A Joint Learning and Game-Theoretic Approach to Multi-Dimensional Resource Management in Fog Radio Access Networks

Yaohua Sun, Siqi Chen, Zeyu Wang, and Shiwen Mao, *Fellow, IEEE*

***Abstract*—Fog radio access networks (F-RANs) have been regarded as a promising paradigm to support latency-sensitive and computation-intensive services by leveraging computing and caching capabilities of fog access points (F-APs). To execute offloaded computation tasks, it is essential to pre-store necessary programs and databases at F-APs, referred as service caching. However, due to system dynamics and the coupling of decisions, cache, radio, and computation resource management at F-RANs is a challenging problem. In this paper, a joint multi-dimensional resource optimization problem is formulated, aiming at minimizing the weighted sum of expected latency cost and caching cost, which is featured by a two-timescale structure. For radio and computation resource allocation on a small timescale, a coalitional game based approach is proposed under given caching decisions. On a larger timescale, services are cached at each F-AP using a multi-agent reinforcement learning algorithm. The advantage of the proposed algorithms is that they can implicitly take the impact of small timescale resource allocation into account while adapting to the long-term statistics of channel coefficients and user service requests. We analyze the proposed algorithms with respect to their convergence, optimality, and complexity, and validate their performance with extensive simulations, where superior performance is observed over several baseline schemes.**

***Index Terms*—Fog radio access networks, multi-dimensional resource management, machine learning, multi-agent reinforcement learning (MARL).**

I. Introduction

EMERGING services, such as virtual reality/augmented reality (VR/AR) and online gaming, are latency-sensitive and computation-demanding, which put heavy burdens on the processors and batteries of user devices [1], [2], [3]. Although cloud computing provides a solution that overcomes this issue, offloading a large number of computation tasks to the cloud unavoidably increases the data traffic on the backhaul/fronthaul as well as user latency [4], [5], [6]. Fortunately, as a promising paradigm, fog radio access networks (F-RANs) can well harmonize fog computing and cloud computing, to achieve lower latency and transmission overhead by allowing user devices to offload computation tasks to fog access points (F-APs) [7], [8].

A. Computation Offloading Optimization in F-RANs

Recently, much attention has been paid to computation offloading optimization in F-RANs. In [9], it was assumed that a task node can offload some tasks to the nearby fog nodes, and a bandit learning based online node selection strategy was proposed for a typical task node to minimize a long-term cost induced by communication and computation. The authors in [10] jointly optimized the task division of a single mobile device among multiple fog nodes and the device's local CPU frequency, aiming at lowering the total execution latency and device energy consumption of all tasks. Under a scenario with multiple mobile users and one fog node, task distribution and resource allocation were formulated as a mixed-integer non-linear programming in [11], which was then solved by a low-complexity suboptimal algorithm. In [12], given a three-layer hierarchical computing architecture, a user task can be offloaded to servers at radio units or distributed units or central units, and the joint optimization of receive beamforming, task distribution, computing resource allocation at each server and the transmission bandwidth division of fronthaul was tackled using variable relaxation and substitution. In order to further reduce the offloading latency and improve spectrum efficiency, several prior works considered power-domain non-orthogonal multiple access (NOMA). In [13], subcarrier sharing among multiple users in a NOMA-enabled F-RAN was modeled as a coalition formation game, where users in the same coalition share the same subcarrier for task data uploading. In [14], a joint user offloading decision, fog node association, and resource allocation problem was investigated to minimize a weighted system cost, and alternating direction method of multipliers based optimization was adopted to solve the problem. By utilizing genetic algorithm, authors in [15] proposed a heuristic approach to the joint optimization of computation offloading decisions and resource allocation for

Manuscript received 28 November 2021; revised 10 May 2022 and 22 July 2022; accepted 3 October 2022. Date of publication 12 October 2022; date of current version 13 February 2023. The work of Shiwen Mao was supported by the NSF under Grants ECCS-1923717 and CNS-2107190. This work was supported in part by the National Natural Science Foundation of China under Grant 62001053, in part by the National Key R&D Program of China under Grant 2020YFB1806703 and in part by China Institute of Communications through Young Elite Scientist Sponsorship Program. The review of this article was coordinated by Dr. Beatriz Lorenzo. *(Corresponding author: Siqi Chen.)*

Yaohua Sun and Zeyu Wang are with the State Key Laboratory of Networking and Switching Technology (SKL-NST), Beijing University of Posts and Telecommunications, Beijing 510300, China (e-mail: sunyaohua@bupt.edu.cn; zeyu.wang@bupt.edu.cn).

Siqi Chen is with the China Mobile Research Institute, Beijing 100032, China (e-mail: chensiqiwl@chinamobile.com).

Shiwen Mao is with the Department of Electrical and Computer Engineering, Auburn University, Auburn, AL 36849-5201 USA (e-mail: smao@ieee.org).

Digital Object Identifier 10.1109/TVT.2022.3214075

0018-9545 © 2022 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Auburn University. Downloaded on August 24,2023 at 05:49:03 UTC from IEEE Xplore. Restrictions apply.

a scenario with multiple UEs and fog nodes, which can lead to a stable convergence solution.

Although the above works achieve good performance, they implicitly assume that the fog nodes or F-APs can execute all kinds of computation tasks offloaded to them. However, executing the offloaded computation tasks not only requires the input data from user devices but also dedicated programs and databases, which should be pre-stored at F-APs. Taking AR service for example, programs for 3D rendering shall be cached at F-APs and this kind of operation is termed *service caching*. Actually, the service caching problem has been considered in several prior works in computation offloading scenarios. In [16], combinatorial contextual bandit learning was utilized to place services effectively at the edge under spatial-temporal dynamics. In [17], genetic algorithm was adopted to solve the problem of service caching and task offloading, aiming at balancing the user perception of service latency and the cost paid by users due to the use of computing resource. In [18], the authors focused on a cache-assisted single-user mobile edge computing system and proposed an alternating minimization based approach to jointly optimize service placement, computation offloading decisions, CPU processing frequency, and transmit power of mobile users. In [19], a service caching and task offloading problem was presented for a dense F-RAN, where an online optimization algorithm was developed by integrating Lyapunov optimization with Gibbs sampling.

B. Other Related Works

In addition to resource allocation in computation offloading scenarios, there have been some studies focusing on traditional radio resource allocation or multi-media content caching or a joint optimization of them in F-RANs.

In [20], aiming at maximizing downlink throughput, authors studied dynamic radio resource allocation, and the primal problem was transformed into an F-AP power allocation problem and a subchannel assignment problem among F-UEs that was solved using matching theory. From the perspective of interference management in a downlink F-RAN, authors in [21] addressed the assignment of resource blocks (RBs) to each user, power allocation among RBs and the power split levels of users using the same RB. Faced with the non-convexity of the resource allocation problem, a decoupled solution was proposed, which was based on Hungarian algorithm and alternating direction method of multipliers. When F-APs equip multi-antennas, beamforming vector design should be considered in radio resource allocation. In [22], a fractional programming and weighted minimum mean square error based approach was developed to optimize downlink beamforming vectors of F-APs, where throughput, energy consumption and content caching cost were comprehensively considered. In [23], authors focused on cost efficiency defined as the ratio of content delivery throughput and caching cost, and path-following algorithms were introduced to maximize cost efficiency via multicast beamforming optimization.

As for multi-media content caching optimization, each UE in [24] can acquire its requested content from either a single F-AP or multiple F-APs via joint transmission, and content transmission delay was analyzed by considering caching states of F-APs, transmission mode selection and content sharing strategy among F-APs. To solve the formulated delay minimization problem, the number of optimization variables were first reduced by carefully analyzing problem structure and then genetic algorithm was adopted to solve a simplified content caching problem. In [25], NOMA transmission from one F-AP to two users was considered and each user can directly cancel the interference of the other owing to local content caching. Then, upper plane method based content caching scheme was developed to minimize F-AP's transmission power under caching space limitation of users. In [26], under the limit of caching update frequency, caching update was formulated as a constrained Markov decision process, whose objective was to minimize the average age of information of served user content requests. To deal with large state space, the primal problem was decomposed into multiple subproblems and each subproblem was further solved using deep reinforcement learning. In [27], authors addressed content caching in an ultra-dense F-RAN featuring a stochastic geometric network model and a spatial-temporal user demand model. With mean-field game theory, a distributed caching scheme was developed, whose complexity is independent of the number of F-APs.

Given the coupling of radio and caching resource allocation, literature [28], [29], [30], [31] studied their two-timescale optimization. In [28], content recommendation was utilized to reshape users' content requests, and the joint optimization of recommendation, content caching and beamforming was studied to minimize content transmission latency. Faced with the resulted two-timescale problem, authors decoupled it into multiple parallel subproblems, each of which corresponded to a given system state. Particularly, in each subproblem, content caching was handled by integer relaxation and greedy rounding technique. The authors in [29] optimized F-APs' content delivery at each time frame while updating cached contents at each F-AP on a larger timescale without knowing the prior content popularity distribution. In [30], long-term content placement and short-term remote radio head (RRH) clustering together with beamforming were jointly addressed to balance the fronthaul traffic and power consumption. In [31], an F-AP cache size allocation and beamforming design problem was tackled, where storage resource was allocated based on the long-term statistical information while the beamformer was designed based on the instantaneous channel state information.

C. Our Contribution and Novelty

In general, cache resource is adjusted on a larger timescale than radio and computation resource [29]. Nevertheless, this feature has not been captured in literatures [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19] studying service caching and resource allocation in computation offloading scenarios. In this paper, we deal with the two-timescale joint optimization of service caching, user service mode selection, subchannel assignment, and computation resource allocation in an F-RAN computation offloading scenario, whose system model is significantly different from those in literatures studying traditional

Authorized licensed use limited to: Auburn University. Downloaded on August 24,2023 at 05:49:03 UTC from IEEE Xplore. Restrictions apply.

multi-media content caching and radio resource allocation. In the F-RAN system, each user can either execute its task locally at an F-AP (i.e., the edge mode) or offload the task to the cloud via RRHs (i.e., the cloud mode). The resulted problem aims at minimizing the long-term weighted sum of average user group latency cost and service caching cost. Although literature [28], [29], [30], [31] in subsection B have addressed two-timescale resource management problems, their proposals are either heuristic or only deal with continuous caching variables. Specifically, authors in [28] used relaxation and rounding technique to optimize content caching, which lacked of rigorous analysis about optimality. In [30], 0-1 caching variables were also relaxed to continuous variables whose values were identified by solving a carefully constructed convex problem. Nevertheless, how to recover these values to integer values was not discussed and the theoretic optimality of resulted long-term performance was unknown. In [31], cache size allocated to each F-AP was defined as continuous variables, which is not the case in our paper. In [29], long-term content caching decisions were made by solving a well designed problem, based on the intuitive that minimizing its objective contributes to less requirement on fronthaul rate, possibly reducing long-term power consumption.

Different from [28], [29], [30], [31], we propose to directly handle long-term integer caching variables with the help of multi-agent RL (MARL), and it can identify which services are cached at each F-AP by utilizing historical samples of user service requests and channel states. With an easily designed reward function, the RL algorithm can take the small timescale resource allocation into account and converge to local optimal caching decisions. Moreover, since our proposal is model-free, it provides a general framework to optimize long-term performance in two-timescale resource management, and can well fit into the case where the explicit relationship between the long-term objective and caching decisions can not be derived. Note that various RL algorithms have been utilized for cache resource management [32], [33], [34], [35], [36], [37]. However, the prior work [35] mainly addressed content update policies instead of content placement decisions, while the work [36] only assumed a single F-AP. The proposed scheme in [37] can be applied to our service caching problem by redesigning the reward function, which will be taken as one of the baselines in our simulation study in Section V. As for resource allocation on small timescale, we take coalition game as our tool, which has been widely applied in user clustering and resource allocation in NOMA enabled wireless networks [13], [38], [39]. Particularly, users as game players participate in a coalition formation game. The service mode together with the transmission subchannel of each user can be determined according to which coalition it joins.

The main contributions of this paper are as follows.

- The problem of computation offloading in an F-RAN with service caching is considered, where each user can choose between two service modes, namely the edge mode and the cloud mode. Under this setting, a joint cache, radio, and computation resource allocation problem is formulated with a two-timescale structure. The objective of radio and computation resource allocation on a small timescale is to minimize average user group latency, while the objective of cache resource optimization on a large timescale is to minimize the long-term weighted sum of the latency and service caching costs.
- To deal with the two-timescale optimization problem, a low-complexity coalition formation based approach is developed for radio and computation resource allocation under fixed service placement, which adapts to instantaneous user service requests and channel states. Then, an MARL algorithm is developed to decide which services are to be cached at each F-AP. With this approach, two main challenges are overcome, which are caused by the long-term objective function that does not have an explicit form and the discrete caching decision variables.
- The convergence, optimality, and complexity of the proposed algorithms are rigorously analyzed. Numerical simulation results are provided to demonstrate their superior performance compared to several baselines. In addition, the effects of several key parameters on system performance are also investigated.

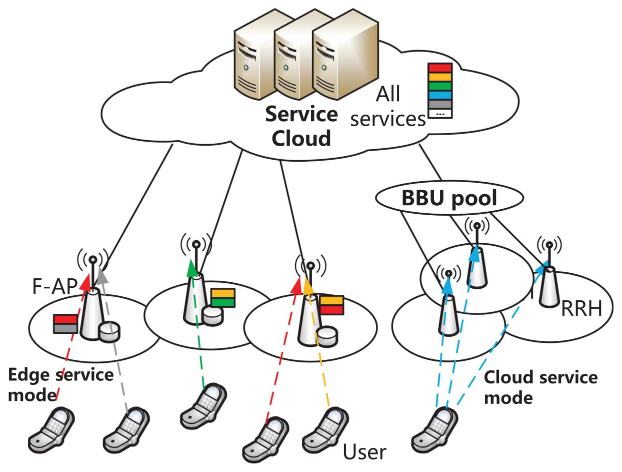

Fig. 1. The F-RAN system model considered in this paper.

The remainder of this paper is organized as follows. The F-RAN system model with service caching and computation offloading is described in Section II. Section III formulates the two-timescale radio, computation, and cache resource allocation problem. The proposed algorithms are developed in Section IV, while our simulation and analysis are presented in Section V. Finally, the paper is concluded in Section VI. The notation used in this paper is summarized in Table I.

II. System Model

As shown in Fig. 1, the F-RAN considered in this paper consists of one service cloud, multiple RRHs, multiple F-APs, and multiple users. Each RRH, F-AP, and user is equipped with a single antenna. The service cloud stores all the services that are of interest to the users, while the RRHs only implement radio frequency functions, which are connected to a baseband unit (BBU) pool through fronthaul links. Each F-AP is capable of caching services as well as local resource management [40]. Each user can either offload the computation task of its requested

Authorized licensed use limited to: Auburn University. Downloaded on August 24,2023 at 05:49:03 UTC from IEEE Xplore. Restrictions apply.

TABLE I
NOTATION

Notation	Definition
$\mathcal{F}$	The set of F-APs
$\mathcal{S}$	The set of services potentially requested by all the users
$\mathcal{U}$	The set of users
$\mathcal{N}$	The set of RRHs
V_f	The reserved caching capacity of F-AP f
δ_s	The input date size for the task execution of service s
ζ_s	The computation workload of service s's computation task
ϵ_s	The storage size required for caching service s
x_u^{f,b_f}	A 0-1 indicator representing whether user u transmits to F-AP f on subchannel b_f $(f \geq 1)$
x_u^{0,b_0}	A 0-1 indicator representing whether user u operates in cloud service mode
$c_{f,s}$	A 0-1 indicator representing whether service s is cached at F-AP f
$y_{u,s}$	A 0-1 indicator representing whether user u requests service s
λ_u	The size of data uploaded by user u
η_u	The computation workload offloaded by user u
$\mathbf{H}$	The collection of all the channel coefficients
$\mathbf{x}$	The collection of all the x_u^{f,b_f} with $f \geq 0$
$\mathbf{C}$	The service caching matrix with (f, s)-th entry being $c_{f,s}$
$\mathbf{Y}$	The service request matrix with (u, s)-th entry being $y_{u,s}$
$d_{f,u}$	The computation resource allocated to user u by F-AP f
$\mathbf{d}$	The collection of all the $d_{f,u}$ with $f \geq 1$
$\mathcal{U}_f$	The user group associated with F-AP f
$\mathcal{U}_0$	The user group associated with RRHs
t_f	The user group latency of $\mathcal{U}_f$ with $f \geq 0$
t	The average user group latency of all user groups
α	The cost of one unit latency
β	The cost of scaling out the F-AP's cache by one unit size
$l_{f,\mathbf{C}}$	The caching cost of F-AP f under caching matrix $\mathbf{C}$
κ	The weight balancing latency and caching costs
Ω	The weighted system cost
Θ	A parameter to normalize the reward in Algorithm 2

service to an F-AP, which is referred to as the edge service mode, or offload the task to the service cloud via RRHs and the BBU pool, which is called the cloud service mode.

The set of services, F-APs, RRHs, and users are denoted by $\mathcal{S} = \{1, 2, \ldots, S\}$, $\mathcal{F} = \{1, 2, \ldots, F\}$, $\mathcal{N} = \{1, 2, \ldots, N\}$, and $\mathcal{U} = \{1, 2, \ldots, U\}$, respectively. Each F-AP $f \in \mathcal{F}$ has an initial cache capacity V_f in GB and possesses computation capability $d_{f,\max}$ in CPU cycles per second. Each service is characterized by a three-tuple $\{\delta_s, \zeta_s, \epsilon_s\}$. Specifically, δ_s and ζ_s denote the input data size in bits and the computation workload in CPU cycles for the service's computation task execution, respectively, and ϵ_s in GB represents the storage size needed to cache the necessary code and data of service s. Define $y_{u,s} = 1$ to indicate that user u requests service s; and $y_{u,s} = 0$ otherwise. Thus the total size of input data and the total computation workload of user u can be expressed as $\lambda_u = \sum_{s\in\mathcal{S}} y_{u,s}\delta_s$ and $\eta_u = \sum_{s\in\mathcal{S}} y_{u,s}\zeta_s$, respectively.

Let $c_{f,s} = 1$ indicate that F-AP f caches the code and data for service s; and $c_{f,s} = 0$ otherwise. Each F-AP f is pre-assigned with a set of subchannels defined by $\mathcal{B}_f$, satisfying $\mathcal{B}_f \cap \mathcal{B}_{f'} = \phi$, for all $f \neq f'$. The bandwidth of each subchannel is W and each user associated with F-AP f shall be allocated with only one subchannel [41]. For the RRHs, we follow the common setting in cloud radio access networks, where the spectrum of bandwidth W_C is fully shared by all the users in the cloud mode [42]. But for convenience, the set of subchannels $\mathcal{B}_0$ is still defined for all the RRHs with $|\mathcal{B}_0| = 1$. Meanwhile, it is assumed that the spectrum occupied by the RRHs is non-overlapping with that of the F-APs and hence there is no mutual interference between the users in the edge mode and those in the cloud mode. Define the radio resource allocation variable of user u as x_u^{f,b_f}. When $f \geq 1$, $x_u^{f,b_f} = 1$ means user u transmits to F-AP f on subchannel $b_f \in \mathcal{B}_f$, i.e., it operates in the edge service mode, while $x_u^{f,b_f} = 0$ otherwise. When $f = 0$, x_u^{f,b_f} indicates whether user u is in the cloud service mode ($x_u^{f,b_f} = 1$) or not ($x_u^{f,b_f} = 0$). Denote the collection of all x_u^{f,b_f}'s with $u \in \mathcal{U}$, $f \in \mathcal{F} \cup \{0\}$ and $b_f \in \mathcal{B}_f$ by a network-wide radio resource allocation vector $\mathbf{x}$. In addition, since the computation resource in the cloud is sufficient, we only consider the computation resource allocation at the F-APs. Define $d_{f,u}$ as the amount of computation resource allocated to user u at F-AP f, and denote the network-wide computation resource allocation vector by $\mathbf{d}$ that is a collection of all $d_{f,u}$'s with $u \in \mathcal{U}$ and $f \in \mathcal{F}$. In the following, the transmission model and computation model for users in different service modes are specified.

A. Edge Service Mode

In the edge service mode, a user first uploads the necessary input data to an F-AP that caches the requested service. Then service program will be executed at this F-AP. Consider user u transmits its data to F-AP f on subchannel b_f. When there is no other user sharing subchannel b_f with user u, its transmission rate is given by

$$R_{u,0}^{f,b_f} = W \log_2\left(1 + \frac{p_u |h_u^{f,b_f}|^2}{n_0\, W}\right), \tag{1}$$

where p_u is the uplink transmit power of user u, h_u^{f,b_f} represents the channel co-efficient between user u and F-AP f on subchannel b_f, and n_0 is the noise power spectral density. Then, the corresponding transmission time is

$$t_{u,0}^{f,b_f} = \frac{\lambda_u}{R_{u,0}^{f,b_f}}. \tag{2}$$

To reduce the transmission waiting time, another user u' will also be allowed to transmit simultaneously on the same subchannel b_f, if needed, by leveraging NOMA as in [45], [46], where only two-user NOMA is considered to limit the signal processing complexity. Specifically, when the channel gain of user u is worse than that of user u', the transmission rate of user u will be the same as that in the case of orthogonal multiple access (OMA). Since the F-APs are allocated with disjoint sets of subchannels, there is no interference from the users associated with other F-APs, and hence the transmission rate of user u can be written as

$$R_{u,1}^{f,b_f} = W \log_2\left(1 + \frac{p_u |h_u^{f,b_f}|^2}{n_0\, W}\right). \tag{3}$$

Authorized licensed use limited to: Auburn University. Downloaded on August 24,2023 at 05:49:03 UTC from IEEE Xplore. Restrictions apply.

The transmission time of user u incurred by uploading data of size λ_u is given by

$$t_{u,1}^{f,b_f} = \frac{\lambda_u}{R_{u,1}^{f,b_f}}. \tag{4}$$

When the channel gain of user u is better than that of user u', the transmission rate of user u under simultaneous transmission of user u' is

$$R_{u,2}^{f,b_f} = W \log_2 \left(1 + \frac{p_u |h_u^{f,b_f}|^2}{p_{u'} |h_{u'}^{f,b_f}|^2 + n_0 W}\right). \tag{5}$$

If user u can complete its transmission within $t_{u,'1}^{f,b_f}$, i.e. $R_{u,2}^{f,b_f} \cdot t_{u,'1}^{f,b_f} \geq \lambda_u$, then $t_{u,2}^{f,b_f} = \lambda_u / R_{u,2}^{f,b_f}$. If user u cannot complete its transmission within $t_{u,'1}^{f,b_f}$, user u will continue to transmit the remaining data without the interference from user u'. In this case, the transmission time of user u can be written as

$$t_{u,2}^{f,b_f} = \frac{\lambda_u - t_{u,'1}^{f,b_f} R_{u,2}^{f,b_f}}{R_{u,0}^{f,b_f}} + t_{u,'1}^{f,b_f}. \tag{6}$$

Thus, the wireless transmission time of user u, which communicates with F-AP f over subchannel b_f, is given by

$$\begin{aligned} t_u^{f,b_f} = & \ \mathbb{I}\left\{\sum_{u\in\mathcal{U}} x_u^{f,b_f} = 1\right\} \cdot t_{u,0}^{f,b} \\ & + \mathbb{I}\left\{\sum_{u\in\mathcal{U}} x_u^{f,b_f} = 2, |h_u^{f,b_f}|^2 < |h_{u'}^{f,b_f}|^2\right\} \cdot t_{u,1}^{f,b_f} \\ & + \mathbb{I}\left\{\sum_{u\in\mathcal{U}} x_u^{f,b_f} = 2, |h_u^{f,b_f}|^2 > |h_{u'}^{f,b_f}|^2\right\} \cdot t_{u,2}^{f,b_f}, \end{aligned} \tag{7}$$

where $\mathbb{I}\{\cdot\}$ is an indicator function that equals to 1 when the condition is true.

The service's task execution time of user u depends on the workload of the offloaded task, η_u, as well as the CPU frequency allocated by F-AP f, $d_{f,u}$, which is calculated as

$$t_u^{ex} = \frac{\eta_u}{d_{f,u}}. \tag{8}$$

B. Cloud Service Mode

In the cloud service mode, user u transmits its input data to the service cloud through RRHs and the BBU pool. Since the services' task execution time in the cloud can be neglected owing to its sufficient computing capability and the fronthaul between RRHs and the BBU pool can be assumed ideal, we mainly consider the wireless transmission time from users to RRHs and the data transmission time over the non-ideal backhaul from the BBU pool to the service cloud. For users accessing the RRHs, whose set is denoted by $\mathcal{U}_0 \subseteq \mathcal{U}$, their transmission is based on uplink joint reception. After re-indexing these users, their set is re-defined as $\mathcal{U}_C = \{1, 2, \ldots, |\mathcal{U}_0|\}$ and u_C is the new index of user $u \in \mathcal{U}$ in $\mathcal{U}_C$. Define $\mathbf{H_C}$ as the channel matrix between all the users in $\mathcal{U}_0$ and the RRHs, whose (n, u_C)th entry is the channel co-efficient between user u and RRH n.

According to [47], the receiving beamforming matrix is given by $\mathbf{V} = \mathbf{A}^{-1}\mathbf{H}_C\mathbf{P}^{\frac{1}{2}}$, where $\mathbf{A} = \mathbf{H}_C\mathbf{P}\mathbf{H}_C{}^H + n_0 W_C \mathbf{I}$ and $\mathbf{P}$ is a diagonal matrix with the u_C-th element P_{u_C} being p_u. For user u, its signal-to-interference-and-noise ratio (SINR) is given by

$$SINR_u = \frac{P_{u_C} \left|\mathbf{v}_{u_C}^H \mathbf{h}_{C,u_C}\right|^2}{\sum\limits_{j\neq u_C, j\in\mathcal{U}_C} P_j \left|\mathbf{v}_{u_C}^H \mathbf{h}_{C,j}\right|^2 + n_0 W_C \mathbf{v}_{u_C}^H \mathbf{v}_{u_C}}, \tag{9}$$

where $\mathbf{v}_{u_C} \in \mathbb{C}^{N\times 1}$ is the u_Cth column of matrix $\mathbf{V}$, and $\mathbf{h}_{C,u_C} \in \mathbb{C}^{N\times 1}$ is the u_Cth column of $\mathbf{H}_C$. The uplink transmission rate from user u to RRHs can be written as

$$R_{C,u} = W_C \log_2 (1 + SINR_u). \tag{10}$$

Therefore, the wireless transmission time of user u is calculated as

$$t_{C,u} = \frac{\lambda_u}{R_{C,u}}. \tag{11}$$

After the centralized signal processing in the BBU pool, it further forwards users' input data to the service cloud over backhaul with a constant transmission rate R_0. To transmit the data of all the users in cloud service mode, the time consumed is given by

$$t^{bk} = \frac{\sum_{u\in\mathcal{U}_0} \lambda_u}{R_0}. \tag{12}$$

III. Problem Formulation

To mitigate the signalling exchanges among F-APs and the BBU pool for identifying network-wide user latency performance, we assume that each F-AP and the BBU pool only care about the latency performance of the user group associated with them. Specifically, user group latency is defined to measure the time elapsed from when users in a group start sending their service requests till when all the users in the group receive their task execution results. Denote the user group $\mathcal{U}_f \subseteq \mathcal{U}$ served by F-AP f by $\mathcal{U}_f = \{u | \sum_{b_f\in\mathcal{B}_f} x_u^{f,b_f} = 1\}$. The user group latency is given by

$$t_f = \max_{u\in\mathcal{U}_f} \sum_{b_f\in\mathcal{B}_f} x_u^{f,b_f} t_u^{f,b_f} + \max_{u\in\mathcal{U}_f} t_u^{ex}, \tag{13}$$

where it is assumed that the latency induced by sending the task execution results to the user group through the downlink is negligible. This assumption is reasonable since the data volume of results is usually very small, e.g., in object detection tasks. Meanwhile, (13) also implicitly means that the computation resource at F-AP f will not be allocated until it receives all the data uploaded by users in $\mathcal{U}_f$, which significantly helps simplify the computation resource allocation design.

For user group $\mathcal{U}_0 \subseteq \mathcal{U}$ in the cloud mode, i.e., $\mathcal{U}_0 = \{u | x_u^{f,b_0} = 1\}$, the user group latency is expressed as

$$t_0 = \max_{u\in\mathcal{U}_0} t_{C,u} + t^{bk}. \tag{14}$$

Authorized licensed use limited to: Auburn University. Downloaded on August 24,2023 at 05:49:03 UTC from IEEE Xplore. Restrictions apply.

Then, the average user group latency is defined as

$$t = \frac{1}{F+1}\sum_{f=0}^{F} t_f, \tag{15}$$

where t_f is set to 0 if $\mathcal{U}_f = \phi$.

Caching services at F-APs will be effective to reduce the user group latency, which allows services' computation tasks to be executed locally. Nevertheless, considering the limited storage capacity of F-APs, caching decisions must be properly made to balance the latency and the caching cost incurred from expanding storage spaces. Considering that the storage resource is often allocated on a larger timescale than the computation and radio resource, a multi-dimensional resource management problem is formulated with a two-timescale structure. Specifically, service caching on the long-term aims to minimize a system cost consisting of the expected latency cost and the caching cost, while the short-term radio and computation resource allocation intends to achieve a minimum average user group latency by taking current channel coefficients, user service requests, and service caching states into account. Let $\mathbf{H}$, $\mathbf{Y}$, and $\mathbf{C}$ denote the collection of all the channel coefficients, $y_{u,s}$, and $c_{f,s}$, respectively. Recall that $\mathbf{x}$ is the network-wide radio resource allocation vector and $\mathbf{d}$ is the network-wide computation resource allocation vector. To minimize a weighted system cost, the corresponding optimization problem is given by (16).

$$\min_{\mathbf{C}} \Omega = \underbrace{\kappa\,\alpha \cdot \mathbb{E}_{\mathbf{H},\mathbf{Y}}\left[\min_{\mathbf{x},\mathbf{d}}\ t(\mathbf{x},\mathbf{d},\mathbf{H},\mathbf{Y},\mathbf{C})\right]}_{latency\ cost} + (1-\kappa)\ \underbrace{\beta\sum_{f\in\mathcal{F}} l_{f,\mathbf{C}}}_{caching\ cost} \tag{16}$$

$$\begin{aligned}
\text{s.t. } & (\text{a1})\ c_{f,s} = \{0,1\}, \forall f\in\mathcal{F}, \forall s\in\mathcal{S},\\
& (\text{a2})\ l_{f,\mathbf{C}} = \begin{cases} 0, & \text{if } \sum_{s\in\mathcal{S}} \epsilon_s c_{f,s} \le V_f,\\ \frac{1}{V_f}\left(\sum_{s\in\mathcal{S}} \epsilon_s c_{f,s} - V_f\right), & \text{otherwise,}\end{cases}\\
& (\text{a3})\ x_u^{f,b_f} \in \{0,1\}, \forall f\in\mathcal{F}\cup\{0\}, \forall b_f\in\mathcal{B}_f, \forall u\in\mathcal{U},\\
& (\text{a4})\ \sum_{u\in\mathcal{U}} x_u^{f,b_f} \le 2, \forall f\in\mathcal{F}, \forall b_f\in\mathcal{B}_f,\\
& (\text{a5})\ \sum_{f\in\mathcal{F}\cup\{0\}}\sum_{b_f\in\mathcal{B}_f} x_u^{f,b_f} = 1, \forall u\in\mathcal{U},\\
& (\text{a6})\ \sum_{b_f\in\mathcal{B}_f} x_u^{f,b_f} \le \sum_{s\in\mathcal{S}} y_{u,s}c_{f,s}, \forall u\in\mathcal{U}, \forall f\in\mathcal{F},\\
& (\text{a7})\ 0 \le d_{f,u} \le \sum_{b_f} x_u^{f,b_f} d_{f,\max}, \forall u\in\mathcal{U}, \forall f\in\mathcal{F},\\
& (\text{a8})\ \sum_{u\in\mathcal{U}} d_{f,u} \le d_{f,\max}, \forall f\in\mathcal{F}
\end{aligned}$$

In (16), α and β represent the unit cost of average user group latency and the cost of expanding F-AP's cache size by one unit, respectively; κ is a weight factor; and $l_{f,\mathbf{C}}$ represents the total caching cost at F-AP f. Based on (a2), if the total storage demands of all the services cached by F-AP f do not exceed its reserved cache capacity, the caching cost of F-AP f is 0 while additional cost is incurred otherwise. Note that fog nodes can be built with virtualization technique [43], [44], based on which caching resource can be dynamically scaled up on demand but with potentially more operation power consumption. Constraint (a3) indicates that the variables in the radio resource allocation vector $\mathbf{x}$ are all integers, while Constraint (a4) states that each subchannel of an F-AP can be occupied by at most two users simultaneously using NOMA [45], [46]. Constraint (a5) states that each user can be served by only one F-AP-subchannel resource pair or by all the RRHs. Constraint (a6) means that each user can be served by an F-AP only when its requested service has been cached at the F-AP [22]. Finally, Constraints (a7) and (a8) jointly provide the feasible region of $\mathbf{d}$.

In [31] and [48], the authors proposed to adopt sample average approximation (SAA) to deal with problems similar to (16), by which the expectation of $t(\cdot)$ over $\mathbf{H}$ and $\mathbf{Y}$ is substituted by the sample average. More specifically, assuming a set of Q samples of $\mathbf{H}$ and $\mathbf{Y}$ are available, the objective of (16) after SAA is written as

$$\hat{\Omega} = \frac{\kappa\alpha}{Q}\sum_{q=1}^{Q} \min_{\mathbf{x}(q),\mathbf{d}(q)} t(\mathbf{x}(q),\mathbf{d}(q),\mathbf{H}(q),\mathbf{Y}(q),\mathbf{C}) + (1-\kappa)\beta\sum_{f\in\mathcal{F}} l_{f,\mathbf{C}}, \tag{17}$$

where $\mathbf{H}(q)$ and $\mathbf{Y}(q)$ are the qth sample of $\mathbf{H}$ and $\mathbf{Y}$, respectively, and $\mathbf{x}(q)$ and $\mathbf{d}(q)$ are the network-wide radio resource allocation and computation resource allocation for given $\mathbf{H}(q)$ and $\mathbf{Y}(q)$, respectively.

Then, the two-timescale problem (16) can be successfully transformed into the following single timescale problem.

$$\begin{aligned}
\min_{\mathbf{C},\{\mathbf{x}(q),\mathbf{d}(q)\}_{q=1}^{Q}} & \frac{\kappa\alpha}{Q}\sum_{q=1}^{Q} t(\mathbf{x}(q),\mathbf{d}(q),\mathbf{H}(q),\mathbf{Y}(q),\mathbf{C}) + (1-\kappa)\beta\sum_{f\in\mathcal{F}} l_{f,\mathbf{C}}, & (18)\\
\text{s.t.} \quad & (\text{a1}), (\text{a2}) \text{ with } (\text{a3})-(\text{a8}) \text{ satisfied for } (\mathbf{H}_{\text{q}}, \mathbf{Y}_{\text{q}}),\\
& q\in\{1,2,\ldots,Q\}.
\end{aligned}$$

Nevertheless, it is quite challenging to solve Problem (18) with traditional optimization approaches. On one hand, to achieve a tight approximation, SAA requires the number of samples Q to be sufficiently large [31], leading to a potentially huge solution space that may be intractable for heuristic search methods (e.g., genetic algorithms (GA)). On the other hand, integer relaxation and convexification techniques may be applied to make Problem (18) more tractable. However, a naive integer recovery scheme can lead to large performance loss, while convexification is not an easy task due to the complicated relationship between t and optimization variables. Facing these challenges, instead of solving the transformed problem (18), this

Authorized licensed use limited to: Auburn University. Downloaded on August 24,2023 at 05:49:03 UTC from IEEE Xplore. Restrictions apply.

paper directly deals with problem (16) by integrating model-free machine learning and game theory as elaborated in the next section.

IV. Algorithm Design

In this section, a short-term radio and computation resource allocation subproblem is first solved by a coalition formation based algorithm under fixed service caching placements, channel states, and user service request states. After that, an MARL algorithm is proposed to optimize the long-term service caching.

A. Short-Term Radio and Computation Resource Allocation Algorithm Design

We first deal with the radio and computation resource allocation subproblem for given long-term service placement $\mathbf{C}$ at the F-APs, which is given by

$$\min_{\mathbf{x},\mathbf{d}} t(\mathbf{x},\mathbf{d},\mathbf{H},\mathbf{Y},\mathbf{C}) \quad \text{s.t. (a3)–(a8).} \tag{19}$$

First, the optimal computation resource allocation is studied under fixed radio resource allocation. With given radio resource allocation, Problem (19) can be decoupled among the F-APs, which is equivalent to the minimization of the maximum computation time among all the users in $\mathcal{U}_f$ for each F-AP f. The corresponding local optimization problem is given by

$$\begin{aligned} \min_{d_{f,u}} \max_{u\in\mathcal{U}_f} & \frac{\eta_u}{d_{f,u}} \\ \text{s.t. } & (\mathrm{b1}) 0 \le \mathrm{d_{f,u}} \le \mathrm{d_{f,max}},\ \mathrm{u}\in\mathcal{U}_\mathrm{f}, \\ & (\mathrm{b2}) \sum_{\mathrm{u}\in\mathcal{U}_\mathrm{f}} \mathrm{d_{f,u}} \le \mathrm{d_{f,max}}, \end{aligned} \tag{20}$$

which is a minimax linear fractional programming problem. We introduce Theorem 1 below to solve this problem.

Theorem 1: When the computation resource allocated to each user $u\in\mathcal{U}_f$ satisfies $d^*_{f,u} = \frac{\eta_u}{\sum_{u\in\mathcal{U}_f}\eta_u} d_{f,\max}$, the optimal objective value of Problem (20) is achieved, which is equal to $t_u^{ex*} = \frac{\eta_u}{d^*_{f,u}} = \frac{\sum_{u\in\mathcal{U}_f}\eta_u}{d_{f,\max}}$.

Proof: First, it is claimed that the optimal solution to Problem (20) is achieved when the equality of Constraint (b2) holds, since consuming unused computation resource can always decrease the objective value. With this in mind, denote the vector constituted by $d^*_{f,u}$ as $\mathbf{d}^*_f$ and define the set of all possible $\mathbf{d}_f$ satisfying Constraint (b1) and the equality of Constraint (b2) by $\mathcal{D}_f$. For any $\mathbf{d}\in\mathcal{D}_f/\mathbf{d}^*_f$, there must exist a user $u\in\mathcal{U}_f$ such that $d_{f,u} < d^*_{f,u}$. Then, we can obtain $t_u^{ex} > t_u^{ex*}$, and thus $\max_{u\in\mathcal{U}_f} t_u^{ex} > t_u^{ex*} = \max_{u\in\mathcal{U}_f} t_u^{ex*}$. Therefore the theorem holds. ■

Based on the above theorem, the closed-form expression of the optimal network-wise computation resource allocation vector $\mathbf{d}^*$ can be derived. For each of its element $d^*_{f,u}$, we have

$$d^*_{f,u} = \begin{cases} \frac{\eta_u}{\sum_{u\in\mathcal{U}_f}\eta_u} d_{f,\max}, & \text{if } \sum_{b_f\in\mathcal{B}_f} x_u^{f,b_f} = 1, \\ 0, & \text{otherwise.} \end{cases} \tag{21}$$

Then, Problem (19) can be equivalently reduced to the following radio resource allocation problem.

$$\min_{\mathbf{x}} t(\mathbf{H},\mathbf{Y},\mathbf{C},\mathbf{x},\mathbf{d}^*(\mathbf{x})) \quad \text{s.t. (a3)–(a6).} \tag{22}$$

Note that resource allocation variables in Problem (22) are integers and the objective contains indicator functions, since t is related to t_u^{f,b_f} given by (7). Hence, optimal algorithms can suffer high complexity. Facing this issue, a low-complexity algorithm is developed that is inspired by coalitional game theory. Specifically, define a coalition set by $\mathcal{M} = \{\mathcal{M}_{f,b_f} | f\in\mathcal{F}\cup\{0\}, b_f\in\mathcal{B}_f\}$, where $\mathcal{M}_{f,b_f}\subseteq\mathcal{U}$, and user u is allocated with resource pair (f,b_f), i.e., $x_u^{f,b_f}=1$, if it is a member of $\mathcal{M}_{f,b_f}$. Thus, solving Problem (22) optimally is equivalent to finding an optimal coalitional structure $\mathcal{M}^*$ such that

i) $\mathcal{M}^*_{f,b_f}\cap\mathcal{M}^*_{f',b_{f'}} = \phi$, for any resource pairs (f,b_f) and $(f',b_{f'})$ with $(f,b_f)\neq(f',sb_{f'})$, in order to satisfy Constraint (a5);
ii) $|\mathcal{M}^*_{f,b_f}|\le 2$, for any resource pair (f,b_f) with $f\ge 1$, to meet constraint (a4);
iii) $u\notin\mathcal{M}^*_{f,b_f}$, for all $b_f\in\mathcal{B}_f$, if the service requested by user u is not cached by F-AP f, which is to satisfy Constraint (a6).

A coalitional structure satisfying the above three conditions is called a feasible structure.

A common solution concept adopted in coalitional games is Nash-stable [49]. Under a Nash-stable coalitional structure $\mathcal{M}$, any user will have no incentive to unilaterally switch from one coalition to another while keeping the structures feasible. To achieve a stable structure, an iterative algorithm is designed as in Algorithm 1. The main idea is that each user u checks each coalition $\mathcal{M}_{f,b_f}$ based on a switching rule, and user u joins a new coalition once the rule holds. More formally, we have the following definition.

Definition 1: (*Coalition Switching Rule*) Consider a feasible coalitional structure $\mathcal{M}$, under which $u\in\mathcal{M}_{f,b_f}$. User u will switch from coalition $\mathcal{M}_{f,b_f}$ to $\mathcal{M}_{f',b_{f'}}$ where $(f,b_f)\neq(f',b_{f'})$, if the newly formed structure $\mathcal{M}_{new}$ after the switching is feasible and meanwhile $t_f(\mathcal{M}_{new}) + t_{f'}(\mathcal{M}_{new}) < t_f(\mathcal{M}) + t_{f'}(\mathcal{M})$.

According to Definition 1, each coalition switching results in a strict decrease in the objective of (22). Hence, there is similarity between our proposal and greedy based approach in terms of optimization behavior. Meanwhile, since the objective of (22) is lower bounded, the convergence of Algorithm 1 directly holds. Moreover, the convergence of Algorithm 1 also indicates that no user is willing to leave its current coalition, which corresponds to a local optimal solution to short-term resource allocation problem (22). The complexity of the algorithm is given by $\mathcal{O}(U\sum_{f\in\mathcal{F}\cup\{0\}}|\mathcal{B}_f|D)$, where D is the number of repetitions till convergence. In addition, since it is intractable to know the exact value of D, it is difficult to analyze convergence speed directly. But at least, according to the time complexity above, we can infer that more F-APs (a lager $|\mathcal{F}|$), more available

Authorized licensed use limited to: Auburn University. Downloaded on August 24,2023 at 05:49:03 UTC from IEEE Xplore. Restrictions apply.

Algorithm 1: Greedy Coalition Formation Based Resource Allocation.

```
1:  Initiate a coalition structure $\mathcal{M}$ and denote the
    coalition that user $u$ joins under $\mathcal{M}$ by $\mathcal{M}(u)$;
2:  **repeat**
3:    **for** user $u = 1 : U$ **do**
4:      **for** each coalition $\mathcal{M}_{f',b_{f'}}$ **do**
5:        **if** the switching rule holds for user $u$, coalition
          $\mathcal{M}(u)$ and $\mathcal{M}_{f',b_{f'}}$ **then**
6:          User $u$ leaves $\mathcal{M}(u)$ and joins $\mathcal{M}_{f',b_{f'}}$;
7:          Update the coalition structure $\mathcal{M}$ with the
            newly formed one;
8:        **else**
9:          The coalition structure remains unchanged;
10:       **end if**
11:     **end for**
12:   **end for**
13: **until** Convergence
14: Derive the final $\mathbf{x}$ and $\mathbf{d}^*(\mathbf{x})$ under converged $\mathcal{M}$;
```

subchannels per F-AP (a larger $|\mathcal{B}_f|$) and more UEs (a larger U) lead to slower convergence speed due to longer convergence time.

B. Long-Term Service Caching Algorithm Design

After solving the short-term radio and computation resource allocation problem, we next tackle the long-term service caching problem as follows:

$$\min_{\mathbf{C}} \Omega = \kappa\alpha \cdot \mathbb{E}_{\mathbf{H},\mathbf{Y}}\left[t(\mathbf{x}^*, \mathbf{d}^*, \mathbf{H}, \mathbf{Y}, \mathbf{C})\right] + (1-\kappa)\beta \cdot \sum_{f\in\mathcal{F}} l_{f,\mathbf{C}},$$
$$\text{s.t.} \quad \text{(a1), (a2)}, \tag{23}$$

where $\mathbf{x}^*$ and $\mathbf{d}^*$ denote the network-wide radio resource allocation and computation resource allocation decisions computed by Algorithm 1, respectively, for given $\mathbf{H}$, $\mathbf{Y}$, and service caching matrix $\mathbf{C}$.

Solving Problem (23) needs to deal with two challenges. First, since it is difficult to derive the explicit relationship of $\mathbf{x}^*$ and $\mathbf{d}^*$ with $\mathbf{C}$, the objective Ω has no explicit form. Second, the elements $c_{f,s}$ in $\mathbf{C}$ are all integers. Although exhaustive search can be adopted to find optimal solution, its complexity grows exponentially with regard to the numbers of F-APs and services. Therefore, we propose to use a model-free MARL algorithm to achieve sub-optimal caching decisions, which is based on stochastic learning automata (SLA) [50], [51]. In our algorithm, to handle the explosive growth of the action space in the single-agent setting, $F \times S$ learning agents are created, each of which corresponds to one of the F-AP-service pairs (f, s). For agent i, its action set is denoted by $\mathcal{A}_i = \{a_{i,1}, a_{i,2}\}$, where $i = (f-1)S + s$. Selecting action $a_{i,1}$ means F-AP f caches service s, i.e., $c_{f,s} = 1$, while taking action $a_{i,2}$ means the opposite case of $c_{f,s} = 0$.

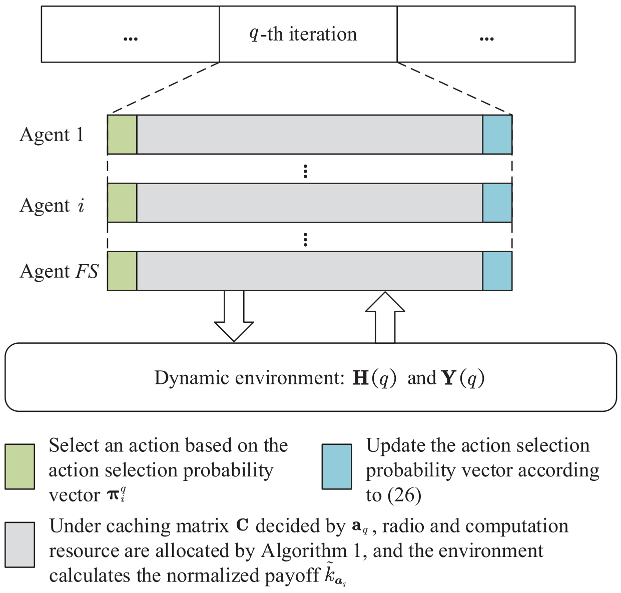
...
q-th iteration
...
Agent 1
Agent i
Agent FS
Dynamic environment: $\mathbf{H}(q)$ and $\mathbf{Y}(q)$
Select an action based on the action selection probability vector $\boldsymbol{\pi}_i^q$
Update the action selection probability vector according to (26)
Under caching matrix $\mathbf{C}$ decided by $\mathbf{a}_q$, radio and computation resource are allocated by Algorithm 1, and the environment calculates the normalized payoff $\widetilde{k}_{\mathbf{a}_q}$

Fig. 2. An illustration of the proposed MARL based service caching algorithm.

Fig. 2 illustrates the training procedure of the proposed MARL based service caching algorithm. Specifically, assume Q samples $\{\mathbf{H}(q), \mathbf{Y}(q)\}_{q=1}^{Q}$ are collected in the past and are used to construct a virtual environment with which all the agents interact. Under the qth sample $(\mathbf{H}(q), \mathbf{Y}(q))$, each learning agent selects an action a_i^q from its action set based on the action selection probability vector $\boldsymbol{\pi}_i^q = [\pi_{i,a_{i,1}}^q, \pi_{i,a_{i,2}}^q]$, where $\pi_{i,a_{i,1}}^q$ and $\pi_{i,a_{i,2}}^q$ represent the probability that agent i chooses action $a_{i,1}$ and action $a_{i,2}$, respectively. Once all the agents determine their actions, the network-wide service caching matrix $\mathbf{C}$ is generated. With the known $\mathbf{C}$, the virtual environment feeds back a common reward to all the agents, which is given below.

$$k_{\mathbf{a}_q} = \kappa\alpha \cdot t\left(\mathbf{H}(q), \mathbf{Y}(q), \mathbf{a}_q\right) + (1-\kappa)\beta \cdot \sum_{f\in\mathcal{F}} l_f(\mathbf{a}_q), \tag{24}$$

where $\mathbf{a}_q = [a_1^q, a_2^q, \ldots, a_{FS}^q]$ is the action profile of all the agents selected under $(\mathbf{H}(q), \mathbf{Y}(q))$. Note that the agents will implicitly cooperate to improve the effectiveness of network-wide service caching by learning from a common reward.

As per [51], the reward should be within [0,1]. Thus, $k_{\mathbf{a}_q}$ is further normalized as follows.

$$\widetilde{k}_{\mathbf{a}_q} = \begin{cases} \frac{\Theta - k_{\mathbf{a}_q}}{\Theta}, & \text{if } k_{\mathbf{a}_q} \le \Theta, \\ 0, & \text{if } k_{\mathbf{a}_q} > \Theta, \end{cases} \tag{25}$$

where Θ is a threshold with a positive constant value. With $\widetilde{k}_{\mathbf{a}_q}$, each agent i updates its action selection probability distribution using the following equations.

$$\pi_{i,a_{i,j}}^{q+1} = \pi_{i,a_{i,j}}^q + \xi\widetilde{k}_{\mathbf{a}_q}(1 - \pi_{i,a_{i,j}}^q), \quad \text{if } a_i^q = a_{i,j},$$
$$\pi_{i,a_{i,j}}^{q+1} = \pi_{i,a_{i,j}}^q - \xi\widetilde{k}_{\mathbf{a}_q}\pi_{i,a_{i,j}}^q, \quad \text{if } a_i^q \ne a_{i,j}, \tag{26}$$

Authorized licensed use limited to: Auburn University. Downloaded on August 24,2023 at 05:49:03 UTC from IEEE Xplore. Restrictions apply.

Algorithm 2: MARL Based Service Caching Algorithm.

```
1: The cloud server initializes an environment based on
   history data $\{\mathbf{H}(q), \mathbf{Y}(q)\}_{q=1}^{Q}$ and creates $F \times S$ SLA
   agents, each setting its initial action selection
   probabilities to 0.5;
2: for data sample $q = 1 : Q$ do
3:    Each agent selects an action $a_i^q$ based on $\boldsymbol{\pi}_i^q$;
4:    Under the caching matrix derived from $\mathbf{a}_q$,
      Algorithm 1 is executed;
5:    Calculate the normalized payoff $\widetilde{k}_{\mathbf{a}_q}$ according
      to (24) and (25);
6:    Each agent updates its action selection probabilities
      according to (26);
7: end for
8: Derive the final caching matrix $\mathbf{C}$ based on
   $\{\boldsymbol{\pi}_i^{Q+1}\}_{i=1}^{FS}$, where $c_{f,s} = 1$ if $i = (f-1)S + s$ and
   $\pi_{i,a_{i,1}}^{Q+1} > \pi_{i,a_{i,2}}^{Q+1}$;
```

where $0 < \xi < 1$ is the learning rate. Moreover, for $q \geq 1$, we have

$$\pi_{i,a_{i,1}}^{q+1} + \pi_{i,a_{i,2}}^{q+1} = \left(\pi_{i,a_{i,1}}^{q} + \pi_{i,a_{i,2}}^{q}\right)\left(1 - \xi\widetilde{k}_{\mathbf{a}_q}\right) + \xi\widetilde{k}_{\mathbf{a}_q}. \tag{27}$$

Then, it can be seen that if $\pi_{i,a_{i,1}}^{1} + \pi_{i,a_{i,2}}^{1} = 1$, we have $\pi_{i,a_{i,1}}^{2} + \pi_{i,a_{i,2}}^{2} = 1 \Rightarrow \pi_{i,a_{i,1}}^{3} + \pi_{i,a_{i,2}}^{3} = 1\ldots$, and hence $\pi_{i,a_{i,1}}^{q+1} + \pi_{i,a_{i,2}}^{q+1} = 1$ is satisfied.

Algorithm 2 describes the procedure of the proposed service caching algorithm. We next analyze its convergence, optimality, and complexity.

1) Convergence: According to Theorems 3.1 and 4.1 in [51], the following lemma holds.

Lemma 1: Since all agents receive the same reward $\widetilde{k}_{\mathbf{a}_q} \in [0, 1]$, Algorithm 2 converges when the learning rate ξ is sufficiently small, and $\mathbf{H}(q)$ and $\mathbf{Y}(q)$ both follow independently and identically distributions. Meanwhile, the converged caching strategy $\boldsymbol{\pi}_i^*$ satisfies

$$\begin{aligned}&\sum_{\mathbf{a}} \mathbb{E}_{\mathbf{H},\mathbf{Y}}\left[\tilde{k}_{\mathbf{a}}(\mathbf{H},\mathbf{Y},\Theta)\right]\prod \pi_{i,a_i}^* \geq \\ &\sum_{\mathbf{a}} \mathbb{E}_{\mathbf{H},\mathbf{Y}}\left[\tilde{k}_{\mathbf{a}}(\mathbf{H},\mathbf{Y},\Theta)\right]\pi_{i,a_i}\prod_{i'\neq i}\pi_{i',a_{i'}}^*, \ \forall i,\ \forall \boldsymbol{\pi}_i.\end{aligned} \tag{28}$$

As for convergence speed, it depends on learning step size ξ in equation (26). A larger ξ means a larger increment in the probability of selecting a certain action and hence a faster convergence speed can be achieved as verified in Fig. 7 in simulation part.

2) Optimality: We have the following theorem on the optimality of the proposed algorithm.

Theorem 2: When the action selection probabilities of all the agents converge to pure strategies with the corresponding action profile denoted by $\mathbf{a}^*$, the service caching matrix $\mathbf{C}$ decided by $\mathbf{a}^*$ is a local optimal solution to Problem (23).

Proof: Since (28) holds for any strategy of agent i, it must hold for any pure strategy of agent i as a special case, which means that agent i selects a deterministic action. On the basis of this fact, when the action selection strategy of each agent converges to a pure strategy, we have the following inequality according to (28).

$$\begin{aligned}&\mathbb{E}_{\mathbf{H},\mathbf{Y}}\left[\tilde{k}_{\mathbf{a}^*}(\mathbf{H},\mathbf{Y},\Theta)\right] \geq \\ &\mathbb{E}_{\mathbf{H},\mathbf{Y}}\left[\tilde{k}_{(a_i,\mathbf{a}_{-i}^*)}(\mathbf{H},\mathbf{Y},\Theta)\right],\ \forall i,\ \forall a_i \in \mathcal{A}_i.\end{aligned} \tag{29}$$

According to (25), when parameter Θ is sufficiently large, we have

$$\widetilde{k}_{\mathbf{a}} = \frac{\Theta - k_{\mathbf{a}}}{\Theta}. \tag{30}$$

Then, the following inequality can be derived following (29) and (30).

$$\mathbb{E}_{\mathbf{H},\mathbf{Y}}\left[k_{\mathbf{a}^*}(\mathbf{H},\mathbf{Y})\right] \leq \mathbb{E}_{\mathbf{H},\mathbf{Y}}\left[k_{(a_i,\mathbf{a}_{-i}^*)}(\mathbf{H},\mathbf{Y})\right],\ \forall i, \forall a_i \in \mathcal{A}_i. \tag{31}$$

Substituting (24) into $\mathbb{E}_{\mathbf{H},\mathbf{Y}}[k_{\mathbf{a}^*}(\mathbf{H},\mathbf{Y})]$, we have

$$\begin{aligned}&\mathbb{E}_{\mathbf{H},\mathbf{Y}}\left[k_{\mathbf{a}^*}(\mathbf{H},\mathbf{Y})\right] \\ =\ &\mathbb{E}_{\mathbf{H},\mathbf{Y}}\left[\kappa\alpha\cdot t\left(\mathbf{H},\mathbf{Y},\mathbf{a}^*\right) + (1-\kappa)\beta\cdot\sum_{f\in\mathcal{F}} l_f(\mathbf{a}^*)\right] \\ =\ &\kappa\alpha\cdot\mathbb{E}_{\mathbf{H},\mathbf{Y}}\left[t\left(\mathbf{H},\mathbf{Y},\mathbf{C}^*\right)\right] + (1-\kappa)\beta\cdot\sum_{f\in\mathcal{F}} l_f(\mathbf{C}^*).\end{aligned} \tag{32}$$

It can be seen that the right-hand-side of (32) is the same as the optimization objective in Problem (23). Based on (31) and (32), the objective value of Problem (23) cannot be further decreased by unilaterally changing the action of any agent. In other words, changing the value of any single element $c_{f,s}$ in $\mathbf{C}^*$ decided by $\mathbf{a}^*$ will not be beneficial. Thus Algorithm 2 converges to a local optimal solution to Problem (23). ■

3) Complexity: In each iteration of Algorithm 2, each agent first generates a random number to select an action based on the action selection probability vector, with complexity $\mathcal{O}(1)$. Then, the radio and computation resource allocation algorithm is executed and the complexity is $\mathcal{O}(U\sum_{f\in\mathcal{F}\cup\{0\}}|\mathcal{B}_f|D)$ as analyzed before. Based on the received reward, each agent updates the action selection probability with fixed number of operations and hence the complexity is also $\mathcal{O}(1)$. Since the operations of each learning agent can be executed in parallel in the cloud computing environment of F-RANs, the total complexity incurred by each iteration of Algorithm 2 is therefore $\mathcal{O}(U\sum_{f\in\mathcal{F}\cup\{0\}}|\mathcal{B}_f|D)$.

V. Simulation Results and Analysis

In this section, our simulation results are provided to evaluate the performance of the proposed algorithms. In the simulations, an F-RAN scenario with 3 F-APs, 5 RRHs, and 10 users is considered, which are distributed in a circular area of radius 100 m, as shown in Fig. 3. The number of services potentially requested by users is 10. All the users are assumed to request services following an identical Zipf distribution with parameter 1. Note that our proposal also works when users have differentiated service request distributions. The path loss exponent of the wireless channels is set to 3 and the small-scale channel fading is modeled by $\mathcal{CN}(0, 1)$. Other main simulation parameters are listed in Table II.

Authorized licensed use limited to: Auburn University. Downloaded on August 24,2023 at 05:49:03 UTC from IEEE Xplore. Restrictions apply.

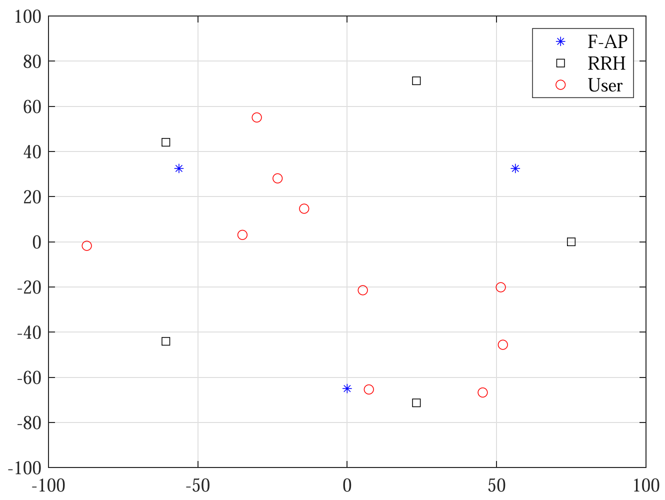

Fig. 3. Simulation scenario of an F-RAN with 3 F-APs, 5 RRHs, and 10 users in a circular area of 100 m radius.

TABLE II
SIMULATION PARAMETER SETTING

Parameters	**Value**
Transmit power of each user	20 dBm
Noise power spectral density	-174 dBm/Hz
Number of subchannels for each F-AP	2
Bandwidth of each subchannel	180 kHz
Bandwidth allocated to the RRHs	1800 kHz
Input data size of service s	$[1.6-2.4]$ Mbits
Computation workload of service s	$[0.15-0.25]\times 10^9$ CPU cycles
CPU speed of F-AP f	5×10^9 CPU cycles/s
Reserved caching capacity of F-AP f	1 GB
Required storage to cache service s	[0.15-0.4] GB

A. Radio and Computation Resource Allocation Algorithm

We first investigate the performance of the radio and computation resource optimization algorithm (Algorithm 1) under a fixed service caching matrix. More specifically, each F-AP caches all the services that are potentially requested by users. To verify the superiority of our proposal, the proposed algorithm is compared with the following baseline schemes.

- *Random allocation scheme with NOMA:* In this scheme, each user randomly selects an F-AP-subchannel resource pair, or operates in the cloud mode, to offload its computation task of the requested service.
- *Proposed scheme with OMA:* In this scheme, if two users are allocated with the same subchannel of an F-AP, these two users offload their computation tasks in turn, i.e., uploading their input data using orthogonal time-domain resources.
- *GA scheme with NOMA:* In this scheme, GA is applied to solve the radio resource allocation problem (19), where NOMA is used to allow two users to share a subchannel.

Note that random resource allocation and genetic algorithm based resource allocation have also been adopted in [52] and [15], respectively.

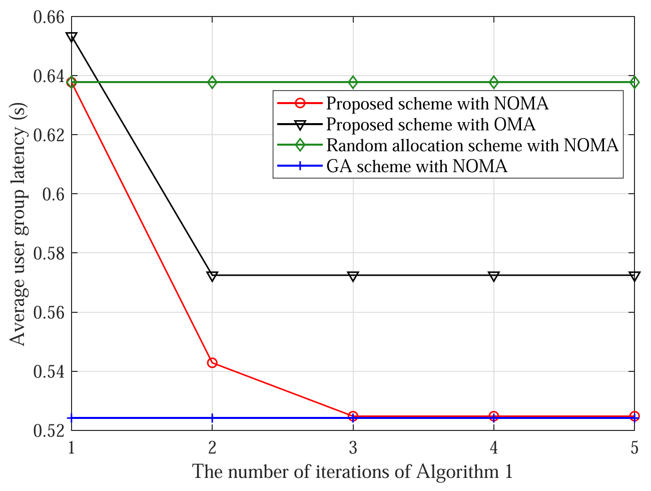

Fig. 4. The convergence performance of Algorithm 1.

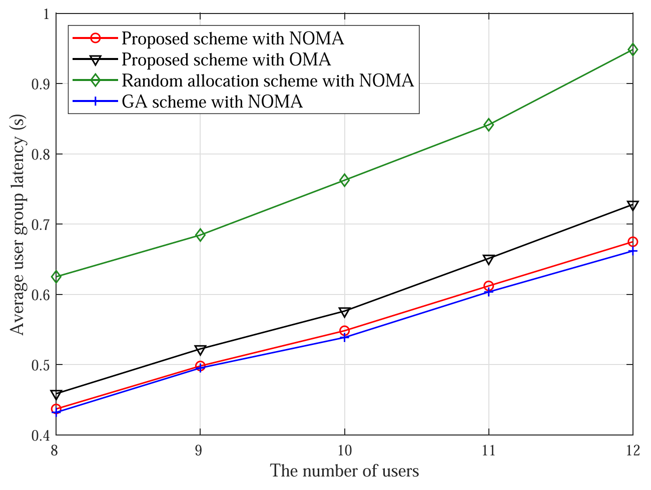

Fig. 5. Average user group latency under different schemes versus the number of users.

Fig. 4 shows the convergence of the proposed radio and computation resource allocation algorithm. In addition to its fast convergence, the proposed algorithm achieves a good performance very close to that of the GA with NOMA scheme, in terms of average user group latency given by (15). In Fig. 5, the metric (15) is evaluated more rigorously under different schemes by varying the number of users, where every result is the average of 200 independent simulations with fixed locations of only F-APs and RRHs as per Fig. 3. First, it can be observed that (15) increases with the number of users under all the four schemes. This is because the number of users that can be served by each F-AP is limited and hence more users have to offload their tasks via the RRHs as U is increased, leading to larger latency. Second, compared with the proposed scheme with OMA, the proposed scheme with NOMA achieves a lower average user group latency, because NOMA allows users to simultaneously transmit on the same subchannel. Moreover, the proposed scheme's performance is very close to that of the GA scheme, but with a much shorter execution time. Especially, when the number of users is

Authorized licensed use limited to: Auburn University. Downloaded on August 24,2023 at 05:49:03 UTC from IEEE Xplore. Restrictions apply.

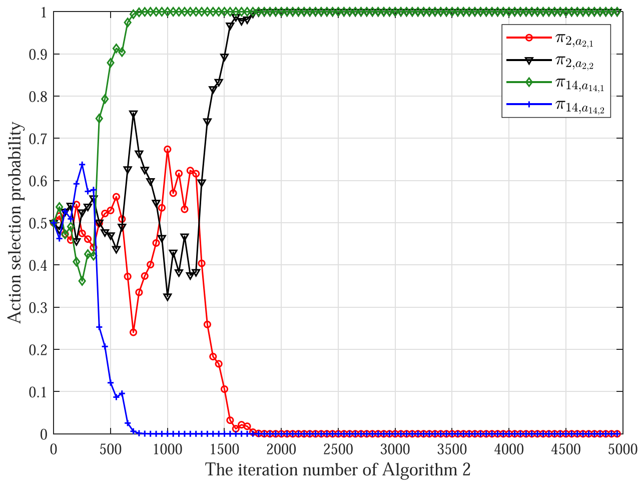

Fig. 6. The evolution of the action selection probabilities of the 2nd agent and the 14th agent.

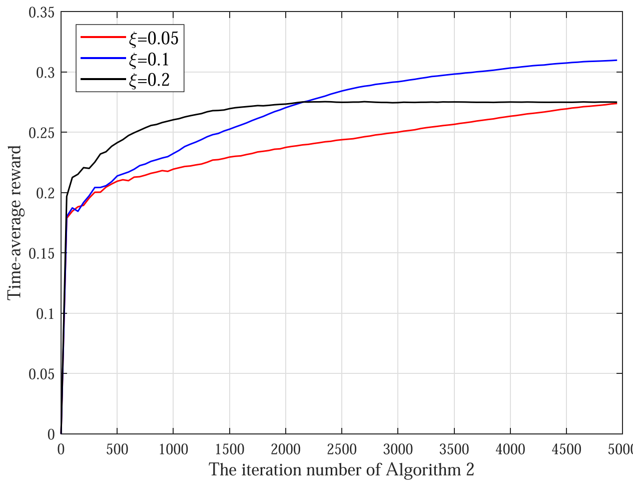

Fig. 7. The impact of learning rate ξ on learning performance.

12, the GA scheme only reduces the latency performance by around 2% over the proposed scheme, but its execution time is 6.52 s that is much longer than the time 0.56 s consumed by the proposed scheme.

B. Service Caching Algorithm

In this section, the performance of the proposed service caching algorithm (Algorithm 2) is examined under the same topology given by Fig. 3. The cost α of unit expected latency and the cost β of expanding F-AP's cache by one unit are set to 2 and 0.2, respectively. If not otherwise specified, the weighting factor κ in (16) is set to 0.5. The learning rate ξ is set to 0.1 and the normalization related parameter Θ is set to 1. The number of samples of $(\mathbf{H}, \mathbf{Y})$ used for constructing the training environment is $Q = 5 \times 10^3$. Note that the samples are independent and identically distributed, which are generated based on the channel model and the service request distribution mentioned above.

Fig. 6 illustrates the evolution of the action selection probability distributions for the 2nd agent and the 14th agent. With the progress of the training procedure, the probability of choosing one action converges to 1 and that of choosing the other action converges to 0 for both agents, which indicates that pure strategies have been achieved in our setting. Fig. 7 presents the impact of the learning rate parameter ξ on learning performance. Define $\sum_{q'=1}^{q} \widetilde{k}_{\mathbf{a}_{q'}}/q$ as the time-average reward during the learning process. It can be seen that the convergence time and learning performance is well balanced when $\xi = 0.1$, which is taken as the learning rate used in the remaining simulations.

Next, the proposed MARL based service caching scheme, named as Caching scheme 1, is compared with the following baseline schemes.

- *Caching scheme 2:* This scheme originates from [37], where a multi-agent multi-armed RL method is adopted for content caching. To adapt this proposal to our problem, the reward of each learning agent is now given by

$$k_{\mathbf{a}_q} = -\kappa\alpha \cdot t\left(\mathbf{H}_q, \mathbf{Y}_q, \mathbf{a}_q\right). \tag{33}$$

Note that caching cost is not involved in the reward because caching decisions in this scheme are made under the strict storage limit of each F-AP.
- *Caching scheme 3:* In this scheme, F-APs cache no service and all users operate in the cloud service mode.
- *Caching scheme 4:* In this scheme, each F-AP caches all the services that are potentially requested by users.
- *Caching scheme 5:* In this scheme, each F-AP caches the most popular services that rank in the top $\Delta\%$ among all the services. In the simulations, we have searched $\Delta \in \{10, 20, \ldots, 90\}$ and chosen the optimal value 40.
- *Caching scheme 6:* In this scheme, we assume that each F-AP knows the service request probability and each F-AP first checks the service with the highest probability to decide whether to cache with probability of 0.5. Then, each F-AP continues to check the next one with lower request probability until the cache size is full.
- *Caching scheme 7:* In this scheme, GA based service caching is utilized, where each individual corresponds to a service caching matrix and the fitness value of an individual is defined as the reciprocal of the optimization objective value of Problem (16). Since the objective value includes the expected value of t, we take the average of t under N_{sam} samples of $\{\mathbf{H}, \mathbf{Y}\}$ to approximate its expectation. Thus, GA based service caching needs to invoke Algorithm 1 for $N_{gen}N_{pop}N_{sam}$ times, where N_{gen}, N_{pop}, and N_{sam} represent the maximum number of generations, the population size, and the number of samples of $\{\mathbf{H}, \mathbf{Y}\}$, respectively. In simulation, $N_{pop} = 20$, $N_{gen} = 200$, and $N_{sam} = 20$. Furthermore, the number of elite individuals, crossover probability, and mutation probability are set to 2, 0.6 and 0.05, respectively.
- *Caching scheme 8:* In this scheme, the service cloud determines the caching matrix by solving the following

Authorized licensed use limited to: Auburn University. Downloaded on August 24,2023 at 05:49:03 UTC from IEEE Xplore. Restrictions apply.

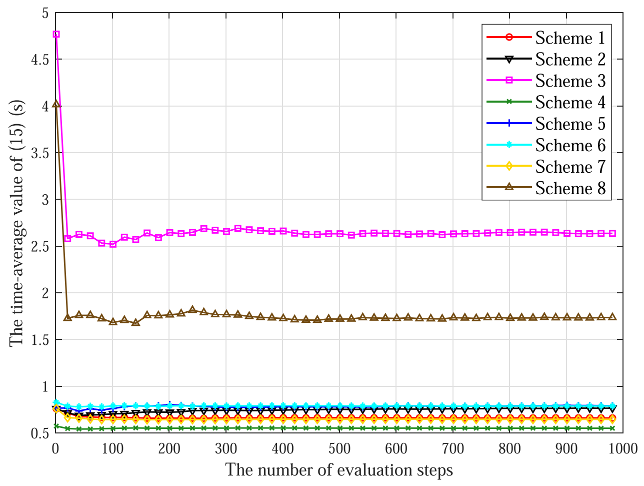

Fig. 8. The evolution of the time average value of (15) under different caching schemes.

optimization problem for each F-AP f with integer relaxation and rounding technique while computation and communication resource are allocated with Algorithm 1.

$$\max_{c_{f,s}} \sum_{s\in\mathcal{S}} c_{f,s}$$

$$\text{s.t. } (1) \sum_{s\in\mathcal{S}} \epsilon_s c_{f,s} \le V_f,$$

$$(2)\ c_{f,s} \in \{0,1\}, \forall s \in \mathcal{S}.$$

The rational behind such formulation is to make each F-AP cache as many services as possible under given caching space.

Note that no caching scheme has been adopted in [28] and most popular caching scheme has been adopted in [25]. Moreover, genetic algorithm based caching has been utilized in [24].

In *Caching scheme 1* and *Caching scheme 2*, service caching matrices are obtained based on the training environment constructed by Q samples of $\{\mathbf{H}, \mathbf{Y}\}$ as mentioned before. To compare the performance of various caching schemes, another 1000 samples of $\{\mathbf{H}, \mathbf{Y}\}$ are generated following the same distribution as that used to generate the previous Q samples. In the evaluation, the caching matrix derived from each scheme remains fixed and only the short-term radio and computation resource allocation is conducted under each sample of $\{\mathbf{H}, \mathbf{Y}\}$. Fig. 8 shows the evolution of the time-average value of (15) calculated as $\sum_{q'=1}^{q} t_{q'}/q$, and its converged value is taken to approximate the expected value of t in (16) under a given caching matrix. In Fig. 9, the caching cost of different caching schemes is shown. Since *Caching scheme 2* and *Caching scheme 6* can strictly meet the caching space constraints of F-APs, their caching costs are 0. *Caching scheme 3* also has 0 caching cost, since all the tasks are executed in the cloud, which, however, incurs high latency cost. In contrast, the proposed *Caching scheme 1* achieves zero caching cost without incurring much latency.

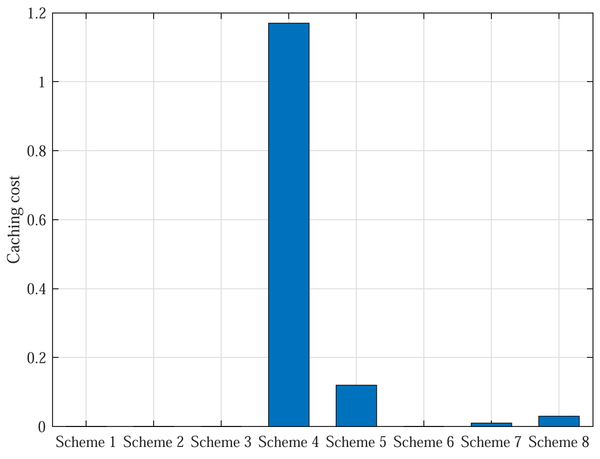

Fig. 9. The caching cost under different caching schemes.

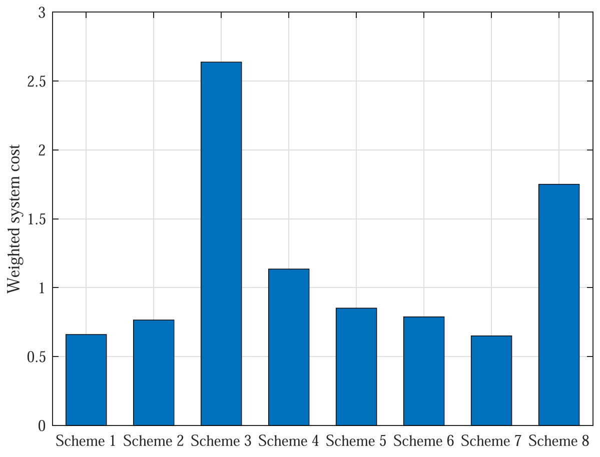

Fig. 10. The objective value of (16) achieved by different caching schemes.

Fig. 10 presents the objective value of (16) achieved by the eight schemes. It can be seen that our proposed Scheme 1 improves the system cost by 13.8% when compared to the state-of-the-art Scheme 2. Meanwhile, Scheme 2 needs to additionally solve a 0-1 Knapsack problem in each iteration, and it also has to explicitly know the parameter of Zipf distribution used to model global service popularity, which are not required in our scheme. Moreover, it is observed that caching all the services leads to poor performance due to the large caching cost, and the proposed scheme outperforms Schemes 3, 4, 5, 6 and 8. This is because the caching strategies of the agents can well adapt to the dynamics in the network environment and align well with the system objective by learning from the designed reward. In addition, although Scheme 7 achieves a competitive performance relative to our proposal, it needs to execute Algorithm 1 of short-term resource allocation for $N_{gen}N_{pop}N_{sam} = 8 * 10^4$ times, but our proposed MARL based caching needs to execute Algorithm 1 for only 2000 times before convergence according to Fig. 6, which is a great reduction in time complexity.

In Fig. 11, the impact of parameter Θ in (25) on the system cost is evaluated. As the value of Θ is increased, the system cost decreases first and then increases. This is because the normalized

Authorized licensed use limited to: Auburn University. Downloaded on August 24,2023 at 05:49:03 UTC from IEEE Xplore. Restrictions apply.

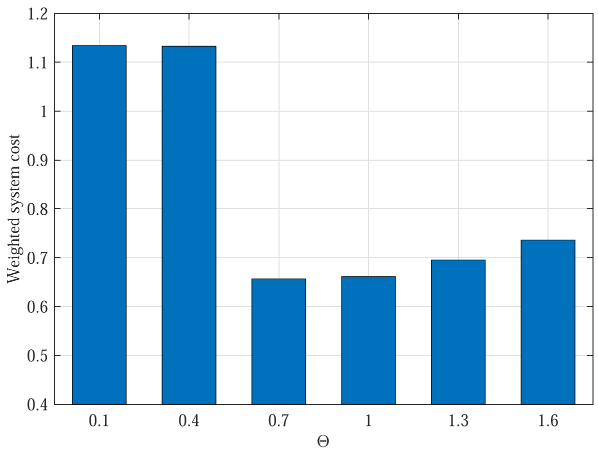

Fig. 11. The influence of parameter Θ on the system cost.

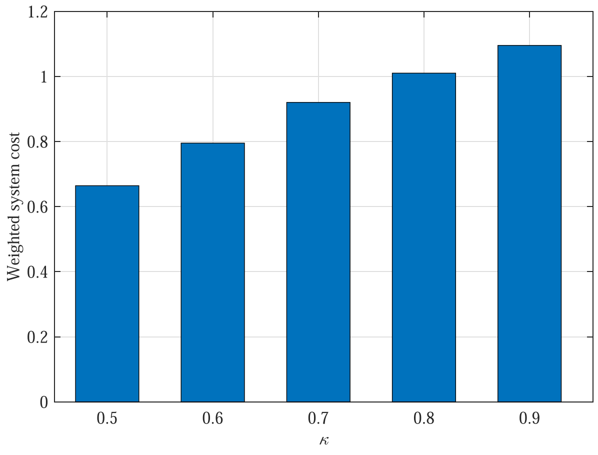

Fig. 12. The influence of the weight κ in (16) on the system cost.

reward $\widetilde{k}_{\mathbf{a}_q}$ has a high chance to become zero when Θ is set to a small value, and hence the much useful information of $k_{\mathbf{a}_q}$ is lost, which makes the strategy learning of agents ineffective. On the other hand, as Θ is further increased, the values of normalized rewards by taking different actions become closer and hence it is difficult for agents to differentiate good actions from bad ones.

In Fig. 12, the impact of the weight κ in (16) is illustrated. It can be seen that a larger $\kappa \geq 0.5$ results in a larger system cost. To explain this phenomenon, Fig. 13 is presented to demonstrate the latency cost and caching cost under varying κ. With a larger κ, learning agents care more about the latency cost while the caching cost plays a less vital role in the system cost. Hence the latency cost gradually decreases while the caching cost goes higher. Particularly, the degree of variation of the latency cost is much smaller than that of the caching cost. This is why a larger κ leads to a higher weighted system cost. The differentiated variation degrees of the two costs are due to the fact that the learning agents have to choose to cache much more services that are rarely requested by users to further reduce the latency cost.

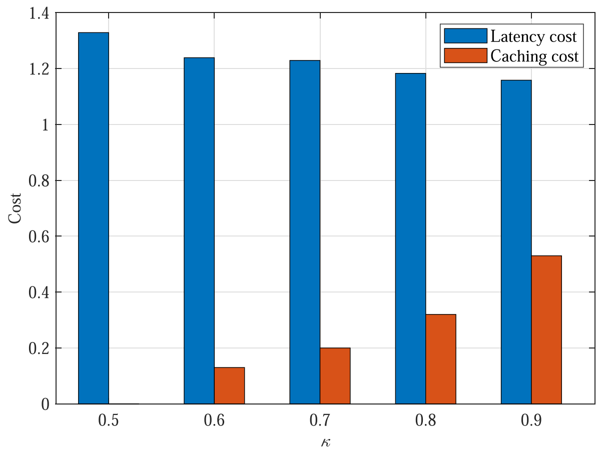

Fig. 13. The influence of the weight κ on the latency cost and the caching cost defined in (16).

VI. Conclusion

In this paper, we studied the multi-dimensional resource management problem in the context of fog radio access networks for service task offloading, which was featured by a two-timescale formulation. Aiming at minimizing the weighted sum of the expected latency cost and the caching cost, a coalition formation based algorithm was proposed to allocate radio and computation resource on a smaller timescale for given service caching, while a multi-agent reinforcement learning based service caching optimization algorithm was developed to decide service placement at the F-APs on a larger timescale. In addition, convergence, optimality, and complexity of both proposed algorithms were analyzed. Through numerical results, the effectiveness of the proposed algorithms was verified through comparison with several baseline schemes, and the impacts of key parameters on the system performance were demonstrated.

References

[1] F. Wang, J. Xu, and Z. Ding, "Multi-antenna NOMA for computation offloading in multiuser mobile edge computing systems," *IEEE Trans. Commun.*, vol. 67, no. 3, pp. 2450–2463, Mar. 2019.
[2] Y. Wu, L. P. Qian, K. Ni, C. Zhang, and X. Shen, "Delay-minimization nonorthogonal multiple access enabled multi-user mobile edge computation offloading," *IEEE J. Sel. Topics Signal Process.*, vol. 13, no. 3, pp. 392–407, Jun. 2019.
[3] J. Dai, Z. Zhang, S. Mao, and D. Liu, "A view synthesis-based 360° VR caching system over MEC-enabled C-RAN," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 10, pp. 3843–3855, Oct. 2020.
[4] T. K. Rodrigues, K. Suto, H. Nishiyama, J. Liu, and N. Kato, "Machine learning meets computation and communication control in evolving edge and cloud: Challenges and future perspective," *IEEE Commun. Surv. Tut.*, vol. 22, no. 1, pp. 38–67, Jan.–Mar. 2020.
[5] J. Wang, L. Zhao, J. Liu, and N. Kato, "Smart resource allocation for mobile edge computing: A deep reinforcement learning approach," *IEEE Trans. Emerg. Topics Comput.*, vol. 9, no. 3, pp. 1529–1541, Jul.–Sep. 2021.
[6] K. Wang, W. Zhou, and S. Mao, "On joint BBU/RRH resource allocation in heterogeneous Cloud-RANs," *IEEE Internet Things J.*, vol. 4, no. 3, pp. 749–759, Jun. 2017.
[7] R. Dong, C. She, W. Hardjawana, Y. Li, and B. Vucetic, "Deep learning for hybrid 5G services in mobile edge computing systems: Learn from a digital twin," *IEEE Trans. Wireless Commun.*, vol. 18, no. 10, pp. 4692–4707, Oct. 2019.

Authorized licensed use limited to: Auburn University. Downloaded on August 24,2023 at 05:49:03 UTC from IEEE Xplore. Restrictions apply.

[8] Y. Sun, M. Peng, and S. Mao, "Deep reinforcement learning based mode selection and resource management for green fog radio access networks," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1960–1971, Apr. 2019.

[9] Z. Zhu, T. Liu, Y. Yang, and X. Luo, "BLOT: Bandit learning-based offloading of tasks in fog-enabled networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 12, pp. 2636–2649, Dec. 2019.

[10] T. Q. Dinh, J. Tang, Q. D. La, and T. Q. S. Quek, "Offloading in mobile edge computing: Task allocation and computational frequency scaling," *IEEE Trans. Commun.*, vol. 65, no. 8, pp. 3571–3584, Aug. 2017.

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

[12] Y. Pan, H. Jiang, H. Zhu, and J. Wang, "Latency minimization for task offloading in hierarchical fog-computing C-RAN networks," in *Proc. Int. Conf. Commun.*, Dublin, Ireland, 2020, pp. 1–6.

[13] Q. Pham, H. T. Nguyen, Z. Han, and W. Hwang, "Coalitional games for computation offloading in NOMA-enabled multi-access edge computing," *IEEE Trans. Veh. Technol.*, vol. 67, no. 12, pp. 12137–12151, Dec. 2018.

[14] Y. Liu, F. R. Yu, X. Li, H. Ji, and V. C. M. Leung, "Distributed resource allocation and computation offloading in fog and cloud networks with non-orthogonal multiple access," *IEEE Trans. Veh. Technol.*, vol. 67, no. 12, pp. 12137–12151, Dec. 2018.

[15] H. Li, H. Xu, C. Zhou, X. Lü, and Z. Han, "Joint optimization strategy of computation offloading and resource allocation in multi-access edge computing environment," *IEEE Trans. Veh. Technol.*, vol. 69, no. 9, pp. 10214–10226, Sep. 2020.

[16] L. Chen, J. Xu, S. Ren, and P. Zhou, "Spatio-temporal edge service placement: A bandit learning approach," *IEEE Trans. Wireless Commun.*, vol. 17, no. 12, pp. 8388–8401, Dec. 2018.

[17] X. Q. Pham, T. D. Nguyen, V. Nguyen, and E. N. Huh, "Joint service caching and task offloading in multi-access edge computing: A QoE-based utility optimization approach," *IEEE Commun. Lett.*, vol. 25, no. 3, pp. 965–969, Mar. 2021.

[18] S. Bi, L. Huang, and Y. 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.

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

[20] Y. Qiu, H. Zhang, K. Long, and M. Guizani, "Subchannel assignment and power allocation for time-varying fog radio access network with NOMA," *IEEE Trans. Wireless Commun.*, vol. 20, no. 6, pp. 3685–3697, Jun. 2021.

[21] I. Randrianantenaina, M. Kaneko, H. Dahrouj, H. ElSawy, and M. S. Alouini, "Interference management in NOMA-based fog-radio access networks via scheduling and power allocation," *IEEE Trans. Commun.*, vol. 68, no. 8, pp. 5056–5071, Aug. 2020.

[22] Z. Yan, M. Peng, and M. Daneshmand, "Cost-aware resource allocation for optimization of energy efficiency in fog radio access networks," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 11, pp. 2581–2590, Nov. 2018.

[23] H. T. Nguyen, H. D. Tuan, T. Q. Duong, H. V. Poor, and W. Hwang, "Collaborative multicast beamforming for content delivery by cache-enabled ultra dense networks," *IEEE Trans. Commun.*, vol. 67, no. 5, pp. 3396–3406, May 2019.

[24] P. Lin, Q. Song, and A. Jamalipour, "Multidimensional cooperative caching in CoMP-integrated ultra-dense cellular networks," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, pp. 1977–1989, Mar. 2020.

[25] Y. Li, M. Jiang, Q. Zhang, and J. Qin, "Cache content placement optimization in non-orthogonal multiple access networks," *IEEE Trans. Commun.*, vol. 68, no. 7, pp. 4580–4591, Jul. 2020.

[26] M. Ma and V. W. S. Wong, "Age of information driven cache content update scheduling for dynamic contents in heterogeneous networks," *IEEE Trans. Wireless Commun.*, vol. 19, no. 12, pp. 8427–8441, Dec. 2020.

[27] H. Kim, J. Park, M. Bennis, S. L. Kim, and M. Debbah, "Mean-field game theoretic edge caching in ultra-dense networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 1, pp. 935–947, Jan. 2020.

[28] X. Yang, Y. Fu, W. Wen, T. Q. S. Quek, and Z. Fei, "Mixed-timescale caching and beamforming in content recommendation aware fog-RAN: A latency perspective," *IEEE Trans. Commun.*, vol. 69, no. 4, pp. 2427–2440, Apr. 2021.

[29] X. Wu, Q. Li, X. Li, V. C. M. Leung, and P. C. Ching, "Joint long-term cache updating and short-term content delivery in cloud-based small cell networks," *IEEE Trans. Commun.*, vol. 68, no. 5, pp. 3173–3186, May 2020.

[30] M. Zhao, Y. Cai, M. Zhao, B. Champagne, and T. A. Tsiftsis, "Improving caching efficiency in content-aware C-RAN-based cooperative beamforming: A joint design approach," *IEEE Trans. Wireless Commun.*, vol. 19, no. 6, pp. 4125–4140, Jun. 2020.

[31] B. Dai, Y. Liu, and W. Yu, "Optimized base-station cache allocation for cloud radio access network with multicast backhaul," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 8, pp. 1737–1750, Aug. 2018.

[32] J. Wang, C. Jiang, H. Zhang, Y. Ren, K. -C. Chen, and L. Hanzo, "Thirty years of machine learning: The road to pareto-optimal wireless networks," *IEEE Commun. Surv. Tuts.*, vol. 22, no. 3, pp. 1472–1514, Jul.–Sep. 2020.

[33] M. Chen, U. Challita, W. Saad, C. Yin, and M. Debbah, "Artificial neural networks-based machine learning for wireless networks: A tutorial," *IEEE Commun. Surv. Tuts.*, vol. 21, no. 4, pp. 3039–3071, Oct.–Dec. 2019.

[34] Z. Chang, L. Lei, Z. Zhou, S. Mao, and T. Ristaniemi, "Learn to cache: Machine learning for network edge caching in the Big Data era," *IEEE Wireless Commun. Mag.*, vol. 25, no. 3, pp. 28–35, Jun. 2018.

[35] C. Zhong, M. C. Gursoy, and S. Velipasalar, "Deep reinforcement learning-based edge caching in wireless networks," *IEEE Trans. Cogn. Commun. Netw.*, vol. 6, no. 1, pp. 48–61, Mar. 2020.

[36] A. Sadeghi, F. Sheikholeslami, and G. B. Giannakis, "Optimal and scalable caching for 5G using reinforcement learning of space-time popularities," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 180–190, Feb. 2018.

[37] W. Jiang, G. Feng, S. Qin, T. S. P. Yum, and G. Cao, "Multi-agent reinforcement learning for efficient content caching in mobile D2D networks," *IEEE Trans. Wireless Commun.*, vol. 18, no. 3, pp. 1610–1622, Mar. 2019.

[38] W. Chen, S. Zhao, R. Zhang, and L. Yang, "Generalized user grouping in NOMA based on overlapping coalition formation game," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 4, pp. 969–981, Apr. 2021.

[39] J. Ding, J. Cai, and C. Yi, "An improved coalition game approach for MIMO-NOMA clustering integrating beamforming and power allocation," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1672–1687, Feb. 2019.

[40] H. Zhang, Y. Qiu, K. Long, G. K. Karagiannidis, X. Wang, and A. Nallanathan, "Resource allocation in NOMA-based fog radio access networks," *IEEE Wireless Commun.*, vol. 25, no. 3, pp. 110–115, Jun. 2018.

[41] T. LeAnh et al., "Matching theory for distributed user association and resource allocation in cognitive femtocell networks," *IEEE Trans. Veh. Tech.*, vol. 66, no. 9, pp. 8413–8428, Sep. 2017.

[42] J. Tang, W. P. Tay, T. Q. S. Quek, and B. Liang, "System cost minimization in cloud RAN with limited fronthaul capacity," *IEEE Trans. Wireless Commun.*, vol. 16, no. 5, pp. 3371–3384, May 2017.

[43] F. Tseng, M. Tsai, C. Tseng, Y. Yang, C. Liu, and L. Chou, "A lightweight autoscaling mechanism for fog computing in industrial applications," *IEEE Trans. Ind. Informat.*, vol. 14, no. 10, pp. 4529–4537, Oct. 2018.

[44] X. Yuan, M. Sun, and W. Lou, "A dynamic deep-learning-based virtual edge node placement scheme for edge cloud systems in mobile environment," *IEEE Trans. Cloud Comput.*, vol. 10, no. 2, pp. 1317–1328, Apr.–Jun. 2022.

[45] Z. Ding, P. Fan, and H. V. Poor, "Impact of non-orthogonal multiple access on the offloading of mobile edge computing," *IEEE Trans. Commun.*, vol. 67, no. 1, pp. 375–390, Jan. 2019.

[46] B. Liu, C. Liu, M. Peng, Y. Liu, and S. Yan, "Resource allocation for non-orthogonal multiple access-enabled fog radio access networks," *IEEE Trans. Wireless Commun.*, vol. 19, no. 6, pp. 3867–3878, Mar. 2020.

[47] C. Fan, Y. J. Zhang, and X. Yuan, "Dynamic nested clustering for parallel PHY-layer processing in cloud-RANs," *IEEE Trans. Wireless Commun.*, vol. 15, no. 3, pp. 1881–1894, Mar. 2016.

[48] J. Tang, T. Q. S. Quek, T. Chang, and B. Shim, "Systematic resource allocation in cloud RAN with caching as a service under two timescales," *IEEE Trans. Commun.*, vol. 67, no. 11, pp. 7755–7770, Nov. 2019.

[49] D. Wu, Y. Cai, R. Q. Hu, and Y. Qian, "Dynamic distributed resource sharing for mobile D2D communications," *IEEE Trans. Wireless Commun.*, vol. 14, no. 10, pp. 5417–5429, Oct. 2015.

[50] C. Fan, B. Li, C. Zhao, W. Guo, and Y. Liang, "Learning-based spectrum sharing and spatial reuse in mm-Wave ultra dense networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 6, pp. 4954–4968, Jun. 2018.

[51] P. S. Sastry, V. V. Phansalkar, and M. A. L. Thathachar, "Decentralized learning of nash equilibria in multi-person stochastic games with incomplete information," *IEEE Trans. Syst., Man, Cybern.*, vol. 24, no. 5, pp. 769–777, May 1994.

[52] H. Zhang, H. Zhang, W. Liu, K. Long, J. Dong, and V. C. M. Leung, "Energy efficient user clustering, hybrid precoding and power optimization in terahertz MIMO-NOMA systems," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 9, pp. 2074–2085, Sep. 2020.

Authorized licensed use limited to: Auburn University. Downloaded on August 24,2023 at 05:49:03 UTC from IEEE Xplore. Restrictions apply.