

Service Capacity Enhanced Task Offloading and Resource Allocation in Multi-Server Edge Computing Environment

Wei Du^{*†}, Tao Lei^{*}, Qiang He[‡], Wei Liu^{*†}, Qiwan Lei^{*}, Hailiang Zhao^{*}, Wei Wang[§]

^{*} Wuhan University of Technology, Wuhan, China

{whutduwei, leitao1995, wliu, leiqliwang} @whut.edu.cn, hliangzhao97@gmail.com

[†] Hubei Key Laboratory of Transportation Internet of Things, Wuhan, China

[‡] Swinburne University of Technology, Melbourne, Australia

qhe@swin.edu.au

[§] East China Normal University, Shanghai, China

wwang@dase.ecnu.edu.cn

Abstract—An edge computing environment features multiple edge servers and multiple service clients. In this environment, mobile service providers can offload client-side computation tasks from service clients' devices onto edge servers to reduce service latency and power consumption experienced by the clients. A critical issue that has yet to be properly addressed is how to allocate edge computing resources to achieve two optimization objectives: 1) minimize the service cost measured by the service latency and the power consumption experienced by service clients; and 2) maximize the service capacity measured by the number of service clients that can offload their computation tasks in the long term. This paper formulates this long-term problem as a stochastic optimization problem and solves it with an online algorithm based on Lyapunov optimization. This NP-hard problem is decomposed into three sub-problems, which are then solved with a suite of techniques. The experimental results show that our approach significantly outperforms two baseline approaches.

Index Terms—edge computing; multi-server task offloading; service capacity enhancement; Lyapunov optimization

I. INTRODUCTION

Edge computing has emerged as a new paradigm for powering applications by offering computing, storage and networking resources at the edge of the cloud [1], [2]. As illustrated on the 5G standardization roadmap, edge servers will be distributed at ultra-dense small-cell base stations (CBS) [3], [4]. In such an environment, the coverages of adjacent edge servers partially overlap to avoid blank areas not covered by any edge servers [5]. Service providers can offload client-side computation tasks from service clients' devices onto edge servers to reduce the service cost measured by the service latency and power consumption experienced by service clients [6], [7]. This is referred to as *task offloading*.

The edge infrastructure provider looks at task offloading from two general perspectives. A large number of edge servers sharing service clients offloaded computation tasks can provide service clients with low service cost from service clients' perspective. That is the benefit of computation loading. However, the deployment of excessive edge servers result in overly high

operational cost from edge infrastructure provider's perspective. After all, the edge infrastructure provider is interested in the overall revenue, i.e., the benefit minus the cost. The economy of scale maximizes the edge infrastructure providers revenue by maximizing the service capacity, i.e., the ability to serve the maximum number of service clients. To achieve a cost-effective solution, the service cost needs to be traded off to allow more service clients to be served by edge servers. Thus, how to trade off the service cost and service capacity is a critical problem in edge computing.

A lot of researchers have focused on solving the task offloading problem in a single-server edge computing environment. However, in a particular area, there are usually multiple edge servers available for offloading service clients' computation tasks. The edge computing environment is in fact a multi-server environment. In such an environment, it is critical and challenging to optimize the utilization of edge computing resources, including computation and transmission resources, with the aim to maximize the service capacity while minimizing the service cost. Firstly, the characteristics of latency-tolerant applications, i.e., the coupling among randomly-arrived tasks, must be captured [8]. Stochastic computation partitioning strategies must be formulated to split the resources to be shared among service clients. Secondly, each service client needs to decide not only how to partition its computation tasks between its device and the edge server but also which edge server to offload its computation tasks to.

In this paper, we propose a holistic solution to computation offloading and resource allocation in multi-server edge computing environment with the aim to serve as many service clients as possible with minimum service cost. The main contributions of this paper are as follows:

- Task offloading in such an environment is modelled as a stochastic optimization problem with multiple optimization objectives. It aims to maximize the number of service clients served with minimum service cost in the long term.

- Based on Lyapunov optimization, the above long-term stochastic optimization problem is converted to a deterministic optimization problem within each time slot, which is then further decomposed into three sub-problems.
- To solve the optimization problem, an online joint task offloading and resource allocation algorithm (OJTORA) powered by a suite of techniques is proposed to solve each sub-problem with low complexity.
- Extensive experiments are conducted to evaluate OJTORA. The results show that OJTORA outperforms the baseline approaches significantly.

The remainder of this paper is organized as follows. Section II reviews related work. Section III presents the edge computing model. Section IV formulates the research problem. Section V introduces the proposed approach. Section VI experimentally evaluates the proposed approach. Section VII concludes this paper.

II. RELATED WORKS

In recent years, joint task offloading and resource allocation in edge computing has attracted many researchers attention. Most existing work studied the problem in a single-server edge computing environment [8]–[11]. Some researchers have considered multi-server scenarios [12]–[18]. The authors of [12] optimize task offloading, uplink transmission power of service clients, and computing resource allocation on edge servers to minimize task completion time and power consumption. In [13], the tradeoff between latency and power consumption was studied. The problem was formulated as computation and transmits power minimization subject to latency and reliability constraints. The authors of [14] studied how to minimize mobile power consumption through data offloading. Centralized and distributed algorithms for power allocation and transmission channel assignment were proposed. In [15], device-edge-cloud edge was investigated. A network-aware multi-user and multi-edge computation partitioning problem was formulated. Computation and radio transmission resources were allocated such that service clients average throughput was maximized. In [16], the shareable storage was considered, and a constant-factor approximation algorithm was proposed to decide the server placement and resource allocation. In [17], the edge task offloading control was investigated in cloud radio access network (C-RAN) environments, and a multi-stage heuristic was proposed to minimize the refusal rate for users task offloading requests. In [18], an online algorithm was proposed to allocate edge server's resource over time.

In the studies of task offloading, service latency and power consumption have been commonly acknowledged as two very important optimization objectives. However, few researchers have considered service capacity, i.e., the total number of service clients served, which is a key issue from the edge infrastructure providers perspective in a multi-server edge computing environment [5]. In [19], A quality-of-experience (QoE) driven LTE downlink scheduling scheme for VoIP applications was proposed to improve the service capacity with

acceptable QoE for all service clients. However, they only consider single-server environments. In addition, the proposed LTE downlink scheduling scheme is specifically designed for VoIP applications and thus is not applicable to most other applications in the edge computing environment.

Our work differentiates from existing work in the following ways. First, we consider a multi-server edge computing environment. Second, our approach is applicable to most latency-tolerant applications in the edge computing environment. Third, we consider both resource allocation and task offloading. Finally, we attempt to achieve the optimization objectives in the long term.

III. SYSTEM MODEL

An example multi-server edge computing scenario is shown in Fig. 1. A service client can offload some or all of its computation tasks to one of its nearby edge servers. Let us denote the set of service clients with U , the set of edge servers with S , the connection between the i th service client and the j th edge server with $c_{i,j} = \{0, 1\}$, where $c_{i,j} = 1$ indicates that the i th service client can access the j th edge server or $c_{i,j} = 0$ otherwise. Let us define $G_i = \{j | c_{i,j} = 1, j \in S\}$, $i \in U$, and $Z_j = \{i | c_{i,j} = 1, i \in U\}$, $j \in S$. Task offloading in the edge computing environment is an ongoing process. Let us denote different time slots with $T = \{1, 2, 3, \dots\}$ and the time slot length is τ . The available bandwidth of each edge server is ω Hz and the noise power spectral density is N_0 .

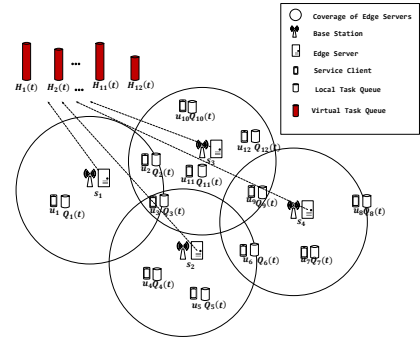


Fig. 1. A multi-server edge environment

A. Computation Task and Task Queue Model

The computation tasks running on service clients' devices are bit-wise independent [1]. Let us denote the amount of tasks of the i th service client in the t th time slot as $A_i(t)$, which are independent and identically distributed in different time slot within $[0, A_{i,max}]$, $A_{i,max} \in \mathbb{R}^+$, with the expectation $\mathbb{E}[A_i(t)] = \lambda_i$, $\lambda_i \in [0, A_{i,max}]$, $i \in U$. As shown in Fig. 2, at the beginning of the t th time slot, the local queue length of the i th service client is $Q_i(t)$. At t th time slot, the amount of tasks which has arrived but not been executed locally or offloaded will be put in $Q_i(t)$. Within the t th time slot, the i th service client will locally execute $D_{l,i}(t)$ tasks and will offload $D_{r,i}(t)$ tasks to an edge server, i.e., $D_{\sum,i}(t) = D_{l,i}(t) + D_{r,i}(t)$.

$$Q_i(t+1) = \max\{Q_i(t) - D_{\sum,i}(t), 0\} + A_i(t), i \in U. \quad (1)$$

As shown in Fig. 2, a service client chooses only one edge server to offload its tasks in each time slot. Thus, we maintain a virtual task queue $H_i(t)$ for each service client, which represents the amount of tasks offloaded but not been executed in all edge servers. In the t th time slot, let us denote the amount of tasks of the i th service client that has been offloaded but not executed by the j th edge server as $H_{i,j}(t)$, where $H_i(t) = \sum_{j \in G_i} H_{i,j}(t)$, the amount of tasks of the i th service client which has been executed by the edge server as $D_{s,i}(t)$.

$$H_i(t+1) = \max\{H_i(t) - D_{s,i}(t), 0\} + D_{r,i}(t), i \in U. \quad (2)$$

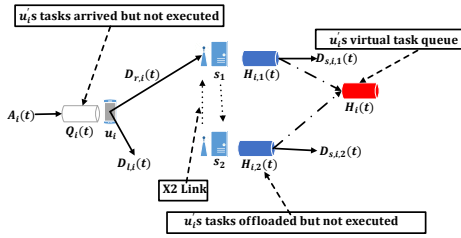


Fig. 2. The task queues of edge system

B. Local Execution and Task Offloading Model

The i th service client's device executes 1bit task using L_i (cycles/bit) CPU cycles [20], [21]. The dynamic frequency and voltage scaling (DVFS) technique is used to adjust the CPU-cycle frequency of service clients' devices [1], [8], [12]. In the t th time slot, $f_{l,i}(t)$ is the CPU-cycle frequency of the i th service client's device.

$$D_{l,i}(t) = \tau f_{l,i}(t) L_i^{-1}, i \in U. \quad (3)$$

According to circuit theories [22], [23], the CPU power consumption of the i th service client's device in the t th time slot is calculated as

$$p_{l,i}(t) = k_{mod,i} f_{l,i}^3(t), i \in U, \quad (4)$$

where $k_{mod,i}$ is the effective switched capacitance of the CPU of the i th service client's device. $f_{max,i}$ is the maximum CPU-cycle frequency of the i th service client's device, i.e.,

$$0 \leq f_{l,i}(t) \leq f_{max,i}, i \in U. \quad (5)$$

In this paper, the operational frequency band of the j th edge server is divided equally into $|Z_j|$ which $B_j = \omega/|Z_j|$. In the t th time slot, let us denote the fading of wireless channels between the i th service client's device and the j th edge server as $\gamma_{i,j}(t)$, the channel power gain from the i th service client's device to the j th edge server as

$$\Gamma_{i,j}(t) = \gamma_{i,j}(t) g_0 \left(\frac{d_0}{d_{i,j}} \right)^\theta, i \in U, j \in S, \quad (6)$$

where g_0 is a constant, θ is an exponent, d_0 is the reference distance [8], and $d_{i,j}$ is the distance between the i th service client's device and the j th edge server. The task offloading variables in the t th time slot are defined as $\mathbf{X}(t) = \{x_{i,j}(t) | i \in U, j \in S\}$, where $x_{i,j}(t) \in \{0, 1\}$, where $x_{i,j}(t) = 1$ indicates that the i th service client's device offloads its computation tasks to the j th edge server in the t th time slot.

$$\sum_{j \in S} x_{i,j}(t) \leq 1, i \in U. \quad (7)$$

The transmit power of the i th service client's device in the t th time slot is denoted by $p_{r,i}(t)$. According to the Shannon-Hartley formula [24], the transmit rate between the i th service client's device and the j th edge server in the t th time slot is

$$r_{i,j}(t) = \begin{cases} B_j \log_2 \left(1 + \frac{\Gamma_{i,j}(t) p_{r,i}(t)}{B_j N_0} \right), & x_{i,j} = 1 \\ 0, & x_{i,j} = 0 \end{cases} \quad (8)$$

Without loss of generality, $p_{r,i}(t)$ must not exceed the maximum transmit power of the i th service client's device.

$$0 \leq p_{r,i}(t) \leq p_{max,i}, i \in U. \quad (9)$$

Based on the above definitions, the number of offloading tasks of the i th service client's device is calculated as

$$D_{r,i}(t) = \sum_{j \in S} r_{i,j}(t) \tau, i \in U. \quad (10)$$

C. Edge Server Scheduling

As shown in Fig. 2, the edge server can transfer input data to a neighboring edge server via an X2 link [25]. The amount of tasks of the i th service client's device that have been executed by the j th edge server in the t th time slot is denoted as $D_{s,i,j}(t)$. Then, $D_{s,i}(t)$ can be expressed as

$$D_{s,i}(t) = \sum_{j \in S} D_{s,i,j}(t), i \in U. \quad (11)$$

For the j th edge server, $f_{max,j}$ is the maximum CPU-cycle frequency and φ_j is the number of CPUs.

$$\sum_{j \in S} D_{s,i,j}(t) L_i \leq \tau \varphi_j f_{max,j}. \quad (12)$$

IV. PROBLEM FORMULATION

This section first defines the service cost function and then formulates the average service capacity, i.e., the long-term average number of service clients that can offload their tasks. Finally, the optimization problem studied in this paper is formulated.

The service cost function is defined as follows:

$$\xi_i(t) = \beta \xi_i^{latency}(t) + (1 - \beta) \xi_i^{power}(t), i \in U, \quad (13)$$

where $\beta \in [0, 1]$. $\xi_i^{latency}(t)$ and $\xi_i^{power}(t)$ are the latency cost and power cost of the i th service client's device in the t th time slot. According to Little's Law, the average execution delay of

a service client's device is proportional to the average amount of its tasks in the edge environment.

$$\xi_i^{\text{latency}}(t) = \alpha(Q_i(t) - D_{\sum,i}) + (1 - \alpha)(H_i(t) - D_{s,i}(t)). \quad (14)$$

where $\alpha \in [0, 1]$. Therefore, $\xi_i^{\text{power}}(t)$ is calculated as

$$\xi_i^{\text{power}}(t) = p_{l,i}(t) + p_{r,i}(t), i \in U. \quad (15)$$

Let us denote the long-term average number of service clients that can offload their tasks as \bar{O} .

$$\bar{O} = \lim_{T \rightarrow +\infty} \frac{1}{T \cdot m} \sum_{t=0}^{T-1} \sum_{i \in U, j \in S} x_{i,j}(t). \quad (16)$$

Constraint (17) requires that the service clients' task queues are stable [26].

$$\lim_{T \rightarrow +\infty} \frac{\mathbf{E}[|Q_i(T)|]}{T} = 0, \lim_{T \rightarrow +\infty} \frac{\mathbf{E}[|H_i(T)|]}{T} = 0. \quad (17)$$

Let us denote the system operation at the t th time slot with $\mathbf{R}(t) \triangleq \{\mathbf{f}(t), \mathbf{p}(t), \mathbf{X}(t), \mathbf{D}(t)\}$, in which $\mathbf{f}(t) \triangleq \{f_{i,i}(t) | i \in U\}$, $\mathbf{p}(t) \triangleq \{p_{r,i}(t) | i \in U\}$, $\mathbf{D}(t) \triangleq \{D_{s,j}(t) | j \in S\}$. Accordingly, the objective functions that minimize the average service cost and maximize the average service capacity respectively are defined below as \mathbf{P}_1 together:

$$\begin{aligned} \mathbf{P}_1 : & \min_{\{\mathbf{R}(t)\}} \lim_{T \rightarrow +\infty} \frac{1}{T} \sum_{t=1}^T \mathbf{E} \left[\sum_{i \in U} \xi_i(t) \right] \\ & \max_{\{\mathbf{R}(t)\}} \bar{O} \\ & \text{s.t. (5)(7)(9)(12)(17), } t \in T \end{aligned}$$

V. ALGORITHM DESIGN

This section introduces an online algorithm for solving the joint task offloading and resource allocation problem \mathbf{P}_1 . Based on Lyapunov optimization, the stochastic optimization problem is converted into a deterministic optimization problem.

A. Lyapunov Optimization-based Online Algorithm

We employ the Lyapunov optimization to ensure the stability of the task queues through minimizing the average service cost. To model the problem as a Lyapunov optimization problem, we define the Lyapunov function as:

$$L(\boldsymbol{\Theta}(t)) = \frac{1}{2} \sum_{i \in U} [Q_i^2(t) + H_i^2(t)]. \quad (18)$$

where $\boldsymbol{\Theta}(t) = [\mathbf{Q}(t), \mathbf{H}(t)]$. Then, the conditional Lyapunov drift is defined as

$$\Delta(\boldsymbol{\Theta}(t)) = \mathbf{E}[(L(\boldsymbol{\Theta}(t+1)) - L(\boldsymbol{\Theta}(t))) | \boldsymbol{\Theta}(t)]. \quad (19)$$

The Lyapunov drift-plus-penalty function is defined as:

$$\Delta_\nu(\boldsymbol{\Theta}(t)) = \Delta(\boldsymbol{\Theta}(t)) + V \cdot \mathbf{E}[\boldsymbol{\xi}(t) | \boldsymbol{\Theta}(t)], \quad (20)$$

where $\boldsymbol{\xi}(t) = \sum_{i \in U} \xi_i(t)$ and $V \in (0, +\infty)$ is a control parameter to keep the balance between task queues and service cost.

The upper bound of $\Delta(\boldsymbol{\Theta}(t))$ with constraints (5), (7), (9) and (12) is defined as:

$$\begin{aligned} \Delta_\nu(t) \leq & C - \mathbf{E} \left[\sum_{i \in U} Q_i(t) (D_{\sum,i}(t) - A_i(t)) | \boldsymbol{\Theta}(t) \right] \\ & - \mathbf{E} \left[\sum_{i \in U} H_i(t) (D_{s,i}(t) - D_{r,i}(t)) | \boldsymbol{\Theta}(t) \right] \\ & + V \cdot \mathbf{E}[\boldsymbol{\xi}(t) | \boldsymbol{\Theta}(t)], \end{aligned} \quad (21)$$

where C is a constant [8].

According to Lyapunov optimization, in order to minimize $\boldsymbol{\xi}(t)$ and maintain the stability of service clients' task queues, we need to minimize the upper bound of $\Delta_\nu(t)$ in each time slot as expressed in formula (21). This minimization is denoted as \mathbf{P}_{PTS} . All the constraints of \mathbf{P}_1 except constraint (17) are included in \mathbf{P}_{PTS} . Algorithm 1 presents the pseudo code of the Online Joint Task Offloading and Resource Allocation Algorithm (OJTORA). The pseudo-code for solving \mathbf{SP}_1 , \mathbf{SP}_2 and \mathbf{SP}_3 are presented below in Section V-B.

Algorithm 1 Online Joint Task Offloading and Resource Allocation (OJTORA)

Initialization: At the beginning of the t th time slot, the $\{\Gamma_{i,j}(t)\}$ and $\{A_i(t)\}$ will be given.

1: Get optimal $\mathbf{f}(t), \mathbf{p}(t), \mathbf{X}(t)$ and $\mathbf{D}(t)$ by solving

$$\begin{aligned} \mathbf{P}_{PTS} : & \min_{\mathbf{R}(t)} - \sum_{i \in U} Q_i(t) D_{\sum,i}(t) \\ & - \sum_{i \in U} H_i(t) (D_{s,i}(t) - D_{r,i}(t)) + V \cdot \boldsymbol{\xi}(t) \\ & \text{s.t. (5)(7)(9)(12)} \end{aligned}$$

in which the optimal $\mathbf{f}(t), \mathbf{p}(t), \mathbf{X}(t)$ and $\mathbf{D}(t)$ can be obtained by solving \mathbf{SP}_1 , \mathbf{SP}_2 and \mathbf{SP}_3 , respectively.

2: According to the iteration formula in (1) and (2), update the $\{Q_i(t)\}$ and $\{H_i(t)\}$.

3: Go to the next time slot, which $t = t + 1$.

B. Optimal Solution to \mathbf{P}_{PTS}

1) Optimal CPU-Cycle Frequencies of Mobile Devices:

The \mathbf{SP}_1 is defined as

$$\begin{aligned} \mathbf{SP}_1 : & \min_{\{f(t)\}} \sum_{i \in U} [- (Q_i(t) + V \cdot \alpha \beta) \tau L_i^{-1} f_{l,i}(t) \\ & + V \cdot (1 - \beta) k_{mod,i} f_{l,i}^3(t)] \\ & \text{s.t. (5)}. \end{aligned}$$

The objective function of \mathbf{SP}_1 is convex and constraint (5) is linear. Therefore, \mathbf{SP}_1 is a convex problem. For each service client, we can find that $f_{l,i}(t)$ is independent. Let us denote the optimal CPU-cycle frequency of the i th service client's device as $f_{l,i}^*(t)$. We can obtain $f_{l,i}^*(t)$ by finding the minimum point of $-(Q_i(t) + V \cdot \alpha \beta) \tau L_i^{-1} f_{l,i}(t) + V \cdot (1 - \beta) k_{mod,i} f_{l,i}^3(t)$. At the same time, $f_{l,i}^*(t)$ must fulfil constraint (5).

$$f_{l,i}^*(t) = \begin{cases} \min \left\{ \sqrt{\frac{(Q_i(t) + V \cdot \alpha \beta) \tau}{3 k_{mod,i} V (1 - \beta) L_i}}, f_{max,i} \right\}, & \text{others} \\ f_{max,i}, & \beta = 1 \end{cases} \quad (22)$$

2) *Optimal Transmit Power Consumption and Task Offloading Policy*: $p_{r,i}^*(t)$ and $\mathbf{X}^*(t)$ refer to the optimal $p_{r,i}(t)$ and the optimal $\mathbf{X}(t)$ respectively. They are obtained by solving \mathbf{SP}_2 :

$$\mathbf{SP}_2 : \min_{\{\mathbf{p}(t), \mathbf{X}(t)\}} \sum_{i \in U} [-\Psi_i(t)D_{r,i}(t) + V(1 - \beta)p_{r,i}(t)]$$

s.t.(7)(9).

When there is $\Psi_i(t) = Q_i(t) - H_i(t) + V \cdot \alpha\beta \leq 0$, the object function of \mathbf{SP}_2 is non-decreasing with $p_{r,i}(t)$. Then, because of constraint (9), $p_{r,i}^*(t) = 0$ means that the i th service client's device will not offload its tasks to edge servers. Let us denote $U_{off}(t) = \{i | \Psi_i(t) \geq 0, i \in U\}$. The optimal transmit power consumption and the optimal task offloading policy can be achieved by solving \mathbf{SP}_2 :

$$\mathbf{SP}_2' : \min_{\{\mathbf{p}(t), \mathbf{X}(t)\}} \sum_{i \in U_{off}(t)} [-\Psi_i(t)D_{r,i}(t) + V(1 - \beta)p_{r,i}(t)]$$

s.t.(7)(9).

This objective function is determined by two variables, i.e., $\mathbf{p}(t)$ and $\mathbf{X}(t)$. It is hard to obtain $p_{r,i}^*(t)$ and $\mathbf{X}^*(t)$ simultaneously. Therefore, we divide \mathbf{SP}_2' into two sub-problems.

First, we assume a $\mathbf{X}^*(t)$. For a fixed computation of offloading policy, we denote the first sub-problem of \mathbf{SP}_2' as \mathbf{SP}_{PWR} , which can be expressed as

$$\mathbf{SP}_{PWR} : \min_{\{\mathbf{p}(t)\}} \sum_{i \in U_{off}(t)} [-\Psi_i(t)D_{r,i}(t) + V(1 - \beta)p_{r,i}(t)]$$

s.t.(9)

where $D_{r,i}(t)$ is non-decreasing with $p_{r,i}(t)$. The objective function of \mathbf{SP}_{PWR} is convex and constraint (9) is linear. Therefore, \mathbf{SP}_{PWR} is a convex problem. Similar to \mathbf{SP}_1 , \mathbf{SP}_{PWR} can be decomposed for individual mobile devices. Let us denote the edge server to which the i th service client offloads its computation tasks in the t th time slot as j^* , there is $x_{i,j^*} = 1$. In addition, $r_{i,j^*}(t)$ can be obtained by

$$r_{i,j^*}(t) = B_{j^*} \log_2 \left(1 + \frac{\Gamma_{i,j^*}(t)p_{r,i}(t)}{B_{j^*}N_0} \right), \quad (23)$$

where $r_{i,j}(t) = 0$ if $j \neq j^*$. Then, $D_{r,i}(t) = r_{i,j^*}(t)\tau$. Therefore, we can obtain $p_{l,i}^*(t)$ by finding the minimum of $-\Psi_i(t)D_{r,i}(t) + V \cdot (1 - \beta)p_{r,i}(t)$. Let us denote $\Lambda_i(t) = \frac{\Psi_i(t)\tau B_{j^*}}{V \cdot (1 - \beta) \ln 2} - \frac{N_0 B_{j^*}}{\Gamma_{i,j^*}(t)}$. Then, the $p_{l,i}^*(t)$ can be written as

$$p_{l,i}^*(t) = \begin{cases} \min\{\Lambda_i(t), p_{max,i}\}, & \text{others} \\ p_{max,i}, & \beta = 1 \end{cases} \quad (24)$$

Secondly, we assume a fixed $p^*(t)$ and denote the second sub-problem of \mathbf{SP}_2' as \mathbf{SP}_{CO} , expressed as follows:

$$\mathbf{SP}_{CO} : \min_{\{\mathbf{X}(t)\}} \sum_{i \in U_{off}(t)} [-\Psi_i(t)D_{r,i}(t)]$$

s.t.(7).

For a service client where $i \in U_{off}(t)$, it independently selects an edge server to offload its computation tasks. Therefore, \mathbf{SP}_{CO} can be decomposed for individual service clients.

In order to solve \mathbf{SP}_{CO} , we propose an algorithm to obtain $\mathbf{X}^*(t)$, as described in Algorithm 2.

Algorithm 2 Offloading Server Selection

Initialization: $\varepsilon_i(t) = \emptyset, \min_i = MIN_DOUBLE$.

```

1: for all  $i \in U$  do
2:   for each  $j \in G_i(t) \& j \notin \varepsilon_i(t)$  do
3:      $\min = \Psi_i(t)D_{r,i}(t);$ 
4:     if  $\min_i \leq \min$  then
5:        $j^* = j;$ 
6:     end if
7:      $G_i(t) = G_i(t) \setminus \{j\}, \varepsilon_i(t) = \varepsilon_i(t) \cup \{j\};$ 
8:   end for
9:    $x_{i,j^*} = 1;$ 
10: end for
```

3) *Computing Resource Allocation of Edge Servers*: All the parts remaining in \mathbf{P}_{PTS} are only related to the allocation of edge servers' computing resources, defined as \mathbf{SP}_3 :

$$\mathbf{SP}_3 : \min_{\{\mathbf{D}(t)\}} \sum_{i \in U} [-[V(1 - \alpha)\beta + H_i(t)] \cdot D_{s,i}(t)]$$

s.t.(12).

The value achieved by the objective function of \mathbf{SP}_3 decreases with $D_{s,i}(t)$. Let us denote $value_i = V(1 - \alpha)\beta + H_i(t)$. It can be found that the greater the $value_i$, the more benefit will be generated when an edge server executes an equivalent amount of tasks. Then, OJTORA employs Algorithm 3 to solve \mathbf{SP}_3 .

Algorithm 3 Computing Resource Allocation

Initialization: According to the value of $value_i$, sort mobile devices in a decreasing order. $i = 1, rest_j = \tau\varphi_i f_{max,j}$.

```

1: while  $i \leq n$  do
2:   if  $\sum_{j \in G_i(t)} rest_j \geq H_i(t)$  then
3:      $D_{s,i}(t) = H_i(t);$ 
4:      $rest_j = rest_j - \frac{rest_j}{\sum_{j \in G_i(t)} rest_j} D_{s,i}(t);$ 
5:   else
6:      $D_{s,i}(t) = \sum_{j \in G_i(t)} rest_j;$ 
7:      $rest_j = 0;$ 
8:   end if
9:    $i = i + 1;$ 
10: end while
```

VI. EXPERIMENTAL EVALUATION

This section evaluates the performance of OJTORA against two baseline approaches. All the experiments were conducted on a machine equipped with Intel Core i5-7400T processor (4 CPUs, 2.4GHz) and 8GB RAM, running Windows 10 x64.

A. Baseline Approaches

To our best knowledge, OJTORA is the first attempt to consider both the resource allocation and task offloading

in a multi-server edge computing environment for latency-tolerant applications. Due to the issue of edge server coverage overlapping, existing approaches designed for the single-server edge computing environment cannot be directly applied to the multi-server environment. Thus, in the experiments, OJTORA is evaluated against two intuitive baseline approaches, namely Random and Greedy:

- **Random:** The Random approach select edge servers randomly to offload service clients' tasks.
- **Greedy:** In (6), a shorter distance between an edge server and a service client's device results in a smaller communication interference. Based on this fact, the Greedy approach selects the closest edge servers to offload service clients' tasks.

B. Experiment Settings

The experiments are set up in a way similar to [8]. In each experiment, a total of n service clients are distributed in an area covered by m base stations. The radius of each base station is 150 m, and there is $\gamma_{i,j}(t) \sim \text{Exp}(1), i \in U, j \in S$. The simulation results are the average values over 10,000 time slots. In general, there is $n=30, m=3$. The $A_{i,max} = 1000$ bits. Table I presents the parameter settings used in the experiments.

TABLE I
SYSTEM PARAMETERS

Parameter	Value	Parameter	Value
ω	10 Hz	N_0	-174 dBm/Hz
g_0	-40 dB	d_0	1 m
θ	4	$k_{mod,i}$	1×10^{-27}
$f_{max,i}$	1 GHz	$p_{max,i}$	500 mW
L_i	737.5 cycles/bit	$f_{max,j}$	2.5 GHz
φ_j	4	τ	2 ms

In the experiments, we vary three parameters that may have an impact on OJTORA:

- **Control Parameter V :** We change the value of V , where $V = 1 \times 10^9, 2 \times 10^9, \dots, 9 \times 10^9$, under four circumstances: 1) $\alpha = 0.3, \beta = 1 \times 10^{-5}$; 2) $\alpha = 0.3, \beta = 1 \times 10^{-6}$; 3) $\alpha = 0.7, \beta = 1 \times 10^{-5}$; and 4) $\alpha = 0.7, \beta = 1 \times 10^{-6}$.
- **Number of Users n :** We change the number of service clients, where $n = 10, 20, 30, 100, 200$, with $\alpha = 0.3, \beta = 1 \times 10^{-5}, V = 1 \times 10^9$.
- **Number of Edge Servers m :** We change the number of edge servers, where $m = 3, 6, 9$, with $\alpha = 0.3, \beta = 1 \times 10^{-5}, V = 1 \times 10^9$.

Two performance metrics are employed to evaluate OJTORA, corresponding to the two optimization objectives.

- **Service Capacity.** Service capacity is measured by the long-term average number of service clients served, as formally defined in (16).
- **Service Cost.** Service cost is defined as $\bar{\xi}_{\Sigma} = \frac{1}{T \cdot n} \sum_{t=0}^{T-1} \sum_{i \in U} \xi_i(t)$. In order to compare the power

consumption and service latency more clearly, two more specific metrics are defined and employed in the evaluation: 1) the average power consumption of service clients' devices, defined as $\bar{p}_{\Sigma} = \frac{1}{T \cdot n} \sum_{t=0}^{T-1} \sum_{i \in U} (p_{l,i}(t) + p_{r,i}(t))$; and 2) the average queue length of service clients' devices: $\bar{q}_{\Sigma} = \frac{1}{T \cdot n} \sum_{t=0}^{T-1} \sum_{i \in U} (Q_i(t) + H_i(t))$.

C. Experimental Results

1) **Optimality:** As shown in Fig. 3, OJTORA outperforms the two baselines approaches significantly in achieving both optimization objectives. Specifically, it outperforms the Random approach by 27.8%-41.1% in power consumption, 23.2%-37.1% in queue length, 0.2%-10.9% in average service capacity and 23.1%-37.2% in average service cost. OJTORA also outperforms the Greedy approach, by 25.6%-39.1% in power consumption, 20.1%-35.7% in queue length, 0.2%-6.7% in average service capacity and 22.1%-35.8% in average service cost. Fig. 3 also shows that the Greedy approach outperforms the Random approach. The reason is that the Greedy approach selects the closest edge servers to offload service clients' tasks. This reduces the power consumed by the data transmission between service clients' devices and edge servers.

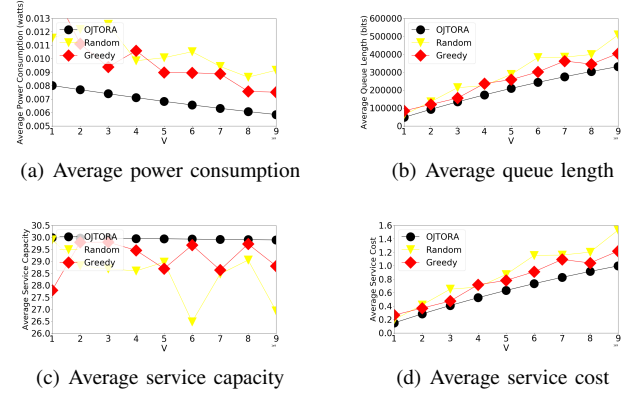


Fig. 3. Average power consumption, average queue length and average service cost, average service capacity ($\alpha = 0.3, \beta = 1 \times 10^{-5}$)

2) **Impact of Parameter V :** As shown in Fig. 4, as V increases, the average service capacities achieved by all three approaches decrease. However, the average service costs increase. According to (22) and (24), as V increases, the transmit power consumption of service clients' devices decreases. This results in the decrease in the average service capacity. With the increase in V , the power consumption increases more rapidly than the latency. As a result, the average service cost increases.

3) **Impact of the Number of Service Clients n :** As shown in Fig. 5, As n increases, the average service cost increases. This is because the competition among service clients for the computing resources on the edge servers becomes fiercer with the increase in n .

4) **The Impact of the Number of Edge Servers m :** As shown in Fig. 6, with the increase in m , the average power consumption, the average queue length and the average service

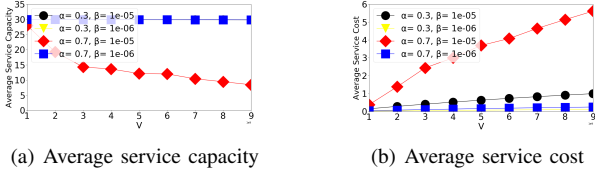


Fig. 4. Average service capacity and average service cost vs V .

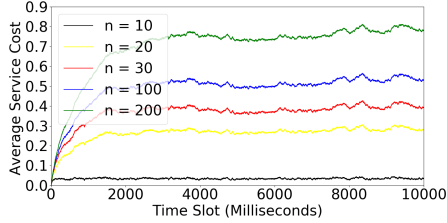


Fig. 5. Average service cost per time slot vs the number of mobile devices.

cost increase. However, when m continues to increase, the average service cost converges and remains steady. This is because the resources are more than sufficient to allow all service clients to be served.

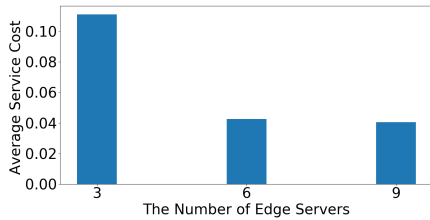


Fig. 6. Average service cost vs the number of edge servers.

VII. CONCLUSION

In this paper, we investigated a joint task offloading and resource allocation problem in multi-server edge computing environments with the objectives to maximize service capacity, i.e., the number of mobile devices served, and to minimize the service cost, i.e., the service latency and power consumption experienced by service clients. To solve this problem, we proposed OJTORA, an online algorithm based on Lyapunov optimization, which converts the stochastic optimization problem to a per-time-slot deterministic optimization problem. The experimental results show that our approach significantly outperforms two baseline approaches. However, OJTORA does not consider the fairness in the resource sharing among service clients because it assumes static bandwidth allocation for now. In the future, dynamical allocation of bandwidth will be studied. We will also extend OJTORA to accommodate users service clients' mobility.

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, 2017.

[2] Y. Zhou and Z. Zhi, "Near-End Cloud Computing: Opportunities and Challenges in the Post-Cloud Computing," *Chinese Journal of Computers*, vol. 41, no. 25, pp. 10–19, 2018.

[3] X. Ge, S. Tu, G. Mao, C.-X. Wang, and T. Han, "5G Ultra-Dense Cellular Networks," *IEEE Wireless Communications*, vol. 23, no. 1, pp. 72–79, 2016.

[4] Y. Qi, Y. Zhou, L. Liu, L. Tian, and J. Shi, "MEC Coordinated Future 5G Mobile Wireless Networks," *Journal of Computer Research and Development*, vol. 55, no. 3, pp. 478–485, 2018.

[5] P. Lai, Q. He, M. Abdelrazek, F. Chen, J. Hosking, J. Grundy, and Y. Yang, "Optimal Edge User Allocation in Edge Computing with Variable Sized Vector Bin Packing," in *International Conference on Service-Oriented Computing (ICSOC)*. Springer, 2018, pp. 230–245.

[6] L. Zhang, S. Wang, and R. N. Chang, "QCSS: A QoE-aware Control Plane for Adaptive Streaming Service over Mobile Edge Computing Infrastructures," in *IEEE International Conference on Web Services (ICWS)*. IEEE, 2018, pp. 139–146.

[7] H. Wu, S. Deng, W. Li, M. Fu, J. Yin, and A. Y. Zomaya, "Service Selection for Composition in Mobile Edge Computing Systems," in *IEEE International Conference on Web Services (ICWS)*. IEEE, 2018, pp. 355–358.

[8] Y. Mao, J. Zhang, S. Song, and K. B. Letaief, "Stochastic Joint Radio and Computational Resource Management for Multi-User Mobile-Edge Computing Systems," *IEEE Transactions on Wireless Communications*, vol. 16, no. 9, pp. 5994–6009, 2017.

[9] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint Optimization of Radio and Computational Resources for Multicell Mobile-Edge Computing," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 1, no. 2, pp. 89–103, 2015.

[10] X. Lyu, H. Tian, C. Sengul, and P. Zhang, "Multi-User Joint Task Offloading and Resource Optimization in Proximate Clouds," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 4, pp. 3435–3447, 2017.

[11] B. Yu, L. Pu, Y. Xie, J. Xu, and J. Zhang, "Joint Task Offloading and Base Station Association in Mobile Edge Computing," *Journal of Computer Research and Development*, vol. 55, no. 3, pp. 537–544, 2018.

[12] T. X. Tran and D. Pompili, "Joint Task Offloading and Resource Allocation for Multi-Server Mobile-Edge Computing Networks," *arXiv preprint arXiv:1705.00704*, 2017.

[13] C.-F. Liu, M. Bennis, and H. V. Poor, "Latency and Reliability-Aware Task Offloading and Resource Allocation for Mobile Edge Computing," in *IEEE Global Communications Conference (GlobeCom Workshops)*. IEEE, 2017, pp. 1–7.

[14] M. Masoudi, B. Khamidehi, and C. Cavdar, "Green Cloud Computing for Multi Cell Networks," in *Wireless Communications and Networking Conference (WCNC)*. IEEE, 2017, pp. 1–6.

[15] L. Yang, J. Cao, Z. Wang, and W. Wu, "Network Aware Multi-User Computation Partitioning in Mobile Edge Clouds," in *International Conference on Parallel Processing (ICPP)*. IEEE, 2017, pp. 302–311.

[16] T. He, H. Khamfroush, S. Wang, T. La Porta, and S. Stein, "Its Hard to Share: Joint Service Placement and Request Scheduling in Edge Clouds with Sharable and Non-Sharable Resources," Technical Report, December 2017.[Online]. Available: <https://1drv.ms/b/s?Tech.Rep.>, 2018.

[17] T. Li, C. S. Magurawalage, K. Wang, K. Xu, K. Yang, and H. Wang, "On Efficient Offloading Control in Cloud Radio Access Network with Mobile Edge Computing," in *International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2017, pp. 2258–2263.

[18] L. Wang, L. Jiao, J. Li, and M. Mühlhäuser, "Online Resource Allocation for Arbitrary User Mobility in Distributed Edge Clouds," in *Distributed Computing Systems (ICDCS)*. IEEE, 2017, pp. 1281–1290.

[19] A. Alfayly, I.-H. Mkwawa, L. Sun, and E. Ifeachor, "Qoe-Driven LTE Downlink Scheduling for Voip Application," in *Consumer Communications and Networking Conference (CCNC)*. IEEE, 2015, pp. 603–604.

[20] A. P. Miettinen and J. K. Nurminen, "Energy Efficiency of Mobile Clients in Cloud Computing," *HotCloud*, vol. 10, pp. 4–4, 2010.

[21] S. Han, X. Wang, L. Xu, H. Sun, and N. Zheng, "Frontal Object Perception for Intelligent Vehicles Based on Radar and Camera Fusion," in *Control Conference (CCC)*. IEEE, 2016, pp. 4003–4008.

[22] X. Chen, "Decentralized Computation Offloading Game for Mobile Cloud Computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 4, pp. 974–983, 2015.

- [23] Y. Wen, W. Zhang, and H. Luo, "Energy-Optimal Mobile Application Execution: Taming Resource-Poor Mobile Devices with Cloud Clones," in *IEEE International Conference on Computer Communications (INFOCOM)*. IEEE, 2012, pp. 2716–2720.
- [24] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. John Wiley & Sons, 2012.
- [25] A. Ndikumana, N. H. Tran, T. M. Ho, Z. Han, W. Saad, D. Niyato, and C. S. Hong, "Joint Communication, Computation, Caching, and Control in Big Data Multi-Access Edge Computing," *arXiv preprint arXiv:1803.11512*, 2018.
- [26] M. J. Neely, "Stochastic Network Optimization with Application to Communication and Queueing Systems," *Synthesis Lectures on Communication Networks*, vol. 3, no. 1, pp. 1–211, 2010.