Hybrid Model-Data Driven Network Slice Reconfiguration by Exploiting Prediction Interval and Robust Optimization

Fengsheng Wei, Shuang Qin, *Member, IEEE*, Gang Feng, *Senior Member, IEEE*, Yao Sun, *Senior Member, IEEE*, Jian Wang, and Ying-Chang Liang, *Fellow, IEEE*

***Abstract*—Proactive reconfiguration of network slices according to uncertain traffic demands is essential to improve network resource utilization while ensuring service quality in 5G-and-beyond systems. Existing researches on network slice reconfiguration are either model-driven or data-driven methods. However, model-driven methods may cause resource over-provisioning due to a lack of prediction mechanism, while data-driven methods are unrealistic in inter-slice reconfiguration that involves costly and time-consuming operations such as VNF migration. To address these issues, in this paper, we propose a Hybrid Model-Data driven (HMD) framework that intelligently performs inter-slice reconfiguration by leveraging prediction interval and robust optimization. We design a Prediction Interval-oriented Predictor (PIP) to produce a prediction interval that can bracket the future traffic demand with a prespecified probability. Based on the prediction interval, we design an inter-slice reconfiguration scheme (named box optimizer) to perform fast inter-slice reconfigurations. To tackle the over-conservativeness of the box optimizer, we further design the ellipsoid optimizer with better optimality at a cost of increased complexity. Numerical results demonstrate that the proposed framework can provide high robustness with low power consumption. Meanwhile, the trade-off between the power consumption and the realized robustness can be flexibly adjusted according to the type of slice and the level of traffic demand fluctuations.**

***Index Terms*—network slice reconfiguration, dynamic network slicing, prediction interval, robust optimization**

I. Introduction

NETWORK slicing is a key technology to support the diversified service provisioning in 5G-and-beyond and 6G networks [2]. Based on Software Defined Network (SDN) and Network Function Virtualization (NFV), network slicing enables on-demand, fully-automated provisioning of services according to the users' traffic demands. However, traffic demands of a network slice are usually uncertain and fluctuate over time. Resource slicing by only taking into account the nominal traffic demand or Peak Hour Interval (PHI) demand without considering the future traffic demand variations may cause a mismatch between resource provisioning and traffic demands [3]. Thus, in order to meet subscribers' ever-changing traffic demands, it is essential to reconfigure network slices adaptively.

A network slice is defined by its Service Function Chain (SFC), which is composed of a number of sequentially interconnected Virtual Network Functions (VNFs) [4]. The reconfiguration of network slices can be either *intra-slice reconfiguration* or *inter-slice reconfiguration*. Intra-slice reconfiguration is performed at small time scales which adjusts the flow path and the association of VNF instance of the flows within individual network slices [5], while inter-slice reconfigurations are performed at large time scales which involves resource scaling between multiple network slices(i.e., slice breathing [6]) and VNF migration (i.e., slice mobility [7]). Some recent work such as [4], [8] propose to adaptively reconfigure network slices to meet users' instantaneous traffic demands. However, due to a lack of prediction mechanism, these reactive schemes may cause Service Level Agreement (SLA) violations since inter-slice reconfiguration involves heavyweight operations such as VNF scaling and migration [9]. Moreover, these operations may cause service interruption due to the time-consuming data transfer process in the migration of virtual machines (or containers) and the associated user data. Therefore, in order to overcome Quality of Service (Qos) degradation, it is of vital importance to reconfigure network slices according to traffic demands in a proactive fashion. Recently, two kinds of methods are widely adopted to perform proactive slice reconfigurations: model-driven methods and data-driven methods.

To tackle the uncertain demand issue, many researchers have proposed model-driven solutions which adopt Robust Optimization (RO) to perform inter-slice reconfiguration [10], [11]. In these investigations, it is assumed that the future traffic demand is within an *uncertainty set*, based on which a RO model regarding network slicing reconfiguration is formulated. The solution of the RO model is "robust" enough such that any constraints in the uncertainty set can be satisfied. However, since the uncertainty sets used in these work are hand-crafted rather than prediction-generated, their width are usually too large and may produce over-conservative solutions. Moreover, due to a lack of prediction mechanism, the width of the

This work was supported by the National Science Foundation of China under Grant 62071091, the Fundamental Research Funds for the Central Universities under Grant ZYGX2020ZB044, and Huawei Cooperation Program under Grant TC20210316002.

F. Wei, S. Qin, G. Feng, J. Wang and Y. C. Liang are with the National Key Laboratory of Science and Technology on Communications, University of Electronic Science and Technology of China, Chengdu 611731, China, and also with the Yangtze Delta Region Institute (Huzhou), University of Electronic Science and Technology of China, Huzhou 313001, P. R. China. (e-mail: weifengsheng@std.uestc.edu.cn; blueqs@uestc.edu.cn; fenggang@uestc.edu.cn; wangjians@std.uestc.edu.cn; liangyc@ieee.org). Y. Sun is with James Watt School of Engineering, University of Glasgow, United Kingdom (email: Yao.Sun@glasgow.ac.uk). S. Qin is the corresponding author.

This work has been partially presented at IEEE Globecom 2020 [1].

uncertainty set in these investigations cannot be adaptively adjusted according to the traffic fluctuations, which will cause resource over-provisioning even in mild uncertainty cases.

Recently, with the advancement of machine learning, a number of researchers seek to reconfigure network slices with data-driven methods by exploiting various machine learning techniques [12], [13]. These methods usually exploit the *predict-then-optimize* paradigm which first predicts the traffic demand and then optimizes the network slice reconfiguration problem based on the predictions. However, the predictors employed in these investigations are all point predictors, which are unable to provide any indication about the accuracy of their predictions. Moreover, since traffic demand is usually affected by various stochastic events, point predictors are usually unreliable and inaccurate in traffic demand forecasting [14]. This will lead to the inaccuracy of the network slice reconfiguration model, thus the obtained reconfiguration solution may not accurately match the future traffic demand, resulting in resource over-provisioning or SLA violation.

Unlike point predictors that predict the traffic demand without indicating any accuracy, *prediction interval-oriented predictors* [14] generate an interval that can bracket future traffic demand with a prespecified probability called *confidence level* $(1-\alpha)$, which can be flexibly specified according to the type of network slice. Moreover, since the traffic demand of a network slice fluctuates with short or long-lived effects (e.g., the mobility of end users and changes in the number of served users), the aggregated traffic demands exhibit periodicity and variability [15], which can be well captured by the prediction intervals. These observations inspire us to exploit prediction intervals for designing better schemes for inter-slice reconfiguration.

In this paper, we propose a Hybrid Model-Data driven (HMD) framework to proactively reconfigure network slices under demand uncertainty. In this framework, we propose a prediction scheme that forecasts the aggregate traffic demands of individual network slices and produces a prediction interval with prespecified accuracy. Based on the prediction interval, we formulate the inter-slice reconfiguration problem as an RO model. Finally, through mathematical transformations and deriving its *robust counterpart* (RC), the RO model is equivalently transformed into a Mixed Integer Linear Program (MILP) as well as a Mixed Integer Second Order Cone Program (MISOCP) and is solved by a state-of-the-art solver.

Our contributions can be summarized as follows:

1) We propose to employ Gated Recurrent Unit (GRU) and the bootstrap method to design the Prediction Interval-oriented Predictor (PIP), which generates prediction intervals for the future traffic demands. Compared with point predictors, PIP can avoid making risky slice reconfiguration decisions under demand uncertainty.

2) By jointly leveraging prediction interval and robust optimization, we design a hybrid model-data driven framework that proactively reconfigures network slices. The combination of prediction interval and robust optimization opens the door for further attempts to address the uncertainty issues in network slicing.

3) We present numerical results that evaluate the performance of the HMD framework based on real traffic data, and obtain new insights that will help Infrastructure providers (InPs) make flexible trade-offs between the power consumption and the realized robustness when performing inter-slice reconfigurations under uncertain traffic demands.

The remainder of the paper is organized as follows. Section II reviews the related work on network slice reconfiguration. In Section III, an overview of the proposed HMD framework is provided. We elaborate the design of PIP in Section IV. In Section V, we present the system model and problem formulation. In Section VI, we derive the RC of the problem and its solution. In Section VII, we present the numerical results, and finally we conclude the paper in Section VIII.

II. Related Work

Recently, there has been some research work on the problem of network slice reconfiguration, which can be roughly classified into three categories, i.e., model-driven methods, data-driven methods and hybrid model-data-driven methods. In this section, we overview the researches that are most relevant to our work.

A. Model-Driven Methods

Model-driven slice reconfiguration researches can be further classified into two classes: reactive schemes [4], [8], [16] and proactive schemes [10], [17]. Reactive slice reconfigurations are performed after a mismatch between resource demand and provisioning occurs, while proactive reconfigurations are performed before the mismatch takes place. In reactive schemes, the network slices are reconfigured to meet their instantaneous traffic demands. Wang *et al.* [4] proposed a hybrid slice reconfiguration model to avoid a mismatch between slices' demands and resource provisioning. Based on column generation, the authors of [8] proposed to implement a make-before-break scheme to perform inter-slice reconfiguration with aim of reducing network operational costs. In [16], the authors proposed a dynamic resource allocation scheme for dynamic 5G network slicing for a vehicular emergency scenario.

For proactive network slice reconfiguration, it is of vital importance to exploit the information about future traffic demands. The authors of [10] proposed a light-robustness optimization model for network slicing, where the future traffic demand is modeled as a random variable that takes values in an interval following an unknown distribution. In [17], the authors proposed a Mixed Integer Program (MIP) model for the network slice design and deployment problem, in which the Γ-robust uncertainty set is used to emulate the uncertain traffic demands. However, due to the lack of prediction mechanism, the uncertainty sets in these researches are too conservative and thus may lead to resource over-provisioning.

B. Data-Driven Methods

Data-driven approaches use data as input to make end-to-end slice reconfiguration decisions [5], [18]–[20]. The authors of [18] proposed an intelligent resource scheduling strategy

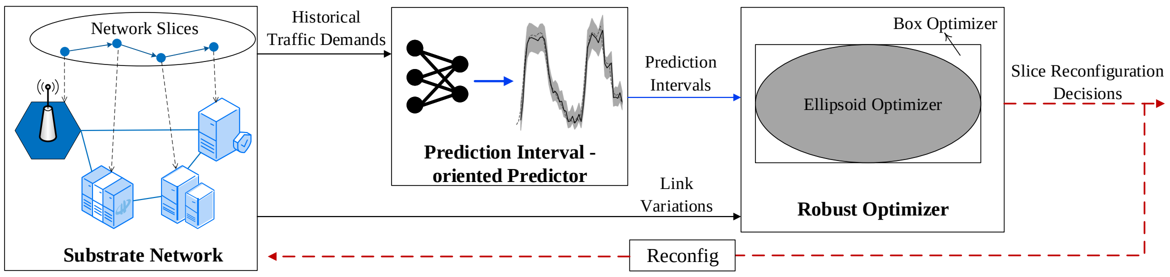


Fig. 1. The architecture of the HMD framework.

by exploiting a collaborative learning framework that consists of Deep Learning (DL) in conjunction with Reinforcement Learning (RL). In [19], the authors addressed the demand-aware inter-slice resource management in network slicing by using DQN within which a discrete normalized advantage function is introduced to avoid unnecessary evaluations of Q-value. To maximize the long-term average revenue under uncertain demands of network slices, the authors of [20] proposed a dynamic resource management framework where DRL is adopted with an extra speedup mechanism. In our previous work [5], we modeled the intra-slice reconfiguration problem as an MDP and solved it by using Branching Dueling Q-network. Since data-driven methods are data-intensive and require numerous trial-and-error interactions with the environment, it is unrealistic to adopt them in inter-slice reconfiguration that involves costly and time-consuming operations such as VNF migration.

C. *Hybrid Model-Data Driven Methods*

To overcome the drawbacks of pure model-driven methods and data-driven methods, a simple yet effective idea is to combine the advantages of these two methods, resulting in the hybrid model-data driven methods for network slice reconfiguration [12], [13], [21]. These solutions usually exploit the predict-then-optimize paradigm. In [12], the authors address the network slice admission control and traffic scheduling based on the predictions of future traffic by using Holt-Winters theory. Whereas in [13], the authors chose to predict the number of VNF instances required to satisfy the uncertain traffic demands. Based on the prediction, an Integer Linear Programming (ILP) is solved to perform proactive VNF relocation with aim of minimizing end-to-end delay. In contrast, in [21], radio slices capacity is predicted, and the prediction results are used to assist the subsequent network slice resource allocation.

However, these solutions are not adequate since only point prediction is employed, resulting in a lack of confidence for the subsequent slice reconfiguration. To the best of our knowledge, interval prediction is rarely exploited in existing hybrid model-data driven methods to address the demand uncertainty in network slicing.

III. Overview of the HMD Framework

The architecture of our proposed HMD framework is shown in Fig. 1. In HMD framework, inter-slice reconfigurations between multiple network slices are performed periodically by the management and orchestration (MANO) [22] entity at discrete time steps. In our considered scenario, a number of network slices of different service types have been deployed over a shared common substrate network. During the lifecycle of these slices, their aggregated traffic demands are dynamic due to the ever-changing demands and possible mobility of the slice users. Therefore, these slices need to be adaptively reconfigured to match the ever-changing demands. To this end, for individual slices, we first train a corresponding predictor (i.e., PIP) by the historical traffic data to produce prediction intervals of the future demands. Thereafter, based on the prediction interval produced by the predictor, an optimization method is designed to make proactive reconfiguration decisions on these slices.

A. *PIP Design Principles*

For individual network slices, a PIP is designed to forecast its traffic demands in future time windows. The aggregated traffic demand of the subscribed users in a network slice is essentially a time series that exhibits distinct long-term trends with strong periodicities [15]. Therefore, it can be predicted by the popular Recurrent Neural Networks (RNNs), such as LSTM, GRU, etc.

However, traditional RNN-based predictors only generate point predictions, thus they suffer from two limitations despite their popularity in traffic demand prediction. First, RNN-based predictors produce unsatisfactory prediction accuracy under high traffic uncertainty. This is owing to the unpredictability of short-lived events, which cause traffic bursts beyond the overall trend [14]. Second, the point predictions generated by RNNs do not provide any information about their accuracy. These drawbacks make RNN-based predictors unable to provide sufficiently accurate traffic predictions and the credibility for subsequent slice reconfigurations.

Instead, prediction interval offers an upper bound and a lower bound that can bracket future traffic demand with a prescribed probability, thereby making the predictions more meaningful and reliable for the subsequent slice reconfigurations. Such prediction intervals can not only produce prediction results for future traffic demands, but also can provide the accuracy of the prediction. Therefore, instead of pursuing an unreliable point prediction, we propose to design an interval predictor for individual network slices. At present, a variety of methods have been proposed to generate prediction intervals, such as bootstrap method, Mean-Variance Estimation (MVE) method, etc [23]. In the design of PIP, due to its adaptability and stability in time series prediction [14], we choose the bootstrap method to produce prediction intervals for individual slices.

B. The Overall Design of the Optimizer

The optimizer is designed to make proactive inter-slice reconfiguration decisions based on the prediction intervals provided by the PIP. In the design of the optimizer, two kinds of uncertainties are considered. The first uncertainty comes from the prediction intervals produced by PIP, which represents the possible realizations of future traffic demand. The second uncertainty originates from the stochastic bandwidth of wireless backhauls in 5G/B5G infrastructure [24]. By jointly considering these uncertainties, an RO model is formulated to conduct inter-slice reconfigurations. Thereafter, two solutions with different complexity and different level of optimality are proposed to resolve the RO, which are named *box optimizer* and *ellipsoid optimizer*.

The box optimizer is designed based on the box uncertainty set that is constructed by directly combining the prediction intervals and the intervals reflect wireless link perturbations. Since the box uncertainty set is a high-dimensional box-shape set, the RO is turned out to be a Robust Mixed Integer Linear Program (RMILP). By deriving its robust counterpart based on Lagrangian Duality, the RMILP is equivalently transformed into a MILP and is solved by state-of-the-art solvers.

However, the box optimizer produces excessively conservative solutions since the worst possible realizations within the box uncertainty set must be satisfied. In practice, the region of the possible future traffic demands is a high-dimensional ellipsoid since the aggregated traffic rate within a network slice follows normal distribution [25]. *Therefore, an ellipsoid uncertainty set that inscribes in the box uncertainty set can provide almost the same coverage probability over the uncertain parameters as that of the box uncertainty set.* Inspired by this idea, we further design an ellipsoid optimizer based on the ellipsoid uncertainty set. Compared with the box optimizer, the ellipsoid optimizer can further reduce the power consumption at a cost of increased complexity on the premise of nearly the same guaranteed robustness. In the following two sections, we will introduce the design of PIP and the optimizers in detail.

IV. The Data-Driven PIP

The framework of the proposed PIP is shown in Fig. 2, which jointly exploits the bootstrap method and GRUs to generate prediction intervals for individual network slices. (Please note that due to space limit, the internal gates of GRU are not depicted in detail in Fig. 2, but are represented by hidden layers instead). The rationale behind this particular design is as follows. First, compared with other single-neural-network-based methods such as MVE and Bayesian methods, bootstrap method exploits an ensemble of Neural Networks (NNs) to better address the model and data uncertainties [26], [27]. Second, compared with LSTM, GRU is more preferred because it has fewer internal gates than LSTM, thus saving a lot of training time without sacrificing performance [28].

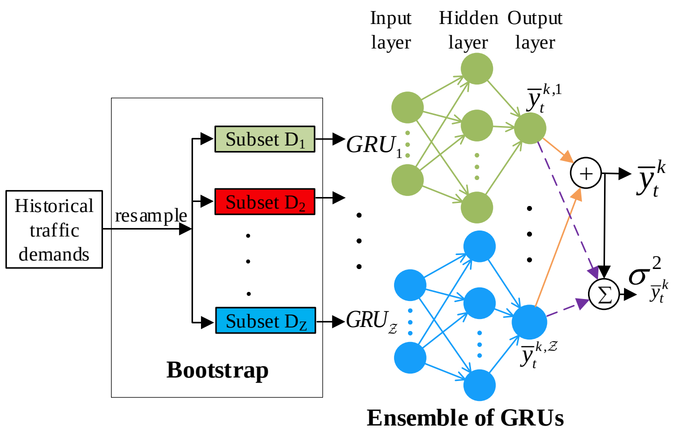


Fig. 2. The framework of the proposed PIP.

Therefore, the design of PIP jointly exploits the advantages of the bootstrap method and GRUs to obtain high-quality prediction intervals in traffic demand forecasting. The detailed principle of PIP is given as follows.

Theoretically, the prediction target (i.e., r_t^k, which is the actual traffic demand of slice k to be predicted at time t) can be expressed by

$$r_t^k = y_t^k + \epsilon_t^k, \tag{1}$$

where y_t^k is the *true regression*, and ϵ_t^k is the *noise* with zero mean and normal distribution [14]. In practice, the true regression y_t^k is approximated by the prediction $\bar{y}_t^k$ made by the GRU.

The basic idea of the bootstrap method is that an ensemble of GRUs can produce a less biased estimate of the true regression y_t^k. Therefore, an ensemble of $\mathcal{Z}$ GRU models are built and y_t^k is approximated by averaging the point predictions of these GRU models, i.e.,

$$\bar{y}_t^k = \frac{1}{\mathcal{Z}} \sum_{i=1}^{\mathcal{Z}} \bar{y}_t^{k,i}, \tag{2}$$

where $\bar{y}_t^{k,i}$ is the output of the ith GRU network. To train these GRU models, $\mathcal{Z}$ training datasets $\{D_i\}_{i=1}^{\mathcal{Z}}$ are resampled from the original dataset with replacement. Thereafter, the subset D_i is used to train the ith GRU model and produce the prediction result $\bar{y}_t^{k,i}$.

In essence, prediction interval quantifies the uncertainty of the difference between the true traffic demand r_t^k and the predicted demand $\bar{y}_t^k$. The difference is given by

$$r_t^k - \bar{y}_t^k = (y_t^k - \bar{y}_t^k) + \epsilon_t^k. \tag{3}$$

The uncertainty of the difference between r_t^k and $\bar{y}_t^k$ is quantified by the variance of $r_t^k - \bar{y}_t^k$. In general, it is supposed that $y_t^k - \bar{y}_t^k$ and ϵ_t^k are statistically independent [14]. Therefore, the variance of $y_t^k - \bar{y}_t^k$ is given by

$$\sigma^2_{r_t^k} = \sigma^2_{\bar{y}_t^k} + \sigma^2_{\epsilon_t^k}, \tag{4}$$

where $\sigma^2_{\bar{y}_t^k}$ is the variance of model misspecification errors, and $\sigma^2_{\epsilon_t^k}$ is the variance caused by random noise. The variance

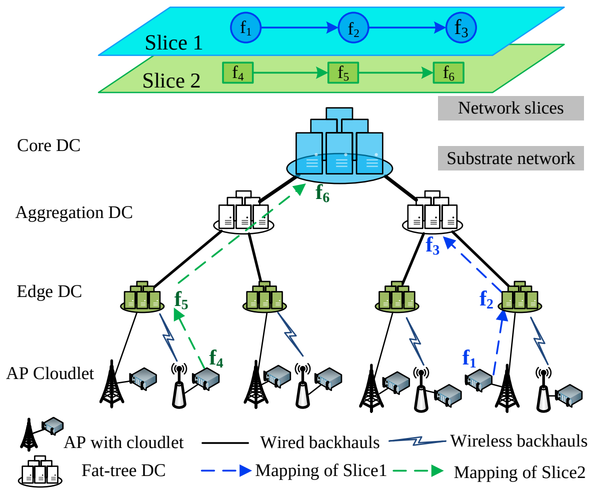


Fig. 3. The considered system model, in which the substrate network is a hierarchical DC-based 5G infrastructure, and each DC is with a fat-tree topology.

of model misspecification errors (i.e., $\sigma^2_{\bar{y}^k_t}$) is approximated by the sample variance of the outputs of these $\mathcal{Z}$ GRU models:

$$\sigma^2_{\bar{y}^k_t} = \frac{1}{\mathcal{Z}-1}\sum_{i=1}^{\mathcal{Z}}(\bar{y}^{k,i}_t - \bar{y}^k_t) \tag{5}$$

From (3) and (4), the variance of the noise (i.e., $\sigma^2_{\epsilon^k_t}$) is approximated by:

$$\sigma^2_{\epsilon^k_t} \simeq E[(r^k_t - \bar{y}^k_t)^2] - \sigma^2_{\bar{y}^k_t}. \tag{6}$$

We train another GRU (named residual GRU) with the same structure to predict the residuals of the input values:

$$\delta^2_{t,k} = max\{(r^k_t - \bar{y}^k_t)^2 - \sigma^2_{\bar{y}^k_t}, 0\}. \tag{7}$$

According to (7), a new dataset, $\{(r^k_\tau, \delta^2_{\tau,k})\}^{t-1}_{\tau=1}$, is constructed to train the residual GRU with the following loss function [14]:

$$L = \frac{1}{2}\sum_{\tau=1}^{t-1}[ln(\sigma^2_{\epsilon^k_t}) + \frac{\delta^2_{t,k}}{\sigma^2_{\epsilon^k_t}}]. \tag{8}$$

Now we know both $\sigma^2_{\bar{y}^k_t}$ and $\sigma^2_{\epsilon^k_t}$, then the variance of $r^k_t - \bar{y}^k_t$ is obtained from (4). Thus, the $(1-\alpha)\%$ prediction interval for r^k_t is generated according to [29]:

$$PI^k_t = [\bar{y}^k_t - z_{\alpha/2}\cdot\sigma_{r^k_t}, \bar{y}^k_t + z_{\alpha/2}\cdot\sigma_{r^k_t}], \tag{9}$$

where $z_{\alpha/2}$ is the $\alpha/2$ quantile of the standard normal distribution. This prediction interval will be regarded as the uncertain parameters of the RO model, which will be elaborated in the following sections.

V. The Optimizer: Problem Formulation

In this section, we present the system model and the problem formulation of inter-slice reconfiguration. The solution of the problem will be given in the next section.

1) *Infrastructure Model*

Fig. 3 shows the network slicing model in 5G networks, which mainly includes two parts: the network slices and the substrate network. For ease of reference, the symbols used in the paper are summarized in Table I. The substrate network is a hierarchical data-center (DC) based 5G infrastructure that is modeled as a directed graph denoted by $G = (V, E)$, in which V and E are sets of substrate nodes (i.e., the physical servers. We use these terms interchangeably) and substrate links respectively. In the considered model, the VNFs of a network slice can be scattered across multiple datacenters. Such deployment can benefit from low latency and radio/location awareness of network edge, as well as the large capacity of the Core network. For instance, for IoT slices, some delay-insensitive network functions, (e.g., subscription, notification, and security) can be deployed at the Core Network, while the mission-critical functions such as device discovery, device registration can be deployed at the network edge [30].

The available computing resource of node i is denoted by C_i. Note that other types of resources such as memory and storage can be incorporated as needed. To capture the uncertain bandwidth of wireless backhauls in 5G/B5G networks [24], we use the stochastic parameter $\widetilde{B}^S_{ij}$ to denote the time-varying bandwidth of substrate link $(i,j) \in E$. Specifically, $\widetilde{B}^S_{ij}$ takes values in the symmetric interval $[B^S_{ij} - \hat{B}^S_{ij}, B^S_{ij} + \hat{B}^S_{ij}]$.

2) *Network Slice Model*

There are $|\mathcal{K}|$ slices deployed over the substrate network, where $\mathcal{K}$ is the set of network slices. The SFC of slice $s_k \in \mathcal{K}$ is defined by a three tuple $(S_k, T_k, \mathcal{F}_k)$, which are ingress node, egress node, and the set of requested VNFs, respectively. The VNF set $\mathcal{F}_k$ consists of l_k ordered VNFs denoted by

TABLE I
Sets, Parameters and Variables Notation

Notation	Description
	Sets
$G = (V, E)$	Substrate network, with V and E are substrate nodes and links respectively
$\mathcal{K}$	Set of network slices
S_k, T_k	Ingress node and egress node of slice k
$\mathcal{F}_k$	Set of VNFs of slice k
Q_k	SFC of slice k, where $Q_k = \mathcal{F}_k \cup \{f^k_0, f^k_{l_k+1}\}$
	Deterministic Parameters
C_i	Available computing resources of node i
$\delta_i(f^k_m)$	Binary parameter indicates whether node i can provide VNF f^k_m
P^i_{idle}	Power consumption of node i on idle state
P^i_{max}	Power consumption of node i at full utilization
P^i_{port}	Power consumption of NIC on node i in active mode
β_b, β_c	Unit bandwidth and computing resources consumed by one unit of data flow
	Uncertain Parameters
$\widetilde{B}^S_{ij}$	Bandwidth of substrate link (i,j)
$\widetilde{r}^k$	Traffic demands of slice k
	Variables
$b_{ij}(e^m_k)$	Bandwidth allocated to virtual link e^m_k by the substrate link (i,j)
$z_{ij}(e^m_k)$	Auxiliary binary variable indicating the positivity of $b_{ij}(e^m_k)$
$y^m_{i,k}$	Binary variable indicates whether virtual node f^k_m of slice k is deployed on substrate node i
γ^i	Binary variable indicates the on-off status of substrate node i

$\mathcal{F}_k = \{f_1^k, \cdots, f_{l_k}^k\}$ [31]. We introduce two dummy VNF f_0^k and $f_{l_k+1}^k$ to denote the ingress node and egress node of slice k respectively. Therefore, the SFC of slice k can be represented by $Q_k = \mathcal{F}_k \cup \{f_0^k, f_{l_k+1}^k\}$. For ease of presentation, we define VNF f_m^k of slice k to be its *virtual node*. Accordingly, the link between f_m^k and f_{m+1}^k of slice k is defined as its *virtual link*, which is denoted by e_k^m. At time t, the prediction interval provided by the PIP on the traffic demand of slice k is denoted by $\left[\bar{r}^k - \hat{r}^k, \bar{r}^k + \hat{r}^k\right]$, where $\hat{r}^k = z_{\alpha/2} \cdot \bar{r}^k$ according to (9). Please note that we omit the subscript t for simplicity of our description.

3) The Constraints

According to the VNF-MANO framework proposed by ETSI, the VNF instances (VNFIs) of a network slice are managed by the VNF Manager (VNFM) [22]. In fact, a VNFI is implemented as a Virtual Machine (VM) or container deployed on a physical node. We use the binary variable $y_{i,k}^m$ to indicate whether the virtual node f_m^k of slice s_k is deployed on node i or not. For virtual node f_m^k, since only the physical nodes that are capable of f_m^k can be its mapping candidates, hence we have

$$y_{i,k}^m \leq \delta_i(f_m^k), \forall f_m^k \in Q_k, \forall k \in \mathcal{K}, i \in V, \tag{10}$$

where $\delta_i(f_m^k)$ is a binary indicator parameter that indicates whether node $i \in V$ can offer function $f_m^k \in \mathcal{F}$, wherein $\mathcal{F}$ denotes the set of VNFs in the system.

In our model, flow splitting is prohibited due to the following reasons. First, flow splitting may increase the number of activated servers and NICs, which will increase the overall power consumption. Moreover, flow splitting will also lead to certain coordination overhead, such as state information maintenance, traffic distribution, .etc [32]. Therefore, to avoid flow splitting, we constrain that each VNF is mapped to exactly one physical node, which means that

$$\sum_{i \in V} y_{i,k}^m = 1, \forall f_m^k \in Q_k, \forall k \in \mathcal{K}. \tag{11}$$

It is worth mentioning that our model allows *VNF-consolidation*, which means that multiple VNFs of a network slice can be consolidated on the same server [33]. VNF-consolidation can help to reduce the number of transit nodes, thus further decreasing the power consumption.

Please note that the node mapping variables corresponding to f_0^k and $f_{l_k+1}^k$, i.e., $y_{i,k}^0$ and $y_{i,k}^{l_k+1}$ for all $i \in V$, are known variables which are decided by the ingress node S_k and egress nodes T_k of slice k, i.e.,

$$y_{i,k}^m = \begin{cases} 1, \text{ if } i = S_k, m = 0 \text{ or } i = T_k, m = |\mathcal{F}_k| + 1, \\ 0, \text{ otherwise }. \end{cases} \tag{12}$$

We use variable $b_{ij}(e_k^m)$ to denote the bandwidth allocated to virtual link e_k^m by the physical link (i,j). We assume that one unit of data flow consumes β_b units of bandwidth resources. To meet the bandwidth requirements on each virtual link, we have

$$\sum_{ij \in E} b_{ij}(e_k^m) \geq \beta_b \tilde{r}^k, \forall m \in \{0, \cdots, l_k\}, \forall k \in \mathcal{K}. \tag{13}$$

In addition, we have the following capacity constraints on each physical link which indicates that the bandwidth allocated to all the slices by link $(i,j) \in E$ should not exceed its available bandwidth, i.e.,

$$\sum_{k \in \mathcal{K}} \sum_{m=0}^{l_k-1} b_{ij}(e_k^m) \leq \widetilde{B}_{ij}^S, \forall ij \in E. \tag{14}$$

We assume that one unit of data flow consumes β_c units of computing resources. Given that the predicted traffic demand $\tilde{r}^k$ of slice k, the amount of computing resource required to meet slice k's demand will be $\beta_c \tilde{r}^k$. By summing up the computing resource requirement of all network slices, we have the following computation capacity constraint:

$$\widetilde{c}_{allc}^i = \sum_{k \in \mathcal{K}} \sum_{f_m^k \in \mathcal{F}_k} \beta_c \tilde{r}^k y_{i,k}^m \leq C_i, \forall i \in V, \tag{15}$$

where $\widetilde{c}_{allc}^i$ is the amount of computing resources allocated to all the slices on the ith substrate node.

We introduce binary variable $z_{ij}(e_k^m)$ to indicate the positivity of $b_{ij}(e_k^m)$, i.e., $z_{ij}(e_k^m) = 1$ if $b_{ij}(e_k^m) > 0$, otherwise $z_{ij}(e_k^m) = 0$. Such nonlinear relationship between $z_{ij}(e_k^m)$ and $b_{ij}(e_k^m)$ can be equivalently transformed into the following linear constraints:

$$\epsilon b_{ij}(e_k^m) \leq z_{ij}(e_k^m) \leq L b_{ij}(e_k^m), \forall ij \in E, f_m^k \in \mathcal{F}_k, k \in \mathcal{K}, \tag{16}$$

where ϵ is a sufficiently small positive constant such that $\epsilon b_{ij}(e_k^m) \leq 1$, and L is a sufficiently large positive constant. Please note that ϵ and L in constraint (16) should not be set too small or too large. If ϵ is too small, the above transformation will not be equivalent due to the possible rounding and tolerance mechanism of the solver [34]. Therefore, we set them as tight as possible, which means $\epsilon = \min_{ij \in E}\{1/(b_{ij}^S + \hat{b}_{ij}^S)\}$ and $L = \max_{ij \in E}\{1/(b_{ij}^S - \hat{b}_{ij}^S)\}$.

By introducing variable $z_{ij}(e_k^m)$ that indicates the positivity of $b_{ij}(e_k^m)$, the nonlinearity in the objective function can be avoided (i.e., the nonlinear relation between N_{port}^i and $b_{ij}(e_k^m)$, see Constraint (21)). Moreover, the path connectivity constraint can be readily expressed by $\{z_{ij}(e_k^m)\}$ and $\{y_{i,k}^m\}$ in linear constraints [5], [35]. Particularly, the following should hold for all $(i,k,j) \in V \times \mathcal{K} \times \mathcal{F}_k \cup \{f_0^k\}$

$$\begin{cases} \sum_j z_{ij}(e_k^m) - \sum_j z_{ji}(e_k^m) = 1, \text{ if } i = S_k, m = 0 \\ \sum_j z_{ij}(e_k^m) - \sum_j z_{ji}(e_k^m) = -1, \text{ if } i = T_k, m = l_k \\ \sum_j z_{ij}(e_k^m) - \sum_j z_{ji}(e_k^m) = (y_{i,k}^m - y_{i,k}^{m+1}), \text{ otherwise} \end{cases} \tag{17}$$

such that the physical links are constrained to form a connected path for each network slice.

4) Objective Function

In recent years, the power efficiency of IT infrastructure has become the focus of attention since energy consumption accounts for a large proportion of the operators' expenditure and the proportion is increasing year by year [36]. Therefore, the objective in our model is to minimize the power consumption of the system. Please note that our proposed HMD framework is also effective for other choices of objectives (such as the cost of resource consumption, the number of activated substrate nodes, and the total link flow in the network, etc) with necessary changes on corresponding constraints.

For substrate node $i \in V$, its power consumption originates from three aspects: the power consumed by the VNFIs, the power consumption of the NICs used for packet forwarding, and the power required to keep node i [36] active, i.e.,

$$\widetilde{P}^i = P^i_{cpu} + P^i_{NIC} + P^i_{static} \tag{18}$$

First, the power consumed by the VNFIs on server i (i.e., P^i_{cpu}) depends on the amount of allocated computing resources [36], i.e.,

$$P^i_{cpu} = (P^i_{max} - P^i_{idle}) \times \widetilde{c}^i_{allc}/C_i, \tag{19}$$

where P^i_{max} is the maximum power consumption at full utilization, P^i_{idle} is the power consumption of server i on idle state, and $\widetilde{c}^i_{allc}$ is defined as in (15).

Second, the power consumption of the NICs on node i (i.e., P^i_{NIC}) is given by [36]:

$$P^i_{NIC} = P^i_{port} \times N^i_{port}, \tag{20}$$

where P^i_{port} is the power consumed by the NICs at active mode on node i, which corresponds to $P_{dynamic}$ in (129) of [36]; N^i_{port} is the number of active NICs on node i, which is given by:

$$N^i_{port} = \sum_{k\in\mathcal{K}} \sum_{m}^{l_k-1} \left(\sum_{j\in V} z_{ij}(e^m_k) + \sum_{j\in V} z_{ji}(e^m_k) \right). \tag{21}$$

Finally, the power required to keep the ith server on (i.e., P^i_{static}) is given by:

$$P^i_{static} = \gamma^i P^i_{idle}, \tag{22}$$

where γ^i is a binary variable indicates whether server i is active or not, i.e., $\gamma^i = 1$ when server i is on, otherwise $\gamma^i = 0$. If no NIC is activated on node i, it means that the node is not used by any slice and can be shut down to save energy. Thus, the on-off state of server i is decided by the positivity of N^i_{port}, i.e., $\gamma^i = 1$ if $N^i_{port} > 0$ otherwise $\gamma^i = 0$. We replace such nonlinear relationships by the following linear constraints:

$$\zeta N^i_{port} \le \gamma^i \le D N^i_{port}, \forall i \in V, \tag{23}$$

where ζ and D are sufficiently small and large positive constants respectively. The value of ζ and D in constraint (23) can be set in a similar ways as ϵ and L in constraint (16).

5) Problem Formulation

Given the traffic prediction intervals and the above network slicing model, the inter-slice reconfiguration problem is formulated as follows:

$$\min_{\mathbf{y},\mathbf{z},\mathbf{b},\boldsymbol{\gamma}} \sum\nolimits_i \widetilde{P}^i \tag{24}$$

$$s.t.\ (10) - (17), (23) \tag{24.1}$$

$$y^m_{i,k} \in \{0,1\}, \forall i \in V, k \in \mathcal{K}, f^k_m \in \mathcal{F}_k \tag{24.2}$$

$$z_{ij}(e^m_k) \in \{0,1\}, \forall ij \in E, k \in \mathcal{K}, m \in \{1,\cdots,l_k-1\} \tag{24.3}$$

$$b_{ij}(e^m_k) \ge 0, \forall ij \in E, k \in \mathcal{K}, m \in \{1,\cdots,l_k-1\} \tag{24.4}$$

$$\gamma^i \in \{0,1\}, \forall i \in V. \tag{24.5}$$

Since $\widetilde{r}^k$ and $\widetilde{B}^S_{ij}$ are uncertain parameters, thus this problem is a Robust Mixed Integer Problem (RMIP). For ease of presentation, we rewrite problem (24) as the following equivalent matrix form:

$$\min_{\mathbf{u}} \tilde{\mathbf{c}}^T \mathbf{u} \tag{25}$$

$$s.t.\ \widetilde{\mathbf{B}}_1 \mathbf{u} \le \widetilde{\mathbf{p}}_1, \tag{25.1}$$

$$\mathbf{B}_2 \mathbf{u} \le \mathbf{p}_2, \mathbf{u} \in \mathcal{U}, \tag{25.2}$$

The dimensions of $\widetilde{\mathbf{B}}_1$ and $\mathbf{B}_2$ are $I_1 \times J$ and $I_2 \times J$ respectively, in which $I_1 = (|V| + |E| + \sum_{k=1}^{|\mathcal{K}|}(l_k+1))$, $I_2 = 2(|V| + |E|)(\sum_{k=1}^{|\mathcal{K}|}(l_k+1)) + 3\sum_{k=1}^{|\mathcal{K}|}(l_k+2) + 2|V|$, $J = 2|E|\sum_{k=1}^{|\mathcal{K}|}(l_k+1) + |V|\sum_{k=1}^{|\mathcal{K}|}(l_k+2) + |V|$. The detailed transformation is described in Appendix A.

VI. The Optimizer: Solutions

In this section, we elaborate the robust optimizer which is used to solve problem (25) in the following steps. First, we derive the standard robust reformulation of problem (25). Second, we construct the box uncertainty set and derive the corresponding box optimizer. Finally, the ellipsoid uncertainty set is constructed upon which the ellipsoid optimizer is designed.

A. Standard Robust Problem Reformulation

The *robust counterpart* (RC) is a tractable deterministic equivalence of the robust problem. To derive the RC of problem (24), we need to transform problem (24) into the standard form of robust problem, whose uncertainty parameters only occur in the Left Hand Side (LHS) of its constraints [37]. To this end, we first eliminate the uncertainty in the objective function by introducing an artificial variable ρ such that problem (25) can be equivalently transformed to:

$$\min_{\mathbf{u},\rho} \rho \tag{26}$$

$$s.t.\ \tilde{\mathbf{c}}^T \mathbf{u} \le \rho \tag{26.1}$$

$$\widetilde{\mathbf{B}}_1 \mathbf{u} \le \widetilde{\mathbf{p}}_1 \tag{26.2}$$

$$\mathbf{B}_2 \mathbf{u} \le \mathbf{p}_2, \mathbf{u} \in \mathcal{U}. \tag{26.3}$$

Next, we eliminate the uncertain parameters in the Right Hand Side (RHS) vector $\widetilde{\mathbf{p}}_1$. Notice that constraints (26.1) together with (26.2) can be rearranged as follows:

$$\begin{pmatrix} \widetilde{\mathbf{B}}_1 & \mathbf{0} \\ \tilde{\mathbf{c}}^T & -1 \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \rho \end{pmatrix} \le \begin{pmatrix} \widetilde{\mathbf{p}}_1 \\ 0 \end{pmatrix} \tag{27}$$

We introduce an auxiliary variable $u' = -1$ and let $\mathbf{x} = (\mathbf{u}, \rho, u')^T$ such that constraint (27) together with constraint (26.3) can be rearranged to the following matrix form:

$$\left(\begin{array}{c} \widetilde{\mathbf{A}} \\ \hdashline \mathbf{B} \end{array}\right) \mathbf{x} = \left(\begin{array}{ccc} \widetilde{\mathbf{B}}_1 & \mathbf{0} & \widetilde{\mathbf{p}}_1 \\ \tilde{\mathbf{c}}^T & -1 & 0 \\ \hdashline \mathbf{B}_2 & \mathbf{0} & \mathbf{p}_2 \end{array}\right) \begin{pmatrix} \mathbf{u} \\ \rho \\ u' \end{pmatrix} \le \mathbf{0}, \tag{28}$$

Finally, we get the standard robust reformulation of (25), which is given by:

$$\min_{\mathbf{x}} \mathbf{h}^T \mathbf{x} \tag{29}$$

$$s.t.\ \widetilde{\mathbf{A}}\mathbf{x} \le \mathbf{0}, \tag{29.1}$$

$$\mathbf{B}\mathbf{x} \le \mathbf{0}, \mathbf{x} \in \mathcal{X}, \tag{29.2}$$

wherein $\mathbf{h} = (\mathbf{0}, 1, 0)^T$. The dimension of $\widetilde{\mathbf{A}}$ is $N \times M$, wherein $N = I_1 + 1$ and $M = J + 2$. The set $\mathcal{X} = \{x_j \mid x_j \in \{0,1\},\ \text{for } j = 1, \cdots, |\mathbf{y}| + |\mathbf{z}| + |\boldsymbol{\gamma}|; x_j \ge 0 \text{ for } j = |\mathbf{y}| + |\mathbf{z}| + |\boldsymbol{\gamma}| + 1, \cdots, J+1; x_j = -1 \text{ for } j = J+2\}$, where $|\boldsymbol{\zeta}|$ represents the dimension of vector $\boldsymbol{\zeta}$.

In problem (29), $\widetilde{\mathbf{A}}$ and $\mathbf{B}$ are the coefficient matrices with and without uncertain parameters respectively. For uncertain matrix $\widetilde{\mathbf{A}}$, its uncertainty set is composed of all the possible realizations of $\widetilde{\mathbf{A}}$. In the following subsections, we will first elaborate the box optimizer where the uncertainty set of $\widetilde{\mathbf{A}}$ is box-shaped. To overcome the over-conservativeness of box optimizer, the ellipsoid optimizer based on the ellipsoid uncertainty set of $\widetilde{\mathbf{A}}$ will be proposed thereafter.

B. The Box Uncertainty Set based Solution

The box uncertainty set is constructed by directly combing the prediction intervals and the link bandwidth variation intervals, which turns out to be a box-shaped uncertainty set $\mathcal{B}$. The i-th row of $\mathcal{B}$ is expressed as:

$$\begin{aligned} \mathcal{B}_i &= \{\tilde{\mathbf{a}}_i \mid \bar{a}_{ij} - \hat{a}_{ij} \le \tilde{a}_{ij} \le \bar{a}_{ij} + \hat{a}_{ij}, j = 1, \cdots, M\} \\ &= \left\{\tilde{\mathbf{a}}_i \mid \mathbf{a}_i^l \le \tilde{\mathbf{a}}_i \le \mathbf{a}_i^u\right\}, \end{aligned} \tag{30}$$

where $\bar{a}_{ij}$ and $\hat{a}_{ij}$ are the mean and standard deviation of $\tilde{a}_{ij}$, respectively. The jth column of $\mathbf{a}_i^l$ and $\mathbf{a}_i^u$ are $\bar{a}_{ij} - \hat{a}_{ij}$ and $\bar{a}_{ij} + \hat{a}_{ij}$ respectively.

The RC of problem (29) based on the uncertainty set $\mathcal{B}$ is the same problem but replacing constraint (29.1) by:

$$\widetilde{\mathbf{A}}\mathbf{x} \le \mathbf{0}, \forall \widetilde{\mathbf{A}} \in \mathcal{B}. \tag{31}$$

From [37], the robust problem with respect to $\mathcal{B}$ is equivalent to the robust problem with respect to $\mathcal{B}_i$, where $\mathcal{B}_i$ is the projection of $\mathcal{B}$ on the subspace of the i-th row of $\widetilde{\mathbf{A}}$ (denoted by $\tilde{\mathbf{a}}_i^T$). Therefore, (31) can be equivalently transformed into:

$$\tilde{\mathbf{a}}_i^T \mathbf{x} \le \mathbf{0}, \forall \tilde{\mathbf{a}}_i \in \mathcal{B}_i, i = 1, \cdots, N. \tag{32}$$

By applying the box uncertainty set in (30), constraint (32) can be transformed into:

$$\left(\max_{\tilde{\mathbf{a}}_i \in \mathcal{B}_i} \tilde{\mathbf{a}}_i^T \mathbf{x}\right) \le \mathbf{0}, \forall i = 1, \cdots, N. \tag{33}$$

By using (30), the inner maximization problem in LHS of (33) is equivalently transformed into the following optimization problem:

$$\max_{\tilde{\mathbf{a}}_i} \mathbf{x}^T \tilde{\mathbf{a}}_i \tag{34}$$

$$s.t.\ \mathbf{a}_i^l \le \tilde{\mathbf{a}}_i \le \mathbf{a}_i^u \tag{34.1}$$

Please note that the optimization variable of problem (34) is $\tilde{\mathbf{a}}_i$ rather than $\mathbf{x}$. The Lagrange dual of (34) is given by

$$\min_{\boldsymbol{\mu}_i, \boldsymbol{\lambda}_i} (\mathbf{a}_i^u)^T \boldsymbol{\mu}_i - (\mathbf{a}_i^l)^T \boldsymbol{\lambda}_i \tag{35}$$

$$s.t.\ \boldsymbol{\mu}_i - \boldsymbol{\lambda}_i = \mathbf{x} \tag{35.1}$$

$$\boldsymbol{\mu}_i \ge \mathbf{0}, \boldsymbol{\lambda}_i \ge \mathbf{0} \tag{35.2}$$

It is important to point out that the minimization operator in (35) can be omitted since it is sufficient (and necessary) that constraint (33) holds if there exists one pair of $(\boldsymbol{\mu}_i, \boldsymbol{\lambda}_i)$ such that $(\mathbf{a}_i^u)^T \boldsymbol{\mu}_i - (\mathbf{a}_i^l)^T \boldsymbol{\lambda}_i \le 0$. Therefore, the RC of problem (25) based on uncertainty set (30) now becomes:

$$\min_{\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\lambda}} \mathbf{h}^T \mathbf{x} \tag{36}$$

$$s.t.\ (\mathbf{a}_i^u)^T \boldsymbol{\mu}_i - (\mathbf{a}_i^l)^T \boldsymbol{\lambda}_i \le \mathbf{0}, \forall i = 1, \cdots, N \tag{36.1}$$

$$\boldsymbol{\mu}_i - \boldsymbol{\lambda}_i = \mathbf{x}, \forall i = 1, \cdots, N \tag{36.2}$$

$$\boldsymbol{\mu}_i \ge \mathbf{0}, \boldsymbol{\lambda}_i \ge \mathbf{0}, \forall i = 1, \cdots, N \tag{36.3}$$

$$\mathbf{B}\mathbf{x} \le \mathbf{0}, \mathbf{x} \in \mathcal{X}, \tag{36.4}$$

where $\boldsymbol{\mu} = \{\boldsymbol{\mu}_i\}_i$, $\boldsymbol{\lambda} = \{\boldsymbol{\lambda}_i\}_i, i = 1, \cdots, N$. Due to the sparsity of its coefficient matrix, the above problem can be efficiently solved by off-the-shelf MIP solvers, such as Gurobi, CPLEX, etc [34], [38].

C. The Ellipsoid Uncertainty Set based Solution

The box optimizer produces excessively conservative solutions since it is based on the box uncertainty set. To overcome the over-conservativeness of the box optimizer and improve its efficiency, we design an ellipsoid optimizer based on the ellipsoid uncertainty set. For one thing, the ellipsoid uncertainty set can provide the same coverage probability as that of the box uncertainty set due to the normality of network traffic demands. Therefore, the ellipsoid optimizer can provide the same robustness as the box optimizer. For another, the ellipsoid uncertainty set has a smaller size since it is essentially a high-dimensional ellipsoid that inscribes in the box uncertainty set. As a result, the ellipsoid optimizer can provide better solutions than the box optimizer.

We denote the expectation and variance-covariance matrix of $\tilde{\mathbf{a}}_i$ as $\mathbf{a}_i$ and $\boldsymbol{\Sigma}_i$, respectively. By using these parameters, we construct the following ellipsoid uncertainty set:

$$\mathcal{E}_i = \left\{\tilde{\mathbf{a}}_i \mid (\tilde{\mathbf{a}}_i - \mathbf{a}_i)^T \boldsymbol{\Sigma}_i^{-1} (\tilde{\mathbf{a}}_i - \mathbf{a}_i) \le 1\right\}, i = 1, \cdots, N \tag{37}$$

Since $\boldsymbol{\Sigma}_i$ is a symmetric positive definite matrix, it can be decomposed as $\boldsymbol{\Sigma}_i = \mathbf{L}_i \mathbf{L}_i^T$ through Cholesky decomposition. Let $\tilde{\mathbf{v}}_i = \tilde{\mathbf{a}}_i - \mathbf{a}_i$, and then $\mathcal{E}_i$ can be expressed as:

$$\mathcal{E}_i = \left\{\tilde{\mathbf{v}}_i + \mathbf{a}_i \mid \tilde{\mathbf{v}}_i^T (\boldsymbol{\Sigma}_i^{-1}) \tilde{\mathbf{v}}_i \le 1\right\} \tag{38}$$

$$= \left\{\tilde{\mathbf{v}}_i + \mathbf{a}_i \mid \tilde{\mathbf{v}}_i^T ((\mathbf{L}_i)^{-1})^T (\mathbf{L}_i)^{-1} \tilde{\mathbf{v}}_i \le 1\right\}. \tag{39}$$

By introducing $\mathbf{Q}_i = (\mathbf{L}_i)^{-1}$ and $\tilde{\mathbf{t}}_i = \mathbf{Q}_i \tilde{\mathbf{v}}_i$, $\mathcal{E}_i$ can be expressed as:

$$\mathcal{E}_i = \left\{ \mathbf{Q}_i^{-1}\tilde{\mathbf{t}}_i + \mathbf{a}_i \mid \left\|\tilde{\mathbf{t}}_i\right\| \leq 1 \right\}, \tag{40}$$

where $\|\cdot\|$ denotes L_2 norm. Applying this ellipsoid uncertainty set, the RC of problem (29) has the same form as (29), except that constraint (29.1) is replaced by

$$\left(\max_{\tilde{\mathbf{a}}_i \in \mathcal{E}_i} \tilde{\mathbf{a}}_i^T \mathbf{x} \right) \leq \mathbf{0}, \forall i = 1, \cdots, N. \tag{41}$$

By using (40), the inner minimization problem in the LHS of (41) is transformed into:

$$\begin{aligned} \max_{\tilde{\mathbf{a}}_i \in \mathcal{E}_i(\theta_i)} \tilde{\mathbf{a}}_i^T \mathbf{x} &= \max_{\|\tilde{\mathbf{t}}_i\| \leq 1} (\mathbf{Q}_i^{-1}\tilde{\mathbf{t}}_i + \mathbf{a}_i)^T \mathbf{x} && (42) \\ &= \max_{\|\tilde{\mathbf{t}}_i\| \leq 1} (\mathbf{Q}_i^{-1}\tilde{\mathbf{t}}_i)^T \mathbf{x} + \mathbf{a}_i^T \mathbf{x} && (43) \\ &= \max_{\|\tilde{\mathbf{t}}_i\| \leq 1} ((\mathbf{Q}_i^{-1})^T \mathbf{x})^T \tilde{\mathbf{t}}_i + \mathbf{a}_i^T \mathbf{x} && (44) \\ &= \left\|(\mathbf{Q}_i^{-1})^T \mathbf{x}\right\| + \mathbf{a}_i^T \mathbf{x}. && (45) \end{aligned}$$

The last equality of the above derivation is based on Cauchy–Schwarz inequality. Based on the above derivation, constraint (41) can be equivalently transformed into:

$$\left\|(\mathbf{Q}_i^{-1})^T \mathbf{x}\right\| + \mathbf{a}_i^T \mathbf{x} \leq 0, \forall i = 1, \cdots, N. \tag{46}$$

Finally, the RC of problem (25) based on the ellipsoid uncertainty set $\mathcal{E}$ is:

$$\begin{aligned} & \min_x \mathbf{h}^T \mathbf{x} && (47) \\ s.t. \ & \left\|\mathbf{Q}_i^T \mathbf{x}\right\| + \mathbf{a}_i^T \mathbf{x} \leq 0, i = 1, \cdots, N, && (47.1) \\ & \mathbf{B}\mathbf{x} \leq \mathbf{p}, \mathbf{x} \in \mathcal{X}, && (47.2) \end{aligned}$$

which is indeed a MISOCP. We can use the last versions of state-of-the-art solvers, such as Gurobi 9.1.1, to solve the above MISOCP since it can now handle MISOCP efficiently [34], [38].

VII. Numerical Results

In this section, we conduct numerical simulations to verify the effectiveness of our proposed HMD framework. We first examine the effectiveness of PIP in producing high-quality prediction intervals by comparing it with existing methods. After that, we demonstrate the performance of the HMD framework through comparison with other benchmarks.

A. Performance Evaluation of PIP

1) Neural Network Structure and Data Preparation

For each slice, we create a PIP by using TensorFlow whose hyper-parameters are listed in Table II. For each PIP, an ensemble of 200 point predictors are constructed to evaluate the model misspecification errors. To estimate the variance of the residual, another point predictor is created. The structures of all these 201 point predictors are the same, which are all two-layer stacked GRU networks as shown in Fig. 2. The internal hidden state of its GRUs is of size 10. We use the widely used Mean Squared Error (MSE) as the loss function and the minibatch size is set to 64. In addition, the learning rate is set to 0.01.

TABLE II
Hyper Parameters of PIP

Parameter	**Value**
Number of NNs ($\mathcal{Z}$)	200
Number of GRU layers of each NNs	2
Size of hidden state	10
Loss functions	MSE
Minibatch size	64
Learning rate	0.01
Historical traffic demands	A5M, B5M [39]
Dataset split	7 : 1.5 : 1.5
Time steps of past observations	80
Time steps ahead	1
Training steps	10000

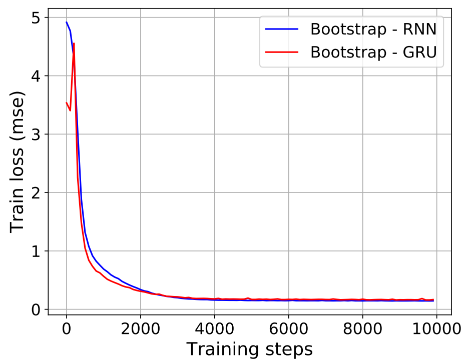


Fig. 4. The training process of the 100th GRU/RNN network of PIP.

We use the Internet traffic time series A5M and B5M as the historical traffic demands of individual slices. These datasets are real traffics that are gathered from the backbone network for every 5 minutes [39]. Since inter-slice reconfigurations are performed at a large time-scale, thus we set the PIP to produce prediction intervals for every one hour. Accordingly, the datasets are resampled with a sampling frequency of 12. Each PIP is trained by 70% of the dataset of the corresponding network slice. The rest of the dataset is equally divided into two parts for evaluation and testing respectively. The number of training steps of each point predictor is 10000. In addition, we use 80 time steps in the past to predict one time step ahead. We compared our approach against the MVE method as well as the RNN-based bootstrap method.

2) Experiment 1 - The Convergence Performance of PIP

Fig. 4 shows the training process of PIP and the bootstrap method implemented with RNN. We plot the training loss of the 100th GRU and RNN network *w.r.t.* training steps. It can be observed that both methods converge with 4000 training steps. In particular, we can see that GRU-based method has a slightly faster convergence rate. This is because that RNN suffers from the *gradient vanishing problem*, which may slow down its training process. In contrast, GRUs exploit internal gates to overcome this defect, thus they have a faster convergence rate. These results suggest that PIP can be implemented as an online predictor under the premise of only a small traffic sample.

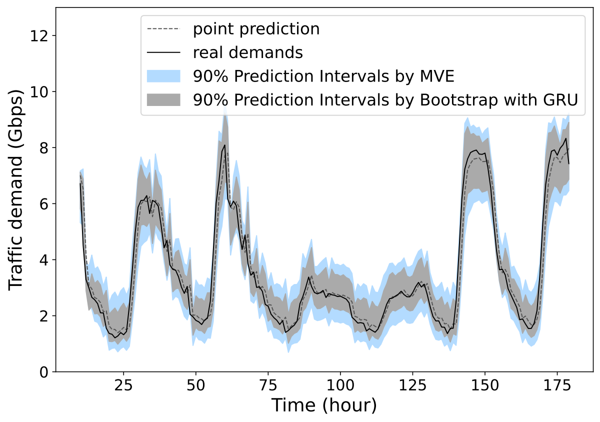


Fig. 5. 90% prediction intervals generated by the bootstrap method with GRU as well as MVE method, whose NMPIWs are 0.41 and 0.64 respectively.

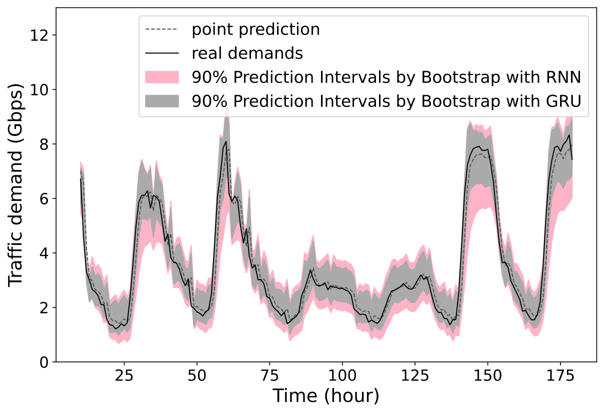


Fig. 6. 90% prediction intervals generated by the bootstrap method with GRU and RNN, whose NMPIWs are 0.41 and 0.51 respectively.

3) *Experiment 2 - Prediction Results on Real Data*

In this experiment, we compare the prediction intervals produced by PIP MVE and RNN-based methods. The prediction intervals with 90% confidence level are produced, which are drawn in Fig. 5 and Fig. 6. Both figures reveal that the width of the prediction intervals produced by PIP is narrower than that of MVE and RNN-based methods. To be precise, the *normalized mean prediction interval width* (NMPIW) of their produced prediction intervals are evaluated, which is defined as [40]:

$$\text{NMPIW} = \frac{1}{n_{test}} \frac{\sum_{t=1}^{n_{test}} (\hat{r}_t^+ - \hat{r}_t^-)}{r_{max} - r_{min}}, \tag{48}$$

where n_{test} is the number of predicted time steps; r_{max} and r_{min} represent the maximum and minimum values of the true targets; $\hat{r}_t^+$ and $\hat{r}_t^-$ are the upper and lower bounds of the prediction interval. In this experiment, the NMPIW of the prediction intervals obtained by PIP, MVE and RNN-based methods are 0.41, 0.64 and 0.51, respectively. It is therefore concluded that under the same confidence level, the prediction interval produced by PIP is the most compact. In addition, we can see that the 90% prediction intervals can cover nearly 100% of the real traffic, while the point predictions become inaccurate when the traffic fluctuates heavily. This result further verifies that point prediction is inadequate in supporting network slice reconfiguration since they are error-prone in high dynamic scenarios. Furthermore, the results show that as the fluctuation of traffic becomes milder, the width of the prediction interval decreases accordingly, and vice versa. Compared with the static traffic interval produced by statistical traffic data [10], a prominent advantage of our proposed PIP is that the width of the produced prediction interval can be automatically adjusted according to the traffic demand variations.

4) *Experiment 3 - Quantitative Assessment of PIP under different Confidence Levels*

To further demonstrate the effectiveness of PIP, we compare the NMPIW of the prediction intervals produced by MVE, RNN and GRU-based (i.e., PIP) methods under different confidence levels in Fig. 7a. We can see that the NMPIWs corresponding to these three methods exhibit an exponential growth *w.r.t.* $(1-\alpha)$. This is the case since the width of the prediction interval is exponential to $(1-\alpha)$ according to (9). Moreover, we can see that the NMPIW of PIP is the lowest and its curve grows much slower than the other two. Since a wide prediction interval leads to a poor solution of slice reconfiguration, thus we use only PIP to implement our HMD framework.

To verify that the ellipsoid uncertainty set has nearly the same coverage probability as the box uncertainty set, we examine the *coverage probability* of these two sets under different confidence levels in Fig. 7b. The coverage probability is defined as:

$$\text{coverage probability} = \frac{1}{n_{test}} \sum_{i=1}^{n_{test}} c_i, \tag{49}$$

where

$$c_i = \begin{cases} 1, r_t \in [\hat{r}_t^-, \hat{r}_t^+] \\ 0, r_t \notin [\hat{r}_t^-, \hat{r}_t^+] \end{cases} \tag{50}$$

in which r_t is the real traffic demand at time t. We can see that the coverage probability of the box uncertainty set is nearly the same as that of the ellipsoid uncertainty set. This is because of the normality property of traffic demands. Furthermore, we find that the coverage probability is much larger than the confidence level $1-\alpha$, which is especially evident at $1-\alpha = 0.1$. Thus, we can conclude that the acquired protection level is much higher than the prespecified confidence level. Moreover, we can observe that the coverage probability decreases with the number of slices. This is reasonable since the occurrence of stochastic events increases with the number of slices, which is hard to predict.

B. Performance Evaluation of HMD Framework

1) *Simulation Settings*

As shown in Fig. 3, the substrate network employed in the simulations is a hierarchical DC-based 5G infrastructure composed of the core DC, aggregation DC, edge DC, and the cloudlets. The topology of each DC is the fat-tree [41], which is widely used in DC deployment. Specifically, the core and the aggregation DCs are 2-ary and 1-ary fat-tree respectively, while the edge DCs are composed of one gateway connected with two edge servers. As a result, the substrate network is

TABLE III
PARAMETERS OF THE SUBSTRATE NETWORK

Parameter	**Value**
Number of nodes	57
Number of links	111
Bandwidth of links $\bar{B}_{ij}^{S}$	$U[10, 200]$ Mbps
Bandwidth perturbation ω	0.25
β_b, β_c	0.5, 0.1
C_i	$U[40, 200]$ vCPU
P_{port}^{i}	$U[0.1, 0.5]$ kW
P_{max}^{i}	$U[100, 120]$ kW
P_{idle}^{i}	$U[5, 15]$ kW

composed of 57 nodes and 111 directed links. The detailed parameters of the simulation are summarized in Table III. To emulate the variations of wireless backhauls, the bandwidth of wireless link $(i, j) \in E$ is set to a uniformly distributed random variable $B_{ij}^{S} \sim U\left[\bar{B}_{ij}^{S}(1-\omega), \bar{B}_{ij}^{S}(1+\omega)\right]$, where ω represents the level of link bandwidth perturbation.

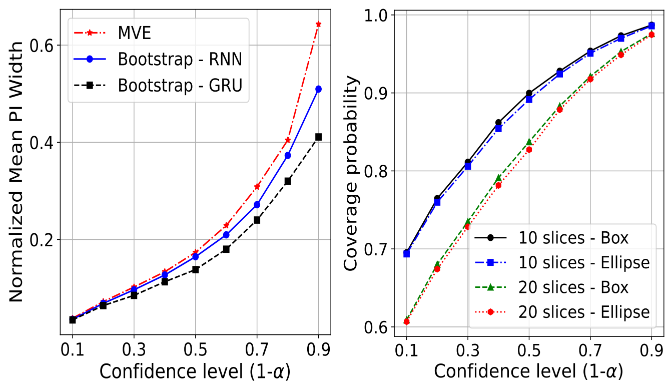
In the considered system, the VNF set of the system contains 8 different types of VNFs. Each server can provide 2 to 4 types of VNFs. In the simulation, network slices are randomly generated from the 5 types of slices, whose SFCs are defined in Table III of [5]. For each slice, a corresponding PIP is constructed and is well trained by the real traffic data.

As we mentioned earlier, pure data-driven methods are data-intensive and require numerous costly trial-and-error interactions. Due to a lack of sufficient data, we are unable to implement data-driven methods. Consequently, the following three algorithms are used as our comparison references:

- *Model-based with 1-std uncertainty set* (Model-based-Std): In this algorithm, the uncertainty set of the optimization problem is not produced by prediction. Instead, it is generated by the arithmetic mean and standard deviation of historical demands [10];
- *Model-based with perfect hindsight parameters* (Model-based-PH): In this algorithm, the optimization problem is solved with perfect hindsight knowledge on all uncertain parameters. This is the ideal case;
- *HDM with point prediction* (HMD-Point): which is an HMD framework implemented by point predictors.

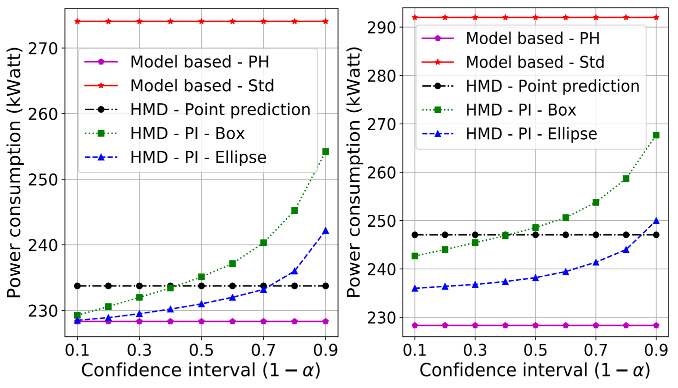
2) *Experiment 4 - The performance of HMD framework in different scenarios*

This experiment aims to evaluate the performance of the HMD framework under different scenarios. We compare the performance of our proposed framework under different confidence levels and link perturbation levels. For high and medium uncertainty scenarios, the link perturbation parameter ω is set to 0.25 and 0.65 respectively. We randomly generate 20 network slices. The results of power consumption versus confidence level under different levels of wireless link perturbations are shown in Fig. 8. We notice that, except for the Model-based-PH algorithm, the power consumption of all algorithms is about 4% higher than that in the medium uncertainty scenario. This is reasonable since more resources should be "reserved" to protect the slices' SLAs against the



(a) NMPIW (b) Coverage probability

Fig. 7. NMPIW of the prediction intervals and the coverage probability of the uncertainty sets.



(a) Medium uncertainty ($\omega = 0.25$) (b) High uncertainty ($\omega = 0.65$)

Fig. 8. Power consumption *vs.* confidence level $(1-\alpha)$ under different levels of wireless link perturbations.

high uncertainty of the wireless links. Furthermore, we can see that the power consumption of the box optimizer and ellipse optimizer increase with the confidence level, which coincides with the trend of the curves in Fig. 7a. This is because in robust optimization, the optimal value of the objective function increases with the width of the uncertainty set. For the same reason, we can observe that the power consumption of the ellipsoid optimizer is about 5% lower than that of the box optimizer. Moreover, it is worth mentioning that the power consumption of Model-based-Std is significantly greater than that of our proposed algorithms.

3) *Experiment 5 - Performance of HMD under different numbers of network slices*

In this experiment, we examine the performance of the HMD framework under different numbers of network slices. The wireless link perturbation parameter is set as $\omega = 0.25$, while the confidence level is set to 90%. The results are shown in Fig. 9. Since the power consumption of Model-based-Std is much greater than the others, the curves of other algorithms will be "squeezed" and become almost indistinguishable. Thus, we omit the curve of Model-based-Std in the figures of this experiment and subsequent simulations. As is expected, we can observe that the power consumption of all algorithms

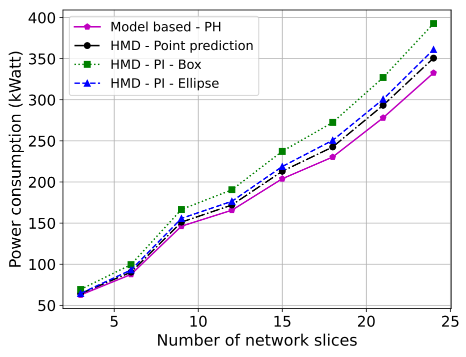


Fig. 9. Power consumption *vs.* the number of network slices.

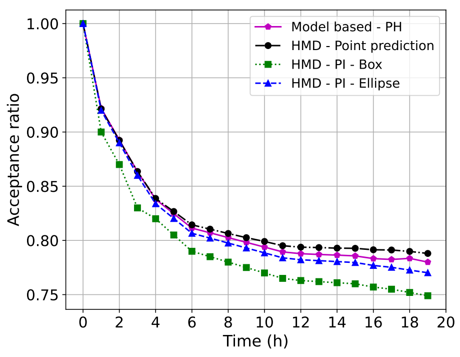


Fig. 10. Acceptance ratio of requests in the full-load scenario over time.

increases approximately linearly with the number of network slices. In addition, the power consumption of the box optimizer and the ellipsoid optimizer is about 10% and 3% higher than that of HMD-Point-Prediction respectively. We would like to highlight that this does not indicate the superior performance of HMD-Point-Prediction but rather demonstrates the superiority of our proposed box/ellipsoid optimizer. Indeed, although the curve of HMD-point-prediction is lower than those of box/ellipsoid optimizer, it is indeed inapplicable in slice reconfiguration since point predictors become inaccurate in high traffic fluctuations as is shown in Fig. 5 and Fig. 6. In contrast, the results demonstrate that our proposed box/ellipsoid optimizer can satisfy the traffic demands of future with nearly 100% probability, with only 3% to 10% additional power consumption.

4) *Experiment 6 - Acceptance ratio of HMD framework in full-load scenario*

Finally, we evaluate the performance of the HMD framework in terms of acceptance ratio in the full-load scenario. In this experiment, the link variation ratio and the confidence ratio are set as $\omega = 0.25$ and $(1-\alpha) = 90\%$ respectively. In real scenarios, there usually exist permanent and/or temporary slices [10]. To emulate the full-load scenario, 20 permanent slices that follow the A5M/B5M traffic tracks are deployed in the substrate network, while a number of temporary slices with fixed traffic demands arrive randomly. The arrival of the temporary slice requests follows a Poisson process with an arrival rate of 10/hour and their duration follows an exponential distribution of 10 minutes on average. In addition, we assume that the traffic demand of the temporary slices is uniformly distributed in $[0.5, 2.0]$ Gbps. The Acceptance Ratio (AR) of the requests is plotted in Fig. 10.

We can observe that the AR of all algorithms decreases rapidly at the beginning of the simulation, and it becomes flat after a period of time. This is because that when the simulation starts, a large number of slice requests arrive at the saturated network will lead to a sharp drop in the AR. When the arrival and the rejection reach an equilibrium, the curve of AR will become flat. In addition, we can also observe that the AR of the ellipse optimizer is about 10% higher than that of the box optimizer. Therefore, it is worth choosing the complex ellipse optimizer rather than the box optimizer to enjoy such performance gain, especially in small-scale networks.

5) *Experiment 7 - Performance evaluation for large-scale networks*

Due to the NP-hardness problem (36), there do not exist exact algorithms that can solve it in polynomial time. Indeed, our proposed HMD-box/ellipsoid algorithms provide a theoretical bound for this problem, which is applicable only in small-scale networks. For the substrate network with hundreds of servers, it is unrealistic and unnecessary to pursue an exact solution for practical application. Instead, we can easily design a high-performance heuristic algorithm that provides sub-optimal solutions for large-scale problems. For example, based on the variable neighborhood search (VNS) and local branching, the *metha-heuristic* algorithm proposed in our previous work [11] can be applied to search for a near-optimal solution. We refer to this heuristic algorithm as *VNS-heu*.

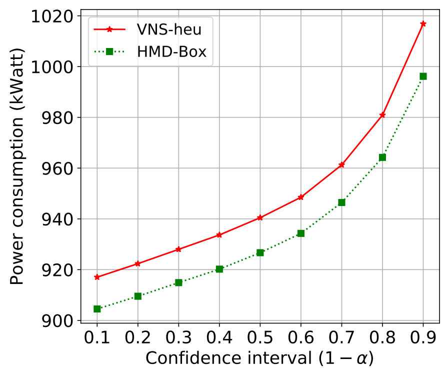


Fig. 11. Power consumption *vs.* confidence level $(1-\alpha)$ under medium uncertainty scenario in the large-scale network.

The performance of VNS-heu and HMD-Box are evaluated in a large-scale substrate network, which is composed of 208 nodes and 768 links. We have compared the power consumption of VNS-heu and the theoretical bound provided by the box optimizer under medium uncertainty scenario in Fig. 11 and Fig. 12. The results show that in the large-scale network, the power consumption of VNS-heu is at most 10% higher than the theoretical bound given by the box optimizer. Therefore, VNS-heu is appreciated in large-scale networks, while in small-scale networks, HMD-box/ellipse is more preferred.

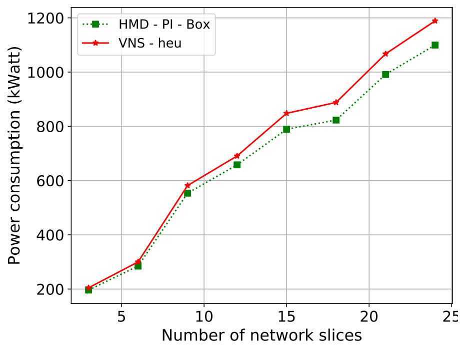


Fig. 12. Power consumption vs. the number of network slices in the large-scale network.

VIII. Conclusions and Future Work

In this paper, we address the problem of inter-slice reconfiguration under demand uncertainty. We have proposed a hybrid model-data driven network slice reconfiguration framework by jointly leveraging prediction interval and robust optimization. We have designed a predictor with controllable prediction accuracy to predict the uncertain traffic demand of the network slice. Thereafter, based on the produced prediction interval, we have exploited robust optimization to make proactive slice reconfiguration. We have compared the performance of our proposed framework with the method based on point predictor as well as the ideal case where the InP has full knowledge about the uncertain parameters. The simulation results have shown that our proposed framework demonstrates superior performance and the performance gap is within 10% compared with the ideal case.

In this work, we focus on the inter-slice reconfiguration at the core network. In the future, we will consider the dynamic network slicing problem in radio access networks. Furthermore, we intend to exploit the stochastic game theory to model the traffic fluctuations of the slices, whereby a distributed learning policy can be trained to perform slice reconfiguration in a cooperative or competitive fashion.

Appendix A
Derivation of the Matrix Form of (24)

We derive the matrix form of problem (24).

A. Decision Variables

The decision variable is denoted by $\mathbf{u} = (\mathbf{y}, \mathbf{z}, \boldsymbol{\gamma}, \mathbf{b})^T$. The node mapping variables is $\mathbf{y} = (\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_{|V|})^T$, in which $\mathbf{y}_i = (\mathbf{y}_{i,1}, \mathbf{y}_{i,2}, \cdots, \mathbf{y}_{i,|\mathcal{K}|})^T$, in which $\mathbf{y}_{i,k} = (y^0_{i,k}, y^1_{i,k}, \cdots, y^{l_k+1}_{i,k})^T$. Therefore, $\mathbf{y}$ is a $|V| \cdot \sum_{k=1}^{\mathcal{K}}(l_k + 2)$-dimensional vector. Similarly, we use an $|E| \cdot \sum_{k=1}^{\mathcal{K}}(l_k + 1)$-dimensional vector $\mathbf{b} = (\mathbf{b}(e_1), \cdots, \mathbf{b}(e_{|\mathcal{K}|}))^T$ to denote the bandwidth allocation variables, in which $\mathbf{b}(e_k) = (\mathbf{b}(e^0_k), \mathbf{b}(e^1_k), \cdots, \mathbf{b}(e^{l_k}_k))^T$, in which $\mathbf{b}(e^m_k) = (b_1(e^m_k), b_2(e^m_k), \cdots, b_{|E|}(e^m_k))^T$. Since each variable $b_l(e^m_k)$ corresponds to an auxiliary variable $z_l(e^m_k)$, thus we use a $|E| \cdot \sum_{k=1}^{\mathcal{K}}(l_k + 1)$-dimensional vector $\mathbf{z}$ to represent all the auxiliary variables $z_l(e^m_k)$.

In addition, we use a $|V|$-dimensional vector $\boldsymbol{\gamma} = (\gamma^1, \cdots, \gamma^{|V|})^T$ to denote the on-off status variables. Therefore, the vector $\mathbf{u}$ is a $2|E| \cdot \sum_{k=1}^{|\mathcal{K}|}(l_k+1) + |V| \cdot \sum_{k=1}^{|\mathcal{K}|}(l_k+3)$-dimensional vector.

B. Constraints

To derive the matrix form of constraint (10), we first introduce the *VNF index matrix* $\boldsymbol{\Delta} = (d^k_{m,j})_{(\sum_{k=1}^{\mathcal{K}}(l_k+2)) \times |\mathcal{F}|}$, where

$$d^k_{m,j} = \begin{cases} 1, & f^k_m \text{ is the } j\text{-th VNF in } \mathcal{F} \\ 0, & \text{otherwise.} \end{cases}$$

And then we introduce the *VNF association matrix* $\boldsymbol{\delta} = (\delta_{i,j})_{|\mathcal{F}| \times |V|}$, where

$$\delta_{i,j} = \begin{cases} 1, & \text{node } j \text{ can provide VNF } f_i \in \mathcal{F} \\ 0, & \text{otherwise.} \end{cases}$$

Therefore, constraint (10) can be represented in the matrix form

$$\mathbf{y} \leq Vec(\boldsymbol{\Delta}\boldsymbol{\delta}),$$

where $Vec(\cdot)$ is the matrix vectorization operator.

The coefficient matrix of constraint (11) is denoted by the $(\sum_{k=1}^{|\mathcal{K}|}(l_k + 2)) \times (|V| \sum_{k=1}^{|\mathcal{K}|}(l_k + 2))$-dimensional matrix $\mathbf{A}_1 = \underbrace{(\mathbf{I}, \mathbf{I}, \cdots, \mathbf{I})}_{|V|}$, where $\mathbf{I}$ is a $\sum_{k=1}^{|\mathcal{K}|}(l_k + 2)$-dimensional identity matrix.

The coefficient matrix of constraint (12) is denoted by $\mathbf{A}_2$, which is a $|V| \cdot \sum_{k=1}^{|\mathcal{K}|}(l_k + 2)$-dimensional diagonal matrix. The ith diagonal element of $\mathbf{A}_2$ is defined as:

$$a^2_{i,i} = \begin{cases} 1, \text{ if } i = S_k, m = 0 \text{ or } i = T_k, m = l_k + 1, \\ 0, \text{otherwise} \end{cases}$$

and the RHS vector of constraint (12) is the $|V| \cdot \sum_{k=1}^{|\mathcal{K}|}(l_k+2)$-dimensional row vector $\mathbf{d}_2 = (\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_{|V|})^T$, in which $\mathbf{h}_i = (\mathbf{h}_{i,1}, \mathbf{h}_{i,2}, \cdots, \mathbf{h}_{i,|\mathcal{K}|})^T$. The kth term of $\mathbf{h}_i$ is

$$\mathbf{h}_{i,k} = (h^0_{i,k}, h^1_{i,k}, \cdots, h^{l_k+2}_{i,k},)^T,$$

where

$$h^m_{i,k} = \begin{cases} 1, & \text{if } i = S_k, m = 0 \text{ or } i = T_k, m = l_k + 1, \\ 0, & \text{otherwise} \end{cases}$$

The coefficient matrix of constraint (13) is denoted by the $(\sum_{k=1}^{|\mathcal{K}|}(l_k + 1)) \times (|E| \cdot (\sum_{k=1}^{|\mathcal{K}|}(l_k + 1)))$-dimensional matrix $\mathbf{A}_3$:

$$\mathbf{A}_3 = \mathbf{diag}(\mathbf{A}^1_3, \cdots, \mathbf{A}^{|\mathcal{K}|}_3),$$

in which $\mathbf{A}^k_3 = \mathbf{diag}(\mathbf{A}^{k,0}_3, \cdots, \mathbf{A}^{k,l_k}_3)$ with $\mathbf{A}^{k,m}_3$ an $|E|$-dimensional all-one row vector. The RHS vector of constraint (13) is:

$$\mathbf{d}_3 = \beta_b(\tilde{r}^1 \mathbf{1}_1, \cdots, \tilde{r}^{|\mathcal{K}|}\mathbf{1}_{|\mathcal{K}|})^T,$$

where $\mathbf{1}_k$ is a $(l_k + 1)$-dimensional all-one column vector.

The coefficient matrix of constraint (14) is denoted by the $|E| \times (|E| \cdot \sum_{k=1}^{\mathcal{K}}(l_k + 1))$-dimensional matrix $\mathbf{A}_4 = (\mathbf{I}_4, \cdots, \mathbf{I}_4)$, in which $\mathbf{I}_4$ is an $|E|$-dimensional identity matrix. The RHS vector of constraint (14) is $\mathbf{d}_4 = (\widetilde{B}_1^S, \cdots, \widetilde{B}_{|E|}^S)^T$.

The coefficient matrix of constraint (15) is denoted by the $|V| \times (|V| \sum_{k=1}^{|\mathcal{K}|}(l_k+2))$-dimensional matrix:

$$\mathbf{A}_5 = \beta_c \cdot \mathbf{diag}(\mathbf{F}, \cdots, \mathbf{F}),$$

in which $\mathbf{F}$ is a $(\sum_{k=1}^{|\mathcal{K}|} l_k + 2)$-dimensional row vector denoted by $\mathbf{F} = (\mathbf{f}_1, \cdots, \mathbf{f}_{|\mathcal{K}|})$. The kth element of $\mathbf{F}$ is a (l_k+2)-dimensional row vector, which is expressed as $\mathbf{f}_k = (\widetilde{r}_k, \cdots, \widetilde{r}_k)$. The RHS vector of constraint (15) is $\mathbf{d}_5 = (C_1, \cdots, C_{|V|})^T$.

For constraint (16), we introduce the following two matrix, whose dimensions are both $(|E| \cdot (\sum_{k=1}^{|\mathcal{K}|}(l_k+1))) \times (2|E| \cdot \sum_{k=1}^{|\mathcal{K}|}(l_k+1) + |V|)$:

$$\mathbf{I}_6^1 = [-\mathbf{A}_6, \mathbf{0}, \epsilon\mathbf{A}_6], \quad \mathbf{I}_6^2 = [\mathbf{A}_6, \mathbf{0}, -L\mathbf{A}_6],$$

in which $\mathbf{A}_6$ is a $|E| \cdot (\sum_{k=1}^{|\mathcal{K}|}(l_k+1))$-dimensional identity matrix, and $\mathbf{0}$ is a $(|E| \cdot (\sum_{k=1}^{|\mathcal{K}|}(l_k+1))) \times |V|$ dimensional zero matrix. The RHS vector of constraint (16) is $\mathbf{0}$.

The coefficient matrix of constraint (17) is denoted by the $(|V| \cdot \sum_{k=1}^{|\mathcal{K}|}(l_k+1)) \times (|V| \cdot \sum_{k=1}^{|\mathcal{K}|}(l_k+2) + |E| \cdot \sum_{k=1}^{|\mathcal{K}|}(l_k+1))$-dimensional matrix $\mathbf{A}_7$:

$$\mathbf{A}_7 = \left(\mathbf{A}_7^1, \mathbf{A}_7^2\right) = \begin{pmatrix} \mathbf{R}_1 & & & \mathbf{T}_1 \\ & \ddots & & \vdots \\ & & \mathbf{R}_{|V|} & \mathbf{T}_{|V|} \end{pmatrix}.$$

$\mathbf{R}_i$ is a $(\sum_{k=1}^{|\mathcal{K}|}(l_k+1)) \times (\sum_{k=1}^{|\mathcal{K}|}(l_k+2))$-dimensional matrix, which can be express as

$$\mathbf{R}_i = \mathbf{diag}(\mathbf{S}_{i,1}, \cdots, \mathbf{S}_{i,|\mathcal{K}|}),$$

in which $\mathbf{S}_{i,k}$ is a $(l_k+1) \times (l_k+2)$-dimensional matrix which is given by $\mathbf{S}_{i,k} = \mathbf{diag}(\mathbf{q}, \cdots, \mathbf{q})$ with

$$\mathbf{q} = \begin{cases} (0,0), & \text{if } i = S_k, m = 0 \text{ or } i = T_k, m = l_k \\ (1,-1), & \text{otherwise.} \end{cases}$$

$\mathbf{T}_i$ is a $(\sum_{k=1}^{|\mathcal{K}|}(l_k+1)) \times (|E| \cdot \sum_{k=1}^{|\mathcal{K}|}(l_k+1))$-dimensional matrix which is given by

$$\mathbf{T}_i = \mathbf{diag}(\mathbf{T}_{i,1}, \cdots, \mathbf{T}_{i,|\mathcal{K}|})$$

in which $\mathbf{T}_{i,k} = \mathbf{diag}(\mathbf{T}_{i,k}^0, \cdots, \mathbf{T}_{i,k}^{l_k})$. The m-th term of $\mathbf{T}_{i,k}$ is given by $\mathbf{T}_{i,k}^m = \mathbf{a}_i$ if $i = T_k, m = l_k$, otherwise $\mathbf{T}_{i,k}^m = -\mathbf{a}_i$, where $\mathbf{a}_i$ is the i-th row of $\mathbf{A}_{adj}$. The RHS vector of constraint (17) is a $|V| \cdot \sum_{k=1}^{|\mathcal{K}|}(1+l_k)$-dimensional column vector that is given by $\mathbf{d}_7 = (\mathbf{g}_1, \mathbf{g}_2, \cdots, \mathbf{g}_{|V|})^T$, in which $\mathbf{g}_i = (\mathbf{g}_{i,1}, \mathbf{g}_{i,2}, \cdots, \mathbf{g}_{i,|\mathcal{K}|})^T$. The k-th term of $\mathbf{g}_i$ is given by $\mathbf{g}_{i,k} = (g_{i,k}^0, g_{i,k}^1, \cdots, g_{i,k}^{l_k})^T$, where

$$g_{i,k}^m = \begin{cases} -1, & \text{if } i = S_k, m = 0 \\ 1, & \text{if } i = T_k, m = l_k \\ 0, & \text{otherwise} \end{cases}$$

For constraint (23), we first express the relationship between N_{port}^i and $z_{ij}(e_k^m)$ in (21) as follows:

$$N_{port}^i = \mathbf{I}_8 \mathbf{A}_8^i \mathbf{z} = \mathbf{I}_8 \cdot \mathbf{diag}(|\mathbf{A}_{adj}^i|, \cdots, |\mathbf{A}_{adj}^i|) \cdot \mathbf{z},$$

where $\mathbf{I}_8$ is an $\sum_{k=1}^{|\mathcal{K}|}(l_k+1)$-dimensional all-one row vector; $\mathbf{A}_8^i$ is a $(\sum_{k=1}^{|\mathcal{K}|}(l_k+1)) \times (|E| \cdot \sum_{k=1}^{|\mathcal{K}|}(l_k+1))$-dimensional diagonal matrix, in which $|\mathbf{A}_{adj}^i|$ is the i-th row of $|\mathbf{A}_{adj}|$. Therefore, $\mathbf{N}_{port} = (N_{port}^1, \cdots, N_{port}^{|V|})^T$ can be represented by

$$\mathbf{N}_{port} = \mathbf{C}_8^1 \mathbf{C}_8^2 \mathbf{z} = \begin{pmatrix} \mathbf{I}_8 & & \\ & \ddots & \\ & & \mathbf{I}_8 \end{pmatrix} \begin{pmatrix} \mathbf{A}_8^1 \\ \vdots \\ \mathbf{A}_8^{|V|} \end{pmatrix} \mathbf{z},$$

where $\mathbf{C}_8^1$ is a $|V| \times (|V| \cdot \sum_{k=1}^{|\mathcal{K}|}(l_k+1))$-dimensional block diagonal matrix, $\mathbf{C}_8^2$ is a $(|V| \cdot \sum_{k=1}^{|\mathcal{K}|}(l_k+1)) \times (|E| \cdot \sum_{k=1}^{|\mathcal{K}|}(l_k+1))$-dimensional matrix. Therefore, constraint (23) can be represented by:

$$\begin{pmatrix} -D\mathbf{A}_8 & \mathbf{E} \\ \zeta\mathbf{A}_8 & -\mathbf{E} \end{pmatrix} \begin{pmatrix} \mathbf{z} \\ \gamma \end{pmatrix} \leq \mathbf{0},$$

where $\mathbf{A}_8 = \mathbf{C}_8^1 \mathbf{C}_8^2$.

Finally, we can obtain the following matrix form for the constraints of problem (24):

$$\begin{cases} \widetilde{\mathbf{B}}_1 \mathbf{u} \leq \widetilde{\mathbf{p}}_1 \\ \mathbf{B}_2 \mathbf{u} \leq \mathbf{p}_2 \end{cases}$$

in which

$$\widetilde{\mathbf{B}}_1 = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & -\mathbf{A}_3 \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{A}_4 \\ \mathbf{A}_5 & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix}, \quad \widetilde{\mathbf{p}}_1 = \begin{pmatrix} -\mathbf{d}_3 \\ \mathbf{d}_4 \\ \mathbf{d}_5 \end{pmatrix},$$

$$\mathbf{B}_2 = \begin{pmatrix} \mathbf{E} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{A}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ -\mathbf{A}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{A}_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ -\mathbf{A}_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\mathbf{A}_6 & \mathbf{0} & \epsilon\mathbf{A}_6 \\ \mathbf{0} & \mathbf{A}_6 & \mathbf{0} & -L\mathbf{A}_6 \\ \mathbf{A}_7^1 & \mathbf{A}_7^2 & \mathbf{0} & \mathbf{0} \\ -\mathbf{A}_7^1 & -\mathbf{A}_7^2 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -D\mathbf{A}_8 & \mathbf{E} & \mathbf{0} \\ \mathbf{0} & \zeta\mathbf{A}_8 & -\mathbf{E} & \mathbf{0} \end{pmatrix}, \quad \mathbf{p}_2 = \begin{pmatrix} Vec(\mathbf{\Delta}\delta) \\ \mathbf{1} \\ -\mathbf{1} \\ \mathbf{d}_2 \\ -\mathbf{d}_2 \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{d}_7 \\ -\mathbf{d}_7 \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix},$$

in which $\widetilde{\mathbf{B}}_1$ and $\widetilde{\mathbf{p}}_1$ are matrices with uncertain parameters, while $\mathbf{B}_2$ and $\mathbf{p}_2$ are deterministic matrices. Constraints (24.2) - (24.5) can be expressed by the set $\mathcal{U} = \{u_j \mid u_j \in \{0,1\}, \text{ for } j = 1, \cdots, |\mathbf{y}| + |\mathbf{z}| + |\boldsymbol{\gamma}|; u_j \geq 0 \text{ for } j = |\mathbf{y}| + |\mathbf{z}| + |\boldsymbol{\gamma}| + 1, \cdots, J\}$.

C. Objective

The coefficient vector of the objective function in (24) can be express as $\tilde{\mathbf{c}} = (\mathbf{c}_y, \mathbf{c}_z, \mathbf{c}_\gamma, \mathbf{0})^T$. The coefficient vector corresponds to $\mathbf{y}$ is $\mathbf{c}_y = (c_{i,k}^{y,m})_{(|V| \cdot \sum_{k=1}^{|\mathcal{K}|}(l_k+2)) \times 1}$, where $c_{i,k}^{y,m} = \beta_c \widetilde{r}^k (P_{max}^i - P_{idle}^i)/C_i$. The coefficient vector corresponds to $\mathbf{z}$ is a $(|E| \cdot \sum_{k=1}^{|\mathcal{K}|}(l_k+1))$-dimensional row vector expressed by $P_{port}^i \mathbf{1}^T \mathbf{C}_7^1 \mathbf{C}_7^2$, where $\mathbf{1}$ is a $|V|$-dimensional all-one column vector. The coefficient of $\boldsymbol{\gamma}$ is $\mathbf{c}_\gamma = (c_\gamma^i)_{|V| \times 1}$, where $c_\gamma^i = P_{idle}^i$.

References

[1] F. Wei, G. Feng, Y. Sun, Y. Wang, and S. Qin, "Proactive network slice reconfiguration by exploiting prediction interval and robust optimization," in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, 2020, pp. 1–6.

[2] S. Zhang, "An Overview of Network Slicing for 5G," *IEEE Wireless Communications*, vol. 26, no. 3, pp. 111–117, Jun. 2019.

[3] 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 Transactions on Networking*, vol. 25, no. 4, pp. 2008–2025, Aug. 2017.

[4] G. Wang, G. Feng, T. Q. S. Quek, S. Qin, R. Wen, and W. Tan, "Reconfiguration in Network Slicing - Optimizing the Profit and Performance," *IEEE Transactions on Network and Service Management*, vol. 16, no. 2, pp. 591–605, 2019.

[5] F. Wei, G. Feng, Y. Sun, Y. Wang, S. Qin, and Y. Liang, "Network slice reconfiguration by exploiting deep reinforcement learning with large action space," *IEEE Transactions on Network and Service Management*, 2020.

[6] N. Salhab, S. E. Falou, R. Rahim, S. E. E. Ayoubi, and R. Langar, "Optimization of the implementation of network slicing in 5G RAN," in *2018 IEEE Middle East and North Africa Communications Conference (MENACOMM)*. IEEE, 2018, pp. 1–6.

[7] R. A. Addad, T. Taleb, H. Flinck, M. Bagaa, and D. Dutra, "Network Slice Mobility in Next Generation Mobile Systems: Challenges and Potential Solutions," *IEEE Network*, vol. 34, no. 1, pp. 84–93, 2020.

[8] A. Gausseran, F. Giroire, B. Jaumard, and J. Moulierac, "Be Scalable and Rescue My Slices during Reconfiguration," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020.

[9] Q. Zhang, F. Liu, and C. Zeng, "Online Adaptive Interference-Aware VNF Deployment and Migration for 5G Network Slice," *IEEE/ACM Transactions on Networking*, vol. 29, no. 5, pp. 2115–2128, Oct. 2021, conference Name: IEEE/ACM Transactions on Networking.

[10] A. Baumgartner, T. Bauschert, A. A. Blzarour, and V. S. Reddy, "Network slice embedding under traffic uncertainties — A light robust approach," in *2017 13th International Conference on Network and Service Management (CNSM)*, Nov. 2017, pp. 1–5.

[11] R. Wen, G. Feng, J. Tang, T. Q. S. Quek, G. Wang, W. Tan, and S. Qin, "On Robustness of Network Slicing for Next-Generation Mobile Networks," *IEEE Transactions on Communications*, vol. 67, no. 1, pp. 430–444, Jan. 2019.

[12] V. Sciancalepore, X. Costa-Perez, and A. Banchs, "RL-NSB: Reinforcement Learning-Based 5G Network Slice Broker," *IEEE/ACM Transactions on Networking*, vol. 27, no. 4, pp. 1543–1557, Aug. 2019.

[13] T. Subramanya and R. Riggio, "Machine learning-driven scaling and placement of virtual network functions at the network edges," in *2019 IEEE Conference on Network Softwarization (NetSoft)*, 2019, pp. 414–422.

[14] A. Khosravi, S. Nahavandi, D. Creighton, and A. F. Atiya, "Comprehensive review of neural network-based prediction intervals and new advances," *IEEE Transactions on Neural Networks*, vol. 22, no. 9, pp. 1341–1356, 2011.

[15] K. Papagiannaki, N. Taft, Z. L. Zhang, and C. Diot, "Long-term forecasting of Internet backbone traffic," *IEEE Transactions on Neural Networks*, vol. 16, no. 5, pp. 1110–1124, 2005.

[16] H. Kukkalli, S. Maheshwari, I. Seskar, and M. Skorupski, "Evaluation of Multi-operator dynamic 5G Network Slicing for Vehicular Emergency Scenarios," in *IFIP Networking 2020 Conference and Workshops, Networking 2020*, 2020, pp. 761–766.

[17] A. Baumgartner, T. Bauschert, A. M. C. A. Koster, and V. S. Reddy, "Optimisation Models for Robust and Survivable Network Slice Design: A Comparative Analysis," in *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, Dec. 2017, pp. 1–7.

[18] M. Yan, G. Feng, J. Zhou, Y. Sun, and Y.-C. Liang, "Intelligent Resource Scheduling for 5G Radio Access Network Slicing," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 8, pp. 7691–7703, Jun. 2019.

[19] C. Qi, Y. Hua, R. Li, Z. Zhao, and H. Zhang, "Deep Reinforcement Learning With Discrete Normalized Advantage Functions for Resource Management in Network Slicing," *IEEE Communications Letters*, vol. 23, no. 8, pp. 1337–1341, Aug. 2019.

[20] N. Van Huynh, D. T. Hoang, D. N. Nguyen, and E. Dutkiewicz, "Real-Time Network Slicing with Uncertain Demand: A Deep Learning Approach," in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, 2019, pp. 1–6.

[21] E. H. Bouzidi, A. Outtagarts, A. Hebbar, R. Langar, and R. Boutaba, "Online based learning for predictive end-to-end network slicing in 5g networks," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020, pp. 1–7.

[22] M. Mechtri, C. Ghribi, O. Soualah, and D. Zeghlache, "NFV Orchestration Framework Addressing SFC Challenges," *IEEE Communications Magazine*, vol. 55, no. 6, pp. 16–23, 2017.

[23] D. A. Nix and A. S. Weigend, "Estimating the mean and variance of the target probability distribution," in *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, vol. 1, 1994, pp. 55–60 vol.1.

[24] U. Siddique, H. Tabassum, E. Hossain, and D. I. Kim, "Wireless backhauling of 5g small cells: challenges and solution approaches," *IEEE Wireless Communications*, vol. 22, no. 5, pp. 22–31, 2015.

[25] Z. Lu, M. Li, and W. Zhao, "Normality of ethernet traffic at large time scales," *Mathematical Problems in Engineering*, vol. 2013, 2013.

[26] Z. Xie, R. Li, and H. Hu, "Interval prediction for time series based on lstm and mixed gaussian distribution," in *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2020, pp. 2648–2655.

[27] J. Lu, J. Ding, X. Dai, and T. Chai, "Ensemble stochastic configuration networks for estimating prediction intervals: A simultaneous robust training algorithm and its application," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 12, pp. 5426–5440, 2020.

[28] S. Yang, X. Yu, and Y. Zhou, "Lstm and gru neural network performance comparison study: Taking yelp review dataset as an example," in *2020 International Workshop on Electronic Communication and Artificial Intelligence (IWECAI)*, 2020, pp. 98–101.

[29] L. Pan and D. N. Politis, "Bootstrap prediction intervals for linear, nonlinear and nonparametric autoregressions," *Journal of Statistical Planning and Inference*, vol. 177, pp. 1–27, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S037837581400175X

[30] J. Y. Hwang, L. Nkenyereye, N. M. Sung, J. H. Kim, and J. S. Song, "IoT service slicing and task offloading for edge computing," *IEEE Internet of Things Journal*, vol. 4662, no. c, pp. 1–22, 2021.

[31] J. Pei, P. Hong, K. Xue, and D. Li, "Efficiently embedding service function chains with dynamic virtual network function placement in geo-distributed cloud system," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 10, pp. 2179–2192, 2019.

[32] M. Ghaznavi, N. Shahriar, R. Ahmed, and R. Boutaba, "Service function chaining simplified," *arXiv preprint arXiv:1601.00751*, 2016. [Online]. Available: https://arxiv.org/pdf/1601.00751.pdf

[33] A. Panda, S. Han, K. Jang, M. Walls, S. Ratnasamy, and S. Shenker, "Netbricks: Taking the v out of NFV," in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*. Savannah, GA: USENIX Association, Nov. 2016, pp. 203–216. [Online]. Available: https://www.usenix.org/conference/osdi16/technical-sessions/presentation/panda

[34] Gurobi Optimization Inc., "Gurobi Optimizer Reference Manual," *www.gurobi.com*, 2021.

[35] N. Zhang, Y. F. Liu, H. Farmanbar, T. H. Chang, M. Hong, and Z. Q. Luo, "Network slicing for service-oriented networks under resource constraints," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2512–2521, Nov. 2017.

[36] M. Dayarathna, Y. Wen, and R. Fan, "Data center energy consumption modeling: A survey," *IEEE Communications Surveys and Tutorials*, 2016.

[37] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski, *Robust optimization*. Princeton University Press, 2009, vol. 28.

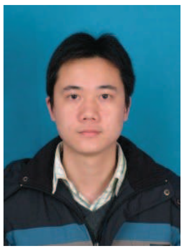
[38] R. Mínguez and V. Casero-Alonso, "Robust solutions of uncertain mixed-integer linear programs using decomposition techniques," 2017.

[39] P. Cortez, M. Rio, M. Rocha, and P. Sousa, "Multi-scale internet traffic forecasting using neural networks and time series methods," *Expert Systems*, vol. 29, no. 2, pp. 143–155, 2012.

[40] R. Ak, V. Vitelli, and E. Zio, "An interval-valued neural network approach for uncertainty quantification in short-term wind speed prediction," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 11, pp. 2787–2800, 2015.

[41] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," in *Proceedings of the ACM SIGCOMM 2008 Conference on Data Communication*, ser. SIGCOMM '08. New York, NY, USA: Association for Computing Machinery, 2008, p. 63–74. [Online]. Available: https://doi.org/10.1145/1402958.1402967

Fengsheng Wei received the B.E. and M.E. degree in School of Communication and Information Engineering at University of Electronic Science and Technology of China (UESTC) in 2012 and 2016, respectively. He is now pursuing his Ph.D. degree in School of Communication and Information Engineering at University of Electronic Science and Technology of China (UESTC). His current research interests include next generation mobile communication systems, machine learning and network slicing in mobile networks.



Shuang Qin received the B.E. degree in Electronic Information Science and Technology, and the Ph.D degree in Communication and Information System from University of Electronic Science and Technology of China (UESTC), in 2006 and 2012, respectively. He is currently an associate professor with National Key Laboratory of Science and Technology on Communications in UESTC. His research interests include cooperative communication in wireless networks, data transmission in opportunistic networks and green communication in heterogeneous networks.

Gang Feng received his BEng and MEng degrees in Electronic Engineering from the University of Electronic Science and Technology of China (UESTC), in 1986 and 1989, respectively, and the Ph.D. degrees in Information Engineering from The Chinese University of Hong Kong in 1998. He joined the School of Electric and Electronic Engineering, Nanyang Technological University in December 2000 as an assistant professor and was promoted as an associate professor in October 2005. At present he is a professor with the National Laboratory of Communications, University of Electronic Science and Technology of China. Dr. Feng has extensive research experience and has published widely in computer networking and wireless networking research. His research interests include resource management in wireless networks, next generation cellular networks, etc. Dr. Feng is a senior member of IEEE.

Yao Sun is currently a Lecturer with James Watt School of Engineering, the University of Glasgow, Glasgow, UK. Dr. Sun has extensive research experience in wireless communication area. He has won the IEEE Communication Society of TAOS Best Paper Award in 2019 ICC. He has been the guest editor for special issues of several international journals. He has been served as TPC Chair for UCET 2021, and TPC member for number of international conferences, including ICC 2022, VTC spring 2022, GLOBECOM 2020, WCNC 2019, ICCT 2019. His research interests include intelligent wireless networking, network slicing, blockchain system, internet of things and resource management in mobile networks. Dr. Sun is a senior member of IEEE.

Jian Wang received the B.E. degree in School of Communication and Information Engineering at University of Electronic Science and Technology of China (UESTC) in 2019. He is now pursuing his Ph.D. degree in School of Communication and Information Engineering at University of Electronic Science and Technology of China (UESTC). His current research interests include B5G/6G communication systems, deep leaning and ad hoc networks.

Ying-Chang Liang (Fellow, IEEE) was a Professor with The University of Sydney, Australia, a Principal Scientist and a Technical Advisor with the Institute for Infocomm Research, Singapore, and a Visiting Scholar with Stanford University, USA. He is currently a Professor with the University of Electronic Science and Technology of China, China, where he also leads the Center for Intelligent Networking and Communications and serves as the Deputy Director of the Artificial Intelligence Research Institute. His research interests include wireless networking and communications, cognitive radio, symbiotic networks, dynamic spectrum access, the Internet-of-Things, artificial intelligence, and machine learning techniques. Dr. Liang is a Foreign Member of Academia Europaea. He received the Prestigious Engineering Achievement Award from The Institution of Engineers, Singapore, in 2007, the Outstanding Contribution Appreciation Award from the IEEE Standards Association, in 2011, and the Recognition Award from the IEEE Communications Society Technical Committee on Cognitive Networks, in 2018. He was a recipient of numerous article awards, including the IEEE Jack Neubauer Memorial Award, in 2014, and the IEEE Communications Society APB Outstanding Paper Award, in 2012. He has been recognized by Thomson Reuters (now Clarivate Analytics) as a Highly Cited Researcher since 2014. He was the Chair of the IEEE Communications Society Technical Committee on Cognitive Networks, and served as the TPC Chair and the Executive Co-Chair of the IEEE Globecom'17. He is the Founding Editor-in-Chief of IEEE Journal on Selected Areas in Communications: Cognitive Radio Series, and the Key Founder and also the Editor-in-Chief of IEEE Transactions on Cognitive Communications and Networking. He is also serving as the Associate Editor-in-Chief for China Communications. He served as a Guest/Associate Editor for IEEE Transactions on Wireless Communications, IEEE Journal of Sele- cted Areas in Communications, IEEE Signal Processing Magazine, IEEE Transactions on Vehicular Technology, and IEEE Transactions on Signal and Information Processing Over Network. He was also the Associate Editor-in-Chief of the Journal on Random Matrices: Theory and Applications (World Scientific). He was a Distinguished Lecturer of the IEEE Communications Society and the IEEE Vehicular Technology Society.