Online User Scheduling and Resource Allocation for Mobile-Edge Computing Systems

Shanfeng Huang*,†, Bojie Lv* and Rui Wang*
*Department of Electrical and Electronic Engineering, Southern University of Science and Technology
†Department of Electrical and Electronic Engineering, The University of Hong Kong
Email: sfhuang@eee.hku.hk, lyubj@mail.sustech.edu.cn, wang.r@sustech.edu.cn

Abstract—In this paper, we investigate the multi-user mobile edge computing (MEC) system where the number of users in the system is varying. We formulate the multi-user scheduling problem as an infinite horizon Markov decision process (MDP) problem aiming at minimizing the weighted sum of energy consumption and latency of mobile users. For such a large MDP problem, traditional dynamic programming approaches such as value iteration and policy iteration will suffer from the curse of dimensionality. This motivates us to resort to approximate MDP where the key lies at value function approximation. We propose a novel value function approximation approach where the value function is approximated by running a well-designed heuristic baseline policy. Based on the approximated value function, one-step policy iteration is applied to jointly optimize the offloading decision, user selection and transmission power for the selected user. It is theoretically shown that our proposed algorithm has performance improvement compared with the baseline policy. Also, simulation results demonstrate our proposed scheme significantly outperforms the baseline policy and several other heuristic policies.

I. INTRODUCTION

The last decade has witnessed an unprecedented increase in mobile data traffic. In the meanwhile, mobile users’ requests on high data rate and low latency are growing exponentially. New mobile applications with intensive computing tasks and stringent latency requirements are springing out. Due to the limited battery lives and computing capabilities of mobile devices, some computation-intensive tasks need to be off-loaded to more powerful cloud servers, leading to Mobile Cloud Computing (MCC) architecture. Despite the plethora of resources in MCC servers, they are typically located far away from mobile users, resulting in huge overhead in backhaul networks and high propagation delay. To cope with the surging data rate demands and improve quality of service (QoS), mobile operators must seek new network architectures that have the capabilities to provide low latency and high bandwidth network access. Mobile Edge Computing (MEC) is envisioned as a promising solution to easing the conflict between resource-hungry delay sensitive applications and resource-limited mobile devices. In an MEC system, the computation, storage, and network resources are moved to the edge of cellular networks, in close proximity to mobile users. Hence, the quality of computation experience including mobile power consumption and computing latency can be improved significantly.

MEC has been intensively investigated in recent years. In [4], the authors consider a single user MEC system powered by wireless energy transfer. A closed-form offloading policy is derived via convex optimization. The authors in [5] extend the work to multi-user case and formulate the multi-user resource allocation problem as a convex optimization problem, and derive an insightful threshold-based optimal offloading strategy. Moreover, game-theory-based algorithms are designed to solve multi-user MEC offloading decision and resource allocation problems in [6], [7]. All the above works, however, only consider one-stage optimization. In many applications, the computation tasks cannot be fully offloaded within one channel coherent time, in which cases multi-stage optimizations should be done over stochastic processes. Several works have been done to tackle the stochastic optimization problems for MEC systems. In [8], the authors propose a Lyapunov-based stochastic control algorithm to deal with the long-term average energy consumption problem. Moreover, the authors in [9] solve the power constrained delay-optimal tasks scheduling problem for MEC system via Markov decision process approach. Unfortunately, these works only consider single-user MEC systems. For multi-user cases, the authors in [10] investigate the power-delay trade-off for multi-user MEC via Lyapunov optimization.

To the best of our knowledge, no one has investigated the case where the number of users in the system is varying over time. In this paper, we consider a more practical yet complex multi-user MEC system where new users may arrive in a stochastic manner with certain amount of computation tasks and the users who have finished their tasks will leave the system. The joint offloading decision, user selection and transmission power optimization problem is formulated as an infinite horizon discounted MDP problem which aims at minimizing the weighted sum of energy consumptions and latencies of mobile users. To avoid curse of dimensionality, we resort to approximate MDP approach where we propose a novel value function approximation method. Based on the approximate MDP, a low-complexity online scheduling algorithm is proposed. Simulation results demonstrate that our proposed scheme can achieve significant performance gains compared with the baseline schemes.

The remainder of this paper is organized as follows. In Section II, the system model will be introduced. The MDP problem formulation will be detailed in Section III, and the
approximate-MDP-based online scheduling algorithm will be illustrated in Section IV. Simulation results will be shown in Section V. Section VI concludes this paper.

II. SYSTEM MODEL

A. Network Model

We consider a mobile edge computing (MEC) system as illustrated in Fig. 1, where one base station (BS) serves a region $C$ and one MEC server is connected with the BS. Mobile devices with computation tasks arrive randomly in the service region $C$. Binary computation offloading model is adopted, and every task should be solely computed at either the mobile device or the MEC server. Thus, mobile devices can choose to compute the task locally or offload it to the MEC server via uplink transmission. As in [11], [4], [12], [10], [5], it is assumed that there are sufficiently many high-performance CPUs at the MEC server so that the computing latency at the MEC server can be neglected. Moreover, due to relatively smaller sizes of computation results, the downloading duration of computation results is also neglected as in [11], [12], [10], [5].

There are a number of mobile devices in the cell region $C$, which may be quasi-static, moving inside or out of the region $C$. The mobile devices with computation tasks are named as active devices in the remainder of this paper. The time axis of computation and uplink transmission scheduling is organized by frames each with a time duration of $T_s$ seconds. In each frame, there is at most one new active device arrived in the cell with probability $P_N \in (0, 1)$. We have no restriction on the distribution and mobility model of the mobile devices in the cell. Instead, the distribution density of the new active device is represented as $\lambda(1)$ for arbitrary location in the cell region $1 \in C$. Thus,

$$\int_C \lambda(1)ds(1) = 1,$$

and

$$\Pr[\text{New active device is in region } C'] = \int_{C'} \lambda(1)ds(1), \forall C' \subseteq C.$$

Moreover, it is assumed that the location of each active device is quasi-statistic in the cell when its task is being transmitted to the MEC server. The active devices become inactive when the computation of the tasks has been completed either locally or remotely at the MEC server.

Every new active device in the cell is assigned with a unique index. Let $U_E(t)$ and $U_L(t)$ be the sets of active devices in the $t$-th frame whose tasks are computed locally and at the MEC server respectively, $D_E(t) \subseteq U_E(t)$ and $D_L(t) \subseteq U_L(t)$ be the subsets of active devices whose computation tasks are accomplished in the $t$-th frame respectively, $n_t$ be the index of new active device arrived in the $t$-th frame. If there is no active device arrival in the $t$-th frame, $\{n_t\} = \emptyset$ where $\emptyset$ represents the empty set. On the other hand, if there is new active device in the $t$-th frame, the BS should determine if the computation task is calculated at the device or MEC server. Let $e_t \in \{0, 1\}$ represents the decision, where $e_t = 1$ means the task is offloaded to the MEC server and $e_t = 0$ means otherwise. Hence, the dynamics of active devices can be represented as

$$U_E(t + 1) = \begin{cases} U_E(t) \cup \{n_t\}/D_E(t) & \text{when } e_t = 1 \\ U_E(t) \cup \{n_t\}/D_E(t) & \text{otherwise} \end{cases}$$

(1)

and

$$U_L(t + 1) = \begin{cases} U_L(t) \cup \{n_t\}/D_L(t) & \text{when } e_t = 0 \\ U_L(t) \cup \{n_t\}/D_L(t) & \text{otherwise} \end{cases}$$

(2)

where operator “/” denotes the set subtraction.

B. Task Offloading Model

The input data for each computation task is organized by segments, each with $b_s$ information bits. Let $d_k$ be the number of input segments for the task at the $k$-th active device. It is assumed that the number of segments for each task is a uniformly distributed random integer between $d_{min}$ and $d_{max}$, i.e. $d_k \sim U(d_{min}, d_{max})$. For the computation tasks offloaded to the MEC server, the input data should be delivered to the BS via uplink transmission. Hence, one uplink transmission queue is established at the active devices of task offloading. Let $Q_E^k(t), \forall k \in U_E(t)$, be the number of segments in the uplink transmission queue of the $k$-th device at the beginning of the $t$-th frame, $\forall t, t$ with $\{n_t\} \neq \emptyset$ and $e_t = 1$, we have

$$Q_E^k(t + 1) = d_{n_t}.$$

In uplink, it is assumed that only one active device is selected for one uplink frame and the uplink transmission bandwidth is denoted as $W$. Let $H_k(t) = \sqrt{P_k(t)}h_k(t), \forall k \in U_E(t)$, be the uplink channel stat information (CSI) from the $k$-th active device to the BS, where $h_k(t)$ and $P_k(t)$ represent the small-scale and pathloss coefficients respectively. $h_k(t) \sim CN(0, 1)$ is complex Gaussian distributed with zero mean and variance $1$. Moreover, it is assumed that $h_k(t)$ is independently and identically distributed (i.i.d.) for different $t$. Let $p_k(t)$ be the uplink transmission power of the $k$-th active device if it is selected in the $t$-th frame. The uplink channel capacity of the $k$-th active device, if it is selected in the $t$-th frame, can be represented by

$$r_k(t) = W \log_2 \left(1 + \frac{p_k(t)P_k(t)h_k(t)^2}{\sigma_s^2} \right).$$
Furthermore, the corresponding number of segments that can be transmitted within one frame can be obtained by

$$\phi_k(t) = \left\lceil \frac{r_k(t)T_s}{b_s} \right\rceil,$$

where $\lceil C \rceil$ is the maximum integer less than $C$. Hence, let $a_t$ be the index of the selected uplink transmission device in the $t$-th frame, we have the following queue dynamics for all $k \in \mathcal{U}_E(t)$

$$Q_k^E(t + 1) = \begin{cases} Q_k^E(t) + \phi_k(t) & \text{if } k = a_t \\ Q_k^E(t) & \text{if } k \neq a_t. \end{cases}$$

where $\lceil C \rceil^+ = \max\{0, C\}$.

C. Local Computing Model

Following the computation models elaborated in [10], the number of CPU cycles for computing one bit of the input task data for the $k$-th active device is denoted as $L_k$, which is determined by the types of applications and can be measured before computation [13]. Denote the local CPU frequency of the $k$-th active device as $f_k$. We assume $L_k$ and $f_k$ are both uniformly distributed random integers, i.e. $L_k \sim \mathcal{U}(L_{\min}, L_{\max})$ and $f_k \sim \mathcal{U}(f_{\min}, f_{\max})$. One input data queue is established at the active devices with the decision of local computing. Let $Q_k^L(t)$, $\forall k \in \mathcal{U}_L(t)$, be the number of segments in the input data queue of the $k$-th device at the beginning of the $t$-th frame, for all $t$ with $\{n_t\} \neq \emptyset$ and $e_t = 0$, we have

$$Q_{n_t}^L(t + 1) = d_{n_t}.$$ 

Moreover, the queue dynamics at all active devices of local computing can be written as

$$Q_k^L(t + 1) = \left\lceil Q_k^L(t) - \frac{f_kT_s}{L_k} \right\rceil, \forall k \in \mathcal{U}_L(t).$$

Hence, the total computation time (measured in terms of frames) for $k$-th active device, whose task is computed locally, is given by

$$T_{\text{loc}}(d_k, f_k, L_k) = \left\lceil \frac{d_k b_s L_k}{f_k T_s} \right\rceil,$$

where $\lceil C \rceil$ is the minimum integer greater than or equal to $C$. Furthermore, the local computation power for $k$-th user can be obtained as

$$p_{\text{loc}}(f_k) = k f_k^3,$$

where $\kappa$ is the effective switched capacitance related to the CPU architecture[14].

III. PROBLEM FORMULATION

In this section, we shall formulate the optimization of the task offloading decision, uplink device selection and power allocation as an infinite-horizon MDP with discounted cost.

A. System State and Scheduling Policy

The system state and scheduling policy are defined as follows.

**Definition 1 (System State).** At the beginning of $t$-th frame, the state of the MEC system is uniquely specified by $S_t = (S_t^E, S_t^L, S_t^N)$, where

- $S_t^E$ specifies the system status regarding the task offloading, including the set of task offloading devices $\mathcal{U}_E(t)$, their uplink CSI $\mathcal{H}_E(t) \triangleq \{H_k(t) | k \in \mathcal{U}_E(t)\}$ and pathloss coefficients $\mathcal{G}_E(t) \triangleq \{\rho_k(t) | k \in \mathcal{U}_E(t)\}$, and their uplink queue state information (QSI) $\mathcal{Q}_E(t) \triangleq \{Q_k^E(t) | k \in \mathcal{U}_E(t)\}$.

- $S_t^L$ specifies the system status regarding the local computing, including the set of local computing devices $\mathcal{U}_L(t)$, the application-dependent parameters $L(t) \triangleq \{L_k(t) | k \in \mathcal{U}_L(t)\}$, their CPU frequencies $\mathcal{F}(t) \triangleq \{f_k(t) | k \in \mathcal{U}_L(t)\}$, and their QSI $\mathcal{Q}_L(t) \triangleq \{Q_k^L(t) | k \in \mathcal{U}_L(t)\}$.

- $S_t^N$ specifies the system status regarding the new active device, including the new user arrival indicator $I_N(t)$, its index $n_t$, pathloss coefficient $\rho_{n_t}(t)$, size $d_{n_t}$, CPU frequency $f_{n_t}$ and $L_{n_t}$.

**Definition 2 (Scheduling Policy).** The scheduling policy $\Omega(S_t) \triangleq \{a(t), p(t), e(t)\}$ is a mapping from the system state $S_t$ to control actions, i.e. the index $a(t)$ of the selected transmitting user at $t$-th frame, the transmission power $p(t)$ and the offloading decision $e(t)$ for the potential new arriving user.

B. Problem Formulation of MEC Scheduling

For an MEC system, the two primary performance metrics are latency and energy consumption. In this paper, we aim at minimizing the weighted sum of latency and energy consumption for mobile users.

According to Little’s law, the average latency is proportional to the average number of users in the system. Thus, the total expected system cost with discount factor $\gamma$ can be expressed as

$$\overline{C} \triangleq \lim_{T \to +\infty} \mathbb{E} \left[ \sum_{t=1}^{T} \gamma^{t-1} \left( |\mathcal{U}_E(t)| + |\mathcal{U}_L(t)| + w[p(t) + \sum_{k \in \mathcal{U}_L(t)} p_{\text{loc}}(f_k)] \right) \right]$$

$$\triangleq \lim_{T \to +\infty} \mathbb{E} \left[ \sum_{t=1}^{T} \gamma^{t-1} g(t) \right],$$

where $w$ is the weighting factor to balance latency cost and power consumption cost. $g(t)$ denotes the one-stage cost.

Given a system state, we are aiming at minimizing the long-term system cost by optimizing the control policy. Hence, the overall optimization problem is defined below.

**Problem 1 (Overall Scheduling Problem).**

$$\min_{\Omega} \overline{C}(\Omega) = \min_{\Omega} \lim_{T \to +\infty} \mathbb{E} \left[ \sum_{t=1}^{T} \gamma^{t-1} g(t) \right].$$ (9)
Problem 1 is an infinite horizon discounted cost MDP problem. Its optimal solution can be achieved by solving the following Bellman’s equation.

\[ V(S_t) = \min_{\Omega(S_t)} \left[ g(t, \Omega(S_t)) + \sum_{S_{t+1}} \gamma \Pr(S_{t+1} | S_t, \Omega(S_t))V(S_{t+1}) \right]. \]  

(10)

The above Bellman’s equation can be solved by standard dynamic programming approaches such as value iteration (VI) and policy iteration (PI). In our problem, however, the tremendous state space and action space make traditional VI and PI approaches computationally impractical.

Note that once a user is scheduled for local computing, its cost is predictable and can be solely calculated, and then it can be removed from the system state. Moreover, since the small-scale fading and the dynamics of the new arrival users are i.i.d. in each frame, they can be expected out in the Bellman’s equation. Thus, the state space of the original Bellman’s equation can be reduced as elaborated in the following lemma whose proof is straightforward and neglected here.

**Lemma 1 (Bellman’s Equation with Reduced State Space).**

\[ \mathbb{E}_{H,N} V(S_t) = W(S_t) = \min_{\Omega(S_t)} \left[ G(t, \Omega(S_t)) + \sum_{S_{t+1}} \gamma \Pr(S_{t+1} | S_t, \Omega(S_t))W(S_{t+1}) \right] \]  

(11)

where the expectation is taken over the CSI of cloud computing users and the dynamics of potential new arriving users. \( S_t \triangleq [U_E(t), \mathcal{G}_E(t), \mathcal{Q}_E(t)] \) is defined as abstract system state. \( G(t) = |U_E(t)| + I_N e(t) + w_p(t) + I_N(t)\mathbb{E}[C(n_t)] \) is the transformed one-stage cost after using predicted local computing cost for the new arriving user \( n_t \) and \( C(n_t) = \sum_{k=1}^{T(d_{n_t}, f_{n_t}, L_{n_t})} \gamma t \} + w_p(t) + I_N(t)\mathbb{E}[C(n_t)] \) is total cost of user \( n_t \) if it is scheduled for local computing.

Hence, the optimal policy can be obtained by the following problem.

**Problem 2 (Optimal Policy).**

\[ \Omega^*(S_t) = \arg \min_{\Omega(S_t)} \left[ G(t, \Omega(S_t)) + \sum_{S_{t+1}} \gamma \Pr(S_{t+1} | S_t, \Omega(S_t))W(S_{t+1}) \right] \]  

(12)

IV. LOW-COMPLEXITY SOLUTION

In this section, we first introduce a heuristic scheduling policy as a baseline policy under which the value function can be derived analytically. Then one-step policy iteration is applied based on the above value function. It can be proved that the new policy has better performance than the baseline scheduling policy.

**A. Baseline Scheduling Policy**

The baseline scheduling policy is elaborated below.

**Policy 1 (Baseline Scheduling Policy II).** Given the system state \( \tilde{S}_t \), the scheduling actions \( \{a_t, p(t), e(t)\} \) are provided below:

- **User selection** \( \{a_t | \forall t\} \) shall be according to first-come-first-serve rule (FCFS). BS only schedules the user with the largest waiting time.
- **The transmission power** \( \{p(t) | \forall t\} \) shall compensate the large-scale fading (link compensate). Thus, the transmission power for the selected user is given by

\[ p(t) = \frac{p_r}{\rho_{a_t}}, \forall t, \]  

(13)

where \( p_r \) is the average receiving power at the BS.
- **The new arrival shall be scheduled for local computing** when the transmission of remaining offloading computation tasks is not finished. When remaining offloading computation tasks are transited completely, the new arrival shall be scheduled for edge computing, i.e.

\[ e(t) = 1 \left( U_E(t) = \emptyset \right), \forall t, \]  

(14)

where \( I(\cdot) \) is the indicator function.

**B. Value Function Corresponding to the Baseline Policy II**

Let \( W_{\Pi}(\cdot) \) be the value function corresponding to the baseline scheduling policy II. Given initial abstract system state \( \tilde{S}_t \), \( W_{\Pi}(\tilde{S}_t) \) is given by

\[ W_{\Pi}(\tilde{S}_t) = \lim_{T \to +\infty} \mathbb{E} \left[ \sum_{\tau=1}^{T} \gamma^{\tau-1} G(\tau, \Pi(S_{\tau})) \big| S_t \right], \forall \tilde{S}_t. \]  

(15)

In order to derive the value function \( W_{\Pi}(\tilde{S}_t) \) analytically, \( U_E(t) \) is a ordered sequence of user indexes, \( k = 1, 2, \ldots, |U_E(t)| \), where the \( k \)-th user in \( U_E(t) \) is denoted as \( e_{k,t} \). Let \( T_{k,t} \) be the number of frames for completing the uplink transmission of the \( e_{k,t} \)-th user. Hence, we have the following approximation

\[ W_{\Pi}(\tilde{S}_t) \approx \mathbb{E} \left[ T_{e_{1,t}} \big| \forall i, t \right] \sum_{k=1}^{T_{e_{1,t}}} \frac{\sum_{T_{k,t}} \gamma^{T_{k,t}} p_r}{1 - \gamma} + \gamma^{T_{e_{1,t}}} - \gamma^{T_{e_{1,t}}} \]  

\[ + \sum_{k=1}^{T_{e_{1,t}}} \gamma^{T_{k,t}} \]  

(16)

where (II) is system cost raised by the new arrival users within next \( \left( \sum_{t=1}^{T_{e_{1,t}}} T_{k,t} \right) \)-th frames, (III) is system cost raised by
Expression of Lemma 2 obtained analytically by the following lemma.

Define \( \bar{S}_1 \triangleq [U_E(1) = 0, G_E(1) = 0, Q_E(1) = \emptyset] \). The value function corresponding to \( \bar{S}_1 \) under policy II can be obtained analytically by the following lemma.

Lemma 2 (Analytical Expression of \( W_{II}(\bar{S}_1) \)). The Analytical Expression of \( W_{II}(\bar{S}_1) \) is given by

\[
W_{II}(\bar{S}_1) = \sum_{t=1}^{\infty} g^T (\gamma M)^{t-1} u = g^T (I - \gamma M)^{-1} u,
\]

where \( u \in \mathbb{R}^{(d_{\text{max}}+1) \times 1} \), \( u_1 = 1 \) and \( u_i = 0 \), for \( i = 2, 3, ..., d_{\text{max}} \).

\( g_i = 1 + w \mathbb{E}_{p_{\text{h},i}}[\bar{p}_i] + P_N \mathbb{E}_{d_{\text{h},i}}[C(n_t)] \), \( \gamma = 2, 3, ..., d_{\text{max}} + 1 \). \( M \in \mathbb{R}^{(d_{\text{max}}+1) \times (d_{\text{max}}+1)} \) denotes transition probability matrix whose entries are given by

- \( M_{1,1} = 1 - P_N \),
- \( M_{1,j} = 0 \), for \( j = 2, 3, ..., d_{\text{max}} \),
- \( M_{i,j} = \frac{p_{\text{h},i}}{p_N} \), for \( j = d_{\text{max}} + 1, 2, ..., d_{\text{max}} + 1 \),
- \( M_{i,j} = 0 \), for \( 1 < i < j \),
- \( M_{i,j} = \exp\left\{ -\frac{[2^{(j-i)H_{\text{h},j}}/(W_{T_s}) - 1] \sigma^2}{p_r} \right\} \) - \( \exp\left\{ -\frac{[2^{(j-i)H_{\text{h},j}}/(W_{T_s}) - 1] \sigma^2}{p_r} \right\} \), for others.

Proof. Please refer to Appendix A.

Based on Lemma 2, we have the following lemma to achieve the asymptotic value function corresponding to the initial abstract state \( \bar{s} \) under the baseline policy II.

Lemma 3 (Asymptotic Expression of Value Function \( W_{II}(\bar{s}) \)). The asymptotic form of \( W_{II}(\bar{s}) \) can be derived as

\[
W_{II}(\bar{s}) \rightarrow W_{II}(\tilde{s}) \triangleq \sum_{k=1}^{\sum_{k=1}^{[\delta_{|t|}]} - \frac{\delta_{|t|}}{1-\gamma} - \frac{\delta_{|t|}}{1-\gamma} - \frac{\delta_{|t|}}{1-\gamma} - \frac{\delta_{|t|}}{1-\gamma}}
+ P_N \mathbb{E}_{d_{\text{h},i}}[\gamma \sum_{k=1}^{[\delta_{|t|}]} T_{k,t} W_{II}(\tilde{S}_1),
\]

where \( T_{k,t} \) is the asymptotic transmission time for \( e_{k,t} \)-th user.

Proof. Please refer to appendix B.

Since the transmission time of one task is much larger than the channel coherent time, it is assumed that the ergodic channel capacity, which averages all possible small-scale channel fading, can be achieved during one task uplink transmission. Hence, we have

\[
T_{k,t} = \frac{Q_{e_{k,t}}(t)}{W_{\delta_{|t|},b} \frac{L_{\text{h}} + b_{\text{h},i}}{\gamma}},
\]

\[\triangleq \tilde{T}_{k,t}, \forall k, t,\]

C. Online Scheduling

With the asymptotic value function corresponding to the baseline policy \( \tilde{W}_{II}(\bar{s}) \), the optimization problem in \( \text{2} \) can be written as

\[
\text{Problem 3 (Online Scheduling Problem)}.
\]

\[\Pi^* = \arg \min_{\Omega(S_t)} \left\{ G(t, \Omega(S_t)) + \sum_{S_{t+1}} \mathbb{E}_{H_{II}} \left[ \tilde{W}_{II}(S_{t+1}) \right] \right\}.
\]

Note that this is an integrated continuous and discrete optimization problem, its solution algorithms are given below.

Algorithm 1 (Online Scheduling with \( \tilde{W}_{II}(\cdot) \) when \( I_N(t) = 0 \). At the \( t \)-th frame, there is no new arrival, \( e(t) = 0 \). The solution of Problem 3 can be obtained below

- Step 1: For each \( k \in U_E(t) \), calculate

\[
\tilde{G}_k^* = \min_{p_k(t)} \left \{ wp_k(t) \right \} + \sum_{S_{t+1}} \gamma \mathbb{E}_{H_{II}} \left[ \tilde{W}_{II}(S_{t+1}) \right].
\]

\[
\tilde{G}_k^* = \min_{p_k(t)} \left \{ wp_k(t) \right \} + \sum_{S_{t+1}} \gamma \mathbb{E}_{H_{II}} \left[ \tilde{W}_{II}(S_{t+1}) \right] .
\]

The optimal uplink power for the above problem are denoted as \( p_k^*(t) \).

- Step 2: Let \( k^* = \arg \min_k \tilde{G}_k^* \), the solution of Problem 3 is given by \( \{ e(t) = 0, a_t = k^*, p_k(t) = p_k^*(t) \} \).

Algorithm 2 (Online Scheduling with \( \tilde{W}_{II}(\cdot) \) when \( I_N(t) = 1 \). At the \( t \)-th frame, there is one new user arrival.

The solution of Problem 3 can be obtained below

- Step 1: Assume \( e(t) = 0 \), for each \( k \in U_E(t) \), calculate

\[
\tilde{G}_k^* = C(n_t) + \min_{p_k(t)} \left \{ wp_k(t) \right \} + \sum_{S_{t+1}} \gamma \mathbb{E}_{H_{II}} \left[ \tilde{W}_{II}(S_{t+1}) \right].
\]

\[
\tilde{G}_k^* = \min_{p_k(t)} \left \{ wp_k(t) \right \} + \sum_{S_{t+1}} \gamma \mathbb{E}_{H_{II}} \left[ \tilde{W}_{II}(S_{t+1}) \right].
\]

\[
\text{Step 2: Assume } e(t) = 1, \text{ for each } k \in U_E(t), \text{ calculate}
\]

\[
\tilde{G}_k^* = \min_{p_k(t)} \left \{ wp_k(t) \right \} + \sum_{S_{t+1}} \gamma \mathbb{E}_{H_{II}} \left[ \tilde{W}_{II}(S_{t+1}) \right].
\]

(24)
• **Step 3:** If \( \min_k \hat{G}_E^k < \min_k G_E^k \), the solution of Problem 2 is given by \( \{ e(t) = 1, a_t = k'_E, p_k(t) = p_{k'}^e(t) \} \), where \( k'_E = \arg \min_k \hat{G}_E^k \). Otherwise, the solution of Problem 2 is given by \( \{ e(t) = 0, a_t = k''_E, p_k(t) = p_{k''}^e(t) \} \), where \( k''_E = \arg \min_k G_E^k \).

**Remark 1 (Complexity Analysis).** Our proposed algorithms achieve significantly lower complexity than traditional value iteration approach. For traditional value iteration method, the time complexity is \( \mathcal{O}(|A|^S) \) and the space complexity is \( \mathcal{O}(|S||A|) \), where \( \mathcal{O}(|S|) \) and \( \mathcal{O}(|A|) \) are the number of system states and actions respectively. For our proposed algorithms, the time complexity is \( \mathcal{O}(|U(t)||N_p) \) and the space complexity is \( \mathcal{O}(1) \), where \( N_p \) is the number of quantization levels of transmission power.

**D. Bounds Analysis**

**Lemma 4 (Upper Bound of Value Function).** The value function \( W(S_t) \) is upper-bounded as

\[
W(S_t) \leq W_{\Pi}(S_t), \forall S_t. \tag{25}
\]

**Proof.** Baseline scheduling policy II is not an optimal scheduling, the fact that approximated value function \( W_{\Pi}(S_t) \) is the upper-bound can be straightforward. \( \square \)

**Corollary 1 (Asymptotic Upper Bound of Value Functions).** Note that \( \hat{W}_{\Pi}(S_t) \) is the asymptotic form of \( W_{\Pi}(S_t) \), we have

\[
W(S_t) \leq \hat{W}_{\Pi}(S_t), \forall S_t, \text{ when } T_{k,t} \to \hat{T}_{k,t}. \tag{26}
\]

**Lemma 5 (Policy Improvement).** Policy II’ satisfies \( W_{\Pi'}(S_t) \leq W_{\Pi}(S_t), \forall S_t \).

**Proof.** Similar to the proof of Policy Improvement Property is chapter II of [15]. \( \square \)

**Corollary 2 (Asymptotic Performance Guarantee).** When \( T_{k,t} \to \hat{T}_{k,t} \), we have \( W_{\Pi}(S_t) \to \hat{W}_{\Pi}(S_t) \). Thus, we have \( W_{\Pi'}(S_t) \leq \hat{W}_{\Pi}(S_t), \forall S_t \).

**V. SIMULATION RESULTS**

In this section, we evaluate the performance of the proposed online scheduling algorithm by numerical simulations. In our simulation, we set the frame duration \( T_s = 10 \text{ ms} \). We assume the input data size of each task is uniformly distributed between 200 to 300 segments, each of a size of 10 Kb. Local CPU frequency is set to 1GHz and 500 CPU cycles are needed to compute one bit of input data. The effective switched capacitance \( \kappa = 10^{-28} \). In addition, we set uplink bandwidth \( W = 10 \text{ MHz} \), noise power \( \sigma^2 = -104 \text{ dBm} \).

We compare our proposed scheduling policy with three benchmark policies including the baseline policy [7] (BSL) as elaborated in section IV-A; all local computing policy (ALC), where all the users execute their tasks locally; and all cloud computing policy (ACC), where all the users offload their tasks to the MEC server for cloud computing.

Fig. 2 shows the average per-user costs versus different user arrival rates under our proposed scheduling policy and three benchmark policies. It can be observed from the figure that the average per-user costs achieved by all the policies grow with the increase of user arrival rate except for the ALC policy. For ALC policy, since all the users computed their tasks on their own devices locally, the user arrival rate has no influence on the average per-user cost. For ACC policy, the average per-user cost grows exponentially with the increase of user arrival rate due to limited wireless transmission capability. It is also shown that our proposed policy always outperforms the BSL policy especially in a reasonable user arrival rate region (0-0.2), where the average per-user cost is reduced by 40% ~ 70%. Besides, it can be seen that as the user arrival rate becomes sufficiently large, the cost of both BSL policy and our proposed policy will converge to the cost of ACC policy. This observation can be explained by Fig. 3 which shows that the ratio of cloud computing users tends to 0, i.e. the ratio of local computing users tends to 1 as the user arrival rate becomes sufficiently large. This also indicates that the BSL
policy and our proposed policy can stabilize the system even when the user arrival rate becomes incredibly large. Moreover, as shown in Fig. [3] the ratio of cloud computing users of our proposed policy is remarkably lager than that of BSL policy. Hence, our proposed policy can better exploit cloud computing to reserve energy and reduce latency.

VI. CONCLUSION

In this paper, we investigate the multi-user mobile edge computing system where the number of users in our system is varying over time. The problem is formulated as an MDP problem. To avoid the curse of dimensionality, we propose a novel value function approximation approach. Based on the approximated value function, one-step policy iteration is applied to derive a refined policy. We theoretically demonstrate that our proposed policy has performance improvement compared with the baseline policy. Also, simulation results validate that our proposed policy significantly outperforms the baseline policy and several other heuristic policies. In the future, we may consider weak cloud computing capability where cloud computing time need to be considered. Besides, the fairness among users is another point of interest which calls for further investigation.

APPENDIX

APPENDIX A: PROOF OF LEMMA[2]

Under the baseline policy Π, from state \( S_1 \), there will be at most one cloud computing user. The system state is reduced to only have the QSI of the cloud computing users after expected the states that are i.i.d in each frame. Thus, the system state becomes \( \{0, 1, \ldots, d_{\text{min}}, \ldots, d_{\text{max}}\} \). \( u \) is the probability distribution vector for each state from initial state \( S_1 \). \( M \) is the transition probability matrix and \( g \) is the expected one-stage cost of each state. The idea of equation [18] is to sum up the expected cost of each frame to obtain the value function \( W_\Pi(\tilde{S}_1) \). Here, we briefly explain how to derive each entry of the transition probability matrix \( M \):

- \( i = 1, j = 1 \). Transiting from 1st state (0 segment) to 1st state means that there is no new user arrival. Hence \( M_{1,1} = 1 - P_N \).
- \( i = 1, j = 2, 3, \ldots, d_{\text{min}} \). Since to minimum number of segments of a new user’s task is \( d_{\text{min}} \), there is no chance of transiting from 0 segment to \( (1, 2, \ldots, d_{\text{min}} - 1) \) segments. Hence, \( M_{1,j} = 0 \).
- \( i = 1, j = d_{\text{min}} + 1, d_{\text{min}} + 2, \ldots, d_{\text{max}} + 1 \). This means there is a new user arrival. The probability of a new user arrival is \( P_N \) and the task size of the new user is uniformly distributed between \( d_{\text{min}} \) to \( d_{\text{max}} \). Thus, the probability of transiting from 1st state (0 segment) to \( j \)-th state (\( j \)-1 segments) for \( j = d_{\text{min}} + 1, d_{\text{min}} + 2, \ldots, d_{\text{max}} + 1 \) is \( M_{1,j} = \frac{P_N}{d_{\text{max}} - d_{\text{min}} + 1} \).
- \( 1 < i < j \leq d_{\text{max}} + 1, i > 1 \) indicates that the current cloud computing queue is not empty. Hence, the cloud computing queue will not increase since the new user will be scheduled for local computing under policy \( \Pi \). Therefore, \( M_{i,j} = 0 \), for \( \forall 1 < i < j \).
- \( i = 2, 3, \ldots, d_{\text{max}} + 1, j = 1 \). This means that the current cloud computing user will finish transmitting the remaining (\( i \)-1) segments with current frame. Hence, \( M_{i,1} = \Pr \left( W \log_2 (1 + \frac{b_s i^2}{\sigma^2}) \geq (i - 1) b_s \right) = \exp \left( \frac{(2^{(i-1)b_s/(WT_s)} - 1)}{\epsilon^2} \right) \).
- \( d_{\text{max}} + 1 \leq i \geq j > 1 \). This means that the cloud computing user will transmit (\( i \)-j) segments within current frame. Hence, \( M_{i,j} = \Pr \left( (i - j)b_s \leq W \log_2 (1 + \frac{p_r b_s}{\sigma^2}) \leq (i - j + 1)b_s \right) = \exp \left( \frac{2^{(i-j)b_s/(WT_s)} - 1}{\epsilon^2} \right) - \exp \left( - \frac{2^{(i-j-1)b_s/(WT_s)} - 1}{\epsilon^2} \right) \).

APPENDIX B: PROOF OF LEMMA[3]

With asymptotic transmission time \( T_{k,t} \) the asymptotic form of (16) can be written as

\[
W_\Pi(\tilde{S}_1) \rightarrow W_\Pi(\tilde{S}_1) \approx \sum_{k=1}^{d(t_1)} \sum_{t=1}^{T} \left[ 1 - \gamma \hat{r}_{k,t} \right] \frac{1}{1 - \gamma} \gamma^{t-1} \left( n_t \right) + \lim_{T \rightarrow +\infty} \sum_{k=1}^{d(t_1)} \left( 1 - \gamma \right) \frac{1}{1 - \gamma} \gamma^{t-1} G(t, \Pi(S_t))
\]

Defined as \( r(\tilde{S}_1) \)

\[
R(\tilde{S}_1) = \lim_{T \rightarrow +\infty} \sum_{k=1}^{d(t_1)} \hat{r}_{k,t} E_{H,N} \left[ \sum_{t=1}^{T} \gamma^{t-1} G(t, \Pi(S_t)) \right] = \gamma \sum_{k=1}^{d(t_1)} \hat{r}_{k,t} W_\Pi(\tilde{S}_1)
\]

REFERENCES

[1] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, “A survey on mobile edge computing: The communication perspective,” IEEE Communications Surveys Tutorials, vol. 19, no. 4, pp. 2322–2358, Fourthquarter 2017.

[2] T. X. Tran, A. Hajisami, P. Pandey, and D. Pompili, “Collaborative mobile edge computing in 5G networks: New paradigms, scenarios, and challenges,” IEEE Communications Magazine, vol. 55, no. 4, pp. 54–61, April 2017.

[3] M. T. Beck, S. Feld, C. Linnhoff-Popien, and U. Pitzschler, “Mobile edge computing,” Informatik-Spektrum, vol. 39, no. 2, pp. 108–114, Apr 2016. [Online]. Available: https://doi.org/10.1007/s00287-016-0957-6

[4] C. You, K. Huang, and H. Chae, “Energy efficient mobile cloud computing powered by wireless energy transfer,” IEEE Journal on Selected Areas in Communications, vol. 34, no. 5, pp. 1757–1771, May 2016.

[5] C. You and K. Huang, “Multiuser resource allocation for mobile-edge computation offloading,” global communications conference, pp. 1–6, 2016.

[6] X. Chen, “Decentralized computation offloading game for mobile cloud computing,” IEEE Transactions on Parallel and Distributed Systems, vol. 26, no. 4, pp. 974–983, April 2015.

[7] X. Chen, L. Jiao, W. Li, and X. Fu, “Efficient multi-user computation offloading for mobile-edge cloud computing,” IEEE/ACM Transactions on Networking, vol. 24, no. 5, pp. 2795–2808, October 2016.

[8] D. Huang, P. Wang, and D. Niyato, “A dynamic offloading algorithm for mobile computing,” IEEE Transactions on Wireless Communications, vol. 11, no. 6, pp. 1991–1995, June 2012.
[9] J. Liu, Y. Mao, J. Zhang, and K. B. Letaief, “Delay-optimal computation task scheduling for mobile-edge computing systems,” in 2016 IEEE International Symposium on Information Theory (ISIT), July 2016, pp. 1451–1455.

[10] Y. Mao, J. Zhang, S. Song, and K. B. Letaief, “Power-delay tradeoff in multi-user mobile-edge computing systems,” global communications conference, pp. 1–6, 2016.

[11] Y. Mao, J. Zhang, and K. B. Letaief, “Dynamic computation offloading for mobile-edge computing with energy harvesting devices,” IEEE Journal on Selected Areas in Communications, vol. 34, no. 12, pp. 3590–3605, Dec 2016.

[12] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, and D. O. Wu, “Energy-optimal mobile cloud computing under stochastic wireless channel,” IEEE Transactions on Wireless Communications, vol. 12, no. 9, pp. 4569–4581, Sep. 2013.

[13] A. P. Miettinen and J. K. Nurminen, “Energy efficiency of mobile clients in cloud computing,” in Usenix Conference on Hot Topics in Cloud Computing, 2010, pp. 4–4.

[14] T. D. Burd and R. W. Brodersen, “Processor design for portable systems,” Journal of Vlsi Signal Processing Systems for Signal Image and Video Technology, vol. 13, no. 2-3, pp. 203–221, 1996.

[15] D. P. Bertsekas, Dynamic Programming and Optimal Control. Athena Scientific, 2000.