Hierarchical Energy Efficient Mobile Edge Computing in IoT Networks

Qun Wang, *Student Member, IEEE,* Le Thanh Tan, *Member, IEEE,*, Rose Qingyang Hu, *Fellow, IEEE*, and Yi Qian *Fellow, IEEE*

***Abstract*—The ever growing demand of Internet of Things (IoT) imposes great challenges in the existing cellular systems and calls for novel approaches for wireless network design. In this paper, we develop a joint energy and computation optimization paradigm in an IoT network. The tasks collected at local IoT devices can be computed at hierarchical mobile edge computing facilities. Both non-orthogonal multiple access (NOMA) and frequency division multiple access (FDMA) are used for computation offloading. The system model considers both long-term and short-term system behaviors and makes the best decision for energy consumption and computation efficiency. The Long Short Term Memory (LSTM) network is applied to predict the long-term workload, based on which the number of active process units in the edge layer is optimized. In the short-term model, resource optimization problem is formulated. Due to the dynamic arrival workload and non-convex features of the problem, the Lyapunov optimization approach and SCALE method are applied to solve this problem. Simulation results show that the proposed scheme can significantly improve the delay and energy consumption performance.**

***Index Terms*—IoT, Mobile Edge Computing, NOMA, Long Short Term Memory, Machine Learning, Offloading Optimization, Hierarchical Edge Computing, Energy Efficiency.**

I. Introduction

Ultra-dense Internet of Things (IoT) has been greatly facilitating the smart environments (e.g. smart city, smart home, etc.) and diverse new applications such as AR/VR and autonomous driving. An IoT network normally connects billions of resource-limited sensor devices (e.g. mobile devices, sensors, and wearable computing devices) via cellular networks [1]. These small IoT sensor devices can collect computation-intensive and delay-sensitive tasks that need to be processed timely [2]. Mobile Cloud Computing (MCC) can help local IoT tasks to process computation-intensive tasks, which are offloaded from local IoT devices to the MCC [4]. However, the communication bandwidth consumption and time delay caused by transferring data to the remote cloud server can hamper the quality delivery for latency-sensitive mission-critical tasks. Towards that end, mobile edge computing (MEC) moves the processing/storage units from the cloud to the nearby edge nodes so that IoT collected tasks can be offloaded to edge nodes that are much closer. By doing so, both energy consumption and delay performance can be improved [5]–[9]. In this paper, a flexible hierarchical edge computing architecture is used, in which edge nodes and cloud servers with diverse power and computation capabilities form two tiers to best serve the end user needs. In reality, User Equipments (UEs) can serve as edge nodes for the nearby IoT devices, which may have limited power and computation capabilities [10]–[13]. In this new paradigm, it is of crucial importance to design efficient resource allocation and task offloading schemes to realize the desired performance gains.

Orthogonal multiple access (OMA) based resource allocation and task offloading schemes have been extensively studied in mobile edge computing [16]–[20]. In [16], a weighted sum user computation efficiency optimization based on time division multiple access (TDMA) was proposed by combining local computing and data offloading. In [17], the maximal delay of the mobile devices was minimized by jointly optimizing sub-carrier and power allocation in MEC networks with orthogonal frequency division multiple access (OFDMA). Kim *et al.* [18] proposed a single-leader-multi-user Stackelberg game model to optimize the energy efficiency and computation capacity of both the mobile users and the edge cloud. In OFDMA-enabled cloud radio access network (C-RAN) with an integrated MEC server, the joint sub-carrier power allocation and tasks partition problem were studied to minimize the user delay in [19]. In [20], the resource allocation for TDMA and OFDMA based multiuser MEC systems has been studied to minimize the weighted sum of mobile energy consumption. It shows that the power allocation has a threshold-based structure with respect to a derived offloading priority. All these works showed that the combination of offloading and local computation outperforms the model that only considers the offloading process.

As an effective scheme to enhance user connectivity and spectral efficiency, non-orthogonal multiple access (NOMA) technique has received much research attention lately. Applying NOMA in MEC can further improve the computational performance and user connectivity in ultra-dense IoT networks [21]–[24]. The successive interference cancellation (SIC) order and computation resource allocation have been jointly optimized in [21], which can minimize the maximum task execution latency for IoT devices under the limitation of computational resources. Sun *et al.* [22] proposed the NOMA communication method with the wireless energy supply for the IoT system. To maximize the harvesting power, the NOMA cognitive radio network with simultaneous wireless informa-

Q. Wang, L. T. Tan, and R. Q. Hu are with the Department of Electrical and Computer Engineering, Utah State University, Logan, Utah 84322-4120, USA. Y. Qian is with the Department of Electrical and Computer Engineering, University of Nebraska-Lincoln, Omaha, NE, USA. Emails: claudqunwang@ieee.org, {tan.le, rose.hu}@usu.edu, yqian@unl.edu.

tion and power transfer was considered in [23]. Pan *et al.* [24] studied the MEC system, which exploits the NOMA for the computational task uploading and results downloading. By optimizing the transmit powers, transmission time allocation, and task offloading partitions, the minimization of total energy consumption was achieved by this work. It was demonstrated that NOMA method can significantly improve energy efficiency compared with OMA method.

In real applications, the users' behavior and environment conditions change dynamically. Moreover, from the measurements in [25], the user traffic behavior changes with some predictable patterns, for example, the number of network activities during night is reduced significantly. The dynamic resource allocation method is needed to capture the dynamics in the networks. Mao *et al.* [26] developed an online joint radio and computational management algorithm for multi user MEC systems, which aims for minimizing the long-term average weighted sum power consumption of the mobile devices and the MEC server. Lyu *et al.* [27] designed a perturbed Lyapunov function to stochastically maximize a network utility balancing throughput and fairness.

In this paper, a hierarchical communication and computation framework for jointly optimizing energy consumption and computation rate is proposed. The hierarchical framework consists of three layers, i.e. sensor layer, edge layer, and cloud server layer. The contributions of this paper are multi-fold.

- The accumulated computing power minimization and computing rate maximization trade-off optimization problem is formulated in this paper. We develop a prediction-based edge node turning on/off algorithm based on the long-term data dynamics to reduce the system operating cost, while we devise the dynamic resource allocation algorithm based on shot-term data dynamics.
- The LSTM network [28] for arrival tasks prediction mode is applied in long-term process unit status decision operation. Furthermore, the real-life user data are employed to test and to validate our proposed algorithms.
- The optimization of offloading transmit power and local processing speed is determined based on Lyapunov optimization method. The close form of solutions are also given.
- Finally, extensive numerical results are presented for characterizing the performance of the proposed algorithms and the cost reduction by using the optimal parameter configuration, which is found for the computational allocations at each edge node.

The paper is organized as follows. In Section II, the system model and the formulated problem are presented. Section III describes the hierarchical framework for jointly optimizing energy consumption and communication delay. Section IV discusses the workload prediction based on LSTM method. Section V presents the optimization of the local process speed and offloading transmit power for different cases. Section VI shows the performance results, followed by the concluding remarks in Section VII.

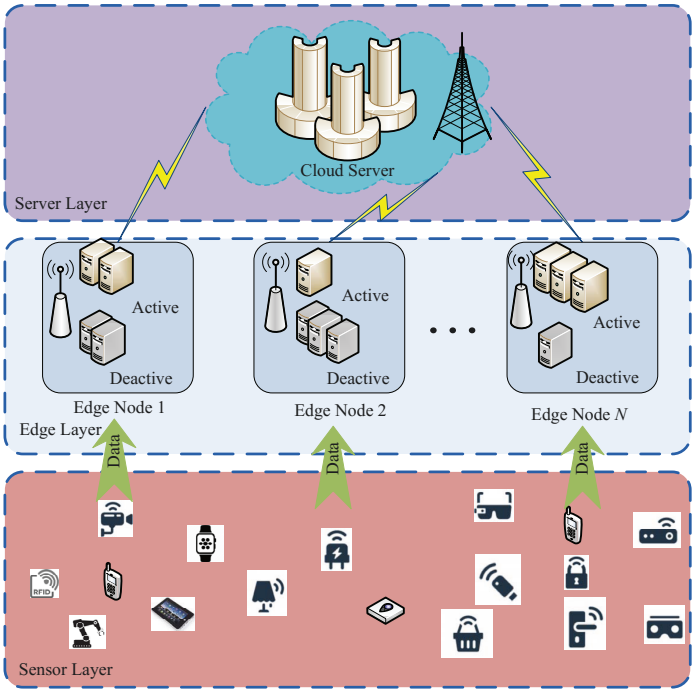

Fig. 1. System model for the three-layer IoT network.

TABLE I
LIST OF SYMBOLS

Symbol	Definition
$A_n(t)$	The workload arriving at edge node.
$Q_n(t)$	The buffer size.
$R_n^{tot}(t)$	The total processing rate of edge node n.
$E_n^{tot}(t)$	The total energy cost of edge node n.
$C_{m,n}$	The number of computation cycles to compute 1 bit of data.
$f_{m,n}(t)$	The computing rate (cycles per second).
$g_n(t)$	The channel gain of edge node n.
$p_n(t)$	The transmit power of edge node n.
ζ	The amplifier coefficient.
$p_n(t)$	The transmission power consumption.
p_r	The constant circuit power.

II. System Model

We consider a three-layer IoT network as described in Fig. 1. The first layer is IoT senor layer, which consists of different IoT sensor devices such as smartphones, environmental sensors, and wearable devices. The second layer is edge layer consisting of mobile edge nodes, while the third layer is the server layer consisting of centralized or cloud servers. All the sensor devices are deployed around randomly distributed edge nodes. IoT sensor devices keep collecting and uploading the data to their associated edge nodes for data processing. There are N edge nodes in the system, which provide the data processing service for the IoT devices. After the massive raw data is received, each edge node can choose to process the data locally, or to offload the data to the more powerful cloud server, or a combination of both. We furthermore assume that each edge node has M processing units (PUs), which can be

turned on or turned off individually based on needs.

Let $A_n(t)$ denote the workload arriving at edge node n at time t. Note that the computing workload at each edge node dynamically changes from time to time but with a predictable pattern in many cases. We allow each edge node adaptively turn on/turn off a subset or all of its PUs to save energy. The operation is done based on the prediction of traffic patterns in a relatively long-term scale.

At t, $A_n(t)$ bits arrive at edge node n from the connected IoT sensor devices. The size of the buffered data at edge node n becomes

$$Q_n(t+1) = \max\{Q_n(t) + A_n(t) - R_n^{tot}(t)\tau, 0\}, \quad (1)$$

where $Q_n(t)$ is the buffer size at t, $R_n^{tot}(t)$ is the total processing rate of edge node n at time t, which include both local processing rate and the cloud processing rate achieved through offloading. We consider a partial offloading for each edge node so it can decide how to partition workload between itself and the cloud server. Table I lists the symbols and their definitions.

A. Local Processing Mode

At t, edge node n computes the workload at the buffer. In particular, the workload for local processing at node n is furthermore divided into M parts. Let $C_{m,n}$ be the number of computation cycles needed to compute one bit of data at $PU(m,n)$. Each PU can compute the data in the entire transmission duration. Let $f_{m,n}(t)$ be the computing rate (cycles per second) at t for $PU(m,n)$ and $s_{m,n}$ represent the status of $PU(m,n)$, where $s_{m,n} = 1$ if $PU(m,n)$ is active and $s_{m,n} = 0$ otherwise. Therefore, the local computing rate of edge node n is calculated as

$$r_n^{local}(t) = \sum_{m=1}^{M} \frac{f_{m,n}(t)}{C_{m,n}} s_{m,n}(t). \quad (2)$$

The energy consumption of local computing is expressed as

$$e_n^{local}(t) = \sum_{m=1}^{M} [\epsilon_{m,n} f_{m,n}^3(t) s_{m,n}(t)\tau + p_{m,n}^{idle} s_{m,n}(t)\tau], \quad (3)$$

where $\epsilon_{m,n}$ is the energy efficiency coefficient for an active $PU(m,n)$ and $p_{m,n}^{idle}$ is the energy consumption of an idle $PU(m,n)$. τ is the duration of each time slot.

B. Data Offloading Mode

In the partial offloading, part of the data in each edge node can be offloaded to the cloud server. Two different offloading schemes are considered, i.e. FDMA and NOMA.

1) FDMA based offloading: Assume the total channel bandwidth between edge nodes and the cloud server is W, which is equally partitioned among N edge nodes by using FDMA. So the bandwidth of each channel is $B = \frac{W}{N}$. Let $g_n(t)$ and $p_n(t)$ represent the channel gain and transmit power for edge node n, respectively. The offloading rate for node n under the FDMA method can be expressed as

$$r_n^{off}(t) = B\log_2(1 + \frac{p_n(t)g_n^2(t)}{\sigma_n^2}). \quad (4)$$

The corresponding energy consumption is

$$e_n^{off}(t) = (\zeta p_n(t) + p_r)\tau, \quad (5)$$

where ζ is the amplifier coefficient. $p_n(t)$ is the transmission power consumption and p_r is the constant circuit power.

2) NOMA based offloading: In NOMA, each edge node pairs with another edge node for transmission. For that, at time t, all the edge nodes are firstly ranked by their channel quality, i.e. $g_1(t) \le g_2(t) \le \cdots \le g_N(t)$. Then, NOMA groups are formed according to the following rules. Node 1 pairs with node $K+1$, node 2 pairs with node $K+2$, $\cdots$, node K pairs with node N, where $K = N/2$. The two nodes in the same NOMA group can offload workload to the cloud server simultaneously on the same radio resource. Successive interference cancellation (SIC) technique is applied at the cloud server to decode the signals for each node [29]. Specifically, let $g_n(t), g_k(t), p_n(t)$, and $p_k(t)$ respectively represent the channel gains and transmit powers for both strong node n and weak node k in the same group, where $g_n(t) \ge g_k(t)$. The cloud server first decodes the signal of the strong node n, then subtracts the decoded signal of node n from the composite received signal and proceeds to decode the signal of the weak node k. When decoding the signal from node n, the signal from node k stays as interference. Compared with FDMA, the bandwidth allocated for each NOMA group is $2B$. Correspondingly, the offloading rates $(r_n^{off}(t), r_k^{off}(t))$ for the strong and weak nodes (n,k) can be expressed as

$$r_n^{off}(t) = 2B\log_2(1 + \frac{p_n(t)g_n^2(t)}{p_k(t)g_k^2(t) + \sigma_n^2}), \quad \text{strong node,} \quad (6)$$

$$r_k^{off}(t) = 2B\log_2(1 + \frac{p_k(t)g_k^2(t)}{\sigma_k^2}), \quad \text{weak node,} \quad (7)$$

where σ_n^2 and σ_k^2 are the noise powers at the strong and weak nodes, respectively. The corresponding total energy consumption for nodes (n,k) using NOMA method is given as

$$e_{n,k}^{off}(t) = (\zeta_n p_n(t) + \zeta_k p_k(t) + 2p_r)\,\tau, \quad (8)$$

where (ζ_n, ζ_k) are the amplifier coefficients. The first two terms represent the transmission power consumption, while the third is the constant circuit power consumption. Moreover, p_r is assumed to be the same for all the edge nodes.

III. Problem Formulation

We aim to jointly design the data offloading and local computing in this work. The total computational throughput $R_n^{tot}(t)$ and the total energy consumption $E_n^{tot}(t)$ for node n at t are expressed as

$$R_n^{tot}(t) = r_n^{local}(t) + r_n^{off}(t), \quad (9)$$

$$E_n^{tot}(t) = e_n^{local}(t) + e_n^{off}(t). \quad (10)$$

Our goal is to achieve a high computational throughput as well as a high energy efficiency by minimizing the power consumption and maximizing the computed bits. These two performance metrics are normally two conflicting goals to

optimize. We exploit the weighted sum to tackle this multi-objective problem and define the system cost as follows [30]:

$$F(\mathbf{x}(t),\mathbf{s}(t)) = \phi_{fe}E_{tot}(\mathbf{x}(t),\mathbf{s}(t)) - \phi_{fs}R_{tot}(\mathbf{x}(t),s(t)), \tag{11}$$

where $x(t) =: \{f_{m,n}(t), p_n(t)\}$, $E_{tot}(\mathbf{x}(t),\mathbf{s}(t)) = \sum_{n=1}^{N} E_n^{tot}(t)$ is the overall energy cost by all the edge nodes at Edge layer, while $R_{tot}(\mathbf{x}(t),s(t)) = \sum_{n=1}^{N} R_n^{tot}(t)$, is the overall system computation throughput. Furthermore, (ϕ_{fe},ϕ_{fs}) are the energy and rate coefficients. The problem is formulated as

$$\begin{aligned}
\mathbf{P1}: \min_{p_n(t),f_{m,n}(t),s_{m,n}(t)} & \limsup_{\mathcal{B}\to\infty} \frac{1}{\mathcal{B}} \sum_{t=0}^{\mathcal{B}-1} \mathbb{E}\left\{F(\mathbf{x}(t),\mathbf{s}(t))\right\} \\
s.t. \quad C1: & \limsup_{\mathcal{B}\to\infty} \frac{1}{\mathcal{B}} \sum_{t=0}^{\mathcal{B}-1} \sum_{n=1}^{N} \mathbb{E}\left\{Q_n(t)\right\} < \infty, \\
C2: & \ f_{m,n}^{min} \le f_{m,n}(t) \le f_{m,n}^{max}, \forall m,n, \\
C3: & \ 0 \le p_n(t) \le P_n^{max}, \forall n, \\
C4: & \ s_{m,n}(T) \in \{0,1\}, \forall m,n.
\end{aligned} \tag{12}$$

The first constraint $C1$ is the queue stability constraint. Constraints $C2$ and $C3$ represent the ranges of edge node processing frequency and transmission power, while constraint $C4$ denotes active or de-active state for each PU. Each PU of an edge node can be turned on or turned off depending on the demands. In the following, we will use the system cost and system efficiency alternately, since minimizing the system cost defined in $\mathbf{P1}$ is the same as maximizing the system efficiency.

Firstly, the problem $\mathbf{P1}$ is an NP-Hard mixed integer nonlinear programming problem, which normally has a very high computational complexity and is very challenging to solve in a real-time manner. On the other hand, the PU turn-on/turn-off operation may not need to be made in real-time for most systems due to on-off overhead concerns and hardware constraints. To address both real time need to allocate computing and communication resources as well as non-real-time need to turn on/off processing units, we propose a two-timescale algorithm to solve this optimization problem. The small timescale problem is executed every time slot t, while the large timescale problem is executed every epoch with the duration of T time slots. Correspondingly, the original problem $\mathbf{P1}$ can be decomposed into two sub-problems. The first sub-problem at large timescale decides how many PUs are needed for each edge node, while the second one at small timescale is the computing/communication resource allocation problem. To solve the first sub-problem, we design the large timescale prediction scheme to estimate the arriving workload, based on which turn-on/turn-off decisions for the PUs at each edge node are made. As a result, the number of active PUs changes from one epoch to another. For the second sub-problem, we aim for minimizing the total cost of both the energy consumption and the delay by using efficient resource allocation.

A. Large Timescale Optimization Model

This sub-problem aims for minimizing the energy consumption from large timescale perspective. The status of PU

Algorithm 1 DYNAMIC TURNING-ON/OFF CONFIGURATION ALGORITHM

1: **Initialization:** Set $\mathbf{s}(t) = \mathbf{1}$;
2: **Optimization:**
3: **while** $t \le T$ **do**
4: Predict the arriving workload for each edge node at the beginning of each epoch T.
5: At every epoch T, we perform the following jobs according to the estimated workload:
6: **if** workload increases **then**
7: Turn on the PUs from de-active set until the available resources can serve the arriving workload.
8: **else if** workload decreases **then**
9: Turn off the PUs from the active set of the edge nodes until reaching the balance of demand and resources.
10: **end if**
11: **end while**
12: **Output:** Set the processing unit status $\mathbf{s}(T) = \mathbf{s}^*(T)$.

$s_{m,n}(T)$, $m = 1, 2, \cdots, M$, at node n is determined in every T time slots. The sub-problem can be formulated as

$$\begin{aligned}
\mathbf{P2}: \min_{\{s_{m,n}(T)\}} & \sum_{T\in\mathbf{T}} F_1\left(\mathbf{x}^*,\mathbf{s}(T)\right) \\
s.t. \quad & s_{m,n}(T) \in \{0,1\}, \forall m,n,
\end{aligned} \tag{13}$$

where $F_1(\mathbf{x}^*,\mathbf{s}(T)) = \phi_{fe}E_{tot}(\mathbf{x}^*,\mathbf{s}(T)) - \phi_{fs}R_{tot}(\mathbf{x}^*,\mathbf{s}(T))$. $\mathbf{x}(t)$ is firstly set to $\mathbf{x}(T-1)^*$ in the objective function, which is the optimal resource allocation in the previous epoch. The decision is made based on both the prediction of the arriving workloads and also the efficiency of turn-on/off. The main idea for deciding turn-on/off of a PU is presented as follows.

The decision to turn on the inactive units depends on the workload state. If the arriving workload to an edge node keeps increasing, this edge node needs to turn on more PUs to support the arising workload. The turning-on operation is also performed at the beginning of each epoch. This operation is the coarse control in the long-term timescale model to protect the negotiated service level agreement (SLA). The finer control is implemented at the short timescale model to regulate the network parameters for the workload processing. This cooperative long-term and short-term timescale models would not only reduce the operating cost but also ensure the system's stability. Therefore, the proposed two-timescale framework is more efficient and flexible. The turning-on/off algorithm at each epoch is summarized in Alg. 1. The remaining task is how to estimate the arriving workloads, which is presented in the sections IV.

B. Small Timescale Optimization Model

In this subsection, we aim for minimizing the system cost at each time slot, given that the optimal value of the status vector $\mathbf{s}_t = \mathbf{s}^*(T)$ is obtained in the large timescale model. The small timescale model directly uses $\mathbf{s}^*(T)$ to seek the optimal value for $\mathbf{x}(t)$. Thus, we formulate the second sub-

problem as follows:

$$\mathbf{P3}: \min_{\mathbf{x}(t)} \quad \sum_{t=t_0}^{t_0+T-1} F_2(\mathbf{x}(t), \mathbf{s}^*(T)) \\ s.t. \quad C1 - C3, \tag{14}$$

where t_0 is the starting point of the current epoch and $F_2(\mathbf{x}(t), \mathbf{s}^*(T)) = \phi_{fe} E_{tot}(\mathbf{x}(t), \mathbf{s}^*(T)) - \phi_{fs} R_{tot}(\mathbf{x}(t), \mathbf{s}^*(T))$. The Lyapunov optimization method and SCALE method are used to make the resource allocation decision in every time slot, which will be presented in Section V.

IV. Large Timescale Workload Prediction

In this section, we firstly present the overview of machine learning, which is applied to the prediction method. Next, we provide some constraints of this method. We then present the long short term memory network to overcome these constraints.

A. Overview of Machine Learning Based Prediction Method

Many stochastic mechanisms have been exploited to effectively predict the workload flows. These methods can be generally classified into two categories, linear prediction methods and nonlinear prediction methods. For the linear prediction methods, one of the best prediction mechanisms in the correlated time series category is the autoregressive–moving-average (ARMA) model [31], [32], while the most commonly used non-linear mechanism is neural network. ARMA is a typical parametric regression model, which assumes that the traffic condition is a stationary process. It implies that the mean, variance and auto-correlation all stay constant. However, the ARMA method cannot capture the rapid variational process underlying the traffic data due to it concentrates on the mean value of the past series data.

The neural network technique is able to model more complicated data by using the distributed and hierarchical feature representation. Recently, many deep learning models that can extract multilevel features have been developed. To train the network parameters, they employ the machine learning such as supervised/unsupervised/semi-supervised learning methods, reinforcement learning schemes and nature-inspired algorithms [12]–[15], [33]. Workload prediction accuracy can be greatly improved. One of the common methods in deep neural network forecast is based on recurrent neural network (RNN) [15], [33], which is used in this paper. In particular, RNN works efficiently with time series and sequence modeling tasks, because it contains a self-recurrent loop that facilitates transporting information from one time slot to another. Note that the traditional neural network cannot achieve the same level performance in the temporal-spatial problems as it does not have the interconnection between nodes in the same layer. RNN introduces hidden units that allows the current state to receive feedback from the previous state.

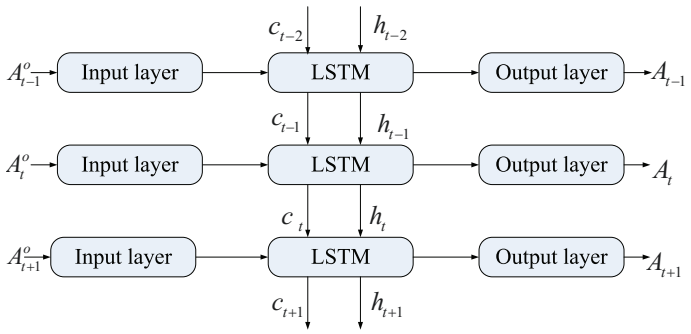

Fig. 2. LSTM network.

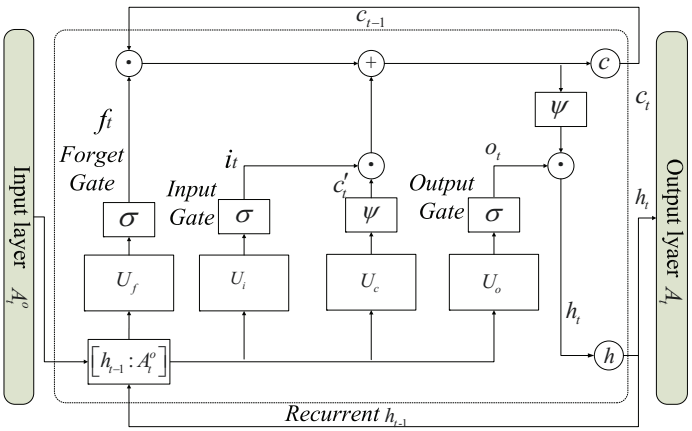

Fig. 3. LSTM memory block.

B. Long Short Term Memory Network

The original RNN only has one state, i.e. $\mathbf{h}$. Therefore it is very sensitive to the short timescale input and has the gradient vanishing problem for large timescale forecasting. To tackle this issue, we use the long short term memory (LSTM) model, which is one of the specially designed RNN models. LSTM [28] does the advanced time series prediction with long temporal dependency. It can learn information with long time spans and determine the optimal time lags autonomously. The key component that makes LSTM work for the long-term dependencies is called memory block. Fig. 2 shows a typical LSTM network, which usually consists of one input layer, one output layer, and one recurrent hidden layer containing the memory block.

The memory block integrated in the LSTM network is illustrated in Fig. 3. Here, the memory block is the recurrently connected subnet, which contains functional modules such as memory cells and gates. The memory cells are used for memorizing the temporal states of the network while the gates are composed of sigmoid layers, which are responsible for controlling the amount of information flows. According to the corresponding functions, gates can be classified into input gates, output gates, and forget gates. The input gate controls how much new information flows into the memory cell and

its weight matrix is defined as $\mathbf{U}_i$. The forget gate controls how much information remains in the current memory cell through a recurrent connection and its corresponding weight matrix is $\mathbf{U}_f$. An output gate controls how much information is used to compute the output activation of the memory block and how much information furthermore flows into the rest of the network. Its weight matrix is denoted by $\mathbf{U}_o$.

Detailed relationships of the entities in the LSTM network can be presented as follows. Recall that the input vector at the input layer at time t is $\mathbf{A}_t^o$, the hidden state vector at time t is $\mathbf{h}_t$, and the memory cell at time t is $\mathbf{c}_t$. Let the operations of dot product and summation of two vectors be denoted by $\cdot$ and $+$, respectively. Let $\sigma(\cdot)$ and $\psi(\cdot)$ denote the *sigmoid* function and the *hyperbolic tangent* function, respectively. So we have the definitions of these functions as

$$\sigma(x) = \frac{1}{1+e^{-x}}, \tag{15}$$

$$\psi(x) = 2\sigma(2x) - 1. \tag{16}$$

Based on the above definitions, the output of the forget gate $\mathbf{f}_t$ can be written as $\mathbf{f}_t = \sigma(\mathbf{U}_{fh}\mathbf{h}_{t-1}+\mathbf{U}_{fa}\mathbf{A}_t^o+\mathbf{b}_f)$, where $\mathbf{U}_{fa}$ and $\mathbf{U}_{fh}$ are the weight matrices of $\mathbf{A}_t^o$ and $\mathbf{h}_{t-1}$, respectively. Furthermore, $\mathbf{b}_f$ is the bias of $\mathbf{f}_t$, while σ is a sigmoid function.

Similar to the input gate, the output vector $\mathbf{i}_t$ of the input gate can be given by $\mathbf{i}_t = \sigma(\mathbf{U}_{ih}\mathbf{h}_{t-1} + \mathbf{U}_{ia}\mathbf{A}_t^o + \mathbf{b}_i)$, where $\mathbf{U}_{ia}$ and $\mathbf{U}_{ih}$ are the weight matrix for $\mathbf{A}_t^o$ and $\mathbf{h}_{t-1}$, while $\mathbf{b}_i$ is the corresponding bias. The input activation vector $\mathbf{c'}_t$ of the memory cell can be calculated as $\mathbf{c'}_t = \psi(\mathbf{U}_{ch}\mathbf{h}_{t-1} + \mathbf{U}_{ca}\mathbf{A}_t^o + \mathbf{b}_c)$, where $\mathbf{U}_{ca}$ and $\mathbf{U}_{ch}$ are the weight matrix for $\mathbf{A}_t^o$ and $\mathbf{h}_{t-1}$, $\mathbf{U}_c = [\mathbf{U}_{ch}, \mathbf{U}_{ca}]$, $\mathbf{b}_c$ is the corresponding bias. Here, the memory cell state vector $\mathbf{c}_t$ can be calculated as $\mathbf{c}_t = \mathbf{f}_t \cdot \mathbf{c}_{t-1} + \mathbf{i}_t \cdot \mathbf{c'}_t$, which is the combination of the input activation vector $\mathbf{c'}_t$ and the long memory $\mathbf{c}_{t-1}$.

Now we consider the last gate, i.e. the output gate. We have the calculation of the vector $\mathbf{o}_t$ of the output gate as $\mathbf{o}_t = \sigma(\mathbf{U}_{oa}\mathbf{A}_t^o + \mathbf{U}_{oh}\mathbf{h}_{t-1} + \mathbf{b}_o)$, where $\mathbf{U}_{oa}$ and $\mathbf{U}_{oh}$ are the weight matrix for $\mathbf{A}_t^o$ and $\mathbf{h}_{t-1}$ respectively, and $\mathbf{b}_o$ is the corresponding bias. The output vector $\mathbf{h}_t$ of the hidden layer can be expressed as $\mathbf{h}_t = \mathbf{o}_t \cdot \psi(\mathbf{c}_t)$. So the output vector $\mathbf{A}_t$ of the output layer is calculated as

$$\mathbf{A}_t = g(\mathbf{U}_N\mathbf{h}_t), \tag{17}$$

where $\mathbf{U}_N$ is the weight matrix of the output layer, $g(\cdot)$ is active function.

Now, we obtain the predicted workload $\mathbf{A}_t$ for the large timescale operation. In Section VI-A, we provide detailed operations of LSTM networks, the configuration for the inputs and the networks as well as the performance evaluations. In particular, our predicted mechanism outperforms the benchmark of ARMA. In the next section, based on the large timescale optimization results, we further determine the small timescale resource allocation optimization for each edge node.

V. Small Timescale Optimization and Configuration Model

In this section, we aim to minimize the system cost at each time slot. Recall that the large timescale model is proposed to solve problem $\mathbf{P2}$, where the optimal value of the status vector $\mathbf{s}(T) = \mathbf{s}^*(T)$ is determined. Given the solution of $\mathbf{s}^*(T)$, the small timescale model seeks the optimal value $\mathbf{x}(t)$ for problem $\mathbf{P3}$. Note that problem $\mathbf{P3}$ is an NP-hard problem, so we develop the Lyapunov optimization method to solve this problem in the following.

A. Overview of Lyapunov Optimization

We firstly provide the brief description of Lyapunov optimization, interested readers can find detailed information in [34]. The dynamic change of the arrival workload in constraint C1 of problem $\mathbf{P3}$ makes the objective function $F_2(x(t), s^*(T))$ hard to be solved. However, Lyapunov optimization method [34] can be used to deal with the dynamic resource allocation problem. In particular, the optimal solution would be obtained by solving a deterministic per-time slot problem with a much lower complexity.

We now formulate the Lyapunov function $L(t)$ as follow:

$$L(t) = \frac{1}{2}\sum_{n=1}^{N} Q_n^2(t). \tag{18}$$

The Lyapunov drift $\triangle(t)$ can be written as

$$\triangle(t) = L(t+1) - L(t) = \frac{1}{2}\sum_{n=1}^{N}[Q_n^2(t+1) - Q_n^2(t)]. \tag{19}$$

Accordingly, the Lyapunov drift-plus-penalty function can be expressed as

$$\triangle_V(t) = \triangle(t) + VF_2(t), \tag{20}$$

where V is a control parameter, while $F_2(t) = F_2(x(t), s^*(T))$ is the objective function of $\mathbf{P3}$.

Based on the definition of $Q_n(t)$, we have

$$\begin{aligned} Q_n^2(t+1) &\le [Q_n(t) + A_n(t) - R_n^{tot}(t)\tau]^2 \\ &\le Q_n^2(t) + A_n^2(t) + {R_n^{tot}}^2(t)\tau^2 + 2Q_n(t)A_n(t) \\ &\quad - 2(Q_n(t) + A_n(t))R_n^{tot}(t)\tau. \end{aligned} \tag{21}$$

Combining (19) and (21) and furthermore employing some mathematical manipulations, we have

$$\begin{aligned} \triangle(t) \le \sum_{n=1}^{N} &\frac{1}{2}(A_n^2(t) + {R_n^{tot}}^2(t)\tau^2) - A_n(t)R_n^{tot}(t)\tau \\ &+ Q_n(t)(A_n(t) - R_n^{tot}(t)\tau). \end{aligned} \tag{22}$$

Substituting $R_n^{tot}(t)$ from (9). Since $\log_2(1+x) \le \frac{x}{\ln 2}$ and $\log_2^2(1+x) \le \frac{2x}{(\ln 2)^2}$, we have

$$\begin{aligned} \triangle(t) \le &\sum_{n=1}^{N}\frac{1}{2}[(A_n^{max})^2 + (\sum_{m=1}^{M}\frac{f_{m,n}^{max}\tau}{C_{m,n}} + \frac{B\gamma_n^{max}\tau}{(\ln 2)})^2] \\ &- A_n^{max}(\sum_{m=1}^{M}\frac{f_{m,n}^{max}\tau}{C_{m,n}} + \frac{B\gamma_n^{max}\tau}{\ln 2}) \\ &+ \sum_{n=1}^{N} Q_n(t)(A_n(t) - R_n^{tot}(t)\tau) \\ \le &D_1 + \sum_{n=1}^{N} Q_n(t)(A_n(t) - R_n^{tot}(t)\tau)), \end{aligned} \tag{23}$$

where A_n^{max} is the maximum arrival workload at edge node n, while γ_n^{max} is the maximum SINR for the edge node n. So D_1 is defined as

$$D_1 = \sum_{n=1}^{N} \frac{1}{2} \left[(A_n^{max})^2 + \left(\sum_{m=1}^{M} \frac{f_{m,n}^{max}\tau}{C_{m,n}} + \frac{B\gamma_n^{max}\tau}{(\ln 2)} \right)^2 \right] - A_n^{max} \left(\sum_{m=1}^{M} \frac{f_{m,n}^{max}\tau}{C_{m,n}} + \frac{B\gamma_n^{max}\tau}{\ln 2} \right). \tag{24}$$

By adding $VF_2(t)$ to both sides of the above inequality (23), we obtain

$$\triangle_V(t) \leq D_1 + VF_2(t) + \sum_{n=1}^{N} Q_n(t)(A_n(t) - R_n^{tot}(t)\tau). \tag{25}$$

The proposed resource allocation algorithm for the small timescale model mainly focuses on minimizing the upper bound of $\triangle_V(t)$ on the right side of (25) at each time slot t.

B. Problem Formulation for Computation and Communication

We now formulate the problem for both communication and computation in the small timescale model. In particular, we aim for performing the following goals: *the workload at buffer queue $Q_n(t)$ can be kept at a stable level, while the system cost can also be minimized for each edge node and the cloud server*. Using derivations in section V-A, problem $\mathbf{P3}$ can be transformed to

$$\begin{aligned} \mathbf{P4}: \quad & \min_{x(t)} D_1 + V[\phi_{fe}E_{tot}(x(t)) - \phi_{fs}R_{tot}(x(t))] \\ & + \sum_{n=1}^{N} Q_n(t)(A_n(t) - R_n^{tot}(x(t))\tau) \\ & s.t. \quad C2, C3. \end{aligned} \tag{26}$$

It is worth mentioning that the objective function of $\mathbf{P4}$ is from the right-hand side of (25) and all the constraints in $\mathbf{P3}$ except the task buffer constraint C1 are retained in $\mathbf{P4}$. We can observe that it is still difficult to determine the optimal solution for $\mathbf{P4}$, where we must seek the optimal edge node local computation frequency and the optimal transmit power for offloading. Therefore, the problem $\mathbf{P4}$ can be further divided into two sub-problems, i.e. problem for local process speed optimization and problem for offloading power optimization.

1) Local Process Speed Optimization: In this subsection, we determine the optimal solution for the processing rates of PUs at each edge node. To obtain the optimal local process frequency, we would solve the following sub-problem

$$\begin{aligned} \mathbf{P4.1}: \quad & \min_{f_{m,n}(t)} \sum_{t=t_0}^{t_0+T-1} F_3(t) \\ & s.t. \quad f_{m,n}^{min} \leq f_{m,n}(t) \leq f_{m,n}^{max}, \end{aligned} \tag{27}$$

where $F_3(t) = V\sum_{n=1}^{N}[\phi_{fe}e_n^{local}(t)\tau - \phi_{fs}r_n^{local}(t)\tau] + \sum_{n=1}^{N} Q_n(t)(A_n(t) - r_n^{local}(t)\tau)$. Problem $\mathbf{P4.1}$ is a convex problem as the objective function is convex and the constraints are linear. The optimal solution is given as

$$f_{m,n}^*(t) = \min(f_{m,n}^{max}, \max(\overline{f}_{m,n}(t), f_{m,n}^{min})), \tag{28}$$

where $\overline{f}_{m,n}(t) = \sqrt{\frac{V\phi_{fs} + Q_n(t)}{3\phi_{fe}V\epsilon_{m,n}C_{m,n}}}$.

We have the following observations of the relation between $f_{m,n}^*(t)$ and the network parameters. In (28), it is readily observed that $f_{m,n}^*(t)$ is a strictly increasing function with respect to $Q_n(t)$. It means that when $Q_n(t)$ increases, the computational frequency $f_{m,n}^*(t)$ increases to keep the local computing buffer queue at the certain acceptable level. On the other hand, $f_{m,n}^*(t)$ decreases with the increases of $C_{m,n}, \epsilon_{m,n}$ and ϕ_{fe}. In particular, when $C_{m,n}$ and ϵ increase, the system needs a higher computational frequency and/or more energy consumption to process one bit of data. Thus, the processing unit m at edge node n must reduce its computational frequency to consume less power. With the increase of ϕ_{fe}, we have a higher priority on the energy consumption and therefore, the computational frequency $f_{m,n}^*(t)$ must be decreased. Of course, we also observe that $f_{m,n}^*(t)$ is a strictly increasing function of ϕ_{fs}

Let us consider the homogeneous scenario, where we have the same computational energy efficiency coefficient $\epsilon_{m,n}$ and the same number of computation cycles $C_{m,n}$ needed to process one bit of data for each processing unit m at node n. In this scenario, we have the same computational rate for all the units at each node. So the workload is equally assigned to each unit at node n.

2) FDMA Based Offloading Power Optimization: The task offloading can employ two methods, i.e. FDMA and NOMA. The optimization sub-problem of the FDMA based offloading is formulated in this subsection, while NOMA based offloading is presented in the next section. So the optimization sub-problem under the FDMA setting is given as

$$\begin{aligned} \mathbf{P4.2}: \quad & \min_{p_n(t)} \sum_{t=t_0}^{t_0+T-1} F_4(t) \\ & s.t. \quad 0 \leq p_n(t) \leq p_n^{max}, \end{aligned} \tag{29}$$

where $F_4(t)$ is defined as

$$\begin{aligned} F_4(t) = & V\sum_{n=1}^{N} \left[\phi_{fe}e_n^{off}(t)\tau - \phi_{fs}r_n^{off}(t)\tau\right] \\ & + \sum_{n=1}^{N} Q_n(t) \left[A_n(t) - r_n^{off}(t)\tau\right]. \end{aligned} \tag{30}$$

Recall that $e_n^{off}(t)$ and $r_n^{off}(t)$ are calculated at (4) and (5). By substituting $e_n^{off}(t)$ and $r_n^{off}(t)$ into the above equation, $F_4(t)$ can be rewritten as

$$\begin{aligned} F_4(t) = & V\tau\sum_{n=1}^{N}\phi_{fe}(\zeta p_n(t) + p_r) - \phi_{fs}B\log_2(1 + \frac{p_n(t)g_n^2(t)}{\sigma_n^2}) \\ & + \sum_{n=1}^{N} Q_n(t)\left[A_n(t) - B\tau\log_2(1 + \frac{p_n(t)g_n^2(t)}{\sigma_n^2})\right]. \end{aligned} \tag{31}$$

The problem $\mathbf{P4.2}$ is a convex function as its objective function is a linear function of convex terms and the constraint is also linear. We can derive its optimal solution as

$$p_n^*(t) = \min(p_n^{max}, \max(\overline{p}_n(t), 0)), \tag{32}$$

where $\overline{p}_n(t) = \frac{(V\phi_s + Q_n(t))B}{\ln 2 V\phi_{fe}\zeta} - \frac{\sigma_n^2}{g_n^2(t)}$.

We have the following observations on the optimal transmit power $p_n^*(t)$. The optimal transmit power $p_n^*(t)$ is non-decreasing with respect to queue size $Q_n(t)$. This indicates that when the queue size of node n increases, the offloading power increases in order to achieve a higher offloading rate. As a result, the workload accumulated in the queue is reduced. The transmit power $p_n^*(t)$ also increases with the increase of bandwidth B. Thus, the offloading rate is increased and we can offload more workloads. The transmit power is a decreasing function of the energy weight ϕ_{fe}. In particular, when ϕ_{fe} is higher, we set the higher priority for optimizing the energy consumption, i.e. energy consumption would be reduced.

3) Offloading Power Optimization for NOMA Method: With NOMA based offloading, the problem $\mathbf{P4.2}$ is reformulated to $\mathbf{P4.3}$. The detailed procedure is presented as follows. Under the NOMA setting, the optimization problem $\mathbf{P4.3}$ can be further divided to K optimization problems with each one denoted as $\mathbf{P4.3.k}$ for each NOMA group k. So problem $\mathbf{P4.3.k}$ is expressed as

$$\begin{aligned} \mathbf{P4.3.k}: \quad & \min_{p_{k,i}(t)} \sum_{t=t_0}^{t_0+T-1} F_5^k(t) \\ & s.t. \;\; 0 \le p_{k,i}(t) \le p_k^{max}, i = 1, 2, \end{aligned} \tag{33}$$

where $F_5^k(t)$ is the objective function, which is given as

$$\begin{aligned} F_5^k(t) &= V\phi_{fe}(\zeta p_{k,1}(t) + \zeta p_{k,2}(t) + 2p_r)\tau \\ &- V\phi_{fs}2B\tau[\log_2(1 + \frac{p_{k,1}(t)g_{k,1}^2(t)}{p_{k,2}(t)g_{k,2}^2(t) + \sigma_{k,1}^2}) \\ &+ \log_2(1 + \frac{p_{k,2}(t)g_{k,2}^2(t)}{\sigma_{k,2}^2})] + Q_{k,1}(t)[A_{k,1}(t) \\ &- 2B\tau\log_2(1 + \frac{p_{k,1}(t)g_{k,1}^2(t)}{p_{k,2}(t)g_{k,2}^2(t) + \sigma_{k,1}^2})] \\ &+ Q_{k,2}(t)[A_{k,2}(t) - 2B\tau\log_2(1 + \frac{p_{k,2}(t)g_{k,2}^2(t)}{\sigma_{k,2}^2})]. \end{aligned} \tag{34}$$

Here, we have two kinds of nodes in a NOMA group, i.e. the strong node and the weak node, which are denoted by the subscripts $_{k,1}$ and $_{k,2}$, respectively. Recall that we use the calculations of the offloading rates and the energy consumption for both strong and weak nodes from (6), (7) and (8).

It is observed that the objective function $F_5^k(t)$ is not a convex function, therefore the optimization problem is a non-convex problem. We exploit the convex relaxation method called SCALE (Successive Convex Approximation for Low Complexity) [35]. In the SCALE method, we use the following observation

$$\alpha\log_2(z) + \beta \le \log_2(1 + z). \tag{35}$$

This is tight at $z = \overline{z} \ge 0$, when the approximation coefficients are given as

$$\begin{aligned} \alpha &= \frac{\overline{z}}{1 + \overline{z}}, \\ \beta &= \log_2(1 + \overline{z}) - \frac{\overline{z}}{1 + \overline{z}}\log_2(\overline{z}). \end{aligned} \tag{36}$$

We now apply the SCALE method, where we use a logarithmic change of variables $\rho_{k,i}(t) = \ln(p_{k,i}(t))$. Furthermore, the terms $(\alpha_{k,1}, \beta_{k,1})$ and $(\alpha_{k,2}, \beta_{k,2})$ calculated by (36) are used for both the strong and weak nodes. After using some simple manipulations, we have the approximation of the $F_5^k(t)$ as

$$\begin{aligned} \tilde{F}_5^k(t) &= V\phi_{fe}[\zeta(\exp(\rho_{k,1}(t)) + \exp(\rho_{k,2}(t))) + 2p_r]\tau \\ &+ Q_{k,1}(t)A_{k,1}(t) + Q_{k,2}(t)A_{k,2}(t) + 2B\tau(V\phi_{fs} \\ &+ Q_{k,1}(t))\{\alpha_{k,1}\log_2[\exp(-\rho_{k,1}(t))\frac{\sigma_{k,1}^2}{g_{k,1}^2(t)} \\ &+ \frac{g_{k,2}^2(t)}{g_{k,1}^2(t)}(\exp(\rho_{k,2}(t) - \rho_{k,1}(t))] - \beta_{k,1}\} \\ &+ 2B\tau(V\phi_{fs} + Q_{k,2}(t))\{\alpha_{k,2}\log_2[\exp(-\rho_{k,2}(t)) \\ &\frac{\sigma_{k,2}^2}{g_{k,2}^2(t)}] - \beta_{k,2}\}. \end{aligned} \tag{37}$$

So the problem $\mathbf{P4.3.k}$ can be transformed into

$$\begin{aligned} \mathbf{P4.4.k}: \quad & \min_{\rho_{k,i}(t)} \tilde{F}_5^k(t) \\ & s.t. \; \rho_{k,i}(t) \le \ln(p_k^{max}). \end{aligned} \tag{38}$$

Lemma 1. *The problem* $\mathbf{P4.4.k}$ *is a convex problem with the given* $\alpha_{k,i}, \beta_{k,i}$.

Proof. The first part of the objective function $\tilde{F}_5^k(t)$ is evidently convex. The last two terms of the objective function are also convex as they are the log-sum-exp functions. The remaining parts are constant. Therefore, the objective function $\tilde{F}_5^k(t)$ is the summation of all convex terms, which is also convex. Furthermore, the constraint is convex. As a result, the problem considered is convex. □

In the following, we utilize the Lagrangian duality technique to solve the problem $\mathbf{P4.4.k}$. We define the Lagrangian function $L(\rho)$ as $L(\rho) = \tilde{F}_5^k(t)$. We firstly solve the stationary condition, i.e. $\partial L(\rho)/\partial\rho_{k,1} = 0$, where $\partial L(\rho)/\partial\rho_{k,1}$ is calculated as

$$\frac{\partial L(\rho)}{\partial(\rho_{k,1})} = V\phi_{fe}\tau\zeta\exp(\rho_{k,1}) - \frac{2B}{\ln 2}\alpha_{k,1}\tau(V\phi_{fs} + Q_{k,1}(t)). \tag{39}$$

Then, we transform the result back to the original solution space after solving the fixed-point equation. So the optimal solution for the strong node in group k at time slot t is given as

$$p_{k,1}^*(t) = \min(p_n^{max}, \max(\overline{p}_{k,1}(t), 0)), \tag{40}$$

where $\overline{p}_{k,1}(t) = \frac{2B\alpha_{k,1}(V\phi_{fs} + Q_{k,1}(t))}{\ln 2V\phi_{fe}\zeta}$.

We have the following observations for the optimal solution as follows. The optimal transmit power $p_{k,1}^*(t)$ for strong node increases with the increase of the rate coefficient ϕ_{fs} and the buffer queue $Q_{k,1}(t)$. This confirms that 1) when the weight of data process rate becomes higher, the edge node increases its offloading power to achieve a higher rate; 2) when the buffer queue $Q_{k,1}(t)$ becomes larger, the edge node also needs to improve its offloading rate to reduce the buffer queue by increasing the transmit power. The optimal transmit power $p_{k,1}^*(t)$ decreases with the increase of the weight of

Algorithm 2 THE SCALE ITERATIVE ALGORITHM FOR $\mathbf{P4.4.k}$

1: **Input settings:** the error tolerance $\xi > 0$, $p_n^{max} > 0$ and ϕ_{fe},ϕ_{fs}, $\alpha_{k,1}^1 = 1$,$\alpha_{k,2}^1 = 1$, $\beta_{k,1}^1 = 0$, $\beta_{k,2}^1 = 0$ and the maximum iteration number I.
2: **Initialization:** $i = 0$, $p_{k,1}^i(t)$ and $p_{k,2}^i(t)$.
3: **Optimization:**
4: **for** $i = 1 : I$ **do**
5: obtain the solution $p_{k,1}^i(t)$ by (40) and solve $p_{k,2}^i(t)$ by (41).
6: **if** $p_{k,1}^i(t) - p_{k,1}^{i-1}(t) \leq \xi$ & &$p_{k,2}^i(t) - p_{k,2}^{i-1}(t) \leq \xi$ **then**
7: the optimal $p_{k,1}^*(t)$ and $p_{k,2}^*(t)$ are obtained.
8: **else**
9: update $\alpha_{k,1}^{i+1}$, $\alpha_{k,2}^{i+1}$, $\beta_{k,1}^{i+1}$ and $\beta_{k,2}^{i=1}$ by (36) and $i = i + 1$.
10: **end if**
11: **end for**
12: **Output:** $\{p_{k,1}^*(t), p_{k,2}^*(t)\}$.

transmission power consumption. When ζ becomes larger, i.e. the system consumes more energy for offloading tasks, it needs to reduce the transmit power and allocate more tasks to the local processing than to offloading.

Similarly, we can determine the optimal solution for the weak node in the same manner. We firstly set $\partial L(\rho)/\partial \rho_{k,2} = 0$ and then use some manipulations to obtain the optimal transmit power. The calculations and derivations of the optimal solution are omitted because they can be done in the simple steps. Finally, the optimal transmit power for the weak node in group k at time slot t can be given as

$$p_{k,2}^*(t) = \min(p_n^{max}, \max(\overline{p}_{k,2}(t), 0)), \tag{41}$$

where $\overline{p}_{k,2}(t) = \frac{-1}{2}(d_3 - \sqrt{d_3^2 + 4d_4})$, $d_3 = \frac{\sigma_{k,1}^2}{g_{k,2}^2(t)} + \frac{d_1 - d_2}{V\phi_{fe}\zeta}$, $d_4 = \frac{d_2\sigma_{k,1}^2}{V\phi_{fe}\zeta g_{k,2}^2(t)}$, $d_1 = \frac{2B}{\ln 2}\alpha_{k,1}(V\phi_{fs} + Q_{k,1}(t))$ and $d_2 = \frac{2B}{\ln 2}\alpha_{k,2}(V\phi_{fs} + Q_{k,2}(t))$. We summarize the procedures of solving problem $\mathbf{P4.4.k}$ in Alg. 2.

4) Algorithm complexity analysis: For Alg. 1, the complexity comes from two parts. The firstpart comes from estimating the workload at each E-node, while the second part comes from the turn on/off operation performed at each PU. Let N and M denote the number of users and the number of PUs of each user, respectively. Then, based on workload estimation and turn on/off operation, the complexity of Alg. 1 is $\mathcal{O}(MN)$.

For Alg. 2, the complexity also comes from two parts. The first part comes from updating the parameters α and β, while the second part comes from calculating the offloading power for each edge node. Let L be the number of iterations required to update the approximation parameters α and β and let N be the number of edge nodes. Then, the complexity of Alg. 2 is $O(LN)$.

VI. NUMERICAL RESULTS

A. Long-Short Term Memory Workload Prediction

The LSTM model performance on traffic prediction is first evaluated using the real traffic dataset [36], which records a total of 91065 user activities and their behaviors between Jan. 2006 and Jan. 2009. These data are widely used in different cloud communication studies and used as the arrival workload in this study [39]- [41]. The original data contains many features such as user ID, user class, sequence number, etc. Here, the number of active users is used as the number of arrived workloads. Therefore, the raw data is transformed to the number of users arriving in every time slot. We assume that each user represents a workload with 1000-bit data that needs to be processed [26] [30].

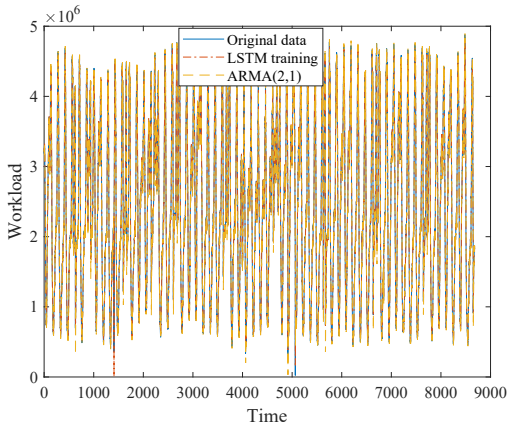

Fig. 4. Comparison between LSTM model and ARMA model in training.

For the efficient learning of LSTM, the original data first is normalized based on min-max normalization as follows:

$$\overline{A}_t^o = \frac{A_t^o - A_{min}^o}{A_{max}^o - A_{min}^o}, \tag{42}$$

where A_{min}^o and A_{max}^o are the minimum and maximum values of the original data [42]. The LSTM network adopted has one input layer with one input, one hidden layer with 4 LSTM blocks, and one output layer that makes a single-value prediction. We use the LSTM method to predict the data arriving at the edge nodes. Each dataset is divided into two parts, where 67% of the dataset are used for training the LSTM model, and the remaining 37% data are used for testing. We also compare the proposed method with ARMA$(2, 1)$ model [37]. The mean absolute performance error (MAPE) is used in this paper for evaluating the prediction errors [43]. The MAPE is the ratio of the error and the true value, which is defined as

$$\text{MAPE} = \frac{1}{T}\sum_{t=1}^{T}\frac{|A_t - \overline{A}_t^o|}{\overline{A}_t^o}. \tag{43}$$

Figs. 4 and 5 illustrate the training outcomes with different methods. For a better observation, we shift the results of the ARMA model [31], [32] and the LSTM model with 1 time slot from the original data. From Fig. 4, both ARMA model and LSTM model can well capture the overall trend of the original data. However, Fig. 5 with finer granularity indicates that ARMA method does not follow the rapid change of the workload flow as well as LSTM. Thus, the ARMA method has larger prediction errors when comparing to the LSTM mechanism. Figs. 6 and 7 show the testing results for the different prediction method, where the LSTM method outperforms the ARMA(2,1) method. The MAPE performances for both training part and testing part of the two methods are presented in Figs. 8 and 9, respectively. We can see that

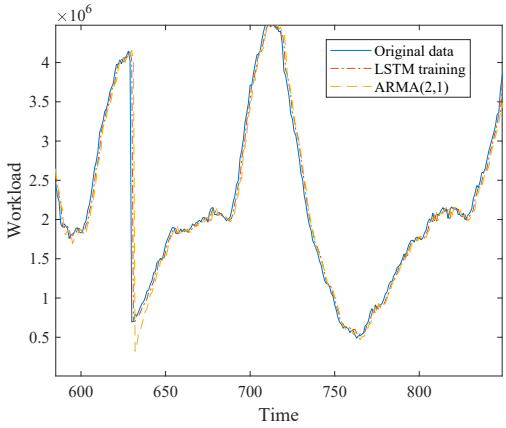

Fig. 5. Detailed training part of LSTM model and ARMA model.

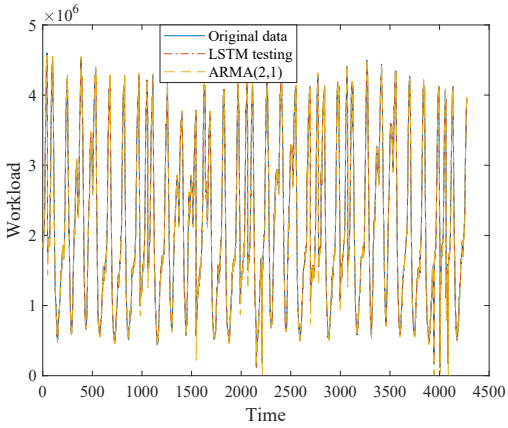

Fig. 6. Comparison between LSTM model and ARMA model in testing.

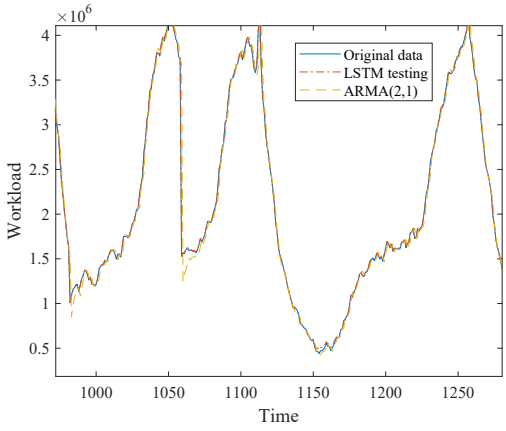

Fig. 7. Detailed testing part of LSTM model and ARMA model.

LSTM can achieve a lower error result. Therefore, the LSTM method is used for the prediction of workload flows in the following experiments.

B. System Cost Optimization

Based on the prediction results, this section gives the performance of the proposed methods. The simulation settings are based on the work in [16], [38]. All the parameters are

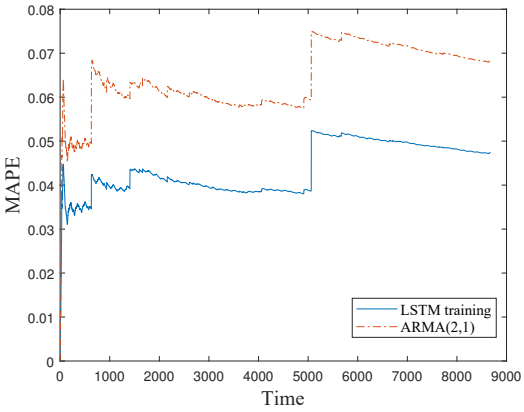

Fig. 8. Comparison of the performance of the mean absolute performance error for training part.

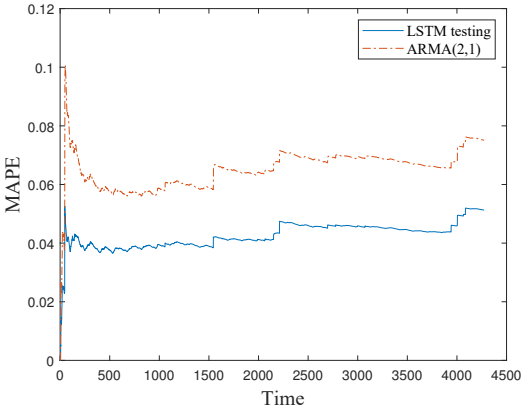

Fig. 9. Comparison of the performance of mean absolute error for testing part.

chosen as follows unless stated otherwise. There are $N = 2$ edge nodes and one centralized server, where each edge node has $M = 10$ processing units. Since the mobility is not considered in this paper, the location for each edge node is fixed during the entire simulation. The offloading transmission power for the communication link between each edge node and the sever is in the range of $\left[p_n^{min}, p_n^{max}\right]$, where $p_n^{min} = 0$ W and $p_n^{max} = 2.5$ W. The channel bandwidth for FDMA is $B = 2$MHz, the local processing capacity for one bit is $C_{m,n} = 1 \times 10^3$ cycles/bit. The computational energy efficiency coefficient is $\epsilon = 1 \times 10^{-27}$, the power weight $\zeta = 2$. The channel between the edge node and the server is modeled as the joint effect of large-scale and small-scale fading, where the channel parameters are given as $g_k^2/\sigma^2 = G_k h_k$, $G_1 = 7$ and $G_2 = 3$. Note that h_k is the unitary Gaussian random variable [16]. The computational capacity of each edge node is set equally in the range of $\left[f_{m,n}^{min}, f_{m,n}^{max}\right]$, where $f_{m,n}^{max} = 10^9$ Hz and $f_{m,n}^{min} = 10^3$ Hz. The circuit power is $p_r = 1$ dBm. To balance the value of throughput and energy, the weights are selected as $\phi_{fe} = 10^6$ and $\phi_{fs} = 0.1$. The results are obtained in different random channel realizations.

The study considers four cases: (1) the proposed PU on/off scheme with FDMA offloading. In this scheme, the data

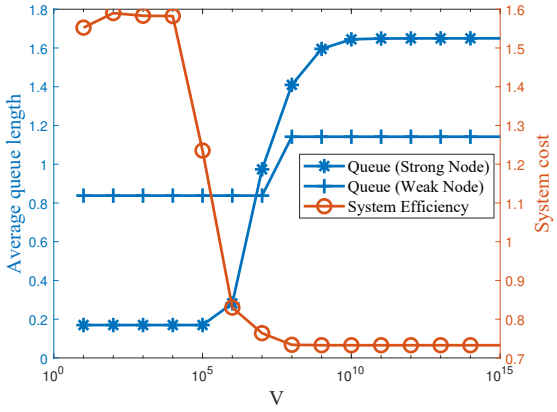

Fig. 10. Normalized system cost and average queue length per edge node vs the control parameter V.

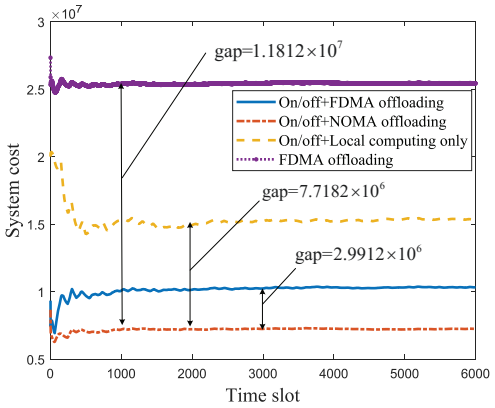

Fig. 11. Comparison of system cost over total duration.

offloading from the edge node to the cloud server adopts the FDMA method based on problem P4.2, thus, it marked as "On/Off + FDMA offloading". (2) The proposed method PU on/off with NOMA offloading "On/Off + NOMA offloading", where the offloading scheme between the edge node and the cloud server is the NOMA method based on the problem P4.3. (3) The benchmark scheme without PU on/off based on FDMA offloading. All the PUs keep in "on" status. The scheme is marked as "FDMA offloading" in the figure. (4) The benchmark scheme without the cloud server assistance. So the system can only process the data locally at edge nodes but allows PUs to turn on/off. The method is marked as "On/Off + local computing only". We set the epoch duration at $T = 50\tau$. All the simulation results are averaged based on 100 independent runs. We note that only the last scheme is the local processing, while the first three schemes have both local processing and cloud processing. To avoid any confusion, we firstly confirm that the term "offloading" does not mean that all the tasks must be offloaded to the cloud. It only means that we use offloading mechanisms, like NOMA or FDMA, to offload partial, or complete, or no tasks to the cloud, while the remaining tasks can be still processed at the local PUs. So we always keep the PUs in "on" status in the third case of "without PU on/off" due to the following reason. In this case, we do not use the large timescale model to predict the workload flow as well as use the turning-on/off algorithm. So we keep the PUs "on" to serve the high demand of workload as we consider the dynamic change of the workload flow.

The relationship between the system cost/average queue length of the task buffers and the control parameter V is presented in Fig. 10 for the proposed "On/Off + NOMA offloading" scheme. For a better illustration, the values of both system cost and queue length are normalized. The system cost firstly maintains at a high level when $V \leq 10^4$, it then decreases with the increase of V. When $V \geq 10^8$, the system cost keeps at a low level. On the other hand, the lengths of buffer queues for both the strong edge node and weak edge node are small when the system cost is high. It then increases when the system cost drops down. The reason is that the parameter V controls the tradeoff between the original cost

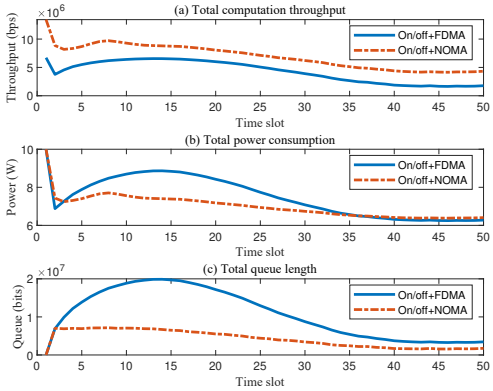

Fig. 12. "On/Off+FDMA" scheme vs. "On/Off+NOMA" scheme.

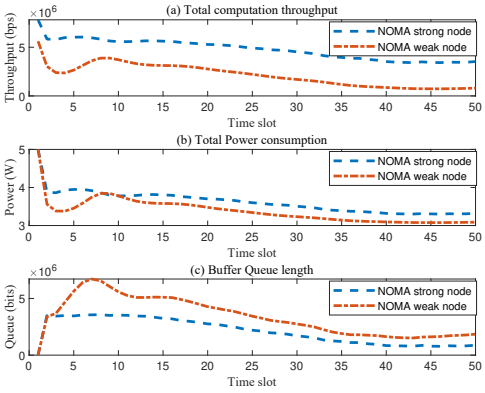

Fig. 13. Strong node vs. weak node in the "On/Off+NOMA" scheme.

function $F(t)$ and the buffer queue stability in the Lyapunov drift-plus-penalty function in (20). Therefore, by increasing the value of V, the system gives more priority to reduce the system cost and less priority to serve the buffered data. The optimal solution of V to achieve a good tradeoff between the system cost and buffer queue stability is around $V = 10^7$ based on the simulation setting. Thus, we choose the control

parameter as $V = 10^7$ in the following simulation results.

We proceed to show the performance of different schemes by using the predicted workloads and parameters setting found above. Fig. 11 shows the system cost for four different schemes defined above. By properly turning on/off processing units, the proposed scheme achieves a much lower total system cost than all other schemes by integrating NOMA, computation offloading and PU On/Off altogether. The only scheme "FDMA offloading" that does not use PU On/Off has a significantly higher system cost than others. Furthermore, computation offloading considerably reduces the system cost. We also observe that all the schemes converge after 500 time slots. This observation confirms the stability and convergence of the proposed methods. Although the workload flows dynamically change over time, all the edge nodes effectively allocate the transmit power for communication and also adjust the processing unit frequency for computation. Therefore, the model can adaptively achieve the optimal and stable system efficiency even with the dynamic traffic behaviors.

We further compare the performance for the two best schemes, namely "On/Off + NOMA offloading" and "On/Off + FDMA offloading" in Fig. 12. Fig. 12(a) shows that the NOMA offloading based scheme achieves a higher computation throughput than the FDMA offloading based scheme. Further in Fig. 12(b), the NOMA offloading based scheme consumes less power than the FDMA offloading based scheme. Combining the two figures, it demonstrates that NOMA based scheme attains a much higher efficiency in energy usage by consuming less energy but gaining a higher computation rate. A higher computation throughput leads to less queued data in the buffer, which is verified in Fig. 12(c).

Another thing worth noticing is that the curves of the total computation throughput, the power consumption and the buffer queue length for the FDMA method in Fig. 12 first go up and then decline in the main observation. However, we would see the fluctuations at the very beginning. This transient behavior is explained as follows. The initial value is high based on the initial parameter setting so that the system can achieve high throughput and low power consumption. There are not too many workloads needed to be processed at the beginning of large timescale. Thus, the system only adjusts parameters for the throughput and power, which can help to keep the higher efficiency. We now explain the system behavior in the main observation. At the beginning of each epoch, the workload is firstly buffered in the queue due to insufficient computation throughput of each node. Due to the increase of the queue length, the system adjusts both the offloading rate and local process speed so that the queued data can be served. At the end of the epoch, with the small queue length, the system can keep the computation throughput stabilized at a lower optimal level and maintains the minimum system cost. On the other hand, we observe that the curves of the NOMA based scheme keep decreasing. At the beginning of the epoch, the computation throughput is high enough to sufficiently serve the arrival and queued data so that queue size does not build up. Ultimately the NOMA and FDMA based schemes converge to the similar queue level and similar power consumption level.

The performance of the strong node and weak node for the NOMA based scheme is provided in Fig. 13. The weak edge node in the NOMA method has a lower throughput and also a lower energy consumption than the strong node. The proposed method aims to minimize the overall system cost. The transmit power for the nodes with different channel quality is adjusted by slightly increasing the power of the strong node and decreasing the weak one. We also have the transient duration at the beginning, which is similar to the observation in Fig. 12. Thus, the system can achieve an overall low system cost but still guarantee the weak node's performance. Therefore, the proposed method can dynamically adjust the resource allocated to each node to achieve the optimal overall system efficiency and meet the performance requirements of each node.

VII. Conclusions

This paper considers a hierarchical architecture that consists of IoT sensor layer, edge computing layer, and cloud server layer. A twin-timescale optimization model was developed to manage the workload offloading in the system to optimize the trade off between the power consumption and overall computation throughput. In the large timescale model, based on predicted workload, the scheme decides how to turn on or turn off processing unit in order to save energy. In the small timescale model, a Lyapunov optimization method was designed to allocate the offloading power and to determine the local process speed for each processing unit. Simulation results reveal that the proposed method can greatly improve system performance by saving energy costs and achieving a high processing rate.

References

[1] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (IoT): a vision, architectural elements, and future directions," *Future generation computer systems*, vol.29, no. 7, pp.1645-1660, Sep. 2013.

[2] R. Q. Hu and Y. Qian, "An energy efficient and spectrum efficient wireless heterogeneous network framework for 5G systems," *IEEE Wireless Comm. Mag.*, vol. 52, no. 5, May 2014.

[3] J. G. Andrews, S. Buzzi, W. Choi, S. V. Hanly, A. Lozano, A. C. Soong, and J. C. Zhang, "What will 5G be?," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 6, pp. 1065-1082, June 2014.

[4] H. Sun, Z. Zhang, R. Q. Hu, and Y. Qian, "Wearable communications in 5G: challenges and enabling technologies," *IEEE Veh. Technol. Mag.*, vol. 13, no. 3, pp. 100-109, Sep. 2018.

[5] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," *Proc. MCC'2012*.

[6] M. Peng, S. Yan, K. Zhang, and C. Wang, "Fog-computing-based radio access networks: issues and challenges," *IEEE Network*, vol. 30, no. 4, pp. 46-53, July-Aug. 2016.

[7] Y. Wang, M. Sheng, X. Wang, L. Wang, and J. Li, "Mobile-edge computing: partial computation offloading using dynamic voltage scaling," *IEEE Trans. Commun*, vol. 64, no. 10, pp. 4268-4282, Oct. 2016.

[8] C. You, K. Huang, H. Chae, and B. H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1397-1411, Mar. 2017.

[9] T. Chen and G. B. Giannakis, "Bandit convex optimization for scalable and dynamic IoT management," in *IEEE Internet Things J.*, vol. 6, no. 1, pp. 1276-1286, Feb. 2019.

[10] J. Du, L. Zhao, J. Feng, and X. Chu, "Computation offloading and resource allocation in mixed fog/cloud computing systems with min-max fairness guarantee," *IEEE Trans. Commun.*, vol. 66, no. 4, pp. 1594-1608, Apr. 2018.

[11] L. T. Tan, R. Q. Hu, and Y. Qian, "D2D communications in heterogeneous networks with full-duplex relays and edge caching,"*IEEE Trans. Ind. Informat.*, July 2018.

[12] L. T. Tan and R. Q. Hu, "Mobility-aware edge caching and computing in vehicle networks: A deep reinforcement learning," *IEEE Trans. Veh. Technol.*, vol. 67, no. 11, pp. 10190-10203, Nov. 2018.

[13] L. T. Tan, R. Q. Hu, and L. Hanzo, "Twin-timescale artificial intelligence aided mobility-aware edge caching and computing in vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3086-3099, Apr. 2019.

[14] L. T. Tan, R. Q. Hu, and L. Hanzo, "Heterogeneous networks relying on full-duplex relays and mobility-aware probabilistic caching," *IEEE Trans, Commun.*, vol. 67, no. 7, pp. 5037-5052, July 2019.

[15] A. Zahin, L. T. Tan, and R. Q. Hu, "A machine learning based framework for the smart healthcare monitoring," *Proc. IETC2020*.

[16] H. Sun, F. Zhou, and R. Q. Hu, "Joint offloading and computation energy efficiency maximization in a mobile edge computing system," *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 3052-3056, Mar. 2019.

[17] M. Li, S. Yang, Z. Zhang, J. Ren, and G. Yu, "Joint subcarrier and power allocation for OFDMA based mobile edge computing system," *Proc. IEEE PIMRC* 2017.

[18] S. Kim, S. Park, M. Chen, and C. Youn, "An optimal pricing scheme for the energy-efficient mobile edge computation offloading with OFDMA," *IEEE Commun. Lett.*, vol. 22, no. 9, pp. 1922-1925, Sep. 2018.

[19] S. Wang, C. Pan, and C. Yin, "Joint heterogeneous tasks offloading and resource allocation in mobile edge computing systems", in *Proc. IEEE WCSP*, Hangzhou, Oct. 2018.

[20] C. You, K. Huang, H. Chae, and B. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1397-1411, Mar. 2017.

[21] L. P. Qian, A. Feng, Y. Huang, Y. Wu, B. Ji, and Z. Shi, "Optimal SIC ordering and computation resource allocation in MEC-aware NOMA NB-IoT networks," *IEEE Internet Things J.*, Oct. 2018.

[22] H. Sun, Q. Wang, S. Ahmed, and R. Q. Hu, "Non-orthogonal multiple access in a mmWave based IoT wireless system with SWIPT," *Proc. VTC*, Sydney, NSW, Jun. 2017.

[23] Y. Wang, Y. Wu, F. Zhou, Z. Chu, Y. Wu, and F. Yuan, "Multi-objective resource allocation in a NOMA cognitive radio network with a practical non-linear energy harvesting model," *IEEE Access*, vol. 6, pp. 12973-12982, 2018.

[24] Y. Pan, M. Chen, Z. Yang, N. Huang, and M. Shikh-Bahaei, "Energy efficient NOMA-based mobile edge computing offloading," *IEEE Commun. Lett.*, Nov. 2018.

[25] D. Lymberopoulos, A. Bamis, and A. Savvides, "Extracting spatiotemporal human activity patterns in assisted living using a home sensor network," *Universal Access Inf.*, vol.10, no. 2, pp. 125-138, June 2011.

[26] Y. Mao, J. Zhang, S. H. Song, and K. B. Letaief, "Stochastic joint radio and computational resource management for multi-user mobile-edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 16, no. 9, pp. 5994-6009, Sep. 2017.

[27] X. Lyu, W. Ni, H. Tian, R. P. Liu, X. Wang, G. B. Giannakis, and A. Paulraj, "Optimal schedule of mobile edge computing for internet of things using partial information," *IEEE J. Select. Areas Commun.*, vol. 35, no. 11, pp. 2606-2615, Nov. 2017.

[28] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, Mar. 1997.

[29] D. Wu, J. Wang, R. Q. Hu, Y. Cai, and L. Zhou, "Energy-efficient resource sharing for mobile device-to-device multimedia communications,"*IEEE Trans. Veh. Technol.*, vol. 63, no. 5, pp. 2093-2103, May 2014.

[30] Y. Mao, J. Zhang, and K. B. Letaief, "A Lyapunov optimization approach for green cellular networks with hybrid energy supplies," *IEEE J. Select. Areas Commun.*, vol. 33, no. 12, pp. 2463-2477, Dec. 2015.

[31] S. J. Huang, and K. R. Shih, "Short-term load forecasting via ARMA model identification including non-Gaussian process considerations," *IEEE Trans. Power Syst.*, vol. 18, no. 2, pp. 673-679, May 2003.

[32] L. T. Tan and L. B. Le, "Compressed sensing based data processing and MAC protocol design for smartgrids," *Proc. WCNC'2015*.

[33] A. Zahin, L. T. Tan, and R. Q. Hu, "Sensor-based human activity recognition for smart healthcare: A semi-supervised machine learning," *Proc. ICAIIC'2019*.

[34] M. J. Neely, "Stochastic network optimization with application to communication and queueing systems," Morgan & Calypool, 2010.

[35] J. Papandriopoulos and J. S. Evans, "SCALE: A low-complexity distributed protocol for spectrum balancing in multiuser DSL networks," *IEEE Trans. Inf. Theo.*, vol. 55, no. 8, pp. 3711-3724, Aug. 2009.

[36] Y. T. Lee, K. T. Chen, Y. M. Cheng, and C. L. Lei, "World of Warcraft avatar history dataset," *Proc. MMSys'2011*.

[37] Q. Wang, L. T. Tan, R. Q. Hu, and G. Wu, "Hierarchical collaborative cloud and fog computing in IoT networks," *Proc. IEEE WCSP2018*.

[38] Q. Wang and F. Zhou, "Fair resource allocation in an MEC-enabled ultra-dense IoT network with NOMA," *Proc. ICC Workshops 2019*, Shanghai, China, 2019, pp. 1-6.

[39] W. Xiao, W. Bao, X. Zhu, C. Wang, L. Chen, and L. T. Yang, "Dynamic request redirection and resource provisioning for cloud-based video services under heterogeneous environment," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 7, pp. 1954-1967, July 2016.

[40] G. Wu, W. Bao, X. Zhu, W. Xiao, and J. Wang, "Optimal dynamic reserved bandwidth allocation for cloud-integrated cyber-physical systems," *IEEE Access*, vol. 5, pp. 26224-26236, 2017.

[41] S. Wang and S. Dey, "Adaptive mobile cloud computing to enable rich mobile multimedia applications," *IEEE Trans. Multimedia*, vol. 15, no. 4, pp. 870-883, June 2013.

[42] Y. Yu, J. Cao, and J. Zhu, "An LSTM short-term solar irradiance forecasting under complicated weather conditions," *IEEE Access*, vol. 7, pp. 145651-145666, 2019.

[43] K. Park, Y. Choi, W. J. Choi, H. Ryu, and H. Kim, "LSTM-based battery remaining useful life prediction with multi-channel charging profiles," *IEEE Access*, vol. 8, pp. 20786-20798, 2020.

Qun Wang (Student Member, IEEE) received the M.S. degree from Xidian University, Xi'an, China, in 2016. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, Utah State University, Logan, UT, USA. His research interests include mobile edge computing, nonorthogonal multiple access, intelligent reflect surface, MIMO, and machine learning.

Le Thanh Tan (Member, IEEE) received the B.Eng. and M.Eng. degrees from Ho Chi Minh University of Technology, Ho Chi Minh City, Vietnam, in 2002 and 2004, respectively, and the Ph.D. degree from Institut National de la Recherche Scientifique, Quebec City, QC, Canada, in 2015, all in telecommunications. He is currently with the Department of Electrical and Computer Engineering, Utah State University, Logan, UT, USA. From 2002 to 2010, he was a Lecturer with Ho Chi Minh University of Technical Education. In 2015, he was a Postdoctoral Research Associate with the Ecole Polytechnique de Montreal, Montreal, QC, Canada. From 2016 to 2017, he was a Postdoctoral Research Associate with Arizona State University, Tempe AZ, USA. His research interests include artificial intelligence, machine learning, Internet of Things, vehicular networks, 5G wireless communications, edge computing, fog computing and cloud computing, information centric networking, software defined networking, and network function virtualization. Dr. Tan was on TPCs of different international conferences including IEEE CROWNCOM, VTC, PIMRC, etc.

Rose Qingyang Hu (Fellow, IEEE) received the B.S. degree from the University of Science and Technology of China, the M.S. degree from New York University, and the Ph.D. degree from the University of Kansas. Besides a decade academia experience, she has more than 10 years of R&D experience with Nortel, Blackberry, and Intel as a Technical Manager, a Senior Wireless System Architect, and a Senior Research Scientist, actively participating in industrial 3 G/4 G technology development, standardization, system level simulation, and performance evaluation. She is a Professor with the Electrical and Computer Engineering Department and Associate Dean for research of College of Engineering at Utah State University. She also directs Communications Network Innovation Lab at Utah State University. Her current research interests include next-generation wireless system design and optimization, Internet of Things, Cyber Physical system, Mobile Edge Computing, V2X communications, artificial intelligence in wireless networks, wireless system modeling and performance analysis. She has published extensively in top IEEE journals and conferences and also holds numerous patents in her research areas. She is currently serving on the editorial boards of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE Communications Magazine and IEEE WIRELESS COMMUNICATIONS. She also served as the TPC Co-Chair for the IEEE ICC 2018. She is an IEEE Communications Society Distinguished Lecturer Class 2015-2018 and a recipient of prestigious Best Paper Awards from the IEEE GLOBECOM 2012, the IEEE ICC 2015, the IEEE VTC Spring 2016, and the IEEE ICC 2016. She is member of Phi Kappa Phi Honor Society.

Yi Qian (Fellow, IEEE) received the Ph.D. degree in electrical engineering from Clemson University. He is a Professor with the Department of Electrical and Computer Engineering, University of Nebraska-Lincoln (UNL). His research interests include communications and systems, and information and communication network security. He was previously Chair of the IEEE Technical Committee for Communications and Information Security. He was the Technical Program Chair for IEEE International Conference on Communications 2018. He serves on the editorial boards of several international journals and magazines, including as the Editor-in-Chief for IEEE Wireless Communications. He was a Distinguished Lecturer for IEEE Vehicular Technology Society. He is currently a Distinguished Lecturer for IEEE Communications Society.