Received December 4, 2018, accepted December 14, 2018, date of publication December 19, 2018, date of current version January 11, 2019.

Digital Object Identifier 10.1109/ACCESS.2018.2888602

Zone-Based Cooperative Content Caching and Delivery for Radio Access Network With Mobile Edge Computing

NING WANG[1], GANGXIANG SHEN[1], (Senior Member, IEEE), SANJAY KUMAR BOSE[2], (Senior Member, IEEE), AND WEIDONG SHAO[1]

[1]School of Electronic and Information Engineering, Soochow University, Suzhou 215006, China
[2]Department of EEE, IIT Guwahati, Guwahati 781039, India

Corresponding author: Gangxiang Shen (shengx@suda.edu.cn)

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 61671313, in part by the Science and Technology Achievement Transformation Project of Jiangsu Province, China under Grant BA2016123, and in part by the Postgraduate Research and Practice Innovation Program of Jiangsu Province under Grant KYCX17_1984.

ABSTRACT Mobile edge computing (MEC) is a promising solution to meet the latency requirement for delay-sensitive services in a 5G radio access network (RAN). Its key idea is to deploy computing and storage capacities at the edge of the RAN to quickly provision content and processing capacities as required by the users. Efficient content caching and delivery are key issues to ensure the success of this technique. This paper proposes a zone-based cooperative content caching and delivery scheme for a RAN supporting MEC (MEC-RAN), where the RAN is modelled as a zone and is further sub-divided into multiple sub-zones. Content items are cooperatively cached and delivered among multiple sub-zones. The caching problem is formulated as a mixed integer linear programming model. We also develop a heuristic cooperative content caching strategy to decide the content items to be cached in each MEC server. This novel strategy divides the storage space in each MEC server into two parts. The first part caches locally popular contents and the second part is used to cooperatively cache zone-wide popular items. We study the proposed scheme both through simulations and implementation on a testbed that consists of a subnetwork on our campus and commercial cloud service from Ali-Cloud. Both of these show that the proposed zone-based scheme performs better than other typical caching strategies in terms of average content delivery latency and balanced loading of the MEC servers.

INDEX TERMS Content delivery network, mobile edge computing, optimal caching strategy, zone-based cooperative content caching.

I. INTRODUCTION

New applications, such as cloud computing, video-on-demand, and online gaming have led to the explosion of online information with a tremendous amount of content and very high network traffic. The deployment of the 4G access network and the popularity of smartphones have made mobile data increase to become a significant fraction of the total Internet traffic. According to Cisco [1], global IP traffic will increase nearly three-fold over the next five years and reach 3.3 ZB by 2021. In this, mobile data traffic is estimated to increase seven-fold and may account for more than 63% of the total IP traffic by 2021. Given this vast increase of mobile traffic and the large amount of content to be cached in the mobile Internet, Radio Access Networks (RANs) are facing a high pressure on their access capacity.

A 5G RAN is required to support three types of services, including those with large bandwidth, large number of connections, and low latency requirements. As a promising solution, the centralized cloud RAN (C-RAN) relocates the computing, storage, and networking functions from the local end to the cloud. This enables operators to manage the system in a more efficient and green way. However, the drawback of this is that there is a high traffic volume between mobile users and cloud servers which would lead to a heavy transmission load on the backhaul of the access system. Moreover, the latency to access a centralized site would also be a limiting issue for delay-sensitive applications. To overcome these drawbacks, solutions have been proposed to distribute a part of computing and storage capacities from the cloud to the network edge. This paradigm is known as Mobile Edge Computing (MEC) [3].

In a RAN supporting MEC (i.e., MEC-RAN), a MEC server is placed at each Base Station (BS) in close proximity to end users. This server generally has limited processing capacity and storage space. It is intended to provide fast processing capacity for latency-sensitive and context-aware applications and quickly accessible local storage for contents such as videos. The benefits of MEC are twofold. First, being close to end users, MEC can significantly shorten the latency of content delivery, thereby achieving better Quality of Experience (QoE) for users. Second, MEC can greatly relieve the load on the backhaul and core networks in a RAN since it is not required to retrieve contents from a remote cloud when these are locally cached.

There have been studies on MEC-RAN focusing on how to cache and deliver contents efficiently [4]. However, most of the existing research [5]–[7] assumes that user requests are evenly distributed and adopts simple content caching strategies. The key contribution of this paper is to propose a novel *zone-based cooperative content caching and delivery* scheme for MEC-RAN. For this, our goal is to achieve low *average content delivery latency* subject to the limited processing capacity and storage space at each MEC server. We also formulate the zone-based cooperative caching problem as a mixed integer linear programming (MILP) model. Then based on the insights obtained from the solution of the MILP model, we further propose a novel *MixCo strategy* to cooperatively cache contents within a zone. The efficiency of the proposed cooperative caching and delivery scheme is verified through simulations and experimentally on a testbed. Simulation and experimental results show the efficiency of the proposed *zone-based cooperative content caching and delivery* scheme.

The key novelty of the proposed scheme includes its hierarchical caching architecture and cooperative caching strategy. Specifically, it divides a MEC-RAN zone into multiple sub-zones, each of which corresponds to a BS. Within a zone, a cooperative caching strategy is employed for efficient sharing of storage space and processing capacity at each MEC server. For this, the storage space of each MEC server is divided into two parts. The first part is dedicated to storing the content items that are the locally most popular in its corresponding sub-zone and the second part is shared zone-wide to store the content items that are the most popular in the whole zone. This storage space division approach balances caching the content items based on both their local and global content popularities, thereby helping to reduce the overall (system-wide) average content delivery latency.

A part of this work has been presented earlier in [2], where a preliminary study showed the benefits of the mixed cooperative caching strategy in reducing the average content delivery latency. The full version of our work presented here considers a comprehensive framework for the *zone-based cooperative content caching and delivery* scheme and proposes an efficient online caching strategy called the *MixCo caching strategy*, which can achieve near optimum caching performance in terms of average content delivery latency. An experimental testbed was also built to implement the proposed zone-based content delivery scheme and the MixCo caching strategy to demonstrate its superior performance.

The rest of this paper is organized as follows. In Section II, we review the related works on MEC and content caching in a RAN. In Section III, we introduce the framework of *zone-based cooperative content caching and delivery*. In Section IV, we introduce our approach to cooperative content caching and present its MILP model. The heuristic MixCo caching strategy for the caching problem is presented in Section V. Simulations and experiments are carried out in Section VI, in which a testbed is introduced and the performance of the proposed scheme is evaluated. Section VII concludes this paper.

II. RELATED WORKS

MEC, first proposed by IBM and Nokia, is now recognized by European 5G PPP as one of the key technologies for the 5G network [8]. The European Telecommunications Standards Institute (ETSI) has been working on its standardization since 2014, and some specifications on how to deploy MEC servers in a RAN have been released recently [9], [10]. MEC provides an IT service environment and computing capacity at the edge of a network. It aims to ensure a highly efficient network by reducing service delivery latency and network operation costs, while also offering an improved user experience [8]. Early studies on MEC have focused on the network architectures. For example, Hu *et al.* [11] showed that using MEC servers within a long-term evolution (LTE) network to offload computing tasks from users' phones can significantly reduce response times. This was further verified in [12] and shown to save up to 51% response time compared to the case of offloading to the cloud. Therefore, the MEC-based architecture was considered promising for the 5G network to realize its 1-ms RTT requirement [13]. In addition to the response time, Bastug *et al.* [14] and Mehta *et al.* [15] also considered the benefit of the MEC-based architecture from the viewpoints of capacity utilization and operation cost. It was found that employing the MEC-based architecture can reduce the backhaul traffic by up to 22% and can save operation costs by up to 67% for some computing-intensive applications.

For a MEC-RAN, it is important to cache and deliver contents efficiently. Bastug *et al.* [14] explored the benefit of proactively caching content items at the edge of a 5G network. It was found that this practice could help improve user experience and alleviate backhaul congestion. Later, Intel in cooperation with China Mobile and iQIYI [16] demonstrated their online video system in the Mobile World Congress (MWC), where MEC servers were deployed near BSs to cache and transform videos for better user QoE [8]. There are also other studies on how to efficiently deliver contents based on similar architectures. Kwak *et al.* [17] proposed a joint CU/BS (Central Unit/Base Station) caching algorithm, which aims to select caching locations dynamically in RANs, for different contents, where the caching locations can be content servers, cloud units, and base stations. In [18], a cluster content caching structure was proposed, where distributed caching and centralized signal processing were done jointly to reduce redundant backhaul traffic and to improve the quality of service (QoS) in a C-RAN. For the more advanced content caching required in MEC-RAN, different cooperative caching schemes were also proposed and studied. Gharaibeh *et al.* [19] allowed each user to access multiple caching servers in neighboring BSs and proposed an online caching algorithm to minimize the total system cost. In [3], a cloud-based RAN caching framework was proposed based on a cooperative hierarchical caching scheme to minimize the cost of content delivery and optimize user QoE.

In addition to MEC-RAN, the fog computing-based RAN (F-RAN) is another important architecture for content caching and distribution in a RAN. In fact, F-RAN is more general than MEC-RAN as it allow any device that has processing and storage capacities to share these capacities in the whole system. Mouradian *et al.* [20] provided a comprehensive survey on F-RAN. For content caching and delivery in F-RAN, Hung *et al.* [21] proposed an algorithm aiming to identify which content item should be cached in the cloud and which content item should be cached in the fog, according to the content features. Xiang *et al.* [22] considered every user's equipment as a fog node and proposed an algorithm to optimize communication modes and resource allocation to tradeoff the average energy efficiency and average delivery latency. Do *et al.* [23] proposed an algorithm to manage the content in a distributed fog network aiming to optimize content delivery from a cloud to fog nodes while minimizing the carbon footprint of the whole system.

MEC-RANs have elicited considerable interest in recent years. However, several open problems remain to be explored which can further improve their content delivery performance. For example, most existing studies assume that the users are distributed evenly in a network and they ignore user mobility. However, in an actual system, the traffic loads from users are often unevenly distributed and users are also mobile, moving from one cell to another. Moreover, these prior studies generally ignore the processing/transmission capacity of the MEC servers considering only their limitations on storage space. For a more accurate optimization design, new system constraints incorporating these practical aspects should be comprehensively addressed. In addition, though hierarchical cooperative caching strategies have been considered, these have not been studied sufficiently. In this paper, we propose a *zone-based cooperative content caching and delivery* scheme for MEC-RAN to achieve a low average content delivery latency considering limited storage space and processing capacity at each MEC server and the fluctuations of user traffic load. The key novelty of the proposed scheme is in its division of the storage space in each MEC server into two parts, with the first part used for caching locally popular contents and the remaining part used for cooperatively caching zone-wide popular contents. Such a division enables the proposed scheme to outperform other existing schemes. Moreover, to demonstrate experimentally the proposed scheme and its performance, we also develop a testbed based on a real campus network and a commercial service from Ali-Cloud, which is also an important contribution.

III. ZONE-BASED COOPERATIVE CONTENT CACHING AND DELIVERY IN MEC-RAN

We first describe the MEC-RAN architecture, for which the proposed caching strategy is applied. Subsequently, we describe how each user's content request is handled in a cooperative way, which is followed by the research problem considered.

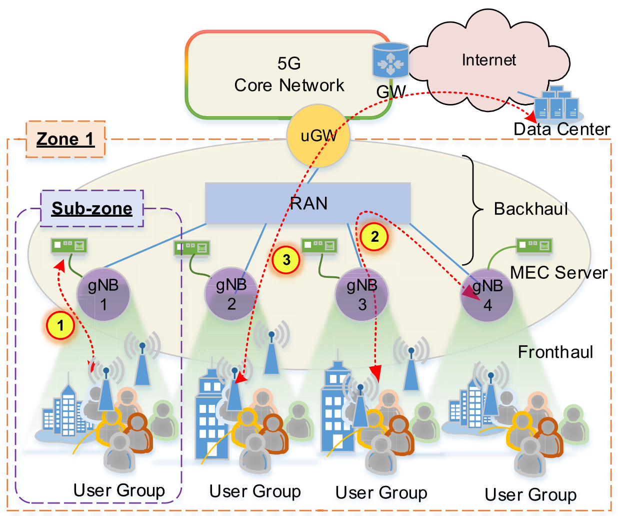


FIGURE 1. MEC-RAN architecture.

A. MEC-RAN ARCHITECTURE

Fig. 1 shows the architecture of a 5G MEC-RAN [8], [24]. The network is divided into three parts, consisting of a fronthaul, a backhaul, and a core network (CN). The fronthaul consists of multiple radio access points, which are connected to a base station (gNB). Usually, each radio access point covers a residential/business area, and each base station serves several such areas. All the users covered by a radio access point are served by this access node and further by the corresponding base station. The backhaul aggregates data traffic from multiple base stations to the core network through a user-plane gateway (uGW). Fiber links are generally employed to provide capacity for the backhaul transmission. Finally, the core network connects to the Internet via a network gateway, where a large data center (DC) is usually deployed nearby. This large DC caches all the contents and provides a very large processing capacity to the users, if required. However, the DC is generally far away from a local user since the fronthaul, the backhaul, and the core networks would lie between them. It would take a long time to deliver contents to users through all of these, which would not be friendly to latency-sensitive applications. To ensure a guaranteed QoE[1] for delay-sensitive applications, MEC servers (or micro-DCs) can be deployed at base stations to provide storage and processing capacity for local users [3].

B. ZONE-BASED COOPERATIVE CONTENT CACHING AND DELIVERY

In a MEC-RAN, how to efficiently cache and deliver contents to users via distributed MEC servers would be an important research problem. For this, we propose a zone-based cooperative content caching and delivery scheme. Specifically, we divide a MEC-RAN into multiple zones where each *zone* contains several base stations, which are connected to the uGW (see Fig. 1) via the backhaul network. Further, each zone is divided into multiple sub-zones where each *sub-zone* consists of a base station and multiple radio access points connected via the fronthaul network. Users in the same sub-zone are considered as a *user group* and all the users in their individual sub-zones receive their services from their local sub-zone's MEC servers at the first priority. Only if the content is not cached locally or if the local MEC server is fully loaded, would a user's request be forwarded to other MEC servers in the same zone. Of course, if no MEC server in the zone has the requested content cached, then the request is forwarded to the remote large DC as the final solution.

In the context of the above hierarchical architecture, there are three possible scenarios for content delivery. In **Scenario 1**, if a local MEC server caches a requested content and it is not fully loaded, then it will serve to provision the content and directly deliver it to a user as shown in Fig. 1. However, if the local MEC server does not have the requested content in its cache or if it is fully loaded, then the request will be forwarded to other MEC servers in the same zone. Any MEC server in the other sub-zones can serve the request if it has the requested content in its cache and is not fully loaded. If such a (non-local) MEC server can be found, then the user's request is forwarded to it, and this (non-local) MEC server delivers the content to the user. This corresponds to **Scenario 2** as shown in Fig. 1. Finally, **Scenario 3** occurs only if none of the MEC servers in the zone cache the content requested or if all of them are fully loaded. In that case, the request is forwarded to the remote large DC via the core network. Here the DC is assumed to host all the contents that may be requested.

It may be noted that the above delivery process is hierarchical and the delivery latency increases when the request is forwarded to a higher-level (i.e., non-local) servers. The content delivery latency of Scenario 1 is the shortest, while it is the longest in Scenario 3, and Scenario 2 has medium latency, which is somewhat longer than that of Scenario 1, but much shorter than that of Scenario 3. If most of the content requests can be served by the local MEC servers and the non-local MEC servers in the same zone, then the overall content delivery latency can be much shorter than the case where this local caching strategy is not implemented. However, a MEC server usually has limited storage space and processing capacity. In order to achieve good performance in terms of content delivery latency, it is important to properly cache contents in each MEC server. This is the focus of this study. Next, we introduce the approaches developed for deciding the cached contents in each MEC server.

IV. ZONE-BASED CONTENT CACHING PROBLEM AND ITS MILP MODEL

To decide which contents should be cached in each MEC server in the proposed cooperative caching and content delivery scheme, we first formulate the caching problem as a mixed integer linear programming (MILP) model. Then based on the insights obtained from the solution of the MILP model, we propose a heuristic caching strategy, i.e., the *MixCo* strategy.

A. PROBLEM STATEMENT

For the MILP model, we consider only one zone since the MEC servers in the same zone cooperatively cache and deliver contents to users. The given parameters of the problem are - (1) the average content delivery latencies of different scenarios shown in Fig. 1, (2) each MEC server's storage and processing capacity,[2] (3) the arrival rate of user requests in each sub-zone, and (4) the distribution of content popularity in each sub-zone and the storage space required by each content item.

The objective of the problem is to decide optimal caching contents in each MEC server so as to achieve the lowest average content delivery latency. The optimization problem is subject to the following two constraints. First, the total number of content items that can be cached by each MEC server is limited by the storage space of each server. Second, the processing capacity of each MEC server is limited and the total number of user requests that can be handled by each MEC server should not exceed its processing capacity.

B. MILP MODEL

We next present the MILP model for the above optimization problem. The sets, parameters, and variables of the model are defined as follows.

Sets:

S	Set of MEC servers in the same zone.
N	Set of content items.

Parameters:

L_1	Average content delivery latency of Scenario 1.
L_2	Average content delivery latency of Scenario 2.
L_3	Average content delivery latency of Scenario 3.

[1] Although there can be other performance metrics such as packet dropping ratio that can affect QoE, in the 5G era the delivery latency is often considered the most important performance metric for QoE.

[2] Here the processing capacity is decided by the computing capacity of the server and the transmission capacity of the link connected to the server. We use the lower of the two as the processing capacity of the server.

Objective :

$$\textit{Minimize } D \tag{1}$$

Subject to :

$$\sum\nolimits_{n\in \boldsymbol{N}} \rho_{i,n}\cdot \nu_n \le M_i \quad \forall i\in \boldsymbol{S} \tag{2}$$

$$\sum\nolimits_{i\in \boldsymbol{S}} \rho_{i,n} \le \psi_n\cdot\Delta \quad \forall n\in \boldsymbol{N} \tag{3}$$

$$\sum\nolimits_{i\in \boldsymbol{S}} \rho_{i,n} \ge \psi_n \quad \forall n\in \boldsymbol{N} \tag{4}$$

$$\xi_{i,n}^{j} \le \rho_{i,n}\cdot\lambda_i\cdot P_{i,n} \quad \forall n\in \boldsymbol{N}, i,j\in \boldsymbol{S} \tag{5}$$

$$\sum\nolimits_{j\in \boldsymbol{S}} \xi_{i,n}^{j} \le \lambda_i\cdot P_{i,n}\cdot\psi_n \quad \forall n\in \boldsymbol{N}, i\in \boldsymbol{S} \tag{6}$$

$$\sum\nolimits_{n\in \boldsymbol{N}, j\in \boldsymbol{S}, j\neq i} \xi_{i,n}^{j}\cdot T + \xi_{i,n}^{i}\cdot T \le K_i \quad \forall i\in \boldsymbol{S} \tag{7}$$

$$D = \frac{\sum_{i\in \boldsymbol{S}}\left(\sum_{n\in \boldsymbol{N}}\left(L_1\cdot\xi_{i,n}^{i} + L_2\cdot\sum_{j\in \boldsymbol{S}, j\neq i}\xi_{i,n}^{j} + L_3\cdot\left(\lambda_i\cdot P_{i,n} - \sum_{j\in \boldsymbol{S}}\xi_{i,n}^{j}\right)\right)\right)}{\sum_{i\in \boldsymbol{S}}\lambda_i} \tag{8}$$

M_i	Maximum storage space (MB) in MEC server i ($i \in \boldsymbol{S}$).
K_i	Maximum processing capacity in MEC server i ($i \in \boldsymbol{S}$), i.e., the maximum number of requests that can be handled by the MEC server simultaneously.
ν_n	Storage space required for caching content item n ($n \in \boldsymbol{N}$).
λ_i	Load of user requests in the sub-zone of gNB (or BS) node i ($i \in \boldsymbol{S}$).
$P_{i,n}$	Popularity of content item n ($n \in \boldsymbol{N}$) in the sub-zone of gNB node i ($i \in \boldsymbol{S}$). The relationship $\sum_{n\in \boldsymbol{N}} P_{i,n} = 1$ always holds.
Δ	A large value.

Variables:

$\rho_{i,n}$	A binary variable that equals 1 if content item n ($n \in \boldsymbol{N}$) is cached by MEC server i ($i \in \boldsymbol{S}$); 0, otherwise.
ψ_n	A binary variable that equals 1 if content item n ($n \in \boldsymbol{N}$) is cached by any MEC server in the same zone; 0, otherwise.
$\xi_{i,n}^{j}$	A real variable that indicates the fraction of request load of content item n ($n \in \boldsymbol{N}$) in sub-zone i ($i \in \boldsymbol{S}$) served by MEC server j ($j \in \boldsymbol{S}$).
D	A real variable that indicates the average content delivery latency in the whole zone.

Objective (1), as shown at the top of this page, is to minimize the average content delivery latency calculated by (8), as shown at the top of this page. In (8) $\xi_{i,n}^{i}$ is the load of content item n requested by users in sub-zone i that is served by its local MEC server. The sum term $\sum_{j\in \boldsymbol{S}, j\neq i}\xi_{i,n}^{j}$ is the total load of content item n requested by users in sub-zone i that is served by other MEC servers. Finally, $\lambda_i\cdot P_{i,n} - \sum_{j\in \boldsymbol{S}}\xi_{i,n}^{j}$ is the load of content item n in sub-zone i offloaded to the remote large DC. Constraint (2), as shown at the top of this page, ensures that the storage space required to cache content items at each MEC server does not exceed its storage limit. Constraints (3) and (4), as shown at the top of this page, ensure that if a content item is cached in the zone, then it must be cached by at least one of the MEC servers in the zone. Constraint (5), as shown at the top of this page, means that if a server j does not cache a content item n, then no request load of this content should be redirected to this server. Constraint (6), as shown at the top of this page, means that for a content item n in sub-zone i, the sum of its load served by different MEC servers in the zone should not exceed its total load. Constraint (7), as shown at the top of this page, ensures that the total load assigned to a MEC server should not exceed its processing capacity. Note that T is the average service time of each request in an MEC server. We normalize this to one.

The computational complexity of a MILP model is generally decided by the dominant numbers of variables and constraints. In the above MILP model, the dominant number of variables is decided by the variable $\xi_{i,n}^{j}$, which is of the order O($|S|^2\cdot|N|$), where $|S|$ is the total number of MEC servers in the zone and $|N|$ is the total number of content items. Similarly, the dominant number of constraints of the model is of the order O($|S|^2\cdot|N|$) due to constraint (5).

C. OPTIMAL RESULTS AND OBSERVATIONS

To visually show how content items would be distributed in each MEC server under an optimal solution, we have used the commercial software AMPL/Gurobi [25] to solve the MILP model for a small test scenario, where we assume that there are three MEC servers in a zone, and each MEC server has the same storage space (i.e., 150 content items) and processing capacity (i.e., 50 requests simultaneously). These MEC servers together form a zone and each server corresponds to a sub-zone, which serves its corresponding user group. For simplicity, we assume that there are a total of 1000 content items and each user group has the same distribution of content popularity.

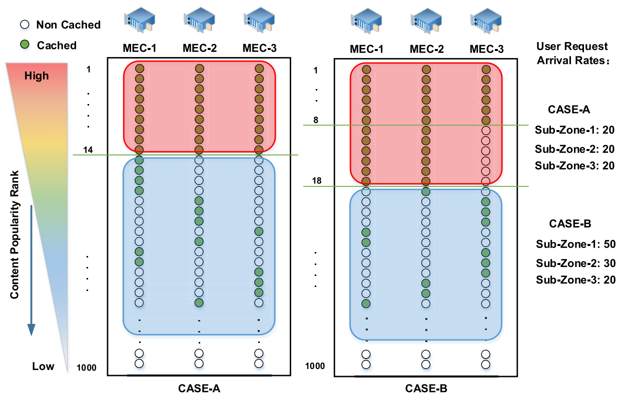


FIGURE 2. An illustration of content items cached in each MEC server (based on the MILP model).

Fig. 2 illustrates the optimal solutions for two different cases. In case A, each sub-zone has the same arrival rate of user requests and the arrival rate is overall lower than that of case B. In case B, the arrival rates of user requests are different in different sub-zones and the arrival rates are higher than that of case A. The extreme right-hand side shows the arrival rates of user requests of each sub-zone. The extreme left-hand side displays the content items to be cached according to their popularities from the highest to the lowest. The two boxes in the middle correspond to the two cases, i.e., case A and case B, illustrating the status of cached items in each MEC server. Note that the status of these cached items was obtained by the MILP model. Green points mean that the corresponding content items are cached in the MEC servers. For both cases, we can see that the items with the highest popularities are cached in all the MEC servers (items 1-14 for case A and items 1-8 for case B). Items with lower, but still relatively higher popularities, are cached in a partial set of MEC servers (items 9-18 in case B), and many items with low popularities are either not cached in any of the MEC servers or are cached in only one MEC server. This observation is reasonable because caching items with high popularities would significantly reduce average delivery latency as compared to the case of caching less popular items.

In addition, comparing the two cases that have different user request arrival rates (or loads), it is interesting to see that the case with a higher load (i.e., case B) has more cached items (i.e., green points) on the upper side in the corresponding box (i.e., the red square with round corners). This implies that it is preferable to reserve more space for caching popular items under a higher load. This is attributed to the following facts. Each MEC server has a limited processing capacity. At a low user request load, this processing capacity is sufficient to process many items with different popularities. As a result, the cached items are well distributed in different popularity regions as shown in case A. However, when the user request load becomes higher, the processing capacity becomes insufficient. This would then require the system to reserve more capacity to process more popular items in order to gain the most performance benefit for low average delivery latency. Therefore, items that are more popular are cached in each MEC server to compensate for the scarce processing capacity (see case B).

V. HEURISTIC CACHING STRATEGY

The MILP model can find an optimal solution to the problem for small network scenarios. However, when the system becomes large, it would be infeasible to use the MILP model to find the optimal solution. To handle such systems, we have developed an efficient heuristic caching strategy, called the *Mixed Cooperative* (*MixCo*) *caching strategy*, which is dedicated to handling the zone-based cooperative content caching and delivery scheme.

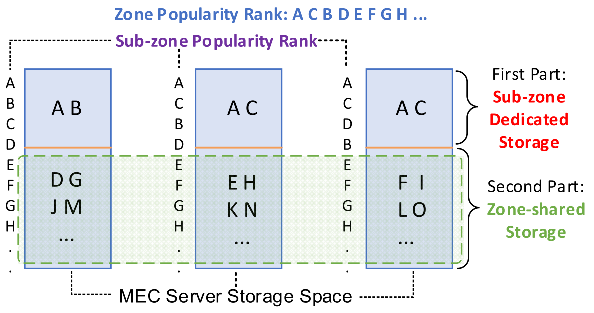


FIGURE 3. An example of the MixCo caching strategy.

According to the results obtained by the MILP model, as shown in Fig. 2, it is interesting to observe that the caching space at each MEC server is divided into two parts. One is covered by a red square with round corners, in which items with high local popularities are cached in almost all the MEC servers of the zone. The other is covered by a blue square with round corners, in which items with relatively lower popularities are cached in *at most one server* in the zone. This observation leads us to the key principle of our proposed MixCo caching strategy. Specifically, we divide the storage space of each MEC server into two parts as shown in Fig. 3. The first part is referred to as *Sub-Zone-Dedicated Storage Space*, which is used to cache the items with the highest *local* popularities in the corresponding sub-zone. The second part is referred to as *Zone-Shared Storage Space*. Note that the percentages of these two storage spaces are different depending on the local content popularities and the load on each MEC server. All the zone-shared storage spaces at the MEC servers are considered together to form a *zone-wide common storage space*. This space is used to cache *cooperatively* the items that have not been cached yet. More specifically, we select the items cached in this zone-wide common storage space based on zone-wide popularities, rather than their local sub-zone popularities. Here the zone-wide popularity of an item is calculated by (9), which is a load-weighted average of its sub-zone popularities.

$$\Gamma_n = \sum_{i \in S} \lambda_i \cdot P_{i,n} \quad \forall n \in N \tag{9}$$

Here λ_i is the load of user requests in sub-zone i and $P_{i,n}$ is the popularity of content item n in sub-zone i. Moreover, to cache as many content items as possible in the zone-shared storage space, each content item cached in this part has only one copy, which means that only one MEC server caches the content item.

In the example of Fig. 3, we first cache the items that have the highest local popularities in each sub-zone-dedicated storage space of the MEC servers. Therefore, items (A, B), (A, C), and (A, C) are cached in the three sub-zone-dedicated storage spaces, respectively. Then for the zone-wide common storage space, which is formed by the zone-shared storage spaces of the three servers, we cache the items that have not been cached according to their zone-wide popularities, where only one copy of each item is cached in the whole zone (i.e., in the second part of the storage space in each server).

In the above process, there are two critical questions. First, of the total storage space in a MEC server, how much should be reserved as the s*ub-zone-dedicated storage space* and how much should be reserved as *zone-shared storage space*? Second, how should we select the items to be cached in the *zone-shared storage space* in each MEC server? We examine these issues next.

A. DECIDING THE SIZES OF SUB-ZONE-DEDICATED STORAGE SPACES

To decide the sizes of the sub-zone-dedicated and zone-shared storage spaces in each MEC server, we developed an algorithm, whose key idea is to increase the size of the sub-zone-dedicated space gradually while examining the system's average delivery latency. If an increase in the sub-zone-dedicated space reduces the latency, then we keep this increase and continue to try further to grow the sub-zone-dedicated space; otherwise, we terminate this process. Since the size of sub-zone-dedicated space is increased gradually, we call this algorithm *Storage Space Growing (SSG)* algorithm. The pseudocode of this algorithm is given below.

Algorithm 1 Storage Space Growing (SSG)

Input: Each MEC server's storage space M_i and processing capacity K_i; each sub-zone's content popularity distribution $P_{i,n}$ and user request load λ_i, the average content delivery latency of the three scenarios in Fig. 1 (L_1, L_2, and L_3).

Output: The size of sub-zone dedicated storage space in each MEC server.

```
Step 1 Sort the sub-zones (or MEC servers) in the system according to their user request arrival rates in
       descending order; initialize the size of sub-zone dedicated storage space in each MEC server to be $f_i = 0$,
       the system delivery latency as $l_{pre} = +\infty$, and $s = |\boldsymbol{S}|$;
Step 2 For $i = 1, i \leq s, i++$ {
           Increase the sub-zone dedicated storage space of the $i^{th}$ server by one unit, i.e., $f_i++$;
           Evaluate the average content delivery latency $l_i$;
       }
Step 3 Find the case that has the lowest delivery latency, i.e., $i^* = \arg l_i$;
Step 4 If $l_{i^*} < l_{pre}$ {
           Update $l_{pre} = l_{i^*}$ ;
           For $i = i^* + 1, i \leq s, i++$ {
               $f_i--$; // recover the sub-zone dedicated storage space of each MEC server whose indexes are
               from $i^* + 1$ to $s$ back to their
               previous sizes by reducing one unit
           }
           Go back to Step 2;
       }
       else
           Go to step 5;
Step 5 Output the final sub-zone storage space size of each MEC server.
```

In this algorithm, we first sort the MEC servers according to their arrival rates of user service requests in descending order (Step 1). In Step 2, we increase the sub-zone dedicated storage space by one unit, server by server. For each increase, as shown in Fig. 4 (in the k^{th} iteration), we then evaluate the average delivery latency of the system; this is denoted as l_i^k. In Step 3, we find the index i^* of the server whose increase of the sub-zone dedicated storage space leads to the lowest average delivery latency l_{i^*}, i.e., by comparing all l_i^k obtained in Step 2, we choose the smallest one. In Step 4, if l_{i^*} is lower than the previous recorded lowest latency l_{pre}, we keep all the storage space increases of the servers whose indexes are not greater than i^*, and recover the storage spaces of all the servers whose indexes are larger than i^* by reducing them by one unit. We then return to Step 2 to repeat the same process, i.e., for the $(k+1)^{th}$ iteration. We again increase the sub-zone dedicated storage space by one unit, server by server, and then evaluate the average delivery latency of the system l_i^{k+1} (see the right-hand side of Fig. 4). Otherwise, we terminate the algorithm to output the final sub-zone storage space size of each MEC server that jointly achieves the lowest average delivery latency (Step 5). This allocation process leads to a staircase shape for the sub-zone dedicated storage spaces in the servers, as shown in case B of Fig. 2 and Fig. 4. Here the servers with higher user request arrival rates are assigned with larger sub-zone dedicated storage spaces, while the servers with lower arrival rates of user requests are assigned with smaller sub-zone dedicated storage spaces.

The SSG algorithm has a key step of evaluating the average delivery latency given a certain distribution of storage spaces in each of the MEC servers. We provide its details later in Section V-C. Specifically, to evaluate the average delivery latency, we first fill the content items that are locally the most popular in each sub-zone in the sub-zone dedicated storage space first, and then for the remaining storage spaces on the MEC servers, we consider them jointly as zone-shared storage space and fill content items using the

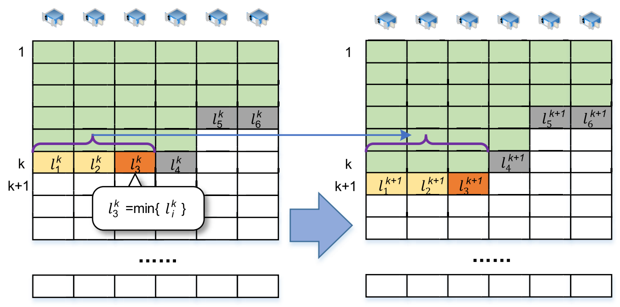

1
k
k+1
l_1^k l_2^k l_3^k l_4^k l_5^k l_6^k
l_3^k =min{ l_i^k }
......
l_1^{k+1} l_2^{k+1} l_3^{k+1} l_4^{k+1} l_5^{k+1} l_6^{k+1}

FIGURE 4. **Assignment of sub-zone dedicated storage space.**

Algorithm 2 Caching Items in the Zone-Shared Storage Space

Input: Each MEC server's dedicated storage space f_i and processing capacity K_i; each sub-zone's content popularity distribution $P_{i,n}$ and user request load λ_i.

Output: The content items cached in the zone-shared storage space of each MEC server.

```
Step 1  Remove all the items that have been cached. Sort
        the remaining items according to their zone-wide
        popularities calculated by (9) in descending order and
        store them in $C_r$.
Step 2  For each $c \in C_r$ {
            Find its arrival rate weighted popularity in each
            sub-zone $i$, i.e., $\lambda_i \cdot P_{i,c}$;
            Sort all the servers based on $\lambda_i \cdot P_{i,c}$ in
            descending order and store them in $S$;
            For each $s \in S$ {
                If $s$ has sufficient processing capacity for $c$, then
                $s$ caches $c$ and break;
            }
            If $c$ is not cached by any server, then it is cached
            by the last server in $S$;
        }
```

steps given next (i.e., Algorithm 2). Therefore, although the steps (or algorithms) from Section V-A to Section V-C are described separately one by one, they are essentially joint, together incorporated in Algorithm 1.

B. CACHING ITEMS IN THE ZONE-SHARED STORAGE SPACE

After we decide the sub-zone dedicated storage space in each MEC server, we cache the content items that are locally the most popular in each sub-zone into the corresponding sub-zone dedicated storage space. Next, for the less popular content items, we cache them in the zone-shared storage space, which is formed by aggregating all the remaining storage spaces in each MEC server. The details on how to select and cache items in the zone-shared storage space are given next in Algorithm 2.

We first remove all the items that have been cached in the sub-zone dedicated storage spaces from the item list. Then for the remaining items, we first sort them based on their zone-wide popularities which are calculated by (9) in descending order. Then we retrieve an item c from this ordered list C_r and find the arrival rate weighted popularity of the item in each sub-zone i, i.e., $\lambda_i \cdot P_{i,c}$. Further, we order the sub-zones (or MEC servers) based on the previous arrival rate weighted popularities in descending order, which forms an MEC server list S. We check the load on each MEC server $s \in S$ to see if the server would be overloaded when the current item c is cached in server s (the details on how to estimate the current load for a server is given in Section V-C). If not, we cache item c in server s and then consider the next content item in the list C_r; otherwise, we consider the next MEC server in S to carry out the same examination. As an extreme case, if each of the MEC servers in S is overloaded when the current item c is cached in it, then we use the last server in S to cache item c since it has the lowest load.

C. ESTIMATING THE CURRENT LOAD OF A SERVER AND THE AVERAGE DELIVERY LATENCY OF A SYSTEM

1) CURRENT LOAD OF A SERVER

In Section V-B, when deciding whether a content item can be cached in a MEC server, we need to estimate the current load of this server to ensure that caching this new item would not overload its processing capacity. We use the following equations to estimate the load.

$$b_i = \sum\nolimits_{n \in N} b_{i,n} \tag{10}$$

$$b_{i,n} = \Big(\lambda_i \cdot P_{i,n} \cdot T + \sum\nolimits_{j \epsilon S, j \neq i} (1-\rho_{j,n}) \cdot \lambda_j \cdot P_{j,n} \cdot T / \sum\nolimits_{j \epsilon S} \rho_{j,n}\Big) \cdot \rho_{i,n} \tag{11}$$

Here $b_{i,n}$ is the estimated load of content item n in MEC server i and therefore (10) finds the total load on server i. $b_{i,n}$ is calculated by (11), which includes two parts. The first part is the load of content item n from the local sub-zone of server i, and the second part calculates the overall load of content item n from other sub-zones. Here T is the average service time of each request in an MEC server. We normalize it to one.

2) AVERAGE DELIVERY LATENCY OF A SYSTEM

Given the content items cached in the storage space of each server, we use the following equations to evaluate the average delivery latency l_a of the whole system.

$$l_a = \sum\nolimits_{i \in S, n \in N} l_{i,n} / \sum\nolimits_{i \in S} \lambda_i \tag{12}$$

$$l_{i,n} = \begin{cases} \lambda_i \cdot P_{i,n} \cdot L_1, & \text{if } \rho_{i,n} = 1, \psi_n = 1 \\ \lambda_i \cdot P_{i,n} \cdot L_2, & \text{if } \rho_{i,n} = 0, \psi_n = 1 \\ \lambda_i \cdot P_{i,n} \cdot L_3, & \text{if } \rho_{i,n} = 0, \psi_n = 0 \end{cases} \tag{13}$$

Here (13) finds the load-weighted delivery latency $l_{i,n}$ for content item n in sub-zone i, where $\rho_{i,n}$ and ψ_n jointly decide whether content item n is cached in server i or in other servers in the same zone. λ_i is the user request load in sub-zone i; $P_{i,n}$ is the popularity of content item n in sub-zone i. L_1, L_2, and L_3

are the content delivery latencies which have been defined in the previous MILP model section.

D. COMPUTATIONAL COMPLEXITY ANALYSES

The computational complexities of the above heuristic algorithms are analyzed as follow. In Algorithm 2 which decides items to be cached in the zone-shared storage space, we have the computational complexity of $O(|S| \cdot |C|)$ to calculate the current load on each server and the average delivery latency of the system, where $|S|$ is the total number of MEC servers and $|C|$ is the total number of content items to be cached. Similarly, in Algorithm 1 which decides the size of the dedicated storage space, there is an iterative loop to judge if the performance would be improved with the increase of the dedicated storage space in each server. In the loop, the major computation is for the evaluation of the average content delivery latency (i.e., Algorithm 2 plus the step of average delivery latency calculation), which has the computational complexity of $O(|S| \cdot |C|)$. In the worst case, the maximum number of iterations that can be taken by the loop is of the order $O(|N| \cdot |S|)$, where $|N|$ is the size of storage space at each MEC server. Therefore, the overall computational complexity of Algorithm 1 is of the order $O(|N| \cdot |S|^2 \cdot |C|)$.

VI. TEST CONDITIONS AND PERFORMANCE ANALYSES

In this part, we evaluate the performance of the proposed caching strategy and heuristic algorithm from two perspectives. We first employ computer simulations to qualitatively analyze the benefit of the proposed scheme. Then we set up a testbed to experimentally verify the performance of the proposed scheme.

TABLE 1. Simulation Settings.

Total # of items	1000 units	**Delivery latency L1**	0.5~1.5 ms
# of MEC servers	3	**Delivery latency L2**	2~4 ms
Server storage space	150 units	**Delivery latency L3**	10~20 ms
Server computing capacity		50 user sessions	

A. SIMULATION STUDIES

For the simulation studies, we have the following conditions and assumptions. There are 1000 content items in the whole system and each item is assumed to take one unit of storage space. Three MEC servers are assumed. Each MEC server is assumed to have 150 units of storage space. To ensure a good QoE, we assume that each MEC server can host at most 50 user connections simultaneously, which corresponds to 50 units of processing capacity. For the delivery latencies of the different scenarios, we assume the ranges of [0.5, 1.5], [2, 4] and [10, 20] ms for L_1, L_2, and L_3, respectively, which are the typical targets in the 5G era. All these simulation parameters are shown in Table 1. We also assume that the remote large DC caches all the content items and can handle all the users' requests. The user request arrival process in each sub-zone is assumed to be a Poison process with an arrival rate in units of requests/second. The distribution of content popularity in each sub-zone is assumed to follow the Zipf distribution [27], [28] (Here we set the value characterizing this distribution, i.e., α, to be 0.88). We employed the commercial software package AMPL/Gurobi [25] (version 7.2) to solve the MILP model on a 64-bit server with 2.4-GHz CPU and 24-GB memory. The MIPGAP for solving the MILP model was set to be 0.001. In addition, we employed JAVA to implement the heuristic algorithm.

We evaluate the performance of the proposed content caching and delivery scheme in terms of the average content delivery latency and the average load on each MEC server. The average content delivery latency is calculated by (12) and (13). The average load on each MEC server evaluates the average usage of processing capacity on each MEC server. For performance comparison, we consider three other content caching schemes, which are the *Distributed*, *Self-Top*, and *Least Recently Used* (LRU) [29] strategies. Specifically, the Distributed strategy caches the content items from the perspective of the whole zone. It caches at most one copy of each content item in the whole zone based on their zone-wide popularities. This strategy can cache as many content items as possible in the zone. In contrast, the *Self-Top* strategy focuses on each sub-zone to cache the most popular items according to their sub-zone popularities. Finally, the LRU strategy caches new recently requested content items to replace the least recently used content items for each sub-zone.

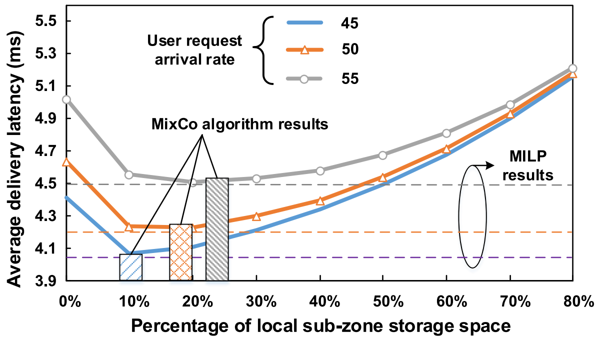


FIGURE 5. Average delivery latency changes with an increasing percentage of local sub-zone storage space.

1) IMPACT OF LOCAL SUB-ZONE STORAGE SPACE

According to the results obtained by the MILP model in the previous simple example, we can see that for the MixCo strategy, it is critical to assign a proper percentage of storage space (i.e., the local sub-zone storage space) for each MEC server to cache content items that are the most popular locally. Therefore, in this section we first evaluate how the average content delivery latency changes with different local sub-zone storage spaces assigned on each MEC server. We show the results in Fig. 5. For simplicity, but without losing generality, we assume that each sub-zone has the same user request arrival rate.

The results show that the average delivery latency changes with different sizes of the local sub-zone storage space under different user request arrival rates. The dotted lines show the results of the MILP model, which function as the best achievable performance. The solid curves show the average delivery latency when different percentages of storage space of a MEC server is assigned as the local sub-zone storage space. We see that at the beginning, increasing the local sub-zone storage space significantly reduces the latency. This is because more popular items can then be locally cached, thereby reducing the delivery latency. However, when the local sub-zone storage space increases further, the latency begins to increase. This means that caching too many locally popular items is not helpful in reducing the average delivery latency of the system and a balance is required to cache both locally popular items and zone-wide popular items. This balance is just the borderline for the local sub-zone storage space and the zone-shared storage space on each server. We observe that for the user request arrival rate $\lambda = 45$ requests/second, the best percentage of the local sub-zone storage space is $\sim$10%, and for $\lambda = 50$ requests/second and $\lambda = 55$ requests/second, these percentages are $\sim$20% and $\sim$25%, respectively. It is reasonable to reserve more space as the local sub-zone storage space when the local user request arrival rate increases since it is always beneficial for an MEC server to process its local requests at the first priority. It is also reasonable that an increasing arrival rate will lead to an increasing average delivery latency of the system. For each of the optimal local sub-zone storage spaces, we have used histogram bars in Fig. 5 to show the lowest delivery latencies for different user request arrival rates. Comparing the results of the histogram bars with those of the MILP model, we see that they are very close to each other. This verifies the efficiency of the proposed MixCo caching scheme in minimizing the average delivery latency of the system.

2) PERFORMANCE COMPARISON BETWEEN CACHING SCHEMES

In this section, we compare the performance in terms of average delivery latency and MEC server load distribution of the different caching strategies, including the *MixCo*, *Distributed*, *Self-Top*, and *LRU* strategies (see Fig. 6). To be more realistic, we consider uneven traffic loads, where the user request arrival rates in the three different sub-zones are 50, 25, and 20 requests/second. This may be considered as an example where the three sub-zones essentially correspond to one business area and two residential areas in the daytime (according to the distribution of their user request arrival rates).

For the average delivery latency, we see that the MixCo strategy is the most efficient as it shows the lowest average delivery latency. The Self-Top strategy ranks second, with the Distributed strategy following, and the LRU strategy is the worst. This is reasonable since the MixCo strategy considers caching content items based on both local sub-zone content popularities and zone-wide popularities. Moreover, the MixCo strategy tries to balance the processing capacity of the MEC server versus its storage space. When its local user request load is high, it tends to reserve more space as sub-zone-dedicated storage space. This is so that more processing capacity can be used for serving local sub-zone users; otherwise, it reserves a smaller space for its local sub-zone, but more space for zone-shared storage space, under which more processing capacity is used to deliver contents to the users in other sub-zones. In contrast, all the other three strategies do not support this cooperative feature for balancing the storage space and the processing capacity, but simply follow their own individual principles. For example, the Distributed strategy does not consider the local sub-zone content popularities, but only the zone-wide popularities. Similarly, the Self-Top strategy only considers the local sub-zone content popularities, but ignores the zone-wide popularities.

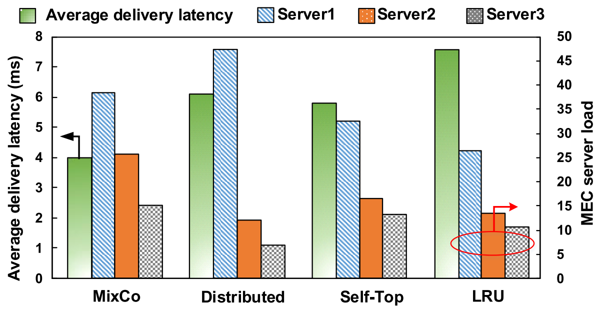


FIGURE 6. Performance comparison between the different caching strategies.

Fig. 6 also shows the load on each MEC server under the different caching strategies. Overall, we see that the load on each server is closely related to their user request arrival rates. The sub-zone of server 1 has the highest user request load, i.e., 50 requests/second, so its server load is the highest. Similarly, because the sub-zone of server 3 has the lowest load, its server load is the lowest. Comparing the load distribution among the servers, we see that the MixCo strategy shows a better balance in the loads on the MEC servers than the other strategies. This is attributed to the feature of the MixCo strategy that reserves a portion of storage space to cache content items based on the zone-wide popularities. This increases the chance for a server to serve a user that in non-local (i.e. other) sub-zones, thereby balancing the loads between the different sub-zones.

B. TESTBED STUDIES

The simulation results demonstrate the efficiency of the proposed zone-based scheme. For further confirmation, we also set up a testbed to verify the performance of the proposed scheme through actual experiments.

1) TESTBED SETTING UP

We set up a testbed that consists of a subnetwork on our campus and a commercial cloud service from Ali-Cloud [26]. The testbed forms a zone with three MEC servers located at different places. One is in our computer lab and the other two are in student dormitories. We use high-end PCs to function as MEC servers. These servers are far away from each other on the campus, and each of them is responsible for content caching and delivery of a sub-zone. The capacity of the link connected to each MEC server is 100 Mb/s. We also use computers in different locations to simulate users, whose maximum communication capacity is set to be 20 Mb/s. To ensure a good user QoE, each MEC server is set to serve at most five users simultaneously. In addition, to simulate a remote large DC, we rent a cloud service from Ali-Cloud, which is located in another city, ∼150 km away from our campus. The maximum communication capacity of the rent cloud server is 500 Mb/s. A total of 1000 content items are cached, and each item takes 1-MB storage space. Each MEC server is assumed to cache at the most 100 items, and the rented cloud server is assumed to have a space that is large enough to host all the items. It should be noted that since 5G networks are still not available, we carried out this experiment based on today's network environment. Therefore, this may lead to a delivery latency longer than the one required by 5G networks. However, this experiment is still valid in verifying the phenomena and conclusions found in our earlier simulation studies.

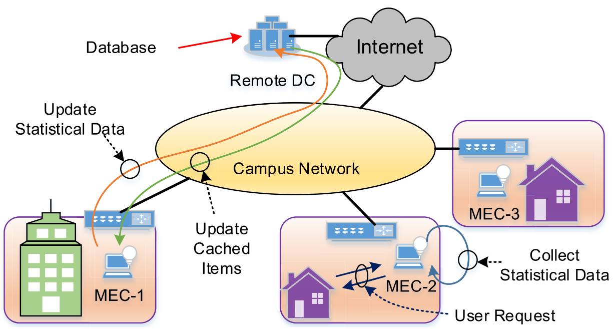


FIGURE 7. Testbed for the proposed zone-based MixCo caching strategy.

The software of the testbed was implemented in JAVA and the proposed MixCo caching strategy was incorporated. Fig. 7 shows a logical flow on how the caching strategy is incorporated and how the content items are decided to be cached in each MEC server. Specifically, each MEC server keeps on collecting statistical information on user requests in its corresponding sub-zone in real time and later uploads this information (suitably abstracted) to a user service database, which is located in the remote large DC. Here the large DC functions as a central controller to run periodically an algorithm based on the MixCo caching strategy, which also decides the items to be cached in each MEC server. It then instructs each MEC server to adjust its cached content items accordingly, which includes clearing less popular items and downloading new popular items from the remote large DC. The frequency of such interactive updating is set to be once per hour. Note that using the large DC to run this caching algorithm has the advantage of reducing the processing load on each MEC server as the large DC has sufficient computing capacity while a local MEC server would generally be of limited capacity.

2) TESTBED RESULTS

To carry out the experiment, we generate a *request list* for each user. The arrival of the requests follows a Poisson process and the popularity of each content item follows the distribution that we defined earlier, i.e., the Zipf distribution with $\alpha = 0.88$. We serve each user request in the following order. It is first served by the MEC server in the same sub-zone; otherwise, the request is forwarded to the other MEC servers in the same zone; if none of the servers in the zone can serve the request, then it is further forwarded to the remote server of Ali-Cloud. We collect the data of delivery latency for each request and finally calculate their average. Here the delivery latency of a request is defined as the time elapsed between the moment when the request is sent and the moment when the first batch of data of the request reaches a user device. For a video application, this first batch of data can be a video frame.

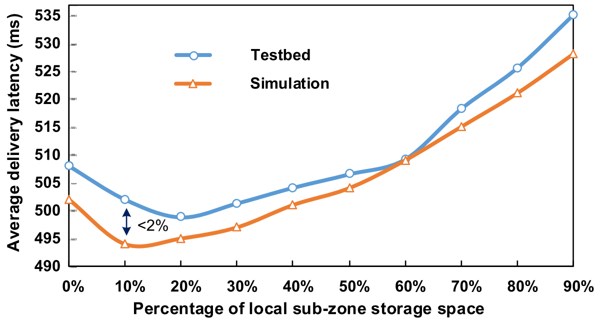


FIGURE 8. Impact of local sub-zone storage space on average delivery latency.

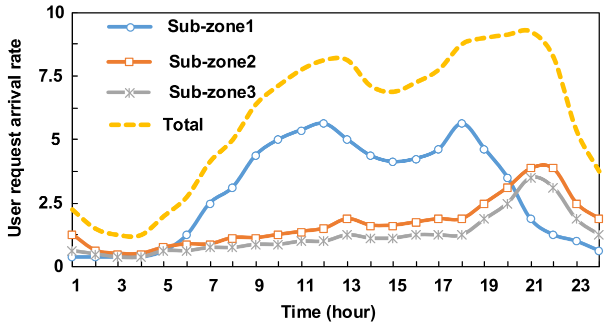


FIGURE 9. Hourly user request arrival rates.

We first evaluate how the average content delivery latency changes with different percentages of local sub-zone storage space in each MEC server. Fig. 8 compares the average content delivery latency in the testbed experiment with that obtained from simulations, when the arrival rate of user requests is one request/second. We can see that overall, the two curves have very similar trends with the average delivery latency first reducing and then increasing with an increasing percentage of local sub-zone storage space. The maximum difference between the delivery latencies obtained, from the testbed and from simulations, is less than 2%.

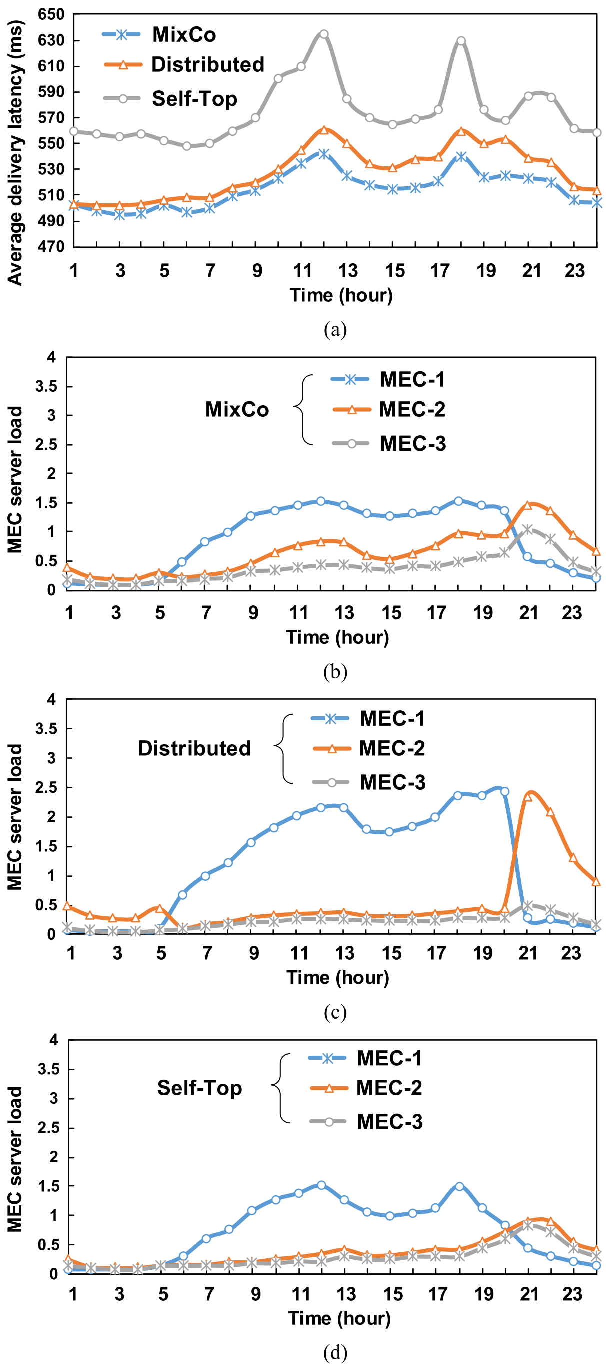


FIGURE 10. Experiment results. (a) Average delivery latency. (b) MEC server loads under the MixCo strategy. (c) MEC server loads under the Distributed strategy. (d) MEC server loads under the Self-Top strategy.

Moreover, a minimum average delivery latency is also observed in the experiment, which agrees with what was found in the previous simulation study. This therefore verifies the effectiveness of the previous simulation study for evaluating the performance of a 5G MEC-RAN.

In addition to a uniform single user request load, we also ran experiments for a period of one day under variable hourly user request loads as shown in Fig. 9. Here sub-zone 1 corresponds to a business area with the highest user request load in the daytime, while the other two sub-zones correspond to residential areas where the user request load distributions are reversed (i.e., less during the day and more at night). This set up simulates user mobility in different time periods, i.e., in the daytime, people move to the business area for their jobs, while in the evening, they go home after work.

The experimental results of the different caching strategies are shown in Fig. 10. Specifically, Fig. 10(a) compares the average content delivery latencies of the three caching strategies. Figs. 10(b)(c)(d) show the load on each MEC server under the different caching strategies. Here again, we see that the average delivery latency is the lowest when the MixCo strategy is employed, and its servers have more balanced loads than those under the other strategies even when user mobility and variable user request loads are considered. The reason for this is the same as what has been reported in the earlier simulation studies and therefore reconfirms the effectiveness of the proposed MixCo strategy.

VII. CONCLUSIONS

To reduce the content delivery latency in a MEC-RAN, we developed a *zone-based cooperative content caching and delivery* scheme that handles user requests in a zone by dividing it into multiple sub-zones. Based on this paradigm, we further proposed a MixCo cooperative caching strategy to divide the caching space in each MEC server into two parts with the first part caching the content items that are the most popular locally and the second part caching the content items that are popular over the zone. To evaluate the performance of the proposed scheme and the caching strategy, we formulated the caching problem as a MILP model. We also developed an efficient MixCo heuristic algorithm to divide the storage space in each MEC server and decide the content items to be cached in these spaces. A testbed was also built to evaluate the performance of the proposed scheme from actual experiments. Both simulation and experimental studies showed the effectiveness of the proposed scheme in terms of lower average delivery latency and better load balancing between the MEC servers in comparison with the other caching strategies. In addition, it is interesting to find an optimal division boundary for the storage space in each MEC server, at which the average delivery latency of the system is minimized. Our simulation studies also showed that the proposed MixCo strategy is very efficient and performs close to the MILP model in terms of the average delivery latency. In addition, it was also found that the simulation results agree well with the experimental results thereby verifying the accuracy of the simulations in evaluating the performance of MEC-RAN.

In future studies, we may further look into the following two key problems. The first is to evaluate how the size of each zone can impact the performance of the proposed scheme. We need to find if there is an optimal zone size, which can best exploit the benefit of the proposed scheme. Second, we may extend the current scheme to a more general scenario, i.e., fog computing, in which user equipment can also be used to cache and deliver content items.

VIII. ACKNOWLEDGMENT

This paper was presented at OFC 2018 [2].

REFERENCES

[1] *Visual Networking Index: Forecast and Methodology, 2016–2021*, Cisco, San Jose, CA, USA, Jun. 2017.

[2] N. Wang, W. Shao, S. K. Bose, and G. Shen, "MixCo: Optimal cooperative caching for mobile edge computing in fiber-wireless access networks," in *Proc. Opt. Fiber Commun. Conf. Expo. (OFC)* Mar. 2018, pp. 1–3.

[3] T. X. Tran, A. Hajisami, P. Pandey, and D. Pompili, "Collaborative mobile edge computing in 5G networks: New paradigms, scenarios, and challenges," *IEEE Commun. Mag.*, vol. 55, no. 4, pp. 54–61, Apr. 2017.

[4] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, 4th Quart., 2017.

[5] H. Ahlehagh and S. Dey, "Video-aware scheduling and caching in the radio access network," *IEEE/ACM Trans. Netw.*, vol. 22, no. 5, pp. 1444–1462, Oct. 2014.

[6] J. Gu, W. Wang, A. Huang, H. Shan, and Z. Zhang, "Distributed cache replacement for caching-enable base stations in cellular networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2014, pp. 2648–2653.

[7] A. Ioannou and S. Weber, "A survey of caching policies and forwarding mechanisms in information-centric networking," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 4, pp. 2847–2886, 4th Quart., 2016.

[8] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing—A key technology towards 5G," ETSI, France, U.K., White Paper 11, 2015, pp. 1–16.

[9] F. Giust *et al.*, "MEC deployments in 4G and evolution towards 5G," ETSI, France, U.K., White Paper 24, 2018, pp. 1–24.

[10] K. Sami *et al.*, "MEC in 5G network," ETSI, France, U.K., White Paper 28, 2018, pp. 1–28.

[11] W. Hu *et al.*, "Quantifying the impact of edge computing on mobile applications," in *Proc. 7th ACM SIGOPS Asia–Pacific Workshop Syst.*, 2016, p. 5.

[12] Y. Gao, W. Hu, K. Ha, B. Amos, P. Pillai, and M. Satyanarayanan, "Are cloudlets necessary?" School Comput. Sci., Carnegie Mellon Univ., Pittsburgh, PA, USA, Tech. Rep. CMU-CS-15-139, 2015.

[13] M. Agiwal, A. Roy, and N. Saxena, "Next generation 5G wireless networks: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 3, pp. 1617–1655, 3rd Quart., 2016.

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

[15] A. Mehta, W. Tärneberg, C. Klein, J. Tordsson, M. Kihl, and E. Elmroth, "How beneficial are intermediate layer data centers in mobile edge networks?" in *Proc. IEEE 1st Int. Workshops Found. Appl. Self Syst.*, Sep. 2016, pp. 222–229.

[16] IQiYi. *Corporate Information*. Accessed: Dec. 27, 2018. [Online]. Available: https://www.iqiyi.com/common/20100420/n4813.html

[17] J. Kwak, Y. Kim, L. B. Le, and S. Chong, "Hybrid content caching for low end-to-end latency in cloud-based wireless networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.

[18] Z. Zhao, M. Peng, Z. Ding, W. Wang, and H. V. Poor, "Cluster content caching: An energy-efficient approach to improve quality of service in cloud radio access networks," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 5, pp. 1207–1221, May 2016.

[19] A. Gharaibeh, A. Khreishah, B. Ji, and M. Ayyash, "A provably efficient online collaborative caching algorithm for multicell-coordinated systems," *IEEE Trans. Mobile Comput.*, vol. 15, no. 8, pp. 1863–1876, Aug. 2016.

[20] C. Mouradian, D. Naboulsi, S. Yangui, R. H. Glitho, M. J. Morrow, and P. A. Polakos, "A comprehensive survey on fog computing: State-of-the-art and research challenges," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 1, pp. 416–464, 1st Quart., 2017.

[21] S.-C. Hung, H. Hsu, S.-Y. Lien, and K.-C. Chen, "Architecture harmonization between cloud radio access networks and fog networks," *IEEE Access*, vol. 3, pp. 3019–3034, 2015.

[22] H. Xiang, M. Peng, Y. Cheng, and H.-H. Chen, "Joint mode selection and resource allocation for downlink fog radio access networks supported D2D," in *Proc. 11th Int. Conf. Heterogeneous Netw. Qual., Rel., Secur. Robustness (QSHINE)*, Aug. 2015, pp. 177–182.

[23] C. T. Do, N. H. Tran, C. Pham, M. G. R. Alam, J. H. Son, and C. S. Hong, "A proximal algorithm for joint resource allocation and minimizing carbon footprint in geo-distributed fog computing," in *Proc. Int. Conf. Inf. Netw. (ICOIN)*, Jan. 2015, pp. 324–329.

[24] "Toward 5G C-RAN: Requirements, architecture and challenges," China Mobile Res. Inst., Beijing, China, White Paper, Nov. 2016.

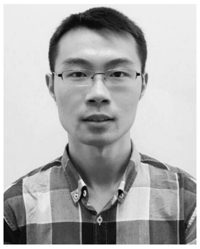
[25] AMPL & Gurobi. (2015). *Linear Programming Optimization Software Package*. [Online]. Available: http://www.gurobi.com

[26] AliCloud. *Alibaba Cloud: Reliable & Secure Cloud Solutions to Empower Your Global Business*. Accessed: Dec. 27, 2018. [Online]. Available: https://www.aliyun.com

[27] D. Wang, W. Zhou, and Y. Liu, "Research on content popularity and its inequality on CDN," *Comput. Eng. Appl.*, vol. 47, no. 6, pp. 102–104, Feb. 2011.

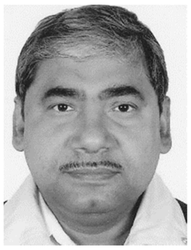
[28] J. Choi, A. S. Reaz, and B. Mukherjee, "A survey of user behavior in VoD service and bandwidth-saving multicast streaming schemes," *IEEE Commun. Surveys Tuts.*, vol. 14, no. 1, pp. 156–169, 1st Quart., 2012.

[29] N. Laoutaris. (2017). "A closed-form method for LRU replacement under generalized power-law demand." [Online]. Available: https://arxiv.org/abs/0705.1970



NING WANG received the B.Sc. degree from the Suzhou University of Science and Technology, China, in 2016. He is currently pursuing the master's degree with the School of Electronic and Information Engineering, Soochow University, China. He was an Assistant Researcher with the Suzhou Key Laboratory of Mobile Networking and Applied Technologies. He was also an R&D Engineer at Microsoft. His current research interest is in the area of broadband and access network.

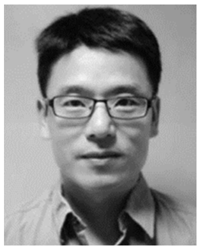
GANGXIANG SHEN (S'98–M'99–SM'12) received the B.Eng. degree from Zhejiang University, China, the M.Sc. degree from Nanyang Technological University, Singapore, and the Ph.D. degree from the University of Alberta, Canada, in 2006. He was a Lead Engineer with Ciena, Linthicum, MD, USA. He was also an Australian ARC Post-Doctoral Fellow with the University of Melbourne. He is currently a Distinguished Professor with the School of Electronic and Information Engineering, Soochow University, China. He has authored and co-authored over 150 peer-reviewed technical papers, among which one of the papers received the highest citations among all the papers published in the IEEE/OSA JOCN. He is a Highly-Cited Chinese Research Scholar selected by Elsevier, from 2014 to 2017, and an Excellent Young Research Scholar sponsored by NSFC. His research interests include integrated optical and wireless networks, spectrum efficient optical networks, and green optical networks. He received the Young Researcher New Star Scientist Award in the 2010 Scopus Young Researcher Award Scheme, China. He was a recipient of the Izaak Walton Killam Memorial Award from the University of Alberta and the Canadian NSERC Industrial Research and Development Fellowship. He was a Secretary for the IEEE Fiber-Wireless Integration Sub-Technical Committee. He is serving as a Voting Member of the IEEE ComSoc Strategic Planning Standing Committee, since 2018, and an IEEE ComSoc Distinguished Lecturer from 2018 to 2019. He has served as TCP chairs for various international conferences in the area of optical networking, including the Symposium Lead Chair of GLOBECOM 2017 and the General TPC Co-Chair of ACP 2018. He was a Lead Guest Editor of the IEEE JSAC Special Issue on Next-Generation Spectrum-Efficient and Elastic Optical Transport Networks and a Guest Editor of the IEEE JSAC Special Issue on Energy-Efficiency in Optical Networks. He is an Associate Editor of the IEEE/OSA JOCN and an Editorial Board Member of Optical Switching and Networking and Photonic Network Communications.



SANJAY KUMAR BOSE (SM'91) received the B.Tech. degree from IIT Kanpur, in 1976, and the master's and Ph.D. degrees from SUNY Stony Brook, USA, in 1977 and 1980, respectively. He was with the Corporate Research and Development Centre, General Electric Company, Schenctady, NY, USA, until 1982. He joined IIT Kanpur as an Assistant Professor and became a Professor, in 1991. He left IIT Kanpur in 2003 to join the Faculty of the School of EEE, NTU, Singapore. In 2008, he left NTU to join IIT Guwahati, where he is currently a Professor with the Department of EEE. He has concurrently held the position of Dean, Alumni Affairs, and External Relations in IIT Guwahati from 2011 to 2014.

He has been involved in various areas in the field of computer networks and queuing systems for the past three decades. He has published extensively in the areas of optical networks, network routing, modeling and analysis of networks and queuing systems and wireless networking. He has also authored various popular text books in the areas of queuing systems, microprocessor systems, and hardware and software of personal computers. He hosts an extremely popular Web site on instructional material for queuing systems.

Dr. Bose is a Fellow of IETE, India, and a member of Sigma Xi and Eta Kappa Nu.



WEIDONG SHAO received the Ph.D. degree in electrical science and technology from the Shanghai Institute of Technical Physics, Chinese Academy of Sciences, Beijing, China, in 2002. Since 2002, he has been with Soochow University, Suzhou, China. His research interests include remote sensing, optoelectronic information processing, and green optical networking.