This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2020.2975897, IEEE Transactions on Industrial Informatics

Exploring Placement of Heterogeneous Edge Servers for Response Time Minimization in Mobile Edge-Cloud Computing

Kun Cao, Liying Li, Yangguang Cui, Tongquan Wei, and Shiyan Hu

***Abstract*—In the past few years, the study on placing edge servers for response time optimization in mobile edge-cloud computing systems has become increasingly popular. Most existing schemes neglect two important aspects: one is the heterogeneity of edge/cloud servers and the other is the response time fairness of base stations, which may significantly degrade the system quality of services to mobile users. In this paper, we conduct the study of deploying heterogeneous edge servers to optimize the expected response time of both the whole and individual base stations. We propose an approach consisting of offline and online stages. At offline stage, the optimal placement strategy of heterogeneous edge servers is produced by using integer linear programming (ILP) technique. At online stage, a mobility-aware game theory based method is developed to deal with the dynamic characteristic of user movement. Experimental results reveal that compared to benchmarking methods, our approach not only reduces system expected response time by 47.37%, but also improves response time fairness of base stations by 71.60%.**

***Index Terms*—Edge server placement, fairness, mobile edge-cloud computing, response time minimization.**

I. Introduction

With the popularity of emerging mobile applications such as augmented/virtual reality, the computation requirements of mobile users have been constantly increasing nowadays. In the conventional architecture of mobile cloud computing, mobile users can offload their requests via base stations to remote cloud servers, and ask these cloud servers to perform task executions on behalf of themselves [1], [2]. However, this paradigm has become infeasible in the era of Internet of things because it inevitably exposes mobile users to high service delays. As an alternative solution, mobile edge-cloud computing [3], [4] that deploys edge servers between mobile users and cloud servers is hence proposed to provide low-latency services for numerous mobile users.

In the past few years, there has been a widespread interest in the study of edge/cloud server deployment for mobile edge-cloud computing systems. The locations of the edge/cloud servers are an important factor in optimizing server energy consumption, deployment costs, and mobile user latency. From the perspective of energy optimization, Li et al. [5] aimed to minimize the whole energy consumption of edge servers by adopting the well-known technique of particle swarm optimization to place homogeneous edge servers. A Benders decomposition based approach was developed by Yang et al. [6] to place cloudlets (i.e., edge servers) for minimizing system energy dissipation under latency constraints. As for deployment costs, Yao et al. [7] investigated how to place heterogeneous edge servers in a heterogeneity and cost-aware way while meeting latency requirements. Based on enumeration and partitioning algorithms, Zhao et al. [8] solved the edge server placement problem for obtaining minimum data traffic in mobile edge computing systems. Recently, some works [9], [10] have been devoted to reducing mobile user latency. Yang et al. [9] presented a novel resource allocation scheme based on predicted distribution of user requests. Chen et al. [10] developed a cloudlet placement framework to optimize both the deployment cost of homogeneous cloudlets and the average response time of mobile users. Nevertheless, all the works [5]–[10] fail to account for the mobility of users, i.e., dynamic characteristic of user movement at runtime. In mobile edge-cloud computing environments, mobile users frequently move from one area to another area within the system, which leads to the fluctuation in workloads of base stations over different scheduling horizons [11], [12].

To alleviate the workload imbalance due to frequent movements of mobile users, recent works [13]–[16] have focused on the design of resource and latency-aware strategies by adjusting base station-to-server computation offloading mappings. For homogeneous edge-cloud computing systems, Xu et al. [13] proposed an auction-based scheduling strategy to allocate available resources of servers for better service quality. Meng et al. [14] formulated the online mapping of base stations to edge servers into the well investigated minimum-cost maximum flow problem. Zhao et al. [15] proposed a ranking based heuristic to minimize the average response time of mobile users at runtime. All the schemes shown in [13]–[15] are dedicated to homogeneous edge/cloud servers; however, edge/cloud servers are usually heterogeneous in real-world system deployment [17]. Given this situation, Rashidi et al. [16] proposed an online task assignment strategy to jointly reduce system response time and task dropping rates of heterogeneous edge servers. However, they assumed that heterogeneous servers have already been properly deployed and did not explore how to deploy these servers in a latency-aware way.

To our best knowledge, most existing design strategies for minimizing system response time do not simultaneously

K. Cao, L. Li, Y. Cui, and T. Wei are with the School of Computer Science and Technology, East China Normal University, Shanghai 200062, China. S. Hu is with the School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, UK. T. Wei is the corresponding author: tqwei@cs.ecnu.edu.cn. This work was partially supported by National Key Research and Development Program of China under Grant 2018YFB2101300 and ECNU xinfuzhihua program.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2020.2975897, IEEE Transactions on Industrial Informatics

consider the heterogeneity of edge/cloud servers and the response time fairness of base stations. On one hand, existing designs concentrate on system response time optimization with no consideration of the heterogeneity of edge/cloud servers, resulting in low server utilizations. For instance, Wang et al. [17] utilized the mixed-integer linear programming (MILP) technique to optimize response time fairness without considering the heterogeneity of servers. On the other hand, these designs are merely intended to minimize system response time, and ignore the optimization of response time fairness of base stations. For instance, Samanta et al. [18] described a service latency-aware decision engine to effectively provide computation offloading services without considering response time fairness of base stations. From the perspective of a mobile edge-cloud computing system, it desires to optimize the average response time of base stations for high system quality of services. However, from the perspective of each base station, it hopes to minimize its own response time as much as possible in order to improve the quality of services of mobile users it serves. Therefore, the cooperation and competition of base stations should be considered during edge server deployment in order to minimize the response time of the system while maintaining the fairness in response time of base stations.

In this paper, we study the optimization of both the expected response time of the entire system and the fairness in expected response time of individual base stations through edge server placement and base station-to-server computation offloading mappings. In addition to an offline method, we also provide an online technique to address the workload fluctuation of base stations incurred due to dynamic characteristic of user movements. The major contributions of this paper are summarized as follows.

- We investigate the problem of deploying heterogeneous edge servers for response time minimization in mobile edge-cloud computing systems. Particularly, we jointly optimize the expected response time of the system and the fairness in expected response time of base stations.
- We propose an effective approach composed of offline and online stages. At offline stage, integer linear programming (ILP) technique is leveraged to produce an optimal solution of edge server placement. At online stage, a game theory based scheme of base station remapping is developed to deal with the mobility of users.
- We conduct extensive experiments to evaluate the developed approach. Experimental results show that compared to benchmarking algorithms, our approach improves system expected response time and response time fairness by up to 47.37% and 71.60%, respectively.

The remainder of this paper is organized as follows. Section II introduces system architecture and model. Section III presents the problem definition of response time minimization and shows an overview of our proposed two-stage approach. Section IV describes the ILP based offline scheme while Section V details the game theory based online scheme. We experimentally investigate the performance of our solution in Section VI and give concluding remarks in Section VII.

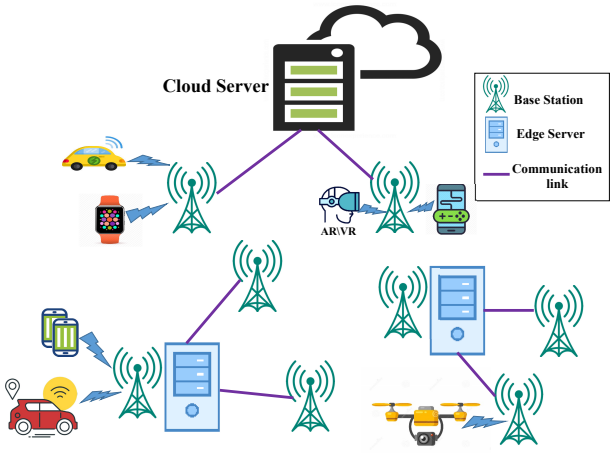


Fig. 1: Illustration of a mobile edge-cloud computing system.

II. System Architecture and Model

In this section, we first introduce the system architecture and then describe the response time model.

A. System Architecture

We consider a representative mobile edge-cloud computing system [3] consisting of multiple mobile users, M base stations $\mathcal{B} = \{B_1, B_2, \cdots, B_M\}$, N edge servers $\mathcal{S_E} = \{S_1, S_2, \cdots, S_N\}$ $(M \geq N)$, and one cloud server S_0. Generally, the number of mobile users is much larger than the number of base stations and edge servers. Let $\mathcal{S} = \{S_0, \mathcal{S_E}\}$ be the server set in which the server indexed by integer number n $(0 \leq n \leq N)$ is referred to as edge/cloud server S_n. The edge/cloud servers are heterogenous in terms of processing capacity. A mobile user sends requests (i.e., tasks) to a base station within his/her proximity, and then the tasks are either forwarded to an edge server for local execution or offloaded to cloud server S_0 for remote execution.

The topology of base stations and edge/cloud servers is represented as an undirected graph $G = (V, E)$. $V = \{\mathcal{B}, \mathcal{S}\}$ denotes the set of base stations and edge/cloud servers. We assume that the placement locations of cloud server S_0 and M base stations have already been determined. On the contrary, the placement locations of edge servers are unknown, and they are one of the optimization variables in this paper. Each edge server should be co-located with one base station and thus it has possible M placement locations in total. E is the set of candidate communication links between base stations and edge/cloud servers for all placement strategies of edge servers. In other words, the computation offloading mapping of base stations to edge/cloud servers is also the optimization variable in this paper. Note that the topology is full-meshed in logical, but it is actually a multihop network. That is, a base station can connect to an edge/cloud server through other base station(s), and the forwarding time in the relay base station(s) is ignored. As an intuitive example, Fig 1 illustrates the target edge-cloud computing system with one feasible solution of placing heterogenous edge servers and mapping base stations to edge/cloud servers.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2020.2975897, IEEE Transactions on Industrial Informatics

B. Response Time Model

1) Communication Delay: For any base station, it needs to transmit mobile users' tasks to an edge/cloud server and ask the edge/cloud server to execute these tasks. The tasks that each base station B_m receives from multiple mobile users are assumed to follow a Poisson process with average arrival rate λ_m (CPU cycles per second). Suppose that base station B_m connects to edge/cloud server S_n through a link with communication capacity $\vartheta_{m,n}$. Note that the connection between base stations and edge/cloud servers are not fully meshed, that is, nodes communicate via multi-hop connection. We assume the bandwidth between connected base stations and edge/cloud servers is equally shared by links. As a result, the communication capacity of individual link is proportionally updated according to the total number of shared links. Let a non-negative variable $d(B_m, S_n)$ denote the Euclidean distance between base station B_m and edge/cloud server S_n. Following the model introduced in [19], the average communication latency of tasks transmitted from base station B_m to edge/cloud server S_n is given by

$$W_{delay}^{(1),m,n} = \frac{d(B_m, S_n)}{\varrho} + \frac{D_m}{\vartheta_{m,n}}. \tag{1}$$

The constant number ϱ and the non-negative variable D_m are the propagation rate of electromagnetic waves and the average task data volume of base station B_m, respectively.

2) Task Execution Delay: We adopt the popular M/G/1 queue model [20] to describe task processing procedure of edge/cloud server S_n. External tasks arrive, wait in a task queue, and get executed with speed p_n (CPU cycles per second). In the M/G/1 queue model, the time distribution of task executions on edge/cloud server S_n can follow any general distribution with mean $\mu_n > 0$ and variance $\delta_n^2 > 0$. Considering edge/cloud server S_n connected to multiple base stations, we use $\mathcal{B}_n$ to denote the set of base stations mapped to edge/cloud server S_n. According to [20], the average task execution delay (including waiting time at task queue) of base station B_m on edge/cloud server S_n is expressed as

$$W_{delay}^{(2),m,n} = \frac{\lambda_m}{p_n} + \frac{(\mu_n^2 + \delta_n^2)(\lambda_m + \Delta_n)}{2(p_n - \lambda_m + \Delta_n)}, \tag{2}$$

where $\Delta_n = \sum_{B_k \in \{\mathcal{B}_n - \{B_m\}\}} \lambda_k$.

3) Total Response Delay: Evidently, the expected response time of a base station is composed of communication delay and task execution delay. Combining (1) and (2), the expected response time of base station B_m is thus depicted as

$$W_{delay}^{tot,m} = \sum_{j=1}^{M}\sum_{n=0}^{N}\Big(\mathcal{P}_{j,n,m}\big(\frac{d(B_m, S_n)}{\varrho} + \frac{D_m}{\vartheta_{m,n}}\big) + \mathcal{P}_{j,n,m}\big(\frac{\lambda_m}{p_n} + \frac{(\mu_n^2 + \delta_n^2)(\lambda_m + \Delta_n)}{2(p_n - \lambda_m + \Delta_n)}\big)\Big). \tag{3}$$

Here, we define a binary decision variable $\mathcal{P}_{j,n,m}$ and it is set to 1 when i) S_n is an edge server co-located with base station $B_j \in \mathcal{B}$ and base station B_m is mapped to the S_n, or ii) S_n is a cloud server and base station B_m is mapped to the S_n for fixed $j = 1$. Otherwise, $\mathcal{P}_{j,n,m}$ is set to 0. The expected response time of the system is then given by the average response time of M base stations, that is,

$$W_{delay}^{avg} = \frac{1}{M}\sum_{m=1}^{M} W_{delay}^{tot,m}. \tag{4}$$

III. Problem Definition and Solution Overview

In this section, we first define the studied optimization problem and then present an overview of our solution.

A. Problem Definition

As aforementioned, from a system point of view, mobile edge-cloud computing systems aim to optimize the average response time of base stations. Nevertheless, from an individual point of view, each base station wishes to reduce its own response time as much as possible, which necessitates the fairness in response time of multiple base stations. Formally, the studied problem in this paper is defined as follows. Given an undirected graph $G = (V, E)$, determine the placement locations of edge servers and mapping strategy of base stations to edge/cloud servers in order to minimize the system expected response time and maximize the fairness in expected response time of base stations. Three constraints should be satisfied. First, each edge server is exactly co-located with one base station. Second, each base station can only be mapped to one edge/cloud server for task execution. Third, the task arrival rate at any edge/cloud server cannot exceed the maximal rate at which tasks can be processed by the edge/cloud server.

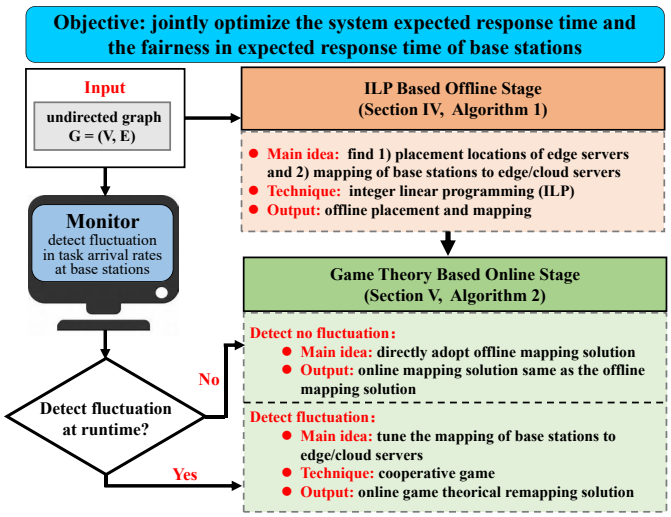


Fig. 2: Overview of our proposed two-stage solution.

B. Solution Overview

We propose an approach composed of offline and online stages to solve the above problem, as shown in Fig. 2. At offline stage, the ILP technique is utilized to derive an optimal scheme of deploying edge servers and mapping base stations. Due to the mobility of mobile users, the task arriving rates at base stations may fluctuate at runtime. Therefore, it is necessary to adjust the offline mapping of base stations to edge/cloud servers at runtime. Given this, we develop a low-cost yet high-performance online scheme. To be specific, when

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2020.2975897, IEEE Transactions on Industrial Informatics

there is no fluctuation in task arriving rates at all base stations at runtime, the offline mapping solution is directly utilized; otherwise, a game theoretic remapping heuristic is triggered. The basic idea of the heuristic is to tune the mapping of base stations with changed task arriving rates to edge/cloud servers. Through the efforts made in two stages, our approach is able to optimize both the expected response time of the entire system and the fairness in expected response time of base stations.

IV. ILP Based Offline Stage

In this section, we present an ILP based scheme to determine the optimal placement locations of edge servers and computation offloading mapping of base stations to edge/cloud servers at design time.

A. Formulate ILP

The target of this paper is to minimize the system expected response time and maximize the response time fairness of base stations. The system expected response time is given by (4). Since the metric of mean deviation [21] can characterize performance fairness between individuals, we adopt the mean deviation of expected response time $\{W_{delay}^{tot,1}, W_{delay}^{tot,2}, \cdots, W_{delay}^{tot,M}\}$ to evaluate the response time fairness of multiple base stations. Specifically, the mean deviation is calculated as the average value of absolute response time difference between each base station and the whole system, that is,

$$W_{delay}^{fair} = \frac{1}{M}\sum\nolimits_{m=1}^{M}(S_{max}^{m} - S_{min}^{m}), \tag{5}$$

where S_{max}^{m} indicates the maximum of $W_{delay}^{tot,m}$ and W_{delay}^{avg}, and S_{min}^{m} suggests the minimum of $W_{delay}^{tot,m}$ and W_{delay}^{avg}. It is clear the smaller the W_{delay}^{fair}, the more fair the expected response time achieved by these base stations. In summary, we define the ILP objective function as minimizing the sum of W_{delay}^{avg} and W_{delay}^{fair}, that is,

$$\min\ \left(W_{delay}^{avg}(\mathcal{P}_{j,n,m}) + W_{delay}^{fair}(S_{min}^{m})\right). \tag{6}$$

Note that we select the variable S_{min}^{m} rather than both S_{min}^{m} and S_{max}^{m} as optimization variable in fairness function W_{delay}^{avg}. This is because the variable S_{max}^{m} can be represented by S_{min}^{m}, as shown in (12). The linear constraints of variable S_{min}^{m} are summarized and listed as follows.

$$S_{min}^{m} \leq W_{delay}^{tot,m}(\mathcal{P}_{j,n,m}) \tag{7}$$

$$S_{min}^{m} \leq W_{delay}^{avg}(\mathcal{P}_{j,n,m}) \tag{8}$$

$$S_{min}^{m} \geq W_{delay}^{tot,m}(\mathcal{P}_{j,n,m}) - \mathcal{Z} \times (1 - \Upsilon_{avg}^{m}) \tag{9}$$

$$S_{min}^{m} \geq W_{delay}^{avg}(\mathcal{P}_{j,n,m}) - \mathcal{Z} \times \Upsilon_{avg}^{m} \tag{10}$$

$$\Upsilon_{avg}^{m} = 0, 1 \tag{11}$$

$$S_{max}^{m} = W_{delay}^{tot,m}(\mathcal{P}_{j,n,m}) + W_{delay}^{avg}(\mathcal{P}_{j,n,m}) - S_{min}^{m} \tag{12}$$

Here, Υ_{avg}^{m} is an auxiliary binary decision variable and it is introduced to imply the relationship of S_{min}^{m}, $W_{delay}^{tot,m}$, and W_{delay}^{avg}. If condition $W_{delay}^{tot,m} < W_{delay}^{avg}$ holds, then $\Upsilon_{avg}^{m} = 1$, else $\Upsilon_{avg}^{m} = 0$. $\mathcal{Z}$ is a large constant number and is set to 10000 in our experiments. To ensure the schedule feasibility of every task in the concerned system, the following linear constraints of variable $\mathcal{P}_{j,n,m}$ must be satisfied.

- *No matter where edge servers are placed, every base station can only be mapped to one edge/cloud server for task execution. That is,*

$$\sum\nolimits_{j=1}^{M}\sum\nolimits_{n=0}^{N}\mathcal{P}_{j,n,m} = 1, \quad \forall B_m \in \mathcal{B}. \tag{13}$$

- *No matter how base stations are mapped, every edge server is exactly co-located with one base station. Let $\mathcal{B}_{S_n \in \mathcal{S}_\mathcal{E}}$ be the set of base stations mapped to edge server $S_n \in \mathcal{S}_\mathcal{E}$, we then have*

$$\sum\nolimits_{j=1}^{M}\mathcal{P}_{j,n,m} = 1, \quad \forall S_n \in \mathcal{S}_\mathcal{E}, \forall B_m \in \mathcal{B}_{S_n \in \mathcal{S}_\mathcal{E}}. \tag{14}$$

- *The task arrival rate at every edge/cloud server cannot exceed the rate at which tasks can be processed by the edge/cloud server. Let λ_m^{off} be the offline task arrival rate at base station B_m, we then have*

$$\sum\nolimits_{j=1}^{M}\sum\nolimits_{m=1}^{M}\mathcal{P}_{j,n,m}\lambda_m^{off} \leq p_n, \quad \forall S_n \in \mathcal{S}. \tag{15}$$

B. ILP Based Offline Algorithm

Algorithm 1: PEMB: Place Edge Servers and Map Base Stations to Edge/Cloud Servers

Input: undirected graph $G = (V, E)$
1 utilize the ILP solver in [22] to tackle the formulated ILP: objective (6), subject to (7)-(15);
2 **return** offline solution $\mathcal{P}^*$.

Algorithm 1 shows our proposed ILP based offline solution. The algorithm takes the undirected graph $G = (V, E)$ as input. An existing ILP solver in [22] is leveraged to deal with the formulated ILP in Section IV-A. By addressing the ILP, the optimal solution $\mathcal{P}^*$ to edge server placement and base station mapping is hence derived.

V. Game Theory Based Online Stage

Due to the fluctuating task arriving rates at base stations, it is desirable to tune the offline mapping of base stations to edge/cloud servers at runtime. In this section, we propose a game theory based online scheme to remap base stations with changed task arriving rates to edge/cloud servers.

A. Formulate Remapping Game

We formulate the remapping of base stations with changed task arriving rates to multiple edge/cloud servers as a cooperative game. In the game, the players are these base stations with changed task arrival rates at runtime. For convenience, let $\mathcal{B}' = \{B_1, B_2, \cdots, B_Q\} \subseteq \mathcal{B}$ be the set of total Q base stations to be remapped. The qth element in set $\mathcal{B}'$ is represented as B_q. The Q players cooperate and compete with each other for edge/cloud servers in order to minimize not only their respective expected response time but also the system expected response time. After a few rounds of games, they will reach a double-win agreement that is referred

to as Nash bargaining solution [23]. At that moment, the system expected response time is minimized and the fairness in expected response time of base stations is also maximized.

Mathematically, the objective function of a cooperative game is expressed as maximizing the product of utility functions associated with individual players. Let $\mathcal{A}_{q,n}$ be a binary decision variable. If base station B_q is remapped to edge/cloud server S_n, $\mathcal{A}_{q,n}=1$ holds; otherwise, $\mathcal{A}_{q,n}=0$ holds. In the context of our remapping game, the utility function of base station B_q is defined as

$$f(\mathcal{A}_{q,n}) = 1 - \sum\nolimits_{n=0}^{N} \mathcal{A}_{q,n}(W_{delay}^{(1),q,n} + W_{delay}^{(2),q,n}), \quad (16)$$

where the expected response time $\sum_{n=0}^{N} \mathcal{A}_{q,n}(W_{delay}^{(1),q,n} + W_{delay}^{(2),q,n})$ is normalized to the range of $[0,1]$. Equation (16) indicates that the longer the expected response time of a base station, the smaller the utility function value of the base station. Formally, the remapping game is readily formulated as follows.

$$\max \quad \mathcal{H}(\mathcal{A}_{q,n}) = \prod_{q=1}^{Q} \Big(1 - \sum_{n=0}^{N} \mathcal{A}_{q,n}(W_{delay}^{(1),q,n} + W_{delay}^{(2),q,n})\Big) \quad (17)$$

$$\text{s. t.} \quad \mathcal{A}_{q,n} = 0, 1, \quad (18)$$

$$\sum\nolimits_{n=0}^{N} \mathcal{A}_{q,n} = 1, \quad (19)$$

$$\sum\nolimits_{q=1}^{Q} \mathcal{A}_{q,n}\lambda_q^{on} \le \Delta p_n. \quad (20)$$

The objective function (17) aims to maximize the product of utility functions of Q base stations. Constraints (18)-(19) ensure that every base station B_q is exactly remapped to one edge/cloud server. Constraint (20) indicates that the task arriving rate at edge/cloud server S_n cannot exceed the computation capacity of edge/cloud server S_n. In (20), λ_q^{on} is the task arrival rate at base station B_q at runtime. Δp_n suggests the remaining computation capacity of edge/cloud server S_n for remapped based stations, and it is calculated as

$$\Delta p_n = p_n - \sum\nolimits_{m=Q+1}^{M} \mathcal{A}_{m,n}\lambda_m^{on}. \quad (21)$$

For base stations from B_{Q+1} to B_M, their online task arriving rates are equal to the offline task arriving rates, which indicates that $\lambda_m^{on} = \lambda_m^{off}$ holds for $\forall m \in [Q+1, \cdots, M]$.

B. Derive Nash Bargaining Solution

Oftentimes, a cooperative game can be converted into a convex optimization problem that refers to minimizing a convex function over convex constraints [24]. In order to convert the problem in (17)-(20) into a convex optimization problem, we perform three operations: 1) take the logarithm of objective function (17), 2) relax the constraint that $\mathcal{A}_{q,n}$ only equals to 0 or 1, and 3) introduce another variable $\mathcal{C}_{q,n}$ satisfying $\mathcal{C}_{q,n} + \mathcal{A}_{q,n} = 0$. Specifically, by taking the logarithm of objective function (17), we have $\max \sum_{q=1}^{Q} \ln\Big(1 - \sum_{n=0}^{N} \mathcal{A}_{q,n}(W_{delay}^{(1),q,n} + W_{delay}^{(2),q,n})\Big)$. It is easy to observe that this logarithmical objective function is equivalent to $\min -\sum_{q=1}^{Q} \ln\Big(1 - \sum_{n=0}^{N} \mathcal{A}_{q,n}(W_{delay}^{(1),q,n} + W_{delay}^{(2),q,n})\Big)$. Furthermore, we relax the constraint that $\mathcal{A}_{q,n}$ only equals to 0 or 1, that is, $\mathcal{A}_{q,n}$ can take any real value in the range of $[0,1]$. In summary, the problem in (17)-(20) is rewritten as

$$\min\ \mathcal{H}(\mathcal{C}_{q,n}) = -\sum_{q=1}^{Q} \ln\Big(1 + \sum_{n=0}^{N} \mathcal{C}_{q,n}(W_{delay}^{(1),q,n} + W_{delay}^{(2),q,n})\Big) \quad (22)$$

$$\text{s. t.} \quad \mathcal{C}_{q,n} \le 0, \quad (23)$$

$$\sum\nolimits_{n=0}^{N} \mathcal{C}_{q,n} = -1, \quad (24)$$

$$\sum\nolimits_{q=1}^{Q} (-\mathcal{C}_{q,n})\lambda_q \le \Delta p_n. \quad (25)$$

Since $\mathcal{A}_{q,n}$ falls into the interval of $[0,1]$ and $\mathcal{C}_{q,n} + \mathcal{A}_{q,n} = 0$ holds, we have constraint (23). The new objective function (22) and constraints (24)-(25) are obtained by substituting $\mathcal{C}_{q,n} = -\mathcal{A}_{q,n}$ into the old objective function (17) and constraints (19)-(20), respectively.

From (22), the first and second derivatives of objective function $\mathcal{H}(\mathcal{C}_{q,n})$ with respect to variable $\mathcal{C}_{q,n}$ are derived as

$$\frac{d\mathcal{H}}{d\mathcal{C}_{q,n}} = -\sum\nolimits_{q=1}^{Q} \frac{\sum_{n=0}^{N}(W_{delay}^{(1),q,n} + W_{delay}^{(2),q,n})}{1 + \sum_{n=0}^{N} \mathcal{C}_{q,n}(W_{delay}^{(1),q,n} + W_{delay}^{(2),q,n})}, \quad (26)$$

$$\frac{d^2\mathcal{H}}{d\mathcal{C}_{q,n}^2} = \sum\nolimits_{q=1}^{Q} \frac{(\sum_{n=0}^{N}(W_{delay}^{(1),q,n} + W_{delay}^{(2),q,n}))^2}{\Big(1 + \sum_{n=0}^{N} \mathcal{C}_{q,n}(W_{delay}^{(1),q,n} + W_{delay}^{(2),q,n})\Big)^2}. \quad (27)$$

Since $\frac{d^2\mathcal{H}}{d\mathcal{C}_{q,n}^2} \ge 0$ holds, the objective function (22) is convex. Besides, it is obvious that the domain of objective function (22) is convex. According to the definition of convex optimization as mentioned before, the problem in (22)-(25) is convex, which motives us to adopt the Lagrange multiplier technique to derive an optimal solution. The Lagrangian function $\mathcal{L}(\mathcal{C}_{q,n}, \chi, \psi_{q,n}, \omega_{q,n})$ is depicted as

$$\begin{aligned}\mathcal{L}(\,\cdot\,) = &-\sum\nolimits_{q=1}^{Q} \ln\Big(1 + \sum\nolimits_{n=0}^{N} \mathcal{C}_{q,n}(W_{delay}^{(1),q,n} + W_{delay}^{(2),q,n})\Big) \\ &+ \chi(\sum\nolimits_{n=0}^{N} \mathcal{C}_{q,n} + 1) + \psi_{q,n}\mathcal{C}_{q,n} - \omega_{q,n}(\sum\nolimits_{q=1}^{Q} \mathcal{C}_{q,n}\lambda_q, \\ &+ \Delta p_n)\chi \in \mathbb{R}, \psi_{q,n} \ge 0, \omega_{q,n} \ge 0, \quad (28)\end{aligned}$$

where χ, $\psi_{q,n}$, and $\omega_{q,n}$ are Lagrange multipliers.

From (28), the first order partial derivative of Lagrangian function $\mathcal{L}(\,\cdot\,)$ with respect to variable $\mathcal{C}_{q,n}$ is given by

$$\frac{\partial\mathcal{L}(\,\cdot\,)}{\partial\mathcal{C}_{q,n}} = \frac{d\mathcal{H}}{d\mathcal{C}_{q,n}} + \chi(N+1) + \psi_{q,n} - \omega_{q,n}\sum_{q=1}^{Q} \lambda_q^{on}. \quad (29)$$

Since optimal $\mathcal{C}_{q,n}^*$ occurs when (29) equals to zero, we have

$$\frac{d\mathcal{H}}{d\mathcal{C}_{q,n}} = \omega_{q,n}^* \sum\nolimits_{q=1}^{Q} \lambda_q^{on} - \chi^*(N+1) - \psi_{q,n}^*, \quad (30)$$

where χ^*, $\psi_{q,n}^*$, and $\omega_{q,n}^*$ denote optimal Lagrange multipliers.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2020.2975897, IEEE Transactions on Industrial Informatics

By substituting (26) into (30), we observe that $\mathcal{C}^*_{q,n}$ satisfies

$$\mathcal{C}^*_{q,n} = \frac{\ln Q + \ln(1+\overline{W^*_Q})}{\chi^*(N+1) + \psi^*_{q,n} - \omega^*_{q,n}\sum_{q=1}^{Q}\lambda^{on}_q}. \quad (31)$$

$\overline{W^*_Q}$ is the minimal expected response time on average of the Q base stations, and it is given by

$$\overline{W^*_Q} = \frac{1}{Q}\sum_{k=1,k\neq q}^{Q}\sum_{n=0}^{N}(-\mathcal{C}^*_{k,n})(W^{(1),k,n}_{delay} + W^{(2),k,n}_{delay}) + \frac{1}{Q}\sum_{n=0}^{N}(-\mathcal{C}^*_{q,n})(W^{(1),q,n}_{delay} + W^{(2),q,n}_{delay}). \quad (32)$$

Since the two sides of (31) are coupled with $\mathcal{C}^*_{q,n}$, directly obtaining $\mathcal{C}^*_{q,n}$ from this equation is extremely difficult. For simplicity, we use $\widehat{W^*_Q}$ to estimate $\overline{W^*_Q}$, that is,

$$\overline{W^*_Q} \approx \widehat{W^*_Q} = \frac{W^{*avg}_{delay}\sum_{m=1}^{M}\lambda^{on}_m}{\sum_{m=1}^{M}\lambda^{off}_m} - \frac{\sum_{k=Q+1}^{M}W^{*,tot,k}_{delay}}{M-Q}. \quad (33)$$

$W^{*,tot,k}_{delay}$ and W^{*avg}_{delay} are the static minimal expected response time of base station B_k and the whole system, respectively, both of which are derived by the offline stage.

As indicated in (31), the key in obtaining the best remapping strategy is to derive three optimum Lagrange multipliers χ^*, $\psi^*_{q,n}$, and $\omega^*_{q,n}$, which is generally performed by solving the dual problem of the original problem in (22)-(25). According to [24], the dual problem $\mathcal{D}(\chi,\psi_{q,n},\omega_{q,n})$ with no duality gap is expressed as

$$\min \quad \mathcal{D}(\,\cdot\,) = \frac{\psi_{q,n}(\ln Q + \ln(1+\widehat{W^*_Q}))}{\chi(N+1)+\psi_{q,n}-\omega_{q,n}\sum_{q=1}^{Q}\lambda^{on}_q} - \sum_{q=1}^{Q}\ln\Big(1+\sum_{n=0}^{N}\frac{(\ln Q+\ln(1+\widehat{W^*_Q}))(W^{(1),q,n}_{delay}+W^{(2),q,n}_{delay})}{\chi(N+1)+\psi_{q,n}-\omega_{q,n}\sum_{q=1}^{Q}\lambda^{on}_q}\Big) - \omega_{q,n}\Big(\sum_{q=1}^{Q}\frac{\lambda^{on}_q(\ln Q+\ln(1+\widehat{W^*_Q}))}{\chi(N+1)+\psi_{q,n}-\omega_{q,n}\sum_{q=1}^{Q}\lambda^{on}_q}+\Delta p_n\Big) \quad (34)$$

$$\text{s. t.} \quad 0\le\psi_{q,n}, \quad 0\le\omega_{q,n}, \quad (35)$$

$$\frac{\ln Q+\ln(1+\widehat{W^*_Q})}{\omega^*_{q,n}\sum_{q=1}^{Q}\lambda^{on}_q - \chi^*(N+1)-\psi^*_{q,n}} \le 1. \quad (36)$$

It is easy to validate that the above dual problem is convex because both the objective function and the constraint set are convex. A significant property of convex optimization problem is the uniqueness of its optimum solution. This property inspires us to readily adopt any kind of greedy search algorithms (e.g., gradient projection method) to solve our dual problem since the derived local optimum solution is in fact the global optimum solution. Recall that during solving the convex problem in (22)-(25), the variable $\mathcal{C}_{q,n}$ falls into the range of $[-1,0]$. However, this indicates that base station B_q can be remapped to more than one edge/cloud servers, which will result in the violation of constraint (13). Given this, we iteratively select the minimal value $\mathcal{C}^*_{q,n}$ from array $(\mathcal{C}^*_{q,0},\mathcal{C}^*_{q,1},\cdots,\mathcal{C}^*_{q,N})$ and try to set it to the value -1 until the first feasible solution under constraint (25) is found. At that time, except $\mathcal{C}^*_{q,n}$, all the elements in array $(\mathcal{C}^*_{q,0},\mathcal{C}^*_{q,1},\cdots,\mathcal{C}^*_{q,N})$ are set to 0. Considering $\mathcal{C}_{q,n}+\mathcal{A}_{q,n}=0$, we finally have the optimal remapping solution as follows.

$$\mathcal{A}^*_{q,n} = \begin{cases} 1 & n = \arg_n\min(\mathcal{C}^*_{q,n}|S_n\in\mathcal{S}), \\ 0 & \text{otherwise.}\end{cases} \quad (37)$$

Algorithm 2: RBEC: Remap Base Stations with Changed Task Arrival Rates to Edge/Cloud Servers

Input: $\mathcal{P}^*$, $\lambda^{off}=\{\lambda^{off}_1,\lambda^{off}_2,\cdots,\lambda^{off}_M\}$, $\lambda^{on}=\{\lambda^{on}_1,\lambda^{on}_2,\cdots,\lambda^{on}_M\}$.

1 $flag \leftarrow 0$;
2 **if** $\lambda^{on}\neq\lambda^{off}$ **then**
3 $flag \leftarrow 1$;
4 **if** $flag == 0$ **then**
5 **return** static mapping strategy $\mathcal{A}^*$.
6 **else**
7 let $\mathcal{B}'=\{B_1,B_2,\cdots,B_Q\}$ be the set of Q base stations with changed task arrival rates;
8 **for** $n=0$ to N **do**
9 calculate Δp_n using (21);
10 **for** $q=1$ to Q **do**
11 **for** $n=0$ to N **do**
12 derive $\mathcal{A}^*_{q,n}$ for base station B_q using (37) and accordingly update $\mathcal{A}^*$;
13 **return** updated mapping strategy $\mathcal{A}^*$.

C. Game Theory Based Online Algorithm

Algorithm 2 describes our game theory based online scheme. Inputs to the algorithm are the offline solution $\mathcal{P}^*$, the set $\lambda^{off}=\{\lambda^{off}_1,\lambda^{off}_2,\cdots,\lambda^{off}_M\}$ of offline task arriving rates, and the set $\lambda^{on}=\{\lambda^{on}_1,\lambda^{on}_2,\cdots,\lambda^{on}_M\}$ of online task arriving rates. Line 1 of the algorithm initializes the flag of task arriving rates to 0. Lines 2-3 check whether or not the task arriving rates at base stations are changed at runtime. If the answer is no, the algorithm directly returns the static mapping strategy $\mathcal{A}^*$ generated by the offline stage (lines 4-5). If the answer is yes, the algorithm first sets the flag of task arriving rates to 1 and then triggers the game theoretic remapping heuristic (lines 6-13). Line 7 constructs a set $\mathcal{B}'$ of total Q base stations with changed task arrival rates. Lines 8-9 calculate the remaining computation capacity of each edge/cloud server. Lines 10-12 redetermine the optimal edge/cloud servers for each base station $B_q\in\mathcal{B}'$ and accordingly update the mapping strategy $\mathcal{A}^*$. Line 13 returns the updated mapping strategy $\mathcal{A}^*$.

TABLE I: Parameters of heterogenous edge servers [26]–[29].

server type	HPE ProLiant	Dell R230	Lenovo TS250	Inspur NP3020
# of servers	50	50	50	50
frequency per core of a server	3.4GHz	3.0GHz	3.9GHz	3.0GHz
# of cores per server	4	6	2	4
computation capacity per server	13.6GHz	18.0GHz	7.8GHz	12.0GHz
normal distribution of execution time (ms)	$\mathcal{N}(20,5)$	$\mathcal{N}(14,8)$	$\mathcal{N}(35,15)$	$\mathcal{N}(17,10)$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2020.2975897, IEEE Transactions on Industrial Informatics

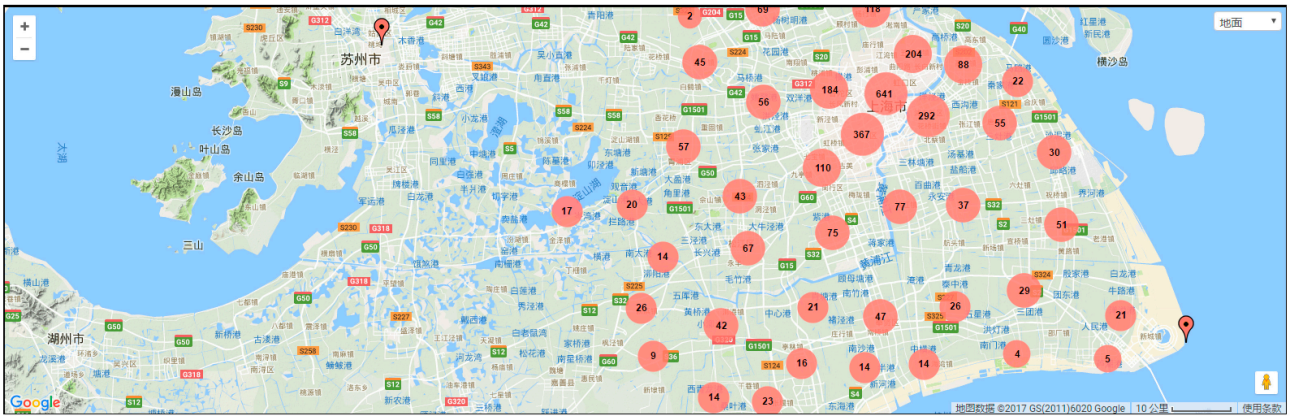


Fig. 3: The location distribution of 3233 base stations from Shanghai Telecom [17], [25]. The number in every red-marked circle indicates how many base stations have already been placed in this region.

VI. Evaluation

A. Experimental Settings

We conduct extensive simulation experiments to validate the effectiveness of our proposed solution. A set consisting of 3233 base stations from Shanghai Telecom [17], [25] is adopted in simulations, and the geographical distribution of these base stations is demonstrated in Fig. 3. The average task arriving rate and data volume of each base station are in the ranges of $[4 \times 10^6, 6 \times 10^8]$ and $[1, 100]$Mb, respectively [30]–[33]. The communication capacity of each link is randomly selected in the range of $[100, 1000]$KB/s [34]. The propagation rate of electromagnetic waves is set to 2×10^5km/s [19]. Based on the Microsoft Azure China (Shanghai) [35], we select a server equipped with 10 cores and 3.6GHz operating frequency per core as the cloud server S_0. The computation capacity p_0 of cloud server S_0 is then calculated as $10 \times 3.6\text{GHz} = 36.0\text{GHz}$. The location of cloud server S_0 is 31.0202 latitude north and 121.4503 longitude east. The task execution time of cloud server S_0 is assumed to follow a normal distribution with mean 10ms and variance 4 [36]. Besides, we construct an edge server set based on the HPE ProLiant MicroServer Gen10 servers [26], Dell R230 servers [27], Lenovo TS250 servers [28], and Inspur NP3020 servers [29]. The parameters of these edge servers are listed in TABLE I.

As introduced in Section III-B, our solution consists of offline and online algorithms. Consequently, two sets of comparative experiments are carried out: one is for offline algorithm PEMB and the other is for online algorithm RBEC. In the first set of simulations, we compare the offline algorithm PEMB with state-of-the-art benchmarking algorithms WRES [17] and DAPP [37]. In the second set of simulations, we compare the online algorithm RBEC with representative benchmarking algorithms RALL and DGEN [38]. The benchmarking algorithms for comparison are summarized as follows.

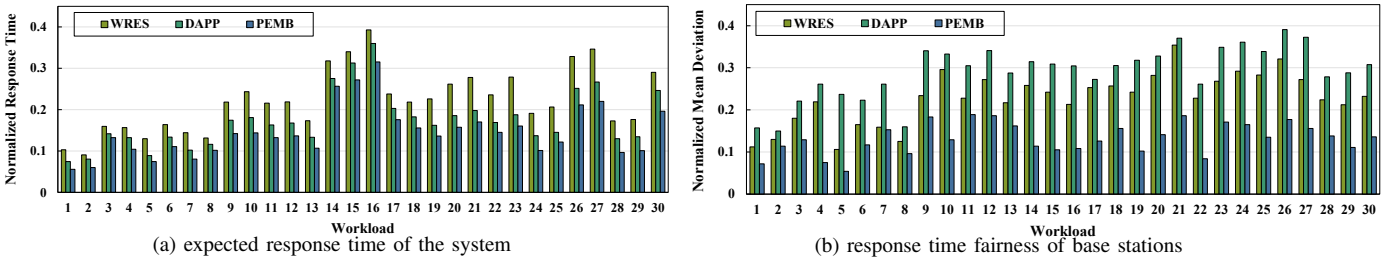
- **DAPP** [37] formulates edge server placement as the classical capacitated K-median problem in a graph such that the system expected response time can be minimized. However, both the edge-cloud computing framework and the response time fairness are not considered.
- **WRES** [17] formulates edge server placement as bi-objective optimization of making balance the workloads of edge servers and minimizing the access delay between mobile users and edge servers. It aims to jointly optimize workloads and response time of edge servers without the help of cloud servers, where the MILP technique is adopted to conduct the deployment of edge servers.
- **RALL** aims to optimize both the system expected response time and the fairness in expected response time of base stations at runtime. Its main idea is to remap all base stations by using ILP technique.
- **DGEN**, similar to scheme RALL, also aims to optimize both the system expected response time and response time fairness of base stations at runtime. It utilizes the technique of genetic algorithm in [38] to only remap base stations with changed task arrival rates.
- **NAVE** is a naive online approach that directly adopts the outputs of our offline scheme PEMB, and it takes no action to deal with the situations where task arrival rates at some base stations have already changed at runtime.

The above algorithms are all implemented in C++, and both sets of simulation experiments are conducted on a desktop computer equipped with 16GB memory and Intel i5 dual-Core 3.5GHz processor. We perform 3000 experiments with varying workload settings of base stations to compare the proposed solution with benchmarking schemes. Note the workload represents different settings of base stations. The parameters in each setting include the task arrival rate, task data volume, and link capacity of the base station. We perform 3000 experiments of different workload settings, and each data point in the result figures is averaged over 100 experiments.

B. Comparison Results

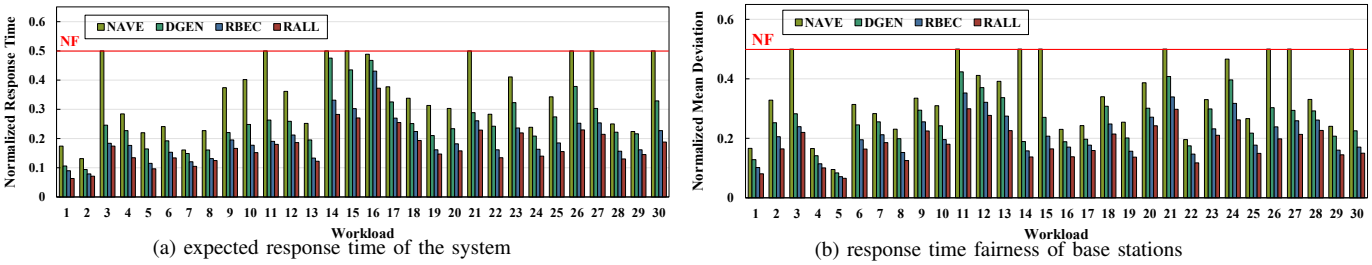
1) Comparison for Offline Scheme: Fig. 4a compares the system expected response time achieved by our offline algorithm PEMB and benchmarking algorithms WRES [17], and DAPP [37]. As observed, PEMB dramatically reduces the system expected response time. Specifically, the system expected response time of PEMB is averagely 34.17% and 15.97% smaller than that of WRES [17] and DAPP [37], respectively.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2020.2975897, IEEE Transactions on Industrial Informatics



(a) expected response time of the system (b) response time fairness of base stations

Fig. 4: The system expected response time and response time fairness achieved by different offline algorithms.



(a) expected response time of the system (b) response time fairness of base stations

Fig. 5: The system expected response time and response time fairness achieved by different online algorithms.

Moreover, the highest improvement achieved by PEMB is up to 47.37%. For example, in the 24th data point, the system expected response time of PEMB and WRES [17] are 0.10 and 0.19, respectively. This is because our scheme utilizes the edge-cloud computing framework and the heterogeneity of edge servers to improve server utilization, and thus reduces the system expected response time.

Fig. 4b plots the mean deviation of expected response time achieved by our offline algorithm PEMB and benchmarking algorithms WRES [17] and DAPP [37]. It has been demonstrated in the figure that PEMB greatly refines performance fairness among base stations. To be specific, the mean deviation of expected response time achieved by PEMB is 42.26% and 54.60% smaller on average than that of WRES [17] and DAPP [37], respectively. In addition, the highest improvement achieved by PEMB is up to 79.17%. For instance, in the 5th data point, the mean deviation of expected response time of PEMB and DAPP [37] are 0.05 and 0.24, respectively. This is because PEMB aims to improve both the system expected response time and the response time fairness of base stations.

2) Comparison for Online Scheme: In this set of simulation experiments, we arbitrarily select 1000 base stations and increase the task arrival rates at these base stations by $5\%-20\%$. Fig. 5a displays the system expected response time achieved by our online algorithm RBEC and three benchmarking online algorithms NAVE, DGEN [38], and RALL. The notation "NF" indicates that the tasks of mobile users cannot be feasibly scheduled due to the violation of constraint (15). As shown in the figure, the system expected response time of RBEC is averagely 43.09% and 23.25% smaller than that of methods NAVE and DGEN [38], respectively. Besides, this figure also reveals that the system expected response time achieved by RBEC is averagely 12.56% higher than that of method RALL. Fig. 5b exhibits the mean deviation of expected response time achieved by different online algorithms. This figure shows that RBEC achieves striking fairness improvement (up to 71.60%) compared with benchmarking methods. Meanwhile, one can also see that our algorithm RBEC is inferior to method RALL, with 16.61% performance difference on average. We draw similar observations when other parameters are changed.

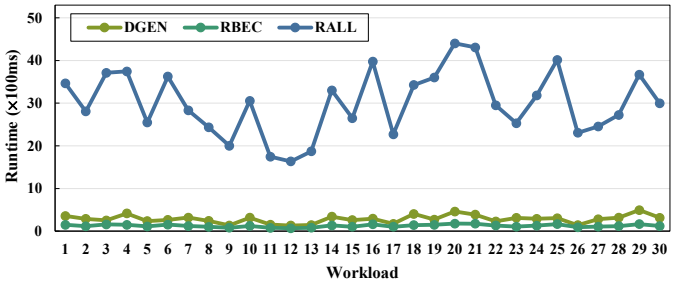


Fig. 6: The runtime of different online algorithms.

Fig. 6 demonstrates the runtime of our algorithm RBEC and benchmarking algorithms RALL and DGEN [38]. Note that the online time overhead of method NAVE is equal to zero since it takes no action at runtime. The results clearly show that our algorithm RBEC averagely achieves 24.08 and 2.26 times of speedup compared with methods RALL and DGEN [38], respectively. Moreover, the speedup of our method RBEC can be up to 25.85 times. The speedup is calculated as the ratio of the runtime consumed by the benchmarking method under test to the runtime consumed by our algorithm RBEC.

From the results shown in Fig. 5 and Fig. 6, we find the online scheme NAVE directly adopting offline solutions at runtime cannot always guarantee task schedulability when task arrival rates at base stations have increased. This is because the task arriving rate at an edge/cloud server at runtime is extremely likely to exceed the maximal computation capacity of the edge/cloud server. Meanwhile, we also find the method RALL outperforms our online scheme RBEC in terms of

system expected response time and response time fairness of base stations. However, it incurs significant time overhead to derive an optimal remapping solution at runtime. On the contrary, our scheme RBEC has the ability to achieve a better tradeoff between the quality of the produced online solution and the time overhead of generating the online solution.

VII. Conclusion

In this paper, we develop a two-stage approach to place and allocate heterogeneous edge servers for response time optimization in mobile edge-cloud computing systems. The offline stage determines the optimal deployment strategy of edge servers and the online stage deals with the dynamic characteristic of user movements at runtime. More importantly, the developed approach jointly considers the edge-cloud computing framework and the performance fairness of base stations. Simulation results show that our algorithms can greatly improve the expected response time of the entire system and individual base stations.

References

[1] H. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: architecture, applications, and approaches," *Wireless Communications and Mobile Computing*, vol. 13, no. 18, pp. 1587–1611, 2013.

[2] K. Cao, J. Zhou, G. Xu, T. Wei, and S. Hu, "Exploring renewable-adaptive computation offloading for hierarchical QoS optimization in fog computing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2019.

[3] H. Liu, F. Eldarrat, H. Alqahtani, A. Reznik, X. Foy, and Y. Zhang, "Mobile edge cloud system: architectures, challenges, and approaches," *IEEE Systems Journal*, vol. 12, no. 13, pp. 2495–2508, 2018.

[4] G. Aujla, N. Kumar, A. Zomaya, and R. Ranjan, "Optimal decision making for big data processing at edge-cloud environment: an SDN perspective," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 2, pp. 778–789, 2018.

[5] Y. Li and S. Wang, "An energy-aware edge server placement algorithm in mobile edge computing," *in Proceedings of the IEEE International Conference on Edge Computing*, pp. 66–73, 2018.

[6] S. Yang, F. Li, M. Shen, X. Chen, X. Fu, and Y. Wang, "Cloudlet placement and task allocation in mobile edge computing," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 5853–5863, 2019.

[7] H. Yao, C. Bai, M. Xiong, D. Zeng, and Z. Fu, "Heterogeneous cloudlet deployment and user-cloudlet association toward cost effective fog computing," *Concurrency and Computation: Practice and Experience*, vol. 29, no. 16, pp. 1–9, 2017.

[8] L. Zhao, J. Liu, Y. Shi, W. Sun, and H. Guo, "Optimal placement of virtual machines in mobile edge computing," *in Proceedings of the IEEE Global Communications Conference*, pp. 1–6, 2017.

[9] L. Yang, J. Cao, G. Liang, and X. Han, "Cost aware service placement and load dispatching in mobile cloud systems," *IEEE Transactions on Computers*, vol. 65, no. 5, pp. 1440–1452, 2016.

[10] L. Chen, J. Wu, G. Zhou, and L. Ma, "QUICK: QoS-guaranteed efficient cloudlet placement in wireless metropolitan area networks," *The Journal of Supercomputing*, vol. 74, no. 8, pp. 4037–4059, 2018.

[11] M. Jia, J. Cao, and W. Liang, "Optimal cloudlet placement and user to cloudlet allocation in wireless metropolitan area networks," *IEEE Transactions on Cloud Computing*, vol. 5, no. 4, pp. 725–737, 2017.

[12] U. Shaukat, E. Ahmed, Z. Anwar, and F. Xia, "Cloudlet deployment in local wireless networks: motivation, architectures, applications, and open challenges," *Journal of Network and Computer Applications*, vol. 62, pp. 18–40, 2016.

[13] J. Xu, B. Palanisamy, H. Ludwig, and Q. Wang, "Zenith: utility-aware resource allocation for edge computing," *in Proceedings of the IEEE International Conference on Edge Computing*, pp. 47–54, 2017.

[14] J. Meng, W. Shi, T. Tan, and X. Li, "Cloudlet placement and minimum-delay routing in cloudlet computing," *in Proceedings of the IEEE International Conference on Big Data Computing and Communications*, pp. 297–304, 2017.

[15] L. Zhao, W. Sun, Y. Shi, and J. Liu, "Optimal placement of cloudlets for access delay minimization in SDN-based Internet of things networks," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1334–1344, 2018.

[16] S. Rashidi and S. Sharifian, "Cloudlet dynamic server selection policy for mobile task off-loading in mobile cloud computing using soft computing techniques," *The Journal of Supercomputing*, vol. 73, no. 9, pp. 3796–3820, 2017.

[17] S. Wang, Y. Zhao, J. Xu, J. Yuan, and C. Hsu, "Edge server placement in mobile edge computing," *Journal of Parallel and Distributed Computing*, vol. 127, pp. 160–168, 2019.

[18] A. Samanta, Z. Chang, and Z. Han, "Latency-oblivious distributed task scheduling for mobile edge computing," *in Proceedings of the IEEE Global Communications Conference*, pp. 1–7, 2018.

[19] K. Miller, "Communication theories: perspectives, processes, and contexts," *Macgraw-Hill*, 2005.

[20] S. Fuhrmann and R. Cooper, "Stochastic decompositions in the M/G/1 queue with generalized vacations," *Operations Research*, vol. 33, no. 5, pp. 1117–1129, 1985.

[21] W. Mendenhall, R. Beaver, and B. Beaver, "Introduction to probability and statistics," *Cengage Learning*, 2012.

[22] S. Lin, B. Schutter, Y. Xi, and H. Hellendoorn, "Fast model predictive control for urban road networks via MILP," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 3, pp. 846–856, 2011.

[23] J. Nash, "The bargaining problem," *Econometrica*, vol. 18, pp. 155–162, 1950.

[24] S. Boyd and L. Vandenberghe, "Convex optimization," *Cambridge University Press*, 2004.

[25] Shanghai Telecom, "The distribution of 3233 base stations," 2019. [Online]. Available: http://www.sguangwang.com/dataset/telecom.zip

[26] HP Inc., "HPE proliant microserver gen10 server," 2019. [Online]. Available: https://buy.hpe.com/b2c/us/en/servers/proliant-microserver/

[27] Dell Inc., "Dell poweredge r230 server," 2019. [Online]. Available: https://i.dell.com/sites/doccontent/shared-content/data-sheets/en/Documents/Dell_PowerEdge_R230_SpecSheet_final.pdf

[28] Lenovo Inc., "Lenovo thinkserver ts250," 2019. [Online]. Available: http://b2b.lenovo.com.cn/dcgProduct/server/ts250.html

[29] Inspur Inc., "Inspur np3020 servers," 2019. [Online]. Available: http://www.szinspur.com/TowerServer/NP3020/

[30] R. Jayaseelan and T. Mitra, "Temperature aware task sequencing and voltage scaling," *in Proceedings of the IEEE International Conference on Computer-Aided Design*, pp. 618–623, 2008.

[31] K. Cao, J. Zhou, P. Cong, L. Li, T. Wei, M. Chen, S. Hu, and X. Hu, "Affinity-driven modeling and scheduling for makespan optimization in heterogeneous multiprocessor systems," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 7, pp. 1189–1202, 2019.

[32] K. Cao, G. Xu, J. Zhou, T. Wei, M. Chen, and S. Hu, "QoS-adaptive approximate real-time computation for mobility-aware IoT lifetime optimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 10, pp. 1799–1810, 2019.

[33] J. Zhou, X. Hu, Y. Ma, J.Sun, T. Wei, and S. Hu, "Improving availability of multicore real-time systems suffering both permanent and transient faults," *IEEE Transactions on Computers*, vol. 68, no. 12, pp. 1785–1801, 2019.

[34] Behr Technologies Inc., "6 leading types of IoT wireless technology and their best use cases," 2019. [Online]. Available: https://behrtechnologies.com/blog/6-leading-types-of-iot-wireless-tech-and-their-best-use-cases/

[35] Microsoft Corporation, "Microsoft Azure China (Shanghai)," 2019. [Online]. Available: https://azure.microsoft.com/en-us/global-infrastructure/china/

[36] M. Jia, W. Liang, Z. Xu, M. Huang, and Y. Ma, "QoS-aware cloudlet load balancing in wireless metropolitan area networks," *IEEE Transactions on Cloud Computing*, 2018.

[37] Z. Xu, W. Liang, W. Xu, M. Jia, and S. Guo, "Efficient algorithms for capacitated cloudlet placements," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 10, pp. 2866–2880, 2016.

[38] M. Qiu, Z. Ming, J. Li, K. Gai, and Z. Zong, "Phase-change memory optimization for green cloud with genetic algorithm," *IEEE Transactions on Computers*, vol. 64, no. 12, pp. 3528–3540, 2015.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2020.2975897, IEEE Transactions on Industrial Informatics

Kun Cao is currently pursuing the Ph.D. degree in Computer Science with the School of Computer Science and Technology, East China Normal University, Shanghai, China. His current research interests are in the areas of Internet of things, real-time embedded systems, and cyber physical systems. He has published over 20 refereed papers in these areas, most of which are published in premium conferences and journals, including IEEE TCAD, IEEE TAES, IEEE TSUSC, IEEE TPDS, IEEE COMST, ACM CSUR, etc. Mr. Cao was a recipient of the Reviewer Award from Journal of Circuits, Systems, and Computers, in 2016.

Liying Li received the B.S. degree from the Department of Computer Science and Technology, East China Normal University, Shanghai, China, in 2017. She is currently pursuing the Ph.D. degree with the School of Computer Science and Technology, East China Normal University, Shanghai, China. Her current research interests are in the areas of cyber physical systems and IoT resource management.

Yangguang Cui received the B.S. degree from the Department of Computer Science and Technology, East China Normal University, Shanghai, China, in 2018. He is currently pursuing the master degree with the School of Computer Science and Technology, East China Normal University, Shanghai, China. His current research interests are in the areas of Internet of things, real-time embedded systems, and high performance computing.

Tongquan Wei (M'11-SM'19) received his Ph.D. degree in Electrical Engineering from Michigan Technological University in 2009. He is currently an Associate Professor in the Department of Computer Science and Technology at the East China Normal University. His research interests are in the areas of Internet of things (IoT), edge computing, cloud computing, and design automation of intelligent systems and cyber physical systems (CPS). He serves as a Regional Editor for Journal of Circuits, Systems, and Computers since 2012. He also served as the Guest Editor of the IEEE TII SS on Building Automation, Smart Homes, and Communities, the ACM TESC SS on Embedded Systems for Energy-Efficient, Reliable, and Secure Smart Homes, and the ACM TCPS SS on Human-Interaction-Aware Data Analytics for Cyber-Physical Systems. He is a senior member of the IEEE.

Shiyan Hu (SM'10) received his Ph.D. in Computer Engineering from Texas A&M University in 2008. He is the Professor and Chair in Cyber-Physical System Security at University of Southampton. His research interests include Cyber-Physical Systems and Cyber-Physical System Security, where he has published more than 100 refereed papers, including 50+ in premier IEEE Transactions. Prof. Hu is the Chair for IEEE Technical Committee on Cyber-Physical Systems. He is the Editor-In-Chief of IET Cyber-Physical Systems: Theory & Applications. He is an Associate Editor for IEEE Transactions on Computer-Aided Design, IEEE Transactions on Industrial Informatics, IEEE Transactions on Circuits and Systems, ACM Transactions on Design Automation for Electronic Systems, and ACM Transactions on Cyber-Physical Systems. He is a Fellow of IET and a Fellow of British Computer Society.