Stochastic Game based Cooperative Alternating Q-Learning Caching in Dynamic D2D Networks

Tiankui Zhang, *Senior Member, IEEE,* Xinyuan Fang, Ziduan Wang, Yuanwei Liu, *Senior Member, IEEE,* Arumugam Nallanathan, *Fellow, IEEE*

***Abstract*—Edge caching has become an effective solution to cope with the challenges brought by the massive content delivery in cellular networks. In device-to-device (D2D) enabled caching cellular networks with time-varying content popularity distribution and user terminal (UT) location, we model these dynamic networks as a stochastic game to design a cooperative cache placement policy. We consider the long-term cache placement reward of all UTs in this stochastic game, where each UT becomes an agent and the cache placement policy corresponds to the actions taken by the UTs. Each UT has the same immediate network reward from content caching and sharing. In an effort to solve the stochastic game problem, we propose a multi-agent cooperative alternating Q-learning (CAQL) based cache placement algorithm. In CAQL, each UT alternatively updates its own cache placement policy according to the stable policy of other UTs during the learning process, until the stable cache placement policy of all the UTs in the cell is obtained. We discuss the convergence and complexity of CAQL, which obtains the stable cache placement policy with low space complexity. Simulation results show that the proposed algorithm can effectively reduce the backhaul load and the average content access delay in dynamic networks.**

***Index Terms*—Cache placement, device-to-device communication, edge caching, stochastic game**

I. Introduction

With the development of mobile network technologies and the popularization of mobile Internet applications, the mobile Internet data traffic and the content diversity have grown explosively recent years. The demand for emerging services such as wireless video transmission, Internet of things (IoT), and automated production will generate more traffic, which is expected to reach 49 Exabytes per month by the end of 2021 [2]. Studies have shown that a large part of traffic in the mobile Internet is generated by repeated transmission of the same highly-popular content [3], that is, the online video and social service have the characteristic of asynchronous content reuse. This characteristic leads to increased pressure on the backhaul link and reduces the user service quality in mobile cellular networks [4].

In order to meet the demands for saving backhaul resources and improving user service quality in mobile cellular networks, the concept of edge caching has been proposed [5, 6], even for unmanned aerial vehicles (UAV) assisted cellular networks [7, 8]. With the increase of the UT storage space, the UT edge caching, which stores the contents in some UTs by means of prefetching, and then share the contents among UTs by device-to-device (D2D) communications [9], has attracted the attention of many researchers. As one of the emerging technologies in the era of 5G communication, D2D communication can utilize the radio frequency band resources in the cellular networks to realize direct communication between LTE terminals within a certain range [10]. D2D communications in cellular networks can increase network throughput, reduce energy consumption, and improve spectrum utilization [11].

Due to the limited storage space of the UT, the cache placement strategy is very important in UT edge caching based on D2D communication. A suitable cache placement strategy can effectively improve the performance of the UT edge caching [12–15]. In [12], a wireless video storage distribution architecture utilizing D2D communication in small base stations was first proposed. The architecture aims to improve video throughput and achieve the purpose of replacing the backhaul link with edge caching. A cache placement algorithm that minimizes the average caching failure rate is proposed in [13], which selects some mobile terminals as service nodes. In [14], researchers use the relationship between the physical layer and the social layer to design a content cache placement strategy to maximize the benefits of the community. In [15], the cache placement is obtained according to the content popularity to maximize the total offloading probability of D2D systems.

Most of the existing researches on UT edge caching are studied in a static model. These studies usually assume that the location topology of the UTs and the popularity of the contents are fixed. Although these assumptions can simplify the problem and facilitate the application of the optimization theory, the environmental factors in the actual situation of cache placement are mostly time-varying. Some studies have considered the impact of user mobility on cache placement strategies [16, 17]. A framework of mobility-aware coded caching has been developed in [16]. The authors in [17] take advantage of the user mobility pattern by the inter-contact times between different users, then propose a mobility-aware cache placement strategy to maximize the data offloading ratio. Some studies also consider designing the cache placement strategy with time-varying popularity profiles [18, 19]. A dynamic edge caching framework has been proposed in [18], which predicts the future requirements of users based on the collected data of past users' demands. The authors in [19] maximizes the offloading probability for cache-enabled D2D communication by exploiting individual user behavior in

Part of this work has been presented at the IEEE Global Communications Conference (GLOBECOM), HI, USA, DEC., 2019 [1].

T. Zhang, X. Fang and Z. Wang are with Beijing University of Posts and Telecommunications, Beijing, China (e-mail: {zhangtiankui, fangxinyuan, wangziduan}@bupt.edu.cn).

Y. Liu and A. Nallanathan are with Queen Mary University of London, London, U.K. (e-mail: {yuanwei.liu, a.nallanathan}@qmul.ac.uk).

sending requests. The time-varying user location and content popularity require intelligent cache placement mechanisms that can adapt to these variations.

Stochastic game is a kind of dynamic game with one or more participants whose state will be probabilistically transferred [20], which has been widely accepted as an effective mathematical tool for solving the dynamic radio resource allocation problem [21]. In practical scenarios, participants usually do not know the transition probability between states in the stochastic game, so the reinforcement learning is a common method to deal with the stochastic game with dynamic environments [22]. Reinforcement learning has been an emerging tool to tackle problems encountered in computation offloading [23], resource allocation [24, 25], and edge caching [26–32] in cellular networks. A learning-based approach to store contents in heterogenous networks has been proposed in [28], which considers the time-varying popularity of unknown cache content and estimates them using their proposed transfer learning-based approach. A Q-learning based BS caching and D2D offloading is proposed in [29], which applies Q-leaning to design a distributed cache placement strategy according to content popularity. In [30], a Q-learning algorithm is developed for finding the best caching policy in an online fashion. In [31], the D2D caching problem is modeled as a multi-agent multi-armed bandit problem, which is solved by stateless Q-learning to coordinate the caching decisions. A multi-agent reinforcement learning with non-perfect content popularity has been designed In [32]. These studies have taken into account the unknown content popularity, but do not consider the time-varying content popularity and UT location in practical scenarios.

A. Motivation and Contribution

As mentioned above, in dynamic networks, most of the existing research contributions aim at solving the problem of edge cache placement under dynamic changes in users' requirements, but rarely consider their mobility. The studies on UT edge cache placement problem are also mostly consider single-user or single-state situations. This article proposes a dynamic cache placement algorithm based on reinforcement learning for multi-content multi-UT caching in a dynamic environment with time-varying content popularity and UT location. We model the cache placement process as a stochastic game, where each UT becomes a agent and the cache placement policy corresponds to the actions taken by the UTs. In order to solve this stochastic game, we propose a multi-agent cooperative alternating Q-learning (CAQL) algorithm. The main contributions of this paper are summarized as follows,

- We model the multi-content multi-UT cache placement process in D2D-enabled caching cellular networks as a fully cooperative stochastic game between UTs. The states of the stochastic game include joint information of content popularity and the location of the UTs in the cellular networks. When formulating the value function of the stochastic game, we consider the content caching and sharing incentive and the content delivery cost of UTs. We maximize the long-term joint reward function as the value function. We discuss the existence of this stochastic game equilibrium.
- We propose a multi-agent CAQL algorithm based on optimal response of each UT to solve the stochastic game problem. In CAQL, all UTs know the stable cache placement policy of others since the stochastic game is modeled as a cooperative game. Each UT learns alternatively to obtain the best response policy. The proposed CAQL algorithm can obtain the best cache placement policy considering long-term rewards.
- We discuss the convergence and spatial complexity of the proposed CAQL algorithm. We prove that CAQL eventually converges to a stable cooperative cache placement policy that aims to maximize the value function in the joint state-action space. We also discuss the space complexity according to the characteristics of the CAQL algorithm. We show that the space complexity of CAQL is lower compared with the traditional multi-agent Q-learning.
- We show that the CAQL based cache placement algorithm can effectively reduce the average content access delay of UTs and the traffic pressure of backhaul in dynamic networks with time-varying UT location and content popularity.

B. Organization and Notations

The rest of the paper is organized as follows. In Section II, D2D-enabled caching cellular networks considering UT mobility is introduced. The model of the stochastic game for cache placement is presented in Section III. The cooperative alternating Q-learning is addressed in Section IV. Simulation results are shown in Section V and conclusions are finally drawn in Section VI. The main symbols and variables used in this paper are summarized in Table I.

TABLE I: Main Symbol and Variable List

Parameter	Description
K	number of contents in one macrocell
N	number of UTs in one macrocell
$C^{(t)}$	D2D communication relationship at time slot t
$\mathcal{C} = \{C_1, C_2, ..., C_I\}$	D2D communication relationship Markov state set
$b_n^{\left(C^{(t)}\right)}$	The data rate of UT n in D2D communication relationship $C^{(t)}$
$h_k^{(t)}$	the popularity of content chunk k at time slot t
$\mathcal{H} = \{H_0, H_1, ..., H_I\}$	content popularity Markov state set
$S^{(t)}$	the state of network at time slot t
a_n	the action of player UT n
$\mathbf{a} = [a_1, a_2, ..., a_N]$	all UTs joint actions
$\mathcal{A}_n = \{1, 2, ..., K\}$	the set of actions that UT n might choose
$\pi_n\left(S^{(t)}\right)$	the cache placement policy of UT n

II. System Model

In this section, we first introduce the D2D-enabled caching cellular networks and the communication model between the UTs. The caching and incentive models are then introduced to illustrate how the UTs overcome selfishness to participate in content caching and sharing.

A. D2D-Enabled Caching Cellular Networks

We consider multimedia contents distribution in D2D-enabled caching cellular networks. There is one macro BS and a set of UTs $\mathcal{N}=\{1,2,...N\}$ in the cell, as shown in Fig. 1. The BS is connected to the backbone network through backhaul links. We assume that the research scenario is for UTs to move within a fixed area, such as a university campus or a corporate campus. We refer to the multi-home-point movement model in [33]. This movement model assumes that each UT has one or more home-points and the UT's activities are centered around these home-points. As the time series progresses, the UTs periodically move between different home-points. We assume that the time that UT transfers between home-points is ignored, which means that communication does not occur during the transfer process. Since the time that UTs transfer between home-points is very short compared to the time the UTs are in their home-point ranges, we assume that the content sharing during the transfer process is ignored. At the same time, we assume that there are several clusters with high home-point density in the research area, such as dormitories, teaching buildings, or canteens on campus. Therefore, the transfer of UTs between home-points can be considered as a transfer between these clusters. If a user often leaves its resident area, it will have less chance of content sharing, since the UTs' revenue of content sharing mainly comes from the frequently content sharing between them by BS incentivization.

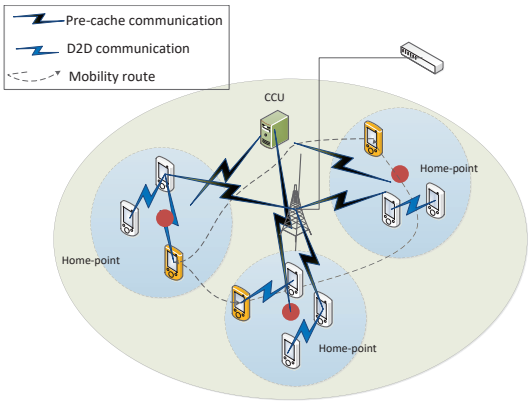

Fig. 1: D2D-enabled caching cellular networks.

We consider a long-term cache placement problem, which is made by a caching control unit (CCU) located in the BS. The entire cache placement process is represented by the time series $\mathcal{T}=\{1,2,...,t,...\}$, where t represents a time slot. Each UT executes a pre-cache at the beginning of each time slot and then performs content sharing via D2D communications. As shown in Fig. 2, the pre-cache part is divided into three phases. In the information exchange phase, UTs upload their current location information to the CCU. By this way, the CCU knows the state of the UTs in the cell. In the cache placement decision phase, the CCU executes a cache placement decision. In the cache delivery phase, the BS transmits multiple contents to multiple UTs according to the cache placement policy determined by the CCU.

Since one of the goals of cache placement is to reduce the peak backhaul load, we can assume that each caching time slot t begins in the off-peak phase of the network, and the length of t can be various. The start time of t can be specified as a recognized low load period, such as three o'clock in the morning or three o'clock in the afternoon. It can also be specified by the operator according to the actual situation of the network load. It means that the pre-cache part starts in the off-peak period, and the content sharing part runs through the peak period of the network until the next low peak period arrives.

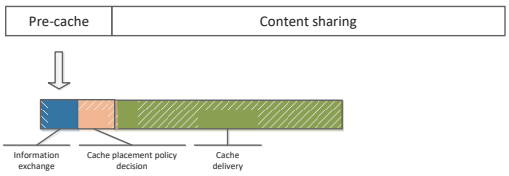

Fig. 2: Structure of caching time slot.

B. Communication Model

We assume that both the cellular UTs and the D2D UTs use non-overlapping orthogonal radio resources, so the interference between the cellular-D2D UTs and the D2D-D2D UTs can be ignored. We assume that each UT can be either a transmitter or a receiver in D2D communication. To quantify the D2D communication relationship between the UTs in the network, we define a received signal power threshold η. Due to the existence of the threshold η , the D2D communication relationship between the UTs is divided into two states: communicable and non-communicable. The specific communication state is determined by the topology relationship of the UTs under the mobility model. The D2D communication relationship between N UTs at time t can be quantified as a finite Markov state transition model $C^{(t)} \in \mathcal{C}=\{C_1,C_2,...,C_I\}$, which measured by the location relationship between them. $C^{(t)}$ is an $N \times N$ matrix, when the received signal power is greater than a received signal power threshold η, $c_{n,n'}^{(t)}=1$, which means that UT n and n' are considered capable of D2D communication, otherwise, $c_{n,n'}^{(t)}=0$. $\mathcal{C}$ represents the set of all states of possible UTs' communication relationships.

In each state, we assume that the transmitter shares the content to the receivers by broadcast. The data rate that UT n broadcast to other UTs in its D2D communication range at time slot t in D2D communication relationship $C^{(t)}$ is

$$b_n^{(C^{(t)})}=\frac{B_D}{N_D}\log\left(1+\frac{g_{n,n'}^{(C^{(t)})}P_n}{\sigma^2}\right), \quad (1)$$

where n' denotes the farthest UT that can perform D2D communication with UT n, B_D denotes the D2D communication bandwidth, N_D is the number of transmitters in D2D communication model, P_n denotes the transmission power of UT n, σ^2 denotes the additive white Gaussian noise power, and $g_{n,n'}^{(C^{(t)})}$ represents channel gain between UT n and n' at time t in D2D communication relationship $C^{(t)}$.

The data rate from the BS to UT n at time t in D2D communication relationship $C^{(t)}$. can be calculated similarly,

$$b_{BS,n}^{(C^{(t)})} = \frac{B_B}{N_B} \log \left(1 + \frac{g_{BS,n}^{(C^{(t)})} P_{BS}}{\sigma^2} \right), \tag{2}$$

where B_B denotes the BS communication bandwidth, N_B is the number of cellular UTs receiving data from the BS, P_{BS} denotes the transmission power of the BS, and σ^2 denotes the additive white Gaussian noise power, and $g_{BS,n}^{(C^{(t)})}$ represents channel gain between the BS and UT n at time t in D2D communication relationship $C^{(t)}$.

As the time series progresses, UTs movement makes the communication relationship $C^{(t)}$ changing, which transfers with a fixed transition probability between different states $\mathbb{P}\{C^{(t+1)}|C^{(t)}\}$.

C. Caching Model

In the mobile networks, we assume that each content file is divided into several unified chunks. Every chunk is the minimal unit of data to be transferred over the network. The size of each content chunk is s. It means that different content files with various data sizes can be divided into various chunks. We assume that each cell has a set of content chunks $\mathcal{K} = \{1, 2, ...K\}$ to be cached, which can be selected according to the user preference in the cell [34]. Since the UTs' caching spaces are limited, we assume that each UT can only store a fewer number of content chunks. For simplicity but without loss of generality, it is assumed that on UT can cache one content chunk with size of s. If a UT has more cache space for more than one chunks, it can be equivalent to multiple UTs with unified cache store size s. For example, UT n has $2s$ cache size, it is treated as UT n_1 cached content k_1 and UT n_2 cached content k_2. In this case, the set of UTs will be $\mathcal{N}$= $\{1, 2, ..., n_1, n_2, ...N\}$ in the cell. A special case is that, UT n_1 intends to receive content k_2 from n_2, which is modeled as $c_{n_1,n_2}^{(t)} = 1$.

We assume that the popularity h_k of content chunk $k(k \in \mathcal{K})$ is time-varying, which is modeled as a finite state Markov sequence. For all content chunks $k \in \mathcal{K}$, $h_k^{(t)} \in \mathcal{H} = \{H_0, H_1, ..., H_I\}$, where $\mathcal{H}$ is the content popularity state set. The popularity of the content chunk k transfers over time between the states, and the transition probability is represented by $\mathbb{P}\{h_k^{(t+1)}|h_k^{(t)}\}$. We assume that in each caching time slot t, the communication relationship $C^{(t)} \in \mathcal{C}$ between the UTs and the content popularity $h_k^{(t)} \in \mathcal{H}$ are constant, and only changes to the next time slot.

D. Incentive Model

In practical applications, UTs are selfish, that is, they only pre-cache the contents according to their own interest, and do not care about the requirements of the surrounding UTs. This kind of selfishness is not conducive to content sharing via D2D communications. To encourage the D2D communications of UTs, the BS rewards each UT that successfully delivers the cached contents to its surrounding UTs. For the BS, providing rewards can maximize the role of D2D communications in replacing the backhaul link during peak hours. For the UTs, pre-caching based on the content requirements of surrounding UTs allows them to obtain incentive from the BS, increase cache space utilization and reduce the content access delay during peak hours. Therefore, both UTs and the BS have motivation to participate in the cache placement policy.

III. Stochastic Game of Cache Placement

In cellular networks, the content popularity and the communication relationship decided by the UTs' location are time-varying. Our goal is to derive a cache placement policy that maximizes UT revenue in this dynamic network environment, while reducing the traffic load of the backhaul. Based on the feature of multi-user participation and the time-varying of the environment, we can model the cache placement problem as a stochastic game between UTs. In this section, we first describe the structure and characters of the fully cooperative stochastic game. Then we introduce the value function considering the long-term reward of the game. We also illustrate the relationship between the best cache placement policy and the equilibrium of the game.

A. Preliminaries

Stochastic game is a dynamic game with one or more participants and a state-to-state probability transfer in game theory [20]. The stochastic game has the characteristics of multiple agents and multiple states. It assumes that all agents participating in the game can observe the complete states, while the transition probability of the state and the reward of the agent depend on the joint action of all agents.

The structure of the stochastic game of cache placement is $\mathcal{G} = \{\mathcal{T}, \mathcal{N}, \mathcal{S}, \mathbb{P}, \mathcal{A}_n, V\}$, where $\mathcal{N}$ denotes the set of agents participating in the stochastic game. In this paper, we consider the UTs in the cell that can perform D2D communication as the agents of the game. $\mathcal{T}$ denotes the time series as we mentioned in Section II. In each time slot t, the network state is defined as $S^{(t)} = [C^{(t)}, \mathbf{h}^{(t)}]$, where $C^{(t)}$ denotes the D2D communication relationship between UTs in time slot t, and $\mathbf{h}^{(t)} = \left[h_1^{(t)}, h_2^{(t)}, ..., h_K^{(t)}\right]$ denotes the popularity set of the K contents in time slot t. We define $\mathcal{S} = \mathcal{H}^K \otimes \mathcal{C}$ as a collection of all the possible states in the dynamic networks, so $S^{(t)} \in \mathcal{S}$. $\mathbb{P}\left\{\mathbf{h}^{(t+1)}|\mathbf{h}^{(t)}\right\} = \prod_{k=1}^{K} \mathbb{P}\{h_k^{(t+1)}|h_k^{(t)}\}$ represents the transition popularity of the K content chunks. We assume that the action of each player UT n is what it will cache at the next time slot, so $a_n \in \mathcal{A}_n$, $\mathcal{A}_n = \{1, 2, ..., K\}$ denotes the set of actions that UT n might choose. We define the vector $\mathbf{a} = [a_1, a_2, ..., a_N]$ to represent the joint action of all UTs in the cell. V denotes the value function, which we will discuss in detail in Part C.

According to the states of the dynamic networks and the action of UTs, we get the possibilities of each UT choosing different actions under different network states, which are defined as the cache placement policy of UT n,

$$\begin{aligned} &\pi_n\left(S^{(t)}\right) = \left[\pi_n\left(1|S^{(t)}\right), ..., \pi_n\left(K|S^{(t)}\right)\right] \\ &\text{s.t.} \quad \pi_n\left(a_n|S^{(t)}\right) \geq 0 \quad \forall a_n \in A_n, \\ &\qquad \sum_{a_n \in A_n} \pi_n\left(a_n|S^{(t)}\right) = 1, \end{aligned} \tag{3}$$

where the constraints indicate that the probability that the UT selects content k to cache under state $S^{(t)}$ is not less than 0, and the sum is 1. In this paper, we assume that the UTs choose the next action according to different network states, which is actually independent of time slot t. It means that regardless of which time slot t a UT reaches a certain state, its policy is the same. Then we can get the transition probability between state $S^{(t)}$ and $S^{(t+1)}$ under action a_n,

$$\mathbb{P}\left(S^{(t+1)}|S^{(t)},a_n\right)=\sum_{a_{-n}}\pi^*_{-n}\left(a_{-n}|S^{(t)}\right)\mathbb{P}\left(S^{(t+1)}|S^{(t)},\mathbf{a}\right), \tag{4}$$

where $\mathbb{P}\left(S^{(t+1)}|S^{(t)},\mathbf{a}\right)=\mathbb{P}\{C^{(t+1)}|C^{(t)}\}\mathbb{P}\{h^{(t+1)}|h^{(t)}\}$, $\mathbf{a}$ denotes the joint action set of all UTs and π^*_{-n} denotes the stable cache placement policy except UT n.

B. Fully Cooperative Stochastic Game

In a stochastic game, if the rewards generated by all agents are the same ($r_1=r_2=...=r_n=r$), then the game is a fully cooperative stochastic game (FCSG) [35]. In the FCSG, the best-reward action of one agent is also a best-reward action of every agent, which means the best-reward action of one agent is in line with the overall interests of all agents [36]. In this kind of cooperative multi-agent system, agents tend to act as a group. In this paper, the CCU collects the state of the UTs in the cell and executes the cache placement policy decision. So the FCSG is operated by the CCU as a master node, which simulates the learning process of multiple agents. Therefore, in order to facilitate the subsequent learning process, we propose two assumptions based on the cooperative strategy of FCSG as follows.

Assumption 1. *All UTs have the same goal and the same reward distribution.*

Assumption 2. *All UTs can always observe the status and state transition of other UTs, that is, the network state is known to all UTs.*

Assumption 1 can use the advantage of FCSG by unifying the rewards of all UTs, thus avoiding the problem of revenue distribution among UTs. **Assumption 2** allows each UT to perceive other UTs' actions over time through the cooperative strategy. Under these assumptions, concurrent learning is still a non-trival problem. Because in the multi-agent learning process, the immediate reward of each UT is not only related to the current network state, but also related to the action of other UTs. In each time slot, for the UT, the randomness of other UT's action policies will lead to the uncertainty of its immediate reward. This makes it difficult to guarantee theoretical convergence for distributed model-free learning, which is widely used in deep learning, although many existing studies have empirically proved the effectiveness of such distributed learning [36].

Therefore, we need to consider optimizing the cooperative strategy of the FCSG. For cooperative learning, the key point is the coupling of value function and learning exploration strategy. The value function is related to the joint-action of all UTs and will change as the exploration strategy changes. Next, we will discuss the setting of the reward and value function of each UT.

C. Reward and Value Function

A reasonable value function is the key to the game. In our game, the value function is related to the immediate reward of environmental feedback due to the action of each UT. We define that under state $S^{(t)}$, the immediate network reward obtained by the joint action $\mathbf{a}$ of all UTs in the cell is $r\left(S^{(t)},\mathbf{a}\right)$.

Since each UT is selfish, we assume that the BS provides incentive for the UTs to cache and share contents through D2D communications. The incentive for each UT is related to the broadcasting data rate, the number of serviced UTs and the popularity of the shared contents. The larger broadcasting data rate and the more popular contents of UT n provides to surrounding UTs, the more incentives it obtains. Each time the UT n satisfies a request of UT n' in its communication range, the obtained incentive is $\lambda h^{(t)}_{a_n}s$, where λ is the unit incentive value, $h^{(t)}_{a_n}$ denotes the popularity of the content cached by UT n according to its action a_n in time slot t, and s represents the size of content chunk. Meanwhile, the cost of content sharing is modeled as a constant times the broadcast power. In the case when the joint action $\mathbf{a}$ of all UTs and the network state are known at time slot t, the immediate network reward, including the incentive and the cost, is calculated as follows,

$$r(S^{(t)},\mathbf{a})=\sum_{n\in\mathcal{N}}\left(b_n^{\left(C^{(t)}\right)}\lambda\sum_{n'\in\mathcal{N}_d}h^{(t)}_{a_n}s-\beta P_n\right), \tag{5}$$

where $b_n^{\left(C^{(t)}\right)}$ denotes the broadcasting data rate of UT n under D2D communication relationship $C^{(t)}$ in time slot t, as defined in (1). $\mathcal{N}_d=\left\{n'|a_{n'}\neq a_n;b_n^{\left(C^{(t)}\right)}>b_i^{\left(C^{(t)}\right)},\forall i\in\mathcal{N}/n;c^{(t)}_{n,n'}=1\right\}$, which denotes the set of UTs who do not cache the interested content and request this content from UT n within its D2D communication range in time slot t. P_n denotes the transmission power of UT n. β denotes the unit cost. (5) is the sum of the content caching and sharing rewards of all UTs in the cell.

We define the combine vector of the cache placement policy under state $S^{(t)}$ for all UTs except UT n as $\mathbf{\Pi}\left(S^{(t)}\right)$, which is expressed as follows,

$$\begin{aligned}\Pi\left(S^{(t)}\right)&=\pi_1\left(S^{(t)}\right)\otimes\pi_1\left(S^{(t)}\right)\otimes\cdots\otimes\pi_{n-1}\left(S^{(t)}\right)\\&\otimes\pi_{n+1}\left(S^{(t)}\right)\otimes\cdots\otimes\pi_N\left(S^{(t)}\right),\end{aligned} \tag{6}$$

where the symbol $\otimes$ represents the Kronecker product operation. $\pi_n\left(S^{(t)}\right)$ denotes the cache placement policy of UT n defined in (3). Then the immediate reward of UT n is

$$r\left(S^{(t)},a_n\right)=\sum_{\Pi\left(S^{(t)}\right)\in\mathbf{\Pi}\left(S^{(t)}\right)}r_e\left(S^{(t)},\Pi\left(S^{(t)}\right),a_n\right), \tag{7}$$

where $a_n\in\mathcal{A}_n$, $\mathcal{A}_n=\{1,2,...,K\}$ denotes the set of actions that UT n might choose, i.e., a set of contents that can be cached. The expected reward for UT n with state $S^{(t)}$ and

specific action policy is given by

$$r_e\left(S^{(t)},\Pi\left(S^{(t)}\right),a_n\right)=\Pi\left(S^{(t)}\right)\pi_n\left(a_n|S^{(t)}\right)r\left(S^{(t)},\mathbf{a}\right). \tag{8}$$

(7) indicates that the reward that UT n obtained when selecting action a_n in a certain state $S^{(t)}$ depends on the joint action policy selected by other UTs.

In order to comply with the requirements of **Assumption 1**, we assume that all UTs converge to the pure policy using the GLIE strategy [37] while learning. So that all UTs have the same reward $r\left(S^{(t)},\mathbf{a}\right)$. Considering long-term reward, we define the expectation of infinite-horizon discounted sum reward as the common value function of all UTs, given the immediate reward of each UT calculated according to (7). The value function is determined by the starting state S and the policy π_n selected by UT n,

$$V^*(S)=\mathbb{E}_{\pi_n}\left(\sum_{k=0}^{\infty}\gamma^{(k)}r\left(S^{(t+k)},a_n\right)|S^{(t)}=S\right), \tag{9}$$

where $\gamma\in[0,1)$ denotes the discount factor. The value function means that the UTs not only consider immediate rewards when making action decisions, but also considers the impact on the future. A larger γ means that the future rewards have more influence on the current decision, and vice versa. For simplicity, we use S' and S to denote $S^{(t)}=S$ and $S^{(t+1)}=S'$ in subsequent formulas involving states update, $\mathbb{P}\left(S'|S,a_n\right)=\mathbb{P}\left(S^{(t+1)}|S^{(t)},a_n\right)$ in (4). According to Bellman's equation [8],

$$\begin{aligned}V^*(S)=&\sum_{a_n}\pi_n^*(a_n|S)+\\&\left[r(S,a_n)+\gamma\sum_{S'\in\mathcal{S}}\mathbb{P}(S'|S,a_n)V^*(S')\right],\end{aligned} \tag{10}$$

which indicates that in the cooperative game, in addition to UT n being studied, we consider the other $N-1$ UTs' action policies as part of the environment. At the same time, if all UTs adopt a pure policy, i.e., only one content is selected to cache per UT, then $\mathbb{P}\left(S'|S,a_n\right)=\mathbb{P}\left(S'|S,\mathbf{a}\right)$.

D. Equilibrium of the Stochastic Game

In the stochastic game model described above, each agent (UT) independently determines its own best cache placement policy to maximize its own value function as the network state evolves, given other UTs' policies. For this decision-making process that pursues best responses to other UTs' policies, we can consider the Nash equilibrium as the solution to the game. Let π^* be the best action, i.e., the best cache placement policy, of all the UTs. π^*_{-n} is the best action of all the UTs expect UT n. The Nash equilibrium satisfies the following properties,

$$V^*\left(S,\pi_n^*,\pi_{-n}^*\right)\geq V^*\left(S,\pi_n,\pi_{-n}^*\right),\qquad\forall n\in\mathcal{N} \tag{11}$$

where π_n is the cache placement policy of UT n, π_{-n} denotes any set of stationary policies for all UT except n. At the Nash equilibrium point, other policies besides the best cache placement policy results in a reduction in the individual value function, so the UTs have no incentive to change their policies. It has been proved that there is a static Markov policy in the stochastic game, that is, the Nash equilibrium of the stochastic game [38]. The Nash equilibrium of the stochastic game is the best-reward action π^*, which is the best cache placement policy set of all UTs in the cellular networks.

Since the CCU usually does not know the transition probability between network states when making the cache placement decisions in the practical scenarios, we explore the reinforcement learning method to discover the cache placement policy to maximize the long-term reward of all UTs in (9) in the stochastic game for such Nash equilibrium.

IV. Multi-Agent Cooperative Alternating Q-Learning

In this section, we propose a multi-agent cooperative alternating Q-learning (CAQL) framework for solving the stochastic game. Then we describe the CAQL based cache placement algorithm. We discuss the convergence, complexity and reliability of the proposed CAQL.

A. Framework of the Multi-Agent CAQL

In the case of the value function of all UTs known as (10), we can define the action-value function $Q^*(S,a_n)$ as

$$Q^*(S,a_n)=r(S,a_n)+\gamma\sum_{S'\in\mathcal{S}}\mathbb{P}(S'|S,a_n)V^*(S'), \tag{12}$$

where

$$V^*(S)=\sum_{a_n}\pi_n^*(a_n|S)Q^*(S,a_n). \tag{13}$$

Therefore, we can further get the pure policy $\pi_n(a_n|S)=\arg\max_{a_n}Q^*(S,a_n)$ for UT n under state S. Next, we use the joint optimization learning of the action-value function to the best cache placement policy $\pi_n(a_n|\mathcal{S})$ for each UT n in all states.

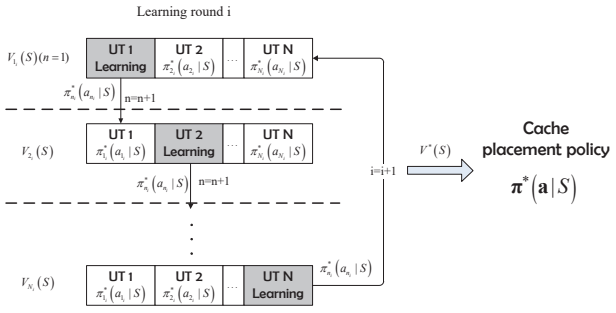

Fig. 3: Framework of the proposed multi-agent CAQL.

Consider the action-value function of each UT, we use a cooperative alternating learning mode based on Q-learning for different initial states of the networks. The framework of the proposed multi-agent CAQL is shown in the Fig 3. We define the learning of UT n in learning round i as learning phase n_i. In the proposed framework, there is only one UT to learn and update the policy in each learning phase, and other UTs maintain their own policies. UT n can know the stability policy choices of all other UTs π^*_{-n} through cooperative strategy of

FCSG in learning phase n_i, and obtain the best response policy $\pi_n(a_n|S) = \arg\max_{a_n} Q^*(S, a_n)$. In fact, the proposed multi-agent CAQL is carried out in the CCU, which simulates the alternating learning process of multiple agents.

The learning process of the learning round i is shown in Fig. 3. In the learning phase 1_i, UT 1 first learns, and at the same time, the pure cache placement policies of other UTs are determined by random initialization. UT 1 continuously uses Q-learning to learn the update of $Q(S, a_{1_i})$ according to the joint action of all UTs until $Q(S, a_{1_i})$ is sufficiently converged. We consider $V_{1_i}(S) = \max_{a_{1_i}} Q^*(S, a_{1_i})$ as the value of the common value function derived at this learning phase and pure policy $\pi_{1_i}(a_{1_i}|S) = \arg\max_{a_{1_i}} Q^*(S, a_{n_i})$ as the cache placement policy UT 1 selected after learning. In the learning phase 2_i, UT 2 repeats the same learning process of UT 1 in learning phase 1_i based on the updated policy $\pi_{1_i}(a_{1_i}|S)$ and the initialized policies that have not been updated by other UTs. After this learning phase, UT 2 obtains the common value function $V_{2_i}(S) = \max_{a_{2_i}} Q^*(S, a_{2_i})$ and the cache placement policy $\pi_{2_i}(a_{2_i}|S)$. By analogy, all UTs learn in turn and update their cache placement policies until UT N is completed in the learning round i. Then the learning process of the i^{th} learning round is repeated in the $(i+1)^{\text{th}}$ round until the common value function $V_{n_i}(S)$ converges to $V^*(S)$. From the framework described above, the proposed CAQL has following properties.

Property 1. *The proposed CAQL is a cooperative environment-aware learning. In CAQL, all UTs have the same immediate reward goal. During the learning process, the current cache placement policies selected by all UTs are public. Each UT can learn the current stable cache placement policies selected by other UTs through the cooperative strategy during the learning process. The selected cache placement policies may be initialized or updated after the previous phases of learning.*

Property 2. *All the agents of CAQL jointly update Q-value and policy. In each phase of the learning process, UT not only learns and updates its Q-value, but also updates its own cache placement policy according to the learning result after the end of each learning phase, so that the common value function is of the same term for all UTs in different learning phases.*

Property 3. *The proposed CAQL has small cache space requirements. Compared with the traditional multi-agent Q-learning that needs to maintain a joint action and Q-value matrix (be known as Q-value table), CAQL regards the actions in each learning phase of other UTs as stable, only considers the impact of the learning agent's decisions on its Q-value. CAQL reduces the space required for the Q-table, especially when the action set is large.*

B. CAQL Based cache placement Algorithm

Firstly, we discuss the Q-learning algorithm for a single UT in each learning phase in this subsection. It can be known from **Assumption 1** that the immediate reward goals of all UTs are the same. When their cache placement policies are stable pure policies, each UT's immediate reward is the network's immediate network reword. **Property 1** and **Property 2** can guarantee the stability of the state transition probability and the calculation of the Q-value in Q-learning.

The learning process of Q-learning is model-free. Even if the agent currently learning has its own action policy, during the learning process, the UT does not act according to the policy, but acts according to another policy independent of the model, thereby balancing the exploration and exploitation in the learning process. The action selection policy can be a previously learned policy, or a policy that has been optimized and matured, such as the widely adopted $\varepsilon-\text{greedy}$ [39]. The UT following the $\varepsilon-\text{greedy}$ policy randomly selects the action to explore with the probability ε, otherwise the greedy action selection policy will be executed according to the existing Q-value with a probability of $(1-\varepsilon)$. In order to meet the conditions of **Assumption 1**, we let UTs adopt the greedy in the limit with infinite exploration (GLIE) strategy in the learning process, which can ensure that each UT's policy converges to a stable state. The combination of GLIE and $\varepsilon-\text{greedy}$ requires that all state-action pairs that have been experienced will be explored indefinitely. At the same time, as the number of explorations increases, the ε-value in $\varepsilon-\text{greedy}$ tends to zero (for example, $\varepsilon=1/t$), which means that the frequency of exploration decreases as the number of explorations increases. We record the $\varepsilon-\text{greedy}$ policy that conforms to GLIE characteristics as $\varepsilon_t-\text{greedy}$. The action selection strategy is

$$a_{n_i} = \begin{cases} \arg\max_{a_{n_i}} Q_{n_i}(S, a_{n_i}) & \text{w.p.} \quad 1-\varepsilon_t; \\ \text{random}\, a_{n_i} \in \mathcal{A} & \text{w.p.} \quad \varepsilon_t. \end{cases} \tag{14}$$

In order to measure the convergence of each learning phase, we define a UT convergence index and a UT convergence threshold for each learning phase. The UT convergence index in the learning phase n_i of UT n is defined as $I_{n_i}^{(p)}$, which is

$$I_{n_i}^{(p)} = \left| R_{n_i}^{(t)} - R_{n_i}^{(t-1)} \right|, \tag{15}$$

where $R_{n_i}^{(t)} = \sum_{S\in\mathcal{S}} \sum_{a_n} \pi_n^*(a_n|S) r(S, a_n)$ denotes the sum of immediate rewards under all network states after UT n updates the policy in the current learning phase in time slot t. When the UT's cache placement policy converges to stability, $R_{n_i}^{(t)}$ tends to be stable as well. Therefore, $I_{n_i}^{(p)}$ is used to indicate the convergence of UT n in the n_i learning phase. We define the UT convergence threshold as κ_p. If $I_{n_i}^{(p)} < \kappa_p$, it means that UT n's cache placement policy is stable between time slot t and $t-1$. Considering the characteristics of the $\varepsilon-\text{greedy}$ exploration, the convergence of UT n in the current learning phase is achieved when the number of consecutive $R_{n_i}^{(t)}$ stable time slots exceeds a certain predefined positive integer value I_t. Then the CAQL terminates learning phase n_i and starts the next learning phase $(n+1)_i$. The Q-value update rule of Q-learning for each UT is given as,

$$\begin{aligned} &Q_{t+1}(S, a_n) = (1-\alpha_t) Q_t(S, a_n) + \\ &\alpha_t \left(r(S, a_n) + \gamma \max_{a_n} Q(S', a_n) \right). \end{aligned} \tag{16}$$

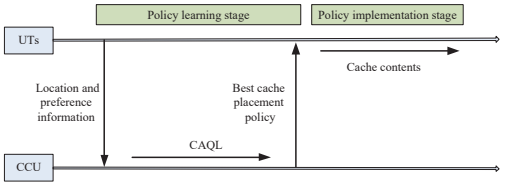

Fig. 4: CAQL operation in CCU.

For the convergence of the entire CAQL, we define the convergence index $I_{n_i}^{(r)}$ to indicate the convergence performance of CAQL in each learning round i,

$$I_{n_i}^{(r)} = \left| R_{n_i} - R_{n_{i-1}} \right|. \tag{17}$$

Then we define the convergence threshold of CAQL as κ_r. When κ_r is small enough, if $I_{n_i}^{(r)} < \kappa_r$, it means that the learning of CAQL reaches convergence at the current round.

The proposed CAQL is operated in the CCU, as shown in Fig. 4, including the policy learning stage and the policy implementation stage. For policy learning, the UTs report their location and content preference information to CCU in the information exchange phase of each time slot. Based on these information, the CCU learns the best cache placement policy for all the UTs in the entire network. For policy implementation, the BS transmits the multiple contents to multiple UTs for caching in each network stage according to the learned policy of the CCU. The procedure of CAQL based cache placement algorithm operated in the CCU is described by **Algorithm 1**.

Algorithm 1 CAQL based cache placement algorithm

- ***Step 1: Initialization***
Randomly initialize the starting network state S and the cache placement policy of all UTs π^*. Initialize the Q-value table (joint action and Q-value matrix) values $Q_n(S, a_n) = 0\,(\forall n \in \mathcal{N})$ of all UTs. Set $n = 1$, $i = 1$ and $n_i = 1_1$. Set the convergence threshold κ_p and κ_r. The maximum number of learning round is L, which means $i \leq L$.
- ***Step 2: Learning***
repeat
  while $n \leq N$ ***do***
    Set the stable time slots $s_{num} = 0$
    repeat
      In the current learning phase n_i with network state S, UT n selects action according to $\varepsilon_t -$ greedy policy.
      Observe the next network state S' after the transfer and obtain the immediate reward $r(S, a_{n_i})$ by (7).
      Update $Q_{n_i}(S, a_{n_i})$ according to (16).
      if $I_{n_i}^{(p)} < \kappa_p$, $s_{num} = s_{num} + 1$
      else $s_{num} = 0$
      Transfer to next state S'
    until $I_{n_i}^{(p)} < \kappa_p$ and $s_{num} > I_t$
    $n = n + 1$
  end while
  $i = i + 1$
until $I_{n_i}^{(r)} < \kappa_r$, or the learning round i reaches the maximum constraint L.
output: Learned cache placement policy $\pi^*(\mathbf{a}|S)$.

In summary, we solve the FCSG by multi-agent cooperative reinforcement learning, which derives the stable cache placement policy of UTs considering long-term rewards. Through the CAQL process, we get a stable Q-value table for all UTs which provides the best cache placement policy for them. In other words, all UTs in the cellular networks learn to know what they should cache under the network state $S \in \mathcal{S}$. Obviously, the proposed algorithm is a centralized algorithm since it is not required control information exchange between UTs. In doing so, we avoid the huge overhead consumption in such D2D-enabled caching cellular networks.

C. Property Evaluations

1) Convergence: We evaluate the convergence of CAQL. To obtain UTs' stable cache placement policies through the CAQL, each UT needs to converge in n_i learning phase. Then as the learning round i increases, all UTs' policies reach convergence and the common value function converges to $V^*(S)$.

We first discuss the convergence of Q-learning for a single UT in each learning phase of the proposed CAQL.

Theorem 1. *For fully cooperative multi-agent systems, if the following conditions are satisfied, the Q-value table of UT n eventually converges to a stable value $Q^*(S, a_n)$ according to the Q-value update rule of Q-learning in* (16)*.*

1) All agents share the same immediate reward $r\left(S^{(t)}, a_n\right)$;

2) In addition to the agent that is learning, the policies of the remaining agents are stable;

3) The learning speed α_t of Q-learning satisfies the condition $\sum_{t=0}^{\infty} \alpha_t = \infty$ $\sum_{t=0}^{\infty} \alpha_t^2 < \infty$.

For a UT that can know current stable policies of all other UTs, its learning process is identical to the single-agent Q-learning. Therefore, **Theorem 1** is clearly established with sufficient number of updates and a decreasing learning rate [40]. We can consider that the Q-learning process of each UT in CAQL is convergent. In our CAQL, only when a UT's learning phase converges will it enter the next UT's learning phase. However, the convergence of **Theorem 1** is achieved with $t \to \infty$. Obviously, in reality, we cannot achieve the condition of $t \to \infty$, so we need to set the convergence index and thresholds above in (15) and (17). It is known by **Theorem 1** that convergence can be achieved when each UT performs Q-learning alone in every learning phase of the CAQL.

During the cooperative alternating learning process of CAQL, each UT makes a best response cache placement policy according to the current policies of other UTs, which will further influence the policies of other UTs in the subsequent learning phases. All UTs alternatively promote updates to their policies by cooperating to perceive each other's policies. In this case, as the learning phase progresses, whether the policies of all participating UTs in CAQL algorithm can converge is the next question we need to explore.

Theorem 2. *When UTs sequentially update their own cache placement policy in the learning process according to the CAQL framework, $\forall S \in \mathcal{S}$, $\{V_{1_1}(S), V_{2_1}(S), ...V_{N_1}(S), V_{1_2}(S), ...\}$ is a non-decreasing Cauchy sequence.*

Proof. We assume that the current network state is S, the initial cache placement policy for all UTs is π^*. From (7), the immediate reward of UT n in learning phase n_i is

$$r\left(S, a_{n_i}\right) = \sum_{a_{-n_i}} \pi^*_{-n_i}\left(a_{-n_i}|S\right) r\left(S, \mathbf{a}\right), \tag{18}$$

where $-n_i$ denotes the remaining UTs in the learning phase n_i except UT n. It can be seen from **Theorem 1** and (13) that after alternating learning in the n_i learning phase, UT n gets the following common value function,

$$\begin{aligned} &V_{n_i}\left(S\right) = \max_{a_{n_i}} Q^*\left(S, a_{n_i}\right) \\ &= \sum_{a-n_i} \pi^*_{-n_i}\left(a_{-n_i}|S\right) r\left(S, a_{-n_i}, a^*_{n_i}\right) \\ &+\gamma \sum_{S'} \sum_{a_{-n_i}} \pi^*_{-n_i}\left(a_{-n_i}|S\right) \cdot \mathbb{P}\left(S'|S, a_{-n_i}, a^*_{n_i}\right) V^*\left(S'\right), \end{aligned} \tag{19}$$

where $\mathbb{P}\left(S'|S, a_{-n_i}, a^*_{n_i}\right)$ denotes the probability of network state transferring from S to S' under current cache placement policy. The best pure policy $\pi^*_{n_i}\left(a^*_{n_i}|S\right) = 1$ with $a^*_{n_i} = \arg\max_{a_{n_i}} Q^*\left(S, a_{n_i}\right)$.

The UT $n+1$ continues learning in the next learning phase $(n+1)_i$, which obtains the common value function as,

$$\begin{aligned} &V_{(n+1)_i}\left(S\right) = \max_{a_{(n+1)_i}} Q^*\left(S, a_{(n+1)_j}\right) \\ &= \sum_{a_{-(n+1)_i}} \pi^*_{-(n+1)_i}\left(a_{-(n+1)_i}|S\right) r\left(S, a_{-(n+1)_i}, a^*_{n+1_i}\right) \\ &+\gamma \sum_{S'} \sum_{a_{-(n+1)_i}} V_s{}^*\left(a_{-(n+1)_i}, S', S\right), \end{aligned} \tag{20}$$

where the optimal state value for the state S and the specific next state S' is given by

$$\begin{aligned} &V_s{}^*\left(a_{-(n+1)_i}, S', S\right) = \pi^*_{-(n+1)_i}\left(a_{-(n+1)_i}|S\right) \\ &\cdot \left(S'|S, a_{-(n+1)_i}, a^*_{(n+1)_i}\right) V^*\left(S'\right). \end{aligned} \tag{21}$$

In learning phase $(n+1)_i$, UT n has learned the stable pure best cache placement policy $\pi^*_{n_i}\left(a^*_{n_i}|S\right)$ in the previous learning phase and select action according to it at current phase. Therefore, we can replace $-(n+1)_i$ with $-(n, n+1)_i$. For the sake of simplicity, we rewrite the $V_{(n+1)_i}(S)$ expression associated with $a^{-(n+1)_i}$ as,

$$V_{(n+1)_i}\left(S\right) = \max_{a_{(n+1)_i}} f\left(a_{-(n,n+1)_i}\right). \tag{22}$$

Similarly, (19) can also be written as,

$$V_{n_i}\left(S\right) = \sum_{a_{(n+1)_i}} \pi^{(n_i)}_{(n+1)_i}\left(a_{(n+1)_i}|S\right) f\left(a_{-(n,n+1)_i}\right), \tag{23}$$

where $\pi^{(n_i)}_{(n+1)_i}\left(a_{(n+1)_i}|S\right)$ denotes the cache placement policy adopted by UT $n+1$ in the previous n_i learning phase. Obviously, the UT $n+1$ in current phase continues to learn based on the best cache placement policy derived from the UT n in the previous phase. From (22) and (23), we have

$$V_{(n+1)_i}\left(S\right) \geq V_{n_i}\left(S\right), \tag{24}$$

which indicates that the sequence $\{V_{1_1}(S), V_{2_1}(S), ...V_{N_1}(S), V_{1_2}(S), ...\}$ is non-decreasing. At the same time, because individual immediate reward is bounded ($r(S^{(t)}, a_n) \leq (N-1)rs - \alpha P_n$), $\gamma \in [0, 1)$, $V_{n_i}(S)$ is bounded. Therefore, $\{V_{1_1}(S), V_{2_1}(S), ...V_{N_1}(S), V_{1_2}(S), ...\}$ is a non-decreasing Cauchy sequence. Then the common value gradually increases and converges as the learning process progresses as follows,

$$V^*\left(S\right) = \mathbb{E}_{\pi_n}\left(\sum_{k=0}^{\infty} \gamma^{(k)} r\left(S^{(t+k)}, a_n\right) |\pi^*\left(\mathbf{a}|S\right)\right), \tag{25}$$

where $\pi^*\left(\mathbf{a}|S\right)$ denotes the stable cache placement policy for all UTs after learning.

Remark 1. *From **Theorem 1** and **Theorem 2**, CQAL eventually converges to a stable cooperative cache placement policy that aims to maximize the common value function in the joint state-action space.*

After the algorithm we proposed converges, the current learning UT no longer changes its own cache placement policy, then the subsequent UTs have no motivations to change their own policies, which obtains the Nash equilibrium. The stable cache placement policy achieves a sub-optimal solution of the common value function (9). Unfortunately, due to the limitations of learning exploration strategies, we can not guarantee that (25) can achieve global optimum.

2) Complexity: In the learning phase, since the policies currently adopted by other UTs in the cooperative environment are known, the learning UT only needs to make decisions and update its own cache placement policy. Each UT only needs to maintain a Q-value table with a space complexity of $|\mathcal{S}| \cdot |\mathcal{A}|$. The entire CAQL algorithm needs to maintain a Q-value table with a space complexity of $N \cdot |\mathcal{S}| \cdot |\mathcal{A}|$. Compared to the traditional multi-agent Q-learning method that needs to maintain the joint action Q-value table (such as Nash-Q [41] with space complexity $|S| \cdot |A|^N$), CAQL requires less space cost. At the same time, since there is only one UT for learning in each learning phase, the required computing resources are relatively small.

3) Reliability: The proposed algorithm is robust in the practical scenarios of dynamic networks. After learning for a period of time, the CCU can make a cache placement decision according to the historical information even when the current network information is unavailable. When the information of an individual UT is unavailable, the CCU that has not received its reported information will delete the UT from the learning agents set and adjust the cache placement policy which ensures the reliability of the CAQL.

V. Numerical Results

In this section, the performance of the proposed CAQL based cache placement algorithm is tested. We verify the convergence of the proposed CAQL in the cache placement policy learning stage. Then we demonstrate the caching performance of the proposed CAQL in the policy implementation stage.

A. Simulation Settings

In the simulation, a macro BS is deployed at the center of the cell and N UTs are distributed in the cell and move according to the multi-home-point movement model in [33],

as we described in Section II. The communication relationship between the UTs can be described by (1). The multi-home-point movement model is used to model the UTs' movement in the cell with a coverage radius $R = 250$ m. We assume that there are $H = 3$ home-points distributed within the coverage. The coverage radius of each home-point is $R_h = 50$ m. The UTs periodically move between different home-points, and the time they stay in each home-point is fixed and non-empty. We modeled the D2D communication relationship between the N UTs as a finite Markov state transition model with three states $\{C_1, C_2, C_3\}$, and assume that the transition probabilities are

$$\mathbb{P}\left\{C^{(t+1)}|C^{(t)}\right\} = \begin{bmatrix} \mathbb{P}\{C_1|C_1\} & \mathbb{P}\{C_1|C_2\} & \mathbb{P}\{C_1|C_3\} \\ \mathbb{P}\{C_2|C_1\} & \mathbb{P}\{C_2|C_2\} & \mathbb{P}\{C_2|C_3\} \\ \mathbb{P}\{C_3|C_1\} & \mathbb{P}\{C_3|C_2\} & \mathbb{P}\{C_3|C_3\} \end{bmatrix} = \begin{bmatrix} 0.2 & 0.8 & 0 \\ 0 & 0.2 & 0.8 \\ 0.8 & 0 & 0.2 \end{bmatrix}. \tag{26}$$

We assume that there are $K = 8$ contents to be cached in the cell. The popularity of K contents follows a *Zipf*-like distribution [42] and the size of each content chunk s is set to 1M bytes. The K content are sorted accordingly in a descending order. The popularity of the k-th content with distribution parameter α_i is

$$h_k(i) = \frac{1}{k^{\alpha_i} \sum_{m=1}^{K} 1/m^{\alpha_i}}, \text{ for } i = 1, 2, \tag{27}$$

where $\alpha_i \geq 0$ is the skewness of popularity. We use $h_k^{(t)}$ to denote the popularity of content k, $\sum_{k=1}^{K} h_k^{(t)} = 1$. The content popularity is modeled by a two-state Markov chain with states $\mathbf{h}_1$ and $\mathbf{h}_2$, that are drawn from Zipf distributions having parameter $\alpha_1 = 0.7$ and $\alpha_2 = 2.5$, respectively. The transition probability matrix is defined as

$$\mathbf{P}_k^{(H_1)} = \begin{bmatrix} \mathbb{P}\{h_k(1)|h_k(1)\} & \mathbb{P}\{h_k(1)|h_k(2)\} \\ \mathbb{P}\{h_k(2)|h_k(1)\} & \mathbb{P}\{h_k(2)|h_k(2)\} \end{bmatrix} = \begin{bmatrix} 0.6 & 0.4 \\ 0.2 & 0.8 \end{bmatrix}. \tag{28}$$

We initialize the network state $S = [C_1, \mathbf{h}_1]$. The network state will change between different time slot which we defined in Fig. 2. The CCU learns the cache placement policy according to **Algorithm 1** in the case where the content popularity and the UTs' D2D communication relationship state continuously transferred. In the simulation, the parameter setting used for D2D communication is from the Technical Report of 3GPP [43], the system bandwidth of D2D communication is 10 MHz uplink and 10 MHz downlink for FDD, and the indoor to indoor channel model is as defined in [43], including the pathloss, shadowing, and the fast fading. The detailed simulation parameters are given in Table II.

B. Simulation Results of Convergence

We first demonstrate the convergence of the proposed CAQL algorithm in the learning stage of cache placement policy. According to (17), we compare the long-term reward of all

TABLE II: Simulation Parameters

Parameter	Value
Carrier frequency	2 GHz
Radio bandwidth	20 MHz
Backhaul data rate	1.5 Mbps
D2D transmit power $P_{n'}^{tx}$	23 dBm
BS transmit power P_{BS}^{tx}	43 dBm
Pathloss from BS to UT	37.6log10(d[km])+ 128.1 dB
Pathloss of D2D channel	40log10(d[km])+ 148 dB
Noise power spectral density	-174 dBm/Hz
Received signal power threshold η	-50 dBm
Convergence threshold κ_p	10^{-3}
Convergence threshold κ_r	0.1
Stable time slots threshold I_t	10^3

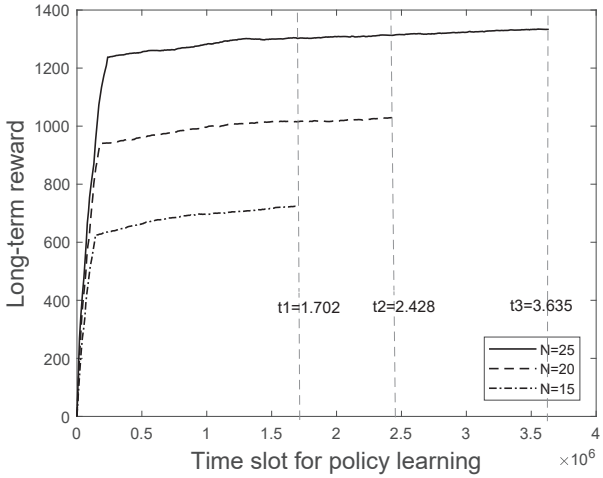

Fig. 5: Learning process of UTs in CAQL.

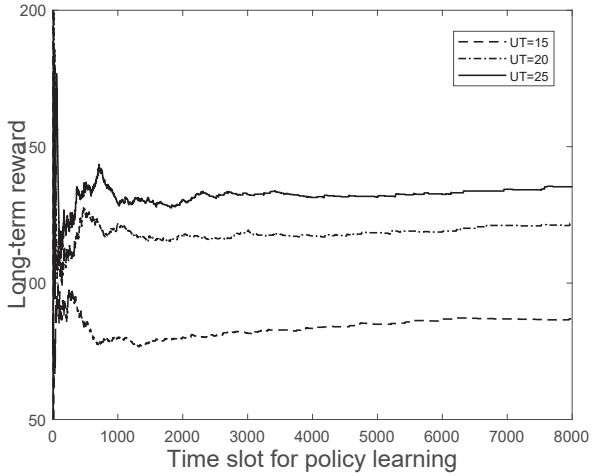

Fig. 6: Learning process of single UT in each learning phase.

network states after the current learning UT updates its policy between current and the last learning round. Fig. 5 shows the convergence of the CAQL, where the number of UTs N is equal to 15, 20, and 25, respectively. The long-term reward of all UTs in the cell is stable after learning is completed. Then we discuss the convergence of single UT in learning phase in Fig. 6. Fig. 6 shows the UT's cache placement policy gradually stabilizes during the learning process in each learning phase. Fig. 5 and Fig. 6 demonstrate the **Remark 1** we proposed in Section IV.C. The number of network state $|\mathcal{S}|$ is determined by the number of contents K, the number of content states $|\mathcal{H}|$, and the number of D2D communication relationship states $|\mathcal{C}|$ between UTs, $|\mathcal{S}| = |\mathcal{H}|^K \cdot |\mathcal{C}|$. Fig. 5 shows that the required number of time slots for convergence nearly has linear relationship with the number of UTs, which verified that the Q-value table of the proposed CAQL has a space complexity of $N \cdot |\mathcal{S}| \cdot |\mathcal{A}|$, as discussed in Section IV.C.

C. Simulation Results of Performance

Next, we compare the caching performance of the proposed CAQL in policy implement stage with the social-aware caching game (SACG) [44], random caching (RC) and popular caching (PC). In the SACG algorithm, the cache placement is based on the statistical user encounter probabilities and the current content popularity. In the RC algorithm, the contents are allocated to UTs randomly in each time slot. In the PC algorithm, UTs cache the most popular contents in each time slot. We statistically give the long-term performance of these cache placement algorithms over time slot and environmental changes.

We compare the long-term reward of all UTs in the cell obtained by different cache placement algorithms, as shown in Fig. 7. As the time series progresses, the D2D communication relationship between UTs and content popularity change. When $t > 100$, the average immediate reward tends to be stable. The proposed CAQL has the highest immediate reward compared with the SACG, RC, and PC. For example, When UT=25, the average immediate reward of CAQL is 1033, which exceeds SACG, RC and PC 5.8%, 14.5% and 35% respectively.

Then we compare the caching performance with the average content access delay and the BS traffic offloading ratio as the criteria in Fig. 8 and Fig. 9. In the simulation, the content access of UT n is given by

$$D_n^{(t)} = \left[\frac{s}{r_{n,n'}^{\max}(t)}\theta + \left(\frac{s}{r_{n,BS}(t)} + \frac{s}{r_{back}} \right)(1-\theta) \right], \quad (29)$$

where $r_{n'}^{max}(t)$ denotes the maximum data rate of D2D communication between UT n and n' ($n' \in \mathcal{N}_n^{(k)}$) during the content delivery in time slot t. r_{back} denotes the data rate of the backhaul link of the BS. Therefore, the average content access delay of N UTs for K contents is given by

$$D^{(t)} = \mathbb{E}_t \left\{ \mathbb{E}_n \left\{ \sum_{k=1}^{K} h_k^{(t)} D_n^{(t)} \right\} \right\}, \quad (30)$$

where $\theta = 1$ if $\mathcal{N}_n^{(k)} \neq \emptyset$ ($\mathcal{N}_n^{(k)} = \{n' | a_{n'}^{(t)} = k, b_{n'}^{(C^{(t)})}(t) > 0\}$), which means that content k is shared by D2D communication in the cell, else $\theta = 0$, which is the case of BS transmission. $\mathbb{E}_n$ indicates the average access time for all UTs to obtain all their required contents. $\mathbb{E}_t$ indicates the average content access time until current t. The BS traffic offloading ratio is calculated as

$$O = \frac{\sum_{k=1}^{K}\sum_{n=1}^{N} h_k^{(t)} \mathbf{1}\left(a_n^{(t)} = k\right)}{\sum_{k=1}^{K}\sum_{n=1}^{N} h_k^{(t)}}, \quad (31)$$

where $\mathbf{1}\left(a_n^{(t)} = k\right) = 1$ when $a_n^{(t)} = k$, else $\mathbf{1}\left(a_n^{(t)} = k\right) = 0$. The larger value of O means more contents are delivered by D2D communications.

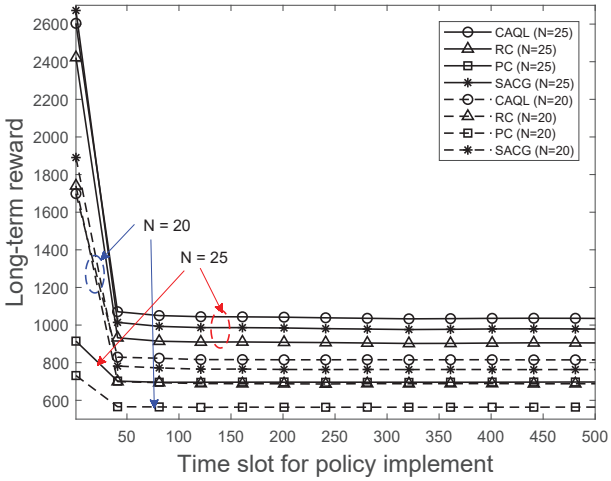

Fig. 7: Long-term reward comparison with varying UT numbers.

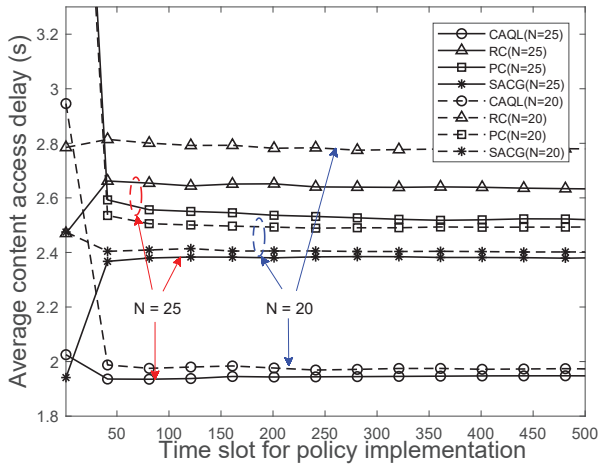

Fig. 8: Averaged content access delay with varying UT numbers.

Fig. 8 compares the time-averaged content access delay for different cache placement algorithms when $N = 20$ and $N = 25$, respectively. As shown in Fig. 8, no matter which algorithm is used for proactive caching at off-peak hours,

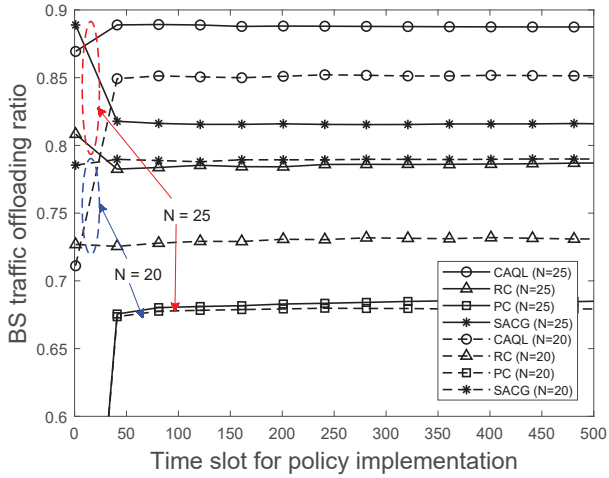

Fig. 9: BS traffic offloading ratio with varying UT numbers.

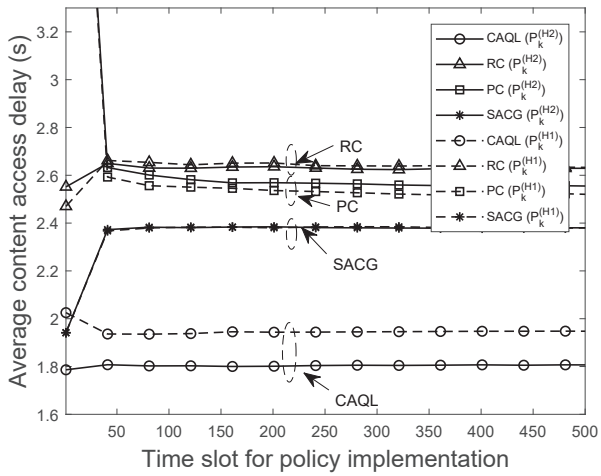

Fig. 10: Average content access delay with different content popularity transition probabilities.

the average content access delay for content delivery during peak hours is reduced since the average content access delay by the BS transmission without caching is about 7 s in our simulation. When $t = 1$, the average content acquisition delay of SACG is the smallest, which is 95.8% and 84% of CAQL when $N = 25$ and $N = 20$, respectively. As the time slot t increases, the average content access delay of CAQL becomes stable gradually. For example, when $t > 200$ and $N = 20$, the average content access delay of CAQL is stable, which is 82%, 71%, and 79% of SACG, RC, and PC, respectively. When the user density is large, it is beneficial for content sharing between UTs. When $N = 25$, the average content access delay is reduced by 2% compared with $N = 20$. Fig. 9 demonstrates the average BS offloading for different caching algorithms. When $N = 25$, the stable average BS offloading ratio of CAQL is the largest, reaching 0.89, which is 10%, 23.6%, and 30% higher than SACG, RC, and PC, respectively. The simulation results in Fig. 8 and Fig. 9 demonstrate that the proposed algorithm can effectively reduce the UT's content access delay and the peak load of the backhaul in the dynamic environments.

Finally, we consider the caching performance of CAQL when content popularity changes more frequently. We increase the randomness of the dynamic networks by a new transition probability matrix $\mathbf{P}_k^{(H_2)}$, which is defined as

$$\mathbf{P}_k^{(H_2)} = \begin{bmatrix} \mathbb{P}\{h_k(1)\,|h_k(1)\} & \mathbb{P}\{h_k(1)\,|h_k(2)\} \\ \mathbb{P}\{h_k(2)\,|h_k(1)\} & \mathbb{P}\{h_k(2)\,|h_k(2)\} \end{bmatrix} = \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{bmatrix}. \quad (32)$$

Fig. 10 and Fig. 11 demonstrate the impact of different content popularity transition probabilities on the average content access delay and BS traffic offloading ratio. When the randomness of environmental changes increases, the average content access delay of CAQL decreases from 1.93s to 1.76s, and the backhaul offloading ratio increases from 86.94% to 90.57%. The simulation results illustrate the performance of our proposed CAQL algorithm in the dynamic environments.

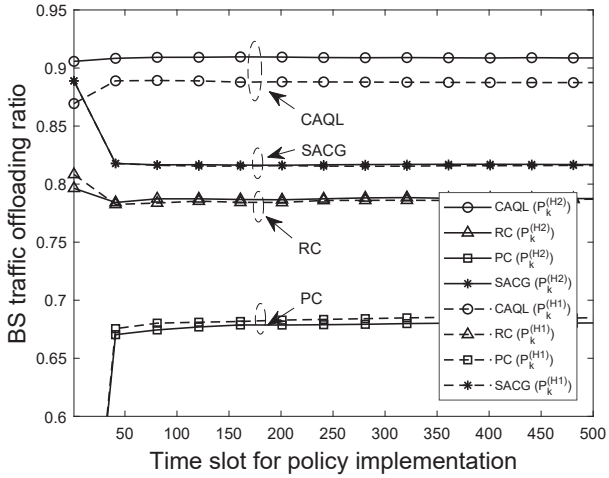

Fig. 11: BS traffic offloading ratio with different content popularity transition probabilities.

VI. Conclusion

In this article, we have investigated the UT edge caching in D2D-enabled caching cellular networks with time-varying UT location and content popularity. The multi-content multi-UT cache placement problem was modeled as a fully cooperative stochastic game of UTs. Then a multi-agent CAQL framework was proposed based on the best response of each UT to solve the stochastic game problem. After CAQL, UTs obtain the best cache placement policy in dynamic networks for long term reward maximization. Simulation results have verified the feasibility and effectiveness of the proposed CAQL based cache placement algorithm.

References

[1] X. Fang, T. Zhang, Y. Liu, and Z. Zeng, "Multi-agent cooperative alternating Q-learning caching in D2D-enabled cellular networks," in *2019 IEEE Global Communications Conference (GLOBECOM)*, Dec. 2019, pp. 1–6.

[2] "Cisco visual networking index: Global mobile data traffic forecast update, 2016 to 2021 white paper," https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html, Mar. 2017.

[3] X. Wang, M. Chen, T. Taleb, A. Ksentini, and V. C. M. Leung, "Cache in the air: exploiting content caching and delivery techniques for 5G systems," *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 131–139, Feb. 2014.

[4] J. G. Andrews, "Seven ways that hetnets are a cellular paradigm shift," *IEEE Commun. Mag.*, vol. 51, no. 3, pp. 136–144, Mar. 2013.

[5] N. Golrezaei, A. F. Molisch, A. G. Dimakis, and G. Caire, "Femtocaching and device-to-device collaboration: A new architecture for wireless video distribution," *IEEE Commun. Mag.*, vol. 51, no. 4, pp. 142–149, Apr. 2013.

[6] E. Bastug, M. Bennis, and M. Debbah, "Living on the edge: The role of proactive caching in 5G wireless networks," *IEEE Commun. Mag.*, vol. 52, no. 8, pp. 82–89, Aug. 2014.

[7] N. Zhao, F. Cheng, F. R. Yu, J. Tang, Y. Chen, G. Gui, and H. Sari, "Caching uav assisted secure transmission in hyper-dense networks based on interference alignment," *IEEE Trans. Commun.*, vol. 66, no. 5, pp. 2281–2294, May 2018.

[8] F. Cheng, G. Gui, N. Zhao, Y. Chen, J. Tang, and H. Sari, "Uav-relaying-assisted secure transmission with caching," *IEEE Trans. Commun.*, vol. 67, no. 5, pp. 3140–3153, May 2019.

[9] D. Liu, B. Chen, C. Yang, and A. F. Molisch, "Caching at the wireless edge: design aspects, challenges, and future directions," *IEEE Commun. Mag.*, vol. 54, no. 9, pp. 22–28, Sep. 2016.

[10] A. Asadi and V. Mancuso, "Network-assisted outband D2D-clustering in 5G cellular networks: Theory and practice," *IEEE Trans. Mob. Comput.*, vol. 16, no. 8, pp. 2246–2259, Aug. 2017.

[11] A. Asadi, Q. Wang, and V. Mancuso, "A survey on device-to-device communication in cellular networks," *IEEE Communications Surveys Tutorials*, vol. 16, no. 4, pp. 1801–1819, Fourthquarter 2014.

[12] M. Ji, G. Caire, and A. F. Molisch, "Fundamental limits of caching in wireless D2D networks," *IEEE Trans. Inf. Theory*, vol. 62, no. 2, pp. 849–869, Feb 2016.

[13] H. J. Kang, K. Y. Park, K. Cho, and C. G. Kang, "Mobile caching policies for device-to-device (D2D) content delivery networking," in *2014 IEEE INFOCOM WKSHPS*, Apr. 2014, pp. 299–304.

[14] X. Chen, X. Gong, L. Yang, and J. Zhang, "Exploiting social tie structure for cooperative wireless networking: A social group utility maximization framework," *IEEE/ACM Trans. Networking*, vol. 24, no. 6, pp. 3593–3606, Dec. 2016.

[15] N. Zhao, X. Liu, Y. Chen, S. Zhang, Z. Li, B. Chen, and M. Alouini, "Caching D2D connections in small-cell networks," *IEEE Trans. Vehic. Tech.*, pp. 1–1, 2018.

[16] R. Wang, X. Peng, J. Zhang, and K. B. Letaief, "Mobility-aware caching for content-centric wireless networks: modeling and methodology," *IEEE Commun. Mag.*, vol. 54, no. 8, pp. 77–83, Aug. 2016.

[17] R. Wang, J. Zhang, S. H. Song, and K. B. Letaief, "Mobility-aware caching in D2D networks," *IEEE Trans. Wireless Commun.*, vol. 16, no. 8, pp. 5001–5015, Aug. 2017.

[18] D. T. Hoang, D. Niyato, D. N. Nguyen, E. Dutkiewicz, P. Wang, and Z. Han, "A dynamic edge caching framework for mobile 5G networks," *IEEE Wireless Commun.*, vol. 25, no. 5, pp. 95–103, Oct. 2018.

[19] B. Chen and C. Yang, "Caching policy for cache-enabled D2D communications by learning user preference," *IEEE Trans. Commun.*, vol. 66, no. 12, pp. 6586–6601, Dec. 2018.

[20] L. S. Shapley, "Stochastic games," *Proc Natl Acad Sci U S A*, vol. 39, no. 10, pp. 1095–1100, 1953.

[21] Y. Liu, S. Bi, Z. Shi, and L. Hanzo, "When machine learning meets big data: A wireless communication perspective," *CoRR*, vol. abs/1901.08329, Mon 2019. [Online]. Available: http://arxiv.org/abs/1901.08329

[22] V. Mnih, K. Kavukcuoglu, D. Silver, A. Rusu, J. Veness, M. G Bellemare, A. Graves, M. Riedmiller, A. K Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–33, Feb. 2015.

[23] T. Z. Oo, N. H. Tran, W. Saad, D. Niyato, Z. Han, and C. S. Hong, "Offloading in HetNet: A coordination of interference mitigation, user association, and resource allocation," *IEEE Trans. Mob. Comput.*, vol. 16, no. 8, pp. 2276–2291, Aug. 2017.

[24] M. Simsek, M. Bennis, and . Gven, "Learning based frequency- and time-domain inter-cell interference coordination in HetNets," *IEEE Trans. Veh. Technol.*, vol. 64, no. 10, pp. 4589–4602, Oct 2015.

[25] D. Niyato, D. I. Kim, P. Wang, and M. Bennis, "Joint admission control and content caching policy for energy harvesting access points," in *2016 IEEE ICC*, May 2016, pp. 1–6.

[26] Y. He, C. Liang, R. Yu, and Z. Han, "Trust-based social networks with computing, caching and communications: A deep reinforcement learning approach," *IEEE Trans. Network Sci. Eng.*, pp. 1–1, 2018.

[27] 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.*, vol. 25, no. 3, pp. 28–35, JUNE 2018.

[28] B. N. Bharath, K. G. Nagananda, and H. V. Poor, "A learning-based approach to caching in heterogenous small cell networks," *IEEE Trans. Commun.*, vol. 64, no. 4, pp. 1674–1686, Apr. 2016.

[29] 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.

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

[31] 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.

[32] W. Jiang, G. Feng, S. Qin, and T. S. P. Yum, "Efficient d2d content caching using multi-agent reinforcement learning," in *2018 IEEE INFOCOM WKSHPS*, April 2018, pp. 511–516.

[33] M. Garetto, P. Giaccone, and E. Leonardi, "Capacity scaling in ad hoc networks with heterogeneous mobile nodes: The super-critical regime," *IEEE/ACM Trans. Networking*, vol. 17, no. 5, pp. 1522–1535, Oct. 2009.

[34] T. Zhang, H. Fan, J. Loo, and D. Liu, "User preference aware caching deployment for device-to-device caching networks," *IEEE Systems Journal*, vol. 13, no. 1, pp. 226–237, Mar. 2019.

[35] C. Boutilier, "Sequential optimality and coordination in multiagent systems," in *Sixteenth International Joint Conference on Artificial Intelligence*, Feb. 1999.

[36] L. Matignon, G. Laurent, and N. Le Fort-Piat, "Independent reinforcement learners in cooperative markov games: A survey regarding coordination problems," *The Knowledge Engineering Review*, vol. 27, pp. 1 – 31, Mar. 2012.

[37] S. Singh, T. Jaakkola, M. Littman, and C. Szepesvri, "Convergence results for single-step on-policy reinforcement-learning algorithms," *Machine Learning*, vol. 38, pp. 287–308, Mar. 2000.

[38] A. Fink, "Equilibrium in a stochastic n-person game," *Journal of Science of Hiroshima University Series A-I Math*, vol. 28(1), p. 8993, 1964.

[39] R. S. Sutton and A. G. Barto, "Reinforcement learning: An introduction," *IEEE Trans. Neural Networks*, vol. 9, no. 5, pp. 1054–1054, Sep. 1998.

[40] J. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, pp. 279–292, Jan. 1992.

[41] J. Hu and M. P. Wellman, "Nash Q-learning for general-sum stochastic games." *Journal of Machine Learning Research*, vol. 4, pp. 1039–1069, Jan. 2003.

[42] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and zipf-like distributions: evidence and implications," in *IEEE INFOCOM '99*, Mar. 1999, pp. 126–134.

[43] 3GPP, "Technical specification group radio access network; study on LTE device to device proximity services," *TR 36.843, Release 12*, pp. 33–46, 2014.

[44] K. Zhu, W. Zhi, L. Zhang, X. Chen, and X. Fu, "Social-aware incentivized caching for D2D communications," *IEEE Access*, vol. 4, pp. 7585–7593, Oct. 2016.