A Game-Theoretic Approach to Cache and Radio Resource Management in Fog Radio Access Networks

Yaohua Sun, Mugen Peng, *Senior Member, IEEE*, and Shiwen Mao, *Fellow, IEEE*

***Abstract*—Fog radio access networks (F-RANs) have been seen as promising paradigms to handle the stringent requirements in the 5G era by utilizing the cache and resource management capabilities of fog access points (FAPs). To achieve better system performance, cache resource and radio resource should be jointly optimized. However, fully centralized optimization can put heavy burden on the resource manager in the cloud. Faced with this issue, a hierarchical resource management architecture is adopted. Specifically, the resource manager in the upper layer maximizes a long-term utility by optimizing cache resource, which is adaptive to the statistics of channel gains and user content requests. In the lower layer, FAPs self-organize into multiple clusters to mitigate inter-FAP interference in each transmission interval given user content requests, channel gains and cache configuration. Under per-FAP fronthaul capacity constraints, interactions among FAPs are further modeled by a coalition formation game. Considering the coupling of FAPs' and the resource manager's strategies and the hierarchy of resource management, the joint cache and radio resource management is formulated as a Stackelberg game with the resource manager and FAPs being the leader and followers, respectively. To achieve Stackelberg equilibrium, a distributed coalition formation algorithm is first developed for FAPs to achieve a stable state. Since there is no closed form for the leader's objective and the leader's strategy is discontinuous, two model-free reinforcement learning (RL) algorithms are utilized, which can approach a global and a local optimal caching strategy, respectively, taking into account the cluster formation behavior of FAPs. Simulation results show that the proposed cluster formation scheme and multi-agent RL based caching scheme outperform the baselines.**

***Index Terms*—Fog radio access networks (F-RANs), cache optimization, clustering, game theory.**

Manuscript received January 11, 2019; revised June 18, 2019; accepted August 4, 2019. Date of publication August 13, 2019; date of current version October 18, 2019. This work was supported in part by the State Major Science and Technology Special Project under Grant 2017ZX03001025-006 and Grant 2018ZX03001023-005, in part by the National Natural Science Foundation of China under Grant 61831002 and Grant 61728101, in part by the National Program for Special Support of Eminent Professionals, and in part by the US NSF under ECCS-1923717. The review of this paper was coordinated by Prof. L. Guo. *(Corresponding author: Mugen Peng.)*

Y. Sun and M. Peng are with the State Key Laboratory of Networking and Switching Technology (SKL-NST), Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: sunyaohua@bupt.edu.cn; pmg@bupt.edu.cn).

S. 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.2019.2935098

I. Introduction

As an advanced architecture, fog radio access networks (F-RANs) can satisfy diverse and stringent communication demands in the 5G era by leveraging edge computing [1], [2], cloud computing and heterogenous networking like unmanned aerial vehicle (UAV) and device-to-device (D2D) communication [3], [4]. Particularly, fog access points (FAPs), which have edge caching and local radio resource management capabilities, can help alleviate the burden on fronthaul links and the cloud [5].

To boost the performance of F-RANs, resource allocation plays a key role, and some literatures have focused on radio resource allocation under fixed cache configuration. In [6]–[9], authors study a special downlink F-RAN that is a cloud radio access network (C-RAN) with each remote radio head (RRH) equipped with a cache. Specifically, content-centric beamforming is designed in [6], which aims at minimizing system cost including backhaul cost and transmission power cost. In [7], system energy consumption is optimized by RRH selection and load balancing, and the author in [8] also minimizes network power consumption via the joint consideration of RRH selection, data assignment, and multicast beamforming. Different from [6]–[8], a latency minimization problem is investigated in [9], which takes into account wireless transmission latency and backhaul downloading latency. When there exist different ways of communication, such as centralized C-RAN mode, FAP mode, and D2D mode, communication mode for each user equipment (UE) should be identified. To this end, joint mode selection and resource allocation is studied in [10], where a distributed reinforcement learning (RL) approach is utilized to alleviate the computing burden on the cloud for an uplink F-RAN.

Instead of optimizing radio resource in F-RANs, another branch of research pays attention to content placement/replacement in edge cache. In [11], [12], echo state networks are employed to estimate the content popularity, based on which contents to be cached at RRHs and UAVs are derived. In [13], the author utilizes extreme learning machine to acquire content popularity, and then a mixed-integer linear programming for content placement is formulated. In [14], learning content popularity is incorporated in multi-armed bandit learning [15], and content caching and sharing are optimized under the current popularity estimation. Without explicitly estimating content popularity, some researchers use RL to directly learn content replacement

policies. In [16], a distributed caching update scheme based on joint-utility-and-strategy-estimation RL is developed, allowing each small cell base station (SBS) to learn its own caching policy. To minimize system transmission cost, the author in [17] utilizes a distributed Q-learning algorithm, while a deep RL approach is adopted in [18] to improve the cache hit rate for a single BS. In addition, a socially aware distributed algorithm and a low complexity heuristic algorithm by exploiting user context information are developed for D2D networks in [19] and [20], respectively. Focusing on vehicular networks, a cooperative caching scheme based on mobility prediction is proposed in [21] and a dynamic content caching strategy is introduced for road side units in [22]. Furthermore, caching probabilities of SBSs are optimized to maximize successful downloading probability in [23].

Although the above works achieve good performance, radio resource and cache resource in F-RANs are optimized separately. However, as indicated in [6], [24]–[27], it is essential to jointly manage radio and cache resource. In [24], a joint content placement and UE association problem is investigated to minimize average downloading latency, and the problem formulation implicity assumes that content placement and UE association are optimized on the same timescale. Nevertheless, it is pointed out in [6] that cache resource is adjusted on a larger timescale compared to the adjustment of radio resource. Considering this fact, authors in [25]–[27] formulate joint cache resource and radio resource optimization as two-timescale optimization problems. Specifically, the long-term cache resource optimization problem relies on statistical information, such as that of content popularity and channel state, while the short-term radio resource optimization problem is related to the current cache resource configuration and instantaneous channel state information.

Actually, two-timescale resource optimization has also been studied in [28], [29] in a C-RAN scenario. The former addresses the joint optimization of resource in the baseband unit pool and beamforming design, while the latter tackles short-term precoding and long-term user-centric RRH clustering. In this paper, considering cache resource and radio resource are optimized on different timescales and to alleviate the burden on the cloud and fronthaul links, a hierarchical resource management architecture is adopted. Under this architecture, the resource manager in the cloud is responsible for adjusting cache resource, which aims to maximize the difference between the long-term system throughput and the cache deployment cost, while FAPs manage radio resource to optimize short-term system throughput.

To mitigate inter-FAP interference, FAPs participate in a coalition formation game to form cooperative clusters, within each of which UEs are served by time division based non-coherent joint transmission (JT). In addition, since the resource manager makes caching decision at first and FAPs perform transmission later on, the interaction between resource manager and FAPs is modeled as a Stackelberg game [30]. Moreover, different from [25]–[29], [31], the long-term utility optimization in our work features discontinuous variables and no closed form expression after sample average approximation, which incurs a big challenge. To overcome this issue, two different RL algorithms are utilized to approach a global or local optimal caching strategy of the resource manager, taking into account cluster formation behavior of FAPs. More formally, our contributions are summarized as follows.

- Considering cache resource and radio resource are optimized on different timescales and to alleviate the burden on the resource manager and fronthaul, a hierarchical resource management architecture is adopted for an F-RAN composed of multiple FAPs. In the lower layer, FAPs organize into multiple clusters to optimize short-term throughput under current channel state, user content requests and cache configuration, and those in the same cluster cooperatively serve UEs via time division based non-coherent JT. In the upper layer, the resource manager in the cloud intends to maximize a utility defined as the difference between long-term system throughput and the cost incurred by cache deployment, and the utility is related to not only the statistical information of channel state and user content request but also radio resource allocation in the lower layer.
- Given the sequential decision making between the resource manager and FAPs, the formulated two-timescale resource optimization problem is modeled as a Stackelberg game, whose stable state, known as Stackelberg equilibrium (SE), is defined. To achieve SE, we first design a distributed cluster formation algorithm for FAPs to reach a stable state, which explicitly considers per-FAP fronthaul capacity constraints. As for the strategy of the resource manager, since the utility function has no closed form and its caching strategy is represented by 0, 1 variables, the analysis of optimal strategy is very challenging. Faced with this problem, a caching algorithm based on single-agent RL (SARL) is proposed. Further, a caching algorithm based on multi-agent RL (MARL) is utilized to overcome the curse of dimensionality.
- The convergence, optimality and complexity of the proposed cluster formation algorithm, SARL based caching algorithm and MARL based caching algorithm are rigorously analyzed. Particularly, the cluster formation result is proved to be Nash-stable, and RL based caching algorithms can approach an optimal or local optimal caching strategy of the resource manager. In addition, the effectiveness of the proposals is verified by simulations.

The remainder of this paper is organized as follows. Section II describes the system model. Section III formulates the concerned two-timescale resource optimization problem, which is further modeled as a Stackelberg game. In Section IV, a distributed cluster formation algorithm is designed for FAPs, and two caching algorithms based on RL are developed. Section V analyzes the properties of the proposals, in terms of convergence, optimality and complexity, and simulation results are presented in Section VI, followed by the conclusion in Section VII. For convenience, some important notations are listed in Table I.

II. System Model

In this section, the model of the downlink F-RAN studied is elaborated, including key components, transmission mode, and other basic assumptions.

TABLE I
SUMMARY OF NOTATIONS

$\mathcal{K}$	The set of FAPs
m_k	The UE associated with FAP k
Φ	The partition of FAPs
R_{m_k}	The data rate of UE m_k
$h_{k'm_k}$	The channel gain between FAP k' and UE m_k
p	The transmission power of each FAP
$\mathcal{F}$	The set of content segments potentially requested by UEs
$x_{m_k,f}$	A 0-1 indicator showing whether UE m_k requests content segment f
$c_{f,k}$	A 0-1 indicator representing whether content segment f is stored in the cache of FAP k
l_k	The number of missing content segments that can be downloaded by FAP k via fronthaul in each transmission interval
$\mathbf{H}$	The channel gain matrix of the whole network
$\mathbf{C}$	The content caching matrix with (f,k)-th entry being $c_{f,k}$
$\mathbf{X}$	The content request matrix of all the UEs with (k,f)-th entry being $x_{m_k,f}$
Ψ	The joint cluster formation strategy of all the FAPs, which is a mapping from the tuple $\{\mathbf{H},\mathbf{X},\mathbf{C}\}$ to Φ
u	System throughput in each transmission interval
$\mathcal{A}$	The action set of the virtual agent in the SARL based caching algorithm
$\mathcal{A}_i$	The action set of the i-th virtual agent in the MARL based caching algorithm
$\pi_{i,j}$	The probability that the i-th virtual agent selects its j-th action in the MARL based caching algorithm
$z_{i,j}$	The utility estimation associated with the j-th action of the i-th virtual agent in the MARL based caching algorithm
a_i	The action taken by the i-th virtual agent in the MARL based caching algorithm
$\mathbf{a}_{-i}$	The action profile of all the virtual agents except the i-th virtual agent in the MARL based caching algorithm
ω	The benefit of unit long-term system throughput
μ	The cost of caching a content segment

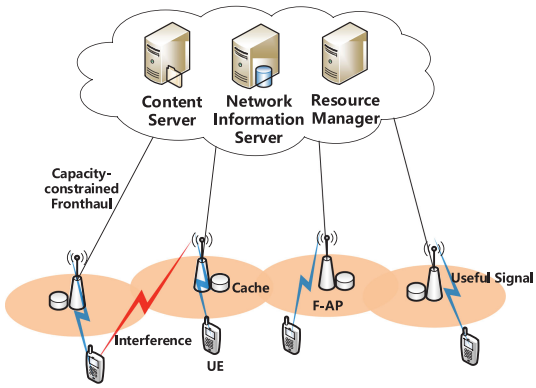

Fig. 1. The F-RAN scenario considered in this paper.

A. Key Components in the F-RAN

The F-RAN concerned is shown in Fig. 1 with a centralized cloud and multiple single-antenna FAPs that connect to the cloud via capacity-limited fronthaul. In the cloud, there is a resource manager, a content server and a network information server. The content server stores all the content segments that are potentially requested by UEs, whose set is denoted by $\mathcal{F}=\{1,2,...,F\}$, while the network information server records a network profile including history channel state information (CSI) and history UE content requests. At the network edge, each FAP has local radio resource management and caching capabilities [5]. Moreover, to alleviate the burden on fronthaul and the resource manager, the resource manager is only responsible for managing resources that are allocated on a large timescale, such as cache resource [6], while radio resource is allocated by fully utilizing the resource management capability of FAPs, which can achieve more timely adaptation to the dynamic environment. Define the set of FAPs as $\mathcal{K}=\{1,2,...,K\}$. Assume that each FAP initially serves one UE, which is a common way of initialization for UE-BS association [32], [33], but a UE is actually allowed to connect to multiple FAPs to take advantage of JT. Meanwhile, FAPs share the same frequency band [32] and hence there is inter-FAP interference.

B. Transmission Mode

To mitigate severe inter-FAP interference incurred by spectrum sharing, an effective way is to group FAPs into clusters, within each of which FAPs perform coordinated multi-point (CoMP), as indicated by [5], [32]–[34]. Denote the set of FAP clusters as $\Phi=\{\psi_1,\psi_2,...,\psi_n\}$ with $n\leq K$, $\psi_i\subseteq\mathcal{K}$, $\psi_i\cap\psi_j=\phi,\forall i,j\in\{1,2,...,n\}$ and $i\neq j$. In the following, Φ is also named as the partition of FAPs. Within each cluster, considering implementation complexity [35], a time division based non-coherent JT scheme is adopted. Although intra-cluster interference is avoided, there is still inter-cluster interference due to the simultaneous transmission of clusters over the same frequency band, and such interference noise suffered by FAP k is calculated as $\sum_{k'\in\mathcal{K}\backslash\psi_i}ph_{k'm_k}$. In this expression, ψ_i is the cluster to which FAP k belongs, m_k is the UE originally associated with FAP k, $k'\in K\backslash\psi_i$ represents FAP k' out of cluster ψ_i, p is the transmission power of each FAP, which is assumed to be constant, and $h_{k'm_k}$ is the channel gain between FAP k' and UE m_k.

Then, the data rate achieved by UE m_k belonging to cluster ψ_i is given by

$$R_{m_k}=\alpha_{m_k}\log_2\left(1+\frac{\sum_{k'\in\psi_i}ph_{k'm_k}}{\sum_{k'\in\mathcal{K}\backslash\psi_i}ph_{k'm_k}+\sigma^2}\right), \quad (1)$$

where $\sum_{k'\in\psi_i}ph_{k'm_k}$ is the received power of useful signal resulted by the JT scheme [35], α_{m_k} is the proportion of time domain resource occupied by UE m_k, and σ^2 is the noise. In this paper, we suppose that the time domain resource is equally allocated to each UE in the cluster, and that is $\alpha_{m_k}=\frac{1}{|\psi_i|}$. When FAP k does not participate in any cluster, the data rate of its user m_k is expressed as $R_{m_k}=\log_2(1+\frac{ph_{km_k}}{\sum_{k'\in\mathcal{K}\backslash\{k\}}ph_{k'm_k}+\sigma^2})$.

C. Two-Timescale Resource Management

Fig. 2 illustrates two-timescale cache and radio resource management. Before n transmission intervals start, the resource manager in the cloud allocates cache size at each FAP and pushes contents into FAPs' caches. At the beginning of each transmission interval with time length T during which CSI is assumed to remain unchanged, each UE requests a content segment, and then each UE is served by a single FAP for a whole transmission interval or served by a cluster of FAPs for a portion of an interval, which depends on the cluster formation result. Corresponding details will be presented later on. Note that CSI condition and user content requests are varying with transmission intervals. By considering a sufficiently large n, the

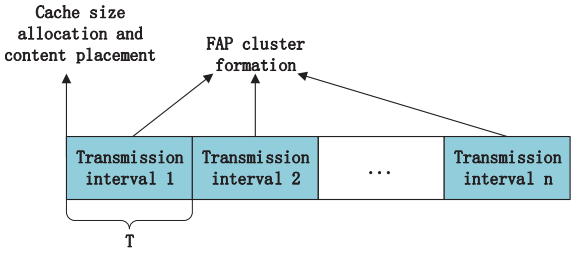

Fig. 2. An illustration of two-timescale cache and radio resource management.

resource manager aims at optimizing a long-term utility that will be presented in (3) by cache size allocation and content placement, which should be adaptive to the statistics of CSI and user content requests, while the cluster formation scheme executed by FAPs in each transmission interval needs to consider current CSI and user content requests. In the Subsection C of the next section, the coupling of long-term cache resource management and short-term radio resource management will be elaborated.

III. Problem Formulation

In this section, a short-term optimization problem and a long-term optimization problem are formulated for FAPs and the resource manager, respectively. Then, considering cache resource management and radio resource management are coupled and the resource manager and FAPs make sequential decisions, the two-timescale resource management is modeled as a Stackelberg game, which takes the resource manager and FAPs as the leader and followers, respectively.

A. Problem Formulation for FAPs

As stated in the system model, for the sake of timely adaptation to CSI variation and alleviating the burden on fronthaul links as well as the resource manager, the resource management capability of FAPs is utilized to perform cluster formation and re-allocate time domain resource among UEs in the same cluster. In addition, it should be noted that there is a precondition for the JT of FAPs, and that is each FAP should acquire the content segments requested by all the UEs in the same cluster. For the content segments that are missing in local caches, they need to be fetched from the content server in the cloud via capacity-limited fronthaul. The short-term resource management problem to be solved by FAPs in each transmission interval is formulated as follows.

$$\begin{aligned}
&\max_{\Phi}\ u = \sum_{m_k} R_{m_k}\left(\Phi, \mathbf{H}\right)\\
&(a)\ \sum_{f}\sum_{k'\in\psi_i} x_{m_{k'},f}\left(1-c_{f,k}\right) \le l_k,\ \forall k \in \psi_i, \forall \psi_i,\\
&(b)\ \psi_i \cap \psi_j = \phi,\ i \ne j,\\
&(c)\ \psi_1 \cup \psi_2 \cdots \cup \psi_n = \mathcal{K}.
\end{aligned} \tag{2}$$

In this problem, $x_{m_{k'},f} = 1$ indicates UE $m_{k'}$ requests content segment f, and $x_{m_{k'},f} = 0$ otherwise. $c_{f,k} = 1$ indicates content segment f is stored in the cache of FAP k, and $c_{f,k} = 0$ otherwise. $\mathbf{H}$ represents the channel gain matrix of the whole network with each entry being the channel gain between an FAP and a UE. Constraint (a) states that the number of missing content segments fetched by each FAP in each transmission interval is limited, due to constrained fronthaul capacity, which implicitly assumes that all the content segments possess equal size [24]. Constraint (b) and (c) state that FAP clusters should be disjoint and include all the FAPs in the network, respectively. In addition, it should be noted that the optimization of cluster formation Φ is not only closely related to $\mathbf{H}$ but also related to the user content requests and content placement. Define $\mathbf{C}$ as the content caching matrix with (f,k)-th entry being $c_{f,k}$ and define the content request matrix of all the UEs as $\mathbf{X}$ with (k,f)-th entry being $x_{m_k,f}$. According to problem (2), it is intuitive that the best joint cluster formation strategy of all the FAPs Ψ^*, which is a mapping from the tuple $\{\mathbf{H}, \mathbf{X}\}$ to Φ, is highly affected by $\mathbf{C}$. To highlight such dependence, we further denote Ψ^* as $\Psi^*_{\mathbf{C}}$, and the system throughput u in each transmission interval can be expressed as $u\left(\Psi^*_{\mathbf{C}}\left(\mathbf{H},\mathbf{X}\right),\mathbf{H}\right)$. *Note that the best joint cluster formation strategy Ψ^* in our paper means it is the best outcome for each FAP and not essentially the global optimal solution to (2).*

B. Problem Formulation for The Resource Manager

Assume that CSI matrix $\mathbf{H}$ and the content request matrix $\mathbf{X}$ are both i.i.d. with regard to each transmission interval. Aiming at maximizing the benefit of caching at FAPs, the problem for the resource manager in the cloud considering the joint cluster formation strategy of FAPs is as follows:

$$\begin{aligned}
&\max_{\mathbf{C}}\ U = \omega \mathbb{E}_{\mathbf{H},\mathbf{X}}\left[u\left(\Psi^*_{\mathbf{C}}\left(\mathbf{H},\mathbf{X}\right),\mathbf{H}\right)\right] - \mu\sum_{f}\sum_{k} c_{f,k}\\
&(d)\ c_{f,k} \in \{0,1\},
\end{aligned} \tag{3}$$

where ω and μ are two weight factors representing the benefit of unit long-term throughput and the cost of caching a content segment, respectively. Note that the cache size and content placement at each FAP will be determined, once $\mathbf{C}$ is derived. The objective of the resource manager includes two parts. The first part is the benefit brought by long term expected system throughput, while the second part is the cost led by cache deployment at FAPs and content pushing. Similar cost has also been taken into account in [26].

C. A Game-Theoretic Perspective

Considering that each FAP can be seen as a rational entity, the FAP cluster formation process in each transmission interval can be naturally modeled as a coalition formation game [36], which helps develop a fully distributed and low complexity algorithm. In literatures, coalitional game has been used to model cooperative behavior among network nodes. For example, in [36], authors use coalition formation game to capture the cooperative behavior of SBSs that form cooperative clusters and then perform TDMA based transmission coordination in the same cluster. In [37], cellular users and D2D users can

form cooperative groups by participating a coalitional game to improve spectrum utilization. Formally, our studied game is defined as follows.

Definition 1: The cluster formation game among FAPs is described by $\mathcal{G} = \{\mathcal{K}, \{u_k\}, \{\underset{k}{\succ}\}\}$, where $\mathcal{K}$ is the set of players, i.e., the set of FAPs, u_k is the utility function of FAP k, and $\underset{k}{\succ}$ is the preference of FAP k over different FAP clusters. u_k and $\underset{k}{\succ}$ are given by

$$u_k = R_{m_k}, \tag{4}$$

and

$$\psi \underset{k}{\succ} \psi_{(k)} \Leftrightarrow \begin{cases} u_k\left(\tilde{\Phi}\right) > u_k\left(\Phi\right), \\ \sum_{k' \in \psi \cup \psi_{(k)}} u_{k'}\left(\tilde{\Phi}\right) > \sum_{k' \in \psi \cup \psi_{(k)}} u_{k'}\left(\Phi\right), \\ \sum_f \sum_{k' \in \psi \cup \{k\}} x_{m_{k'},f}\left(1 - c_{f,k''}\right) \le l_{k''}, \\ \forall k'' \in \psi \cup \{k\}, \end{cases} \tag{5}$$

respectively, with $\psi \in \Phi \backslash \{\psi_{(k)}\}$, $\psi_{(k)}$ being the cluster to which FAP k belongs under cluster formation Φ, and $\tilde{\Phi} = \Phi \backslash \{\psi, \psi_{(k)}\} \cup \{\psi \cup \{k\}, \psi_{(k)} \backslash \{k\}\}$. Note that FAP k can also choose to form singleton $\{k\}$ with $\psi = \{\}$ in (5).

The above definition states that FAP k prefers another cluster ψ to its current cluster if and only if: 1) The data rate of its user is strictly improved; 2) The sum data rate of UEs involved in the cluster transfer of FAP k should be increased; 3) After FAP k joins the cluster ψ, the fronthaul capacity constraints of all the FAPs in ψ should be satisfied. Note that the leaving of FAP k will not increase the fronthaul capacity demands for the FAPs staying in $\psi_{(k)}$, and hence their fronthaul capacity constraints are still met.

It has been stated in Subsection A that the joint cluster formation strategy of FAPs highly depends on the strategy of the resource manager, i.e., $\mathbf{C}$. On the other hand, the strategy $\mathbf{C}$ of the resource manager also depends on the joint cluster formation strategy of FAPs according to problem (3), and hence the strategies of FAPs and the resource manager are coupled. Moreover, since the resource manager makes its decision on cache deployment and content placement first, and then FAPs react to its strategy by playing a coalition formation game in each transmission interval, *the interaction between the resource manager and FAPs can be modeled as a Stackelberg game that takes the resource manager and FAPs as the leader and followers, respectively [30].* The equilibrium of the formulated game is defined as follows.

Definition 2: For a solution $\{\mathbf{C}^*, \Psi^*_{\mathbf{C}^*}\}$ to the Stackelberg resource management game, Stackelberg equilibrium (SE) is achieved if and only if

$$\begin{aligned} &U\left(\mathbf{C}^*, \Psi^*_{\mathbf{C}^*}\right) \ge U\left(\mathbf{C}, \Psi^*_{\mathbf{C}}\right), \\ &\psi^*_{(k)}(t) \underset{k}{\succ} \psi^*_i(t),\ \forall k,\ \forall t,\ \forall \psi^*_i(t) \in \Phi^*(t) / \left\{\psi^*_{(k)}(t)\right\}, \end{aligned} \tag{6}$$

where $\psi^*_{(k)}\left(t\right) \in \Phi^*\left(t\right) = \Psi^*_{\mathbf{C}^*}\left(\mathbf{H}\left(t\right), \mathbf{X}\left(t\right)\right)$ is the cluster to which FAP k belongs in transmission interval t when the caching strategy of the resource manager is $\mathbf{C}^*$.

This definition states that once SE is achieved, each FAP has no incentive to deviate from its cluster formed in any transmission interval if the strategies of the resource manager and other FAPs are fixed, and meanwhile the resource manager has no incentive to adjust cache resource with FAPs reacting by taking the best joint cluster formation strategy.

IV. Algorithm Design

In this section, we first develop a cluster formation algorithm for FAPs to achieve a stable cluster formation result, under any fixed strategy of the resource manager. Then, taking the behavior of FAPs into account, two algorithms based on RL are proposed to approach a global or local optimal strategy of the resource manager.

A. Cluster Formation Algorithm Design

In this subsection, our aim is to design a distributed algorithm for FAPs to reach a stable partition, given any tuple of $\{\mathbf{H}, \mathbf{X}, \mathbf{C}\}$. To this end, an algorithm based on the preference order of FAPs over clusters is developed, whose procedure is illustrated in Algorithm 1. Initially, each FAP does not cooperate with each other, and then each FAP starts to seek for a cluster to join. In Subsection A of Section V, it will be shown that Algorithm 1 will finally converge after finite repeats of Stage 2, and joining the cluster under the FAP partition output by Algorithm 1 is the best strategy for each FAP, which means a stable state is achieved. In addition, since checking preference order only needs the local information within the involved clusters, the proposed cluster formation algorithm is distributed.

Finally, note that the clustering algorithm relies on the calculation of UE data rate given by (1) that is related to instantaneous CSI. Hence, FAPs have to re-cluster in each coherence time, which can cause large signaling overhead. To overcome this issue, instead of considering instantaneous data rate, clustering can be performed based on average UE data rate that is related to only large scale channel gain and its analysis can follow literature [38]. Once the average UE data rate expression is obtained, our proposed clustering algorithm can be applied. In addition, synchronization among FAPs is another big concern. Fortunately, the adopted non-coherent JT does not require stringent synchronization [39] and hence is more practical than other coherent CoMP schemes like interference alignment in [32] and zero-forcing beamforming in [33].

B. Cache Optimization Algorithm Design

After developing the algorithm for followers that results in a stable state under any fixed strategy of the leader, the remaining task to solve the formulated Stackelberg game is to derive the optimal leader's strategy, taking the followers' behavior into account. Up to now, Stackelberg game has been applied to various resource optimization problems in wireless networks. For example, in [40], [41], interference control in heterogenous networks is modeled as a Stackelberg game with a macro BS as the leader who sets an interference price. In [42], authors apply Stackelberg game to resource allocation in physical layer

Algorithm 1: Cluster Formation Algorithm for FAPs Based on Preference Order in Each Transmission Interval.

1: **Stage 1:**
FAPs initialize a partition $\Phi_{ini} = \{\{1\}, \{2\}, ..., \{K\}\}$, which means FAPs behave non-cooperatively.
Denote the current coalition to which FAP k belongs as $\psi_{(k)}$ with $\psi_{(k)} = \{k\}$ initially.

2: **Stage 2:**
For FAP $k = 1 : K$
For each cluster $\psi \in \Phi_{ini}$
If $\psi \underset{k}{\succ} \psi_{(k)}$
$\Phi_{ini} \leftarrow \Phi_{ini} \backslash \{\psi_{(k)}, \psi\} \cup \{\psi_{(k)} \backslash \{k\}, \psi \cup \{k\}\}$.
End If
End For
Check whether FAP k can form singleton $\{k\}$.
If yes, $\Phi_{ini} \leftarrow \Phi_{ini} \backslash \{\psi_{(k)}\} \cup \{\psi_{(k)} \backslash \{k\}, \{k\}\}$.
End For

3: **Stage 3:**
Repeat **Stage 2** until Algorithm 1 converges.

security while the management of cross-interference among operators in a wireless network with unlicensed spectrum is modeled as a multi-leader-multi-follower game in [43]. Different from [40]–[43], the leader's objective in our game is a long-term metric, and meanwhile the strategy of the leader is discontinuous, which both incur new challenges.

On the other hand, different approaches have been proposed to optimize long-term performance metrics in literatures studying two-timescale resource allocation, and the core ideas of their proposals can be classified into three categories. The first one is to use sample average approximation (SAA) to transform the expected objective into the average of objectives over multiple environmental samples and then design optimization algorithms [27], [28], [31]. The second idea is to directly prove the convexity of the long-term objective or transform the objective into a convex form, and then sub-gradient algorithm is utilized [25], [26]. The last idea is to adopt local stochastic cutting plane [29]. Particularly, the proposals in [27], [28], [31] are based on the closed form of the objective after SAA, while the proposals in [25], [26], [29] handle continuous variables. However, in our work, even after using SAA, the resulted objective has no closed form, since there is no closed form for $\Psi_{\mathbf{C}}$ in (3), and meanwhile our optimization variables are discrete. To overcome the challenges caused by no closed form and the discontinuity of optimization variables, model-free RL is used in this paper.

1) Towards the Optimal Caching Strategy: Assume that the cloud has stored a history profile $\Gamma = \{\{\mathbf{H}_1, \mathbf{X}_1\}, \{\mathbf{H}_2, \mathbf{X}_2\}, \ldots, \{\mathbf{H}_T, \mathbf{X}_T\}\}$ in the network information server, where each $\{\mathbf{H}_t, \mathbf{X}_t\} \in \Gamma$ is the tuple of the channel gain matrix and user content request matrix in history transmission interval t. *It is assumed that the channel gain distribution of each link and the content request distribution of each user are both invariant for a sufficiently long time. Under this assumption, the optimization problem (3) for a future, long period is the same as that corresponding to a past, long period, which makes it possible to utilize the collected history profile to solve problem (3) for a future period.*

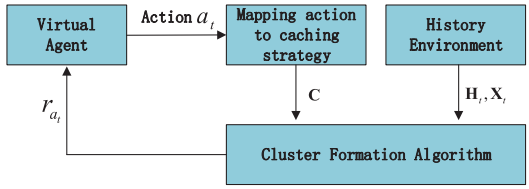

Fig. 3. An illustration of the SARL based caching algorithm.

To exploit RL, the resource manager creates a virtual agent, whose action set is defined as $\mathcal{A} = \{a_1, a_2, \ldots, a_L\}$ with $L = 2^{KF}$, where each action $a_l \in \mathcal{A}$ represents a possible cache allocation matrix $\mathbf{C}$. Then, Algorithm 2 is proposed to help the resource manager approach its optimal strategy by learning from the history network data. Specifically, at the start of each history transmission interval t, the virtual agent selects a caching strategy $a_t \in \mathcal{A}$ following a certain policy like ε-greedy policy. Given action a_t, the proposed cluster formation algorithm is executed, and then the environment feeds back a signal r_{a_t} to the virtual agent to guide the learning process. r_{a_t} is given by

$$r_{a_t} = \omega u\left(a_t, \mathbf{H}_t, \mathbf{X}_t\right) - \mu \sum_f \sum_k c_{f,k}\left(a_t\right), \tag{7}$$

where $u\left(a_t, \mathbf{H}_t, \mathbf{X}_t\right)$ is the system throughput in transmission interval t defined in (2), and $c_{f,k}\left(a_t\right)$ represents the caching decision associated with content segment f at FAP k under action a_t.

Once the feedback signal is generated for transmission interval t, the virtual agent makes the following update:

$$Q_{a_l}^{t+1} = Q_{a_l}^t + \lambda_t \mathbb{I}_{\{a_t = a_l\}} \left(r_{a_t} - Q_{a_l}^t\right), \tag{8}$$

where $\mathbb{I}_{\{a_t = a_l\}}$ equals to 1 if action a_l is taken in transmission interval t and equals to 0 otherwise. After Algorithm 2 converges, the optimal caching strategy is determined by $a^* = \arg\max_{a_l} Q_{a_l}$ with Q_{a_l} representing the convergence value of $Q_{a_l}^t$, and $Q_{a_l} = U_{a_l}$ according to Theorem 1 in the next section, where U_{a_l} is the objective value of the resource manager by taking action a_l. *At the beginning of another future, long period, the resource manager first takes action a^*, and then FAPs form clusters in each subsequent transmission interval by Algorithm 1. In this way, SE will be achieved, following Definition 2.* To make the process of Algorithm 2 more intuitive, Fig. 3 is drawn.

2) Overcoming the Curse of Dimensionality: Although the optimal caching strategy can be achieved by using Algorithm 2, the number of actions of the created single agent is 2^{KF}, which exponentially increases with the number of FAPs and content segments. To overcome this issue, a multi-agent RL (MARL) based caching algorithm is proposed in Algorithm 3. The basic idea is to create $D = KF$ virtual agents, and the action set of agent i is denoted as $\mathcal{A}_i = \{a_{i,1}, a_{i,2}\}$. When $i = (k-1)F + f$, taking action $a_{i,1}$ means caching content segment f at FAP k, while taking $a_{i,2}$ means the opposite.

The difference between Algorithm 3 and Algorithm 2 lies in the update process based on the signal fed back by the

Algorithm 2: Single-Agent RL (SARL) Based Caching Algorithm.

1: **Stage 1:**
The resource manager initializes a virtual environment based on history network data $\Gamma = \{\{\mathbf{H}_1, \mathbf{X}_1\}, \{\mathbf{H}_2, \mathbf{X}_2\}, ..., \{\mathbf{H}_T, \mathbf{X}_T\}\}$ and creates a virtual agent, whose action set is $\mathcal{A} = \{a_1, a_2, \ldots, a_L\}$ with $L = 2^{KF}$. Initialize $t = 1$, $Q^t_{a_l} = 0$ for each a_l, the exploration probability ε and the learning rate λ_t.

2: **Stage 2:**
For History transmission interval $t = 1 : T$
Generate a random number z
If $z < \varepsilon$
The virtual agent selects a caching strategy a_t with equal probability.
Else
The virtual agent selects the action associated with the maximum Q-value.
End if
Under a_t, $\mathbf{H}_t$ and $\mathbf{X}_t$, Algorithm 1 is executed to output a cluster formation result.
The environment calculates and feedbacks the signal r_{a_t} to the virtual agent.
The virtual agent makes an update according to equation (8).
End For

environment. Specifically, in transmission interval t, each virtual agent $i \in \{1, 2, \ldots, D\}$ makes the following update:

$$z^{t+1}_{i,j} = \alpha_t I_{\left\{a^t_i = a_{i,j}\right\}} \left(r_{a^t_i} - z^t_{i,j}\right) + z^t_{i,j},$$
$$\pi^{t+1}_{i,j} = \pi^t_{i,j} + \lambda_t \left(\beta\left(z^t_{i,j}\right) - \pi^t_{i,j}\right), \quad (9)$$

where a^t_i is the action taken by virtual agent i at the start of transmission interval t, $r_{a^t_i}$ is the corresponding feedback signal that follows the same definition as that in Algorithm 2, which also means all virtual agents share the same feedback signal, $\pi^t_{i,j}$ is the probability to select action $a_{i,j}$, $z^t_{i,j}$ is the estimated utility by taking action $a_{i,j}$, $\beta(z^t_{i,j}) = \frac{\exp(\kappa z^t_{i,j})}{\exp(\kappa z^t_{i,1}) + \exp(\kappa z^t_{i,2})}$ with κ balancing the tradeoff between exploration and exploitation, and λ_t and α_t are learning rates. Meanwhile, λ_t and α_t should satisfy the following conditions:

$$\sum_{t\geq 1} \lambda_t = \infty, \ \sum_{t\geq 1} {\lambda_t}^2 \leq \infty,$$
$$\sum_{t\geq 1} \alpha_t = \infty, \ \sum_{t\geq 1} {\alpha_t}^2 \leq \infty,$$
$$\lim_{t\to\infty} \frac{\lambda_t}{\alpha_t} = 0, \quad (10)$$

which are essential for the convergence of Algorithm 3 [44].

According to Theorem 2 in the next section, Algorithm 3 can finally approach a global or local optimal caching strategy for the resource manager. *Therefore, when the resource manager implements the caching strategy output by Algorithm 3 and then FAPs form clusters in subsequent transmission intervals by Algorithm 2, the resulting state can be seen as a weak version of SE, since the resource manager can gain higher utility by deviating from the current caching strategy.* To make the process of Algorithm 3 more intuitive, Fig. 4 is drawn.

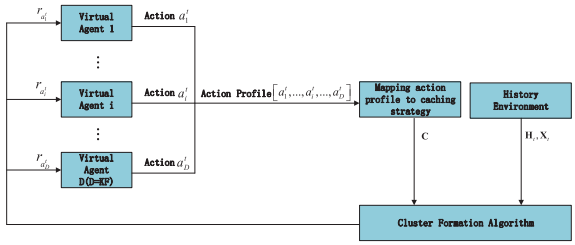

Fig. 4. An illustration of the MARL based caching algorithm.

Algorithm 3: Multi-Agent RL (MARL) Based Caching Algorithm.

1: **Stage 1:**
The resource manager initializes a virtual environment based on history network data $\Gamma = \{\{\mathbf{H}_1, \mathbf{X}_1\}, \{\mathbf{H}_2, \mathbf{X}_2\}, ..., \{\mathbf{H}_T, \mathbf{X}_T\}\}$ and creates KF virtual agents. Initialize $t = 1$, λ_t and α_t. Set $z^t_{i,j} = 0$ and $\pi^t_{i,j} = 0.5$ for virtual agent i and its action $a_{i,j}$.

2: **Stage 2:**
For History transmission interval $t = 1 : T$
Each virtual agent i generates a random number l_i
If $l_i < \pi^t_{i,1}$
Virtual agent i takes action $a_{i,1}$.
Else
Virtual agent i takes action $a_{i,2}$.
End if
Under the action profile of all virtual agents $\mathbf{a}_t = [a^t_1, a^t_2, \ldots, a^t_{KF}]$, $\mathbf{H}_t$ and $\mathbf{X}_t$, Algorithm 1 is executed to output a cluster formation result.
The environment calculates and feedbacks the signal $r_{a^t_i}$ to each virtual agent.
Each virtual agent makes an update according to equation (9).
End For

V. The Properties of the Proposed Algorithms

In this section, the properties of the proposals are analyzed, in terms of convergence, optimality and complexity. In addition, a theoretic guidance on the practical selection of parameter κ for the MARL based caching algorithm is provided.

A. The Properties of The Cluster Formation Algorithm

1) Convergence: First, we discuss the convergence of the cluster formation algorithm. From (1), it can be observed that the achieved data rate by UE m_k is not influenced by the cluster formation of FAPs outside the cluster to which it belongs.

Therefore, when the partition of FAPs is updated in Stage 2 of Algorithm 1 based on the preference order, the system throughput will strictly increase. Since throughout is bounded, after finite repeats of Stage 2, the partition of FAPs will finally be unchanged. Hence, Algorithm 1 will be guaranteed to converge.

2) Stability: Once Algorithm 1 converges to a final FAP partition Φ^*, it can be directly concluded that $\psi_{(k)} \succ_k \psi$ for any FAP k and any $\psi \in \Phi^*/\{\psi_{(k)}\}$. This stable state is Nash-stable [37].

3) Optimality: The relationship between local optimality and stability is revealed by the following theorem.

Theorem 1: Any local optimal FAP partition must be Nash-stable, while a Nash-stable partition may not be local optimal.

Proof: Under a local optimal partition, switching any FAP from its current cluster to another cluster will not contribute to a strict increase of system throughput. Assume that there exists a local optimal partition that is not Nash-stable. Then, it means a certain FAP is willing to join a different cluster, which will strictly improve the throughput according to the definition of preference order. Hence, contradiction occurs and it is concluded that any local optimal partition must be Nash stable. On the other hand, since a Nash-stable partition is achieved based on the preference order that requires a strict improvement of FAP's own performance, it is still possible to further raise the system throughput by switching an FAP from its current cluster to another one under a Nash-stable partition. Thus, a Nash-stable partition may not be local optimal. ■

4) Complexity: Since Stage 2 of Algorithm 1 needs to check at most $K(K+1)$ preference orders for each repeat, the complexity of Algorithm 1 is $\mathcal{O}(IK^2)$ with I being the number of repeats of Stage 2 for convergence.

B. The Properties of The SARL Based Caching Algorithm

In the following, the convergence and optimality of Algorithm 2 are rigorously proved.

Theorem 2: Algorithm 2 finally converges with $Q_{a_l}^t = U_{a_l}$ for each action a_l, as $t \to \infty$, where U_{a_l} is the objective value defined in (3) under caching action a_l.

Proof: For Algorithm 2, when the exploration probability is always kept non-zero, each action can be visited infinitely. Meanwhile, the feedback signal given by (7) is bounded. In addition, because it has been assumed that the distributions of channel gain matrix $\mathbf{H}$ and user content request matrix $\mathbf{X}$ are i.i.d across transmission intervals, r_{a_l} follows the same distribution over time. At last, according to [45], by setting λ_t in the form of $\frac{1}{(t+y)^\rho}$ with $y > 0$ and $\frac{1}{2} < \rho \leq 1$, we have

$$\sum_{t \geq 1} \lambda_t = \infty, \tag{11}$$

and

$$\sum_{t \geq 1} \lambda_t^2 \leq \infty. \tag{12}$$

Therefore, following [46], $Q_{a_l}^t$ finally converges to $\mathbb{E}_{\mathbf{H},\mathbf{X}}[r_{a_l}]$ as $t \to \infty$.

Meanwhile, note that

$$\begin{aligned}
&\mathbb{E}_{\mathbf{H},\mathbf{X}}[r_{a_l}] \\
&= \mathbb{E}_{\mathbf{H},\mathbf{X}}\left[\omega u(a_l, \mathbf{H}, \mathbf{X}) - \mu \sum_f \sum_k c_{f,k}(a_l)\right] \\
&= \mathbb{E}_{\mathbf{H},\mathbf{X}}[\omega u(a_l, \mathbf{H}, \mathbf{X})] - \mathbb{E}_{\mathbf{H},\mathbf{X}}\left[\mu \sum_f \sum_k c_{f,k}(a_l)\right] \\
&= \omega \mathbb{E}_{\mathbf{H},\mathbf{X}}[u(a_l, \mathbf{H}, \mathbf{X})] - \mu \sum_f \sum_k c_{f,k}(a_l),
\end{aligned} \tag{13}$$

where the right-most side is exactly the objective value of the resource manager by taking action a_l. Hence, it can be concluded that

$$Q_{a_l}^t = \mathbb{E}_{\mathbf{H},\mathbf{X}}[r_{a_l}] = U_{a_l}, \tag{14}$$

as $t \to \infty$. After Algorithm 2 converges, the optimal caching strategy of the resource manager can be directly identified by the action with the maximum Q value. ■

At last, the complexity of Algorithm 2 is analyzed. Since generating a random number and making an update based on (8) both incur computation complexity $\mathcal{O}(1)$, the total complexity of Algorithm 2 is decided by the complexity of Algorithm 1. Therefore, the complexity of Algorithm 2 is $\mathcal{O}\left(IK^2\right)$ with I being the number of repeats of Stage 2 in Algorithm 1 for convergence.

C. The Properties of The MARL Based Caching Algorithm

In this subsection, we employ the techniques from stochastic approximation to prove the convergence of Algorithm 3, and the optimality is analyzed based on the property of potential game. Moreover, the total complexity is presented as well.

1) Convergence: Denote $\boldsymbol{\pi}_{-i}$ as the strategy profile of all the virtual agents except agent i. Under any fixed $\boldsymbol{\pi}_{-i}$, the convergence of the utility estimation $z_{i,j}^t$ directly holds by following the proof of Theorem 2, when $\pi_{i,j} \neq 0$. More specifically, we have

$$z_{i,j}^t = \mathbb{E}_{\mathbf{H},\mathbf{X},\boldsymbol{a}_{-i}}\left[r_{a_{i,j}}(\mathbf{H}, \mathbf{X}, \boldsymbol{a}_{-\mathbf{i}})\right], \text{ as } t \to \infty, \tag{15}$$

where $r_{a_{i,j}}(\mathbf{H}, \mathbf{X}, \boldsymbol{a}_{-i})$ is the feedback signal from the environment when virtual agent i takes action $a_{i,j}$ and all the other agents take action profile $\boldsymbol{a}_{-i}$. For brevity, we define

$$\bar{r}_{i,j}(\boldsymbol{\pi}_{-i}) = \mathbb{E}_{\mathbf{H},\mathbf{X},\boldsymbol{a}_{-i}}\left[r_{a_{i,j}}(\mathbf{H}, \mathbf{X}, \boldsymbol{a}_{-i})\right]. \tag{16}$$

Next, based on the result in [45], the update of strategy, i.e., the second equation in (9), can be approximated by the following ordinary differential equation (ODE):

$$\dot{\pi}_{i,j} = \beta\left(z_{i,j}^t\right) - \pi_{i,j}. \tag{17}$$

According to the last constraint in (10), $\pi_{i,j}^t$ changes on a slower timescale than the timescale on which $z_{i,j}^t$ is learned. Then, the ODE can be analyzed as if the utility estimation is accurate [46]. Denote $\boldsymbol{\pi}_i = [\pi_{i,1}, \pi_{i,2}]$ and $\boldsymbol{\beta}_i(\boldsymbol{\pi}_{-i}) =$

$[\beta(\bar{r}_{i,1}(\boldsymbol{\pi}_{-i})), \beta(\bar{r}_{i,2}(\boldsymbol{\pi}_{-i}))]$, in which

$$\beta(\bar{r}_{i,j}(\boldsymbol{\pi}_{-i})) = \frac{\exp(\kappa \bar{r}_{i,j}(\boldsymbol{\pi}_{-i}))}{\exp(\kappa \bar{r}_{i,1}(\boldsymbol{\pi}_{-i})) + \exp(\kappa \bar{r}_{i,2}(\boldsymbol{\pi}_{-i}))}, \quad (18)$$

for $j = 1, 2$.

Then, based on [47], ODE (17) can be expressed as

$$\begin{aligned}\dot{\boldsymbol{\pi}}_i &= \boldsymbol{\beta}_i(\boldsymbol{\pi}_{-i}) - \boldsymbol{\pi}_i \\ &= \arg\max_{\mathbf{y}\in\Delta} \left[\mathbf{y}\mathbf{r}_{i,\boldsymbol{\pi}_{-i}} + \frac{1}{\kappa}\left(-\sum y_j \ln y_j\right)\right] - \boldsymbol{\pi}^i, \quad (19)\end{aligned}$$

where $\mathbf{r}_{i,\boldsymbol{\pi}_{-i}} = [\bar{r}_{i,1}(\boldsymbol{\pi}_{-i}), \bar{r}_{i,2}(\boldsymbol{\pi}_{-i})]^T$ and $\Delta = \{\mathbf{y} | y_1 + y_2 = 1, y_1 > 0,\ y_2 > 0\}$. Since all the virtual agents always receive the same feedback signal, a strict Lyapunov function for the dynamic (19) exists [47]. Therefore, the rest point of (19) is globally asymptotically stable [48], and hence we have

$$\begin{aligned}\pi_{i,j} &= \beta(\bar{r}_{i,j}(\boldsymbol{\pi}_{-i})) \\ &= \frac{\exp(\kappa \bar{r}_{i,j}(\boldsymbol{\pi}_{-i}))}{\exp(\kappa \bar{r}_{i,1}(\boldsymbol{\pi}_{-i})) + \exp(\kappa \bar{r}_{i,2}(\boldsymbol{\pi}_{-i}))}, \text{ as } t \to \infty. \quad (20)\end{aligned}$$

2) Optimality: In the following, we discuss the optimality of the MARL based caching algorithm by utilizing the properties of entropy function and potential game theory. First, based on (19), we have

$$\begin{aligned}\boldsymbol{\beta}_i(\boldsymbol{\pi}_{-i}) &= \arg\max_{\boldsymbol{\pi}_i\in\Delta} \left[\boldsymbol{\pi}_i \mathbf{r}_{i,\boldsymbol{\pi}_{-i}} + \frac{1}{\kappa}\left(-\sum \pi_{i,j} \ln \pi_{i,j}\right)\right] \\ &= \arg\max_{\boldsymbol{\pi}_i\in\Delta} \left[\bar{r}_i(\boldsymbol{\pi}_i, \boldsymbol{\pi}_{-i}) + \frac{1}{\kappa} G(\boldsymbol{\pi}_i)\right], \quad (21)\end{aligned}$$

where $\bar{r}_i(\boldsymbol{\pi}_i, \boldsymbol{\pi}_{-i}) = \boldsymbol{\pi}_i \mathbf{r}_{i,\boldsymbol{\pi}_{-i}}$ and $G(\boldsymbol{\pi}_i)$ is the entropy function. According to the property of $G(\boldsymbol{\pi}_i)$, we have $0 \le G(\boldsymbol{\pi}_i) \le G\left(\frac{1}{2}\right) = \ln 2$. Denote the convergence strategy of virtual agent i as $\boldsymbol{\pi}_i^*$ that satisfies $\boldsymbol{\pi}_i^* = \boldsymbol{\beta}_i\left(\boldsymbol{\pi}_{-i}^*\right)$ based on the result in (20). Then, the global or local optimality of the MARL based caching algorithm is given by the following theorem.

Theorem 3: Algorithm 3 approaches a global or local optimal solution to problem (3) as $\kappa \to +\infty$.

Proof: From (21), the following inequality holds:

$$\begin{aligned}&\bar{r}_i\left(\boldsymbol{\pi}_i, \boldsymbol{\pi}_{-i}^*\right) + \frac{1}{\kappa} G(\boldsymbol{\pi}_i) \\ &\le \bar{r}_i\left(\boldsymbol{\beta}_i\left(\boldsymbol{\pi}_{-i}^*\right), \boldsymbol{\pi}_{-i}^*\right) + \frac{1}{\kappa} G\left(\boldsymbol{\beta}_i\left(\boldsymbol{\pi}_{-i}^*\right)\right). \quad (22)\end{aligned}$$

Then, we have

$$\begin{aligned}&\bar{r}_i\left(\boldsymbol{\pi}_i, \boldsymbol{\pi}_{-i}^*\right) - \bar{r}_i\left(\boldsymbol{\beta}_i\left(\boldsymbol{\pi}_{-i}^*\right), \boldsymbol{\pi}_{-i}^*\right) \\ &\le \frac{1}{\kappa}\left(G\left(\boldsymbol{\beta}_i\left(\boldsymbol{\pi}_{-i}^*\right)\right) - G(\boldsymbol{\pi}_i)\right) \le \frac{1}{\kappa} \ln 2. \quad (23)\end{aligned}$$

According to (20), as $\kappa \to +\infty$, virtual agent i will intend to play a deterministic action $a_i^* = a_{i,j}$ that leads to the largest $\bar{r}_{i,j}(\boldsymbol{\pi}_{-i})$, since $\exp(\kappa \bar{r}_{i,j}(\boldsymbol{\pi}_{-i})) \gg \exp(\kappa \bar{r}_{i,j'}(\boldsymbol{\pi}_{-i}))$ for any $j' \ne j$ at this time.

Hence, as $\kappa \to +\infty$, (23) will approach the following inequality:

$$r_{a_i}\left(\boldsymbol{a}_{-i}^*\right) - r_{a_i^*}\left(\boldsymbol{a}_{-i}^*\right) \le 0, \quad (24)$$

where $r_{a_i}(\boldsymbol{a}_{-i}) = \mathbb{E}_{\mathbf{H},\mathbf{X}}[r_{a_i}(\mathbf{H}, \mathbf{X}, \boldsymbol{a}_{-i})]$ and a_i is an arbitrary action. Note that all the virtual agents share a common feedback signal defined by (7), and hence we have

$$\begin{aligned}r_{a_i}(\boldsymbol{a}_{-i}) &= \mathbb{E}_{\mathbf{H},\mathbf{X}}[r_{a_i}(\mathbf{H}, \mathbf{X}, \boldsymbol{a}_{-i})] \\ &= \mathbb{E}_{\mathbf{H},\mathbf{X}}\left[\omega u(a_i, \boldsymbol{a}_{-i}, \mathbf{H}, \mathbf{X}) - \mu \sum_f \sum_k c_{f,k}(a_i, \boldsymbol{a}_{-i})\right] \\ &= \omega \mathbb{E}_{\mathbf{H},\mathbf{X}}[u(a_i, \boldsymbol{a}_{-i}, \mathbf{H}, \mathbf{X})] - \mu \sum_f \sum_k c_{f,k}(a_i, \boldsymbol{a}_{-i}) \\ &= \Omega(a_i, \boldsymbol{a}_{-i}). \quad (25)\end{aligned}$$

Thus, $(a_i, \boldsymbol{a}_{-i})$ is a pure strategy Nash equilibrium of the exact potential game with virtual agents as players and $\Omega(a_i, \mathbf{a}_{-i})$ as the potential function. In addition, $\Omega(a_i, \mathbf{a}_{-i})$ is exactly the same as the objective of the resource manager that is defined in (3). Since pure strategy Nash equilibrium maximizes the potential function either globally or locally, it is concluded that Algorithm 3 approaches a global or local optimal caching strategy of the resource manager, as $\kappa \to +\infty$. ■

3) Complexity of Algorithm 3: In the action selection stage, each virtual agent needs to generate a random number to select an action, whose complexity is $\mathcal{O}(1)$. Then, the cluster formation algorithm in Algorithm 1 is executed, whose complexity is $\mathcal{O}(IK^2)$. After receiving the feedback signal from the environment, each virtual agent makes an update following (9) in parallel. Since the update equations in (9) just involve fixed number of operations of exponents, additions and multiplications, the complexity for each virtual agent is $O(1)$. Therefore, the total complexity of Algorithm 3 is $\mathcal{O}(IK^2)$, where I is the number of repeats of Stage 2 in Algorithm 1 that is needed for convergence.

4) The Selection of Parameter κ: Although Algorithm 3 can approach the local or global optimal caching strategy of the resource manager, this result is established on the convergence of multi-agent RL, which requires $t \to \infty$. That is the resource manager should collect infinite amount of history network data. To achieve a good performance with limited data, parameter κ should be selected carefully. To facilitate the analysis, we first show that the sum of action selection probabilities for each virtual agent is always equal to 1.

Theorem 4: For any virtual agent i, if $\sum_j \pi_{i,j}^1 = 1$, the strategy update equation in (9) ensures that $\sum_j \pi_{i,j}^t = 1$ with $t \ge 2$.

Proof: By the second equation in (9), we have

$$\sum_j \pi_{i,j}^{t+1} = \sum_j \pi_{i,j}^t + \lambda_t \sum_j \left(\beta\left(z_{i,j}^t\right) - \pi_{i,j}^t\right)$$
$$= \sum_j \pi_{i,j}^t + \lambda_t \left(1 - \sum_j \pi_{i,j}^t\right). \quad (26)$$

Then, it can be seen that $\sum_j \pi_{i,j}^1 = 1 \Rightarrow \sum_j \pi_{i,j}^2 = 1 \Rightarrow \sum_j \pi_{i,j}^3 = 1 \ldots$, and hence $\sum_j \pi_{i,j}^t = 1$ with $t \geq 2$. ■

Next, we analyze the impact of κ on the strategy evolution by involving the following theorem.

Theorem 5: For virtual agent i, assume that action $a_{i,1}$ has a larger utility estimation than action $a_{i,2}$. Then, $\beta\left(z_{i,1}^t\right)$ monotonously increases with respect to κ.

Proof: By taking the derivative of $\beta\left(z_{i,1}^t\right)$ with respect to κ, we have

$$\frac{d\beta\left(z_{i,1}^t\right)}{d\kappa} = \beta\left(z_{i,1}^t\right) \frac{\exp\left(\kappa z_{i,2}^t\right)\left(z_{i,1}^t - z_{i,2}^t\right)}{\exp\left(\kappa z_{i,1}^t\right) + \exp\left(\kappa z_{i,2}^t\right)}. \quad (27)$$

Since $z_{i,1}^t > z_{i,2}^t$, $\frac{d\beta\left(z_{i,1}^t\right)}{d\kappa} > 0$. Thus, when κ becomes larger, $\beta\left(z_{i,1}^t\right)$ will increase, and the probability of selecting action $a_{i,1}$ will have a higher opportunity to increase based on the second equation in (9). Correspondingly, the probability of selecting action $a_{i,2}$ will decrease based on Theorem 4. ■

According to the above theorem, a practical way of setting κ is as follows.

$$\kappa_t = \frac{t}{b}, \quad (28)$$

where b is a constant. This setting adjusts the value of κ with transmission interval index t. Specifically, κ is set to a small value at initial stages to facilitate the full estimation of utility associated with each action. Then, when t becomes large, κ also becomes large, and the virtual agent intends to select the action with the largest utility estimation at this time, which helps improve its own performance. A similar idea is also adopted in [10].

VI. Simulation Results and Analysis

In this section, the convergence and effectiveness of the proposals are demonstrated, and corresponding simulation results are analyzed.

A. Basic Simulation Setting

The channel gain $h_{k'm_k}$ between FAP k' and UE m_k, $\forall k'$ and k, is composed of small scale fading and path loss with the path loss exponent $\alpha = 4$. More specifically, in each transmission interval, $h_{k'm_k}$ is randomly generated by $h_{k'm_k} = |g_{k'm_k}|^2 d_{k'm_k}^{-\alpha}$ with $g_{k'm_k} \sim \mathcal{CN}(0,1)$ and $d_{k'm_k}$ being the distance between FAP k' and UE m_k. In addition, each UE requests a content segment in each transmission interval by following Zipf distribution, and the probability of content segment f to be requested

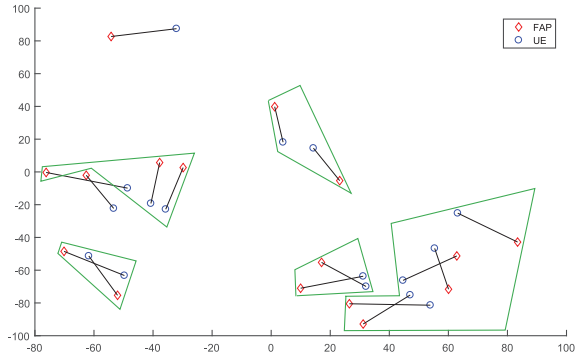

Fig. 5. The simulation scenario.

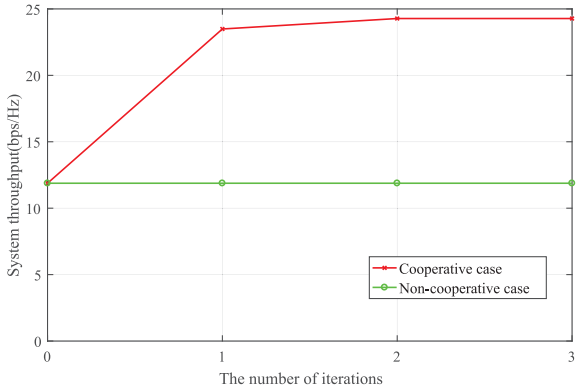

Fig. 6. The convergence of Algorithm 1.

is given by

$$\frac{1/f^\gamma}{\sum_{q=1}^F 1/q^\gamma}. \quad (29)$$

There are 100 content segments potentially requested by UEs, and the number of FAPs is 16. The noise power is set to -104 dBm, and the transmission power of each FAP is 30 dBm. Unless otherwise stated, simulation setting in the following is consistent with this subsection.

B. The Simulation of Cluster Formation Algorithm

First, we verify the convergence of Algorithm 1 for the scenario shown in Fig. 5 where the locations of FAPs are randomly generated by following uniform distribution within a circle of radius 100 m, and the location of each UE is generated in a similar way with the circle center being its associated FAP and the circle radius being 30 m. A channel gain matrix $\mathbf{H}$ and a user content request matrix $\mathbf{X}$ under $\gamma = 0.6$ are randomly generated. In addition, each FAP caches the most popular 20 content segments, and the number of content segments that can be downloaded by each FAP in each transmission interval is 2. From Fig. 6, it can be seen that Algorithm 1 converges within only three iterations, which allows FAPs to well adapt to the dynamic environment, and meanwhile a significant throughput improvement is observed compared to the case where each FAP only serves its own UE non-cooperatively. The cooperative FAPs

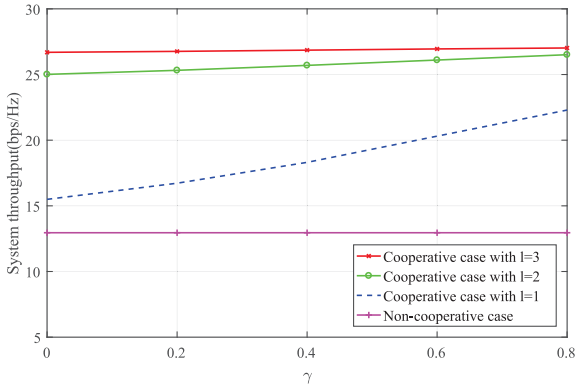

Fig. 7. The impact of fronthaul capacity with γ varying.

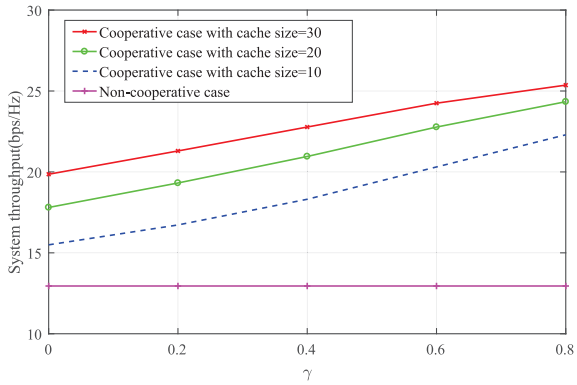

Fig. 8. The impact of cache size at each FAP with γ varying.

in the same cluster are highlighted using green polygons in Fig. 5.

Moreover, the impacts of fronthaul capacity and cache size at each FAP are investigated under various γ in Fig. 7 and 8, and these figures focus on the average performance of each case in one coherence time. Specifically, each data point is got by averaging throughput results over 10000 realizations of $\{\mathbf{H}, \mathbf{X}\}$. In each realization, the locations of each FAP and UE are randomly generated to avoid biased performance evaluation results caused by considering only a certain given network topology. Moreover, in Fig. 7, we assume the number of content segments that can be downloaded by each FAP in each transmission interval is the same, i.e., $l_k = l, \forall k$ in constraint (a) of (2), and l is varied from 1 to 3 to reflect different fronthaul capacity. Each FAP caches the most popular 10 content segments. From 7, it can be seen that a larger gamma leads to a higher throughput for a fixed l. This is because UEs will request fewer popular content segments, and hence their requests have higher probabilities to be locally met by FAPs, facilitating more cooperation among FAPs. Furthermore, under a fixed gamma, a larger l also contributes to more cooperation among FAPs, thus resulting in better system performance. However, the gain brought by increasing l becomes less for a larger γ. This is because more requests can be satisfied by local caches and the need for fronthaul capacity is less stringent.

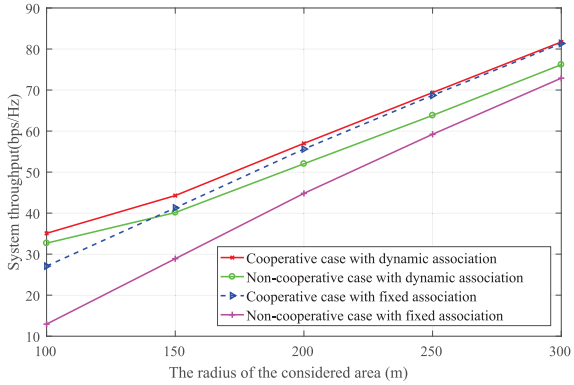

Fig. 9. The impact of dynamic association on system performance.

In Fig. 8, we set $l_k = l = 1, \forall k$, and each FAP caches the most popular F' segments with F' also representing the cache size. First, it still holds that a larger gamma brings about a higher throughput for a fixed cache size for the same reason stated before. Second, since a larger cache size makes FAPs have more chances to cooperate, system performance is raised.

At the end of this subsection, we further consider a comparison scheme, which is dynamic UE-FAP association based on matching theory [49]. The core idea is to see the UE-FAP association as a one-to-one matching between them and their preference over each other is based on current channel gain, which facilitates good adaptation to the varying radio environment. Note that in this scheme, FAPs are still non-cooperative, since each FAP only serves its own UE that is identified by the association result. Fig. 9 illustrates the performance comparison result among four schemes under different radius of the area where FAPs and UEs are distributed. These schemes include non-cooperative case with fixed association that is the same with non-cooperative case in previous figures, non-cooperative case with dynamic association, cooperative case with fixed association and cooperative case with dynamic association. In cooperative case with dynamic association, UE association is first adjusted by matching based dynamic association, whose result is taken as the initial association state for the proposed cluster formation algorithm.

From Fig. 9, it can be seen that non-cooperative case with dynamic association outperforms non-cooperative case with fixed association significantly. This is because the preference lists of UEs and FAPs are generated based on channel gains, and hence dynamic association can switch the link causing high interference in fixed association to a useful link for a UE. In addition, although non-cooperative case with dynamic association outperforms cooperative case with fixed association when the area radius is 100 m, cooperative case with fixed association has more superior performance when the network becomes less dense. For example, cooperative case with fixed association improves system throughput by 7.7% when the area radius is 250 m. Moreover, it is observed that cooperative case with dynamic association always has better performance than non-cooperative case with dynamic association, which demonstrates the necessity to conduct cluster formation. At last,

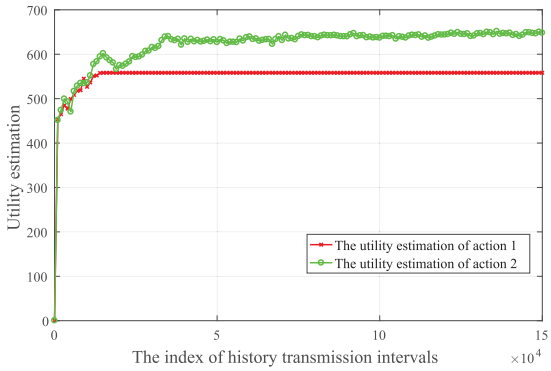

Fig. 10. The evolution of utility estimations of the 25th virtual agent.

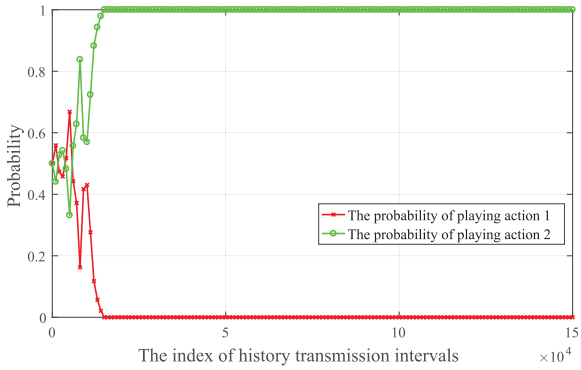

Fig. 11. The evolution of the strategy of the 25th virtual agent.

according to the performance comparison between cooperative cases with fixed association and dynamic association, adjusting FAP-UE association before cluster formation is beneficial to system performance.

C. The Simulation of Caching Algorithm

When the number of FAPs is 16, even if $F = 5$, there are around 10^{24} different $\mathbf{C}$. Considering the curse of dimensionality and the SARL adopted in Algorithm 2 is a well-known RL algorithm, we only evaluate the proposed MARL based caching algorithm, i.e., Algorithm 3. In this part, γ in Zipf distribution is assumed to be identical among UEs and is set to 0.8. In each transmission interval, each FAP can fetch only one content segment from the cloud via fronthaul, i.e, $l_k = 1$, $\forall k$. For the update equations in (9), learning parameter κ is taken as $t/10^5$, and α_t and λ_t are taken as $\frac{1}{(t+1)^{0.6}}$ and $\frac{1}{(t+1)^{0.7}}$, respectively. In addition, there are $F = 50$ content segments potentially requested by UEs, and that is $KF = 800$ virtual agents need to be created by the resource manager. The benefit ω of unit long-term system throughput is set to 40, and the cost μ of caching a content segment is set to 0.8. To use MARL to learn caching strategy, $15 * 10^4$ tuples of $\{\mathbf{H}, \mathbf{X}\}$ are generated and utilized to construct history network environment. For the considered $15 * 10^4$ transmission intervals, the locations of FAPs and UEs keep unchanged and follow the locations in Fig. 5.

Fig. 10 and 11 demonstrate the evolution of the utility estimation and strategy of the created, 25th virtual agent, respectively. Here, taking action 1 means caching the content segment with index 25 at FAP 1, while taking action 2 means the opposite. Initially, the utility estimation associated with each action is 0 and the virtual agent selects each action with equal probability. Then, with the increase of the transmission interval index, the parameter κ in (9) will continuously grow from a very small value to a larger one. Correspondingly, the virtual agent intends to explore the two actions and then prefers to select the action with larger utility estimation. Here, whenever the probability of selecting a certain action is over 0.99, the probability will be set to 1 in the subsequent learning process to achieve a deterministic caching matrix.

Then, to illustrate the superior performance of the caching strategy derived by MARL, we compare it with other several baselines.

- *Scheme 1:* In this scheme, each FAP caches all the content segments potentially requested by UEs.
- *Scheme 2:* In this scheme, there is no content segment cached at each FAP.
- *Scheme 3:* In this scheme, each FAP caches each content segment with probability of 0.5. Since this caching strategy is random, its performance is got by generating 1000 realizations of content caching matrix $\mathbf{C}$ and then taking the average performance.
- *Scheme 4:* In this scheme, distributed Q-learning based caching [17] is adopted. Specifically, the definition of agents, their action sets and the reward feedback follows MARL based caching. In addition, the algorithm procedure is also similar to MARL based caching except the ways of action selection and policy update. For action selection, the ε-greedy scheme [50] is adopted and agent i makes the following update after receiving the feedback signal [51]:

$$Q_{i,j}^{t+1} = \alpha_t I_{\left\{a_i^t = a_{i,j}\right\}} \left(r_{a_i^t} - Q_{i,j}^t\right) + Q_{i,j}^t, \tag{30}$$

  where $Q_{i,j}$ is the Q value associated with action $a_{i,j}$. After the algorithm terminates, the action with the highest Q value is identified as the best action for each agent and the final cache matrix is got based on the action profile composed of all these best actions.

To facilitate the comparison, another 1000 randomly tuples of $\{\mathbf{H}, \mathbf{X}\}$ are generated, which simulates 1000 future transmission intervals. At the beginning of this future period, caching matrix $\mathbf{C}$ is initialized for different caching schemes and kept fixed for the whole 1000 transmission intervals. Fig. 12 shows the evolution of the average system throughput that is calculated as

$$\frac{\sum_{t=1}^{t'} u_t}{t'}, \tag{31}$$

where t' is the index of the current transmission interval and u_t is the system throughput in transmission interval t. Once this metric is invariant, it can be seen as a good approximation of $\mathbb{E}_{\mathbf{H},\mathbf{X}}\left[u\left(\Psi_{\mathbf{C}}\left(\mathbf{H}, \mathbf{X}\right), \mathbf{H}\right)\right]$ and then the converged value can be used to calculate the utility of the resource manager in (3). Fig. 13 and Fig. 14 show the cost and achieved utility of the resource manager under different caching schemes, respectively.

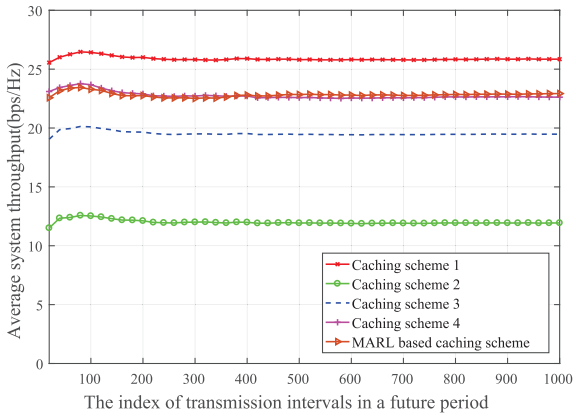

Fig. 12. The average throughput under different caching schemes.

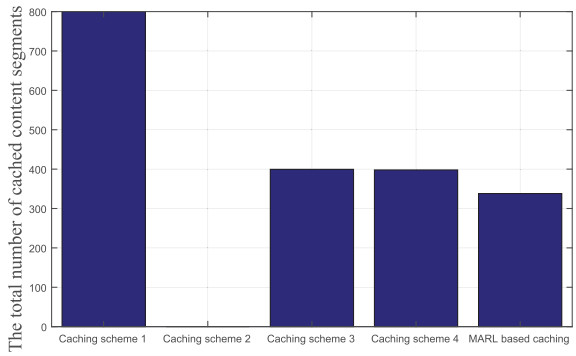

Fig. 13. The total number of cached content segments under different caching schemes.

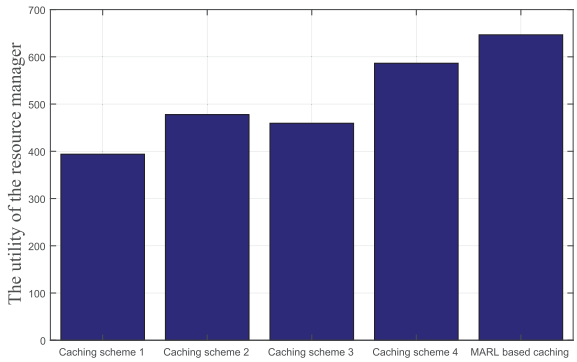

Fig. 14. The utility of the resource manager under different caching schemes.

According to Fig. 14, our proposal significantly outperforms *Scheme* 1–3. This is because MARL considers the optimization objective of the resource manager and the cluster formation behavior of FAPs by leaning from the feedback signal of the history environment, which is the key to make proper caching decisions. For *Scheme 1*, it achieves the best system throughput, since all the content segments are locally available and hence the system can fully enjoy the benefits brought by FAP cluster formation. However, the high cost led by caching all content segments at each FAP unavoidably harms the utility. For *Scheme 2*, it induces no caching cost but the system throughput is the worst due to much less FAP cooperation chances under very limited fronthaul capacity. Hence, the utility reached by *Scheme 2* is not satisfying. At last, relative to *Scheme* 4 that is based on distributed Q-learning, our proposal improves resource manager's utility by 10.3%, which again verifies its superiority.

VII. Conclusion

In this article, a joint cache and radio resource optimization problem has been studied for fog radio access networks, which includes a long-term caching optimization problem and a short-term cluster formation problem. Under a hierarchical resource management architecture, the considered problem has been further modeled as a Stackelberg game between the resource manager in the cloud and fog access points (FAPs), and the cluster formation behavior of FAPs is captured by a coalition formation game. To achieve Stackelberg equilibrium, a distributed cluster formation algorithm is first developed for FAPs to reach a stable partition under any fixed caching strategy of the resource manager. Then, faced with the challenges incurred by no closed form and 0–1 caching decision variables, two reinforcement learning algorithms are designed for the resource manager to approach an optimal or local optimal caching strategy. By simulations, the effectiveness of the proposals has been demonstrated.

References

[1] T. G. Rodrigues, T. G. Rodrigues, K. Suto, H. Nishiyama, N. Kato, and K. Temma, "Cloudlets activation scheme for scalable mobile edge computing with transmission power control and virtual machine migration," *IEEE Trans. Comput.*, vol. 67, no. 9, pp. 1287–1300, Sep. 2018.

[2] T. G. Rodrigues, K. Suto, H. Nishiyama, and N. Kato, "Hybrid method for minimizing service delay in edge cloud computing through VM migration and transmission power control," *IEEE Trans. Comput.*, vol. 66, no. 5, pp. 810–819, May 2017.

[3] D. Takaishi, Y. Kawamoto, H. Nishiyama, N. Kato, F. Ono, and R. Miura, "Virtual cell-based resource allocation for efficient frequency utilization in unmanned aircraft systems," *IEEE Trans. Veh. Technol.*, vol. 67, no. 4, pp. 3495–3504, Apr. 2018.

[4] F. Tang, Z. Md. Fadlullah, N. Kato, F. Ono, and R. Miura, "AC-POCA: Anti-coordination game based partially overlapping channels assignment in combined UAV and D2D based networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 2, pp. 1672–1683, Feb. 2018.

[5] M. Peng, S. Yan, K. Zhang, and C. Wang, "Fog computing based radio access networks: Issues and challenges," *IEEE Netw.*, vol. 30, no. 4, pp. 46–53, Jul. 2016.

[6] M. Tao, E. Chen, H. Zhou, and W. Yu, "Content-centric sparse multicast beamforming for cache-enabled cloud RAN," *IEEE Trans. Wireless Commun.*, vol. 15, no. 9, pp. 6118–6131, Sep. 2016.

[7] D. Chen and V. Kuehn, "Adaptive radio unit selection and load balancing in the downlink of fog radio access network," in *Proc. Global Commun. Conf.*, Washington, DC, USA, Dec. 2016, pp. 1–7.

[8] X. Peng, Y. Shi, J. Zhang, and K. B. Letaief, "Layered group sparse beamforming for cache-enabled green wireless networks," *IEEE Trans. Commun.*, vol. 65, no. 12, pp. 5589–5603, Dec. 2017.

[9] X. Yang, J. Zheng, Z. Fei, and B. Li, "Optimal file dissemination and beamforming for cache-enabled C-RANs," *IEEE Access*, vol. 6, pp. 6390–6399, Nov. 2017.

[10] Y. Sun, M. Peng, and H. Vincent Poor, "A distributed approach to improving spectral efficiency in uplink device-to-device enabled cloud radio access networks," *IEEE Trans. Commun.*, vol. 66, no. 12, pp. 6511–6526, Dec. 2018.

[11] M. Chen, M. Mozaffari, W. Saad, C. Yin, M. Debbah, and C. S. Hong, "Caching in the sky: Proactive deployment of cache-enabled unmanned aerial vehicles for optimized quality-of-experience," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 5, pp. 1046–1061, May 2017.
[12] M. Chen, W. Saad, C. Yin, and M. Debbah, "Echo state networks for proactive caching in cloud-based radio access networks with mobile users," *IEEE Trans. Wireless Commun.*, vol. 16, no. 6, pp. 3520–3535, Jun. 2017.
[13] S. M. S. Tanzil, W. Hoiles, and V. Krishnamurthy, "Adaptive scheme for caching YouTube content in a cellular network: Machine learning approach," *IEEE Access*, vol. 5, pp. 5870–5881, Mar. 2017.
[14] J. Song, J. Song, M. Sheng, T. Q. S. Quek, C. Xu, and Xijun Wang, "Learning-based content caching and sharing for wireless networks," *IEEE Trans. Commun.*, vol. 65, no. 10, pp. 4309–4324, Oct. 2017.
[15] Y. Xue, P. Zhou, S. Mao, D. Wu, and Y. Zhou, "Pure-exploration bandits for channel selection in mission-critical wireless communications," *IEEE Trans. Veh. Technol.*, vol. 67, no. 11, pp. 10995–11007, Nov. 2018.
[16] S. Hassan, S. Samarakoon, M. Bennis, M. Latva-aho, and C. S. Hong, "Learning-based caching in cloud-aided wireless networks," *IEEE Commun. Lett.*, vol. 22, no. 1, pp. 137–140, Jan. 2018.
[17] W. Wang, R. Lan, J. Gu, A. Huang, H. Shan, and Z. Zhang, "Edge caching at base stations with device-to-device offloading," *IEEE Access*, vol. 5, pp. 6399–6410, Mar. 2017.
[18] C. Zhong, M. Gursoy, and S. Velipasalar, "A deep reinforcement learning-based framework for content caching," in *Proc. 52nd Annu. Conf. Inf. Sciences Syst.*, Princeton, NJ, USA, Mar. 2018, pp. 1–6.
[19] C. Ma *et al.*, "Socially aware caching strategy in device-to-device communication networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 5, pp. 4615–4629, May 2018.
[20] X. Zhao, P. Yuan, H. li, and S. Tang, "Collaborative edge caching in context-aware device-to-device networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 10, pp. 9583–9596, Oct. 2018.
[21] L. Yao, L. Yao, A. Chen, J. Deng, J. Wang, and G. Wu, "A cooperative caching scheme based on mobility prediction in vehicular content centric networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 6, pp. 5435–5444, Jun. 2018.
[22] Z. Su, Z. Su, Y. Hui, Q. Xu, T. Yang, J. Liu, and Y. Jia, "An edge caching scheme to distribute content in vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 6, pp. 5346–5356, Jun. 2018.
[23] Y. Chen, Y. Chen, M. Ding, J. Li, Z. Lin, G. Mao, and L. Hanzo, "Probabilistic small-cell caching: Performance analysis and optimization," *IEEE Trans. Veh. Technol.*, vol. 66, no. 5, pp. 4341–4354, May 2017.
[24] Y. Wang, X. Tao, X. Zhang, and G. Mao, "Joint caching placement and user association for minimizing user download delay," *IEEE Access*, vol. 4, pp. 8625–8633, Dec. 2016.
[25] A. Liu and V. K. N. Lau, "Mixed-timescale precoding and cache control in cached MIMO interference network," *IEEE Trans. Signal Process.*, vol. 61, no. 24, pp. 6320–6332, Dec. 2013.
[26] A. Liu and V. K. N. Lau, "Cache-enabled opportunistic cooperative MIMO for video streaming in wireless systems," *IEEE Trans. Signal Process.*, vol. 62, no. 2, pp. 390–402, Jan. 2014.
[27] 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.
[28] J. Tang, L. Teng, T. Q. S. Quek, T.-H. Chang, and B. Shim, "Exploring the interactions of communication, computing and caching in cloud RAN under two timescale," in *Proc. IEEE 18th Int. Workshop Signal Process. Advances Wireless Commun.*, Sapporo, Japan, Jul. 2017, pp. 1–6.
[29] A. Liu and V. K. N. Lau, "Two-timescale user-centric RRH clustering and precoding optimization for cloud RAN via local stochastic cutting plane," *IEEE Trans. Signal Process.*, vol. 66, no. 1, pp. 64–76, Jan. 2018.
[30] Z. Han, Z. Han, D. Niyato, W. Saad, T. Baar, and A. Hjrungnes, *Game Theory in Wireless and Communication Networks*, Cambridge, U.K.: Cambridge Univ. Press, 2012.
[31] R. Sun, H. Baligh, and Z. Q. Luo, "Long-term transmit point association for coordinated multipoint transmission by stochastic optimization," in *Proc. 14th Workshop Signal Process. Advances Wireless Commun.*, Darmstadt, Germany, Jun. 2013, pp. 330–334.
[32] F. Pantisano, M. Bennis, W. Saad, M Debbah, and M. Latva-aho, "Interference alignment for cooperative femtocell networks: A game-theoretic approach," *IEEE Trans. Mobile Comput.*, vol. 12, no. 11, pp. 2233–2246, Nov. 2013.
[33] J. Moon and D. Cho, "Efficient cell-clustering algorithm for inter-cluster interference mitigation in network MIMO systems," *IEEE Commun. Lett.*, vol. 15, no. 3, pp. 326–328, Mar. 2011.
[34] S. Bassoy, H. Farooq, M. A. Imran, and A. Imran, "Coordinated multi-point clustering schemes: A survey," *IEEE Commun. Surv. Tuts.*, vol. 19, no. 2, pp. 743–764, Second Quarter, 2017.
[35] R. Tanbourgi, S. Singh, J. G. Andrews, and F. K. Jondral, "Analysis of non-coherent joint-transmission cooperation in heterogeneous cellular networks," in *Proc. IEEE Int. Conf. Commun.*, Sydney, Australia, Jun. 2014, pp. 5160–5165.
[36] M. Ahmed, M. Peng, M. Abana, S. Yan, and C. Wang, "Interference coordination in heterogeneous small-cell networks: A coalition formation game approach," *IEEE Syst. J.*, vol. 12, no. 1, pp. 604–615, Mar. 2018.
[37] D. Wu, Y. Cai, R. 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.
[38] D. Liu, S. Han, C. Yang, and Q. Zhang, "Semi-dynamic user-specific clustering for downlink cloud radio access network," *IEEE Trans. Veh. Technol.*, vol. 65, no. 4, pp. 2063–2077, Apr. 2016.
[39] L. Liu, Y. Zhou, V. Garcia, L. Tian, and J. Shi, "Load aware joint CoMP clustering and inter-cell resource scheduling in heterogeneous ultra dense cellular networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 3, pp. 2741–2755, Mar. 2018.
[40] X. Kang, R. Zhang, and M. Motani, "Price-based resource allocation for spectrum-sharing femtocell networks: A Stackelberg game approach," *IEEE J. Sel. Areas Commun.*, vol. 30, no. 3, pp. 538–549, Apr. 2012.
[41] C. Sun, M. Peng, Y. Sun, Y. Li, and J. Jiang, "Distributed power control for device-to-device network using Stackelberg game," in *Proc. Wireless Commun. Netw. Conf.*, Istanbul, Turkey, May 2014, pp. 1344–1349.
[42] X. Tang, P. Ren, Y. Wang, and Z. Han, "Combating full-duplex active eavesdropper: A hierarchical game perspective," *IEEE Trans. Commun.*, vol. 65, no. 3, pp. 1379–1395, Mar. 2017.
[43] H. Zhang, Y. Xiao, L. X. Cai, D. Niyato, L. Song, and Z. Han, "A multi-leader multi-follower Stackelberg game for resource management in LTE unlicensed," *IEEE Trans. Wireless Commun.*, vol. 16, no. 1, pp. 348–361, Jan. 2017.
[44] L. Tang, X. Chen, and S. He, "When social network meets mobile cloud: A social group utility approach for optimizing computation offloading in cloudlet," *IEEE Access*, vol. 4, pp. 5868–5879, Sep. 2016.
[45] S. M. Perlaza, H. Tembine, and S. Lasaulce, "How can ignorant but patient cognitive terminals learn their strategy and utility?," in *Proc. 11th Int. Workshop Signal Process. Advances Wireless Commun.*, Marrakesh, Morocco, Jun. 2010, pp. 1–5.
[46] D. S. Leslie and E. J. Collins, "Convergent multiple-timescales reinforcement learning algorithms in normal form games," *Ann. Appl. Probab.*, vol. 13, pp. 1231–1251, Feb. 2002.
[47] J. Hofbauer and W. H. Sandholm, "On the global convergence of stochastic fictitious play," *Econometrica*, vol. 70, no. 6, pp. 2265–2294, 2002.
[48] J. Hofbauer and E. Hopkins, "Learning in perturbed asymmetric games," *Ese Discuss. Papers*, vol. 52, no. 1, pp. 133–152, 2000.
[49] Y. Gu, W. Saad, M. Bennis, M. Debbah, and Z. Han, "Matching theory for future wireless networks: Fundamentals and applications," *IEEE Commun. Mag.*, vol. 53, no. 5, pp. 52–59, May 2015.
[50] 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. 1960–971, Apr. 2019.
[51] C. Fan, B. Li, C. Zhao, W. Guo, and Y.-C. 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.

Yaohua Sun received the bachelor's degree (with first class Hons.) in telecommunications engineering (with management) and the Ph.d. degree in communication engineering both from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2014 and 2019, respectively. He is currently an Assistant Professor with the State Key Laboratory of Networking and Switching Technology (SKL-NST), BUPT. His research interests include IoT, edge computing, resource management, (deep) reinforcement learning, network slicing, and fog radio access networks. He was the recipient of the National Scholarship in 2011 and 2017, and he has been a Reviewer for IEEE TRANSACTIONS ON COMMUNICATIONS, IEEE TRANSACTIONS ON MOBILE COMPUTING, IEEE SYSTEMS JOURNAL, JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, IEEE COMMUNICATIONS MAGAZINE, IEEE WIRELESS COMMUNICATIONS MAGAZINE, IEEE WIRELESS COMMUNICATIONS LETTERS, IEEE COMMUNICATIONS LETTERS, and IEEE INTERNET OF THINGS JOURNAL.

Mugen Peng (M'05–SM'11) received the Ph.D. degree in communication and information systems from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2005. Afterward, he joined BUPT, where he has been a Full Professor with the School of Information and Communication Engineering, since 2012. During 2014, he was also an Academic Visiting Fellow at Princeton University, USA. He leads a Research Group focusing on wireless transmission and networking technologies in BUPT. He has authored and coauthored over 90 refereed IEEE journal papers and over 300 conference proceeding papers. His main research interests include wireless communication theory, radio signal processing, cooperative communication, self-organization networking, heterogeneous networking, cloud communication, and Internet of Things.

Dr. Peng was a recipient of the 2018 Heinrich Hertz Prize Paper Award, the 2014 IEEE ComSoc AP Outstanding Young Researcher Award, and the Best Paper Award in the JCN 2016, IEEE WCNC 2015, IEEE GameNets 2014, IEEE CIT 2014, ICCTA 2011, IC-BNMT 2010, and IET CCWMC 2009. He is currently or have been on the editorial/associate editorial board of the IEEE COMMUNICATIONS MAGAZINE, IEEE ACCESS, IEEE INTERNET OF THINGS JOURNAL, IET COMMUNICATIONS, and *China Communications*.

Shiwen Mao (S'99–M'04–SM'09–F'19) received the Ph.D. degree in electrical and computer engineering from Polytechnic University, Brooklyn, NY, USA, in 2004. He is the Samuel Ginn Distinguished Professor and Director of the Wireless Engineering Research and Education Center (WEREC) at Auburn University, Auburn, AL, USA. His research interests include wireless networks, multimedia communications and smart grid. He is a Distinguished Speaker of the IEEE Vehicular Technology Society. He is on the Editorial Board of IEEE TRANSACTIONS ON MOBILE COMPUTING, IEEE TRANSACTIONS ON MULTIMEDIA, IEEE INTERNET OF THINGS JOURNAL, IEEE MULTIMEDIA, IEEE NETWORKING LETTERS, *ACM GetMobile*, among others. He received the Auburn University Creative Research & Scholarship Award in 2018 and NSF CAREER Award in 2010, as well as the 2017 IEEE ComSoc ITC Outstanding Service Award, the 2015 IEEE ComSoc TC-CSR Distinguished Service Award, the 2013 IEEE ComSoc MMTC Outstanding Leadership Award, and the NSF CAREER Award in 2010. He is a co-recipient of the IEEE ComSoc MMTC Best Journal Paper Award in 2019 and Best Conference Paper Award in 2018, the Best Demo Award from IEEE SECON 2017, the Best Paper Awards from IEEE GLOBECOM 2016 & 2015, IEEE WCNC 2015, and IEEE ICC 2013, and the 2004 IEEE Communications Society Leonard G. Abraham Prize in the Field of Communications Systems.