Two Time-Scale Caching Placement and User Association in Dynamic Cellular Networks

Tiankui Zhang, *Senior Member, IEEE,* Yue Wang, Wenqiang Yi, *Member, IEEE,* Yuanwei Liu, *Senior Member, IEEE,* Chunyan Feng, *Senior Member, IEEE* and Arumugam Nallanathan, *Fellow, IEEE*

Abstract

With the rapid growth of data traffic in cellular networks, edge caching has become an emerging technology for traffic offloading. We investigate the caching placement and content delivery in cache-enabling cellular networks. To cope with the time-varying content popularity and user location in practical scenarios, we formulate a long-term joint dynamic optimization problem of caching placement and user association for minimizing the content delivery delay which considers both content transmission delay and content update delay. To solve this challenging problem, we decompose the optimization problem into two sub-problems, the user association sub-problem in a short time scale and the caching placement in a long time scale. Specifically, we propose a low complexity user association algorithm for a given caching placement in the short time scale. Then we develop a deep deterministic policy gradient based caching placement algorithm which involves the short time-scale user association decisions in the long time scale. Finally, we propose a joint user association and caching placement algorithm to obtain a sub-optimal solution for the proposed problem. We illustrate the convergence and performance of the proposed algorithm by simulation results. Simulation results show that compared with the benchmark algorithms, the proposed algorithm reduces the long-term content delivery delay in dynamic networks effectively.

Index terms— caching placement, edge caching, multiple time scales, user association

This work was supported by National Natural Science Foundation of China under Grant 61971060. Part of this work has been presented in 2021 PIMRC [1].

Tiankui Zhang, Yue Wang and Chunyan Feng are with the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: {zhangtiankui, 199712wang, cyfeng}@bupt.edu.cn).

Wenqiang Yi, Yuanwei Liu and Arumugam Nallanathan are with the School of Electronic Engineering and Computer Science, Queen Mary University of London, London E1 4NS, U.K. (e-mail: {w.yi,yuanwei.liu, a.nallanathan}@qmul.ac.uk).

I. Introduction

In recent years, with the popularization of mobile devices and the growing of multimedia applications, the mobile Internet data traffic and content diversity have grown explosively. The traditional network structure cannot adapt to the rapid growth of traffic. In order to meet the huge demands for content traffic in mobile cellular networks, edge caching is proposed as a promising technique [2].

Because the distribution of files requested by users follows Pareto's principle, most of the traffic is composed of a small number of files. By caching these files in base stations (BS), we can reduce the content transmission delay generated by repeated transmission of the same high-popular content, which can further reduce the pressure of backhaul link, save the network bandwidth resources and improve the quality of experience (QoE) of users [2, 3]. However, there are still some challenges in the application of edge caching technology, such as the limited storage space of BS and the demand for massive content. Therefore, which file should be cached in the BS is particularly important for content delivery in cellular networks.

Edge caching technology mainly considers how to implement caching placement and content delivery. Many researchers have studied on the caching placement strategy in cellular network, which mainly solves the problem of which content should be cached in BS [4–10]. In [4], according to the topology of fog radio access networks (Fog-RANs), both centralized and distributed transmission aware caching placement strategies were designed to minimize users' average download delay. Considering the diverse content preference of different users, the authors maximized a weighted sum of network performance and user fairness to obtain the optimal caching placement strategy in [5]. In [6], a distributed caching placement algorithm based on belief propagation (BP) was developed to minimize the downloading latency in heterogeneous cellular networks (HCN).

Besides, there are many studies focusing on improving caching performance by leveraging device-to-device (D2D) communication or unmanned aerial vehicle (UAV) communication [11–18]. Considering the limited backhaul capacity, the authors studied UAV assisted secure transmission for scalable videos in hyper-dense networks via caching in [11]. In [12], the authors proposed a scheme to guarantee the security of UAV-relayed wireless networks with caching via jointly optimizing the UAV trajectory and time scheduling. According to the popularity, the caching placement algorithm was formulated to maximize the total offloading probability of the D2D

system [13]. A recent work [14] developed a model for synthesizing user preference from content popularity and proposed a caching placement algorithm to maximize the offloading probability for cache-enabled D2D communications. The authors jointly optimized the uncoded cache placement and linear coded D2D delivery to minimize the worst-case D2D delivery load in a cache-aided D2D system [15]. Condidering both UAV and D2D communications, [16] focused on the caching placement and trajectory optimization of cache-enabled networks. In [18], considering a multi-UAV assisted wireless network, the authors jointly optimized caching placement, UAV trajectory and transmission power to maximize the minimum throughput.

Considering caching placement, jointly optimizing caching placement and content delivery can greatly improve cache performance gains [19, 20]. In [19], the service caching placement, computation offloading decisions, and system resource allocation were jointly optimized in mobile edge computing (MEC) systems. In [20], the authors minimized the average file transmission delay by jointly optimizing caching placement and bandwidth allocation in two-tier heterogeneous networks. Caching placement strategy is strongly coupled with user association strategy. While caching placement strategy directly affects the deployment of user association strategy, the result of user association can indirectly affect the design of caching placement strategy [21–24]. In [21], the authors formulated a joint optimization of the caching placement and user association policy to reduce the content delivery delay. In [23], considering the conditions of the backhaul in terms of delay and wireless channel quality, a joint optimization of the caching placement and user association policy was studied in HCN. In cache-enabled dense small cell networks (DSCNs), the authors in [24] optimized energy consumption and file delivery delay in the same time. Moreover, they obtained the optimal caching placement and user association at two separate stages (caching stage and delivery stage), respectively.

However, the above studies only consider edge caching technology in static scenarios. In practical scenarios, user locations, user requests and wireless channel environments are all time-varying. The optimum at a certain point of time cannot guarantee the optimum over a long period of time. In Fog-RANs, a distributed deep Q-learning-based content caching scheme based on user preference prediction and content popularity prediction was proposed in [25], which maximized the caching hit rate. In [26], the authors took advantage of time-varying user terminal movement and content popularity, and then modeled these dynamic networks as a stochastic game to design a cooperative caching placement strategy. In [27], the authors proposed a long-term caching placement and resource allocation optimization problem for the content delivery delay

minimization in dynamic UAV-assisted cellular networks. The researchers in [28] considered the content popularity in terms of time and space, and then proposed an online content popularity prediction algorithm.

The above works consider the caching placement and content delivery in the same time scale. However, while the wireless channel environment changes in milliseconds, content popularity changes much slower in practical scenarios. For example, a newly released movie would stay popular for a few days, and then it gradually becomes less popular. Due to the dynamic changes of user location and channel environment, user association should be performed in a short time scale. Because content caching placement is largely affected by content popularity, the content caching placement should be updated in a long time scale. If content caching placement and user association are operated in the same time scale, edge servers need to update the cached service frequently. Considering the limited capacity storage and bandwidth of the edge server, frequent updates bring additional burden to the system [29] and deteriorate the user experience.

Therefore, a few works have considered both caching placement and content delivery over multiple time scales [30–36]. In [30], the researchers proposed an iterative algorithm to solve the joint optimization of content placement and content delivery in different time scales. In [31], under the constraint of user satisfaction, a file caching scheme was proposed to minimize the average time. In [32], the authors proposed a two time-scale dynamic caching scheme for adaptive bitrate streaming in vehicular networks. In [33], in order to enhance the QoE for the video streaming, the authors designed the long-term transmission-aware caching placement and the short-term transmission strategy in cloud radio access networks. In [34], the authors proposed an optimal BS-user association algorithm in the short time scale and a greedy content caching algorithm in the long time scale. In [35], the authors studied the BS association and file caching problem considering the spatial diversity of the file popularity in different time scales. All of these studies assume that the entire network states are known beforehand or the law of change is known. They are suitable for some specific use cases or need some special knowledge in advance. Dynamic network states are complex and unstable, and hence it is difficult to get the states directly. It requires intelligent mechanisms that can adapt to these variations.

Furthermore, reinforcement learning based algorithms have been an emerging tool to obtain optimal solutions [25, 26, 37, 38]. In [37], the authors proposed an extended deep deterministic policy gradient algorithm (DDPG) to learn control policies of UAV over multiple objectives. In [38], a multi-agent multi-armed bandit algorithm is developed for to solve the D2D caching

problem. However, these studies do not consider the time-varying content popularity and user 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 in the same time scale and ignore the differences in the change rate of user locations and content popularity. A few works have considered the time-scale problem, but they rarely consider that the dynamic network states are unknown. Therefore, we adopt intelligent mechanism to learn the change law of dynamic networks in different time scales. In addition, they omit the delay caused by the dynamic update of cached content. In actual systems, cached content needs to be obtained from the content provider's server, and this delay cannot be ignored. Sometimes, it affects the system performance significantly.

This article studies the joint optimization of multiple time-scale caching placement and user association in a dynamic environment with time-varying user location and content popularity. We propose an optimization problem to minimize the long-term sum delay of content transmission delay and content update delay. In order to solve this long-term dynamic optimization problem, we optimize caching placement and user association in a long and a short time scales, respectively. Due to the high complexity of the long-term problem in dynamic networks, we utilize deep reinforcement learning algorithm to learn the strategy by interacting with environment and obtain caching placement. Meanwhile, to reduce the computational complexity, we adopt a distributed computing process to obtain the user association. The main contributions of this paper are summarized as follows:

- We propose a framework of cache-enabling cellular networks for multiple time-scale caching placement and user association. In the dynamic network with time-varying user location and content popularity, we consider the spatial and temporal diversity of users' requests and the realistic time-scale separation. Then we propose a joint dynamic optimization problem of caching placement and user association to minimize the long-term content delivery delay. Specifically, the content delivery delay is composed of content transmission delay and content update delay.
- We propose a joint optimization algorithm of caching placement and user association based on multiple time-scales. Since the time scale of user movement is shorter than content popularity, we decompose the optimization problem into two sub-problems. For short time-

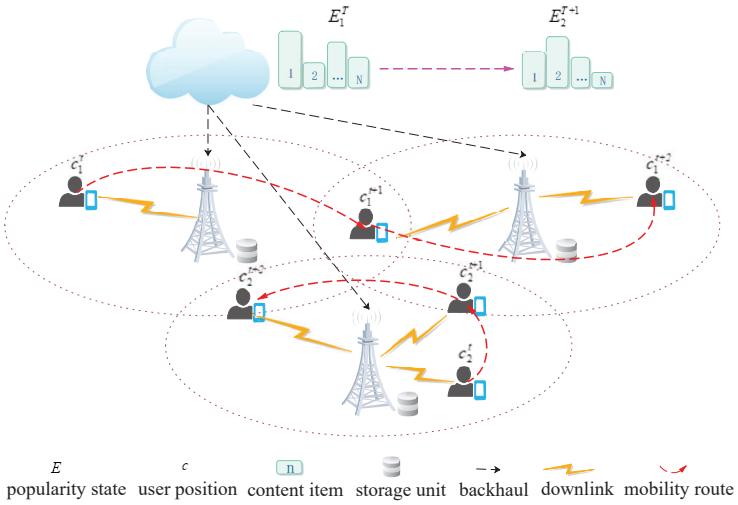

Fig. 1: Cache-enabling cellular networks with dynamic user locations and content popularity.

scale user association, we propose a low-complexity distributed user association algorithm based on BP. For long time-scale caching placement, we model the sub-problem as a Markov decision process (MDP) and propose a caching placement algorithm based on DDPG. We realize the combination of BP algorithm and DDPG algorithm. Specifically, the proposed DDPG algorithm is improved by BP which helps calculate the reward in DDPG.

- We provide the simulation results to verify the convergence and network performance of the proposed algorithm. The simulation results show that the joint optimization of caching placement and user association can significantly improve content delivery efficiency. Compared with the benchmark algorithms, the proposed algorithm can effectively reduce the content transmission delay and content update delay, and obtain a better network performance.

B. Organization

The rest of this paper is organized as follows: We present the system model and formulate the optimization problem in Section II. In Section III, we propose an iterative algorithm to solve the optimization problem. Then, simulation results are provided in Section IV and conclusions are drawn in Section V. The main symbols and variables used in this paper are summarized in Table I.

II. System Model and Problem Formulation

As shown in Fig. 1, we consider a cellular network consisting of M BSs and K mobile users. We define the set of BSs as $\mathcal{M} = \{1, 2, ..., M\}$. Each BS is equipped with a cache device with the cache capacity of z bits. Due to its capacity limitation, it can only store some of the files in the file library. BS is connected to the core network through backhaul links and all files are stored in the core network. We denote the set of users dy $\mathcal{K} = \{1, 2, ..., K\}$. We assume that users move within a fixed area, such as a university campus.

Suppose that the library has N files, denoted by $\mathcal{N} = \{1, 2, ..., N\}$. Without loss of any generality, we assume all the files have the same size of f bits. Because the file can be divided into blocks of the same length [2]. BS obtains files from core network through backhaul links and caches popular files during off-peak time.

As shown in Fig. 1, we consider a dynamic scenario with time-varying user location and content request. We optimize a long-term caching placement problem. The entire caching placement process is represented by the time series $\mathcal{T} = \{1, 2, ..., T\}$. T represents a time frame, which is the long time scale. We suppose that the content popularity changes over the long time scale. Each BS performs a pre-cache at the beginning of each time frame. The cached contents remain unchanged during a time frame. For the convenience of description, each T is discretized d into several equal time slots denoted by $T = \{1, 2, ..., t\}$. We assume that $T = Gt$, where G is a constant, t represents a time slot, which is the short time scale. User position changes over the short time scale and users request files from BS in each time slot t.

A. Transmission Model

User mobility can be quantified as a finite Markov state transition model $k \in \mathcal{K}$, $c_k \in \mathcal{C} = \{C_1, C_2, \cdots, C_I\}$. We assume that there are I states. $\mathcal{C}$ represents the set of all states of possible user locations and the transition probability between different states is represented as $P\{C^{t+1}|C^t\}$. We assume that the user location state C^t is constant in time slot t, and changes in the next time slot. The connectivity between the users and the BSs in time slot t is denoted by a $K \times M$ matrix L^t. $l^t_{km} = 1$ indicates that user k associates with BS m in time slot t, and $l^t_{km} = 0$ otherwise. Since each user associates with one BS in the same time slot, we have $\sum_{\mathrm{m}\in\mathcal{M}} l^t_{\mathrm{km}} = 1$. The set of users who associate with BS m in time slot t is denoted by $\mathcal{U}_m = \{k \in \mathcal{K} | l^t_{km} = 1\}$.

In this work, the symbol B represents system bandwidth, p_m is the average transmission power at BS m, and σ^2 is the variance of the Gaussian noise. The path-loss between BS m and the

TABLE I: Main Symbol and Variable List

Notation	Description
M	Number of BSs
K	Number of users
N	Number of contents
G	The ratio of long time scale to short time scale $T = Gt$
z	Cache capacity of BS
f	Size of content
T	The long time scale
t	The short time scale
l_{km}^t	Indicator of whether user k associated with BS m in time slot t
x_{nm}^T	Indicator of whether content n cached in BS m in time frame T
$\mathcal{C} = \{C_0, C_1, \cdots, C_I\}$	User position Markov state set
$\mathcal{E} = \{E_0, E_1, \cdots, E_J\}$	Content popularity Markov state set
$P\{C^{t+1}\|C^t\}$	The transition probability of user position
$P\{E^{T+1}\|E^T\}$	The transition probability of content popularity
p_{nk}^T	Probability of user k request content n in time frame T
B	System bandwidth
p_m	Transmission power of BS m
σ^2	Variance of the Gaussian noise
h_{km}^t	Channel coefficient between BS m and user k in time slot t
α	Pathloss exponent
$SINR_{km}^t$	SINR of user k associated with BS m in time slot t
R_{km}^t	Transmission rate from BS m to user k in time slot t
D_{nk}^t	Content transmission delay when user k requests content n in time slot t
D_1^T	The average delay in time frame T
D_2^T	Content update delay in time frame T

user k is modeled as $(d_{km}^t)^{-\alpha}$, where d_{km}^t is the distance between BS m and user k in time slot t, and α is the path-loss exponent. The random channel between BS and user is Rayleigh fading, whose coefficient h_{km}^t has the average power of one. We assume that all the downlink channels spanning from the BSs to the users are independent and identically distributed (i.i.d.).

The signal-to-interference-plus-noise ratio (SINR) of user k from BS m in time slot t is

$$SINR_{km}^{t} = \frac{\left(h_{km}^{t}\right)^{2}\left(d_{km}^{t}\right)^{-\alpha}P_{m}}{\sum\limits_{q\in\mathcal{M}\setminus\{m\}}\left(h_{kq}^{t}\right)^{2}\left(d_{kq}^{t}\right)^{-\alpha}P_{q}+\sigma^{2}} \ . \tag{1}$$

Users can access to the system only if the signal can meet the minimum signal threshold according to 3GPP standards.

In order to make full use of the spectrum resources, we divide the bandwidth according to the number of users associated with the BS [28]. Therefore, the downlink transmission rate from BS m to user k in time slot t is

$$R_{km}^{t} = \frac{B}{\sum_{k=1}^{K} l_{km}^{t}} \log\left(1 + SINR_{km}^{t}\right) \ . \tag{2}$$

B. Content Request and Cache Model

Content popularity distribution is based on users' requests for contents. We assume the probability for users request each content, i.e., popularity of each content, follows a Zipf distribution. Let $p_{nk}^{T} = 1$ indicates that user k requests the content n in time frame T, otherwise $p_{nk}^{T} = 0$. Content popularity is time-varying, and we model this time-varying popularity as a finite state Markov sequence $E \in \mathcal{E} = \{E_1, E_2, \cdots, E_J\}$. Each state is a probability distribution of users' requests. We assume that there are J states. The popularity of the content chunk k transfers over time between the states, and the transition probability is represented by $P\{E^{T+1}|E^{T}\}$. We assume that the content popularity E^{T} is constant in time frame T, and changes in the next time frame.

We assume that each BS has a finite-capacity storage, denoted by z, and the BS actively caches content in each time frame. If the requested content is cached in the associated BS, the content will be transmitted directly to the user. Otherwise, the BS would obtain the content from core network through backhaul link, and then transmits it to the user. Let us denote caching placement strategy by X^{T}, where $x_{nm}^{T} = 1$ represents the BS m caches content n in time frame T, otherwise $x_{nm}^{T} = 0$. The set of BSs which cache content n in time frame T is denoted by $\mathcal{A}_n = \left\{m \in \mathcal{M} | x_{nm}^{T} = 1\right\}$.

C. Delivery Delay Model

In our system, the content delivery delay is composed of content transmission delay and content update delay.

1) Content Transmission Delay: Since each time frame T is divided into several time slots t, the content transmission delay is composed of content transmission delay in each time slot t.

The content transmission delay from BS m to user k in time slot t is represented by D_{nk}^t. Since each user can associate with one fix BS in the same time slot, we consider the condition that $l_{km}^t = 1$. According to whether BS caches content n, we have the following two cases.

Case 1: BS m has cached content n in time slot t, and user k can directly get content n from BS m. The content transmission delay in time slot t is

$$D_{nk}^t = p_{nk}^T \frac{f}{R_{km}^t}, l_{km}^t x_{nm}^T = 1 \ . \tag{3}$$

Case 2: BS m does not cache content n in time slot t, which is $l_{km}^t x_{nm}^T = 0$. BS m downloads content n from core network and then transmits the content to users. BSs connect to core network through a wired backhaul link. Without loss of generality, we assume that the wired backhaul link is one hop, and the average backhaul delay from BS to core network is [39]

$$D_0 = \left(\left(1 + 1.28\frac{\lambda_\mathrm{b}}{\lambda_g}\right)\kappa\right)(a + bf) \ , \tag{4}$$

where λ_b and λ_g denote the density of BSs and gateways, respectively. In addition, a, b and κ are constants that reflect the processing capability of the nodes. Specifically, the first and second terms represent the effect of number of connecting nodes and packet size on delay, respectively. It is obvious that the bigger the package size, the bigger the delay. The content transmission delay in time slot t is

$$D_{nk}^t = p_{nk}^T D_0 + p_{nk}^T \frac{f}{R_{km}^t}, l_{km}^t = 1, x_{nm}^T = 0 \ . \tag{5}$$

So the content transmission delay when user k requests content n in time slot t is

$$D_{nk}^t = \sum_{m \in \mathcal{M}} l_{\mathrm{km}}^t \left(\left(1 - x_{nm}^T\right) p_{nk}^T D_0 + p_{nk}^T \frac{f}{R_{km}^t}\right) \ . \tag{6}$$

The average delay in time frame T is

$$D_1^T = \frac{1}{K} \sum_{t \in T} \sum_{k \in \mathcal{K}} \sum_{n \in \mathcal{N}} D_{nk}^t \ . \tag{7}$$

Remark 1. *From (6), we notice that the cache capacity z of BS affects caching placement strategy X^T. The larger z causes a decrease in content transmission delay. At the same time, the larger z can get higher cache hit ratio and lower backhaul bandwidth cost.*

2) Content Update Delay: The caching placement strategy is updated every time frame, so content update delay is determined by the delay caused by downloading updated content from core network. According to the caching placement strategy, if the required content n has been cached at the BS m in the $T-1$ time frame denoted by $x_{nm}^{T-1}=1$, there is no need to re-download the content in the T time frame. Otherwise $x_{nm}^{T-1}=0$, the BS would download the content from core network. Therefore, content update delay in time frame T is

$$D_2^T=\frac{1}{M}\sum_{n\in\mathcal{N}}\sum_{m\in\mathcal{M}}D_0\max\left\{0,\left(x_{nm}^T-x_{nm}^{T-1}\right)\right\}\ , \tag{8}$$

where D_0 is the average backhaul delay from BS to core network according to (4).

Remark 2. *From (6) (8), we notice that the content transmission delay and the content update delay are obtained in short time scale and long time scale respectively. The long time scale T is determined by the time interval of content popularity change and the short time scale t is determined by the time interval of user location change. The ratio of long time scale to short time scale G affects caching placement strategy X^T and user association L^t. G reflects the relative speed of the change of content popularity and the change of user location. The larger G causes an decrease in content update delay.*

D. Problem Formulation

Given the above models, we formulate the joint optimization problem of caching placement and user association for minimizing the long-term content delivery delay. The content delivery delay in each time frame contains content transmission delay and content update delay. Let us hence formulate the delay optimization problem as follows:

$$\min_{X^T L^t}\bar{D}=\sum_{T\in\mathcal{T}}\left(D_1^T+D_2^T\right) \tag{9a}$$

$$\text{s.t.}\sum_{n\in\mathcal{N}}x_{nm}^T f\le z,\forall m\in\mathcal{M} \tag{9b}$$

$$x_{nm}^T\in\{0,1\},\forall n\in\mathcal{N},\forall m\in\mathcal{M} \tag{9c}$$

$$\sum_{\mathrm{m}\in\mathcal{M}}l_{\mathrm{km}}^t=1,\forall \mathrm{k}\in\mathcal{K} \tag{9d}$$

$$l_{km}^t\in\{0,1\},\forall \mathrm{k}\in\mathcal{K},\forall m\in\mathcal{M} \tag{9e}$$

The constrain (9b) is the caching capacity limitation of each BS. The constrain (9d) means each user can associate with one BS in the same time slot t. The optimization problem in (9) is an integer programming problem, which is NP-complete. Hence, it is very challenging to find the optimal solution to problem (9). In the next section, we show how to solve this optimization problem.

Remark 3. *From (7), we find that the long-term content delivery delay contains the content transmission delay of each user. From (9d), we notice that the user must access to a BS according to user association strategy. Hence, the system ensures that each user's request would be satisfied in each time slot. By this way, the system can guarantee that all the access users can get the basic service. This means that our optimization objective focuses on the system performance as well as users' quality of service.*

III. Proposed Algorithm for Minimization Delay

Compared with the time scale of content popularity changing, content request occurs in a shorter time scale. Therefore, we apply a multiple time-scale approach to make user association and caching placement decisions respectively. Generally, a short time-scale user association decision can affect the caching placement decision for the next time frame. The long time-scale caching placement decision can affect the short time-scale user association decision in this time frame.

We decompose the optimization problem into the following two sub-problems: (i) a short time-scale user association algorithm; (ii) a long time-scale caching placement algorithm. Every time slot, the user association decision is made based on the known caching placement decision. However, caching placement decision must consider both the time varying user association and the content popularity. We now discuss how to address the two sub-problems.

1) User association problem: For a given caching placement solution X^T and user location C^t, probability of user request E^T, the problem becomes the following short time-scale problem:

$$\begin{aligned} &\min_{L^t} \bar{D}^{\mathrm{t}} = \tfrac{1}{K} \sum_{k\in\mathcal{K}} \sum_{n\in\mathcal{N}} D^t_{nk} \\ &\text{s.t.} \sum_{\mathrm{m}\in\mathcal{M}} l^t_{\mathrm{km}} = 1, \forall \mathrm{k} \in \mathcal{K} \\ &\quad l^t_{km} \in \{0,1\}, \forall \mathrm{k} \in \mathcal{K}, \forall m \in \mathcal{M} \end{aligned} \quad . \tag{10}$$

2) Content caching placement problem: Based on the user association $\{L^t, L^{2t}, \cdots, L^{xt}\}$ in time frame $T-1$, the problem becomes the following long time-scale problem:

$$\begin{aligned} &\min_{X^T} \bar{D} = \sum_{T\in\mathcal{T}} \left(D_1^T + D_2^T\right) \\ &\text{s.t.} \sum_{n\in\mathcal{N}} x_{nm}^T f \le z, \forall m \in \mathcal{M} \\ &\quad x_{nm}^T \in \{0,1\}, \forall n \in \mathcal{N}, \forall m \in \mathcal{M} \end{aligned} . \tag{11}$$

A. Short Time-Scale User Association Algorithm

There is a complicated global functions of many discrete variables to solve. Although we can exploit the exhaustive search algorithm to solve the optimization problem, the algorithm complexity is exponential. So it is only suitable for small scale networks. With a problem of product form, the given function can be decomposed into several "local" functions. We can derive the approximately various marginal functions from the global function by BP algorithm. Compared with centralized computing algorithm, the BP algorithm reduces the computational complexity greatly. Meanwhile, the BP algorithm can achieve a good balance in convergence and accuracy [4, 40].

We propose a BP based distributed algorithm to solve the user association problem (10) by sum-product algorithm [41]. The distributed BP algorithm is based on the factor graph. In the bipartite factor graph, we divide the vertex set into variable nodes and function nodes. The two vertexes connected by one edge called adjacent nodes. And each variable node is adjacent to one or several function nodes. Similarly, each function node is connected to one or several variable nodes. We now establish the factor graph model [41] and map the optimization problem (10) to it.

1) Factor Nodes: To apply BP algorithm, we make some equivalent transformation to the optimization problem. Problem (10) is equivalent to maximizing $-\frac{1}{K}\sum_{k\in\mathcal{K}}\sum_{n\in\mathcal{N}} D_{nk}^t$, subject to the constraints $\sum_{\mathrm{m}\in\mathcal{M}} l_{\mathrm{km}}^t = 1$ for all k. The problem (10) is first transformed into an unconstrained optimization problem as

$$\max_{L^t} \prod_{\mathrm{k}\in\mathcal{K}} \mathrm{h}_k^t\left(L\right) \prod_{n\in\mathcal{N},\mathrm{k}\in\mathcal{K}} \eta_{nk}^t\left(X, L\right), \tag{12}$$

where

$$\eta_{nk}^t\left(X, L\right) = \exp\left(-D_{nk}^t\right), \tag{13}$$

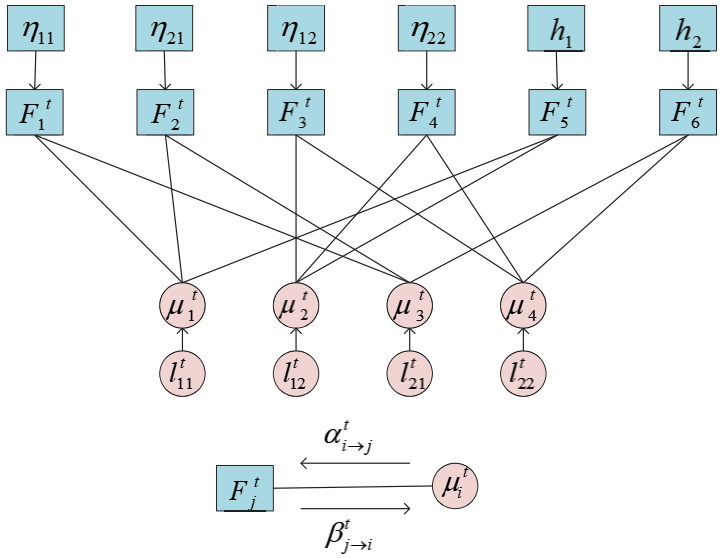

Fig. 2: An example of 2 users, 2 BS and 2 content items.

$$h_k^t(L) = \begin{cases} 1, \sum\limits_{\mathrm{m}\in\mathcal{M}} l_{\mathrm{km}}^t = 1, \\ 0, \text{otherwise.} \end{cases} . \tag{14}$$

In this product form, the optimization problem can be decomposed into many factors, such as $\eta_{nk}^t(X)$ and $h_k^t(L)$. To simplify the calculation, we define all the factors by (15). Every factor corresponds to a factor node in the factor graph. As shown in Fig. 2, we establish the factor graph model [41] and map the optimization problem (10) to it by the following rule:

$$F_j^t \doteq \begin{cases} \eta_{nk}^t, j = \sum\limits_{l=1}^{k-1} |\mathcal{F}_l| + \xi(n,k), \\ \mathrm{h}_k^t, j = \sum\limits_{k=1}^{K} |\mathcal{F}_k| + k. \end{cases} \tag{15}$$

where $\mathcal{F}_{\mathrm{k}} = \{f_n | p_{nk} > 0\}$ denotes the set of files which may be requested by user k, and $|\mathcal{F}_{\mathrm{k}}|$ is the number of elements in the set, and $\xi(n,k)$ denotes the index of file k in the set $\mathcal{F}_k$.

2) Variable Nodes: Each variable node is related to a user association strategy. For simplicity, the two-dimensional matrix L^t is transformed into an one-dimensional vector by the following rule:

$$\mu_i{}^t \doteq \mathrm{l}_{km}^t, \mathrm{i} = \mathrm{M}(k-1) + m \tag{16}$$

The process of BP based user association algorithm is as follows.

1) Initialization: Initialize the variable nodes and messages.

2) Message Update: Let $m_{\mu_i \to F_j}^{t+1}(l)$ denotes the message from a variable node to a factor node defined by

$$m_{\mu_i \to F_j}^{t+1}(l) = \prod_{h \in \Gamma_i^{\mu} \backslash \{j\}} m_{F_h \to \mu_i}^{t}(l). \tag{17}$$

And $m_{F_j \to \mu_i}^{t+1}(\mathrm{l})$ denotes the message from a factor node to a variable node defined as

$$m_{F_j \to \mu_i}^{t+1}(l) = \max_{\sim\{i\}} \left\{ F_j(l) \prod_{h \in \Gamma_{\mathrm{j}}^{F} \backslash \{i\}} m_{\mu_h \to F_j}^{t}(l_h) \right\}. \tag{18}$$

The traditional BP algorithm updates the message by sum-product algorithm according to (17) (18) [41, 42]. We simplify the traditional messages to the log-domain message ratios [4], denoted as $\alpha_{\mathrm{i}\to j}^{t} = \log\left(\frac{m_{\mu_i \to F_j}^{t}(1)}{m_{\mu_i \to F_j}^{t}(0)}\right)$ and $\beta_{j\to i}^{t} = \log\left(\frac{m_{F_j \to \mu_i}^{t}(1)}{m_{F_j \to \mu_i}^{t}(0)}\right)$, respectively.

Variable Node Update: We have

$$\alpha_{i\to j}^{t+1} = \sum_{l \in \Gamma_i^{\mu} \backslash \{j\}} \beta_{l\to i}^{t}. \tag{19}$$

Factor Node Update: When $F_j \doteq \eta_{nk}$, we have

$$\beta_{j\to i}^{t+1} = p_{nk}^{t}\left(D_{nk}^{t}\left(L_{i,0}^{t}\right) - D_{nk}^{t}\left(L_{i,1}^{t}\right)\right) \tag{20}$$

where L obtains from the message obtained in the previous iteration.

When $F_j \doteq h_k$, we have

$$\beta_{j\to i}^{t+1} = -\alpha_{l\to j}(t) \tag{21}$$

3) Belief Update: The belief ratio in the logarithmic domain can be obtained by

$$\tilde{b}_i^t = \log\left(\frac{b_i^t(1)}{b_i^t(0)}\right) = \sum_{j \in \Gamma_i^{\mu}} \beta_{j\to i}^{t}. \tag{22}$$

As a result, the estimation of $\hat{\mu}_i^t$ can be expressed as

$$\hat{\mu}_i^t = \begin{cases} 1, \text{if } \tilde{\mathrm{b}}_i^t > 0, \\ 0, \text{if } \tilde{\mathrm{b}}_i^t < 0. \end{cases} \tag{23}$$

In each iteration, each variable node updates its belief and makes an estimate of the belief

until it converges.

The process of BP based user association algorithm is summarized in **Algorithm 1**.

Algorithm 1 BP based user association algorithm

Initialize: Set BP message $\alpha_{i\to j}^{t+1}=0, \beta_{j\to i}^{t+1}=0 \forall i,j$. Initial ${\mu_i}^t$ and define node ordering η.
repeat
  for each node $\eta_k \in \eta$ **do**
    Update the message $\alpha_{i\to j}^{t+1}$ according to (19).
    Update the message $\beta_{j\to i}^{t+1}$ according to (20) (21).
    Calculate the belief $\tilde{b}_i^t$ according to (22).
    Estimate $\hat{\mu}_i^t$ using (23) and update ${\mu_i}^t$.
  end for
until Convergence
Output: user association L^t.

B. Long Time-Scale Content Caching Algorithm

Because of the bandwidth cost, the caching placement strategy must consider the update cost. It may not be the optimal strategy in this moment. In order to solve this complicated long-term optimization problem, we exploit DDPG algorithm to interact with the environment to learn the optimal caching placement strategy.

We propose a DDPG based caching placement algorithm for minimizing the long-term content delivery delay in dynamic networks. Specifically, in the short time scale, we obtain the user association based on the caching placement decisions which is unchangeable during the long time scale. Then we calculate the long-term reward and solve the long-term optimization problem. Every long time scale, the reward is obtained by accumulating the short time-scale reward.

1) Markov Decision Process Model: We model the sub-problem as MDP which is defined by $\langle \mathcal{A},\mathcal{S},\mathcal{P},r,\gamma \rangle$. In the MDP, the agent will learn how to choose the action to get more future reward.

A. $\mathcal{S}$ is the set of states, $s_j \in S$, which indicates the system information. The feature of state s^T is composed of content popularity in time frame T, denoted by $s^T{=}E^T$. The state space is equivalent to all possible content popularity states.

B. $\mathcal{A}$ is the set of actions, which are caching strategies. The feature of action a^T is composed of BSs' caching strategies in time frame T, denoted by $a^T{=}\mathrm{X}^T$. The action space is equivalent to of all possible X, which is 2^{mn}.

C. The state transition probability is denoted by $P\left(s^{T+1}|s^T, a\right)$.

D. r represents the instant reward, which means the long time-scale delivery delay in our system. Specifically, with the given system state, we can get the optimal user association in every time slot by BP. According to the optimal user associations $\{L^t, L^{2t}, \cdots, L^{xt}\}$ in this time frame, we can obtain the average long time-scale delivery delay by $r = D_1^T + D_2^T$.

E. γ is the discount factor, indicating the coefficient for calculating the cumulative reward.

At each iteration, the agent observes the current state and then chooses an action to execute. After this, the environment will give a reward r to the agent and move into the next step according to the state transition probability $P\left(s^{T+1}|s^T, a\right)$. After several iterations, the agent can learn the optimal caching placement strategy and obtain an optimal long-term reward.

MDP describes the long-term loss of the current action by defining value function denoted by $Q(s, a)$, which is expressed as

$$Q(s, a) = \mathbb{E}\left\{\sum_{\tau=T}^{\mathcal{T}} \gamma^{\tau-T} r_\tau|_{s=s_T, a=a_T)}\right\}, \tag{24}$$

where s_T and a_T are the state and the action of time frame T respectively. The optimal solution is calculated through $\pi^*(s) = \arg\min_a Q^*(s, a), \forall s \in S$, which is the optimal caching placement strategy. Then the value function is updated when the network chooses an action a under the state s by Bellman equation, which is defined as

$$Q_{T+1}(s, a) = (1-\eta) Q_T(s, a) + \eta\left\{r + \gamma \max_a Q(s', a')\right\}, \tag{25}$$

where s, a and r represent the state, the action and the reward of time frame T. s' and a' are the state and the action of time frame $T+1$ and η represents the learning rate. So we can find the optimal dynamic caching placement by (24) (25).

2) Deep Deterministic Policy Gradient: The DDPG is model free and uses deep neural network for function approximation. As shown in Fig. 3, DDPG is an actor-critic method, which has actor network and critic network [43]. The actor network and critic network both have a target network and an online network. The actor network generates the determine action on the current state. The critic network obtains the Q-value and evaluates the current action by calculating the temporal-difference (TD)-error. At the same time, it determines the direction and speed for updating the actor network.

In the critic network, $Q(s, a)$ is learned using the Bellman equation (25). We train the critic

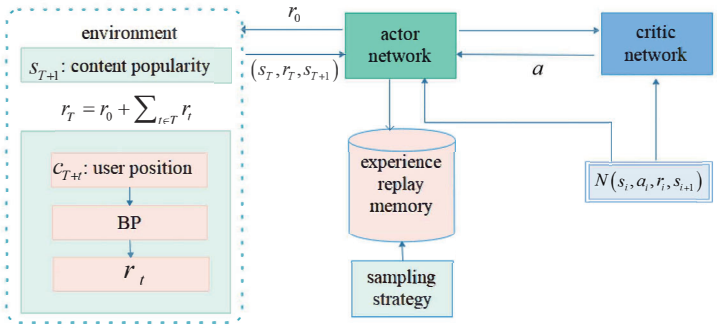

Fig. 3: Framework of the proposed DDPG.

network by minimizing the TD-error:

$$L\left(\theta^{Q}\right)=\frac{1}{Y}\sum_{i}\left(y_{i}-Q\left(s_{i},a_{i}|\theta^{Q}\right)\right)^{2}, \tag{26}$$

where

$$y_{i}=r_{i}\left(s_{i},a_{i}\right)+\gamma Q'\left(s_{i+1},\mu'\left(s_{i+1}|\theta^{\mu'}\right)|\theta^{Q'}\right). \tag{27}$$

And Y is the number of samples which represented by (s_i, a_i, r_i, s_{i+1}). Q' and Q are the Q-value of critic target network and critic online network, respectively. Besides, $\theta^{Q'}$ is critic target network parameter, θ^{Q} is critic online network parameter, $\theta^{\mu'}$ is actor target network parameter.

The actor network is trained by policy gradient. And we modify the action parameters of policy gradient to obtain a larger Q-value. Then the actor network updates the parameters by

$$\nabla_{\theta^{\mu}}J\approx\frac{1}{Y}\sum_{i}\left(\nabla_{a}Q\left(s,a|\theta^{Q}\right)\right)|_{s=s_{i},a=\mu(s_{i})}\nabla_{\theta^{\mu}}\mu\left(s|\theta^{\mu}\right)|_{s_{i}}, \tag{28}$$

where θ^{μ} is actor network parameter and $\mathrm{a}=\mu\left(s_{i}\right)$ is the deterministic behavior strategy.

The process of DDPG based caching placement algorithm is summarized in **Algorithm 2**.

C. Dynamic Solution to the Optimization Problem

For the optimization problem, we decompose it into two sub-problems. As shown in Fig. 4, we propose a short time-scale user association algorithm and a long time-scale caching placement algorithm to solve the optimization problem.

As we can see, the proposed user association algorithm can obtain the sub-optimal solution online. However, the proposed caching placement algorithm needs a learning process which is offline. Therefore, our system has a process to train model before application. In this process,

Algorithm 2 DDPG based caching placement algorithm

Set hyper parameters and initialize all the networks.
repeat
  for each step **do**
    Update content popularity.
    Select action and update caching placement and get the update delay r_0.
    for $t = 1$ to T **do**
      Update user location.
      Update user association L^t by Algorithm 1.
      update r by $r = r + r_t$.
    end for
    Get reward $r = r + r_0$.
    Store transition (s_T, a_T, r_T, s_{T+1}) in the memory space.
    Train the critic network with (26).
    Train the actor network with (28).
    Update the target networks by soft replace.
  end for
until Convergence
Output: caching placement X^T.

the chosen action may be not the optimal. In the dynamic network, our system can effectively solve the long-term problem after the training process. Hence, the proposed algorithm is divided into two processes. One process is the offline process described in **Algorithm 1** and **Algorithm 2**. Another process is the online process described in **Algorithm 3**.

In each time frame, our system gathers the data in the environment to obtain the content popularity state. According to **Algorithm 2**, the system calculates the caching placement strategy and transmit it to BSs. Then, BSs pre-cache the corresponding contents. In each time slot, The system updates the user location information if the users move to new positions. According to such information, the system obtains the user association strategy by **Algorithm 1**.

D. Analysis of the Proposed Algorithms

Let us analyze the temporal computational complexity and the convergency of the proposed algorithms.

Algorithm 1: In the traditional BP algorithm [41], factor nodes updating has the temporal computation complexity of $\gamma_f = O\left(N\left(N+1\right)\right)$, and variable nodes updating has the temporal computation complexity of $\gamma_v = O\left(M2^{M+1}\right)$. We assume that the algorithm can obtain the

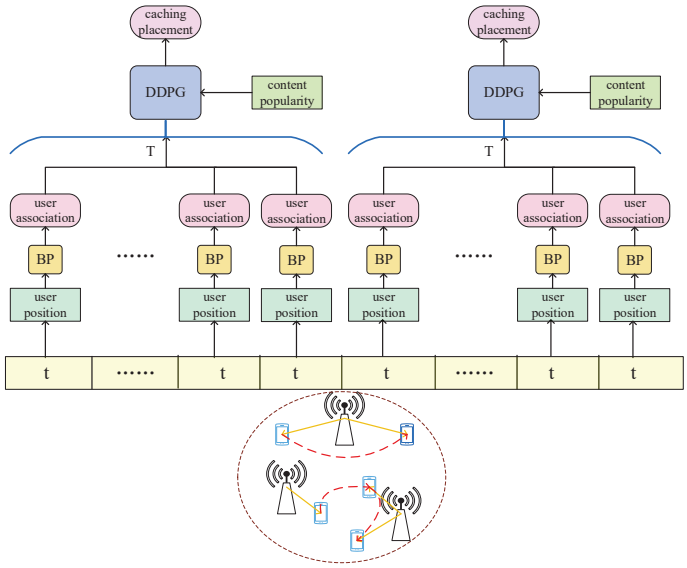

Fig. 4: Framework of the proposed two time-scale algorithm.

Algorithm 3 Online joint user association and caching placement algorithm

1: Set hyper parameters and initialize all the networks.
2: **repeat**
3: Train the networks by **Algorithm 1** and **Algorithm 2**.
4: **until** Convergence
5: **for** $\mathrm{T} \in \mathcal{T}$ **do**
6: Update content popularity.
7: **for** $\mathrm{t} \in T$ **do**
8: Update user location.
9: According to the X^{T-1}, update user association L^t by **Algorithm 1**.
10: Get the short time-scale reward according to **Algorithm 1**.
11: **end for**
12: Get the long time-scale reward.
13: According to long time-scale reward, update caching placement X^T by **Algorithm 2**.
14: **end for**

suboptimal solution with C iterations. Therefore, the complexity of traditional BP is $\gamma_{bp} = O\left(C \max\left(\gamma_f, \gamma_v\right)\right)$.

The proposed BP algorithm offloads the computation to each sub-function [4]. Factor nodes updating has the temporal computation complexity of $\gamma_f = O\left(N\left(N+1\right)\right)$, and variable nodes updating has the temporal computation complexity of $\gamma_v = O\left(\max\left(M\left(M-1\right), M\log\left(M-1\right)\right)\right)$. We assume that the algorithm can obtain the suboptimal solution with C iterations. As a result, the complexity of traditional BP is $\gamma_{bp} = O\left(C \max\left(\gamma_f, \gamma_v\right)\right)$. As mentioned above, the proposed

BP algorithm can effectively reduce the computational complexity compare to the traditional BP algorithm.

The convergence of the proposed BP algorithm depends on the structure of the graph. For graphs without circles, the maximum a posteriori (MAP) problem can be effectively calculated through the exchange of messages. The proposed algorithm can reach convergence after several iterations [42]. The performance of the BP algorithm mainly depends on the objective function, as well as the scale, sparsity and cycles of the basic factor graph. However, for other general graphs, such as graphs with cycles, the BP algorithm cannot guarantees convergence because of the cycles.

Algorithm 2: As mentioned above, we define the number of BSs as M and the number of users as K. The number of contents is expressed as N. In DDPG, two stages requesting enormous computation capacity are the training of actor network and critic network. The temporal computation complexity of actor network is $O\left(Z_{Ai}Z_{Au}+\sum_{u=1}^{U-2}Z_{Au}Z_{Au+1}+Z_{Au}Z_{AU}\right)$, where $Z_{Ai}=S$ represents the number of neurons in input layer. The number of neurons in output layer is $Z_{Au}=MN$. Therefore, the actor network has the temporal computation complexity of $O\left(\gamma_a=\delta_{MB}T_1\left(Z_{Ai}Z_{Au}+\sum_{u=1}^{U-2}Z_{Au}Z_{Au+1}+Z_{Au}Z_{AU}\right)\right)$, where δ_{MB} represents the size of mini batch, and T_1 is the upper bound of training step. The temporal computation complexity of critic network is $O\left(\gamma_b=\delta_{MB}T_1\left(Z_{Ci}Z_{Cu}+\sum_{u=1}^{U-2}Z_{Cu}Z_{Cu+1}+Z_{Cu}\right)\right)$, where the number of neurons in the input layer is $Z_{Ci}=MN+S$ and the number of neurons in the output layer is $Z_{Cu}=1$. Therefore, the proposed DDPG algorithm has the temporal computation complexity of $O\left(\max\left(\gamma_a,\gamma_b\right)\right)$.

Algorithm 3: During each iteration, two subproblem algorithms are performed to solve two subproblems. The complexity of **Algorithm 1, 2** has been analyzed above in Subsection 1, 2. The complexity of **Algorithm 3** is $O\left(\max\left(\gamma_a,\gamma_b,G\gamma_{bp}\right)\right)$, where γ_{bp} is the complexity of **Algorithm 1** and G represents the length of time frame.

IV. Numerical Results

In this section, we show the performance of the proposed algorithm based on the simulation results. We consider a cellular network with M=4 BSs, and the distance between neighboring BS is 200 m. K users are randomly distributed in the area. We model the user mobility as a finite Markov state transition model with three states $\{C_1, C_2, C_3\}$, which represent the user

location distribution respectively. We assume that the transition probabilities of the user position states are

$$P^C = \begin{bmatrix} \mathrm{p}_{11}^C & \mathrm{p}_{12}^C & \mathrm{p}_{13}^C \\ \mathrm{p}_{21}^C & \mathrm{p}_{22}^C & \mathrm{p}_{23}^C \\ \mathrm{p}_{31}^C & \mathrm{p}_{32}^C & \mathrm{p}_{33}^C \end{bmatrix} = \begin{bmatrix} 0.6 & 0.1 & 0.3 \\ 0.3 & 0.6 & 0.1 \\ 0.1 & 0.3 & 0.6 \end{bmatrix}. \tag{29}$$

Remark 4. *From (1) (2), we notice that the SINR which influences the content transmission delay is related to user position. Hence, the user location may slightly affect the delay. However, as long as the signal meets the minimum threshold, the user would access to the BS with smaller content transmission delay and get the service. It doesn't affect the service received by users and the effectiveness of the proposed algorithm. Therefore, the proposed algorithm can perform well with different distributions of user positions (29).*

The popularity of the contents follows a Zipf-like distribution. Whether the user requests the content or not is determined by the content popularity. We model the content popularity as a finite Markov state transition model with three states $\{E_1, E_2, E_3\}$, which have parameters γ_1=0.2, γ_2=0.5 and γ_3=0.7 respectively. The content popularity of content n with parameter γ_j is

$$\mathrm{e}_n^{(j)} = \frac{n^{-\gamma_j}}{\sum_{i=1}^{N} i^{-\gamma_j}}, \text{for } j = 1, 2, 3. \tag{30}$$

where $\gamma_j \geq 0$ controls the skewness of popularity. We initial the network state by s=E^1. Then we assume that the transition probabilities of the content popularity states are

$$P^E = \begin{bmatrix} \mathrm{p}_{11}^E & \mathrm{p}_{12}^E & \mathrm{p}_{13}^E \\ \mathrm{p}_{21}^E & \mathrm{p}_{22}^E & \mathrm{p}_{23}^E \\ \mathrm{p}_{31}^E & \mathrm{p}_{32}^E & \mathrm{p}_{33}^E \end{bmatrix} = \begin{bmatrix} 0.6 & 0.2 & 0.2 \\ 0.1 & 0.7 & 0.2 \\ 0.2 & 0.3 & 0.5 \end{bmatrix}. \tag{31}$$

We define the average bandwidth cost as

$$C_{bandwidth}^T = \frac{1}{M n_{req}^T} \sum_{m \in M} \sum_{\mathrm{n} \in N} f \max\left\{0, \left(x_{nm}^T - x_{nm}^{T-1}\right)\right\}, \tag{32}$$

where n_{req}^T is the number of user requests in T. The detailed simulation parameters are given in table II.

TABLE II: Simulation Parameters

Parameter	Value
Number of base stations M	4
Size of each content	10 Mbytes
System bandwidth B	10 MHz
Transmit power of BS p	46 dbm
Noise power σ_2	-176 dBm/Hz
Path loss exponent α	4
Discount factor of reward γ	0.9
Learning rate η	0.00001
Constant a	10μs
Constant b	0.01μs/bit
Constant κ	1

To prove the effectiveness of the proposed algorithm, we compare it with three benchmark algorithms which are solving the problem based on short time scale, long time scale and two time scales respectively. Specifically, we consider the following algorithms:

- BP algorithm: In each time slot, we obtain the optimal caching placement matrix and user association matrix by BP [4].
- DDPG algorithm: The caching placement and user association remain unchanged in one time frame, and we update them by DDPG algorithm which can obtain the long-term optimal caching placement and user association.
- Least frequently used (LFU) and maximum carrier to interference ratio (Max C/I) algorithm: The optimal caching placement matrix is calculated by LFU algorithm in each time frame. And the user association matrix is obtained by Max C/I algorithm in each time slot.

First, we demonstrate the convergence of the proposed algorithm by Fig. 5. In the simulation, we set $N = 20$, $K = 5$ and $T = 20$ t. Fig. 5 shows that the long-term content delivery delay gradually converges with the number of iterations increasing. Fig. 5 proves that the long-term content delivery delay with different parameters has different convergence speeds. For the parameters, v is the noise added to action selection, and ξ is the discount factor of the noise. From Fig. 5 we can see, the proposed algorithm with bigger noise v can achieve a better convergence value while it has a slightly slower convergence rate. Because the agent can get a bigger action space with the greater noise and the agent is more likely to get the optimal solution. Besides, the larger ξ means that noise reduces faster. With shorter episodes to explore the action space, the algorithm can easily obtain a local optimal solution. Otherwise, if we set ξ too small, the

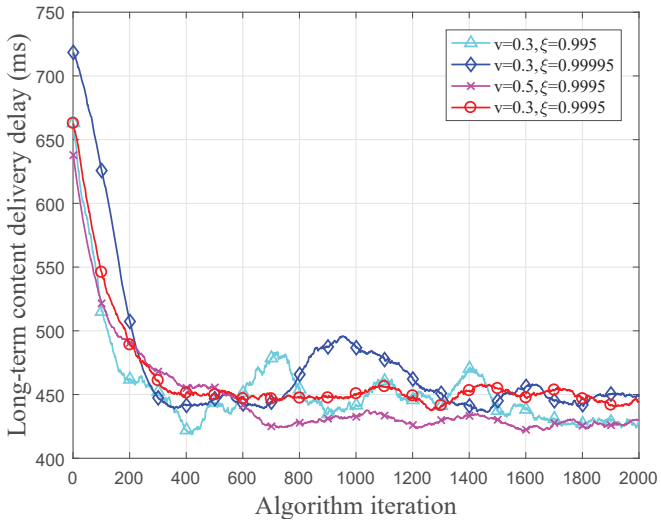

Fig. 5: Convergence of proposed algorithm comparison with different parameters.

TABLE III: Parameters of Fig. 6

Parameter	Transition probabilities of user position states	Transition probabilities of content popularity states
3C2E	$\begin{bmatrix} 0.6 & 0.1 & 0.3 \\ 0.3 & 0.6 & 0.1 \\ 0.1 & 0.3 & 0.6 \end{bmatrix}$	$\begin{bmatrix} 0.6 & 0.4 \\ 0.3 & 0.7 \end{bmatrix}$
2C3E	$\begin{bmatrix} 0.6 & 0.4 \\ 0.3 & 0.7 \end{bmatrix}$	$\begin{bmatrix} 0.6 & 0.2 & 0.2 \\ 0.1 & 0.7 & 0.2 \\ 0.2 & 0.3 & 0.5 \end{bmatrix}$
2C2E	$\begin{bmatrix} 0.6 & 0.4 \\ 0.3 & 0.7 \end{bmatrix}$	$\begin{bmatrix} 0.6 & 0.4 \\ 0.3 & 0.7 \end{bmatrix}$
3C3E	$\begin{bmatrix} 0.6 & 0.1 & 0.3 \\ 0.3 & 0.6 & 0.1 \\ 0.1 & 0.3 & 0.6 \end{bmatrix}$	$\begin{bmatrix} 0.6 & 0.2 & 0.2 \\ 0.1 & 0.7 & 0.2 \\ 0.2 & 0.3 & 0.5 \end{bmatrix}$

noise would be too large for a long time, which affects the stability of algorithm convergence. As a consequence, when we set the parameters of noise, we need to do a balance between the speed of convergence and the stability of the algorithm.

Fig. 6 shows the convergence of the proposed algorithm with varying user position states and content popularity states. The detailed simulation parameters are given in table III. When we have two user position states and two content popularity states, there are some fluctuations in

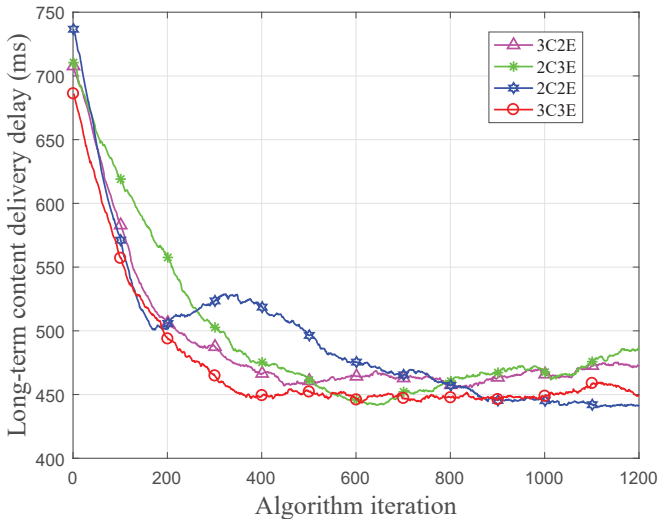

Fig. 6: Convergence of proposed algorithm comparison with varying user position states and content popularity states.

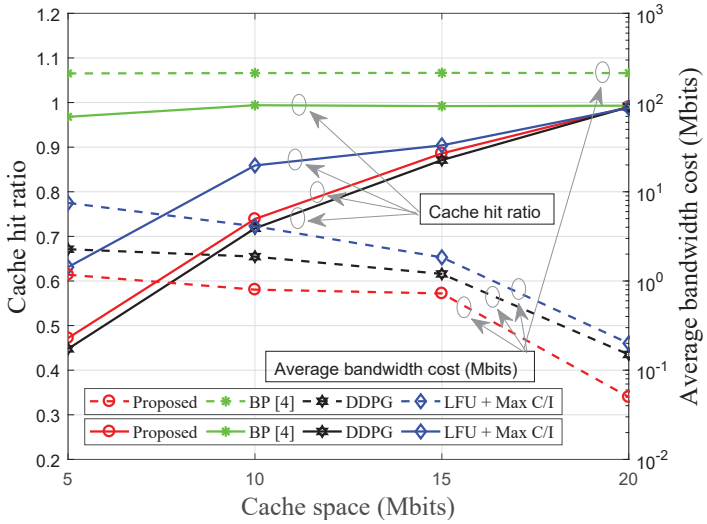

Fig. 7: Cache hit ratio and average bandwidth cost comparison with varying cache capacity.

the convergence process of the proposed algorithm, but the convergence point is lower. Besides, we verify the convergence of the proposed algorithm with different transition probabilities. And the simulation result shows the availability of the algorithm in different dynamic environments.

Then Fig. 7 reveals the cache hit ratio and average bandwidth cost with varying sizes of the cache capacity. As we can see, the solid lines denote cache hit ratio. The proposed algorithm achieves higher cache hit ratio with larger cache capacity. Compared with other algorithms,

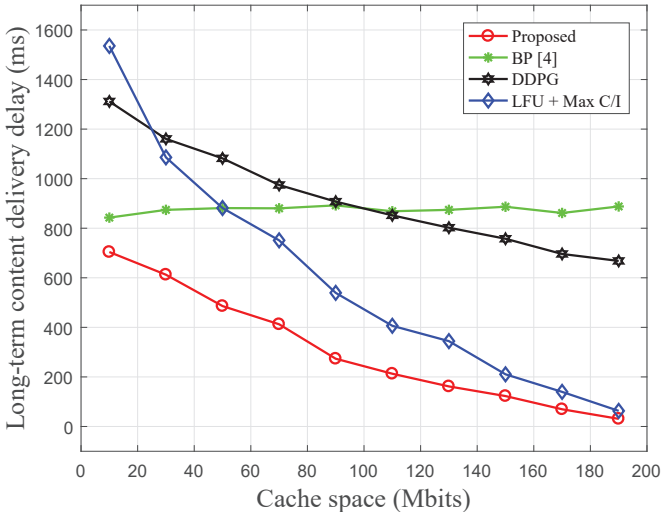

Fig. 8: Long-term content delivery delay comparison with varying cache capacity.

the proposed algorithm does not get the best cache hit ratio. However, the proposed algorithm considers both content transmission delay and content update delay. As we can observe from the dotted lines in Fig. 7, the proposed algorithm can greatly reduce the average bandwidth cost which is brought by the content updating. This is a good proof of **Remark 1**. Because BP algorithm obtains the caching placement and user association in each time slot, BP algorithm gets the highest cache hit ratio and the most average bandwidth cost. The average bandwidth cost of the proposed algorithm is lower than LFU and Max C/I algorithm, since the optimization objective is the long-term content delivery delay, which is composed of content transmission delay and content update delay. According to (8) (32), we find that the smaller the content update delay, the lower the average bandwidth cost. And the proposed algorithm can get a tradeoff between the content caching placement and content updating.

We evaluate the performance of the proposed algorithm with varying sizes of the cache capacity z. Fig. 8 shows that the long-term content delivery delay of the proposed algorithm reduces significantly compared to other algorithms. When $z = 50$ Mbytes, the performance of proposed algorithm is 0.45, 0.55, 0.45 times higher than BP algorithm, DDPG algorithm, LFU and Max C/I algorithm, respectively. The long-term content delivery delay decreases with the size of cache capacity increasing, except the delay of BP algorithm. Since BP algorithm updates the caching placement and user association in each time slot, the total content update delay in one time frame brings a large time overhead. Therefore, the delay obtained by BP algorithm is larger. The

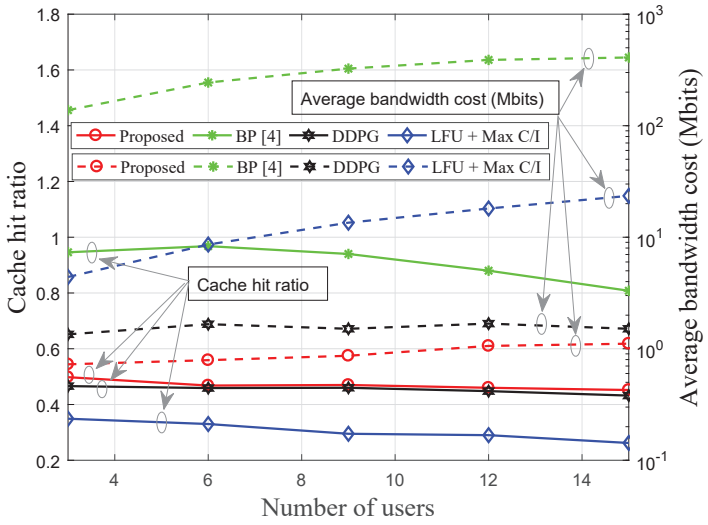

Fig. 9: Cache hit ratio and average bandwidth cost comparison with varying user numbers.

delay of DDPG algorithm reduces more slowly compared to the proposed algorithm, because DDPG updates the caching placement and user association in the same time scale which brings larger content transmission delay. And the delay of proposed algorithm, LFU and Max C/I gets better performance which verifies the superiority of the multiple time scales.

Next, we demonstrate the the effectiveness of the proposed algorithm with varying user number. As shown in 9, the proposed algorithm can get higher cache hit ratio and lower bandwidth cost compared with other benchmark algorithms. It is clear that the proposed algorithm can get the most steady performance. The result is consistent with that in Fig. 7. BP algorithm performs well in a single moment when we ignore the bandwidth cost. But in practical application scenarios, it is obvious that we cannot ignore the cost.

Fig. 10 shows the long-term content delivery delay comparison with varying number of users. In the simulation, we set the cache capacity as 50 Mbytes, and other parameters remain constant. Compared with other algorithms, the proposed algorithm can obtain a smaller delay and its performance is very stable. As we can see, the delay increases with the number of users increasing, but the impact on delay is small for the proposed algorithm. In DDPG algorithm, caching placement and user association are both updated in long time scale. More users will bring more uncertainty to user association. Therefore, the delay of DDPG algorithm changes greatly. The delay of BP, LFU and Max C/I increases slower. At the same time, the proposed algorithm shows significant performance.

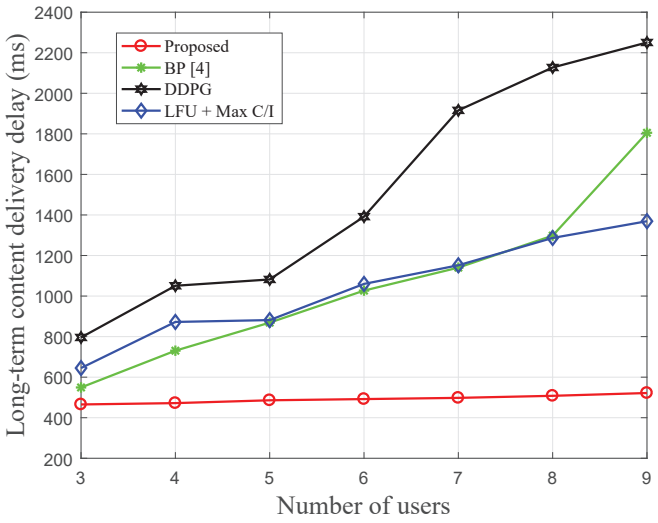

Fig. 10: Long-term content delivery delay comparison with varying user numbers.

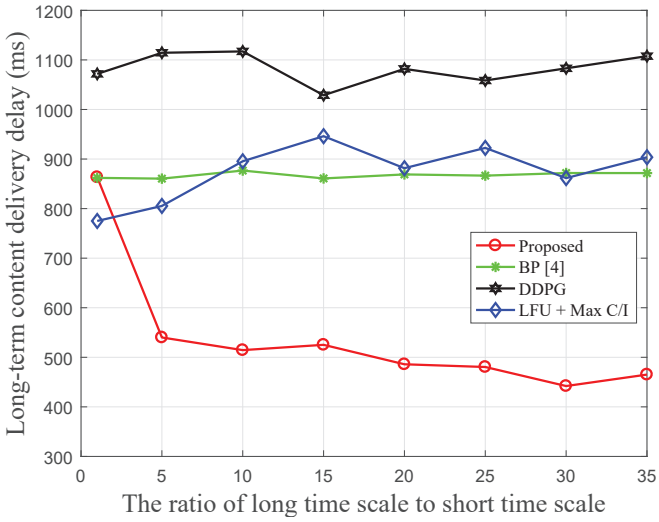

Fig. 11: Long-term content delivery delay comparison with varying time scale.

Fig. 11 demonstrates the long-term content delivery delay comparison with varying time scale. With the time scale increasing, the proposed algorithm get a smaller delay and a stable convergence point gradually. From Fig. 11 we can see, when $T = t$, each algorithm updates caching placement strategy and user association strategy in the same time-scale, and the proposed algorithm has no obvious advantages. However, the proposed algorithm performs better with time-scale increasing. The content update delay is significantly reduced in the proposed algorithm which applies multiple time-scale method, which verifies **Remark 2**. This provides a good proof

for the application of the multiple time-scale algorithm.

V. Conclusion

This paper has investigated cache-enabling cellular networks with time-varying user location and content popularity. To adapt to the dynamic environment, we have formulated a long-term joint optimization problem of caching placement and user association. We have proposed a joint optimization algorithm of caching placement and user association based on multiple time scales. Simulation results have demonstrated the convergence of the proposed algorithm and verified the feasibility and effectiveness of the proposed algorithm. Besides, from the simulation results, we can see that the multiple time-scale method can greatly improve the system performance.

References

[1] Y. Wang, C. Feng, and T. Zhang, "Deep reinforcement learning based caching placement and user association for dynamic cellular networks," in *2021 IEEE 32nd Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2021, pp. 1–6.

[2] 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, 2014.

[3] 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, 2016.

[4] J. Liu, B. Bai, J. Zhang, and K. B. Letaief, "Cache placement in Fog-RANs: From centralized to distributed algorithms," *IEEE Trans. Wireless Commun.*, vol. 16, no. 11, pp. 7039–7051, 2017.

[5] D. Liu and C. Yang, "Caching at base stations with heterogeneous user demands and spatial locality," *IEEE Trans. Commun.*, vol. 67, no. 2, pp. 1554–1569, 2019.

[6] J. Li, Y. Chen, Z. Lin, W. Chen, B. Vucetic, and L. Hanzo, "Distributed caching for data dissemination in the downlink of heterogeneous networks," *IEEE Trans. Commun.*, vol. 63, no. 10, pp. 3553–3568, 2015.

[7] J. Song, H. Song, and W. Choi, "Optimal content placement for wireless femto-caching network," *IEEE Trans. Wireless Commun.*, vol. 16, no. 7, pp. 4433–4444, 2017.

[8] L. E. Chatzieleftheriou, M. Karaliopoulos, and I. Koutsopoulos, "Jointly optimizing content caching and recommendations in small cell networks," *IEEE Mobile Commput.*, vol. 18, no. 1, pp. 125–138, 2019.

[9] X. Li, X. Wang, K. Li, Z. Han, and V. C. M. Leung, "Collaborative multi-tier caching in heterogeneous networks: Modeling, analysis, and design," *IEEE Trans. Wireless Commun.*, vol. 16, no. 10, pp. 6926–6939, 2017.

[10] S. Zhang, P. He, K. Suto, P. Yang, L. Zhao, and X. Shen, "Cooperative edge caching in user-centric clustered mobile networks," *IEEE Mobile Commput.*, vol. 17, no. 8, pp. 1791–1805, 2018.

[11] 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, 2018.

[12] 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, 2019.

[13] N. Zhao, X. Liu, Y. Chen, S. Zhang, Z. Li, B. Chen, and M. Alouini, "Caching D2D connections in small-cell networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 12, pp. 12 326–12 338, 2018.

[14] 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, 2018.

[15] A. M. Ibrahim, A. A. Zewail, and A. Yener, "Device-to-device coded-caching with distinct cache sizes," *IEEE Trans. Commun.*, vol. 68, no. 5, pp. 2748–2762, 2020.

[16] J. Ji, K. Zhu, D. Niyato, and R. Wang, "Probabilistic cache placement in UAV-assisted networks with D2D connections: Performance analysis and trajectory optimization," *IEEE Trans. Commun.*, vol. 68, no. 10, pp. 6331–6345, 2020.

[17] T. Zhang, X. Fang, Y. Liu, G. Y. Li, and W. Xu, "D2D-enabled mobile user edge caching: A multi-winner auction approach," *IEEE Trans. Veh. Technol.*, vol. 68, no. 12, pp. 12 314–12 328, 2019.

[18] J. Ji, K. Zhu, D. Niyato, and R. Wang, "Joint cache placement, flight trajectory, and transmission power optimization for multi-UAV assisted wireless networks," *IEEE Trans. Wireless Commun.*, vol. 19, no. 8, pp. 5389–5403, 2020.

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

[20] Z. Yang, C. Pan, Y. Pan, Y. Wu, W. Xu, M. Shikh-Bahaei, and M. Chen, "Cache placement in two-tier hetnets with limited storage capacity: Cache or buffer?" *IEEE Trans. Commun.*, vol. 66, no. 11, pp. 5415–5429, 2018.

[21] W. Teng, M. Sheng, K. Guo, and Z. Qiu, "Content placement and user association for delay minimization in small cell networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 10, pp. 10 201–10 215, 2019.

[22] Z. Chen, J. Lee, T. Q. S. Quek, and M. Kountouris, "Cooperative caching and transmission design in cluster-centric small cell networks," *IEEE Trans. Wireless Commun.*, vol. 16, no. 5, pp. 3401–3415, 2017.

[23] 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, 2016.

[24] H. Wu, H. Lu, F. Wu, and C. W. Chen, "Energy and delay optimization for cache-enabled dense small cell networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 7, pp. 7663–7678, 2020.

[25] F. Jiang, Z. Yuan, C. Sun, and J. Wang, "Deep Q-learning-based content caching with update strategy for fog radio access networks," *IEEE Access.*, vol. 7, pp. 97 505–97 514, 2019.

[26] T. Zhang, X. Fang, Z. Wang, Y. Liu, and A. Nallanathan, "Stochastic game based cooperative alternating Q-learning caching in dynamic D2D networks," *IEEE Trans. Veh. Technol.*, pp. 1–1, 2021.

[27] T. Zhang, Z. Wang, Y. Liu, W. Xu, and A. Nallanathan, "Caching placement and resource allocation for cache-enabling UAV NOMA networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 11, pp. 12 897–12 911, 2020.

[28] Y. Jiang, M. Ma, M. Bennis, F. Zheng, and X. You, "User preference learning-based edge caching for fog radio access network," *IEEE Trans. Commun.*, vol. 67, no. 2, pp. 1268–1283, 2019.

[29] X. Chen, W. Ni, T. Chen, I. B. Collings, X. Wang, R. P. Liu, and G. B. Giannakis, "Multi-timescale online optimization of network function virtualization for service chaining," *IEEE Mobile Commput.*, vol. 18, no. 12, pp. 2899–2912, 2019.

[30] W. Jing, X. Wen, Z. Lu, and H. Zhang, "User-centric delay-aware joint caching and user association optimization in cache-enabled wireless networks," *IEEE Access.*, vol. 7, pp. 74 961–74 972, 2019.

[31] J. Tadrous, A. Eryilmaz, and H. El Gamal, "Proactive content download and user demand shaping for data networks," *IEEE Netw.*, vol. 23, no. 6, pp. 1917–1930, 2015.

[32] Y. Guo, Q. Yang, F. R. Yu, and V. C. M. Leung, "Cache-enabled adaptive video streaming over vehicular networks: A dynamic approach," *IEEE Trans. Veh. Technol.*, vol. 67, no. 6, pp. 5445–5459, 2018.

[33] R. Sun, Y. Wang, N. Cheng, L. Lyu, S. Zhang, H. Zhou, and X. Shen, "QoE-driven transmission-aware cache placement and cooperative beamforming design in Cloud-RANs," *IEEE Trans. Veh. Technol.*, vol. 69, no. 1, pp. 636–650, 2020.

[34] J. Kwak, L. B. Le, and X. Wang, "Two time-scale content caching and user association in 5G heterogeneous networks," in *IEEE Proc. of Global Commun. Conf. (GLOBECOM)*, 2017, pp. 1–6.

[35] J. Kwak, L. B. Le, H. Kim, and X. Wang, "Two time-scale edge caching and BS association for power-delay tradeoff in multi-cell networks," *IEEE Trans. Commun.*, vol. 67, no. 8, pp. 5506–5519, 2019.

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

[37] Y. Yu, J. Tang, J. Huang, X. Zhang, D. K. C. So, and K.-K. Wong, "Multi-objective optimization for UAV-assisted wireless powered IoT networks based on extended DDPG algorithm," *IEEE Trans. Commun.*, vol. 69, no. 9, pp. 6361–6374, 2021.

[38] 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, 2019.

[39] G. Zhang, T. Q. S. Quek, M. Kountouris, A. Huang, and H. Shan, "Fundamentals of heterogeneous backhaul design-analysis and optimization," *IEEE Trans. Commun.*, vol. 64, no. 2, pp. 876–889, 2016.

[40] J. Chuan, L. Wang, and J. Wu, "Belief propagation based distributed content delivery scheme in caching-enabled D2D networks," in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, 2019, pp. 1–5.

[41] F. R. Kschischang, B. J. Frey, and H. . Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 498–519, 2001.

[42] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky, "Map estimation via agreement on trees: message-passing and linear programming," *IEEE Trans. Inf. Theory*, vol. 51, no. 11, pp. 3697–3717, 2005.

[43] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *Comput Sci*, vol. 8, no. 6, pp. 1–14, 2015.