fig. 3. From time step $(m-1)$ to m, the run length either increases by 1 or resets to 0. For notation simplification, we omit the subscript M denoting the medium timescale, and use $\boldsymbol{x}_m$ to denote $\boldsymbol{x}_M[0:m]$. We also use $\boldsymbol{x}_m^{(L)}$ to denote $x_M[(m - L_m) : m]$, which is a time series in the same run before the $(m+1)$-th traffic sample, given the run length L_m at time step m.

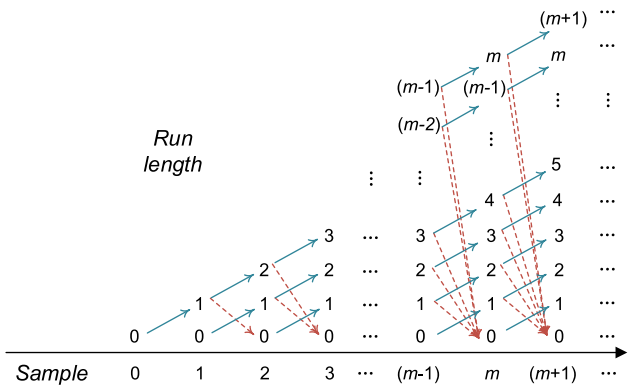


Fig. 3: An illustration of run length growth.

The joint probability of run length and observed time series at time step m, i.e., $\Pr(L_m, \boldsymbol{x}_m)$, is updated recursively from the joint probability at the previous time step, i.e., $\Pr(L_{m-1}, \boldsymbol{x}_{m-1})$, for $m \ge 1$, given by

$$\overbrace{\Pr(L_m, \boldsymbol{x}_m)}^{m\text{-th iteration}} = \sum_{L_{m-1}} \left\{ \underbrace{\Pr(L_m | L_{m-1})}_{\text{conditional prior on run length}} \underbrace{\Pr(x_m | L_{m-1}, \boldsymbol{x}_{m-1}^{(L)})}_{\text{predictive model}} \overbrace{\Pr(L_{m-1}, \boldsymbol{x}_{m-1})}^{(m-1)\text{-th iteration}} \right\}. \tag{13}$$

With initialization $\Pr(L_0 = 0, x_0) = 1$, for any observed value of x_0, the joint probability represents a relative likelihood. The underlying condition for (13) is that run length L_m is independent of $\boldsymbol{x}_m$, given L_{m-1}. The conditional prior on run length, i.e., $\Pr(L_m | L_{m-1})$, is a probability mass distribution with two outcomes, i.e., $L_m = L_{m-1}+1$ and $L_m = 0$, as given in Appendix. The predictive model, i.e., $\Pr(x_m | L_{m-1}, \boldsymbol{x}_{m-1}^{(L)})$, evaluates the probability that x_m belongs to the same run as $\boldsymbol{x}_{m-1}^{(L)}$ (i.e., $x_M[(m - 1 - L_{m-1}) : (m-1)]$), given L_{m-1}. With a Gaussian-Inverse-Gamma prior on the unknown mean, μ, and variance, ν^2, of the i.i.d Gaussian distribution, the predictive model is described by a student-t distribution with mean $\mu_{m-1}^{(L)}$ and standard deviation $\nu_{m-1}^{(L)}$, as given in Appendix. For each possible value of L_{m-1}, both $\mu_{m-1}^{(L)}$ and $\nu_{m-1}^{(L)}$ take different values. Through normalization, the posterior distribution of run length, $\Pr(L_m | \boldsymbol{x}_m)$, is given by

$$\Pr(L_m = m' | \boldsymbol{x}_m) = \frac{\Pr(L_m = m', \boldsymbol{x}_m)}{\sum_{L_m=0}^{m} \Pr(L_m | \boldsymbol{x}_m)}, \quad \forall m' = 0, 1, \cdots, m. \tag{14}$$

For traffic parameter learning and resource demand prediction, deterministic change points are required. Define the most probable run length at time step m as

$$\hat{L}_m = \text{argmax}_{L_m = \{0, \cdots, m\}} \Pr(L_m | \boldsymbol{x}_m). \tag{15}$$

The mean and standard deviation of the student-t predictive model corresponding to the most probable run length at time step m, i.e., $\hat{L}_m$, is seen as the estimated mean and standard deviation of the nonstationary medium-timescale time series at time step m, denoted by $\mu_m^{(\hat{L}_m)}$ and $\nu_m^{(\hat{L}_m)}$ respectively. Time step m is identified as a change point if the following two conditions are satisfied. First, the gap between the most probable run lengths at time steps $(m-1)$ and m, i.e., $\hat{L}_{m-1}$ and $\hat{L}_m$, is larger than a threshold Δ_L, given by

$$\hat{L}_{m-1} - \hat{L}_m > \Delta_L; \tag{16}$$

Second, the normalized absolute difference between the estimated mean plus standard deviation at time step m and $(m-1)$ is beyond a predefined threshold Δ_d, given by

$$\frac{\left|\left(\mu_m^{(\hat{L}_m)}+\nu_m^{(\hat{L}_m)}\right)-\left(\mu_{m-1}^{(\hat{L}_{m-1})}+\nu_{m-1}^{(\hat{L}_{m-1})}\right)\right|}{\mu_{m-1}^{(\hat{L}_{m-1})}+\nu_{m-1}^{(\hat{L}_{m-1})}} > \Delta_d. \tag{17}$$

The k-th detected change point, denoted by $\hat{\mathcal{C}}_M(k)$, is an estimated value of the real change point $\mathcal{C}_M(k)$, i.e., the index of the first medium-timescale traffic sample in the k-th stationary traffic segment. The BOCPD algorithm has a linear space and time complexity per time-step in the number of medium-timescale traffic samples after the previously detected change point [17]. The stochastic BOCPD method results in a latency between $\mathcal{C}_M(k)$ and $\hat{\mathcal{C}}_M(k)$, as illustrated in Fig. 2, in which the real and detected change points are indicated by the black and red vertical lines, respectively. The latency cannot be avoided since it is inherent to the BOCPD algorithm. We exploit the latency for a *look-back* traffic parameter learning.

B. Traffic Parameter Learning

Let m_0 be a small integer[1] such that (m_0-1) medium-timescale traffic samples before the $\hat{\mathcal{C}}_M(k)$-th one belong to the k-th stationary traffic segment. The m_0 medium-timescale traffic samples including the $\hat{\mathcal{C}}_M(k)$-th one correspond to $\frac{m_0T_M}{T_S}$ small-timescale traffic samples within the same time duration, given by

$$\begin{aligned}&\boldsymbol{x}_M[(\hat{\mathcal{C}}_M(k)-m_0+1):\hat{\mathcal{C}}_M(k)]\Rightarrow\\&\boldsymbol{x}_S\left[\frac{(\hat{\mathcal{C}}_M(k)-m_0+1)T_M}{T_S}:\left(\frac{(\hat{\mathcal{C}}_M(k)+1)T_M}{T_S}-1\right)\right]\\&=\boldsymbol{x}_S\left[\hat{\mathcal{C}}_S(k):\left(\hat{\mathcal{C}}_S(k)+\frac{m_0T_M}{T_S}-1\right)\right]\end{aligned} \tag{18}$$

where $\hat{\mathcal{C}}_S(k)=\frac{(\hat{\mathcal{C}}_M(k)-m_0+1)T_M}{T_S}$ is the estimated k-th change point in small timescale. The $\frac{m_0T_M}{T_S}$ traffic samples are used to learn fBm traffic parameters of the k-th stationary traffic segment. Compared with a *look-ahead* counterpart, the *look-back* mechanism avoids another latency after the detected change point, for collecting sufficient traffic samples.

We consider a modified shifted discrete timeline, $\tilde{t}$, with $\tilde{t} = t - \hat{\mathcal{C}}_S(k)$, for the k-th stationary traffic segment. Correspondingly, we have $\tilde{x}_S(\tilde{t}) = x_S(t-\hat{\mathcal{C}}_S(k))$, representing the number of packets arrived in the $\tilde{t}$-th modified shifted small time interval. The cumulative number of packet arrivals in the k-th stationary traffic segment before $\tilde{t}$, is given by

$$\tilde{A}_k(\tilde{t})=\begin{cases}\sum_{\tilde{t}'=1}^{\tilde{t}}\tilde{x}_S(\tilde{t}'-1), & 1\le\tilde{t}\le\hat{C}_S(k+1)-\hat{C}_S(k)-1\\ 0, & \tilde{t}=0.\end{cases} \tag{19}$$

We use $\{\tilde{A}_k(\tilde{t}), 0 \le \tilde{t} \le \frac{m_0T_M}{T_S}-1\}$ to learn fBm traffic parameters of the k-th stationary traffic segment. Consider the following Gaussian process regression (GPR) model

$$\tilde{A}_k(\tilde{t}) \sim \mathcal{GP}(\lambda(k)\tilde{t}, \psi_k(\tilde{t}_1,\tilde{t}_2)) \tag{20}$$

where $\lambda(k)\tilde{t}$ is the mean function and $\psi_k(\tilde{t}_1,\tilde{t}_2)$ is the fBm covariance function given by

$$\psi_k(\tilde{t}_1,\tilde{t}_2) = \frac{\sigma^2(k)}{2}\left(\tilde{t}_1^{2H(k)}+\tilde{t}_2^{2H(k)}-|\tilde{t}_1-\tilde{t}_2|^{2H(k)}\right). \tag{21}$$

The fBm traffic parameters $\{\lambda(k),\sigma(k),H(k)\}$ are referred to as hyper-parameters in the GPR framework [20], [31]. Let $\boldsymbol{t}_k = [0,1,\cdots,(\frac{m_0T_M}{T_S}-1)]$ be training inputs and $\boldsymbol{A}_k = [\tilde{A}_k(0),\tilde{A}_k(1),\cdots,\tilde{A}_k(\frac{m_0T_M}{T_S}-1)]$ be training outputs. Then, we have the following joint Gaussian distribution

$$\boldsymbol{A}_k \sim \mathcal{N}(\lambda(k)\boldsymbol{t}_k, \Psi_k) \tag{22}$$

where Ψ_k is a $\frac{m_0T_M}{T_S}$-by-$\frac{m_0T_M}{T_S}$ covariance matrix, with $\Psi_k(i,j) = \psi_k(i,j)$. The GPR model is trained, i.e., the hyper-parameters are learned, by maximizing the following log-marginal likelihood function with a gradient optimizer

$$\begin{aligned}\log\Pr(\boldsymbol{A}_k|\boldsymbol{t}_k;\{\lambda(k),\sigma(k),H(k)\}) = -\frac{1}{2}\Big[&(\boldsymbol{A}_k-\lambda(k)\boldsymbol{t}_k)^T\\&\Psi_k^{-1}(\boldsymbol{A}_k-\lambda(k)\boldsymbol{t}_k)+\log|\Psi_k|+\frac{m_0T_M}{T_S}\log 2\pi\Big].\end{aligned} \tag{23}$$

For traffic parameter learning, it has $\mathcal{O}(\frac{m_0T_M}{T_S}^3)$ time complexity and $\mathcal{O}(\frac{m_0T_M}{T_S}^2)$ space complexity due to the inversion of a covariance matrix in (23). Such a complexity is feasible on a desktop computer for dataset sizes up to a few thousands [32]. There are sparse approximation algorithms to reduce the complexity of Gaussian process regression [32].

To evaluate the learning accuracy, one-step-ahead predictions for t_0 subsequent small time intervals are performed using the trained GPR model. The one-step-ahead prediction at time $\tilde{t}$ ($\frac{m_0T_M}{T_S}-1 \le \tilde{t} \le \frac{m_0T_M}{T_S}+t_0-2$) is to predict $\tilde{A}_k(\tilde{t}^*)$, given $\tilde{t}^* = \tilde{t}+1$ and a set of observed data $\mathcal{D} = (\boldsymbol{t},\boldsymbol{A})$ with $\boldsymbol{t} = [0,1,\cdots,\tilde{t}]$ and $\boldsymbol{A} = [\tilde{A}_k(0),\tilde{A}_k(1),\cdots,\tilde{A}_k(\tilde{t})]$. The GPR gives a Gaussian posterior distribution of $\tilde{A}_k(\tilde{t}^*)$ conditioned on $\tilde{t}^*$ and $\mathcal{D}$, as

$$\Pr(\tilde{A}_k(\tilde{t}^*)|\tilde{t}^*,\mathcal{D}) \sim \mathcal{N}(\mu_{\mathcal{GP};k}(\tilde{t}^*), \sigma^2_{\mathcal{GP};k}(\tilde{t}^*)) \tag{24}$$

with

$$\begin{cases}\mu_{\mathcal{GP};k}(\tilde{t}^*) = \boldsymbol{\psi}_k(\boldsymbol{t},\tilde{t}^*)^T(\Phi)^{-1}\boldsymbol{A}\\ \sigma^2_{\mathcal{GP};k}(\tilde{t}^*) = \psi_k(\tilde{t}^*,\tilde{t}^*)-\boldsymbol{\psi}_k(\boldsymbol{t},\tilde{t}^*)^T(\Phi)^{-1}\boldsymbol{\psi}_k(\boldsymbol{t},\tilde{t}^*).\end{cases} \tag{25}$$

In (25), $\boldsymbol{\psi}_k(\boldsymbol{t},\tilde{t}^*)$ is a $(\tilde{t}+1)$-by-1 vector with the i-th component equal to $\psi_k(i,\tilde{t}^*)$, and Φ is a $(\tilde{t}+1)$-by-$(\tilde{t}+1)$ covariance matrix with $\Phi(i,j) = \psi_k(i,j)$. The mean, $\mu_{\mathcal{GP};k}(\tilde{t}^*)$, is taken as a point estimate for the prediction output, and the

[1]The value of m_0 should be smaller than the minimum value of the most probable run lengths at any detected change points.

variance, $\sigma^2_{\mathcal{GP};k}(\tilde{t}^*)$, provides an uncertainty measure for the point estimate. With the predictive distribution for $\tilde{A}_k(\tilde{t}^*)$, the traffic sample in time interval $\tilde{t}^*$, i.e., $\tilde{x}_S(\tilde{t}^*)$, is predicted as

$$\hat{\tilde{x}}_S(\tilde{t}^*) = \mu_{\mathcal{GP};k}(\tilde{t}^*) - \tilde{A}_k(\tilde{t}^* - 1). \tag{26}$$

The prediction error of the t_0 traffic samples in the k-th stationary traffic segment, ϱ_k, is defined as the normalized root-mean-squared deviation between the t_0 predicted traffic samples and the corresponding ground truth, given by

$$\varrho_k = \frac{\sqrt{\sum_{\tilde{t}^*=\frac{m_0 T_M}{T_S}}^{\frac{m_0 T_M}{T_S}+t_0-1} \left(\hat{\tilde{x}}_S(\tilde{t}^*) - \tilde{x}_S(\tilde{t}^*)\right)^2}}{(x_S^{max} - x_S^{min})\sqrt{t_0}}. \tag{27}$$

The normalization constant is the scale of small-timescale traffic samples, i.e., $(x_S^{max} - x_S^{min})$. A smaller ϱ_k value indicates a higher learning accuracy for traffic parameters.

C. Resource Demand Prediction

With the learned fBm traffic parameters, the resource demand of the k-th stationary traffic segment can be predicted. Consider an fBm traffic input with parameters $\{\lambda, \sigma, H\}$ to an infinite buffer, with a constant service rate of R packets per small time interval. The buffer overflow probability, i.e., the probability that queue length q is beyond a threshold q_B, is approximately given by [21], [33]

$$\Pr(q > q_B) \simeq \exp\left(-\inf_{t\geq 0} \frac{[q_B + (R-\lambda)t]^2}{2\sigma^2 t^{2H}}\right) \tag{28}$$

which has been shown accurate even for a small value of q_B by simulation studies. Correspondingly, the delay violation probability can be approximated by

$$\Pr(d_S > D_S) \simeq \exp\left(-\inf_{t\geq 0} \frac{[RD_S + (R-\lambda)t]^2}{2\sigma^2 t^{2H}}\right) \tag{29}$$

where $d_S = \frac{d}{T_S}$ is the random VNF packet processing delay in number of small time intervals, and $D_S = \frac{D}{T_S}$ is the corresponding delay bound. To provide probabilistic QoS guarantee (i.e., $\Pr(d_S > D_S) \leq \varepsilon$) to the VNF with minimum resources, we should find

$$\min\left\{R \mid \forall t \geq 0, [RD_S + (R-\lambda)t]^2 \geq (-2\log\varepsilon)\sigma^2 t^{2H}\right\} \tag{30}$$

which leads to

$$R_{min} = \sup_{t\geq 0} \frac{\lambda t + \sqrt{-2\log\varepsilon}\,\sigma t^H}{t + D_S}. \tag{31}$$

The value of t achieving the supremum can be obtained by setting the derivative of R_{min} with respect to t to zero, i.e.,

$$\frac{\sqrt{-2\log\varepsilon}\,\sigma D_S H t^{H-1} + \sqrt{-2\log\varepsilon}\,\sigma(H-1)t^H + \lambda D_S}{(t+D_S)^2} = 0. \tag{32}$$

With the fBm resource provisioning model given in (31), the predicted resource demand (in packet/s) of the k-th stationary traffic segment, denoted by $R(k)$, is calculated from the learned fBm traffic parameters, i.e., $\{\lambda(k), \sigma(k), H(k)\}$, the QoS requirement, and the small time interval length, i.e., T_S.

IV. Deep Reinforcement Learning for Dynamic VNF Migration

The BOCPD algorithm locates the prior-unknown change points of the nonstationary traffic, which determines the boundaries between consecutive stationary traffic segments. They are also boundaries between consecutive decision epochs (with variable lengths) for VNF scaling and necessary VNF migrations. The length of decision epoch k is equal to $(\hat{\mathcal{C}}_M(k+1) - \hat{\mathcal{C}}_M(k))T_M$. Once change point $\hat{\mathcal{C}}_M(k)$ is detected, the resource demand $R(k)$ of the upcoming k-th stationary traffic segment is predicted, based on which a VNF migration decision is made.

A. VNF Migration Problem Formulation

For VNF migration, we jointly consider the migration cost and load balancing. Let $\{a_k^n, n \in \mathcal{N}_C\}$ be a binary variable set, with $a_k^n = 1$ if the VNF is placed at NFV node n during the k-th decision epoch, and $a_k^n = 0$ otherwise. Let a_k ($0 \leq a_k \leq |\mathcal{N}_C| - 1$) be an integer denoting the VNF location during decision epoch k, with $a_k = n$ if the VNF is placed at NFV node n. The relationship between $\{a_k^n\}$ and a_k is given by

$$a_k^n = \begin{cases} 1, & \text{if } a_k = n \\ 0, & \text{otherwise.} \end{cases} \tag{33}$$

Define the background resource loading factor of NFV node n during decision epoch k, denoted by $\eta_n^B(k)$, as the average ratio between the amount of processing resources (in packet/s) allocated to background traffic at NFV node n during decision epoch k and the processing resource capacity $R^{(n)}$ (in packet/s) of NFV node n. The resource loading factor of NFV node n during decision epoch k, denoted by $\eta_n(k)$, is dependent on both $\eta_n^B(k)$ and VNF placement, given by

$$\eta_n(k) = \eta_n^B(k) + \frac{a_k^n R(k)}{R^{(n)}}. \tag{34}$$

The cost for imbalanced loading during decision epoch k is defined as the maximum resource loading factor among all NFV nodes in $\mathcal{N}_C$, given by

$$c_k^{(1)} = \max_{n\in\mathcal{N}_C} \eta_n(k), \tag{35}$$

since minimizing $c_k^{(1)}$ achieves load balancing among all the candidate NFV nodes. Assume that each VNF migration incurs the same migration cost. Then, we can use the total number of migrations to denote the total migration cost, given by

$$c_k^{(2)} = \begin{cases} \sum_{n\in\mathcal{N}_C} \sum_{n'\in\mathcal{N}_C\setminus n} a_{k-1}^n a_k^{n'}, & \text{if } k > 0 \\ 0, & \text{if } k = 0 \end{cases} \tag{36}$$

where a_{k-1}^n is a known value at decision epoch k (> 0). In the single VNF scenario, we have $c_k^{(2)} = 1$ for $k > 0$ if the VNF placement changes from decision epoch $(k-1)$ to decision epoch k, and $c_k^{(2)} = 0$ otherwise. The total cost is a weighted combination of the two costs, given by

$$c_k = \omega c_k^{(1)} + (1-\omega) c_k^{(2)} \tag{37}$$

where ω is a weighting factor in $(0, 1)$. In stepwise optimization for cost minimization in the short term, total cost c_k is

minimized at each decision epoch k, subject to processing resource capacity constraints at the NFV nodes, i.e., the resource loading factors of all NFV nodes should not exceed 1. For cost minimization in the long run, the VNF migration problem can be formulated as an MDP, with the state, action, and reward defined as follows:

- *State* − At decision epoch k, the state is composed of four parts: the k-th change point, the predicted resource demand of the k-th stationary traffic segment, the background resource loading factors of all candidate NFV nodes during decision epoch k, and the previous VNF placement a_{k-1}. Thus, the state for decision epoch k is represented as $\boldsymbol{s}_k = [\hat{\mathcal{C}}(k), R(k), \{\eta_n^B(k)\}, a_{k-1}]$. Here, $\hat{\mathcal{C}}(k)$ is a real number representing the k-th estimated change point in hour, given by

$$\hat{\mathcal{C}}(k) = \frac{\hat{\mathcal{C}}_M(k) T_M}{3600} \mod 24 \tag{38}$$

  where the modulo operation limits $\hat{\mathcal{C}}(k)$ in $[0, 24)$;
- *Action* − The action at decision epoch k is the new VNF placement, i.e., a_k. We use a_k instead of $\{a_k^n\}$ as the action to limit the dimensionality of action space;
- *Reward* − In an unconstrained MDP, the violation of resource capacity constraints is penalized by an extra term in reward. Hence, the reward for decision epoch k is

$$r_k = -\left(c_k + c^{(v)} v_k\right) \tag{39}$$

  where c_k is the total cost for VNF migration at decision epoch k as given in (37), v_k is a binary flag indicating whether there is penalty due to resource overloading, and $c^{(v)}$ is a constant representing the level of penalty. Assume that resource overloading is only due to improper VNF placement, i.e., the background traffic does not overload the NFV nodes ($\eta_n^B(k) < 1$). Then, the penalty flag is defined as where

$$v_k = \begin{cases} 1, & c_k^{(1)} > \eta_U \\ 0, & \text{otherwise} \end{cases} \tag{40}$$

  where η_U ($0 < \eta_U \leq 1$) is an upper limit for the maximum resource loading factor without penalty. In practice, we select η_U as a number close to but smaller than 1, e.g., $\eta_U = 0.95$, to penalize loading factors close to 1, with the consideration that the penalty cannot be completely avoided in a learning-based solution due to exploration. Moreover, if the predicted resource demand is very large, it is possible that there is no feasible VNF placement without resource overloading. A potential solution is to throttle the traffic when resource overloading is foreseen to happen. Here, we assume that the VNF placement without resource overloading is always feasible and do not consider traffic throttling.

B. Penalty-Aware Deep Q-Learning Algorithm

We solve the MDP by an RL approach, when transition probabilities among states are unavailable. Consider an episodic task, in which an RL agent interacts with the VNF migration environment in a sequence of episodes, with a finite number of learning steps in each episode. Here, a learning step corresponds to a decision epoch, and an episode corresponds to a time duration such as one day, one week, or one month. At the beginning of an episode, the VNF placement is initialized at NFV node n_0. Within an episode, an agent observes state $\boldsymbol{s}_k$ and takes action a_k at the beginning of decision epoch k. At the end of decision epoch k, the agent receives reward r_k, and sees new state $\boldsymbol{s}_{k+1}$. The goal is to find a policy, $\pi(\boldsymbol{s})$, mapping a state to an action, to maximize the expected cumulative (episodic) discounted reward $\mathbb{E}(\sum_{k=0}^{K-1} \gamma^k r_k)$, where K is the number of variable-length decision epochs in an episode, and $\gamma \in (0, 1]$ is the discount factor. In Q-learning [24], a state-action value function is defined as

$$Q(\boldsymbol{s}_k, a_k) = \mathbb{E}\left[\sum_{k'=k}^{K-1} \gamma^{k'-k} r_{k'} | \boldsymbol{s}_k, a_k\right] \tag{41}$$

The Q-learning is an off-policy algorithm adopting the ϵ-greedy policy

$$\pi(\boldsymbol{s}_k) = \begin{cases} \underset{a}{\text{argmax}}\, Q(\boldsymbol{s}_k, a), & \text{with probability } (1-\epsilon) \\ \text{random action}, & \text{with probability } \epsilon \end{cases} \tag{42}$$

where ϵ is the exploration probability. We use a gradually decreasing ϵ from 1 to a minimum value ϵ_0, with a step size Δ_ϵ, to transit smoothly from exploration to exploitation.

The formulated MDP is featured by a high-dimensional combinational state space and a low-dimensional discrete action space. To tackle the curse of dimensionality, deep Q-learning adopts two deep Q networks (DQNs) with the same neural network structure as Q function approximators, i.e.,

Algorithm 1: Penalty-aware deep Q-learning

Initialize: Evaluation and target DQNs with random weights, set learning parameters as listed in Table II.
for *each episode* **do**
  Initialize VNF placement at NFV node n_0.
  for *each learning step (decision epoch)* **do**
    Observe current state $\boldsymbol{s}_k$, select an action a_k according to the ϵ-greedy policy in (42).
    Execute action a_k, collect reward r_k and penalty flag v_k, and see the next state $\boldsymbol{s}_{k+1}$.
    Store transition $(\boldsymbol{s}_k, a_k, r_k, v_k, \boldsymbol{s}_{k+1})$ into replay memory, with initial priority $p_k = \max_{j<k} p_j$.
    for J *iterations* **do**
      Sample a transition $(\boldsymbol{s}_j, a_j, r_j, v_j, \boldsymbol{s}_{j+1})$ with probability $\Pr(j)$.
      Compute importance-sampling weight w_j.
      Compute target value y_j and TD error δ_j.
      Update transition priority p_j.
      Perform a gradient descent, i.e., $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \xi\,(w_j \delta_j)\,\nabla_{\boldsymbol{\theta}} Q(\boldsymbol{s}_j, a_j)$.
    Decrease ϵ by a step Δ_ϵ, if $\epsilon > \epsilon_0$.
    Every $K_{\boldsymbol{\theta}}$ steps, set $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}$.
Output: Trained evaluation and target DQNs.

evaluation DQN (Q) with weights $\boldsymbol{\theta}$ and target DQN ($\hat{Q}$) with slowly updated weights $\hat{\boldsymbol{\theta}}$ [34]. Every $K_{\boldsymbol{\theta}}$ learning steps, $\hat{\boldsymbol{\theta}}$ is replaced by $\boldsymbol{\theta}$. The policy in (42) is based on evaluation DQN, which is trained by minimizing a loss function

$$\mathbb{L}(\boldsymbol{\theta}) = \mathbb{E}\left[(y_k - Q(\boldsymbol{s}_k, a_k; \boldsymbol{\theta}))^2\right] \tag{43}$$

through gradient descent on $\boldsymbol{\theta}$, where y_k is a target value estimated by target DQN, given by

$$y_k = r_k + \gamma \max_a \hat{Q}(\boldsymbol{s}_{k+1}, a; \hat{\boldsymbol{\theta}}). \tag{44}$$

If an episode terminates at the k-th learning step, y_k is set as r_k. A gradient descent on $\boldsymbol{\theta}$ is performed by

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \frac{1}{2}\xi \nabla_{\boldsymbol{\theta}} \mathbb{L}(\boldsymbol{\theta}) = \boldsymbol{\theta} + \xi \delta_k \nabla_{\boldsymbol{\theta}} Q(\boldsymbol{s}_k, a_k) \tag{45}$$

where ξ is the learning rate, and $\delta_k = y_k - Q(\boldsymbol{s}_k, a_k; \boldsymbol{\theta})$ is the temporal-difference (TD) error.

Experience replay is introduced in deep Q-learning for stable convergence [34]. At each learning step, $\boldsymbol{\theta}$ is updated with a mini-batch (size equal J) of experiences $(\boldsymbol{s}_j, a_j, r_j, \boldsymbol{s}_{j+1})$ uniformly sampled from a replay memory. Experience replay breaks the temporal correlation among experiences, and liberates RL agents from learning with transitions in the same order as they appear. Prioritized experience replay achieves more learning efficiency through further liberating RL agents from considering transitions in the same frequency as they appear [35], [36]. It assigns a priority, p_j, for transition j sampled from the replay memory, which is the magnitude of TD error δ_j plus a very small value o. The sampling probability of transition j is

$$\Pr(j) = \frac{p_j^{\varsigma_0}}{\sum_{j=1}^{M} p_j^{\varsigma_0}} \tag{46}$$

where M is the size of replay memory and ς_0 determines the level of prioritization.

In the VNF migration problem, it is desired that the deep Q-learning algorithm converges to a solution without resource overloading penalty in the whole episode. However, such experiences are rare at the early learning stage with a lot of exploration, especially if an episode contains a large number of transitions. To learn more from such rare desired experiences, we extend the prioritized experience replay technique to consider penalty-awareness. Among the original prioritized transitions with high absolute TD errors, we place more priority on those transitions with zero penalty, given by

$$p_j = \varphi|\delta_j| + (1-\varphi)(1-v_j) + o \tag{47}$$

where $\varphi \in [0, 1]$ is a parameter controlling the relative importance of TD error and penalty avoidance. In practice, we select φ close to 1, e.g., 0.99, to incorporate penalty-awareness without significant degradation on convergence speed. Correspondingly, a five-tuple transition $(\boldsymbol{s}_j, a_j, r_j, v_j, \boldsymbol{s}_{j+1})$ instead of the original four-tuple transition, $(\boldsymbol{s}_j, a_j, r_j, \boldsymbol{s}_{j+1})$, is stored in the replay memory at every learning step. A deep Q-learning algorithm with penalty-aware prioritized experience replay is presented in Algorithm 1. The prioritization leads to a loss of diversity, which can be corrected with an importance-sampling weight w_j, given by [36]

$$w_j = (B \cdot \Pr(j))^{-\varsigma_1} / \max_{j'}(w_{j'}) \tag{48}$$

where ς_1 controls the level of compensation. The TD error δ_j is replaced by a weighted TD error $w_j\delta_j$ in a gradient descent step with transition j, as given in line 13 of Algorithm 1.

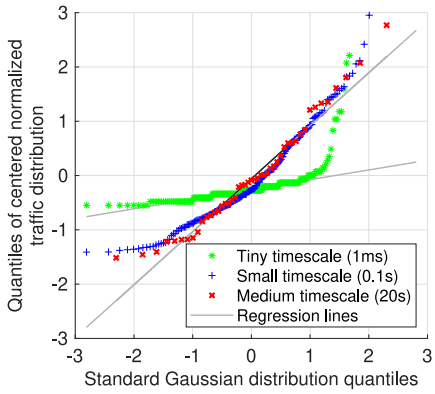


Fig. 4: Quantile-quantile (QQ) plots for different timescales.

TABLE I: Traffic sets with different randomness levels

Traffic set	Change points	Resource demands
1	Detected	Predicted
2	Detected $\pm$ $[0, 0.1]$ hour	Predicted
3	Detected	Predicted $\pm$ $[0\%, 5\%]$
4	Detected $\pm$ $[0, 0.1]$ hour	Predicted $\pm$ $[0\%, 5\%]$

V. Performance Evaluation

We use a real-world backbone traffic trace from the MAWI working group of the WIDE project for performance evaluation, which provides packet-level information collected from Internet backbone links [15]. We select two most recent 48-hour-long traces collected from the transit link of WIDE backbone connecting the upstream ISP. The specific days are 2018/05/09, 2018/05/10, 2019/04/09, and 2019/04/10. We extract the *http* traffic from port 443 as an aggregate service flow, and select $T_S = 0.1$ s and $T_M = 20$ s as the lengths of small and medium time intervals, respectively. The quantile-quantile (QQ) plots for the distribution of centered normalized number of packet arrivals (with mean equal to 0 and standard deviation equal to 1) in different timescales (within the same stationary traffic segment) versus a standard Gaussian distribution are presented in Fig. 4. It shows that the traffic distributions in both small and medium timescales are approximately Gaussian with heavy tails. The traffic distribution in a tiny timescale (1 ms) is more bursty and completely not Gaussian due to insufficient aggregation of packet arrivals in each tiny time interval. The two thresholds in change point detection, i.e., Δ_L and Δ_d, are set as 10 and 5%, respectively. We select $m_0 = 4$ to have 800 small-timescale traffic samples in each stationary traffic segment for traffic parameter learning, which gives high computation efficiency while achieving a good accuracy. For each daily traffic trace, change points are detected, and resource demands are predicted for the

TABLE II: List of parameters in deep Q-learning

ξ	Learning rate	10^{-6}
γ	Discount factor	0.9
ϵ_0	Minimum exploration probability	0.01
Δ_ϵ	Step size of exploration probability	5×10^{-6}
$K_{\boldsymbol{\theta}}$	Number of steps to replace $\hat{\boldsymbol{\theta}}$ by $\boldsymbol{\theta}$	200
M	Memory size	2000
J	Batch size	200
φ	Weight in the priority	0.99

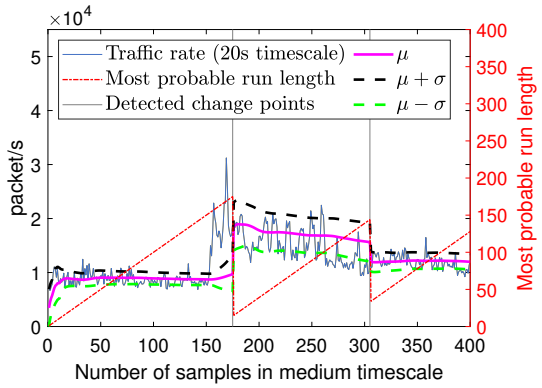


Fig. 5: Results of change point detection for a nonstationary traffic segment.

identified stationary traffic segments. There are 25, 20, 26, and 24 detected change points in the four daily traffic traces, respectively. The accuracy of traffic parameter learning is evaluated with $t_0 = 1000$ small-timescale traffic samples.

For dynamic VNF migration, we consider one VNF initially placed at NFV node n_0, with another five candidate NFV nodes located in its neighborhood. All NFV nodes have the same processing capacity $R^{(n)} = 125000$ packet/s. The background resource loading factor of each NFV node varies between 10% and 90%, in different patterns with peaks at different time in a day. We set weighting factor $\omega = 0.6$ and penalty level $c^{(v)} = 5$. For deep Q-learning, an episode corresponds to one week, to have sufficient learning steps (decision epochs) with periodic dynamics in both change points and resource demands within an episode. The weekly traffic in one episode is artificially composited by daily traffic of the four days in random order, with different randomness levels in both change points and resource demands, as described in Table I. The randomness level (in hour) around change points follows a uniform distribution in $[0, 0.1]$, and the randomness level around resource demands follows a uniform distribution in $[0\%, 5\%]$. We use a DQN structure with one hidden layer of 20 neurons and Relu as the activation function, with important learning parameters summarized in Table II.

Fig. 5 shows results of change point detection for a nonstationary traffic segment in 8000 s, corresponding to 400 medium time intervals. A zigzag trend is observed for the most probable run length. The detected change points are indicated by gray vertical lines. Online estimation of mean and standard deviation in a student-t distribution corresponding to the most probable run length at each time step (in medium timescale) is a byproduct of change point detection. It is observed that both statistics are stable between the detected change points.

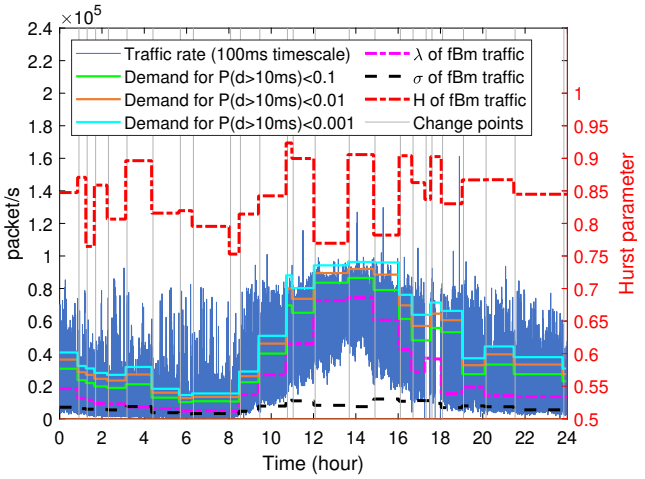


Fig. 6: Traffic parameter learning and resource demand prediction for daily traffic.

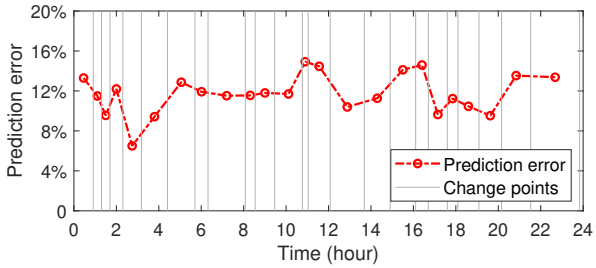


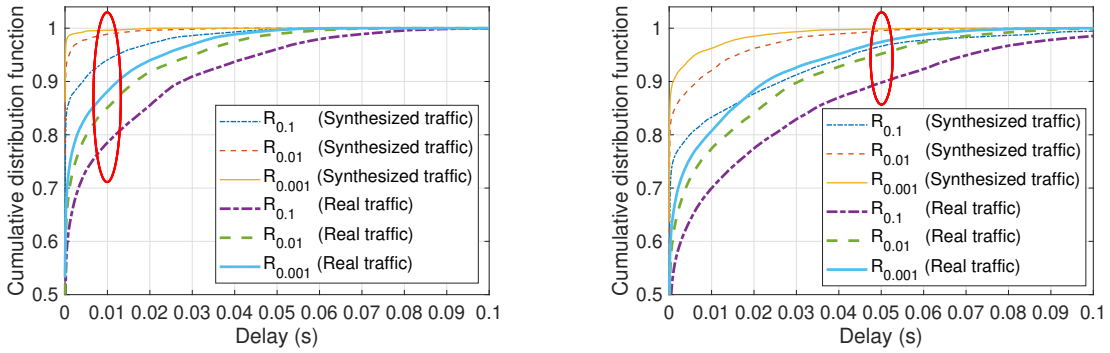
Fig. 7: Evaluation of traffic parameter learning accuracy for daily traffic.

We see that both conditions in (16) and (17) are satisfied for the two detected change points, i.e., the most probable run length drops by more than Δ_L, and the change in mean plus standard deviation is sufficiently large. We also observe that the change points are detected after the occurrence of statistical changes, which verifies the effectiveness of the *look-back* traffic parameter learning.

To evaluate the accuracy of the proposed traffic parameter learning method, we simulate six groups of fBm traffic traces in discrete time with different traffic parameters and resource demands, as given in Table III. The simulated fBm traffic is generated following a wavelet-based algorithm [37]. The length of a time unit is not specified. The resource demand to satisfy the QoS requirement $\Pr(d > 0.01 \text{ time units}) \leq 0.01$ is denoted by $R_{0.01}$. For each group of fBm traffic, 200 sample paths are generated, with 1000 traffic samples in each sample path. The traffic parameters of each sample path are estimated by the first 800 traffic samples using both the proposed learning method and a classical benchmark method. In the benchmark method, the mean and variance are estimated by the sample mean and variance, and the Hurst parameter is estimated separately using a wavelet-based approach [37]. In the proposed learning method, the three parameters are learned together, reaching a compromise among them to maximize the log-marginal likelihood function given in (23). The results of both methods are given in Table III. It is observed that the mean and Hurst parameter given by both methods are close to the simulated parameters, but the standard deviation is not as accurate. However, the level of underestimation given by

TABLE III: Traffic parameters and resource demands of simulated fBm traffic

Group	Real				Estimated (Benchmark)				Estimated (Proposed)				
	λ	σ	H	$R_{0.01}$	$\bar{\lambda}$	$\bar{\sigma}$	$\bar{H}$	$R_{0.01}$	$\bar{\lambda}$	$\bar{\sigma}$	$\bar{H}$	$R_{0.01}$	ϱ
1	800	200	0.7	1429.6	797.9	216.3	0.698	1490.9	795.4	218.8	0.693	1501.4	0.1507
2	800	200	0.8	1403.4	804.4	191.4	0.796	1378.2	798.6	198.6	0.793	1397.8	0.1373
3	800	200	0.9	1422.5	801.3	182.4	0.894	1363.1	798.8	210.6	0.893	1453.2	0.1246
4	900	300	0.7	1895.7	898.5	325.6	0.697	1997.1	895.8	327.3	0.693	2008.3	0.1489
5	900	300	0.8	1836.4	896.9	287.5	0.796	1791.4	895.3	296.4	0.792	1822.0	0.1380
6	900	300	0.9	1848.7	899.6	271.8	0.895	1752.1	898.9	310.8	0.891	1879.0	0.1218



(a) $D = 10$ ms (b) $D = 50$ ms

Fig. 8: Distribution of VNF packet processing delay for both the synthesized traffic and the real traffic.

the learning method is much lower than that given by the benchmark method. The accuracy of traffic parameter learning is also evaluated by the average prediction error, ϱ, for the last 200 traffic samples in each sample path, as given in Table III.

Fig. 6 shows results of the proposed change-point-driven traffic parameter learning and resource demand prediction scheme for a real-world daily traffic trace. The detected change points identify different stationary traffic segments. For each stationary traffic segment, the three learned fBm traffic parameters $\{\lambda(k), \sigma(k), H(k)\}$ are plotted. We observe that the Hurst parameter is within $[0.5, 1)$, indicating self-similarity and LRD of the traffic. The predicted resource demands for the identified stationary traffic segments are given, for QoS requirements $\Pr(d > 10ms) \leq \varepsilon$ with ε = 0.1, 0.01, and 0.001. As expected, the resource demand increases when ε decreases. The average prediction error evaluated by $t_0 = 1000$ traffic samples in each identified stationary traffic segment is plotted in Fig. 7. It can be seen that the average prediction errors for the real-world traffic trace is comparable to that of the simulated fBm traffic traces given in Table III.

To evaluate QoS performance of the proposed resource demand prediction scheme, we conduct packet-level simulations using the python Simpy package, to gather sufficient packet delay information for a smooth characterization of the VNF packet processing delay distribution, with a $60s$-long stationary traffic segment from the real-world traffic trace as the VNF traffic input. Different amount of resources are allocated to the VNF according to the predicted resource demands for different QoS requirements. For simplicity, we use R_ε to represent the predicted resource demand for a probabilistic delay guarantee, i.e., $\Pr(d > D) \leq \varepsilon$. Since traffic parameter learning is performed in 0.1 s timescale, traffic burstiness in time granularities smaller than 0.1 s cannot be captured. Hence, we use both the real packet arrival trace and a less-bursty synthesized packet arrival trace for QoS evaluation. In the synthesized packet arrival trace, the numbers of packet arrivals in 0.1 s timescale are the same as the real packet arrival trace, but the packet inter-arrival time within each 0.1 s time interval follows an exponential distribution. Fig. 8 shows the distribution of VNF packet processing delay for both traffic traces. Two groups of delay requirements with different delay bounds, i.e., $D = 10$ ms and $D = 50$ ms, are used for QoS evaluation. In each group, ε is set as 0.1, 0.01, and 0.001. For the same QoS requirement, the amount of resources allocated for both traffic are the same. However, the delay performance of the synthesized traffic is better than that of the real traffic, due to less traffic burstiness in time granularities smaller than 0.1 s. For the synthesized traffic, the delay violation probability is within the corresponding upper limits. For the real traffic, the delay violation probability occasionally exceeds the required upper limit, especially for the stringent QoS requirements such as $\Pr(d > 10ms) \leq 0.001$, due to traffic burstiness in time granularities below 0.1 s.

In addition, we compare the QoS performance between the proposed fBm model based resource demand prediction scheme and a benchmark M/M/1 model based counter-

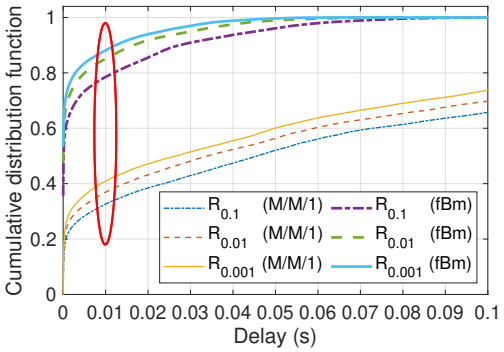
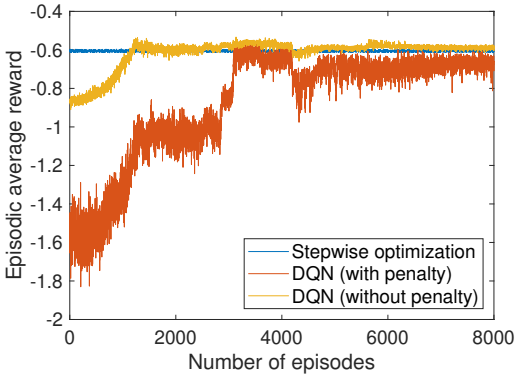


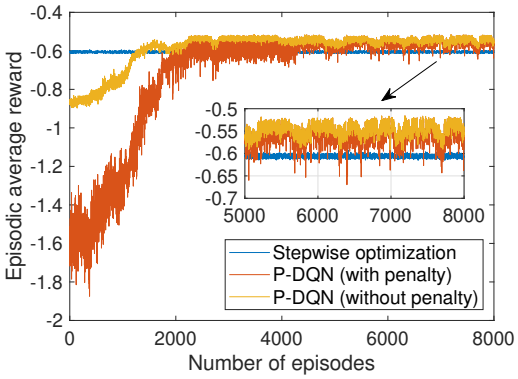
Fig. 9: QoS performance comparison between the fBm model and M/M/1 model based resource demand prediction schemes.

part [22]. Both methods use the learned traffic parameters for resource demand prediction. In the proposed scheme, the resource demand is predicted from all three learned traffic parameters, i.e., $\{\lambda, \sigma, H\}$, based on the fBm resource provisioning model given by (31). In the benchmark scheme, the resource demand is predicted from the first learned traffic parameter, i.e., λ, based on an M/M/1 resource provisioning model. For an M/M/1 queue with arrival rate λ (in packet/s) and service rate R (in packet/s), the delay violation probability is $\Pr(d > D) = e^{-(R-\lambda)D}$ [38]. Hence, the minimum amount of resources (in packet/s) to guarantee the QoS requirement $\Pr(d > D) \leq \varepsilon$ is $R_{min} = \lambda - \frac{\log \varepsilon}{D}$. Fig. 9 shows the VNF packet delay distribution with the real packet arrivals and with different amount of resources allocated to the VNF, based on predicted resource demands given by the two models. A gap is observed between the delay performance of the two models. The proposed model, with the ability to capture the bursty nature of traffic, gives a better estimation of resource demands.

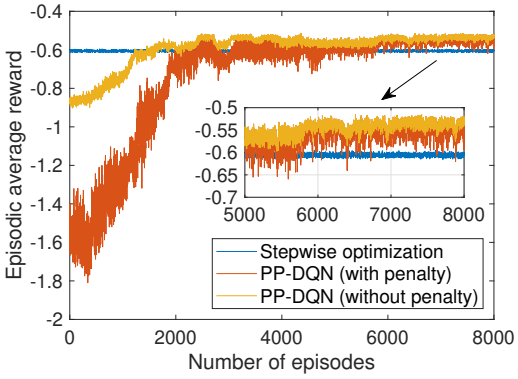
The performance of the proposed deep Q-learning algorithm with penalty-aware prioritized experience replay (PP-DQN) is compared with two benchmark algorithms, i.e., deep Q learning with uniformly sampled experience replay (DQN), and deep Q-learning with prioritized experience relay (P-DQN). All three deep Q-learning algorithms are compared with a common benchmark, i.e., stepwise optimization. The comparison is performed using traffic set 1 in Table I. Fig. 10 shows the evolution of episodic average reward with respect to the number of episodes during the learning process, using the three deep Q-learning algorithms. Both the full reward including penalty and the partial reward without penalty are plotted, with a gap indicating the penalty. It is observed that DQN converges to a poor solution which is worse than the stepwise optimization benchmark in terms of episodic average reward. The penalty is high, inferring that the DQN does not learn a solution to minimize the resource overloading penalty in the long run. Both the P-DQN and PP-DQN algorithms take advantages of the prioritized experience replay for convergence to solutions that outperform the stepwise optimization benchmark in most time after convergence. It demonstrates that both P-DQN and PP-DQN after convergence can capture the weekly traffic patterns (in both change points and resource demands) and background resource loading patterns at the candidate NFV nodes, and make intelligent VNF migration



(a) DQN



(b) P-DQN



(c) PP-DQN

Fig. 10: Episodic average reward versus the episode number for the three deep Q-learning algorithms.

decisions accordingly. In contrast, when a VNF migration is required, the stepwise optimization benchmark favors VNF migration to a lightly loaded NFV node in the current decision epoch, which can be heavily loaded in the following decision epochs. As illustrated in Fig. 10 (b) and (c), the proposed PP-DQN achieves slightly more gain in terms of penalty suppression compared with P-DQN. The episodic average rewards (with penalty) of P-DQN and PP-DQN after convergence are -0.5502 and -0.5408 respectively. Fig. 11 shows that the training loss of PP-DQN as defined in (43) converges faster to a smaller value. We also examined the learning curve of PP-DQN with $\varphi = 0.5$, which gives an average training loss of 0.0155 after convergence, demonstrating the benefit

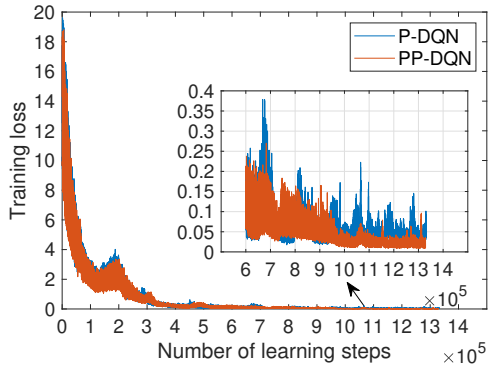


Fig. 11: Training loss of the evaluation Q networks.

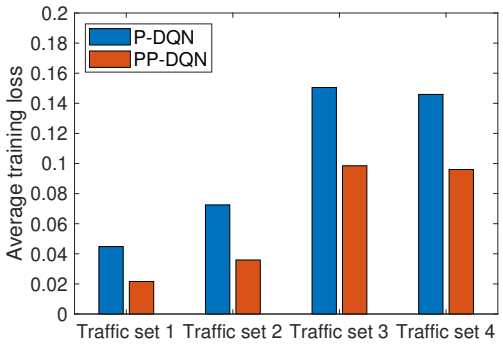


Fig. 12: Average training loss after convergence.

of reducing weight φ on further loss reduction. However, the benefit on additional reward improvement is not significant. To evaluate the impact of traffic randomness, we compare the average training loss of both P-DQN and PP-DQN after convergence in Fig. 12, using four traffic sets with different randomness levels in change points and resource demands, as in Table I. With more randomness especially in resource demand, the average training loss of both P-DQN and PP-DQN increases. However, the PP-DQN outperforms P-DQN for all the traffic sets.

VI. Conclusion

In this paper, we study a dynamic VNF scaling problem in an SDN/NFV-enabled 5G network slicing scenario, to guarantee the required delay performance in the presence of real-world time-varying traffic with nonstationary characteristics. A change-point-driven traffic parameter learning and resource demand prediction scheme is proposed, based on which dynamic VNF migration decisions are made at variable-length decision epochs using a newly proposed penalty-aware deep Q-learning algorithm. An fBm traffic model is employed for each identified stationary traffic segment, based on properties of Gaussianity and self-similarity of the real-world traffic, which are verified by the QQ plots and the scale of the learned Hurst parameters, respectively. The proposed traffic parameter learning method achieves a better accuracy than a benchmark method for the simulated fBm traffic, benefiting from a compromise among the three traffic parameters. The traffic parameter learning accuracy is also demonstrated by the average prediction error with the trained GPR models. For the proposed resource demand prediction scheme, packet-level simulations show occasional QoS violation for a real-world packet arrival trace especially for the stringent QoS requirements, and QoS satisfaction for a synthesized packet arrival trace with less traffic burstiness. To provide better QoS provisioning, a potential approach is to explicitly incorporate the measured QoS information as a feedback in the state and make decisions accordingly to achieve QoS satisfaction in the long run, which remains as our future work. The proposed penalty-aware deep Q-learning algorithm achieves performance gains in terms of both training loss reduction and episodic average reward maximization.

Appendix: Bayesian Online Change Point Detection

Under the assumption of *a priori* geometric inter-arrivals of change points, the conditional prior probability distribution over the run length, denoted by $\Pr(L_m|L_{m-1})$, is given by

$$\Pr(L_m|L_{m-1}) = \begin{cases} 1-(1/\chi), & \text{if } L_m = L_{m-1}+1 \\ 1/\chi, & \text{if } L_m = 0 \\ 0, & \text{otherwise} \end{cases} \quad \text{(A1)}$$

where χ is the prior average run length. We set $\chi = \frac{3600}{T_M}$ in simulation, which is equivalent to one hour in time duration. The conditional prior has nonzero mass at only two outcomes, i.e., the run length either grows by 1, or resets to 0. Accordingly, there are two branches for $\Pr(L_m, \boldsymbol{x}_m)$, given by (A2). The predictive model $\Pr(x_m|L_{m-1}, \boldsymbol{x}_{m-1}^{(L)})$ evaluates the probability that x_m belongs to the same run as $\boldsymbol{x}_{m-1}^{(L)}$, given run length L_{m-1}. Under the i.i.d Gaussian assumption with unknown mean μ and variance ν^2, a Normal-Inverse-Gamma (NIG) prior is placed on μ and ν^2, given by

$$\Pr(\mu, \nu^2) \sim \mathcal{N}\left(\mu|\mu_0, \frac{\nu^2}{\kappa_0}\right) \mathcal{IG}\left(\nu^2|\alpha_0, \beta_0\right) \quad \text{(A3)}$$

where $\{\mu_0, \kappa_0, \alpha_0, \beta_0\}$ are prior parameters. Conjugate Bayesian analysis [39], [40] gives an NIG posterior on μ and ν^2 given $\boldsymbol{x}_{m-1}^{(L)}$, represented as

$$\Pr(\mu,\nu^2|\boldsymbol{x}_{m-1}^{(L)}) \sim \mathcal{N}\left(\mu|\mu_{m-1}^{(L)}, \frac{\nu^2}{\kappa_{m-1}^{(L)}}\right) \mathcal{IG}\left(\nu^2|\alpha_{m-1}^{(L)}, \beta_{m-1}^{(L)}\right) \quad \text{(A4)}$$

where $\{\mu_{m-1}^{(L)}, \kappa_{m-1}^{(L)}, \alpha_{m-1}^{(L)}, \beta_{m-1}^{(L)}\}$ are referred to as sufficient statistics corresponding to $\boldsymbol{x}_{m-1}^{(L)}$. Each possible value of run length L_{m-1} corresponds to a group of sufficient statistics. The posterior predictive distribution for x_m given L_{m-1} and $\boldsymbol{x}_{m-1}^{(L)}$, i.e., $\Pr(x_m|L_{m-1}, \boldsymbol{x}_{m-1}^{(L)})$, is described by a student-t distribution, represented as

$$\Pr(x_m|L_{m-1}, \boldsymbol{x}_{m-1}^{(L)}) \sim t_{2\alpha_{m-1}^{(L)}}\left(x_m|\mu_{m-1}^{(L)}, \frac{\beta_{m-1}^{(L)}(\kappa_{m-1}^{(L)}+1)}{\kappa_{m-1}^{(L)}\alpha_{m-1}^{(L)}}\right) \quad \text{(A5)}$$

where $\mu_{m-1}^{(L)}$ is the mean, $2\alpha_{m-1}^{(L)}$ is the degrees of freedom, and $\frac{\beta_{m-1}^{(L)}(\kappa_{m-1}^{(L)}+1)}{\kappa_{m-1}^{(L)}\alpha_{m-1}^{(L)}}$ is the scale. The standard deviation of the

$$\overbrace{\Pr(L_m, \boldsymbol{x}_m)}^{m\text{-th iteration}} = \begin{cases} \left(1-\frac{1}{\chi}\right) \Pr\left(x_m | L_{m-1}, \boldsymbol{x}_{m-1}^{(L)}\right) \Pr\left(L_{m-1}, \boldsymbol{x}_{m-1}\right), & \text{if } L_m = L_{m-1}+1 \\ \frac{1}{\chi} \sum_{L_{m-1}=0}^{m-1} \Pr\left(x_m | L_{m-1}, \boldsymbol{x}_{m-1}^{(L)}\right) \Pr\left(L_{m-1}, \boldsymbol{x}_{m-1}\right), & \text{if } L_m = 0. \end{cases} \tag{A2}$$

student-t distribution, denoted by $\nu_{m-1}^{(L)}$, is given by

$$\nu_{m-1}^{(L)} = \sqrt{\frac{\beta_{m-1}^{(L)}\left(\kappa_{m-1}^{(L)}+1\right)}{\kappa_{m-1}^{(L)}\left(\alpha_{m-1}^{(L)}-1\right)}}. \tag{A6}$$

After new observation x_m is available, sufficient statistics corresponding to $\boldsymbol{x}_m^{(L)}$ for $\forall L_m > 0$ are updated as

$$\mu_m^{(L)} = \frac{\kappa_{m-1}^{(L)}\mu_{m-1}^{(L)} + x_m}{\kappa_{m-1}^{(L)}+1} \tag{A7a}$$

$$\kappa_m^{(L)} = \kappa_{m-1}^{(L)} + 1 \tag{A7b}$$

$$\alpha_{m+1}^{(L)} = \alpha_{m-1}^{(L)} + \frac{1}{2} \tag{A7c}$$

$$\beta_m^{(L)} = \beta_{m-1}^{(L)} + \frac{\kappa_{m-1}^{(L)}\left(x_m - \mu_{m-1}^{(L)}\right)^2}{2\left(\kappa_{m-1}^{(L)}+1\right)}. \tag{A7d}$$

For $L_m = 0$, the sufficient statistics are updated from the prior parameters of the NIG distribution.

REFERENCES

[1] V.-G. Nguyen, A. Brunstrom, K.-J. Grinnemo, and J. Taheri, "SDN/NFV-based mobile packet core network architectures: A survey," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1567–1602, Apr. 2017.

[2] W. Zhuang, Q. Ye, F. Lyu, N. Cheng, and J. Ren, "SDN/NFV-empowered future IoV with enhanced communication, computing, and caching," *Proc. IEEE*, vol. 108, no. 2, pp. 274–291, Feb. 2020.

[3] H. Li, K. Ota, and M. Dong, "LS-SDV: Virtual network management in large-scale software-defined IoT," *IEEE J. Select. Areas Commun.*, vol. 37, no. 8, pp. 1783–1793, Aug. 2019.

[4] J. Chen, Q. Ye, W. Quan, S. Yan, P. T. Do, P. Yang, W. Zhuang, X. Shen, X. Li, and J. Rao, "SDATP: An SDN-based traffic-adaptive and service-oriented transmission protocol," *IEEE Trans. Cogn. Commun. Netw.*, vol. 6, no. 2, pp. 756–770, June 2020.

[5] O. Alhussein, P. T. Do, Q. Ye, J. Li, W. Shi, W. Zhuang, X. Shen, X. Li, and J. Rao, "A virtual network customization framework for multicast services in NFV-enabled core networks," *IEEE J. Select. Areas Commun.*, vol. 38, no. 6, pp. 1025–1039, June 2020.

[6] A. Gupta, B. Jaumard, M. Tornatore, and B. Mukherjee, "A scalable approach for service chain mapping with multiple SC instances in a wide-area network," *IEEE J. Select. Areas Commun.*, vol. 36, no. 3, pp. 529–541, Mar. 2018.

[7] M. Ghaznavi, A. Khan, N. Shahriar, K. Alsubhi, R. Ahmed, and R. Boutaba, "Elastic virtual network function placement," in *Proc. IEEE CloudNet'15*, Oct. 2015, pp. 255–260.

[8] K. Qu, W. Zhuang, Q. Ye, X. Shen, X. Li, and J. Rao, "Traffic engineering for service-oriented 5G networks with SDN-NFV integration," *IEEE Netw.*, vol. 34, no. 4, pp. 234–241, July/Aug. 2020.

[9] Q. Ye, W. Zhuang, X. Li, and J. Rao, "End-to-end delay modeling for embedded VNF chains in 5G core networks," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 692–704, Feb. 2019.

[10] Q. Ye, J. Li, K. Qu, W. Zhuang, X. Shen, and X. Li, "End-to-end quality of service in 5G networks – Examining the effectiveness of a network slicing framework," *IEEE Veh. Technol. Mag.*, vol. 13, no. 2, pp. 65–74, June 2018.

[11] V. Eramo, E. Miucci, M. Ammar, and F. G. Lavacca, "An approach for service function chain routing and virtual function network instance migration in network function virtualization architectures," *IEEE/ACM Trans. Netw.*, vol. 25, no. 4, pp. 2008–2025, Mar. 2017.

[12] S. Dräxler, H. Karl, and Z. Á. Mann, "JASPER: Joint optimization of scaling, placement, and routing of virtual network services," *IEEE Trans. Netw. Service Manag.*, vol. 15, no. 3, pp. 946–960, June 2018.

[13] X. Fei, F. Liu, H. Xu, and H. Jin, "Adaptive VNF scaling and flow routing with proactive demand prediction," in *Proc. IEEE INFOCOM'18*, Apr. 2018, pp. 486–494.

[14] Z. Luo, C. Wu, Z. Li, and W. Zhou, "Scaling geo-distributed network function chains: A prediction and learning framework," *IEEE J. Select. Areas Commun.*, vol. 37, no. 8, pp. 1838–1850, Aug. 2019.

[15] "MAWI Working Group Traffic Archive," http://mawi.wide.ad.jp/mawi/, 2020, [Online; accessed 12-August-2020].

[16] S. Liu, M. Yamada, N. Collier, and M. Sugiyama, "Change-point detection in time-series data by relative density-ratio estimation," *Neural Networks*, vol. 43, pp. 72–83, July 2013.

[17] R. P. Adams and D. J. MacKay, "Bayesian online changepoint detection," University of Cambridge, Cambridge, U.K., Tech. Rep., 2007.

[18] G. Comert and A. Bezuglov, "An online change-point-based model for traffic parameter prediction," *IEEE Trans. Intell. Transport. Syst.*, vol. 14, no. 3, pp. 1360–1369, Sep. 2013.

[19] C. Fraleigh, F. Tobagi, and C. Diot, "Provisioning IP backbone networks to support latency sensitive traffic," in *Proc. IEEE INFOCOM'03*, Apr. 2003, pp. 1871–1879.

[20] J. Kim and G. Hwang, "Adaptive bandwidth allocation based on sample path prediction with Gaussian process regression," *IEEE Trans. Wireless Commun.*, vol. 18, no. 10, pp. 4983–4996, Oct. 2019.

[21] Y. Cheng, W. Zhuang, and L. Wang, "Calculation of loss probability in a finite size partitioned buffer for quantitative assured service," *IEEE Trans. Commun.*, vol. 55, no. 9, pp. 1757–1771, Aug. 2007.

[22] K. Qu, W. Zhuang, Q. Ye, X. Shen, X. Li, and J. Rao, "Dynamic flow migration for embedded services in SDN/NFV-enabled 5G core networks," *IEEE Trans. Commun.*, vol. 68, no. 4, pp. 2394–2408, Apr. 2020.

[23] L. Guo, J. Pang, and A. Walid, "Dynamic service function chaining in SDN-enabled networks with middleboxes," in *Proc. IEEE ICNP'16*, Nov. 2016, pp. 1–10.

[24] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 2011.

[25] S. Chinchali, P. Hu, T. Chu, M. Sharma, M. Bansal, R. Misra, M. Pavone, and S. Katti, "Cellular network traffic scheduling with deep reinforcement learning," in *Proc. AAAI'18*, Feb. 2018.

[26] J. Wang, L. Zhao, J. Liu, and N. Kato, "Smart resource allocation for mobile edge computing: A deep reinforcement learning approach," *IEEE Trans. Emerg. Topics Comput.*, to appear, doi: 10.1109/TETC.2019.2902661.

[27] H. Li, K. Ota, and M. Dong, "Learning IoT in edge: Deep learning for the Internet of things with edge computing," *IEEE netw.*, vol. 32, no. 1, pp. 96–101, Jan.-Feb. 2018.

[28] J. Li, W. Shi, N. Zhang, and X. Shen, "Delay-aware VNF scheduling: A reinforcement learning approach with variable action set," *IEEE Trans. Cogn. Commun. Netw.*, 2020, to appear, doi: 10.1109/TCCN.2020.2988908.

[29] J. Liu, H. Guo, J. Xiong, N. Kato, J. Zhang, and Y. Zhang, "Smart and resilient EV charging in SDN-enhanced vehicular edge computing networks," *IEEE J. Select. Areas Commun.*, vol. 38, no. 1, pp. 217–228, Jan. 2020.

[30] B. Krithikaivasan, Y. Zeng, K. Deka, and D. Medhi, "ARCH-based traffic forecasting and dynamic bandwidth provisioning for periodically measured nonstationary traffic," *IEEE/ACM Trans. Netw.*, vol. 15, no. 3, pp. 683–696, June 2007.

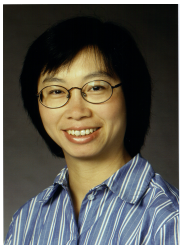
[31] A. Bayati, V. Asghari, K. Nguyen, and M. Cheriet, "Gaussian process regression based traffic modeling and prediction in high-speed networks," in *Proc. IEEE GLOBECOM'16*, Dec. 2016, pp. 1–7.

[32] C. K. Williams and C. E. Rasmussen, *Gaussian Processes for Machine Learning*. MIT press Cambridge, MA, 2006, vol. 2, no. 3.

[33] H. S. Kim and N. B. Shroff, "Loss probability calculations and asymptotic analysis for finite buffer multiplexers," *IEEE/ACM Trans. Netw.*, vol. 9, no. 6, pp. 755–768, Dec. 2001.

[34] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.
[35] Z. Xu, J. Tang, J. Meng, W. Zhang, Y. Wang, C. H. Liu, and D. Yang, "Experience-driven networking: A deep reinforcement learning based approach," in *Proc. IEEE INFOCOM'18*, Apr. 2018, pp. 1871–1879.
[36] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," in *Proc. ICLR'16*, May 2016, pp. 1–7.
[37] P. Abry and F. Sellan, "The wavelet-based synthesis for fractional Brownian motion proposed by F. Sellan and Y. Meyer: Remarks and fast implementation," *Applied and Computational Harmonic Analysis*, vol. 3, no. 4, pp. 377–383, Oct. 1996.
[38] H. Kobayashi and B. L. Mark, *System Modeling and Analysis: Foundations of System Performance Evaluation*. Pearson Education India, 2009.
[39] K. P. Murphy, "Conjugate bayesian analysis of the gaussian distribution," University of British Columbia, Vancouver, Canada, Tech. Rep., 2007.
[40] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, U.S.A.: Springer, 2006.

Kaige Qu (S'19) received her B.Sc. degree in communication engineering from Shandong University, Jinan, China, in 2013. She received her M.Sc. degrees in integrated circuits engineering and electrical engineering from Tsinghua University, Beijing, China, and KU Leuven, Leuven, Belgium, respectively, in 2016. She is currently pursuing her Ph.D. degree with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. Her research interests include resource allocation in SDN/NFV-enabled networks, 5G and beyond, and machine learning for future networking.



Weihua Zhuang (M'93−SM'01−F'08) received the B.Sc. and M.Sc. degrees from Dalian Marine University, China, and the Ph.D. degree from the University of New Brunswick, Canada, all in electrical engineering. She has been with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, since 1993, where she is a Professor and a Tier I Canada Research Chair of Wireless Communication Networks. She is the recipient of 2017 Technical Recognition Award from IEEE Communications Society Ad Hoc & Sensor Networks Technical Committee, and a co-recipient of several best paper awards from IEEE conferences. Dr. Zhuang was the Editor-in-Chief of IEEE Transactions on Vehicular Technology (2007−2013), Technical Program Chair/Co-Chair of IEEE VTC Fall 2017 and Fall 2016, and the Technical Program Symposia Chair of the IEEE Globecom 2011. She is a Fellow of the IEEE, the Royal Society of Canada, the Canadian Academy of Engineering, and the Engineering Institute of Canada. Dr. Zhuang is an elected member in the Board of Governors and VP Publications of the IEEE Vehicular Technology Society.

Xuemin (Sherman) Shen (M'97−SM'02−F'09) received his Ph.D. degree in electrical engineering from Rutgers University, New Brunswick, NJ, USA, in 1990. He is currently a University Professor with the Department of Electrical and Computer Engineering, University of Waterloo, Canada. His research focuses on network resource management, wireless network security, social networks, 5G and beyond, and vehicular ad hoc and sensor networks. He is a registered Professional Engineer of Ontario, Canada, an Engineering Institute of Canada Fellow, a Canadian Academy of Engineering Fellow, a Royal Society of Canada Fellow, a Chinese Academy of Engineering Foreign Fellow, and a Distinguished Lecturer of the IEEE Vehicular Technology Society and Communications Society. Dr. Shen received the R.A. Fessenden Award in 2019 from IEEE, Canada, James Evans Avant Garde Award in 2018 from the IEEE Vehicular Technology Society, Joseph LoCicero Award in 2015 and Education Award in 2017 from the IEEE Communications Society. He has also received the Excellent Graduate Supervision Award in 2006 and Outstanding Performance Award 5 times from the University of Waterloo and the Premiers Research Excellence Award (PREA) in 2003 from the Province of Ontario, Canada. He served as the Technical Program Committee Chair/Co-Chair for the IEEE Globecom'16, the IEEE Infocom'14, the IEEE VTC'10 Fall, the IEEE Globecom'07, the Symposia Chair for the IEEE ICC'10, the Tutorial Chair for the IEEE VTC'11 Spring, and the Chair for the IEEE Communications Society Technical Committee on Wireless Communications. He was the Editor-in-Chief of the IEEE INTERNET OF THINGS JOURNAL from 2017 to 2019. He is the Vice President on Publications of the IEEE Communications Society.

Xu Li is a senior principal researcher at Huawei Technologies Canada. He received a PhD (2008) degree from Carleton University, an M.Sc. (2005) degree from the University of Ottawa, and a B.Sc. (1998) degree from Jilin University, China, all in computer science. Prior to joining Huawei, he worked as a research scientist (with tenure) at Inria, France. His current research interests are focused in 5G and beyond. He contributed extensively to the development of 3GPP 5G standards through 90+ standard proposals. He has published 100+ refereed scientific papers and is holding 40+ issued US patents.

Jaya Rao received his B.Sc. and M.Sc. degrees in Electrical Engineering from the University of Buffalo, New York, in 2001 and 2004, respectively, and his Ph.D. degree from the University of Calgary, Canada, in 2014. He is currently a Senior Research Engineer at Huawei Technologies Canada, Ottawa. Since joining Huawei in 2014, he has worked on research and design of CIoT, URLLC and V2X based solutions in 5G New Radio. He has contributed for Huawei at 3GPP RAN WG2, RAN WG3, and SA2 meetings on topics related to URLLC, network slicing, mobility management, and session management. From 2004 to 2010, he was a Research Engineer at Motorola Inc. He was a recipient of the Best Paper Award at IEEE WCNC 2014.