Cognitive-Caching: Cognitive Wireless Mobile Caching by Learning Fine-Grained Caching-Aware Indicators

Yixue Hao, Min Chen, Donggang Cao, Wenlai Zhao, Ivan Petrov, Vitaly Antonenko, and Ruslan Smeliansky

Abstract

Caching content on mobile devices reduces not only the transmission of backhaul links but also the latency of content acquisition. However, in the existing caching schemes, the joint effect of the caching-aware indicator of users and content dimensions on the caching strategy is not considered. Thus, how to learn fine-grained caching-aware indicators and design caching schemes are still challenging issues. In order to solve these problems, in this article, we first propose the cognitive caching architecture and give the fine-grained caching-aware indicators metric. Then we design a cognitive caching scheme that includes what to do in cognitive caching and how to do cognitive caching. Finally, experimental results show that the proposed cognitive caching scheme is superior to other caching schemes in terms of learning regret and caching cost.

Introduction

With the rapid growth of mobile devices, wireless traffic data has also been increasing rapidly. According to Cisco, mobile data traffic will reach 49 EB per month by 2021 [1]. This will bring a great burden to the radio network. To meet the requirements for high-speed transmission of a large amount of mobile data, the base stations have to be ultra-densely deployed in the fifth generation (5G) network [2]. By adopting such a dense deployment, the network capacity can be significantly increased [3]. However, this densified deployment poses a challenge to the backhaul link.

In order to address this challenge, researchers have proposed edge caching. That is, during off-peak periods of the network, the popular content can be proactively cached on edge clouds or mobile devices. Caching strategy based on the edge cloud has been widely studied [4, 5]. In this article, we focus on mobile device caching, which refers to caching content on mobile devices. Neighboring mobile devices can obtain requested content through device-to-device (D2D) communication [6], which not only alleviates the pressure on the backhaul link but also reduces the transmission delay. Therefore, mobile caching has attracted much attention.

There are many studies on wireless mobile caching that separately analyze the effects of user mobility, mobile devices' storage, content popularity, and so on. Considering that the mobility of users may increase the probability of D2D encounters between users, Chen *et al*. [7] designed a green and mobility-aware caching scheme by jointly optimizing user mobility and transmission power, which can not only improve the content cache hit ratio, but also reduce energy consumption. Based on the user mobility model, Wang *et al*. [8] improved the data offloading rate by establishing a mobility-aware caching strategy. Furthermore, Muller *et al*. [9] established a content caching strategy with the cache hit ratio as a target, taking into account user context. Considering the content popularity, Yang *et al*. [10] aimed to improve the cache hit ratio by predicting content popularity.

However, all these caching strategies consider the impact of a single cache metric on the caching strategy. In this article, we call the factors that affect caching *caching-aware indicators* and divide them into two categories: user caching-aware indicators and content caching-aware indicators. Furthermore, the cache indicators of users and content interact mutually. Thus, in the actual caching scheme design, we need to take into account the user caching-aware indicators (e.g., heterogeneity of storage and computing resources of mobile devices) and content caching-aware indicators (e.g., content size and popularity). We name these composite metrics *fine-grained caching-aware indicators* and consider their impact on caching schemes.

When considering fine-grained caching-aware indicators, we need not only to give its metric, but also to consider the impact of its randomness on cache decisions. For example, in mobile caching, the D2D communication links and content popularity among users change randomly over time, and these changes are uncertain. Moreover, the randomness of the fine-grained caching-aware indicators make traditional caching schemes inapplicable. How to learn fine-grained caching-aware indicators online and make cache decisions will be a challenge for mobile caching. Fortunately, multi-arm bandit (MAB) learning as a kind of reinforcement learning has been widely applied in sequential decision making [11]. It can make optimal decisions by learning what is unknown (i.e., exploration) and

Yixue Hao and Min Chen are with the School of Computer Science and Technology, Huazhong University of Science and Technology; Donggang Cao (corresponding author) is with Peking University; Wenlai Zhao is with the Beijing National Research Center for Information Science and Technology, Tsinghua University; Ivan Petrov, Vitaly Antonenko, and Ruslan Smeliansky are with Lomonosov Moscow State University.

Digital Object Identifier:
10.1109/MWC.001.1900273

1536-1284/20/$25.00 © 2020 IEEE

using what is learned (i.e., exploitation). The traditional MAB algorithm selects the best arm from a set of candidate arms by balancing exploration and exploration. However, cognitive caching needs to learn the fine-grained caching-aware indicators to select multiple candidate mobile devices, which is more complex than the traditional MAB problem.

Furthermore, the existing caching strategies are mainly focused on improving network performance [12]. In order to make mobile caching more practical, the economic problems caused by resource consumption in mobile caching need to be addressed [13]. These problems can be divided into two parts: storage cost problems and transmission cost problems. Storage cost problems refer to the content cache on a mobile device, which occupies the storage capacity of the mobile device, so the network operator needs to pay a certain fee for using that storage resource. On the other hand, transmission cost problems refer to the consumption of transmission resources of a mobile device when it transmits the cached content to the requesting user, and in this case, the network operator also needs to pay a certain fee. As for wireless mobile caching, most studies only consider a single cost when designing the caching strategy.

To address these problems, in this article, we propose a cognitive caching architecture and scheme to minimize the caching cost. Specifically, we first introduce the cognitive caching architecture and model the fine-grained caching-aware indicator. Then, based on the MAB theory, we propose a cognitive caching scheme by learning fine-grained caching-aware indicators online. Finally, we evaluate the cognitive caching by a simulation experiment. The experiment results show that the cognitive caching scheme has lower caching cost and learning regret than the traditional caching schemes.

In summary, the contributions of this article are as follows:

- To the best of our knowledge, the concept of caching-aware indicators is introduced for the first time. Furthermore, we give the cognitive caching architecture and the fine-grained caching-aware indicators metric.
- Based on the MAB theory, we design a cognitive caching scheme that includes what to do in cognitive caching and how to do cognitive caching by learning fine-grained caching-aware indicators (e.g., user location, mobile device status) online.
- Extensive experiments demonstrate that the proposed cognitive caching strategy is superior to other caching strategies in terms of learning regret and caching cost.

The remainder of this article is organized as follows. In the following section we give the cognitive caching architecture. Next, the cognitive caching scheme is introduced. Then we present the experimental setup and results for cognitive caching. Finally, we conclude the article.

Cognitive Caching Architecture

In this section, we present the cognitive caching architecture. To be specific, we first introduce the caching-aware indicator, going from coarse- to fine-grained. Then we propose the cognitive caching architecture. Finally, we give the fine-grained caching-aware indicators metric.

Coarse-grained	Consider only one or more of the user caching-aware indicators (e.g., time, location, age, occupation, device, network condition) or content caching-aware indicators (e.g., content size, popularity, type)
Fine-grained	Consider both user and content caching-aware indicators closely

TABLE 1. The comparison of coarse-grained and fine-grained caching-aware indicators.

Caching-Aware Indicator: From Coarse-to-Fine

For mobile device caching, we give the concept of caching-aware indicators, which include user caching-aware indicators and content caching-aware indicators. The user caching-aware indicators include the user's location, time, mobile device status, and so on. The content caching-aware indicators contain content size, type, popularity, and so on. When considering caching-aware indicators, existing mobile caching schemes consider only one factor in a coarse-grained manner. For example, considering the user's mobility, a mobility-aware caching strategy is proposed [7]. However, this caching strategy does not take into account the change of a user's request for content caused by the user's mobility, thus affecting the design of caching strategy, especially when users request location-based content. Therefore, we call these caching strategies *coarse-grained caching-aware indicators-based caching strategies*.

From the above discussion, we can see that the existing caching scheme has limitations in user granularity and content granularity, and we need to consider fine-grained caching-aware indicators. Furthermore, existing caching strategies do not take both user and content granularity into account. Thus, we consider not only the fine-grained cache metrics, but also user and content granularity. Table 1 gives the comparison of coarse-grained and fine-grained caching-aware indicators.

Cognitive Caching Architecture

We propose the cognitive caching architecture, as shown in Fig. 1. The architecture includes heterogeneous mobile devices and an edge cloud. Mobile devices can be either content requesters or content servers, and each mobile device has a certain amount of storage capacity, which can cache the content. Furthermore, mobile devices can communicate directly via out-of-band D2D link (e.g., Bluetooth), or through in-band D2D links (e.g., base-station-assisted D2D architecture). Since the out-of-band D2D link is unstable and difficult to control, in this article, we consider in-band D2D links. The in-band D2D link assumes that the base station knows which devices can communicate with each other in D2D communication. However, the base station does not have prior knowledge of user and content caching-aware indicators. Furthermore, edge cloud has a certain storage and computing capacity to compute and store user and content fine-grained caching-aware indicators.

To achieve cognitive caching, we need to deploy a caching agent on edge cloud that includes a learning module and a selection module, as shown in Fig. 1. The learning module learns user and content caching-aware indicators. The selection module makes cache decisions based on the learning module, that is, selecting which

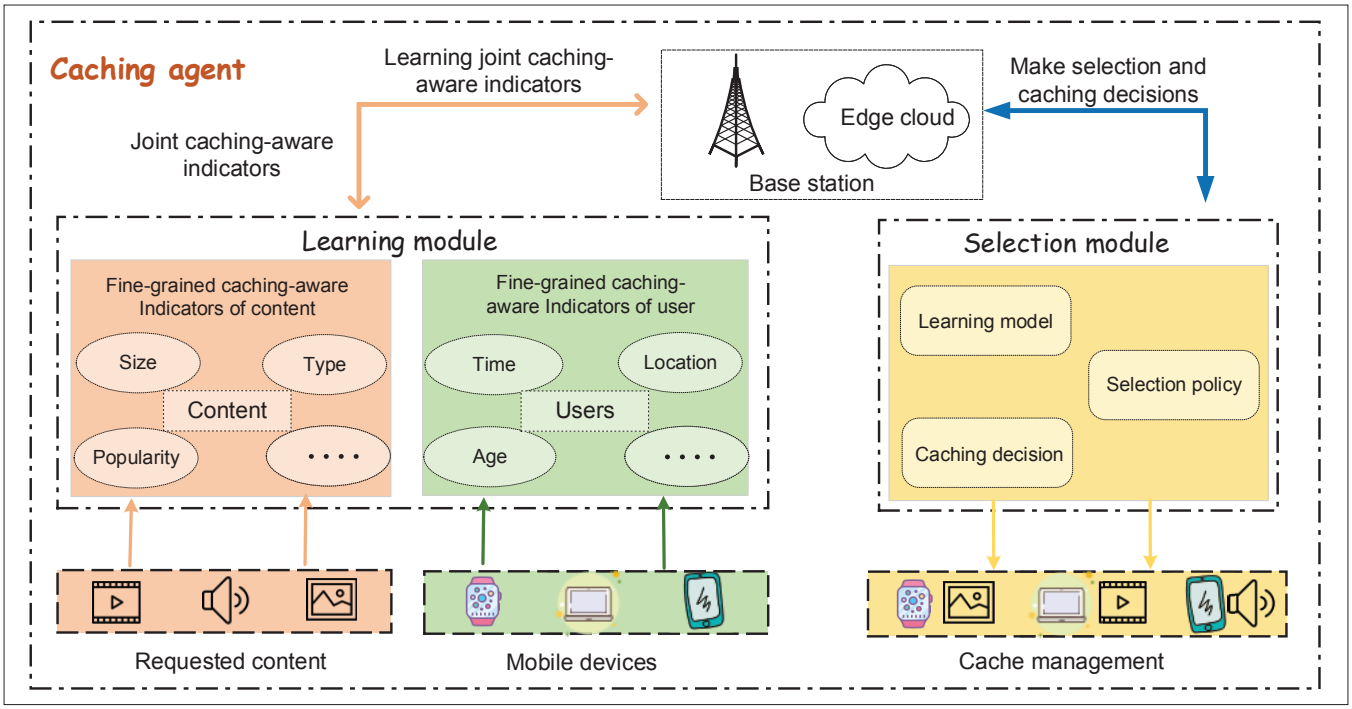

FIGURE 1. Illustration of cognitive caching architecture.

devices are connected to the request content device for caching. The specific process is as follows. When the caching agent receives the user content request, the caching agent first observes which devices are connected to the request content device through D2D communication. Then, by observing connected devices' and users' caching-aware indicators, edge cloud formulates the caching scheme through learning users and content caching-aware indicators in real time. Thus, through cognitive caching, we implement learning while caching.

Fine-Grained Caching-Aware Indicators Metric

Through the discussion above, for content caching, it is important to note that both the user caching-aware indicator and the content caching-aware indicator affect the user cache scheme. For instance, when a user is in a static state and a network is in good condition, users tend to cache high-quality video content, while when user mobility is high and network condition is poor, users tend to cache low-quality video content. Therefore, in this article, we consider the fine-grained caching-aware indicator of both user and content, and decide how to choose the caching mobile devices to minimize the caching cost while ensuring high service quality.

Specifically, the user's fine-grained caching-aware indicators include the time, location, and device status (e.g., device storage capacity, battery status), which have multiple dimensions. Considering the disunity of these indicator units, to unify these indicators, we use min-max normalization to normalize the user caching-aware indicator to the range [0, 1]. For example, we present the user's age and occupation normalization. For age, we can divide 0 to 100 years old into 10 parts on average and normalize them to the range of [0, 1]. For example, age 23 corresponds to 0.23 and falls within the range [0.2, 0.3]. For occupations, if there are 20 occupations in total, we randomly sort the occupations and divide the interval of [0, 1] into 20 parts on average. Thus, the second-ranked occupation (e.g., student) falls in the range of [0.1, 0.2].

Furthermore, similar to the user fine-grained caching-aware indicators standardization, we normalize multidimensional content caching-aware indicators (e.g., the size of the content, content popularity) to the range [0,1]. Furthermore, we join the caching-aware indicators of user and content. To represent the fine-grained caching-aware indicators of users and content, we set up a hypercube with a value of [0,1]. For example, if a user is 23 years old, a student, and caches 21Mb of content, the user falls into a [0.2, 0.3] × [0.1, 0.2] × [0.2, 0.3] hypercube. Thus, we give the metric of fine-grained caching-aware indicators.

We give a simple example of how to use fine-grained indicators of user and content. Specifically, at time slot t, the caching agent receives the request for video from user Alice, and observes that there are three users, Bob, Cindy, and Eva, within Alice's D2D communication range. By learning the fine-grained caching-aware indicators of users and video, the caching agent knows that Eva can cache this content better. Then the caching agent selects Eva to cache the video, and Eva transmits the cached video to Alice.

Cognitive Caching Scheme

In this section, we propose the cognitive caching scheme as shown in Figs. 2 and 3, which includes what to do in cognitive caching and how to do cognitive caching.

What to do Cognitive Caching

We illustrate what to do in cognitive caching in Fig. 2. As shown in Fig. 2, there are four candidate content servers in the D2D communication range of a content requester. By learning the fine-

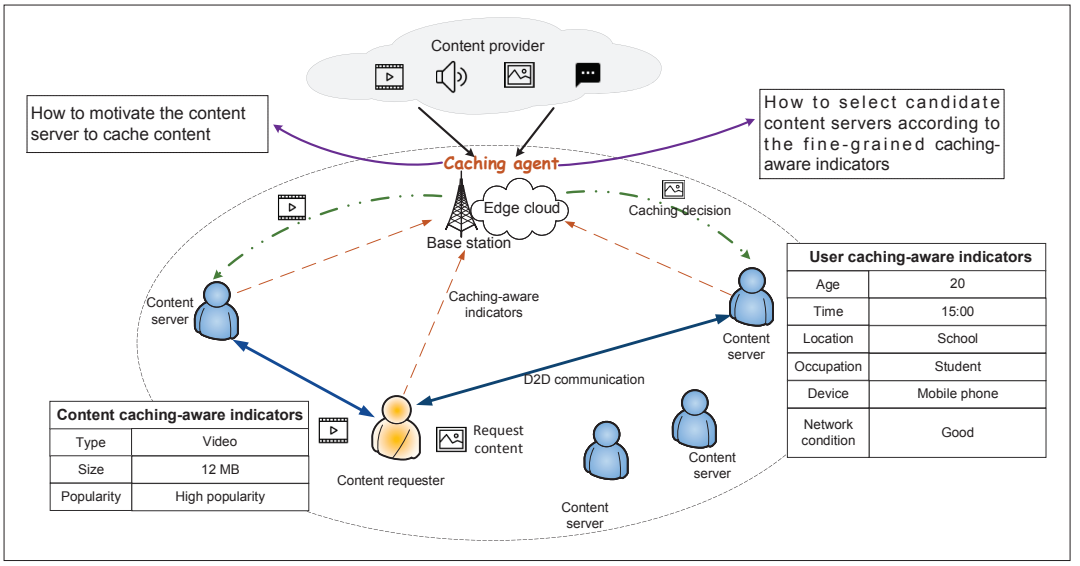

FIGURE 2. Illustration of what to do in cognitive caching in a device to device network.

grained caching-aware metrics of users and content, the caching agent selects two of them for content caching. Thus, we need to address two key challenges in our cognitive caching design:

1. How to select candidate content servers according to the fine-grained caching-aware indicators. Due to mobile device mobility, D2D connections between mobile devices are dynamic, and the fine-grained caching indicators are random, which make it difficult to predict.
2. How to motivate the content server to cache content. This is due to the fact that caching and transmitting content occupies the storage and communication resources of the content server.

To address these challenge, we assume that the system is running in a discrete period of time, and the mobility does not change much; thus, the users will not leave base station coverage in a given time slot. Furthermore, we consider the caching-aware indicators of the requested content and user, and decide how to choose the content caching device so that the requesting user can obtain the requested content through D2D communication. Specifically, when the content requester acquires the content through D2D communication, there is a user that caches the content, and this user transmits the content to the requester. Therefore, we can obtain the transmission delay. Furthermore, based on [12], we assume that at time slot t, the base station knows that users can achieve D2D communication, but does not know the transmission rate. This is because the transmission rate is related to random caching-aware indicators.

Now, we give the probability that the user requester will get the content through D2D communication. In D2D networks, when a content requester requests a content, it first observes whether it caches that content itself. If the content is already cached, it can be obtained directly. If there is no such content in the cache, the content requester observes whether the requested content can be obtained from a user encountered through D2D communication. If the content cannot be obtained through D2D communication within the predefined deadline, the content requester acquires the requested content from the edge cloud. Therefore, when the transmission delay is less than the given deadline, the requested content can be obtained through D2D communication. Otherwise, the requested content cannot be obtained through D2D communication.

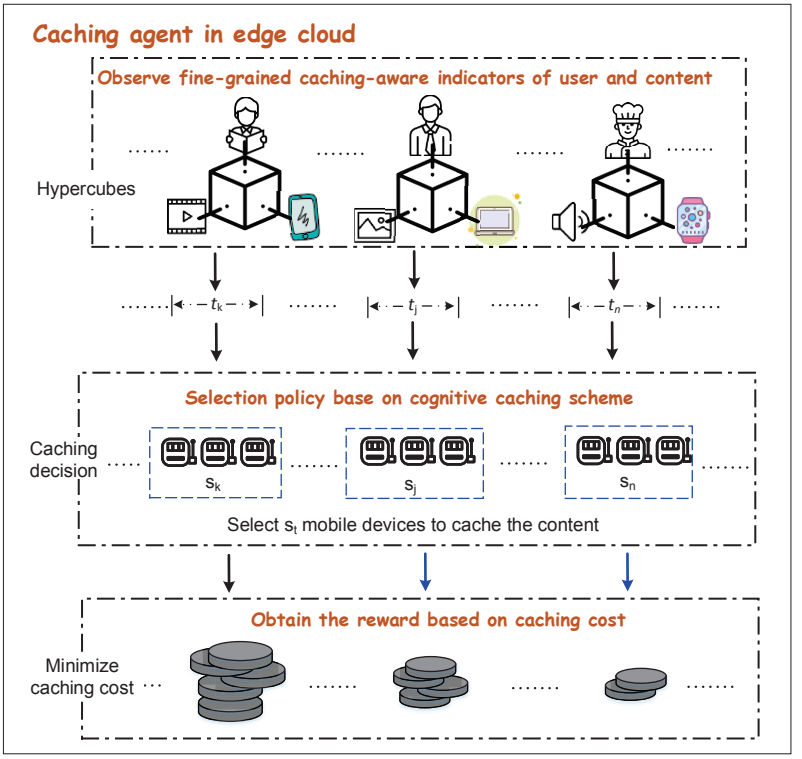

FIGURE 3. Illustration of how to do cognitive caching.

Furthermore, in a D2D caching network, caching content on a mobile device can reduce the content transmission cost. However, the content cached on a mobile device occupies its storage resource. Therefore, network operators have to pay a certain fee to motivate mobile users to participate in mobile device caching. Specifically, we assume that a content requester can communicate with b_t users through D2D communication in time

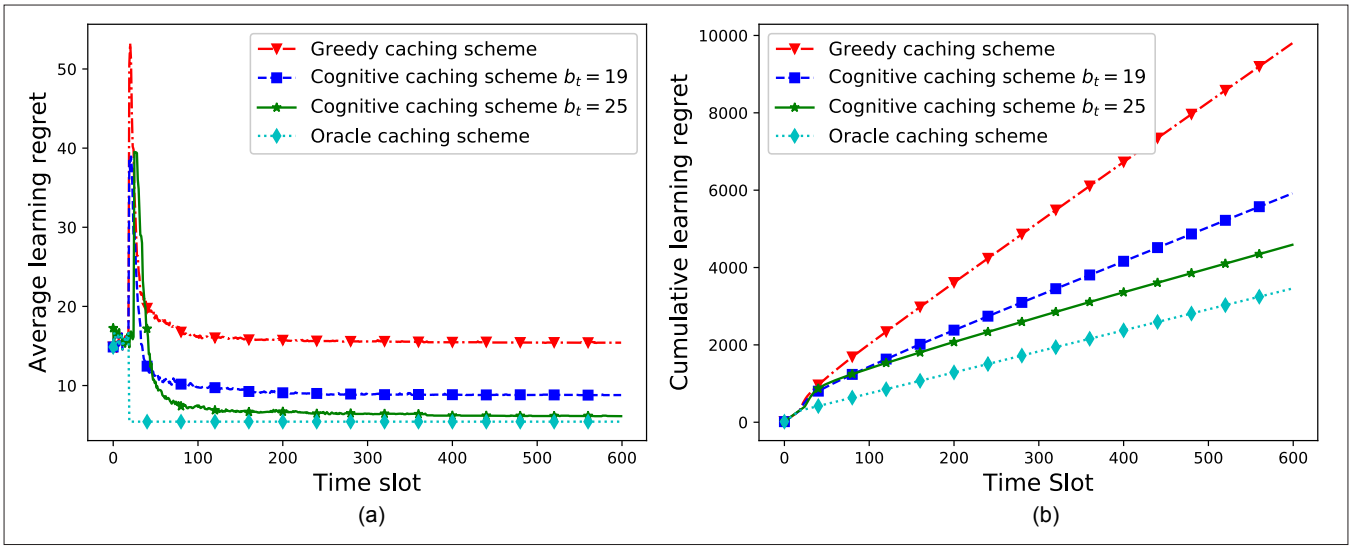

FIGURE 4. Learning regret under greedy caching scheme, cognitive caching scheme b_t = 19, cognitive caching scheme b_t = 25, and oracle caching scheme: a) average learning regret under different caching schemes; b) cumulative learning regret under different caching schemes.

slot t. As for the caching cost, if we already know the price of the storage unit of the content server, when b_t users are selected to cache, we can obtain the total storage cost.

In addition to the storage cost, we also need to consider the content transmission cost. Specifically, the transmission cost includes the cost of obtaining content from the base station and from a mobile device, respectively. Furthermore, the communication cost of a content transmission from the base station is greater than that through mobile devices. Thus, we can obtain the caching cost, which includes storage cost and transmission cost. Specifically, when the requested content can be obtained through D2D communication, the caching cost contains storage cost and D2D transmission cost. However, when the requested content cannot be obtained through D2D communication, the caching cost includes storage cost and base station transmission cost.

According to the above analysis, our goal is to minimize the caching cost by selecting the appropriate candidate content servers based on the user and content fine-grained caching-aware indicators. Based on the above model and the at-least-one probabilistic rule, we can obtain the expectation of the caching cost. Thus, we can give the cognitive caching optimization problem. Specifically, in time slot t, the cache agent knows the total number of content servers that can communicate with the content requester through D2D communication, so the constraint of the optimization problem is that the decision cache device is less than the total number of content servers set. The goal of optimization is to minimize the expectation of the caching cost. For the cognitive caching problem, the caching agent needs to deal with continuously random caching-aware indicators.

How to Do Cognitive Caching

As for the cognitive caching problem, our goal is to reduce the caching cost of a network operator as much as possible on the basis of ensuring the quality of user service. If the network operator (e.g., edge cloud) has prior knowledge about the caching-aware indicators, the cognitive caching problem can be transformed into a problem of solving the smallest subset, which can be solved by using a greedy algorithm. However, the caching-aware indicators of users and content change over time, and the D2D connections between users are dynamic. Thus, the network operator cannot have a priori knowledge about the caching-aware indicators. To solve it, we discuss how to do cognitive caching, that is, deploy caching strategy by learning the fine-grained caching-aware indicators online.

The cognitive caching problem can be mapped to a MAB problem. The traditional MAB problem has a fixed number of arms and an unknown reward [14]. In the MAB problem, the user is faced with multiple arms. In every time slot, the user can select an arm to observe the reward of an unexplored arm (i.e., exploration) or choose the best reward arm based on the arms that have been selected (i.e., exploitation). Then, through balanced exploration and exploitation, an appropriate optimal arm is selected. In cognitive caching, we aim to select the content server with the minimum caching cost under the influence of the fine-grained caching -aware indicator. Thus, the number of content servers is considered to be equivalent to arms in the MAB problem. The caching cost can be equal to the reward, but it should be noted that the goal is to minimize the reward.

The upper confidence bound (UCB) is a classical solution to the MAB problem. Thus, in this article, we use the UCB algorithm to address the cognitive caching problem. However, the cognitive caching problem is different from the traditional UCB solution, because the caching agent needs to consider the fine-grained caching-aware indicators and select multiple content servers (i.e., s_t) at each time slot. Thus, in order to solve this problem, we improve and extend the existing UCB algorithm and propose a cognitive caching algorithm that can make the cache decisions according to the fine-grained caching-aware indicator. The illustration of cognitive caching is shown in Fig. 3. First, the caching agent observes the fine-grained

caching-aware indicators of user and content, and divides them into hypercubes. Then the caching agent decides which content to select and cache through the cognitive algorithm. Finally, the caching agent obtains the reward based on the caching cost.

Now, we describe how to do cognitive caching in detail. We first give the number of times of selecting each content server at time slot t. When the content server is caching contents, we can obtain the caching cost that needs to be paid to the content server. Then we normalize the fine-grained caching-aware indicators in time slot t to the range [0, 1] and apply the fine-grained caching-aware indicators to the caching decision. Finally, the caching agent selects a content server based on the number of times the content server is explored, fine-grained caching-aware indicators, and caching cost.

For the number of times the content server is explored, based on the traditional UCB, if there is an unexplored content server at time slot t, the caching agent will select this content server, make it cache the contents at time slot t, and obtain the caching cost. Moreover, the caching agent will update the number of times the content server has been selected. It should be noted that for the cognitive caching problem, the caching agent needs to consider the fine-grained caching-aware indicators and select s_t content servers at time slot t. We present content server selection based on the fine-grained caching-aware indicator in the next.

For the fine-grained caching-aware indicators and caching cost, because the caching cost is nonlinear, the caching decision of content servers is related to the caching cost of all the candidate content servers and the caching-aware indicators of each candidate content server. Thus, in the process of selecting the cache servers, we use the average caching cost. The learning process of the cognitive caching is as follows: Based on the observed fine-grained metrics of users and content, the caching agent exploits the content servers with lower average caching cost, and explores candidate content servers that have not yet been fully explored. Thus, the cognitive caching selection balances exploration and exploitation.

Furthermore, we use average learning regret and cumulative regret to evaluate the cognitive caching algorithm. The average learning regret refers to the difference between the caching cost of the selected content servers and the minimum caching cost achieved by the optimal content servers. The cumulative regret refers to the cumulative regrets over time. We choose s_t to make the learning regret smallest.

Performance Evaluation

In this section, we present the cognitive caching performance evaluation. In the experiment, we assume that mobile devices are randomly distributed within the coverage range of the base station. We divide the time horizon into 600 time slots. For the fine-grained caching-aware indicator, we set the dimension of the fine-grained caching-aware indicator space to 30. Furthermore, to assess the impact of candidate content servers on cognitive caching, we present two scenarios. In the first scenario, we assume that there are 19 candidate content servers (i.e., $b_t = 19$) around a content requester, and in the second scenario, we set the number of candidate content servers to be 25 (i.e., $b_t = 25$). Moreover, the number of content servers selected at each time slot t is $S_t = 3$. We use different algorithms to obtain the cumulative learning regret, average learning regret, and caching cost to verify the proposed cognitive caching algorithm.

For requested content, we set the deadline for obtaining the content via D2D communication to 150 s. According to [13, 15], we set the caching cost per bit to 35. The cost of transmission content through D2D and edge cloud is set to 1 and 50, respectively. Furthermore, to evaluate the cognitive caching strategy, we compared cognitive caching with the following mobile caching schemes.

Oracle Caching Scheme: This scheme assumes that the caching agent already knows the content servers, content fine-grained caching-aware indicator, and transmission rate.

Greedy Caching Scheme: In this scheme, we use the UCB algorithm to greedily select s_t content servers, that is, s_t content servers with the least caching cost are directly selected. For the oracle and greedy schemes, we set the number of candidate content servers to 19.

Performance Analysis

In this experiment, we first evaluate the average learning regret for different mobile caching schemes. As shown in Fig. 4a, the greedy caching scheme, oracle caching scheme, and cognitive caching scheme all tend to be stable at $t = 150$ s, and the cognitive caching scheme tends to be stable faster than the greedy caching scheme. From Fig. 4b, it can be concluded that the cumulative learning regret of each caching scheme increases linearly, so the average learning regret of these schemes are convergent. Meanwhile, in Fig. 4b, it can be seen that the growth rate of cumulative learning regret of the cognitive caching scheme is lower than that of the greedy caching scheme.

Furthermore, in Fig. 4, it can be seen that the learning regret of the oracle caching scheme is the lowest. This is because, in the oracle caching scheme, it is assumed that the base station already knows the fine-grained caching-aware indicators, so the content is optimally cached. The greedy caching scheme uses UCB to greedily select content servers from the candidate content server set. Therefore, the average learning regret and cumulative learning regret of this scheme are the worst. In addition, in Fig. 4, it can be seen that the cognitive caching scheme performs significantly better than the greedy caching scheme. This can be explained by the fact that cognitive caching can learn the fine-grained caching-aware indicators and then perform reasonable content caching.

In Fig. 5, it can be seen that the average caching cost of the cognitive caching scheme is lower than those of the greedy caching scheme after $t = 100$ s. Furthermore, from Figs. 4 and 5, we can see that in the cognitive caching strategy, the average learning regret, cumulative learning regret, and average caching cost decrease with the number of candidate content servers increasing. This is because as the number of candidate content servers increases, the cognitive caching scheme has more choices to explore.

We use learning regret and cumulative regret to evaluate the cognitive caching algorithm. The learning regret refers to the difference between the caching cost of the selected content servers and the minimum caching cost achieved by the optimal content servers. The cumulative regret refers to the cumulative regrets over time. We choose s_t to make the learning regret smallest.

Simulation results show that compared to the other caching schemes, the proposed cognitive caching scheme has lower learning regret and caching cost. In future work, we will consider the design of cognitive caching strategy in specific scenarios, such as virtual reality scenarios. Furthermore, we will consider the cognitive caching strategy in hierarchical network architecture.

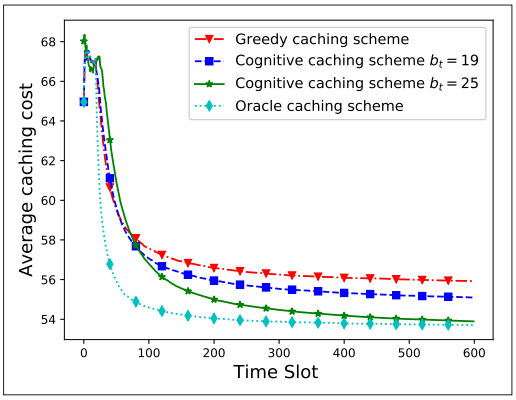

FIGURE 5. Average caching cost of the greedy caching scheme, cognitive caching scheme b_t = 19, cognitive caching scheme b_t = 25, and oracle caching scheme.

Conclusion

In this article, we propose a cognitive caching architecture and scheme through the study of fine-grained caching-aware indicators, which can optimize caching cost. First, we propose the cognitive caching architecture and a fine-grained caching-aware indicator metric. Then, based on the MAB theory, we design an online cognitive caching scheme that includes what to do in cognitive caching and how to do cognitive caching. Simulation results show that compared to the other caching schemes, the proposed cognitive caching scheme has lower learning regret and caching cost. In future work, we will consider the design of cognitive caching strategy in specific scenarios, such as virtual reality scenarios. Furthermore, we will consider the cognitive caching strategy in hierarchical network architecture.

Acknowledgments

This work is supported by the National Key R&D Program of China (2017YFE0123600), and the Russian Ministry of Science and Higher Education, grant #05.613.21.0088, unique ID RFME-FI61318X0088. Dr. Yixue Hao's work was supported by the National Natural Science Foundation of China under Grant 61802138.

References

[1] Cisco Visual Networking Index: Forecast and Trends, 2017–2022 White Paper.
[2] T. Chen *et al.*, "Softmobile: Control Evolution for Future Heterogeneous Mobile Networks," *IEEE Wireless Commun.*, vol. 21, no. 6, Dec. 2014, pp. 70–78.
[3] M. Chen *et al.*, "Opportunistic Task Scheduling Over Co-Located Clouds in Mobile Environment," *IEEE Trans. Services Computing*, vol. 11, no. 3, 2018, pp. 549–61.
[4] G. S. Paschos *et al.*, "The Role of Caching in Future Communication Systems and Networks," *IEEE JSAC*, vol. 36, no. 6, 2018, pp. 1111–25.
[5] T. X. Tran *et al.*, "Cooperative Hierarchical Caching and Request Scheduling in a Cloud Radio Access Network," *IEEE Trans. Mobile Computing*, vol. 17, no. 12, 2018, pp. 272–43.
[6] Z. Zhou *et al.*, "Energy-Efficient Stable Matching for Resource Allocation in Energy Harvesting Based Device-to-Device Communications," *IEEE Access*, vol. 5, 2017, pp. 15 184–96.
[7] M. Chen *et al.*, "Green and Mobility-Aware Caching in 5G Networks," *IEEE Trans. Wireless Commun.*, vol. 16, no. 12, 2017, pp. 834–61.
[8] R. Wang *et al.*, "Exploiting Mobility in Cache-Assisted D2D Networks: Performance Analysis and Optimization," *IEEE Trans. Wireless Commun.*, vol. 17, no. 8, 2018, pp. 5592–5605.
[9] S. Müller *et al.*, "Context-Aware Proactive Content Caching with Service Differentiation in Wireless Networks," *IEEE Trans. Wireless Commun.*, vol. 16, no. 2, 2016, pp. 1024–36.
[10] P. Yang *et al.*, "Content Popularity Prediction Towards Location-Aware Mobile Edge Caching," *IEEE Trans. Multimedia*, vol. 21, no. 4, 2019, pp. 915–29.
[11] W. Chen *et al.*, "Combinatorial Multi-Armed Bandit and Its Extension to Probabilistically Triggered Arms," *J. Machine Learning Research*, vol. 17, no. 1, 2016, pp. 1746–78.
[12] D. Liu *et al.*, "Caching at the Wireless Edge: Design Aspects, Challenges, and Future Directions," *IEEE Commun. Mag.*, vol. 54, no. 9, Sept. 2016, pp. 22–28.
[13] J. Krolikowski, A. Giovanidis, and M. Di Renzo, "Optimal Cache Leasing from a Mobile Network Operator to a Content Provider," *Proc. IEEE INFOCOM 2018*, 2018, pp. 2744–52.
[14] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-Time Analysis of the Multiarmed Bandit Problem," *Machine Learning*, vol. 47, no. 2–3, 2002, pp. 235–56.
[15] T. Deng *et al.*, "Cost-Optimal Caching for D2D Networks with User Mobility: Modeling, Analysis, and Computational Approaches," *IEEE Trans. Wireless Commun.*, vol. 17, no. 5, 2018, pp. 308–94.

Biographies

Yixue Hao (yixuehao@hust.edu.cn) received his B.E. degree from Henan University, China, and his Ph.D. degree in computer science from Huazhong University of Science and Technology (HUST), China, in 2017. He is currently working as a postdoctoral scholar in the School of Computer Science and Technology at Huazhong University of Science and Technology. His research includes 5G networks, the Internet of Things, and mobile edge computing.

Min Chen [SM'09] (minchen2012@hust.edu.cn) has been a full professor in the School of Computer Science and Technology at HUST since February 2012. He is Chair of the IEEE Computer Society STC on big data. His Google Scholars Citations reached 20,000+ with an h-index of 69. He received the IEEE Communications Society Fred W. Ellersick Prize in 2017 and the IEEE Jack Neubauer Memorial Award in 2019. His research focuses on cyber physical systems, IoT sensing, 5G networks, SDN, healthcare big data, and so on.

Donggang Cao (caodg@pku.edu.cn) is a research professor at the Software Institute, School of Electronics Engineering and Computer Science, Peking University, Beijing. He received his Ph.D. degree in computer software and theory from Peking University in 2004. His research interests include system software, parallel and distributed computing, cloud computing, and so on.

Weilai Zhao (zhaowenlai@tsinghua.edu.cn) received his Ph.D. degree from the Department of Computer Science and Technology at Tsinghua University. His research interests are heterogeneous parallel computing and distributed machine learning.

Ivan Petrov finished his Ph.D. at Lomonosov Moscow State University in 2019. His research interests are security (specifically exploit detection), networking (SDN), and clouds (resource schedulers). During his career at Lomonosov Moscow State University he worked on exploit detection tools for mobile and IoT platforms, shellcode detection algorithms at a network level, algorithms and tools for detecting compromised SDN switches in enterprise networks, and schedulers for meta-cloud computing environments.

Vitaly Antonenko is a Ph.D. student and researcher in the Computer Systems Laboratory (LVK lab), Moscow State University, and from 2005 to 2010, he studied as a student at Moscow State University. His main research interests are in the areas of SDN and NFV technologies, network protocols, cloud infrastructures, and cloud applications. He also participates in the university studying process, as a lecturer of a "practical work in computer networks" seminar for students.

Ruslan Smelionsky has participated in 42 research and industrial projects on simulation of computer networks, design and development of embedded real-time systems, OS for distributed computing systems, formal verification, and software engineering, of the Russian Foundation for Basic Research, Russian Academy of Science, Sun Microsystems, Cisco Systems, and Daimler AG as both principal investigator and project coordinator. Since 1996 he has given lectures on computer networks in the CMC Department, Lomonoscov Moscow State University, and has published 188 books and papers.