


RESEARCH

Open Access

# ECDU: an edge content delivery and update framework in Mobile edge computing



Chuntao Ding<sup>\*†</sup> , Ao Zhou<sup>†</sup>, Jie Huang, Ying Liu and Shangguang Wang

## Abstract

Content delivery network (CDN) has gained increasing popularity in recent years for facilitating content delivery. Most existing CDN-based works upload the content generated by mobile users to the cloud data center firstly. Then, the cloud data center delivers the content to the proxy server. Finally, the mobile users request the required content from the proxy server. However, uploading all the collected content to the cloud data center increases the pressure on the core network. In addition, it also wastes a lot of bandwidth resources because most of the content does not have to be uploaded. To make up for the shortcomings of existing CDN-based works, this article proposes an edge content delivery and update (ECDU) framework based on mobile edge computing architecture. In the ECDU framework, we deploy a number of content servers to store raw content collected from mobile users, and cache pools to store content that frequently requested at the edge of the network. Thus, it is not necessary to upload all content collected by mobile users to the cloud data center, thereby alleviating the pressure of the core network. Based on content popularity and cache pool ranking, we also propose edge content delivery (ECD) and edge content update (ECU) schemes. The ECD scheme is to deliver content from cloud data center to cache pool, and the ECU scheme is to mitigate the content to appropriate cache pools in terms of its request frequency and cache pool ranking. Finally, a representative case study is provided and several open research issues are discussed.

**Keywords:** Mobile edge computing, Edge content delivery, Edge content update, Edge server, Cache pool, Content server

## 1 Introduction

Uploading contents collected by mobile users to the cloud data center is a straightforward way to address the tension between resource-constrained mobile devices and resource-intensive tasks. With the proliferation of mobile devices, a large number of devices are connected and generate huge amounts of data traffic. According to Cisco Visual Networking index, mobile devices and connections will grow up to 11.6 billion by 2021, and the share of smart devices and connections will also increase from 46% in 2016 to 75% in 2021 [1]. Therefore, uploading all contents to the remote cloud data center will consume a huge amount of network bandwidth [2].

To compensate for these shortcomings, researchers from both academic and industry are looking to push contents and infrastructures to the vicinity of mobile users to alleviate the pressure on the core network and improve the quality of experience (QoE) of mobile users [3–8]. The content delivery network (CDN) [9–13] is proposed to improve the QoE of mobile users. The CDN is a network of edge servers strategically placed across the globe with the purpose of delivering contents to mobile users as fast as possible. Using the CDN is one of the most effective ways to deliver contents with high performance and reliability. In the CDN framework, mobile users upload all contents directly to the cloud data center. When a mobile user requests a content, the mobile user first detects whether there is the requested content on the proxy server that is close to the mobile user. If there is no required content on the proxy server, the proxy server sends a request to the cloud data center to detect whether there is

\*Correspondence: [ctding@bupt.edu.cn](mailto:ctding@bupt.edu.cn)

<sup>†</sup>Chuntao Ding and Ao Zhou contributed equally to this work.

<sup>1</sup>The State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Xitucheng Road, 100876 Beijing, China

a required content. If yes, the cloud data center transmits the required content to the proxy server, and the proxy server transmits the content to the mobile user. Otherwise, the mobile user requests content service failed. In the CDN framework, the mobile user obtains the required content from the proxy servers rather than the cloud data center, which can reduce the load on the core network and improve the QoE of mobile users to some extent. However, the CDN framework uploads all contents to the cloud data center first consumes a considerable amount of bandwidth of the core network. In real-world scenarios, mobile users have the dual role of content providers and consumers. In addition, most contents do not have to be uploaded to the cloud data center because only a small portion of contents is frequently requested.

Mobile edge computing (MEC) [14–21] provides cloud capabilities in close proximity to mobile users to make up for the shortcomings of CDN. Driven by the potential capabilities of the MEC architecture, we propose an edge content delivery and update (ECDU) framework. In the ECDU framework, contents are not uploaded to the cloud data center first, but uploaded to the edge server first. Thus, the load on the core network will be significantly reduced. The ECDU framework consists of two schemes, an edge content delivery (ECD) scheme and an edge content update (ECU) scheme. The ECD scheme prioritizes the top-ranking cache pools to store higher priority contents. Note that, the priority of contents is defined according to the number of times the user requests access to the contents within a certain period of time. The ECU scheme is to upload popular contents to different levels of cache pools and the cloud data center. Note that, popular contents refers to contents that are frequently requested by mobile users.

## 2 Methods

The ECDU framework consists of three layers: the end layer, the edge layer and cloud layer, and two schemes: the ECD scheme and the ECU scheme. Note that, the ECD and ECU schemes are performed in a collaborative manner. In the following, we discuss the ECDU framework in detail from three aspects: architecture, ECD scheme, and ECU scheme.

### 2.1 Architecture

The architecture of the proposed ECDU framework consists of three layers, the end layer, the edge layer, and the cloud layer. The main characteristics and functions of each layer are as follows.

#### 2.1.1 End layer

The end layer consists of various mobile devices, such as Google Glass, Apple Watch, Smart Phone, Laptop, and Vehicle. In general, these mobile devices generate

a large number of contents. In addition, they also consume contents. Nevertheless, mobile devices are resource-constrained, such as having limited computing capability and storage capacity. Therefore, it is necessary to upload the collected contents to the servers with powerful computing power and large storage space for computing and storage.

#### 2.1.2 Edge layer

The edge layer is located between the end layer and the cloud layer, and is composed of base stations and edge servers. The role of base stations is to communicate with mobile devices, the cloud data center, and other base stations. In addition, the edge server also consists of content servers and cache pools. Note that, in this article, base station, content server, and cache pool are bound together. In this article, the content server has unlimited storage capacity and is used to store all the raw contents uploaded from mobile users. The cache pool has limited storage capacity and is used to store the contents that are frequently requested by mobile users. For example, if a mobile user uploads a video, the mobile user first uploads the video to the content server through LTE or WiFi connection. If the video is frequently requested, the ECDU framework will migrate it to the cache pool according to the ECU scheme. Otherwise, the video will be stored on the content server.

#### 2.1.3 Cloud layer

In the cloud layer, the cloud data center is considered to have unlimited computation capability and unlimited storage capacity. However, uploading all the collected contents from mobile devices to the cloud data center consumes a lot of bandwidth and can even cause network congestion. In order to reduce the load on the core network and network transmission delays, it is necessary to avoid excessive content uploading. In addition, according to the Pareto principle [22, 23], the majority of contents are not needed to be uploaded to the cloud data center because mobile users often only request a small number of contents. In the ECDU framework, the cloud data center only stores and delivers popular contents.

## 2.2 Edge content delivery scheme

The edge content delivery (ECD) scheme is used to efficiently deliver popular contents to appropriate cache pools and works as follows.

### 2.2.1 Initialization

We assume that all contents are stored in the cloud data center, that no contents have been deployed in cache pools, and that all stations are fixed in location. The base stations are considered as vertices, the communication costs between base stations are represented by weights

between vertices. All base stations form a weighted complete graph. In addition, it is forbidden to copy contents in the initial stage.

**2.2.2 Where to deliver**

The Floyd-Warshall algorithm is used to search for the best solution because it can be used to search for the shortest path between all pairs of vertices. By operating the Floyd-Warshall algorithm, we can obtain a list sorted by cache pool ranking. The first  $K$  cache pools are called top- $K$  cache pools. Note that, the top- $K$  cache pools are the  $K$  cache pools that provide the fastest response for mobile users. The higher the ranking is, the better the location of the cache pool is.

**2.2.3 Which to deliver**

Because all contents are stored in the cloud data center during the initial stage, the cloud data center can know how often the content is requested.

For simplicity, we assign the priority of the content based on the frequency it is requests (e.g., within a certain time range, such as within six months). The more the number of requests is, the higher the priority is. Here, we prioritize the delivery of higher priority contents to cache pools.

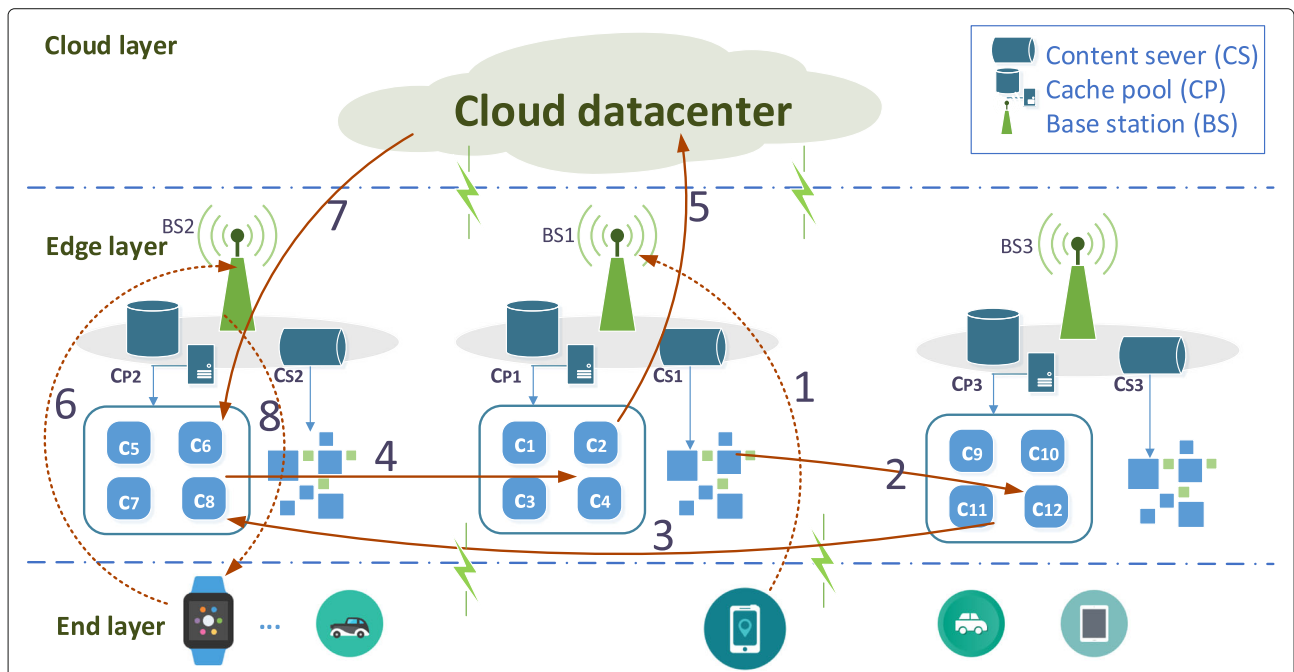
**2.2.4 Delivery strategy**

To better deliver contents, we chose the top- $K$  cache pools as the key cache pools for storing popular contents. For

the top- $K$  cache pools, we first deliver the content with the highest priority to the optimal cache pool and then the content with the second highest priority, until the optimal cache pool is full. The optimal cache pool refers to the cache pool that provides the fastest response for mobile users. The suboptimal cache pool refers to the cache pool that can provide the fastest response for mobile users except for the optimal cache pool. Then, we deliver the content to the suboptimal cache pool, and so on, until all the top- $K$  cache pools are full. The remaining cache pools follow the same way. Note that, to facilitate the content updates, we only use about  $\varepsilon$  times of the storage capacity of the remaining cache pools to store contents.

**2.3 Edge content update scheme**

The edge content update (ECU) scheme is to update the content in the cache pools and cloud data center. For instance, the raw content stored in content servers can be mitigated to cache pools if it becomes popular (e.g., frequently requested). The content stored in cache pools can only be mitigated to different ranked cache pools and the cloud data center, according to the time-varying content popularity. To clearly describe the update process, we give an example to illustrate it, as shown in Fig. 1. In Fig. 1,  $C_{P1}$ ,  $C_{P2}$ , and  $C_{P3}$  represent the cache pools, and the ranking of them is  $C_{P1} > C_{P2} > C_{P3}$ , and  $C_{P1}$  is the top- $K$  cache pool (here,  $K = 1$ ).  $BS1$ ,  $BS2$ , and  $BS3$  represent base stations,  $C_{S1}$ ,  $C_{S2}$ , and  $C_{S3}$  are content servers,  $c_1, c_2, \dots, c_{12}$



**Fig. 1** Edge content update process. The detailed process is as follows: (1) a mobile user uploads a content such as  $c_{14}$  to  $C_{S1}$  through  $BS1$ ; (2)  $C_{S1}$  copies a copy of  $c_{14}$  and migrates it to  $C_{P3}$ ; (3)  $C_{P3}$  migrates  $c_{14}$  to  $C_{P2}$ ; (4)  $C_{P2}$  migrates  $c_{14}$  to  $C_{P1}$ ; (5)  $C_{P1}$  uploads  $c_{14}$  to the cloud data center; (6) other mobile users request  $c_{14}$  from  $C_{P2}$  through  $BS2$ ; (7) the cloud data center delivers  $c_{14}$  to  $C_{P2}$ ; (8)  $C_{P2}$  provides  $c_{14}$  service to mobile users

represent the content,  $r_{c_1}, r_{c_2}, \dots, r_{c_{12}}$  represent the number of times the content have been requested, and  $r_{c_1} > r_{c_2} > \dots > r_{c_{12}}$ . The content on the top- $K$  cache pools is popular content. Here,  $c_1, c_2, c_3, c_4$  are popular content. We consider the following three situations:

### 2.3.1 The content on cache pools can satisfy the user's request

If the content requested by a mobile user is cached in the cache pool, the cache pool will transmit the content to the mobile user. Suppose that the request frequency of a content such as  $c_{10}$  keep increasing, the edge content update scheme will be performed as follows:

**Case 1** *With the number of times  $c_{10}$  has been requested increased until  $r_{c_{10}}$  more than  $r_{c_8}$   $\delta$  times, where  $\delta$  is a constant,  $C_{P_3}$  sends a request to  $C_{P_2}$  to check whether  $C_{P_2}$  has enough storage capacity for  $c_{10}$ . If yes,  $C_{P_3}$  migrates  $c_{10}$  to  $C_{P_2}$ . Otherwise,  $C_{P_2}$  migrates the least requested content, i.e.,  $c_8$  or even  $c_7$  to  $C_{P_3}$  for the storage of  $c_{10}$ . Simultaneously,  $C_{P_2}$  sends a request to check whether  $C_{P_3}$  has enough storage capacity for  $c_8$ . If yes,  $C_{P_2}$  migrates  $c_8$  to  $C_{P_3}$ . If not,  $C_{P_3}$  deletes the least requested content  $c_{12}$  or even  $c_{11}$ .*

**Case 2** *For the content stored in top- $K$  server  $C_{P_1}$ ,  $C_{P_1}$  counts the number of requested for the content through each cache pool. If the number of requested of  $c_2$  through BS2 more than  $\varepsilon$  times of the total number of requested and  $C_{P_2}$  has enough storage capacity for  $c_2$ , cloud data center delivers  $c_2$  to  $C_{P_2}$  directly. Otherwise,  $C_{P_2}$  has not enough storage capacity currently, similar to case 1,  $C_{P_2}$  migrates  $c_8$  or even  $c_7$  to  $C_{P_3}$ .*

### 2.3.2 The content on cache pools cannot satisfy the user's request

When a mobile user requests a new content, e.g.,  $c_{13}$ , which is not stored in the cache pool, the cloud data center checks whether the cache pool with the lowest ranking (i.e.,  $C_{P_3}$ ) has enough storage capacity for  $c_{13}$ . If yes, the cloud data center delivers  $c_{13}$  to  $C_{P_3}$ . Otherwise,  $C_{P_3}$  deletes the least requested content  $c_{12}$  or even  $c_{11}$ , until  $C_{P_3}$  has enough storage capacity. After that, the cloud data center delivers  $c_{13}$  to  $C_{P_3}$ .

### 2.3.3 User requests to upload a content

When a mobile user uploads a content, such as  $c_{14}$ , the mobile user uploads  $c_{14}$  to the content server  $C_{S_1}$ , and the cache pool  $C_{P_1}$  adds the description of  $c_{14}$ . Thus, the mobile user requests the content by the description of the content stored in  $C_{P_1}$ . If the request frequency  $r_{c_{14}}$  is greater than  $r_{c_{12}}$  by  $\delta$  times and  $C_{P_3}$  has enough storage capacity,  $C_{S_1}$  copies a copy of  $c_{14}$  and migrates it to  $C_{P_3}$ . If there is no more storage capacity for  $c_{14}$ ,  $C_{P_3}$  deletes  $c_{12}$  or

even  $c_{11}$ . If and only if  $c_{14}$  is migrated to  $C_{P_1}$ ,  $C_{P_1}$  copies a copy of  $c_{14}$  and uploads it to the cloud data center.

## 3 Results

To clearly illustrate the ECDU framework, Fig. 2 gives an example of an analysis of YouTube, since YouTube is the largest video site, and many mobile devices download, watch, and share videos from YouTube every day. In Fig. 2, A–E represent the base station, and the numbers on the lines are the cost (response time or bandwidth of the core network) of communication with each other. Note that, since the base station, content server and cache pool are bound together, that A–E can represent them in the context. We assume that there are 1000 videos stored in the cloud data center,  $v_1, v_2, \dots, v_{1000}$ , respectively.

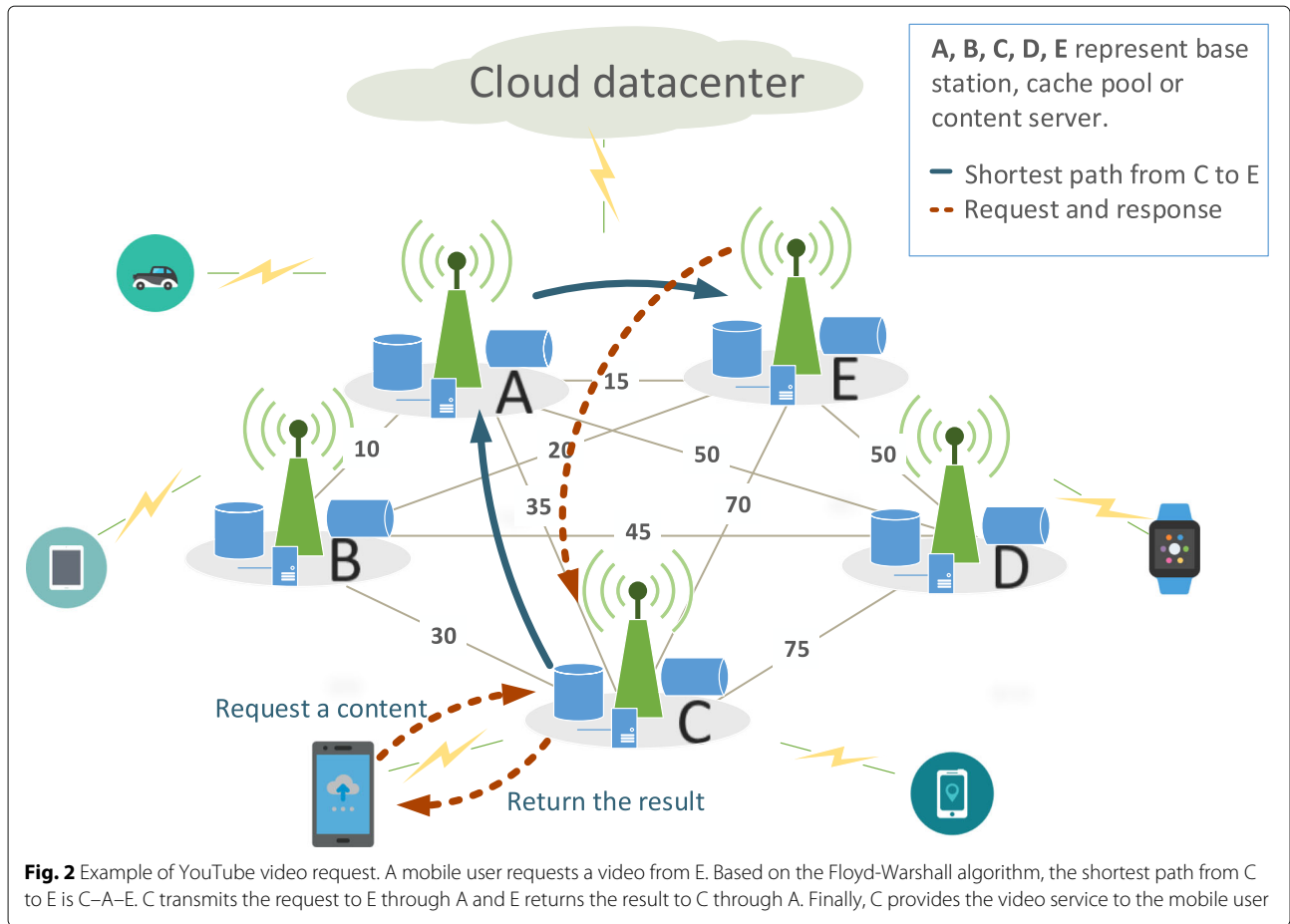
Based on the ECDU scheme, we first determine where to deliver the videos from the cloud data center. The Floyd-Warshall algorithm is used to find the ranking of each base station based on the principle of the smaller the cost is, the higher ranking is expected. Table 1 shows the ranking result, and the ranking order is B, A, E, C, D.

Then, based on the priority of videos, we decide which to deliver, by the number of times that the video is requested with a certain time range. The priority of those videos is  $v_1 > v_2 > \dots > v_{1000}$ . In the case study, we assume each cache pool and proxy server can store 10 videos and  $\delta=10\%$ ,  $\varepsilon=1/3$ .

From Table 1, the top- $K$  cache pools (e.g.,  $K = 2$ ) are B and A. Therefore, the cache pool B stores the top 10 videos, and the cache pool A stores the videos with the priority between 10 and 20. E, C, and D only store three videos. In summary, in the initial delivery phrase,  $v_1, \dots, v_{10}$  are delivered to B,  $v_{11}, \dots, v_{20}$  are delivered to A,  $v_{21}, v_{22}, v_{23}$  are delivered to E,  $v_{24}, v_{25}, v_{26}$  are delivered to C,  $v_{27}, v_{28}, v_{29}$  are delivered to D.

If the requested number of  $v_{25}$  in C increased over time, C migrates  $v_{25}$  to E when  $v_{25}$  is 10% more than  $v_{23}$  in E according to our update scheme. Supposing that E has no more storage capacity for  $v_{25}$ , E migrates  $v_{23}$  to C at the same time. We believe that videos in B and A are popular content. Thus, as the number of requests to a popular content  $v_{11}$  through C reaches 1/3 of total requests, the cloud data center delivers  $v_{11}$  to C. Specifically, if a mobile user requests a new video that is not available on these five servers. Then, we search it in the cloud data center and deliver it to D. Additionally, the storage capacity is not enough, D deletes  $v_{29}$  directly.

Furthermore, suppose that a mobile user under the coverage of E requests to upload a video. For the ECDU framework, the mobile user uploads it to the content server firstly. Other mobile users are allowed to request the video only with the permission of cache pool E. If the number of requests to the video more than 10% of  $v_{29}$  in D, E copies the video and migrates it to D. D deletes  $v_{29}$  if



its storage capacity is not enough. If and only if the video is migrated to A or B, which can be uploaded to the cloud data center.

Assume the distance between mobile users and the cloud data center is 1000. In the CDN framework, the distance between the proxy servers and the mobile users, as well as the distance between the cloud data center and the proxy servers are 500. While in the ECDU framework, the distance between the edge server and the mobile users is 100, and 900 for the distance between the edge server and the cloud data center. Suppose that there are 10 requests for a new video, respectively. The CDN framework delivers these 10 new videos to the proxy servers who requests it. Suppose A, B, C, D, and E store two videos, respectively. The ECDU framework delivers these 10 new videos to D, without effect to those popular content. The CDN framework does not differ from our framework in terms of cost at delivery, both of them are  $1000 \times 10$ . However, after the delivery to the proxy servers, when there are other requests for these 10 videos from mobile users, the cost of CDN is  $(110 + 105 + 190 + 235 + 135) \times 2 \times 500$ . For the ECDU framework, the cost is  $235 \times 10 \times 100$  in the worst case, less than 69.67% of CDN as well as

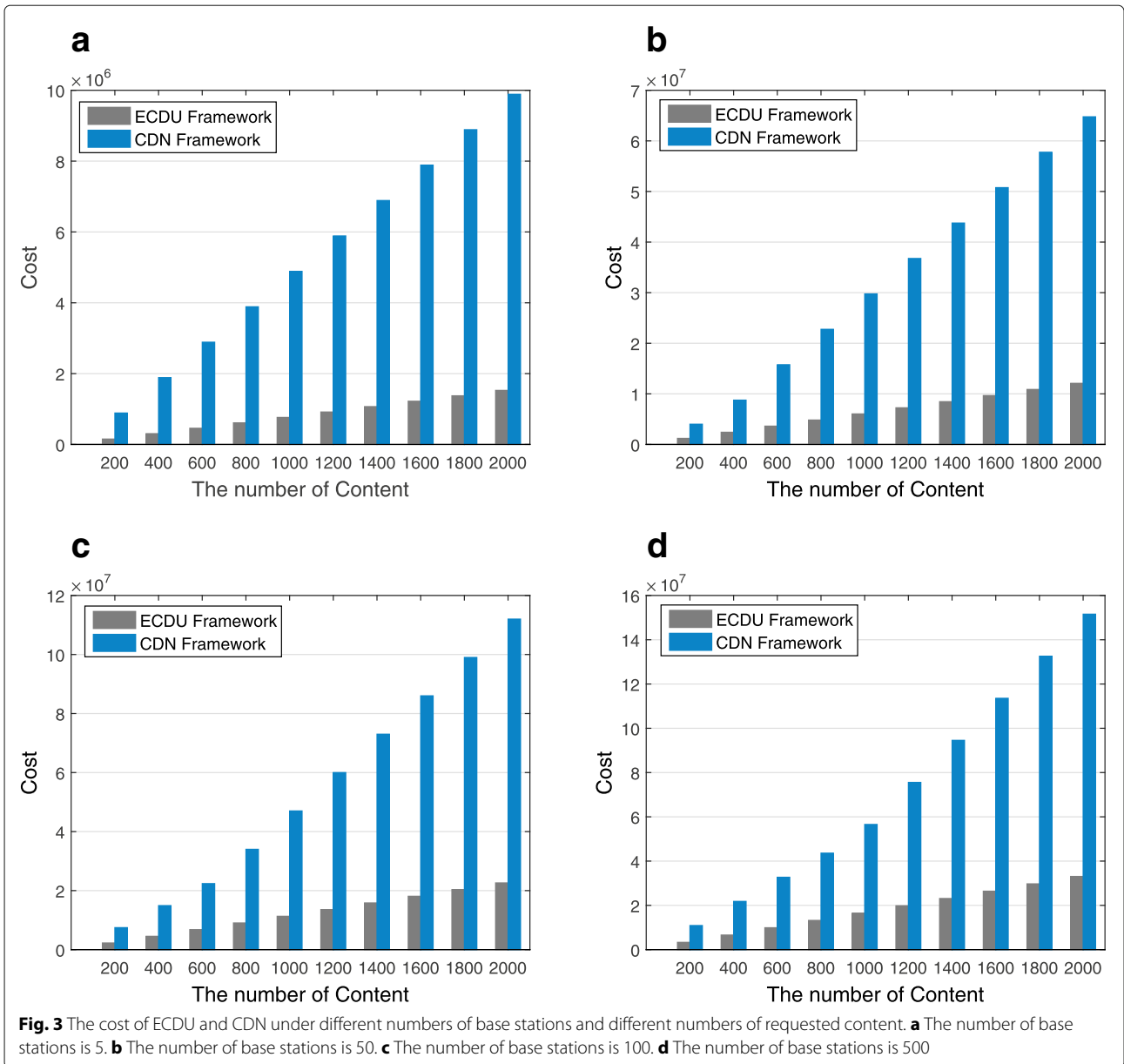
$105 \times 10 \times 100$  for the best case, which is about 86.45% less than CDN. In summary, the cost of the ECDU framework saves about 73.03% in average, compared with CDN. In addition, when a mobile device requests upload a new video, which is uploaded to the cloud data center directly in the CDN framework. Its cost is  $1000 \times 100\%$ . According to the Pareto principle, the possibility of uploading to the cloud data center is only 20%. The price is  $100 \times 100\% + 900 \times 20\%$  in the ECDU framework. In this way, the ECDU framework saves 72% of the cost.

Figure 3 shows the cost of ECDU and CDN frameworks at different numbers of base stations and different numbers of requested content. From Fig. 3, we can see that the cost of the ECDU framework is lower than CDN

**Table 1** The ranking of five content servers

Server	A	B	C	D	E	Total cost	Ranking
A	0	10	35	50	15	110	2
B	10	0	30	45	20	105	1
C	35	30	0	75	50	190	4
D	65	45	75	0	50	235	5
E	15	20	50	50	0	135	3





framework. The main reason is that in the ECDU framework, all the content stored in the content servers and mobile devices do not need to request content from the cloud data center, which can save a lot of cost. In the CDN framework, all the content is stored in the cloud data center, and when the requested content does not stored in the proxy servers, mobile users have to request the content from the cloud data center, which consumes a lot of cost. Because mobile devices are far away from the cloud data center. From Fig. 3, it can be seen that the cost of the ECDU framework is always lower than the CDN framework in different situations, which indicates that the ECDU framework can reduce the load on the core network and response time.

#### 4 Discussion

We summarize the comparison between ECDU and CDN frameworks in Table 2. Following [15, 24], we explain three typical differences between ECDU and CDN frameworks. Note that, the ECDU is not a substitute for the CDN, they can be combined to provide better service to mobile users.

##### 4.1 Distance to mobile users

The communication delay can be significantly reduced. This is because the distance between edge servers and mobile users is less than the distance between proxy servers and mobile users. In the ECDU framework, edge servers are deployed near mobile users. In general, the

**Table 2** Comparison of features: ECDU vs. CDN

	ECDU	CDN
Distance to mobile users	Small (tens to hundreds of meters)	Large (tens of kilometers)
Upload strategy	Upload all the captured content to the content server and upload popular content to the cloud data center	Upload all the captured content to the cloud data center directly
The load on the core network	Light load	Heavy load
Delivery strategy	Deliver popular content to the cache pools	Deliver all the requested content to the proxy servers
Peer communication	Autonomic communication with data transmission	Without autonomic communication
Back-haul usage	Infrequent use (alleviate congestion)	Frequent use (can cause congestion)
Collaborative Decision Making	Yes	N/A

distance between edge servers and mobile users is tens to hundreds of meters (e.g., one hop). However, in the CDN framework, proxy servers are typically deployed in multiple locations on multiple backbones. The distance from proxy servers to mobile users is tens of kilometers. Therefore, the ECDU framework can provide mobile users with lower communication latency services, which can significantly improve the QoE of mobile users.

#### 4.2 Collaborative decision-making

In the ECDU framework, in addition to storage, the cache pool has computing capabilities and can perform collaborative decision. In the CDN framework, when, where, and what to deliver are all determined by the cloud data center [25–27]. In addition, there is no autonomic cooperation between proxy servers. The tasks of the proxy servers are limited to storing and transferring under the order of the cloud data center. In contrast, cache pools have various decision capabilities. The communication between cache pools is maintained by themselves, not by the cloud data center. The advantage of collaborative decision-making is that the cache pool can get the requested times of all the content stored in other cache pools. Cache pools can cooperate with each other and facilitate to decide when, where and which content to migrate.

#### 4.3 Upload strategy

In the ECDU framework, we deploy a content server and a cache pool for each base station. Thus, mobile users only need to upload the collected contents to the content server instead of the cloud data center, which can reduce the load on the core network. In reality, only a small number of contents need to be uploaded to the cloud data center. In the ECDU framework, we design an ECU scheme to further reduce the load on the core network, rather than directly uploading the large number of contents directly to the cloud data center. In contrast, content providers upload the collected contents directly to the cloud data center in the CDN. According to the Pareto principle,

most contents may not be frequently requested by mobile users.

### 5 New challenges

The proposed ECDU framework brings various benefits, such as reducing the load on the core network and improving the QoE for mobile users, while introducing the following new challenges.

#### 5.1 Network Integration and coordination

Under various potential deployment scenarios over multiple RANs (e.g., WLAN, LTE), the integration of the MEC network should be emphasized at both the architectural and protocol levels. In addition, the collaboration between mobile devices and back-haul segments of converged networks in 5G is also a vital issue.

#### 5.2 Resource management

In practice, the computing and storage resources of cache pools are limited and can only support a limited number of contents, and the limited resources should be allocated to meet dynamic needs of mobile devices. Therefore, it is a challenge to design a reasonable resource management scheme for the network with high QoE.

#### 5.3 Cloud data center and edge servers coexistence

In order to support a more diverse set of emerging services in the 5G network, the cloud data center and edge servers should coexist and complement each other. However, some parts of the service may be executed on the mobile device itself, on the edge server, or in the cloud data center. Given the available infrastructure and resource requirement of the service, identifying which part of the service to offload to edge server/cloud data center is a critical task. Further research is required to find intelligent strategies for coexistent cloud data center and edge server systems under realistic network conditions and situations.

#### 5.4 Security and privacy

In the ECDU framework, mobile devices upload the collected contents, which commonly contain sensitive and

private information (e.g., personal clinical data and business financial records) to the cache pool rather than the cloud data center. Therefore, these contents should be properly preprocessed in the cache pools before being migrated between different cache pools or being uploaded to the cloud data center.

## 6 Conclusion

In this article, we propose an ECDU framework that reduces the load on the core network and improves the QoE of mobile users. In the ECDU framework, a content server and a cache pool are introduced at the edge of the network for storing raw contents and popular contents collected by mobile users. As a result, mobile users upload all contents to the content server instead of the cloud data center, which reduces the load on the core network. In addition, we propose an ECD scheme that prioritizes the top-ranking cache pools for higher priority content to reduce the response time. We also propose an ECU scheme that updates the content in the cache pool and cloud data center, based on the time-varying content polarity. The case study demonstrates that the ECDU framework reduces the load on the core network and improves the QoE for mobile users and significantly reduces costs.

### Abbreviations

CDN: Content delivery network; ECD: Edge content delivery; ECDU: Edge content delivery and update; ECU: Edge content update; MEC: Mobile edge computing; QoE: Quality of experience

### Acknowledgements

Not applicable.

### Authors' contributions

CT conceived this study, designed and performed the simulations, analyzed the results, and wrote the manuscript; others participated in the discussion and proofreading work and were in charge of part of the analysis and experimentation. All authors read and approved the final manuscript.

### Funding

This work was supported in part by the National Natural Science Foundation of China (Grant No.61602054), and BUPT Excellent Ph. D. Students Foundation (Grant No.CX2018207).

### Availability of data and materials

Not applicable.

### Competing interests

The authors declare that they have no competing interests.

Received: 27 August 2019 Accepted: 30 October 2019

Published online: 12 December 2019

### References

1. Cisco, Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update 2017-2022. [https://learningnetwork.cisco.com/community/learning\\_center/white-papers](https://learningnetwork.cisco.com/community/learning_center/white-papers)
2. A. Zhou, S. Wang, C.-H. Hsu, M. Hsu, K.-s. Wong, Virtual machine placement with (m,n)-fault tolerance in cloud data center. *Clust. Comput.* **1**–13 (2017)
3. F. Dobrian, V. Sekar, A. Awan, I. Stoica, D. Joseph, A. Ganjam, J. Zhan, H. Zhang, in *Proceedings of the ACM SIGCOMM 2011 Conference*. Understanding the impact of video quality on user engagement, (2011), pp. 362–373. <https://doi.org/10.1145/2018436.2018478>
4. L. Sun, M. Ma, W. Hu, H. Pang, Z. Wang, Beyond 1 million nodes: A crowdsourced video content delivery network. *IEEE MultiMed.* **24**(3), 54–63 (2017)
5. A. Mukherjee, D. De, D. G. Roy, A power and latency aware cloudlet selection strategy for multi-cloudlet environment. *IEEE Trans. Cloud Comput.* **7**(1), 141–154 (2019)
6. M. Jia, J. Cao, W. Liang, Optimal cloudlet placement and user to cloudlet allocation in wireless metropolitan area networks. *IEEE Trans. Cloud Comput.* **5**(4), 725–737 (2017)
7. H. Liu, H. Kou, C. Yan, L. Qi, Link prediction in paper citation network to construct paper collocated graph. *EURASIP J. Wirel. Commun. Netw.* **2019**(1) (2019). <https://doi.org/10.1186/s13638-019-1561-7>
8. L. Qi, Q. He, F. Chen, W. Dou, S. Wan, X. Zhang, X. Xu, Finding all you need: Web apis recommendation in web of things through keywords search. *IEEE Trans. Comput. Soc. Syst.* **6**(5), 1036–1072 (2019)
9. M. Hajimirsadeghi, N. B. Mandayam, A. Reznik, Joint caching and pricing strategies for popular content in information centric networks. *IEEE J. Sel. Areas Commun.* **35**(3), 654–667 (2017)
10. Y. Bang, J.-K. K. Rhee, K. Park, K. Lim, G. Nam, J. D. Shinn, J. Lee, S. Jo, J.-R. Koo, J. Sung, Y.-i. Seo, T. Choi, K.-I. Kim, J. Park, C. H. Yun, Cdn interconnection service trial: implementation and analysis. *IEEE Commun. Mag.* **54**(6), 94–100 (2016)
11. I. Benkacem, T. Taleb, M. Bagaa, H. Flinck, Optimal vnfs placement in cdn slicing over multi-cloud environment. *IEEE J. Sel. Areas Commun.* **36**(3), 616–627 (2018)
12. Q. Fu, B. Rutter, H. Li, P. Zhang, C. Hu, T. Pan, Z. Huang, Y. Hou, Taming the wild: a scalable anycast-based cdn architecture (t-sac). *IEEE J. Sel. Areas Commun.* **36**(12), 2757–2774 (2018)
13. Y. Cui, N. Dai, Z. Lai, M. Li, Z. Li, Y. Hu, K. Ren, Y. Chen, Tailcutter: Wisely cutting tail latency in cloud cdns under cost constraints. *IEEE/ACM Trans. Netw.* **27**(4), 1612–1628 (2019)
14. W. Gong, L. Qi, Y. Xu, Privacy-aware multidimensional mobile service quality prediction and recommendation in distributed fog environment. *Wirel. Commun. Mob. Comput.* **2018**, 1–8 (2018)
15. N. Abbas, Y. Zhang, A. Taherkordi, T. Skeie, Mobile edge computing: a survey. *IEEE Internet Things J.* **5**(1), 450–465 (2018)
16. X. Chen, L. Jiao, W. Li, X. Fu, Efficient multi-user computation offloading for mobile-edge cloud computing. *IEEE/ACM Trans. Netw.* **24**(5), 2795–2808 (2016)
17. T. X. Tran, A. Hajisami, P. Pandey, D. Pompili, Collaborative mobile edge computing in 5g networks: New paradigms, scenarios, and challenges. *IEEE Commun. Mag.* **55**(4), 54–61 (2017)
18. T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, D. Sabella, On multi-access edge computing: a survey of the emerging 5g network edge cloud architecture and orchestration. *IEEE Commun. Surv. Tutorials.* **19**(3), 1657–1681 (2017)
19. M. Li, F. R. Yu, P. Si, Y. Zhang, Energy-efficient machine-to-machine (m2m) communications in virtualized cellular networks with mobile edge computing (mec). *IEEE Trans. Mob. Comput.* **18**(7), 1541–1555 (2019)
20. T. X. Tran, D. Pompili, Adaptive bitrate video caching and processing in mobile-edge computing networks. *IEEE Trans. Mob. Comput.* **18**(9), 1965–1978 (2019)
21. S. Wang, C. Ding, N. Zhang, X. Liu, A. Zhou, J. Cao, X. Shen, A cloud-guided feature extraction approach for image retrieval in mobile edge computing. *IEEE Trans. Mob. Comput.* **8**, 1–14 (2019)
22. Y. Nikulin, M. M. Makela, Stability and accuracy functions for a multicriteria boolean linear programming problem with parameterized principle of optimality from condorcet to pareto. *Eur. J. Oper. Res.* **207**(3), 1497–1505 (2010)
23. S. Cato, Weak independence and the pareto principle. *Soc. Choice Welf.* **47**(2), 295–314 (2016)
24. B. P. Rimal, D. P. Van, M. Maier, Mobile edge computing empowered fiber-wireless access networks in the 5g era. *IEEE Commun. Mag.* **55**(2), 192–200 (2017)
25. P. Medagliani, S. Paris, J. Leguay, L. Maggi, C. Xue, H. Zhou, in *Proceedings of the IFIP/IEEE Symposium on Integrated Network and Service Management*. Overlay routing for fast video transfers in cdn, (2017), pp. 531–536. <https://doi.org/10.23919/inm.2017.7987323>



26. P. A. Frangoudis, L. Yala, A. Ksentini, Cdn-as-a-service provision over a telecom operator's cloud. *IEEE Trans. Netw. Serv. Manag.* **14**(3), 702–716 (2017)
27. M. Wichtlhuber, R. Reinecke, D. Hausheer, An sdn-based cdn/isp collaboration architecture for managing high-volume flows. *IEEE Trans. Netw. Serv. Manag.* **12**(1), 48–60 (2015)

### **Publisher's Note**

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

**Submit your manuscript to a SpringerOpen<sup>®</sup> journal and benefit from:**

- ▶ Convenient online submission
- ▶ Rigorous peer review
- ▶ Open access: articles freely available online
- ▶ High visibility within the field
- ▶ Retaining the copyright to your article

---

Submit your next manuscript at ▶ [springeropen.com](https://www.springeropen.com)

---