Energy-Aware Collaborative Service Caching in a 5G-Enabled MEC With Uncertain Payoffs

Zichuan Xu, *Member, IEEE*, Lizhen Zhou, *Student Member, IEEE*, Haipeng Dai, *Member, IEEE*, Weifa Liang, *Senior Member, IEEE*, Wanlei Zhou, *Senior Member, IEEE*, Pan Zhou, *Member, IEEE*, Wenzheng Xu, *Member, IEEE*, and Guowei Wu, *Member, IEEE*

***Abstract*— Mobile edge computing (MEC) is an enabling technology for low-latency AI applications, by caching AI services originally deployed in remote data centers to 5G base stations in network edge. Due to limited computing resource of 5G base stations, not all services can be cached in base stations to meet the resource demands of user requests. Also, if the workload of a 5G base station reaches to its resource capacity, the energy consumption of the base station will be pushed up exponentially. To reduce the energy consumption and overcome resource limitations on base stations, an alternative is to allow the base stations to collaborate with each other to admit user requests. In this paper, we investigate the problem of collaborative service caching and request offloading between a 5G-enabled MEC and remote data centers, while meeting the quality of service (QoS) requirements of users, and resource capacities on base stations that are operated by multiple selfish network service providers. We aim to maximize the total payoff of all base stations. To this end, we first propose a two-stage optimization framework: In the first stage, we develop a mechanism that adopts a best-reply rule for dynamically distributed coalition formation. In the second stage, we propose a near-optimal payoff allocation method by devising a randomized algorithm with a provable approximation ratio. We then evaluate the performance of the proposed optimization framework by extensive experimental simulations. Simulation results show that the proposed framework outperforms its counterparts by achieving at least 30% higher payoff and 20% lower energy consumption of base stations.**

***Index Terms*— Mobile edge computing (MEC), 5G networks, service caching, request offloading, game theory, approximation algorithms.**

I. Introduction

WITH the rapid development of AI technologies, our daily life is increasingly exposed to a plethora of AI applications including autonomous vehicles, face verification, online games, virtual/augmented reality (VR/AR), etc. Such applications are delay-sensitive and computationally-demanding [6]. Executing computing-intensive AI applications on user mobile devices, typically is constrained by low energy capacities on the devices, which is inefficient and may not meet the ultra-low delay requirements of their users. The emerging 5G technologies and mobile edge computing (MEC) are capable to shorten the end-to-end delays of AI applications [24], [27], [38] significantly, by pushing computing infrastructures to the edge of a 5G-enabled MEC, thereby enabling service provisioning within the proximity of mobile users. However, typical 5G base stations consume up to twice or even more power than 4G base stations in terms of data processing and data transmission [58], and this pushes up the operational costs of network service providers. Reducing the energy consumption of 5G base stations thus is urgently needed to reduce its operational cost of network service providers. It is noted that the energy consumption of 5G base stations typically become much quicker when they are overloaded due to a need for more intense communications and processing power. As such, in this paper, we investigate the enabling of collaborations among base stations in the process of service caching and task offloading, to balance the workloads from overloaded base stations to light loaded ones and thus reduce the overall energy consumption.

Due to the fully distributed nature of 5G-enabled MECs, *horizontal* collaboration among base stations and *vertical* collaboration between base stations and remote data centers have been envisioned as the keys to unleash the full potential and lower the energy consumption of 5G-enabled MECs [9], [31]. As shown in Figure 1, on one hand, horizontal collaborations

Manuscript received April 29, 2021; revised September 15, 2021; accepted October 18, 2021. Date of publication November 2, 2021; date of current version February 17, 2022. The work of Zichuan Xu is funded by the National Natural Science Foundation of China (NSFC) with grant numbers 62172068, 62172071, 61802048, 61802047, and the "Xinghai Scholar" program. The work of Weifa Liang was supported by a grant from the City University of Hong Kong with Project No: 9380137/CS. The work done by Pan Zhou is supported in part by NSFC under Grant 61972448. The associate editor coordinating the review of this article and approving it for publication was W. Saad. *(Corresponding author: Zichuan Xu.)*

Zichuan Xu, Lizhen Zhou, and Guowei Wu are with the Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province and the School of Software, Dalian University of Technology, Dalian 116620, China (e-mail: z.xu@dlut.edu.cn; zhou_lizhen@mail.dlut.edu.cn; wgwdut@dlut.edu.cn).

Haipeng Dai is with the State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China (e-mail: haipengdai@nju.edu.cn).

Weifa Liang is with the Department of Computer Science, City University of Hong Kong, Hong Kong (e-mail: weifa.liang@cityu.edu.hk).

Wanlei Zhou is with the Institute of Data Science, City University of Macau, Macau, China (e-mail: wlzhou@cityu.mo).

Pan Zhou is with the Hubei Engineering Research Center on Big Data Security, School of Cyber Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: panzhou@hust.edu.cn).

Wenzheng Xu is with the Department of Computer Science, Sichuan University, Chengdu 610065, China (e-mail: wenzheng.xu@scu.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCOMM.2021.3125034.

Digital Object Identifier 10.1109/TCOMM.2021.3125034

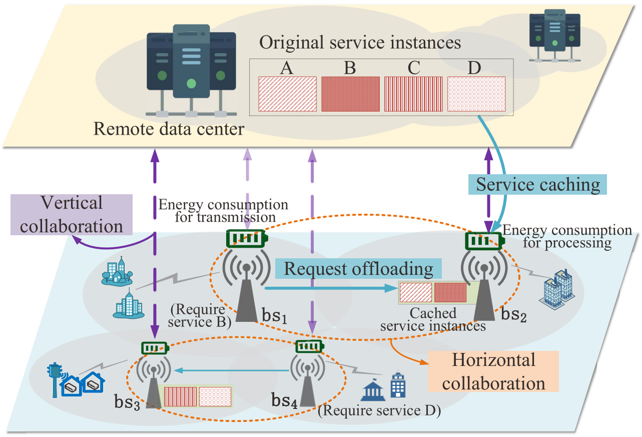


Fig. 1. An example of a 5G-enabled MEC with horizontal collaborations among base stations and vertical collaborations between base stations and remote data centers.

are needed to harness the limited computing resource capacities on base stations. Particularly, the amount of computing resource attached to a base station is usually very limited to process all user requests. Collaborative request offloading enables a base station to forward its overloaded requests to the other base stations for processing, such that the energy consumption of base stations can be balanced and reduced. On the other hand, vertical collaborations between base stations and remote data centers are necessary to meet the service requirements of user requests. Specifically, network service providers need to cache their services originally deployed in remote data centers to the base stations of the 5G-enabled MEC. Such services may not be able to permanently cached in the 5G-enabled MEC, given the limited capacity and network dynamics. Thus, services can be cached in an on-demand manner while their original service instances are kept in remote data centers. In case the cached service instance is no longer needed, it will be removed from the 5G-enabled MEC and its original one continues to serve user requests. In addition, each service is usually stateful [19]. Since the states are critical for future request implementations, we need to keep the state consistent between the cached and original service instances. Therefore, the states generated in request processing will be synchronized between a cached service instance and its original one. In this paper, we study the problem of *collaborative service caching and request offloading* for stateful network services in a 5G-enabled MEC via leveraging both horizontal and vertical collaborations.

Collaborative service caching and request offloading in a 5G-enabled MEC pose several challenges. First, 5G small-cell base stations broadcast over millimeter wave (mmWave), whose transmission range is within only a few hundred meters. A single network service provider may not just deploy a necessary number of base stations to cover all its users, due to a limited budget. This means that each base station needs to collaborate with the other base stations owned by different network service providers [12], [30]. Since network service providers are selfish, each base station has uncertain information on the payoff functions of the other base stations. Therefore, designing a near-optimal, distributed coalition formation mechanism with uncertain payoff functions is the first key challenge to enable the collaborative service caching and request offloading in 5G-enabled MECs. Second, implementing requests in base stations consumes the significant amount of energy on data transmission and processing of the requests. For example, reports show that 5G base stations consume 80% of the overall energy consumption in a 5G-enabled MEC (which consists of the power consumption of switches, servers, base stations, etc.), while their resource utilization usually does not exceed 20% [20]. Therefore, how to optimize the energy consumption of base stations in the 5G-enabled MEC is another key challenge to reduce the operational cost of 5G-network service providers.

Both service caching and request offloading are effective approaches to optimize overall system performance of 5G-enabled MECs. Service caching however is different from request offloading. Service caching refers to the provisioning and caching of various application services and their related libraries/databases in base stations, while request offloading refers to the assignment of user computation requests to different servers. Furthermore, collaborative service caching and request offloading focus on both horizontal collaborations among base stations and vertical collaborations between base stations and remote data centers, to achieve complementary and mutual benefits of remote data centers and local base stations.

There are several studies on request offloading and service placement [15], [36], [44], [56], most of them did not explore horizontal collaboration among base stations in an MEC network [36], [44]. However, there are studies that consider edge collaborations [13], [35], [47]. For example, the study in [47] explored the collaboration among services, i.e., services cached in the same cloudlet are considered to form a coalition. The study in [13] considered collaborations among base stations. Most of the studies ignored the energy consumption of base stations. Furthermore, the diversity of user computing resource demands and the vertical collaboration between base stations and remote data centers have not been considered.

To the best of our knowledge, we are the first to explore energy-aware service caching and request offloading for stateful network services via leveraging both horizontal collaboration among base stations in a 5G-enabled MEC and vertical collaboration between base stations and remote data centers. The main contributions of this paper are summarized as follows.

- We consider a 5G-enabled MEC managed by multiple network service providers. We formulate the collaborative service caching and request offloading problem with uncertain payoff functions of different network service providers, by considering both energy consumption of base stations and delay requirements of user requests.
- We propose an optimization framework that consists of a distributed coalition formation procedure and a near-optimal payoff allocation algorithm, with the aim to maximize the total payoff of all coalitions. The framework allows each base station to make its own

decision independently based on a minimum set of information, such that the frequency of interactions (message exchanges for coalition formation) in the system is minimized.
- We evaluate the performance of the proposed optimization framework through extensive experimental simulations. Simulation results show that the performance of the proposed algorithms outperform existing studies by achieving at least 30% higher payoffs and 20% lower energy consumption of base stations.

The remainder of the paper is arranged as follows. Section II surveys the state-of-the-art on this topic. Section III introduces the system model, notations and problem formulation. Section IV presents the optimization framework for the problem. Section V provides experimental results on the performance of the proposed algorithms, and Section VI concludes the paper.

II. Related Work

Recently, computation offloading is the central focus of many studies in mobile edge computing [22], [33], [38], [43], [45], [54], [55], which deals with offloading user requests from mobile devices to edge servers. For example, Chen and Hao [10] investigated the problem of task offloading in a software-defined ultra-dense edge network, with the aim to minimize the processing delay of the task while saving battery life of user equipment. Chen *et al.* [11] developed a novel online small-cell base stations peer offloading framework to avoid large latencies at overloaded base stations and provide high quality of service to end users. These studies however did not consider service caching and task offloading for stateful services with state updating from remote data centers to local base stations in an MEC.

Service caching and placement has also been studied in MECs in the past years to improve the quality of service requirements of mobile applications [5], [8], [16], [21], [23], [31], [34], [39], [46], [48]–[50], [53], [56]. Existing works on service caching focused mainly on when to cache, where to cache, and what types of services to cache. For example, Bi *et al.* [8] considered a single edge server that assists mobile users to execute a series of computing tasks, where service caching and computing offloading decision can be optimized separately. Gao *et al.* [16] considered the problem of joint network selection and service placement in MEC environments, with an aim to improve the switching and communication delays of mobile users. In [21], a controller named S-Cache was designed to cache most popular services based on CPU, RAM, and disk resources in an edge cloud. In [31], edge service providers made caching decisions based on whether they can meet the users' utility requirements. Ma *et al.* [34] leveraged the power of user mobility prediction to find an optimal service placement in MEC networks. Zhang *et al.* [52] proposed a novel service pushing and caching scheme in overlay networks while the cache size of the router, the maximum number of requests each router can serve, and the entire time delay are limited. Poularakis *et al.* [41] investigated the service placement and the routing of user requests to corresponding edge servers, and also considered multidimensional (storage-computation-communication) constraints. Xie *et al.* [46] investigated the problem of offloading user tasks to base stations and caching the service requested by the user, such that the computation latency is minimized while meeting a long-term energy consumption constraint. Xu *et al.* [48] devised a stable Stackelberg congestion game between an infrastructure provider and multiple network service providers for the service caching problem in a mobile service market. Xu *et al.* [49] considered bursty user demands under the uncertainty of processing latency in MECs, and solved the problem of dynamic service caching and task offloading. Zhang *et al.* [56] investigated the popularity-based caching strategy while considering the capacity constraints on edge servers. There are also several related studies focusing on task scheduling [4], [57], request routing [14], [41], and workload scheduling [35]. The mentioned mechanisms in these studies however cannot be directly applied to our problem, because horizontal collaboration among base stations is largely ignored. Further, none of these existing studies considered the energy consumption of base stations and the uncertain payoff of base stations in MEC networks.

Although there are several investigations on horizontal service caching [13], [35], [47], most of them ignored the vertical collaboration between base stations and remote data centers. For example, Ma *et al.* [35] explored the cooperation between edge nodes to optimize service caching and workload scheduling with the aim to minimize service response time and outsourcing business volume. Xu *et al.* [47] studied the problem of service caching among multiple network service providers in mobile edge networks from a static perspective.

Closely related to our study in this paper is the study in [13], which investigated the collaborative service placement problem that small-cell base stations can collaboratively decide which services to place so that the number of available services at an edge system is maximized. Our work differs from the study in [13] as follows. We consider dynamic and energy-aware horizontal and vertical collaborations with selfish players and uncertain payoffs in MECs, while only static horizontal collaborations among base stations are explored in [13].

In summary, we are the first to investigate the problem of energy-aware service caching and request offloading in a 5G-enabled MEC via enabling both horizontal collaboration among base stations and vertical collaboration between base stations and remote data centers. We assume that each base station does not know the payoffs of other base stations. Making decisions under the uncertainty of payoffs is largely ignored by existing studies on edge collaborations. We are the first to develop novel coalition formation mechanisms for collaborative service caching and request offloading under the payoff uncertainty.

III. System Model and Problem Formulation

In this section, we first introduce the system model, notions and notations. We then describe the delay and energy consumption models, and finally we define the problems precisely.

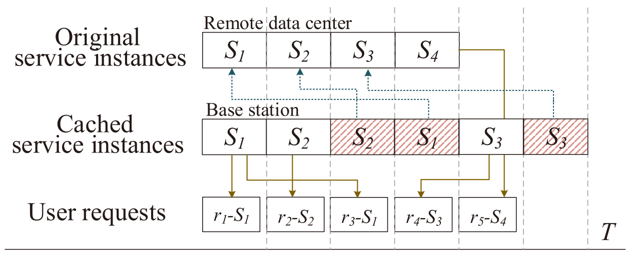


Fig. 2. An example of temporary service caching. The cached service instances of S_1, S_2, and S_3 in the base station process requests r_1, r_2, and r_3, and r_4, respectively. However, r_5 is processed by the original service instance in the remote data center, because there is no cached service instance of S_4 in the base station. When the service is no longer requested by the user, the cached service instances of S_1, S_2, and S_3 are destroyed, and the state data generated needs to be synchronized with the original service instances.

A. System Model

We consider a 5G-enabled mobile edge cloud (MEC) $G = (\mathcal{BS} \cup \mathcal{DC}, E)$ consisting of a set $\mathcal{BS}$ of 5G base stations and a set $\mathcal{DC}$ of data centers with abundant computing resources, as shown in Fig. 1. Base stations are owned by different network service providers and deployed in densely populated urban areas, such as shopping malls, sports centers, airports and train stations. Each base station has a transmission range and coverage area, users within the coverage area of a base station can transmit and receive signals from the base station. Denote by bs_i such a base station in $\mathcal{BS}$ with $1 \leq i \leq |\mathcal{BS}|$. The coverage areas of different base stations may overlap with each other. User equipment (UE), such as mobile phones and VR headsets, access the services in the 5G-enabled MEC via connecting to their registered base stations. Each UE can be within the transmission ranges of multiple base stations. However, each user located at a location only has a single registered base station at each time. Let ue_j and $\mathcal{UE}$ be a UE and the set of UEs respectively, with $1 \leq j \leq |\mathcal{UE}|$.

In a 5G-enabled MEC, base stations have both computing and bandwidth resources to execute network services and implement user requests. The computing resource of base stations in $\mathcal{BS}$ is virtualized as containers. The bandwidth resource is to guarantee data transmission rates from/to base stations. For example, a "r5.12xlarge" VM instance in Amazon EC2 has network bandwidth resource of 10 Gbps [1]. Denote by $C(bs_i)$ and $B(bs_i)$ the computing and bandwidth capacities of base station $bs_i \in \mathcal{BS}$ in the MEC, respectively.

B. Collaborative Service Caching and Request Offloading

We focus on services that are originally deployed in remote data centers, which need to be cached into base stations of a 5G-enabled MEC to improve latencies experienced by users and save the energy consumption of the base stations. However, due to the concerns on data security and privacy, load balancing and limited computing capacity on each base station, not all base stations have cached instances of each service. We thus consider *temporary service caching* that caches services from remote data centers to base stations of the 5G-enabled MEC, which can be seen in Fig. 2. Let S_l be a service, and denote by $\mathcal{S}$ a set of such services. Clearly, we have $S_l \in \mathcal{S}$ with $1 \leq l \leq |\mathcal{S}|$. The instance of service S_l will serve user requests once it is cached into the 5G-enabled MEC, which is referred to as a *cached service instance*. The *original service instance* of S_l in the remote data center will continue serving user requests if its service is not cached to base stations, or the cached service instance has been destroyed. Each S_l is stateful, and its state data generated by a cached instance has to be updated to its original service instance in a remote data center; otherwise, the service may not function correctly when the cached instance is destroyed.

Since base stations have overlapping coverage areas in 5G-enabled MECs, there is no need to always cache an instance of a service S_l to the registered base station of a user requiring S_l. Instead, the instance of each service S_l can be cached into a nearby base station of its registered base station. Then, a request requiring S_l can be offloaded to the base station with cached service instance from the registered base station of the request. In other words, registered base stations can offload their received user requests to other base stations with cached service instances if these base stations are within a group of common interest. Such grouping of base stations is referred to as a *coalition*. We refer to such service caching as *collaborative service caching*. The key of collaborative service caching is to determine: (1) the optimal number of instances of each service S_l to cache, (2) the caching locations of the instances in the 5G-enabled MEC, and (3) finding stable coalitions of the base stations.

There is a set R of requests that need to be implemented by services in $\mathcal{S}$. Let r_j be a request in R that is represented by a tuple $\langle ue_j, S_l, W_j, \rho_j \rangle$, where ue_j is the UE that generates request r_j, S_l is its required service, W_j is the number of instructions of request r_j, and ρ_j is the amount of data of r_j. Assume that C_{unit} and B_{unit} are the amounts of computing and bandwidth resources needed in base stations to process a unit amount of data of each request, respectively. The computing and bandwidth resources demanded by request r_j thus are $C_{unit} \cdot \rho_j$ and $B_{unit} \cdot \rho_j$, respectively. To implement request r_j demanding service S_l, r_j can be offloaded to a base station with a cached service instance of S_l.

C. Delay Models

The delays incurred by implementing a request in a cached service instance and an original service instance of S_l can be significantly different. We assume that if a user request offloaded to the cached instance cannot be completed, it will be served by the original instance in the remote data center. The delays of implementing a request usually are due to the transmission of its data to the service and processing the transferred data, which are referred to as *processing delay* and *transmission delay*, respectively.

The processing delay $\delta_{i,j}$ of processing the data of request r_j in base station bs_i is

$$\delta_{i,j} = \rho_j \cdot \eta_i, \tag{1}$$

where η_i is a given constant representing how long base station bs_i processes a unit amount of data.

To make sure that the amount of data ρ_j of each request r_j is processed by its demanded service S_l, its data needs to be sent from its registered base station to the base station that has an instance of S_l. Following studies in [25], [51], the delay $\gamma_{i,j}$ of transmitting the data of r_j from its registered base station to a base station bs_i can be formulated as

$$\gamma_{i,j} = \rho_j \cdot \sigma_j \cdot h_{j,i}, \tag{2}$$

where σ_j is a given constant, representing how long a link transmits a unit amount of data of r_j, and $h_{j,i}$ is the number of hops from the registered base station of r_j to base station bs_i.

Each ue_j has a different Quality of Service (QoS) in terms of delay requirements, denote by d_j^{req} the delay requirement of request r_j. Notice that we assume that each ue_j issues a single request, this assumption can be easily extended to multiple requests, by treating each request from a different virtual user. Let y_{ji} be a binary decision variable that determines whether request r_j is offloaded to a base station bs_i with a cached instance of S_l, then the delay requirement of each r_j is, i.e.,

$$y_{ji} \cdot (\delta_{i,j} + \gamma_{i,j}) \leq d_j^{req}. \tag{3}$$

D. Energy Consumption of 5G Base Stations

Base stations of a 5G-enabled MEC consume energy on both data transmission and data processing. Specifically, each user request accesses the MEC network by sending its data to its registered base station. The data of the request may be processed in its registered base station or offloaded to a different base station for processing. As such, energy is consumed to transmit the data of the request. In addition, a significant amount of energy can be consumed by computing resources of base stations for processing the data of requests. We thus define the following two types of energy consumption of base stations.

1) Data Transmission Energy: The amount of energy consumed by base station bs_i in transferring the data of request r_j is proportional to the amount of data that needs to be transferred. Let $c_{i,j}^t$ be the amount of energy consumed due to transmitting a unit data from base station bs_i to base station bs_j if both base stations are in a coalition. Then, the amount of energy consumed $e_{i,j}^t$ by bs_i on the data transfer between bs_i and bs_j is

$$e_{i,j}^t = c_{i,j}^t \cdot \rho_j. \tag{4}$$

2) Data Processing Energy: Following existing studies [28], [29], the energy consumption of a base station bs_i for data processing is proportional to the rate of accessing its processing unit and peak power. Let $e_{i,j}^p$ be the amount of energy consumed on implementing request r_j in base station bs_i, which consists of the energy consumption of its processing unit, idle power, and leakage power, i.e.,

$$e_{i,j}^p = \delta_{i,j} \cdot \Big((\xi_i \cdot W_j / \delta_{i,j}) P_i^{max} + P_i^{idle} + P_i^{leak} \Big), \tag{5}$$

where ξ_i is a given parameter that is used to calculate the access rate of processing units as shown in the power model in [28], $\delta_{i,j}$ is the delay of processing request r_j in base station bs_i, P_i^{max} is the peak power of all processing units of bs_i, P_i^{idle} is the idle power, and P_i^{leak} is the leakage power generated by leakage current of the computing units of bs_i. Note that such parameters can be obtained following the configurations of the computing units of base stations [28].

E. Cooperative Game and Transferable Utility

A group of base stations with common interest is considered as a coalition. The base stations in the same coalition share their resources, jointly cache services, and implement user requests, such that the total payoff is maximized. Analogically, in a cooperative game, players can form coalitions and make binding agreements on how to allocate overall payoff to the members in each coalition. We thus consider each base station as a player in a cooperative game, and all base stations interact with each other to form stable coalitions.

Considering that the resources in a 5G-enabled MEC are offered to users on a pay-as-you-go basis [26]. Users pay for the implementation of their requests. Let p_j be the payment that ue_j pays for offloading its request r_j to a base station with a cached service instance for processing. We assume that the payment due to service caching and request offloading is for the coalition instead of individual base stations. The payment thus is 'transferable' among the members in the coalition, where the 'transferable utility' is a widely adopted concept in cooperative game theory [40]. Let C_m be the mth coalition that is formed by the base stations in $\mathcal{BS}$. Denote by u_m the payoff obtained due to serving a set of requests by the cached service instances in base stations of coalition C_m. Let $\mathcal{S}_m$ be the set of services that are cached into the base stations in coalition C_m. Let $\{\mathcal{R}\}_m$ be a set of requests that are implemented by the cached instances of services in $\mathcal{S}_m$. Denote by $u(C_m)$ the payoff obtained by coalition C_m, which can be calculated by

$$u(C_m) = \sum\nolimits_{r_j \in \{\mathcal{R}\}_m} \sum\nolimits_{i=1}^{|C_m|} y_{ji} \cdot p_j. \tag{6}$$

We adopt the *core* concept to capture the equilibrium status of a cooperative game. The core of a cooperative game with transferable utility is the set of feasible allocations that cannot be blocked by any coalition of players. This implies that core allocations are stable in the sense that, once a core allocation is achieved, no subset of players can gain by deviating from the core [3].

F. Problem Definition

Given a 5G-enabled MEC G consisting of remote data centers and base stations with limited resources in an MEC network, there are a set of services $\mathcal{S} = \{S_l \mid 1 \leq l \leq L\}$ to be cached from remote data centers to base stations, and a set of requests in R to be assigned to the cached service instances in G. *The collaborative service caching and request offloading problem* is to find the optimal number of base stations to form a collection $\mathcal{C}$ of coalitions, where each coalition works collaboratively for caching services in $\mathcal{S}$ and offloads user requests in R to the base stations with cached service instances, such that the total payoff of all formed coalitions in $\mathcal{C}$ is maximized, subject to the delay requirement of each request, the computing and bandwidth resource capacities on each base station.

IV. A Distributed Optimization Framework

In this section we first devise an optimization framework for the collaborative service caching and request offloading problem. We then show that the proposed game is stable and converges to the core of the cooperative game.

A. The Distributed Optimization Framework With Uncertain Payoff

Each base station decides in which coalitions to join, and demands a payoff for its contribution. A number of service instances of each service S_l are cached in some if not all base stations of the coalition. All the rest base stations of the coalition forward their requests to the base stations with cached service instances for processing. In other words, in each coalition, the base stations with cached service instances consume their computing resources to process the forwarded requests from the other base stations in the coalition. Such resource consumption of the base stations can be considered as a 'contribution' to the coalition. On the other hand, the base stations without cached instances 'contribute' requests to the coalition. These two types of contributions of base stations need to be rewarded; otherwise, the coalition will not be stable. Namely, each base station in the coalition demands a payoff for its contribution, which is referred to as the *expected payoff*. The base station is willing to stay in a coalition only if its expected payoff can be met by the coalition.

An important challenge of the coalition formation and payoff allocation is that each base station has uncertain information of the payoff functions of other base stations owned by other network service providers. The decision of each base station may affect the current coalition structure, thereby impacting its payoff. However, without the coalition structure, the payoff of each base station in the coalition cannot be obtained, because the payoff of a base station depends on both its contribution and the number of base stations in the coalition. Instead, to attract base stations to join in, each coalition publishes its *promised payoffs* for base stations if they choose to join in.

The basic idea behind the proposed distributed mechanism is to allow each base station to make its decision based on its expected payoff and the promised payoffs of coalitions. Recall that we aim to enable near optimal decisions based on incomplete and uncertain information. In addition, an efficient distributed solution usually takes the minimum set of information on payoff to make decisions, such that the number of interactions (messages for coordination) among base stations is minimized. This however may reduce the quality of the obtained solution compared with the mechanism with complete information. To leverage a careful trade-off of efficiency and optimality of distributed solutions, we aim to find a minimum set of information to make decisions. We observe that the expected payoff and the promised payoffs of coalitions constitute a set of information can be obtained in many scenarios. We thus define this as the minimum set of information that each base station can obtain from a collaborative service caching environment. For example, when the promised payoff of the coalition can meet the expected payoff of the base station, the base station can choose to join one of the coalitions. Likewise, each coalition may not know which base stations will join in and what are the expected payoffs of base stations, until the base stations make their joining in decisions. Due to unknown information of payoff functions, a smart method is to make sure that each coalition does not violate the expected payoffs of its base stations.

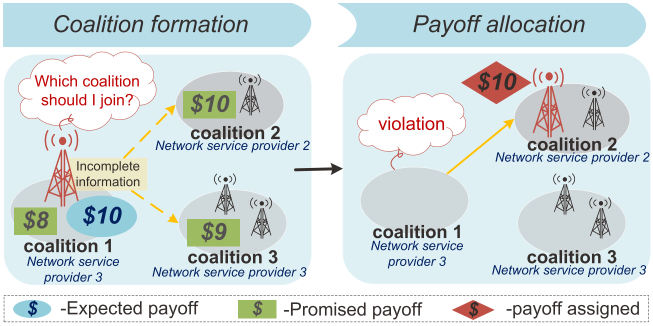


Fig. 3. A two-stage optimization framework based on incomplete information.

We adopt a two-stage optimization framework, as shown in Fig. 3. The first stage allows the base stations to form coalitions based on the payoff expectations and payoff promises of coalitions. The second stage deals with a near-optimal service caching, request association, and payoff allocation, such that the payoff expectations are met by the formed coalitions. The first and second stages are referred to as the *coalition formation procedure* and the *payoff allocation procedure*, respectively. The proposed optimization framework is shown in **Algorithm** 1, which is referred to as `Framework`.

```
Algorithm 1 Framework
Input: A set of coalitions with each coalition $C_m$ having a number of base stations in it.
Output: A caching decision for service, and payoff allocations.
Initially, multiple coalitions are formed, and each coalition contains a single base station;
Invoke algorithm PayoffAlloc to allocate the payoffs of the formed coalitions;
for each time slot $t$ do
    /*Stage 1: Coalition Formation*/
    Invoke algorithm CoalSelect for each coalition $C_m$;
    /*Stage 2: Payoff Allocation*/
    Each base station claims its expected payoff in its selected coalition;
    Invoke algorithm PayoffAlloc to allocate payoffs for time slot $t + 1$;
```

B. Distributed Coalition Formation Procedure With Incomplete Information

We consider 'myopic players', where each base station as a player seeks to maximize its payoff for the next period, conditional on the feasibility of the solution. Note that such players also have adaptive expectations. Given any coalition structure, each of such myopic base stations faces three options: (1) it can stay in its current coalition; (2) it can join in one of the other coalitions if the selected coalition can meet its expectation on payoff; and (3) it can form a singleton coalition by itself. As a myopic player, each base station chooses the coalition that promises the maximum expected payoff. If there are multiple such coalitions, the base station chooses one randomly.

Denote by u_i the payoff received by base station bs_i. The final payoff received by bs_i in C is determined by the payoff allocation strategy of coalition C. We assume that each bs_i claims its expected payoff before joining in any coalition. Clearly, the expected payoff of bs_i may depend on its contributions to the coalition (either request offloading or service caching). For example, if bs_i has a cached service instance of r_j, it expected payoff is related to its energy consumption due to processing; otherwise, it needs to forward the request to other base stations, meaning that its expected payoff is proportional to its energy consumption due to transmission. Let $u_{i,j}^{dmd,p}$ be the expected payoff if base station bs_i has a cached service instance of r_j; otherwise, denote by $u_{i,j}^{dmd,f}$ the expected pay off if bs_i forwards r_j to other base stations in its coalition. Therefore, considering its role in the coalition, base station bs_i calculates its expected payoff by

$$u_{i,j}^{dmd,p} = \beta_i \cdot e_{i,j}^{p}, \tag{7}$$

if there is one cached instance of service S_l and processes the data of request r_j in the coalition; otherwise,

$$u_{i,j}^{dmd,f} = \alpha_i \cdot e_{i,j}^{t}, \tag{8}$$

where α_i ($\alpha_i > 0$) captures bs_i's valuation on its contribution of offloading a request to the coalition, and β_i ($\beta_i > 0$) is a valuation on its contribution of caching a service instance for r_j. Such valuations of each base station are set in advance.

In a 5G-enabled MEC, due to user mobility, mobile users change their locations quite often over time. Thus, base stations have to make their decisions without foreseeing the future system dynamics. Each base station follows the best-reply rule by selecting the coalition with the maximum promised payoff. However, the promised payoff is not the received payoff, since the received payoff of each base station in a coalition can only be determined after all base stations having joined in the coalition. Hence, a coalition may not be able to meet the expected payoff of its base stations, if the coalition admits an unexpected number of base stations with high total expected payoff. As such, a base station may choose to deviate from its current coalition if its current coalition cannot meet its expected payoff u_i^{dmd}. Therefore, the coalition structure evolves over time, too. The coalition formation procedure is described in **Algorithm** 2.

Algorithm 2 `CoalSelect`

```
Input: A set of coalitions with each coalition $C_m$ having a number of base stations.
Output: The selected coalition of each base station.
Each coalition publishes its promised payoff if base stations join in the coalition;
Each base station $bs_i$ publishes its expected payoff of joining a coalition;
for each base station $bs_i$ do
    if $bs_i$ is not in any coalition then
        $bs_i$ joins the coalition with the maximum promised payoff, following the best-reply rule;
    else
        Let $C_m$ be the current coalition of $bs_i$;
        if $C_m$ cannot meet its expected payoff || the delay requirement of its requests cannot be met || its computing and bandwidth resource capacities are violated then
            Randomly choose another coalition $C_{m'}$;
        else
            Stay in its current coalition;
```

C. Near-Optimal Payoff Allocation Procedure

After each base station decides in which coalition it will join, we need to decide the payoff allocation in each coalition. To this end, we need to select a number of base stations in each coalition under the current coalition structure to cache services. The contribution of each base station in the coalition thus can be determined, and its payoff then can be allocated accordingly. The coalition structure dynamically changes until it eventually converges to such a state that no base stations deviate from their current coalitions. The payoff allocation plays a vital role in guaranteeing the existence of a core allocation. The reason is that a player will leave a coalition if its expected payoff is not satisfied.

We adopt a randomized rounding method to cache service instances and assign each request of each coalition to the cached service instances jointly. In the following we formulate the payoff allocation problem in each coalition C_m as an ILP solution. Let R_i be the requests that access the mobile edge cloud via base station bs_i. Recall that $\{\mathcal{R}\}_m$ is the set of requests of the base stations in coalition C_m. Clearly $\{\mathcal{R}\}_m = \cup_{bs_i \in C_m} R_i$. Let x_{li} be a binary variable that indicates whether service S_l of network service provider sp_l is cached in base station bs_i. Denote by y_{ji} an indicator variable that shows whether request r_j is assigned to a cached instance of S_l in base station bs_i. The problem then can be formulated as an ILP as follows.

$$\text{Maximize} \quad \sum\nolimits_{i=1}^{|C_m|} \sum\nolimits_{r_j \in \{\mathcal{R}\}_m} y_{ji} \cdot p_j, \tag{9}$$

$$\text{subject to} \quad \sum_{i=1}^{|C_m|} x_{li} = 1, \quad \forall S_l \in \mathcal{S}_m \tag{10}$$

$$y_{ji} \leq x_{li}, \quad \forall r_j \in \{\mathcal{R}\}_m \tag{11}$$

$$\sum_{j=1}^{|\{\mathcal{R}\}_m|} y_{ji} \cdot \rho_j \cdot C_{unit} \leq C(bs_i) \tag{12}$$

$$\sum_{j=1}^{|\{\mathcal{R}\}_m|} y_{ji} \cdot \rho_j \cdot B_{unit} \leq B(bs_i) \tag{13}$$

$$\sum_{i=1}^{|C_m|}\sum_{r_j\in R_i}\sum_{i'=1}^{|C_m|} y_{ji'}\cdot u_{i,j}^{dmd,f}$$

$$\sum_{j=1}^{\{\mathcal{R}\}_m}\sum_{i=1}^{|C_m|} y_{ji}\cdot u_{i,j}^{dmd,p} \le u(C_m), \qquad (14)$$

$$+\, y_{ji}\cdot(\delta_{i,j}+\gamma_{i,j}) \le d_j^{req} \qquad (15)$$

$$x_{li}, y_{ji} \in \{0,1\}, \qquad (16)$$

where Constraint (10) says that each service S_l has to be cached into a base station. Constraint (11) ensures that each request r_j can only be offloaded to a base station with an instance of its service S_l. Constraints (12) and (13) ensure that the computing and bandwidth resource capacities of each bs_i are not violated. Constraint (14) says that the expected payoff of each base station has to be met to make sure it stays in the current coalition C_m. Constraint (15) guarantees the delay requirement of each offloaded request has to be met. Constraint (16) ensures that x_{li} and y_{ji} are binary indicator variables.

We relax Constraint (16) into

$$0 \le x_{li}, y_{ji} \le 1. \qquad (17)$$

Then, the **ILP** is relaxed into an **LP** with the objective shown in (9), subject to Constraints (10), (11), (12), (13), (14), (15), and (17).

The optimal fractional solution to the **LP** can be obtained in polynomial time [2]. It however may not be a feasible solution to the original problem due to the fraction values of x_{li} and y_{ji}. To make the solution feasible, we need to round the fractional solution to an integer solution, by utilizing a randomized rounding technique. For service S_l, we use X_{li} to denote an event that S_l is cached into base station bs_i.

A natural rounding method is to treat each fractional value of x_{li} as a probability, and randomly rounds the variable with the probability. However, this means that multiple services can be assigned to a base station randomly, according to the corresponding probability. This may cause significant resource violation of each base station with high probability, if many services with non-negative values of x_{li} are assigned to bs_i. To reduce the probabilities of violating the capacities of base stations, we assign S_l to bs_i with probability $\frac{1}{2}x_{li}$. Similarly, for each request $r_j \in \{\mathcal{R}\}_m$, we use Y_{ji} to denote an event that request r_j is offloaded to an instance of its service S_l in bs_i for processing. The detailed algorithm is given in **Algorithm** 3, which is referred to as `PayoffAlloc`.

Algorithm 3 `PayoffAlloc`

```
Input: $G = (\mathcal{BS} \cup \mathcal{DC}, E)$, a coalition $C_m$ having a set of base stations.
Output: A caching decision for a service, and payoff allocations.
Relax Constraint (16) of ILP into Constraint (17) and obtain an LP;
Obtain fractional solutions $x$ and $y$ by solving the LP;
for each service $S_l$ do
    Choose a base station $bs_i$ for $S_l$ by setting $X_{li} = 1$ with probability $\frac{1}{2}x_{li}$, no base station is chosen with probability $1 - \frac{1}{2}x_{li}$;
for each request $r_j \in \{\mathcal{R}\}_m$ do
    Assign $r_j$ to base station $bs_i$ with probability $\frac{1}{2}y_{ji}$;
if all $X_{li}$ and $Y_{ji}$ define a feasible solution then
    return $X_{li}$ and $Y_{ji}$;
else
    return infeasible;
```

D. Algorithm Analysis

The rest is to analyze the performance of the proposed algorithm in the proposed optimization framework.

Lemma 1: Assuming that $C(bs_i) \ge \frac{24\ln|\mathcal{BS}|C_{unit}}{\gamma}$, the obtained solution by **Algorithm** 3 is a feasible solution with the computing capacity $C(bs_i)$ of each base station being violated with a probability of $\frac{1}{|\mathcal{BS}|^2}$, where γ is defined as the ratio of the data volume ρ_j of r_j to the minimum data volume, i.e., $\gamma = \frac{\rho_j}{\min_{r_j\in\{\mathcal{R}\}_m}\rho_j}$ and C_{unit} is the amount of computing resource of base stations used to process a unit amount of data of each request.

Proof: Clearly, each request r_j will be implemented in a single base station. In the following we show that the computing capacity of each base station is violated with a small probability. Recall that in `Algorithm` 3, service S_l is cached into base station bs_i with probability $\frac{1}{2}x_{li}$. This indicates that $X_{li} = 1$ with a probability $\frac{1}{2}x_{li}$; otherwise, $X_{li} = 0$ with probability $1 - \frac{1}{2}x_{li}$.

Notice that a request can only be assigned to a base station with a cached instance of its requested service, we then calculate the probability $Pr[Y_{ji} = 1]$ following the conditional probability, i.e.,

$$Pr[Y_{ji} \mid X_{li} = 1] = (x_{li}\cdot y_{ji})/4. \qquad (18)$$

Let Y_i be the event that the data volume of requests that are assigned to base station bs_i for processing. Clearly, $Y_i = \sum_{j=1}^{|\{\mathcal{R}\}_m|}\rho_j Y_{ji}$, where $\{\mathcal{R}\}_m$ is the set of requests that are implemented by the cached instances of services in coalition C_m. Its expectation $E(Y_i)$ is

$$E(Y_i) = \sum\nolimits_{j=1}^{|\{\mathcal{R}\}_m|}\rho_j\cdot E(Y_{ji}). \qquad (19)$$

We bound the expectation of event Y_i. Assume that a request $r_{j'}$ is offloaded to base station $bs_{i'}$. Since all events are independent, given another base station $bs_i \in C_m$, we then have

$$\begin{aligned} E(Y_i \mid Y_{j'i'} = 1) &= \sum_{j=1}^{|\{\mathcal{R}\}_m|}\rho_j\cdot E(Y_{ji}) \\ &= \sum_{j=1}^{|\{\mathcal{R}\}_m|}\rho_j\cdot(x_{li}\cdot y_{ji})/4 \qquad (20) \\ &\le \left(\frac{\gamma C(bs_i)}{4C_{unit}}\right), \qquad (21) \end{aligned}$$

where the derivation from Eq. (20) to Ineq. (21) is because the facts that each base station bs_i can maximally implement an amount $\frac{C(bs_i)}{C_{unit}}$ of data and $\gamma \geq 1$. Calculating the probability that the capacity of each base station is violated is to calculate

$$Pr[Y_i \geq C(bs_i) \mid Y_{j'i'} = 1]. \tag{22}$$

By a Chernoff bound [37] with $\mu = E(Y_i)$, we have

$$\begin{aligned} &Pr[Y_i \geq C(bs_i) \mid Y_{j'i'} = 1] \\ &\quad = Pr[Y_i \geq (1+\delta)E(Y_i) \mid Y_{j'i'} = 1] \\ &\quad = Pr[Y_i \geq 2E(Y_i) \mid Y_{j'i'} = 1], \text{assuming } \delta = 1, \\ &\quad \leq exp\big(-(E(Y_i))/3\big) \\ &\quad \leq exp\big(-(\gamma \cdot C(bs_i))/(12 \cdot C_{unit})\big) \\ &\quad \leq exp\big(-(12 \ln |\mathcal{BS}|)/6\big) = 1/|\mathcal{BS}|^2. \end{aligned} \tag{23}$$

■

Lemma 2: Assuming that $B(bs_i) \geq \frac{24 \ln |\mathcal{BS}| B_{unit}}{\gamma}$, the obtained solution by **Algorithm** 3 is a feasible solution with the bandwidth capacity $B(bs_i)$ of each base station being violated with a probability of $\frac{1}{|\mathcal{BS}|^2}$, where $\gamma = \frac{\rho_j}{\min_{r_j \in \{\mathcal{R}\}_m}\{\rho_j\}}$. The proof is similar to Lemma 1, omitted.

Lemma 3: Assuming that $\rho_{max} \cdot (\eta_{max} + \sigma_{max} \cdot h_{j,i}) \geq 24 \cdot \ln |\mathcal{BS}|$, the delay requirement of each request is violated with a small probability of $\frac{1}{|\mathcal{BS}|^2}$, where $\eta_{max} = \max_{bs_i \in C_m}\{\eta_i\}$, $\sigma_{max} = \max_{bs_i \in C_m}\{\sigma_i\}$.

Proof: Recall that Y_{ji} represents the event that request r_j is offloaded to its service S_l that is cached to base station bs_i for processing. Its probability can be calculated by Eq. (18). Its delay expectation can be calculated by

$$E(Y_{ji}) = \frac{1}{4} x_{li} \cdot y_{ji} \leq \frac{1}{4}. \tag{24}$$

Denote by Q_{ji} the delay experienced by request r_j in bs_i. Its expectation thus is

$$\begin{aligned} E[Q_{ji}] &= \rho_j \cdot (\eta_i + \sigma_i \cdot h_{j,i}) \cdot E(Y_{ji}) \\ &\leq \frac{\rho_j \cdot (\eta_{max} + \sigma_{max} \cdot h_{j,i})}{4} \\ &\leq \frac{\rho_{max} \cdot (\eta_{max} + \sigma_{max} \cdot h_{j,i})}{4}. \end{aligned} \tag{25}$$

By a Chernoff bound [37] with $\mu = E(Y_{ji})$ and $\delta = 1$, we can calculate the probability of violating the delay requirement d_j^{req} of request r_j by

$$\begin{aligned} Pr[Q_{ji} \geq d_j^{req}] &= Pr[Q_{ji} \geq 2E(Q_{ji})] \\ &\leq exp\Big(-\frac{E(Q_{ji})}{3}\Big) \\ &\leq exp\Big(-\frac{\rho_{max} \cdot (\eta_{max} + \sigma_{max} \cdot h_{j,i})}{12}\Big) \\ &\leq exp\Big(-\frac{24 \ln |\mathcal{BS}|}{12}\Big) \\ &= \frac{1}{|\mathcal{BS}|^2}. \end{aligned} \tag{26}$$

■

Lemma 4: If $(1+|\mathcal{BS}|)|R| \cdot p_{max} \geq 24 \ln |\mathcal{BS}|$, the expected payoff of each base station is violated with a probability of $\frac{1}{|\mathcal{BS}|^2}$, where p_{max} is defined as the maximum payment that is paid by requests, i.e., $p_{max} = \max_{r_j \in R}\{p_j\}$.

Proof: The expected payoff of each base station is proportional to the energy it consumed due to receiving data from UE of r_j or processing the data of r_j by caching an instance of S_l. Each implemented request has a payment p_j. Thus, showing whether the expected payoff of each base station can be met is to show that the energy cost incurred by each base station in coalition C_m does not exceed the total payment it receives.

Recall that X_{li} is the event that service S_l is cached into base station bs_i, and Y_{ji} is the event that request r_j is offloaded to bs_i. We can calculate the expected payment $E[P_m]$ received by coalition C_m by

$$\begin{aligned} E[P_m] &= \sum_{i=1}^{C_m} \sum_{j=1}^{\{\mathcal{R}\}_m} Y_{ji} \cdot p_j \\ &= \sum_{i=1}^{C_m} \sum_{j=1}^{\{\mathcal{R}\}_m} \frac{1}{4} x_{li} \cdot y_{ji} \cdot p_j \leq (|R| \cdot p_{max})/4. \end{aligned} \tag{27}$$

For each base station bs_i, the set of requests that access the 5G-enabled MEC via it is $R(bs_i)$. The expected payoff of base station bs_i due to receiving data of requests via bs_i thus is

$$\begin{aligned} E[u_i^t] &= \sum_{r_j \in R(bs_i)} \frac{1}{4} x_{li} \cdot y_{ji} \cdot \alpha_i \cdot e_{i,j}^t \\ &\leq \sum_{j=1}^{\{\mathcal{R}\}_m} \frac{1}{4} x_{li} \cdot y_{ji} \cdot \alpha_i \cdot e_{i,j}^t \\ &\leq \sum\nolimits_{j=1}^{\{\mathcal{R}\}_m} \frac{1}{4} x_{li} \cdot y_{ji} \cdot p_j, \end{aligned} \tag{28}$$

assuming that the payment p_j of request r_j can always cover the energy cost of data transmission via a base station, i.e., $p_j \geq \alpha_i \cdot e_{i,j}^t$.

Similarly, the expected payoff of bs_i for caching service instances can be calculated by

$$\begin{aligned} E[u_i^p] &= \sum\nolimits_{S_l \in \mathcal{S}_m} \frac{1}{2} \cdot x_{li} \cdot \beta_i \cdot e_{i,j}^p \\ &\leq \sum\nolimits_{S_l \in \mathcal{S}_m} \frac{1}{2} \cdot x_{li} \cdot p_j \\ &\leq \frac{|\mathcal{S}| \cdot p_{max}}{2}, \end{aligned} \tag{29}$$

assuming that the payment p_j of request r_j can always cover the energy cost of data processing of a base station, i.e., $p_j \geq \beta_i \cdot e_{i,j}^t$.

The total expected payoff of all base stations in C_m is

$$\begin{aligned} E[TD_m] &= \sum\nolimits_{bs_i \in C_m} E[u_i^t] + E[u_i^p] \\ &\leq E[P_m] + \sum\nolimits_{bs_i \in C_m} E[u_i^p] \leq (1 + |\mathcal{BS}|) \cdot E[P_m]. \end{aligned} \tag{30}$$

By a Chernoff bound [37] with $\mu = \sum_{bs_i \in C_m} E[u_i^t] + E[u_i^p]$ and $\delta = 1$, we have

$$\begin{aligned} Pr[TD_m \geq P_m] &= Pr[TD_m \geq 2E(TD_m)] \\ &\leq exp\Big(-\frac{E(TD_m)}{3}\Big) \\ &\leq exp\Big(-\frac{(1 + |\mathcal{BS}|) \cdot E[P_m]}{3}\Big) \end{aligned}$$

$$\leq exp\big(-\frac{(1+|\mathcal{BS}|)|R|\cdot p_{max}}{12}\big)$$
$$\leq exp\big(-(24\ln|\mathcal{BS}|)/12\big)=1/|\mathcal{BS}|^2. \quad (31)$$

■

Theorem 1: Given a 5G-enabled MEC network $G=(\mathcal{BS}\cup\mathcal{DC},E)$ with a set $\mathcal{BS}$ of 5G base stations and a set $\mathcal{DC}$ of data centers, and a set of R requests to be assigned to the cached instances in G, the proposed optimization framework, i.e., **Algorithm** Framework, for the collaborative service caching and request offloading problem dynamically forms coalitions and converges to a stable coalition structure with a probability of $(1-\frac{1}{|\mathcal{BS}|^2})^{|\mathcal{BS}|}$. Also, **Algorithm** Framework delivers a solution for the collaborative service caching and request offloading problem that violates the bandwidth and computing resource capacities with a very low probability of $\frac{1}{|\mathcal{BS}|^2}$, and its running time in each time slot t is $O(|R||\mathcal{BS}|+\mathcal{S}|\mathcal{BS}|)^{\omega}L)$, where $\omega\approx 2.37$ is the exponent of matrix multiplication.

Proof: In the coalition formation process, each base station seeks to deviate from its current coalition as long as its expected payoff cannot be met. Following Lemma 4, each base station has a maximum probability of $\frac{1}{|\mathcal{BS}|^2}$ to change its coalition. However, this probability is very small, and most base stations choose to stay in their coalition. Therefore, we conclude that the proposed optimization framework converges to a stable coalition structure with a minimum probability of $(1-\frac{1}{|\mathcal{BS}|^2})^{|\mathcal{BS}|}$.

So far, we assumed that each base station is payoff-sensitive, that is, it will deviate from its current coalition when its expected payoff is not satisfied. The proposed optimization framework can also deal with base stations with delay-sensitive requests. For example, if a base station has a great portion of delay-sensitive requests, it may choose the other coalitions if the current one cannot meet the delays of its requests. However, as shown by Lemma 3, the probability of by doing so is low. Therefore, no matter which base stations are delay- or payoff-sensitive, the proposed optimization framework converges to a stable coalition with a high probability.

The solution feasibility of the proposed algorithm Framework is shown in Lemmas 2 and 3.

We then analyze the running time of algorithm Framework in each time slot t. The most time consuming part of the algorithm lies in solving the LP. Using Lee and Song's algorithm [32], this procedure can take $O(n^{\omega}L)$ where $n=|R||\mathcal{BS}|+\mathcal{S}|\mathcal{BS}|$ is the number of variables, $\omega\approx 2.37$ is the exponent of matrix multiplication, and L is the input bits. The time complexity of the algorithm in each time slot t thus is $O(|R||\mathcal{BS}|+\mathcal{S}|\mathcal{BS}|)^{\omega}L)$. ■

V. Simulations

In this section we evaluate the performance of the proposed optimization framework against existing studies by extensive simulations.

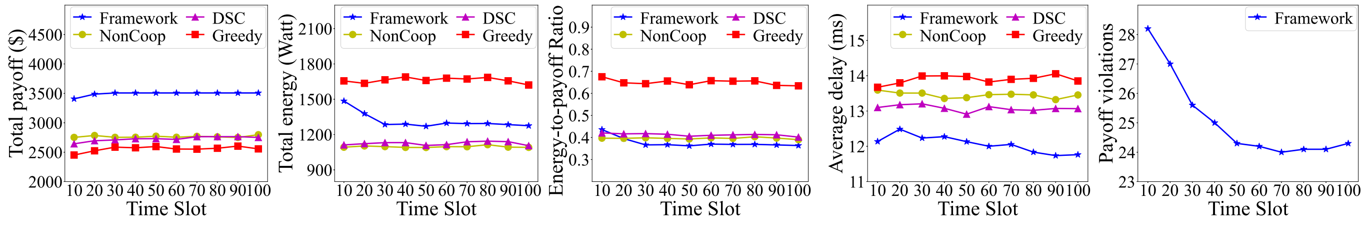
A. Parameter Settings

We consider a 5G-enabled MEC network with 5 to 10 remote data centers and 100 base stations. Specifically, following existing studies [13], we adopt the Poisson point process [7] for the deployment of base stations. The network that is used to interconnect the base stations in the 5G-enabled MEC is generated using GT-ITM [17], with each pair of base stations having a probability of 0.1 of being connected. Each base station has a computing capacity in the range 8,000 to 16,000 MHz [47]. The bandwidth capacity of each base station varies between 100 $Mbps$ and 1,000 $Mbps$ [47]. The energy consumed due to transmitting a unit amount of data to base station is set within [0.14, 0.32] Watt [8]. The maximum GPU power of each base station is randomly withdrawn from [0.5, 0.9] Watt and the idle GPU power is e^{-5} Watt [8]. The amounts of data ρ_j that need to be processed by each request is set within [10, 50] Mega Bytes (MB) [8]. The traffic of each base station is mainly due to the transmission of data from the requests that are registered to it. Considering that each user send its request to the closest base station, each request has a location that is randomly generated within the area of the network. The time η_i each base station bs_i takes on processing unit amount (one MB) of data is varied from ranges [0.1, 0.5] milliseconds, and the time σ_j a link takes on transmitting a unit amount (one MB) of data of r_j along a link are varied from range [0.01, 0.05] milliseconds [42]. The delay requirement of each user of base station is a random value between 10 and 50 milliseconds [18]. The payment that each UE pays for offloading its request to a base station for processing is set within [\$50, \$100], which depends on the amount of data of each UE request. Unless otherwise specified, we will adopt these default settings in our experiments. Each value in the figures is the mean of the results by applying each mentioned algorithm on 80 different topologies of the 5G-enabled MEC with the same network size.

We evaluate the proposed algorithm Framework with the following benchmarks: (1) a non-cooperative mechanism NonCoop: each base station caches a set of services serving the maximum number of requests; (2) a greedy approach Greedy: each base station greedily selects a service and its requests that can maximize its own payoff; (3) a dynamic service caching algorithm DSC in [44]: the algorithm assigns each request a priority according to the number of base stations in its transmission range, and DSC schedules the request with the highest priority first.

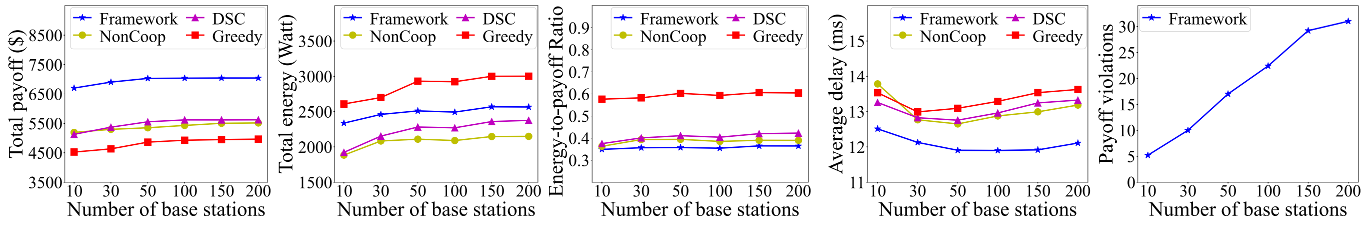
B. Performance Evaluation

We first evaluate the performance of algorithms Framework, NonCoop, DSC, and Greedy in terms of the total payoff of base stations, total amount of energy consumed by base stations, energy-to-payoff ratio, average delay of a request, and number of payoff violations in a period of 100 time slots. There are 100 base stations, 50 requests, 50 services. From Fig. 4 (a), we can see that algorithm Framework has at least 36%, 28%, and 26% higher payoffs than algorithms Greedy, DSC, and NonCoop, respectively. In addition, as depicted in Fig. 4 (b), the total energy obtained by algorithm Framework decreases as the time slot increases, but those of algorithms Greedy, DSC,



(a) Total payoff. (b) Total energy. (c) Energy-to-payoff ratio. (d) Average delay. (e) The number of violations.

Fig. 4. The performance of algorithms Framework, NonCoop, DSC, and Greedy in a period of 100 time slots.



(a) Total payoff. (b) Total energy. (c) Energy-to-payoff ratio. (d) Average delay. (e) The number of violations.

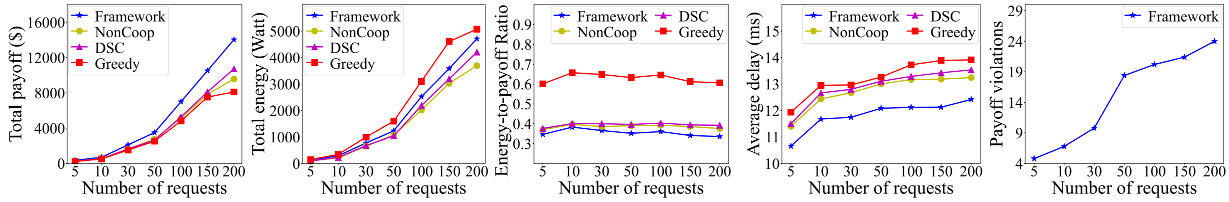
Fig. 5. The impact of the number of base stations on the performance of algorithms Framework, NonCoop, DSC, and Greedy.

and NonCoop do not change much. This is due to the fact that algorithm Framework adopts a collaborative service caching and request offloading, by allowing base stations in each coalition to share the workload of requests. This saves more computing resource to cache more service instances. From Fig. 4 (c), we can see that the solution delivered by algorithm Framework has the lowest energy-to-payoff ratio. Framework thus is the most cost-efficient algorithm with the least energy consumption. As shown in Fig. 4 (d), the solution of Framework has the lowest delay of each admitted request, because it carefully deals with delay requirements of requests in the coalition selection procedure. Also, from Fig. 4 (e), we can see that the payoff violation in each coalition gradually decreases and then tends to converge. This verifies that the proposed algorithm is stable that each base station finally chooses to stay in its current coalition.

C. Impact of Different Parameters

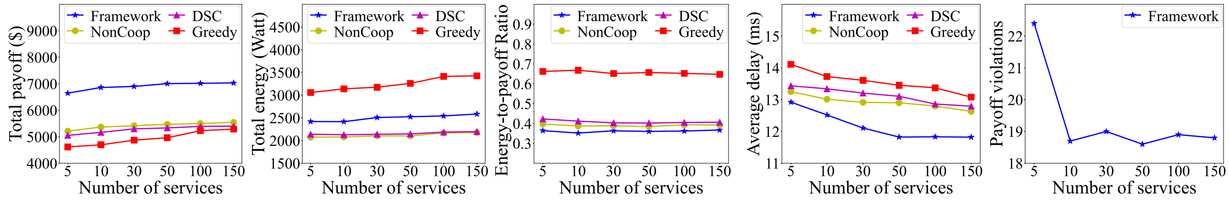
We then study the impact of the number of base stations on the performance of algorithms Framework, NonCoop, DSC, and Greedy, by varying the number of base stations from 10 to 200 while the number of services are set at 50 and the number of requests are set at 100. From Fig. 5, we can see that algorithm Framework consistently delivers the highest payoff, lowest energy-to-payoff ratios and average delays compared with NonCoop, DSC, and Greedy. We can also see that with the growth of the number of base stations, the total payoff of algorithm Framework keeps increasing. This is because each base station has more choices of coalition to obtain a higher payoff. As depicted in Figures 5 (b) and (c), the total energy consumption for implementing user requests increases with the growth of the number of base stations. However, the energy-to-payoff ratios remain basically unchanged with the growth of the number of base stations. The rationale behind is that the algorithms can always choose better cost-efficient base stations to cache services. As shown in Fig. 5 (d), the average delay decreases first, and then increases with the growth of the number of base stations. The reason is that at the very beginning, a small number of base stations with limited resources are usually insufficient to cache most services and process a large number of requests, resulting in a high average latency of requests. However, when the number of base stations is very large, each base station randomly selects a coalition in the coalition selection procedure, and a higher number of base stations means that base stations with higher delays can be selected with a higher probability. Similarly, the increasing of the number of base stations also increases the possibility of payoff violations, which can be seen in Fig. 5 (e).

We now investigate the impact of the number $|R|$ of requests on the system performance of algorithms Framework, NonCoop, Greedy, and DSC, by varying $|R|$ from 5 to 200 while fixing the number of services at 50. From Fig. 6 (a) and (b), it can be seen that the total payoff and energy consumption increase with the growth of $|R|$. The reason is that the accumulative payment of admitted requests is increasing with the increase on the number of offloaded requests, and the total payoff of all base stations keeps increases accordingly. Also, due the same reason, similar trends can be found on the total energy consumed by all base stations in request processing and transmitting. Furthermore, as illustrated in Fig. 6 (c), the energy-to-payoff ratio obtained by algorithm Framework is still the lowest among the four algorithms. From Fig. 6 (d), we can see that the average delay experienced by each request is increasing with the growth of $|R|$. The rationale behind is that each base station becomes saturated and the requests of its registered UEs have a higher probability of being forwarded to other base stations in the coalition. As expected, we can see from Fig. 6 (e) that the number of payoff violations increases as $|R|$ grows.



(a) Total payoff. (b) Total energy. (c) Energy-to-payoff ratio. (d) Average delay. (e) The number of violations.

Fig. 6. The impact of the number $|R|$ of requests on the performance of algorithms `Framework`, `NonCoop`, `DSC`, and `Greedy`.



(a) Total payoff. (b) Total energy. (c) Energy-to-payoff ratio. (d) Average delay. (e) The number of violations.

Fig. 7. The impact of the number $|S|$ of services on the performance of algorithms `Framwork`, `NonCoop`, `DSC`, and `Greedy`.

We finally evaluate the impact of the number $|S|$ of services on the performance of algorithms `Framework`, `NonCoop`, `Greedy`, and `DSC`, by varying $|S|$ from 5 to 150 while fixing the number of requests at 100. The results are shown in Fig. 7, from which we can see that the total payoff of all base stations is increasing as the number of services increases. The reason is that with more services being allowed to cache, the possibility of coalition formation increases. A higher possibility of coalition usually improves the resource utilization of base stations, so that more requests can be admitted, thereby increasing the total payoff and total energy of all base stations. The above reason can also be evidenced in Fig. 7 (a) and Fig. 7 (b). Also, the formed coalitions greatly reduce the high processing and transmission delay of requests, as shown in Fig. 7 (d). From Fig. 7 (e), it can be seen that the payoff violations are gradually decreasing and tending to stabilize, which indicate that the stable coalitions are gradually formed with the growth of $|S|$.

VI. Conclusion

In this paper, we studied the collaborative service caching and request offloading problem in a 5G-enabled MEC, such that the total payoff of all base stations is maximized through optimizing the energy consumption of base stations. We first proposed a two-stage optimization framework that consists of a novel distributed coalition formation procedure with uncertain payoffs of the network service providers, and a near-optimal payoff allocation method. Specifically, in the first stage of the framework, we devised a distributed coalition formation algorithm by adopting a best-reply rule, assuming that each base station makes its decisions based on its payoff expectation and the payoff promise of coalitions. In the second stage of the framework, we proposed an exact solution for the problem by formulating the coalition as an ILP, and a randomized algorithm with a provable approximation ratio then is derived from the ILP formulation through linear relaxation. We finally evaluated the performance of the proposed optimization framework by simulations. Simulation results show that the performance of the proposed algorithms outperform their counterparts by achieving at least 30% higher payoffs and 20% lower energy consumption of base stations.

Acknowledgment

The authors appreciate the anonymous referees and the associate editor for their constructive comments and valuable suggestions, which helped them, improve the quality and presentation of the paper greatly.

References

[1] *Amazon Pricing*. Accessed: Nov. 2021. [Online]. Available: https://aws.amazon.com/emr/pricing/

[2] A. Fréville, “The multidimensional 0–1 knapsack problem: An overview,” *Eur. J. Oper. Res.*, vol. 155, no. 1, pp. 1–21, May 2004.

[3] T. Arnold and U. Schwalbe, “Dynamic coalition formation and the core,” *J. Econ. Behav. Org.*, vol. 49, no. 3, pp. 363–380, Nov. 2002.

[4] H. A. Alameddine, S. Sharafeddine, S. Sebbah, S. Ayoubi, and C. Assi, “Dynamic task offloading and scheduling for low-latency IoT services in multi-access edge computing,” *IEEE J. Sel. Areas Commun.*, vol. 37, no. 3, pp. 668–682, Mar. 2019.

[5] W. C. Ao and K. Psounis, “Distributed caching and small cell cooperation for fast content delivery,” in *Proc. 16th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, Jun. 2015, pp. 127–136.

[6] R. E. Bailey, J. J. Arthur III, and S. P. Williams, “Latency requirements for head-worn display S/EVS applications,” *Proc. SPIE*, vol. 5424, pp. 98–109, Aug. 2004.

[7] F. Baccelli and B. Błaszczyszyn, “Stochastic geometry and wireless networks: Applications,” *Found. Trends Netw.*, Vol. 4, nos. 1–2, pp. 1–312, 2010.

[8] S. Bi, L. Huang, and Y.-J.-A. Zhang, “Joint optimization of service caching placement and computation offloading in mobile edge computing systems,” *IEEE Trans. Wireless Commun.*, vol. 19, no. 7, pp. 4947–4963, Jul. 2020.

[9] X. Cao, G. Tang, D. Guo, Y. Li, and W. Zhang, “Edge federation: Towards an integrated service provisioning model,” *IEEE/ACM Trans. Netw.*, vol. 28, no. 3, pp. 1116–1129, Jun. 2020.

[10] M. Chen and Y. Hao, “Task offloading for mobile edge computing in software defined ultra-dense network,” *IEEE J. Sel. Areas Commun.*, vol. 36, no. 3, pp. 587–597, Mar. 2018.

[11] L. Chen, J. Xu, and S. Zhou, “Computation peer offloading in mobile edge computing with energy budgets,” in *Proc. GLOBECOM IEEE Global Commun. Conf.*, Dec. 2017, pp. 1–6.

[12] Z. Chang, K. Zhu, Z. Zhou, and T. Ristaniemi, “Service provisioning with multiple service providers in 5G ultra-dense small cell networks,” in *Proc. IEEE 26th Annu. Int. Symp. Pers., Indoor, Mobile Radio Commun. (PIMRC)*, Aug. 2015, pp. .

[13] L. Chen, C. Shen, P. Zhou, and J. Xu, "Collaborative service placement for edge computing in dense small cell networks," *IEEE Trans. Mobile Comput.*, vol. 20, no. 2, pp. 377–390, Feb. 2021, doi: 10.1109/TMC.2019.2945956.

[14] V. Farhadi *et al.*, "Service placement and request scheduling for data-intensive applications in edge clouds," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, Apr. 2019, pp. 1279–1287.

[15] H. Guo, J. Liu, J. Ren, and Y. Zhang, "Intelligent task offloading in vehicular edge computing networks," *IEEE Wireless Commun.*, vol. 27, no. 4, pp. 126–132, Aug. 2020, doi: 10.1109/MWC.001.1900489.

[16] B. Gao, Z. Zhou, F. Liu, and F. Xu, "Winning at the starting line: Joint network selection and service placement for mobile edge computing," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, Apr. 2019, pp. 1459–1467.

[17] *GT-ITM*. Accessed: Nov. 2021. [Online]. Available: http://www.cc.gatech.edu/projects/gtitm/

[18] R. Gupta, S. Tanwar, S. Tyagi, and N. Kumar, "Tactile internet and its applications in 5G era: A comprehensive review," *Int. J. Commun. Syst.*, vol. 32, no. 14, Sep. 2019, Art. no. e3981.

[19] P. Bodik, A. Fox, M. J. Franklin, M. I. Jordan, and D. A. Patterson, "Characterizing, modeling, and generating workload spikes for stateful services," in *Proc. 1st ACM Symp. Cloud Comput. (SoCC)*, 2010, pp. 241–252.

[20] *5G Energy Efficiency Explained*. Accessed: Nov. 2021. [Online]. Available: https://www.a10networks.com/blog/5g-energy-efficiency-explained/

[21] C.-K. Huang, S.-H. Shen, C.-Y. Huang, T.-L. Chin, and C.-A. Shen, "S-cache: Toward an low latency service caching for edge clouds," in *Proc. ACM MobiHoc Workshop Pervas. Syst. IoT Era (PERSIST-IoT)*, 2019, pp. 49–54.

[22] D. Huang, P. Wang, and D. Niyato, "A dynamic offloading algorithm for mobile computing," *IEEE Trans. Wireless Commun.*, vol. 11, no. 6, pp. 1991–1995, Jun. 2012.

[23] T. He, H. Khamfroush, S. Wang, T. La Porta, and S. Stein, "It's hard to share: Joint service placement and request scheduling in edge clouds with sharable and non-sharable resources," in *Proc. IEEE 38th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2018, pp. 365–375.

[24] Y. Huang, Y. Lu, F. Wang, X. Fan, J. Liu, and V. C. M. Leung, "An edge computing framework for real-time monitoring in smart grid," in *Proc. IEEE Int. Conf. Ind. Internet (ICII)*, Oct. 2018, pp. 99–108.

[25] S. Hu and G. Li, "Dynamic request scheduling optimization in mobile edge computing for IoT applications," *IEEE Internet Things J.*, vol. 7, no. 2, pp. 1426–1437, Feb. 2020.

[26] M. Huang, W. Liang, X. Shen, Y. Ma, and H. Kan, "Reliability-aware virtualized network function services provisioning in mobile edge computing," *IEEE Trans. Mobile Comput.*, vol. 19, no. 11, pp. 2699–2713, Nov. 2020.

[27] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing—A key technology towards 5G," *ETSI White Paper*, vol. 11, no. 11, pp. 1–16, 2015.

[28] S. Hong and H. Kim, "An integrated GPU power and performance model," *ACM SIGARCH Comput. Archit. News*, vol. 38, no. 3, pp. 280–289, 2010.

[29] C. Luo and R. Suda, "A performance and energy consumption analytical model for GPU," in *Proc. IEEE 9th Int. Conf. Dependable, Autonomic Secure Comput.*, Dec. 2011, pp. 658–665.

[30] N. C. Luong, P. Wang, D. Niyato, Y. Liang, Z. Han, and F. Hou, "Applications of economic and pricing models for resource management in 5G wireless networks: A survey," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3298–3339, 4th Quart., 2019.

[31] Y. Liang, J. Ge, S. Zhang, J. Wu, Z. Tang, and B. Luo, "A utility-based optimization framework for edge service entity caching," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 11, pp. 2384–2395, Nov. 2019.

[32] M. B. Cohen, Y. T. Lee, and Z. Song, "Solving linear programs in the current matrix multiplication time," in *Proc. 51st Annu. ACM SIGACT Symp. Theory Comput.*, Jun. 2019, pp. 938–942.

[33] M. Mukherjee, V. Kumar, A. Lat, M. Guo, R. Matam, and Y. Lv, "Distributed deep learning-based task offloading for UAV-enabled mobile edge computing," in *Proc. IEEE INFOCOM Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Jul. 2020, pp. 1208–1212.

[34] H. Ma, Z. Zhou, and X. Chen, "Leveraging the power of prediction: Predictive service placement for latency-sensitive mobile edge computing," *IEEE Trans. Wireless Commun.*, vol. 19, no. 10, pp. 6454–6468, Oct. 2020.

[35] X. Ma, A. Zhou, S. Zhang, and S. Wang, "Cooperative service caching and workload scheduling in mobile edge computing," 2020, *arXiv:2002.01358*.

[36] S. Misra and N. Saha, "Detour: Dynamic task offloading in software-defined fog for IoT applications," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 5, pp. 1159–1166, May 2019.

[37] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 2005.

[38] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 72–79, Sep. 2016.

[39] H. Pang, J. Liu, X. Fan, and L. Sun, "Toward smart and cooperative edge caching for 5G networks: A deep learning based approach," in *Proc. IEEE/ACM 26th Int. Symp. Qual. Service (IWQoS)*, Jun. 2018, pp. .

[40] H. Peters, "Cooperative games with transferable utility," in *Game Theory*. Berlin, Germany: Springer, 2015, pp. 151–169.

[41] K. Poularakis, J. Llorca, A. M. Tulino, I. Taylor, and L. Tassiulas, "Joint service placement and request routing in multi-cell mobile edge computing networks," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, Apr. 2019, pp. 10–18.

[42] S. Wang, Y. Guo, N. Zhang, P. Yang, A. Zhou, and X. Shen, "Delay-aware microservice coordination in mobile edge computing: A reinforcement learning approach," *IEEE Trans. Mobile Comput.*, vol. 20, no. 3, pp. 939–951, Mar. 2021.

[43] J. Wang, J. Hu, G. Min, A. Y. Zomaya, and N. Georgalas, "Fast adaptive task offloading in edge computing based on meta reinforcement learning," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 1, pp. 242–253, Jan. 2021.

[44] Q. Xie, Q. Wang, N. Yu, H. Huang, and X. Jia, "Dynamic service caching in mobile edge networks," in *Proc. IEEE 15th Int. Conf. Mobile Ad Hoc Sensor Syst. (MASS)*, Oct. 2018, pp. 73–79.

[45] Z. Xu *et al.*, "Energy-aware inference offloading for DNN-driven applications in mobile edge clouds," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 4, pp. 799–814, Apr. 2021.

[46] J. Xu, L. Chen, and P. Zhou, "Joint service caching and task offloading for mobile edge computing in dense networks," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, Apr. 2018, pp. 207–215.

[47] Z. Xu, L. Zhou, S. Chi-Kin Chau, W. Liang, Q. Xia, and P. Zhou, "Collaborate or separate? Distributed service caching in mobile edge clouds," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, Jul. 2020, pp. 2066–2075.

[48] Z. Xu *et al.*, "To cache or not to cache: Stable service caching in mobile edge-clouds of a service market," in *Proc. IEEE 40th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Nov. 2020, pp. 421–431.

[49] Z. Xu *et al.*, "Learning for exception: Dynamic service caching in 5G-enabled MECs with bursty user demands," in *Proc. IEEE 40th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Nov. 2020, pp. 1079–1089.

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

[51] Q. Ye *et al.*, "End-to-end delay modeling for embedded VNF chains in 5G core networks," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 692–704, Feb. 2018.

[52] W. Zhang, J. Xiong, L. Gui, B. Liu, M. Qiu, and Z. Shi, "Distributed caching mechanism for popular services distribution in converged overlay networks," *IEEE Trans. Broadcast.*, vol. 66, no. 1, pp. 66–77, Mar. 2020.

[53] G. Zhao, H. Xu, Y. Zhao, C. Qiao, and L. Huang, "Offloading dependent tasks in mobile edge computing with service caching," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, Jul. 2020, pp. 1997–2006.

[54] L. Zhang, R. Chai, T. Yang, and Q. Chen, "Min-max worst-case design for computation offloading in multi-user MEC system," in *Proc. IEEE INFOCOM Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Jul. 2020, pp. 1075–1080.

[55] B. Zhou, A. V. Dastjerdi, R. N. Calheiros, and R. Buyya, "An online algorithm for task offloading in heterogeneous mobile clouds," *ACM Trans. Internet Technol.*, vol. 18, no. 2, pp. 1–25, Mar. 2018.

[56] N. Zhang, S. Guo, Y. Dong, and D. Liu, "Joint task offloading and data caching in mobile edge computing networks," *Comput. Netw.*, vol. 182, Dec. 2020, Art. no. 107446.

[57] T. Zhu, J. Li, Z. Cai, Y. Li, and H. Gao, "Computation scheduling for wireless powered mobile edge computing networks," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, Jul. 2020, pp. 596–605.

[58] *5G Construction: Energy and Emissions*. Accessed: Nov. 2021. [Online]. Available: https://www-file.huawei.com/-/media/corp2020/pdf/publications/communicate/comm-89-en2.pdf

Zichuan Xu (Member, IEEE) received the B.Sc. and M.E. degrees in computer science from the Dalian University of Technology, China, in 2008 and 2011, respectively, and the Ph.D. degree from The Australian National University in 2016. From 2016 to 2017, he was a Research Associate at the Department of Electronic and Electrical Engineering, University College London, U.K. He is currently an Associate Professor with the School of Software, Dalian University of Technology, where he is also a Xinghai Scholar. His research interests include mobile edge computing, cloud computing, network function virtualization, software-defined networking, the Internet of Things, algorithmic game theory, and optimization problems.

Lizhen Zhou (Student Member, IEEE) received the B.E. degree in computer science and technology from Tianjin University, China, in 2019. She is currently pursuing the Ph.D. degree in software engineering with the Dalian University of Technology. Her research interests include mobile edge computing, the Internet of Things, and network function virtualization.

Haipeng Dai (Member, IEEE) received the B.S. degree from the Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai, China, in 2010, and the Ph.D. degree from the Department of Computer Science and Technology, Nanjing University, Nanjing, China, in 2014. He is an Associate Professor with the Department of Computer Science and Technology, Nanjing University. His research papers have been published in many prestigious conferences and journals, such as ACM MobiSys, ACM MobiHoc, ACM VLDB, ACM SIGMETRICS, ACM UbiComp, IEEE INFOCOM, IEEE ICDCS, IEEE ICNP, IEEE SECON, IEEE IPSN, IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, IEEE/ACM TRANSACTIONS ON NETWORKING, IEEE TRANSACTIONS ON MOBILE COMPUTING, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, and *ACM Transactions on Sensor Networks*. His research interests are mainly in the areas of the Internet of Things, mobile computing, and data mining. He is an ACM Member. He serves/served as the Poster Chair for the IEEE ICNP'14, the Track Chair for the ICCCN'19, and a TPC Member for the ACM MobiHoc'20, IEEE INFOCOM'20, IEEE ICDCS'20, IEEE ICNP'14, IEEE IWQoS'19, IEEE IPDPS'20, IEEE MASS'18–19, IEEE ICC'14–18, IEEE ICCCN'15–18, and IEEE Globecom'14–18. He received the Best Paper Award from IEEE ICNP'15, the Best Paper Award Runner-Up from IEEE SECON'18, and the Best Paper Award Candidate from IEEE INFOCOM'17.

Weifa Liang (Senior Member, IEEE) received the B.Sc. degree from Wuhan University, China, in 1984, the M.E. degree from the University of Science and Technology of China in 1989, and the Ph.D. degree from The Australian National University in 1998, all in computer science. He is currently a Professor with the Department of Computer Science, City University of Hong Kong, Hong Kong, prior to that position, he was a Professor with The Australian National University. His research interests include design and analysis of energy efficient routing protocols for wireless *ad-hoc* and sensor networks, the Internet of Things, mobile edge computing (MEC), network function virtualization (NFV), software-defined networking (SDN), design and analysis of parallel and distributed algorithms, approximation algorithms, combinatorial optimization, and graph theory. He currently serves as an Associate Editor for the IEEE TRANSACTIONS ON COMMUNICATIONS.

Wanlei Zhou (Senior Member, IEEE) received the B.E. and M.E. degrees in computer science and engineering from the Harbin Institute of Technology, Harbin, China, in 1982 and 1984, respectively, the Ph.D. degree in computer science and engineering from The Australian National University in 1991, and the D.Sc. degree (a higher doctorate degree) from Deakin University in 2002. He is currently the Vice Rector (Academic Affairs) and the Dean of the Institute of Data Science, City University of Macau, Macau, China. Before joining the City University of Macau, he held various positions including the Head of the School of Computer Science, University of Technology Sydney, Australia, and the Alfred Deakin Professor, the Chair of Information Technology, the Associate Dean, and the Head of the School of Information Technology, Deakin University, Australia. He was also a Lecturer with the University of Electronic Science and Technology of China; a System Programmer with HP, MA, USA; a Lecturer with Monash University, Melbourne, Australia; and with the National University of Singapore. He has authored or coauthored more than 400 papers in refereed international journals and refereed international conferences proceedings, including many articles in IEEE TRANSACTIONS and journals. His main research interests include security, privacy, and distributed computing.

Pan Zhou (Member, IEEE) received the B.S. degree from the Advanced Class of the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2006, the M.S. degree from the Department of Electronics and Information Engineering, HUST, in 2008, and the Ph.D. degree from the School of Electrical and Computer Engineering, Georgia Institute of Technology (Georgia Tech), Atlanta, USA, in 2011. He was a Senior Technical Member at Oracle Inc., USA, from 2011 to 2013. He is currently an Associate Professor with the School of Electronic Information and Communications, HUST. His current research interests include big data analytics and machine learning, security and privacy, and information networks.

Wenzheng Xu (Member, IEEE) received the B.Sc., M.E., and Ph.D. degrees in computer science from Sun Yat-sen University, Guangzhou, China, in 2008, 2010, and 2015, respectively. He was a Visitor at both The Australian National University and The Chinese University of Hong Kong. He is currently an Associate Professor with Sichuan University. His research interests include wireless *ad-hoc* and sensor networks, mobile computing, approximation algorithms, combinatorial optimization, online social networks, and graph theory.

Guowei Wu (Member, IEEE) received the Ph.D. degree from Harbin Engineering University, China, in 2003. He is currently a Professor with the School of Software, Dalian University of Technology (DUT), China. He has published more than 100 journals and conference papers. His research interests include embedded real-time systems, cyber-physical systems, and smart edge computing.