Near-Optimal Content Service Algorithm with Procurement and Placement in Edge Networks

Chun-An Yang[§‖], Shih-Chieh Chen[†‖], Yi-Hsuan Peng[†], Yu-Wen Chen[‡], Jian-Jhih Kuo[†*], and Ming-Jer Tsai[§]
[§]Dept. of Computer Science, National Tsing Hua University, Hsinchu, Taiwan
[†]Dept. of Computer Science and Information Engineering, National Chung Cheng University, Chiayi, Taiwan
[‡]Computer Systems Technology, New York City College of Technology, Brooklyn, New York, USA
Emails: s110065505@m110.nthu.edu.tw, sam2003122367@alum.ccu.edu.tw, s1040338@alum.ccu.edu.tw, YWChen@citytech.cuny.edu, lajacky@cs.ccu.edu.tw, mjtsai@cs.nthu.edu.tw

***Abstract*—Telecom carriers have announced new content services by making a partnership with content providers and offered popular video streaming to customers. Via the integration with edge networks, telecom carriers can procure various contents from content providers and place them on edge servers proximate to end users to serve real-time requests with high bandwidth and ultra-low latency. Nevertheless, it is challenging to consider the content procurement, placement, and services jointly due to the user preference, user distribution, storage capacity of edge server, economic costs, etc. Telecom carriers would like to balance the procuring, placing, and transfer costs. To address this problem, the paper formulates an optimization problem and then proposes an approximation algorithm. Finally, the simulation results manifest that our algorithm outperforms other baselines.**

I. Introduction

With the increasing demand of streaming media for the modern people, telecom carriers have not only provided services of telecommunications, wireless communications, Internet, etc. but also announced new content services (e.g., video, reel) to customers by making a partnership with content providers (CPs) or video streaming platforms (e.g., Netflix, YouTube) in recent years [1], [2]. The collaboration between telecom carriers and CPs offers diverse services and popular streaming from telecom carriers, making millions of users easily access their favorite videos for CPs. Via the integration with edge networks, telecom carriers can place various contents on edge servers proximate to end users to serve real-time requests with high bandwidth and ultra-low latency [3]–[6]. Overall, emerging content services make a better lifestyle.

To provide content services to end users, the system architecture can be divided into three layers, including CPs, edge management platform (EMP), and edge servers, as illustrated in Fig. 1. In the upper layer, each competitive CP provides a set of contents with price from its content server to the EMP via backhaul networks. In the middle layer, the EMP determines whether to procure content sets from different CPs and how to place contents on edge servers based on users' requests.[1] In the lower layer, each edge server satisfies content requests of its neighboring end users. In addition, the content request is able to be served by other edge servers via sidehaul networks if there is no requested content in the proximate edge server [7]. Therefore, the system jointly considers the content procurement, placement, and services.

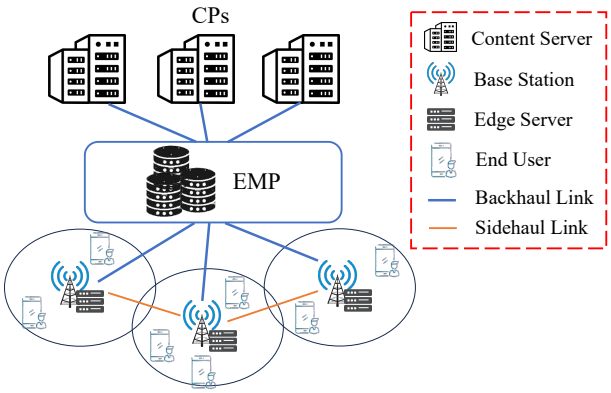


Fig. 1. System architecture.

However, it is difficult to determine all the considerations at the same time due to the user preference, user distribution, storage capacity of edge server, economic costs, etc [8]–[10]. According to the user request, the EMP needs to pay the procuring cost for accessing requested contents and the placing cost for deploying contents varying with different edge servers [5]. Moreover, the transfer costs are considered for wired backhaul links from content servers to edge servers and wireless sidehaul links among edge servers, respectively [7]. Thus, the EMP of the telecom carrier would like to balance the procuring, placing, and transfer costs such that the minimum total cost is taken for content services.

Optimizing the total cost leads to three main challenges: 1) *Content replica placement*. The end users requesting the same content may be located in different places. The number and density of replicas are thus critical. Under a fixed density, the large number of deployed replicas causes a higher placing cost but a lower transfer cost, since end users may be closer to requested contents. In contrast, the placing cost decreases while the transfer cost increases in the small number of replicas. Under a fixed number, a denser (or sparser) replica placement may incur a higher (or lower) transfer cost. 2) *Different content placement*. Due to the limited storage capacity of edge servers,

* indicates the corresponding author; ‖ denotes the equal contributions.

This work was supported by the National Science and Technology Council, Taiwan, under Grants 111-2221-E-007-044-MY3, 111-2628-E-194-001-MY3, and 113-2221-E-194-040-MY3.

[1]The user request can be predicted according to the user preference and distribution [7]–[9]. Thus, we use the expected request to determine the content procurement and placement.

it is difficult to determine where to deploy different contents and consider the placing and transfer costs simultaneously. 3) *Content procurement from different CPs*. Some contents may be owned by several CPs, and thereby a proper way of content procurement can efficiently reduce the procuring and transfer costs. Overall, this problem is complicated. Due to the page limit, related works are summarized in Appendix A of [11].

To address above challenges, we present an optimization problem named Ser**vi**ce Procurement and **De**ployment Pr**o**blem for Edge Networks (VIDEO), which asks for a content procurement and placement method such that all user requests can be satisfied and the total cost is minimized. To solve the VIDEO, we design an approximation algorithm named **Re**quest-satisfied Service **P**rocurement and Dep**l**o**y**ment Algorithm (REPLY), including three phases: 1) Request Group Distribution, 2) Content Placement Selection, and 3) Content Procurement Determination. Finally, the simulation results manifest that the proposed REPLY outperforms other baselines.

II. The Optimization Problem – VIDEO

The Ser**vi**ce Procurement and **De**ployment Pr**o**blem for Edge Networks (VIDEO) asks for the content procurement from content providers and the content placement on edge servers in order to serve end users' requests. The system considers a set of content servers $S = \{s_1, s_2, ..., s_{|S|}\}$ and each content server $s \in S$ provides a set of contents C_s with its price p_s. There are a set of edge servers $B = \{b_1, b_2, ..., b_{|B|}\}$, each of which has the storage capacity $\mathcal{B}_b$ and the content placing cost h_b.[2] The backhaul transfer costs from content servers to the EMP and from the EMP to edge servers are f_s and f_b, respectively, and the sidehaul transfer cost between two edge servers is $f_{b',b}$, where $b, b' \in B$. The units of the two transfer costs are α and β, respectively.[3] There is a set of requested contents $C = \{c_1, c_2, ..., c_{|C|}\}$.[4] The end user requests content c from edge server b, given as $\mathcal{R}_{c,b} \in \{0, 1\}$. If $\mathcal{R}_{c,b} = 0$, there is no request. The goal is to minimize the total cost of the EMP, including the procuring, placing, and transfer costs.

To formulate the problem, we create the following three decision variables. The first one $x_s \in \{0, 1\}$ denotes whether the EMP procures contents from the content server $s \in S$. That is, if the contents provided by the content server $s \in S$ are procured, then $x_s = 1$; otherwise, $x_s = 0$. The second one $y_{c,b} \in \{0, 1\}$ indicates whether the content $c \in C$ is placed on the edge server $b \in B$. If so, then $y_{c,b} = 1$. Otherwise, $y_{c,b} = 0$. The last one $z_{c,b',b} \in \{0, 1\}$ represents whether the content $c \in C$ is transferred from the edge server b' to $b \in B$. Similarly, $z_{c,b',b} = 1$ if the content is transferred, and otherwise $z_{c,b',b} = 0$. Then, the four different costs, including the procuring cost $cost_{proc}$, placing cost $cost_{plac}$, backhaul transfer cost $cost_{back}$, and sidehaul transfer cost $cost_{side}$, can be defined as follows.

$$cost_{proc} = \sum_{s \in S} p_s \cdot x_s \tag{1}$$

$$cost_{plac} = \sum_{b \in B} \sum_{c \in C} h_b \cdot y_{c,b} \tag{2}$$

$$cost_{back} = \alpha(\sum_{s \in S} f_s \cdot x_s + \sum_{b \in B} \sum_{c \in C} f_b \cdot y_{c,b}) \tag{3}$$

$$cost_{side} = \beta \sum_{b \in B} \sum_{b' \in B} \sum_{c \in C} f_{b',b} \cdot z_{c,b',b} \tag{4}$$

Finally, the VIDEO can be formulated as an integer linear programming (ILP) in the following.

$$\min \quad cost_{proc} + cost_{plac} + cost_{back} + cost_{side} \tag{5a}$$

$$\text{s.t.} \quad y_{c,b} \le \sum_{s: c \in C_s} x_s, \qquad \forall c \in C, b \in B \tag{5b}$$

$$z_{c,b',b} \le y_{c,b'}, \qquad \forall c \in C, b, b' \in B \tag{5c}$$

$$\sum_{c \in C} y_{c,b} \le \mathcal{B}_b, \qquad \forall b \in B \tag{5d}$$

$$\sum_{b' \in B} z_{c,b',b} \ge \mathcal{R}_{c,b}, \qquad \forall c \in C, b \in B \tag{5e}$$

$$x_s, y_{c,b}, z_{c,b',b} \in \{0, 1\}, \ \forall s \in S, c \in C, b, b' \in B \tag{5f}$$

The objective function (5a) aims to minimize the total cost. The constraint (5b) makes sure that the content c placed on the edge server b is procured from some content servers s. The constraint (5c) ensures that the content c transferred from the edge server b' to b is placed on the edge server b'. The constraint (5d) limits that the placement cannot exceed the storage capacity of the edge server b. To the end, the constraint (5e) ensures that the request of content c from the edge server b is satisfied.

Note that the VIDEO is NP-hard since the vertex cover problem [13] can be reduced to the VIDEO. Due to the page limit, the proof is provided in Appendix B of [11].

Theorem 1. The VIDEO is NP-hard.

III. Algorithm Design — REPLY

To efficiently solve the VIDEO, we design an approximation algorithm named **Re**quest-satisfied Service **P**rocurement and Dep**l**o**y**ment Algorithm (REPLY), which adopts linear programming (LP) rounding to address the challenges. Thus, we relax the decision variables in the ILP (i.e., LP relaxation), i.e.,

$$x_s, y_{c,b}, z_{c,b',b} \ge 0, \qquad \forall s \in S, c \in C, b, b' \in B, \tag{6}$$

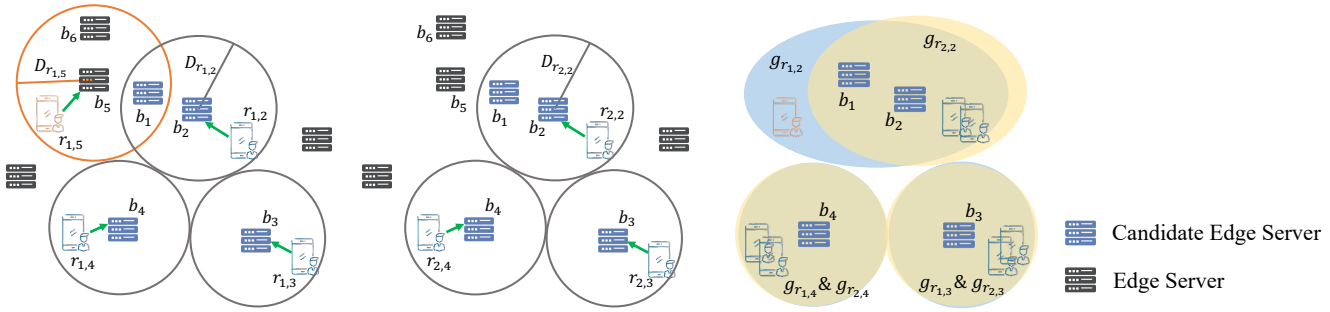
acquire the optimum LP solution by an LP solver (e.g., Gurobi), and then round it to the (integral) solution. For ease of presentation, let $(\tilde{x}, \tilde{y}, \tilde{z})$ denote the optimum LP solution.

Intuitively, letting each user request $r_{c,b}$ be served by the edge server $b' \in B$ with the highest $\tilde{z}_{c,b',b}$ (i.e., content c transferred from edge server b' to b) can provide a high-quality service with the lowest transfer cost. However, it may cause a considerable placing cost since the user requests may be served by diverse edge servers. To balance the placing

[2]The content placing cost can be monetary due to the equipment, computing ability, and storage capacity varying with different edge servers [3].

[3]The transfer costs can be regarded as the transmission time. Specifically, the backhaul transfer via wired communication is cost by data size and data rate, while the sidehaul transfer via wireless communication is cost by data size, data rate, and transmission distance [7]. Note that due to the path loss, the longer the transmission distance is, the lower the data rate is. Thus, the transmission time is proportional to the transmission distance. Finally, the transmission time can be transformed into monetary term [6].

[4]Assume that each requested content has unit size since a video is usually stored into several chunks with the same size [12].



(a) Group distribution for c_1 (b) Group distribution for c_2 (c) Final group distribution

Fig. 2. Example of RGD.

and transfer costs, for each different content, we distribute its user requests into several disjoint groups, identify some candidate edge servers for each group, and select one among the candidates later. To this end, for group distribution, the REPLY introduces a novel notion of *scope* for each user request $r_{c,b}$, which is a coverage centered at edge server b with a radius $D_{r_{c,b}} = \gamma \sum_{b' \in B} f_{c,b',b} \cdot \tilde{z}_{c,b',b}$, where $\gamma > 1$ is a tunable knob to bound the edge server's transfer distance. Then, it iteratively selects the user request $r_{c,b}$ with the lowest $D_{r_{c,b}}$ from those requests that have not been distributed to any group, creates a group $g_{r_{c,b}}$ for $r_{c,b}$, and then adds each user request $r_{u,v}$ into $g_{r_{c,b}}$ if 1) $r_{u,v}$ has not been distributed to any group, 2) $D_{r_{c,b}} \leq D_{r_{u,v}}$, and 3) $r_{c,b}$ and $r_{u,v}$ share an edge server b' in their scopes (i.e., $\exists b', f_{c,b',b} \leq D_{r_{c,b}}$ and $f_{u,b',v} \leq D_{r_{u,v}}$). In this way, all user requests are distributed into disjoint groups to prevent every request from being served by the edge server individually. Then, to ensure the capacity of edge servers, we make *virtual copies* of each candidate edge server and find the minimum perfect matching between request groups and edge servers. Last, we will determine the content procurement by iteratively calculating the *cost-efficiency index (CEI)* for each content set and selecting the one with the lowest CEI.

The proposed REPLY includes three phases: 1) Request Group Distribution (RGD), 2) Content Placement Selection (CPS), and 3) Content Procurement Determination (CPD). Specifically, the RGD distributes each different requested content into several disjoint groups by their scopes and identifies some candidate edge servers in each group. Then, the CPS selects one virtual copy of edge server for each group and assigns all requests in the group to the selected edge server. Finally, the CPD determines the content procurement with CEI. Later, we show that the scope, virtual copy, and CEI are the cornerstones of the REPLY to achieve the approximation ratio.

A. Request Group Distribution (RGD) Phase

After acquiring the optimum LP solution, the RGD sorts the user requests for the same content in a non-decreasing order of the scope radius $D_{r_{c,b}}$ of requests. Then, for each user request $r_{c,b}$ in this order, the RGD creates a group $g_{r_{c,b}}$ and distributes $r_{c,b}$ into $g_{r_{c,b}}$. For each request $r_{u,v}$ that has not been distributed to any group, the RGD distributes $r_{u,v}$ into $g_{r_{c,b}}$ if $D_{r_{c,b}} \leq D_{r_{u,v}}$ and there exists $b' \in B$ such that $f_{c,b',b} \leq D_{r_{c,b}}$ and $f_{u,b',v} \leq D_{r_{u,v}}$ (i.e., $r_{c,b}$ and $r_{u,v}$ share an edge server b' in their scopes). Subsequently, the RGD identifies each edge server $b' \in B$ as the candidate edge server for each group $g_{r_{c,b}}$ if $\tilde{z}_{c,b',b} > 0$ and $D_{r_{c,b}} \geq f_{c,b',b}$. Similarly, the RGD executes the above steps for each different requested content until all different contents finish the group distribution.

TABLE I
EXAMPLE OF RGD.

	$c_1, b_1 \to b_2$	c_1, b_2	$c_1, b_1 \to b_5$	c_1, b_5	$c_1, b_6 \to b_5$
$f_{c,b',b}$	6	0	8	0	8
$\tilde{z}_{c,b',b}$	0.6	0.4	0.3	0.4	0.3

It is worth noting that with grouping, the RGD can efficiently reduce the number of edge servers for the following second phase while preventing the same content from being placed highly densely. Therefore, it can help balance the number and density of requested contents, improving the placing and sidehaul transfer costs and mitigating the first challenge.

Fig. 2 illustrates an example for the RGD. Assume that there are two requested contents c_1 and c_2. The requests for contents c_1 and c_2 are $r_{1,2}$, $r_{1,3}$, $r_{1,4}$, $r_{1,5}$ and $r_{2,2}$, $r_{2,3}$, $r_{2,4}$, respectively. The optimum LP solutions $\tilde{z}_{c,b',b}$ are shown in Table I. Let $\gamma = 2$. In Fig. 2(a), the scopes of $r_{1,2}$ and $r_{1,5}$ are with radii $D_{r_{1,2}} = 2 \times (0.6 \times 6 + 0.4 \times 0) = 7.2$ and $D_{r_{1,5}} = 9.6$, respectively. Since $D_{r_{1,2}} < D_{r_{1,5}}$, the RGD creates group $g_{r_{1,2}}$ and distributes $r_{1,2}$ and $r_{1,5}$ into $g_{r_{1,2}}$. Note that group $g_{r_{1,5}}$ will not be created since $r_{1,5}$ has been distributed into $g_{r_{1,2}}$. After that, edge servers b_1 and b_2 are identified as the candidate edge servers in group $g_{r_{1,2}}$. For content c_1, the RGD creates groups $g_{r_{1,3}}$ and $g_{r_{1,4}}$ subsequently. Then, the RGD distributes disjoint groups for content c_2 by the same way, and three groups might be created, i.e., $g_{r_{2,2}}$, $g_{r_{2,3}}$, and $g_{r_{2,4}}$, as shown in Fig. 2(b). Finally, Fig. 2(c) presents the final group distribution and there are totally six groups and four candidate edge servers. Note that the groups for the same content are disjoint, while the groups for the different contents may be overlapping.

B. Content Placement Selection (CPS) Phase

To further minimize the costs of placing on edge servers and backhaul transfer from the EMP to edge servers, the CPS aims to find a suitable edge server for each group, while meeting the storage capacity constraints as much as possible.

After the group distribution of each requested content, the CPS first builds a bipartite graph H that consists of one set of groups and the other set of candidate edge servers in all groups. Let G denote the set of groups. For the set of candidates, the CPS makes at most $k_{b'} = \lceil \frac{\gamma}{\gamma - 1} \cdot \sum_{g_{r_{c,b}} \in G} \tilde{z}_{c,b',b} \rceil$ virtual copies with capacity of 1 for each candidate edge server b'

TABLE II
EXAMPLE OF CPS.

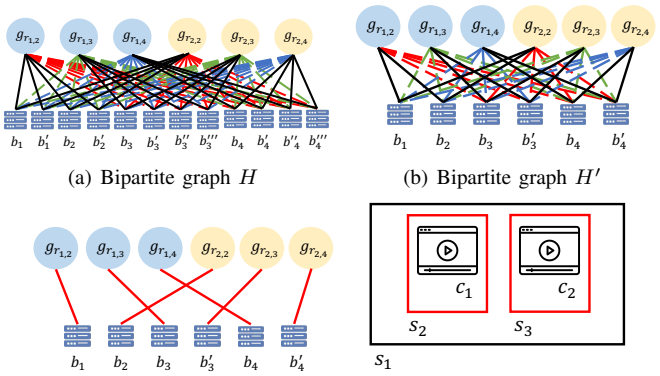
	$g_{r_{1,2}}$	$g_{r_{2,2}}$	$g_{r_{1,3}}$	$g_{r_{2,3}}$	$g_{r_{1,4}}$	$g_{r_{2,4}}$
b_1	0.6	0.4	0	0	0	0
b_2	0.4	0.6	0	0	0	0
b_3	0	0	1	1	0	0
b_4	0	0	0	0	1	1

in each group. Then, the costs between each group and its edge server copies are set to the content placing cost h_b plus the backhaul transfer cost f_b (i.e., $h_b + f_b$). In contrast, the costs between each group and the remaining edge server copies in other groups (i.e., not virtual copies of its edge servers) are set to ∞. Finally, the CPS finds a minimum-cost perfect matching on H by executing the Hungarian algorithm [14] to obtain one-to-one matching between a group and a virtual copy. Thereby, we assign an edge server to each group, and all requested contents in the group will be served by the selected edge server, further solving the second challenge.

Following Fig. 2, we show the example of the CPS. Assume that the storage capacity of edge servers b_1, b_2, b_3, and b_4 are 1, 1, 2, and 2, sequentially. Based on the result output by the RGD, the optimum LP solutions $\tilde{z}_{c,b',b}$ between groups and edge servers are shown in Table II. The CPS builds the bipartite graph H as shown in Fig. 3(a). Since $k_{b_1} = \lceil \frac{2}{2-1} \cdot \sum_{g_{r_{c,b}} \in G} \tilde{z}_{c,1,b} \rceil = \lceil 2 \cdot (0.6 + 0.4) \rceil = 2$, we make two virtual copies, i.e., b_1 and b'_1. Similarly, we have $k_{b_2} = 2$, $k_{b_3} = 4$, and $k_{b_4} = 4$. Note that $k_{b'}$ is the largest number of virtual copies of b'. Let $B_{c,b}$ denote the set of candidate edge servers in the scope of $r_{c,b}$. For each group $g_{r_{c,b}}$, we have $\sum_{b' \in B_{c,b}} \tilde{z}_{c,b',b} \geq \frac{\gamma-1}{\gamma}$, and thus the copies are at most $\frac{\gamma}{\gamma-1} \cdot \tilde{z}_{c,b',b}$ for each b'. It can further be reduced to the bipartite graph H' in Fig. 3(b). Since $\sum_{b' \in B_{1,2}} \tilde{z}_{1,b',2} = 1$ and $\sum_{b' \in B_{2,2}} \tilde{z}_{2,b',2} = 1$, we only need to have $\lceil 0.6 + 0.4 \rceil = 1$ copy for b_1. The number of copies for b_2, b_3, b_4 are 1, 2, and 2, respectively. Then, each group is connected to its candidate edge servers with costs $h_b + f_b$ (i.e., black lines). The remaining dashed lines are with costs ∞. After that, the CPS obtains the optimal one-to-one matching by the Hungarian algorithm, as shown in Fig. 3(c). The content c_1 will be placed on edge servers b_1, b_3, b_4, and the content c_2 will be placed on edge servers b_2, b_3, b_4.

C. Content Procurement Determination (CPD) Phase

After the content placement selection, the CPD aims to determine how to procure the requested contents from different CPs such that the costs of procuring and backhaul transfer from content servers to the EMP are minimized, while all user requests are satisfied. For a CP's content set, its cost is obtained by the procuring and backhaul transfer costs (i.e., $p_s + f_s$), and its CEI is the ratio of its cost to the number of remaining unprocured contents in it. The CPD then iteratively re-calculates the CEI of each content set and selects the set with the lowest CEI. The CPD terminates until all requested contents are procured. Following Fig. 2, we show the example of the CPD to procure two contents c_1 and c_2. Assume that there are three CPs with content servers s_1, s_2, and s_3, respectively. In Fig. 3(d), the content server s_1 provides contents c_1 and c_2 with $p_{s_1} + f_{s_1} = 5$, the content server s_2 provides content c_1



(a) Bipartite graph H (b) Bipartite graph H'

(c) Minimum perfect matching (d) Content procurement result

Fig. 3. Examples of CPS and CPD.

with $p_{s_2} + f_{s_2} = 2$, and the content server s_3 provides content c_2 with $p_{s_3} + f_{s_3} = 2$. In the first round, the CEIs are $\frac{5}{2}$, $\frac{2}{1}$, $\frac{2}{1}$ for each content set, so the CPD selects the content set of s_2. In the second round, the CEIs of s_1 and s_3 are updated to $\frac{5}{1}$, $\frac{2}{1}$, so the CPD selects the content set of s_3. The final content procurement result is shown by the red boxes in Fig. 3(d).

The REPLY is an approximation algorithm. Due to the page limit, the detailed proof is provided in Appendix C of [11]. Note that the storage capacity violation is negligible and can be omitted since $|C|$ is usually sufficiently large.

Theorem 2. When $\gamma = \ln |C|$, the proposed REPLY is a $(\ln |C| + 1, \min\{2, \frac{\ln |C|}{\ln |C| - 1}\})$-approximation algorithm for the VIDEO, where $|C|$ is the number of requested contents.

To deal with the violation, the REPLY allows the requests to be distributed into other edge servers outside its scope with the remaining capacity and lower backhaul transfer and placing costs. Specifically, in the phase of the CPS, we make $\mathcal{B}_b$ virtual copies with capacity of 1 for each edge server b. Then, the costs between each group and its candidate edge server copies are set to $h_b + f_b$. In contrast, the costs between each group and the remaining edge server copies in other groups are set to ∞. Finally, we can find a minimum-cost perfect matching by the Hungarian algorithm [14] without the violation.

IV. PERFORMANCE EVALUATION

We evaluate the performance of the proposed REPLY. This section includes simulation settings and numerical results.

A. Simulation Settings

1) Dataset: To make our simulation more realistic, we use two real-world datasets. **1) Edge Networks**: extracted from [7] with its dataset [15]. We consider a real network and random networks, respectively. The real network adopts the topology of Milan City Center with 30 edge servers. The random networks are generated by Erdös-Rényi random graph [16] with at most 100 edge servers. The average sidehaul data rate between two edge servers is 4.5 Gbps in the real network. **2) User Requests**: extracted from YouTube dataset [17]. We consider 180 videos, including 11 categories. The video is about 1 hour, and its size is approximately 5 GB. The requested videos will be randomly assigned to edge servers, and the number of requests is based on the popularity level of videos.

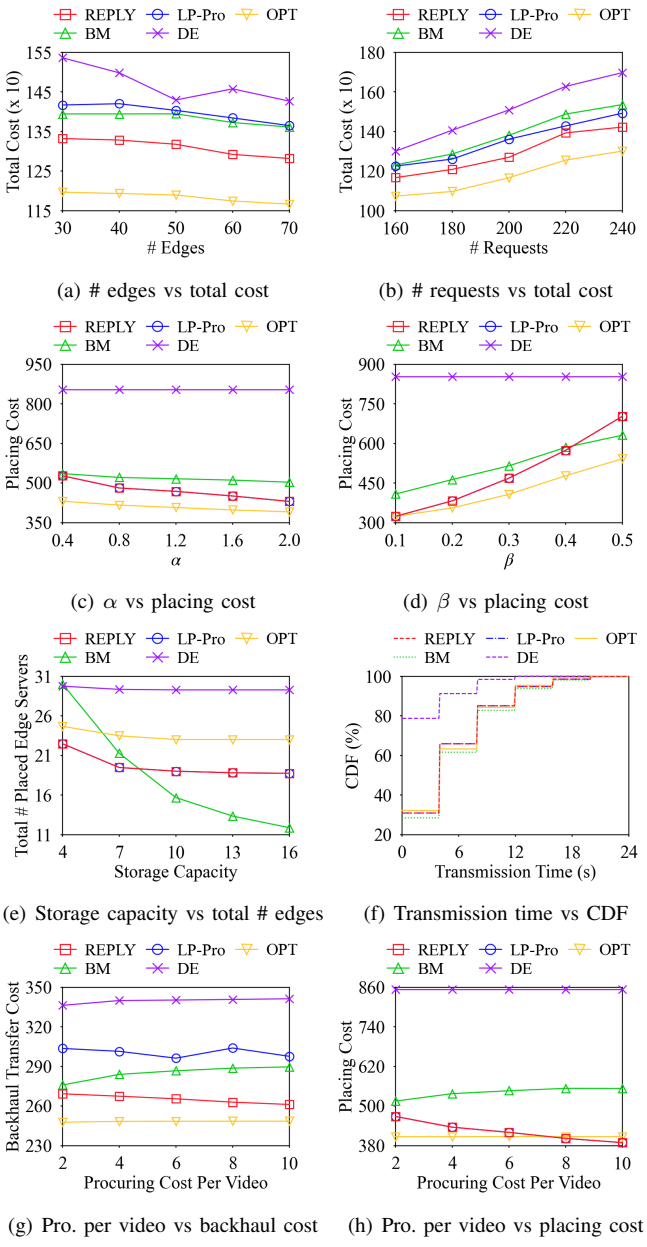
TABLE III
SIMULATION PARAMETERS.

Parameter	Default Value
# of edge servers $\|B\|$	30
# of videos $\|C\|$	180
# of CPs $\|S\|$	100
# of requests	200
1080p video size with length 1 hr	5 GB
Edge storage size	50 GB
Edge storage capacity $\mathcal{B}_b$	10
Avg. procuring cost per video	2 USD
Placing cost per capacity	0.5 USD
Placing cost per GB	0.08 USD
Backhaul data rate (content servers to EMP)	50 Gbps
Backhaul data rate (EMP to edge servers)	50 Gbps
Sidehaul data rate	10 Gbps (highest)
Backhaul transfer cost per hr α	1.2 USD
Sidehaul transfer cost per hr β	0.3 USD
Scope parameter γ	1.3

2) Parameter Settings: The default values of parameters are listed in Table III. Specifically, we arbitrarily combine with multiple similar categories, where their videos are composed as a CP's content set. The number of CPs is 100. The storage size of edge servers is by Gaussian distribution [15] with the mean of 50 GB and the standard deviation of 5. Thus, the default storage capacity will become $50/5 = 10$ (i.e., edge size divided by video size). In addition, the procuring cost is adopted by the price for renting a video on YouTube, which is average 2 USD per video [18]. The placing cost is set based on the edge storage capacity plus placing size adopted by the price of Amazon EBS, which is 0.08 USD/GB [19]. The backhaul data rate is set by Gaussian distribution with the mean of 50 Gbps and the standard deviation of 5. Note that the data rate can be transformed into transmission time using video size divided by data rate. To the end, the transfer costs per hour α and β are set as 1.2 and 0.3 USD/hr according to [20].

3) Baselines: We compare the performance of the REPLY with four baselines. **1) Bipartite Matching (BM)**: makes $\mathcal{B}_b$ virtual copies for each edge server b, builds a bipartite graph between all requests and all edge server copies based on the optimum LP solution $\tilde{z}$, sets the costs of links with $\tilde{z} > 0$ as the backhaul and placing costs (and others are ∞), and finds a minimum-cost perfect matching by the Hungarian algorithm [14]. The part of content procurement is the same as the REPLY. **2) LP-based Procurement (LP-Pro)**: rounds the fractional $\tilde{x}$ to 1 in a non-increasing order based on the optimum LP solution until all requested contents are procured. The part of content placement follows the REPLY. **3) Differential Evolution (DE)** [10]: sets the initial number of placed contents as the maximum and gradually eliminates the placed contents based on the distance of edge servers until at least one request exceeds the certain transmission time. The part of content procurement follows the REPLY. **4) OPT**: uses Gurobi to derive the optimal solution of ILP (5a)−(5f).

4) Performance Metrics: The performance is evaluated with the following metrics. 1) Total cost, 2) Placing cost, 3) Backhaul transfer cost, 4) Total number of placed edge servers, and 5) Cumulative distribution function (CDF): the percentage of user requests with transmission time less than or equal to a value. Each simulation result is averaged over 15 trials. More performance metrics are presented in Appendix D of [11].



(a) # edges vs total cost (b) # requests vs total cost

(c) α vs placing cost (d) β vs placing cost

(e) Storage capacity vs total # edges (f) Transmission time vs CDF

(g) Pro. per video vs backhaul cost (h) Pro. per video vs placing cost

Fig. 4. Effects of different parameters.

B. Numerical Results

The numerical results in Fig. 4 manifest the effects of the number of edge servers and requests, α, β, storage capacity, transmission time, and average procuring cost per video. Due to the page limit, more experimental results with various parameters are shown in Appendix D of [11].

The proposed REPLY achieves the lowest total cost and outperforms other baselines, as shown in Figs. 4(a) and 4(b). In Fig. 4(a), the total cost of all algorithms slightly decreases as the number of edge servers increases. In Fig. 4(b), the total cost of all algorithms increases as the number of requested videos increases. Overall, the REPLY can efficiently balance the procuring, placing, backhaul transfer, and sidehaul transfer

costs to provide a better content procurement, placement, and service jointly. The detailed explanation is as follows.

1) The First and Second Challenges: The effects of backhaul transfer cost α and sidehaul transfer cost β on the placing cost are shown in Figs. 4(c) and 4(d). In Fig. 4(c), the placing cost of all algorithms tends to decrease as α increases, since all of them will reduce the number of placed edge servers and select those edge servers with cheaper backhaul transfer cost, further leading to a lower placing cost. Specifically, the REPLY has better performance than the BM and DE, since it can efficiently distribute request groups and determine content placement. The BM selecting the minimum backhaul transfer and placing costs for all requests may have a lower placing cost but incur a higher sidehaul transfer cost. In Fig. 4(d), the placing cost of all algorithms increases as β increases, since they will increase the number of placed edge servers when the sidehaul transfer cost enhances, which causes a higher placing cost. Although the REPLY has a higher placing cost when $\beta = 0.5$, it can achieve a lower sidehaul transfer cost.

The effect of edge storage capacity on the total number of placed edge servers is shown in Fig. 4(e). As the storage capacity increases, the total number of placed edge servers for all algorithms will decrease. Specifically, the BM determines the content placement individually for each request, and thus the number of edge servers is larger when the storage capacity is 4. Compared with the BM and DE, the REPLY balances the placing, backhaul transfer and sidehaul transfer costs and determines the suitable total number of placed edge servers. Fig. 4(f) further manifests the effect of sidehaul transmission time on the CDF. The result shows that in the REPLY, most 85% requests are satisfied within 8 s and that no request transmitted exceeds 20 s. The DE has about 80% requests transmitted immediately since it gradually reduces the number of placed edge servers without exceeding a certain transmission time. However, the DE neglects the backhaul transfer and placing costs of edge servers. Thus, the REPLY can select the suitable content placement and ensure a moderate transmission time for end users, mitigating the first and second challenges.

2) The Third Challenge: The effect of procuring cost per video on the backhaul transfer cost is shown in Fig. 4(g). The backhaul transfer cost of all algorithms tends to slightly decrease as the procuring cost per video increases. Since the content procurement is needed as long as there are requests for contents, all of them will try to reduce the backhaul transfer cost with the increasing procuring cost. Specifically, the REPLY achieves better performance than the BM, LP-Pro, and DE, since the content procurement of the REPLY can efficiently reduce the procuring and backhaul transfer costs. Although the BM and DE use the same content procurement as the REPLY, the backhaul transfer costs are still different due to the different content placement. The BM may have a lower backhaul transfer cost but a higher placing cost, since it cannot efficiently deal with the number and density of placed edge servers. The effect of procuring cost per video on the placing cost is shown in Fig. 4(h). The placing cost of the REPLY decreases when the procuring cost per video increases and also achieves a better performance, since it will similarly try to reduce the placing cost with the increasing procuring cost. Overall, the proposed REPLY can balance the content procurement and placement, solving the third challenge.

Although the OPT can obtain the optimal solution, it requires longer and its time grows rapidly than other algorithms. Thus, the proposed REPLY can achieve the solution close to the OPT in a short running time compared with others.

V. Conclusion

This paper investigates an optimization problem called VIDEO for content services in edge networks. To tackle with challenges, the paper designs an approximation algorithm named REPLY with three phases. In the first phase, each different requested content is distributed into several disjoint groups and each group has some candidate edge servers. In the second phase, it selects one edge server for each group and assigns all requests in the group to the selected edge server. In the last phase, it determines the content procurement. Overall, the paper proves the NP-hardness and the approximation ratio in the theoretical analysis, and the simulation results manifest that the proposed REPLY outperforms other baselines by up to 20% in the experimental evaluation.

References

[1] "Telefónica lures netflix into its content aggregator platform," 2018. [Online]. Available: https://reurl.cc/671G36

[2] "Video platform expansion into various countries: Impact on telecom operators, engineers, and company project procurement, and diversification analysis," 2024. [Online]. Available: https://reurl.cc/801yV7

[3] Y. Zhang, X. Lan, J. Ren, and L. Cai, "Efficient computing resource sharing for mobile edge-cloud computing networks," *IEEE/ACM Trans. Netw.*, vol. 28, no. 3, pp. 1227–1240, 2020.

[4] C. Yi *et al.*, "Workload re-allocation for edge computing with server collaboration: A cooperative queueing game approach," *IEEE Trans. Mob. Comput.*, vol. 22, no. 5, pp. 3095–3111, 2023.

[5] Z. Lv and L. Qiao, "Optimization of collaborative resource allocation for mobile edge computing," *Comput. Commun.*, vol. 161, pp. 19–27, 2020.

[6] Z. Xu *et al.*, "Collaborate or separate? distributed service caching in mobile edge clouds," in *IEEE INFOCOM*, 2020.

[7] B. Xiang, J. Elias, F. Martignon, and E. Di Nitto, "Resource calendaring for mobile edge computing: Centralized and decentralized optimization approaches," *Comput. Netw.*, vol. 199, p. 108426, 2021.

[8] V. Farhadi *et al.*, "Service placement and request scheduling for data-intensive applications in edge clouds," *IEEE/ACM Trans. Netw.*, vol. 29, no. 2, pp. 779–792, 2021.

[9] Q. Tang *et al.*, "Joint service deployment and task scheduling for satellite edge computing: A two-timescale hierarchical approach," *IEEE J. Sel. Areas Commun.*, pp. 1–1, 2024.

[10] Y. Wang *et al.*, "Joint deployment and task scheduling optimization for large-scale mobile users in multi-UAV-enabled mobile edge computing," *IEEE Trans. Cybern.*, vol. 50, no. 9, pp. 3984–3997, 2020.

[11] C.-A. Yang *et al.*, "Near-optimal content service algorithm with procurement and deployment in edge networks (technical report)," Apr 2024. [Online]. Available: https://reurl.cc/6dD3dd

[12] A. A. Haghighi, S. Shah Heydari, and S. Shahbazpanahi, "Dynamic QoS-aware resource assignment in cloud-based content-delivery networks," *IEEE Access*, vol. 6, pp. 2298–2309, 2018.

[13] D. P. Williamson and D. B. Shmoys, *The Design of Approximation Algorithms*. Cambridge University Press, 2011.

[14] D. B. West, *Introduction to Graph Theory*. Prentice Hall, 2001.

[15] B. Xiang, J. Elias, F. Martignon, and E. Di Nitto, "A dataset for mobile edge computing network topologies," *Data Br.*, vol. 39, p. 107557, 2021.

[16] P. Erdös and A. Rényi, "On random graphs I," *Publ. Math. Debrecen*, vol. 6, no. 290-297, p. 18, 1959.

[17] "YouTube dataset," 2008. [Online]. Available: https://reurl.cc/bDZMLv

[18] "Buy or rent movies and TV shows on YouTube," 2024. [Online]. Available: https://reurl.cc/aLpQo4

[19] "Amazon EBS," 2024. [Online]. Available: https://reurl.cc/D4WdzN

[20] "Amazon EC2," 2024. [Online]. Available: https://reurl.cc/4j21rY

APPENDIX A
RELATED WORK

The content service in edge networks mainly focuses on two parts, including 1) resource allocation and service placement and 2) service placement and request scheduling. Some related works are summarized as follows.

A. Resource Allocation and Service Placement

This part aims to determine service placement and optimize resource allocation to edge servers. Y. Zhang et al. develop a convex programming and a heuristic algorithm to deal with the resource management sharing between the cloud and edge servers by formulating as two problems, i.e., social welfare maximization and profit maximization, respectively [3]. C. Yi et al. introduce a cooperative queueing game approach to solve competitions and collaborations among strategic edge servers in sharing their computing capacities, such that the expected cost is minimized [4]. These two works focus on resource allocation and management. Z. Lv et al. propose two heuristic algorithms for service placement and three heuristic algorithms for resource allocation, such that the access delay for edge servers is minimized in mobile edge networks [5]. Z. Xu et al. design a randomized rounding algorithm for service caching and a game-theoretical mechanism for resource sharing, such that the social cost of all network service providers is minimized [6]. These two works address resource allocation and service placement, but they do separately.

B. Service Placement and Request Scheduling

This part aims to determine service placement and request scheduling in order to minimize transmission delay and enhance quality of experience for users. B. Xiang et al. propose a heuristic algorithm to jointly optimize the request admission, scheduling, routing, and the amount of processing and storage capacity reserved on edge servers and an alternating direction method of multipliers to solve the resource allocation decision in a distributed fashion [7]. V. Farhadi et al. develop an approximation algorithm for service placement and then an optimal and a heuristic algorithm for homogeneous and heterogeneous request scheduling, such that the expected number of served requests is maximized [8]. Q. Tang et al. introduce a soft actor-critic-based deep reinforcement learning method to learn a stationary scheduling policy and a heuristic-based atomic orbital search approach for service placement, such that the cost is minimized [9]. Y. Wang et al. present a differential evolution algorithm to optimize the deployment of UAVs and a greedy algorithm for task scheduling, such that the system energy consumption is minimized [10]. Most of works address service placement and requesting scheduling separately.

Our problem VIDEO combines both two parts and determines content procurement, placement, and service jointly. There are few works considering all of them together.

APPENDIX B
PROOF OF THEOREM 1

We prove the theorem by reducing the vertex cover problem to the VIDEO. Given a graph $G = (V, E)$, the vertex cover problem asks whether there is a subset of edges $E' \subseteq E$ covering all the nodes $v \in V$. To accomplish the reduction from the vertex cover problem to the VIDEO, we have to show how to construct an instance of the VIDEO (i.e., C, S, B, p_s, f_s, f_b) for an arbitrary given graph G' for the vertex cover problem. The construction steps are detailed as follows. First, we create only one edge server b, set its storage capacity $\mathcal{B}_b$ to $|V|$, and add it to B. Then, for each vertex in $v \in V$, we create a corresponding content c and add it to C. Subsequently, for each edge $e \in E$, we create a corresponding content provider s, make s have the corresponding contents of the vertices of e, and set $p_s + f_s + f_b = 1$. The above reduction can be done in the polynomial time. In this way, the vertex cover problem can be solved if the solution of the VIDEO can be found. However, the vertex cover problem is a known NP-hard problem and can only be solved when $P = NP$. Therefore, the theorem holds.

APPENDIX C
PROOF OF THEOREM 2

For ease of presentation, let (x^*, y^*, z^*), $(\tilde{x}, \tilde{y}, \tilde{z})$, and $(\hat{x}, \hat{y}, \hat{z})$ denote the optimum ILP solution, the optimum LP solution, and the REPLY's solution, respectively. The analyses of sidehaul transfer cost, placing cost, and procurement cost are as follows. We first analyze the sidehaul transfer cost. Let $B_{c,b}$ represent the set of candidate edge servers in the scope of the request $r_{c,b}$ (i.e., edge servers with $\tilde{z}_{c,b',b} > 0$) by the RGD. If the REPLY creates group $g_{r_{u,v}}$ for a request $r_{u,v}$, then request $r_{u,v}$'s transmission time is:

$$\begin{aligned}\sum_{b' \in B} f_{u,b',v} \cdot \hat{z}_{u,b',v} &= \sum_{b' \in B_{u,v}} f_{u,b',v} \cdot \hat{z}_{u,b',v} \le D_{r_{u,v}} \\ &= \gamma \sum_{b' \in B} f_{u,b',v} \cdot \tilde{z}_{u,b',v}. \end{aligned} \tag{7}$$

Otherwise, suppose that group $g_{r_{c,b}}$ created for request $r_{c,b}$ contains the request $r_{u,v}$ (i.e., $D_{r_{c,b}} \le D_{r_{u,v}}$), and then, by triangle inequality, request $r_{u,v}$'s transmission time is:

$$\begin{aligned}\sum_{b' \in B} f_{u,b',v} \cdot \hat{z}_{u,b',v} &= \sum_{b' \in B_{u,v}} f_{u,b',v} \cdot \hat{z}_{u,b',v} \le D_{r_{u,v}} + 2D_{r_{c,b}} \\ &\le 3D_{r_{u,v}} \\ &\le 3\gamma \sum_{b' \in B} f_{u,b',v} \cdot \tilde{z}_{u,b',v}. \end{aligned} \tag{8}$$

Therefore, by Eqs. (7) and (8), each request $r_{u,v}$'s transmission time is at most $3\gamma \sum_{b' \in B} f_{u,b',v} \cdot \tilde{z}_{u,b',v}$.

Subsequently, we analyze the placing cost. Let $\overline{y}$ denote the optimum solution of bipartite matching. In the CPS, we have

$$\sum_{b' \in B} (h_{b'} + f_{b'}) \cdot \overline{y}_{c,b'} \le \frac{\gamma}{\gamma - 1} \cdot \sum_{b' \in B} (h_{b'} + f_{b'}) \cdot \tilde{y}_{c,b'}. \tag{9}$$

That is because

$$\sum_{b' \in B_{c,b}} \tilde{y}_{c,b'} \ge \sum_{b' \in B_{c,b}} \tilde{z}_{c,b',b} \ge \frac{\gamma - 1}{\gamma} \tag{10}$$

for each request $r_{c,b}$; otherwise,

$$\sum_{b' \notin B_{c,b}} \tilde{z}_{c,b',b} = 1 - \sum_{b' \in B_{c,b}} \tilde{z}_{c,b',b} > \frac{1}{\gamma}, \tag{11}$$

implying a contradiction as follows:

$$\begin{aligned}
&\sum_{b'\in B} f_{c,b',b}\cdot\tilde{z}_{c,b',b}\\
&=\sum_{b'\in B_{c,b}} f_{c,b',b}\cdot\tilde{z}_{c,b',b}+\sum_{b'\notin B_{c,b}} f_{c,b',b}\cdot\tilde{z}_{c,b',b}\\
&>\sum_{b'\in B_{c,b}} f_{c,b',b}\cdot\tilde{z}_{c,b',b}+\frac{1}{\gamma}\cdot D_{r_{c,b}}\geq\frac{1}{\gamma}\cdot D_{r_{c,b}}\\
&=\frac{1}{\gamma}\cdot\gamma\sum_{b'\in B} f_{c,b',b}\cdot\tilde{z}_{c,b',b}\\
&=\sum_{b'\in B} f_{c,b',b}\cdot\tilde{z}_{c,b',b}. \qquad (12)
\end{aligned}$$

By Eq. (10), $\tilde{y}$ will contain the fractional edge servers in each group $g_{r_{c,b}}$ by a fractional satisfaction factor of at least $\frac{\gamma-1}{\gamma}$, implying that Eq. (9) holds. Therefore, the placing cost equal to the minimum-cost perfect matching is:

$$\begin{aligned}
\sum_{b'\in B}(h_{b'}+f_{b'})\cdot\hat{y}_{c,b'}&=\sum_{b'\in B}(h_{b'}+f_{b'})\cdot\overline{y}_{c,b'}\\
&\leq\frac{\gamma}{\gamma-1}\cdot\sum_{b'\in B}(h_{b'}+f_{b'})\cdot\tilde{y}_{c,b'}. \qquad (13)
\end{aligned}$$

Afterwards, we then analyze the procuring cost. Assume the CPD sequentially procures from the content servers s_1, s_2, s_3, and so on in each round. Since the CPD greedily selects a content set with the lowest CEI, we have

$$(p_{s_1}+f_{s_1})\cdot\hat{x}_{s_1}\leq\frac{1}{|C|}\cdot\sum_{s\in S}(p_s+f_s)\cdot x_s^*, \qquad (14)$$

$$(p_{s_2}+f_{s_2})\cdot\hat{x}_{s_2}\leq\frac{1}{|C|-|C_{s_1}|}\cdot\sum_{s\in S}(p_s+f_s)\cdot x_s^*, \qquad (15)$$

$$(p_{s_3}+f_{s_3})\cdot\hat{x}_{s_3}\leq\frac{1}{|C|-|C_{s_1}|-|C_{s_2}|}\cdot\sum_{s\in S}(p_s+f_s)\cdot x_s^*, \qquad (16)$$

and so on. Therefore, the procuring cost is:

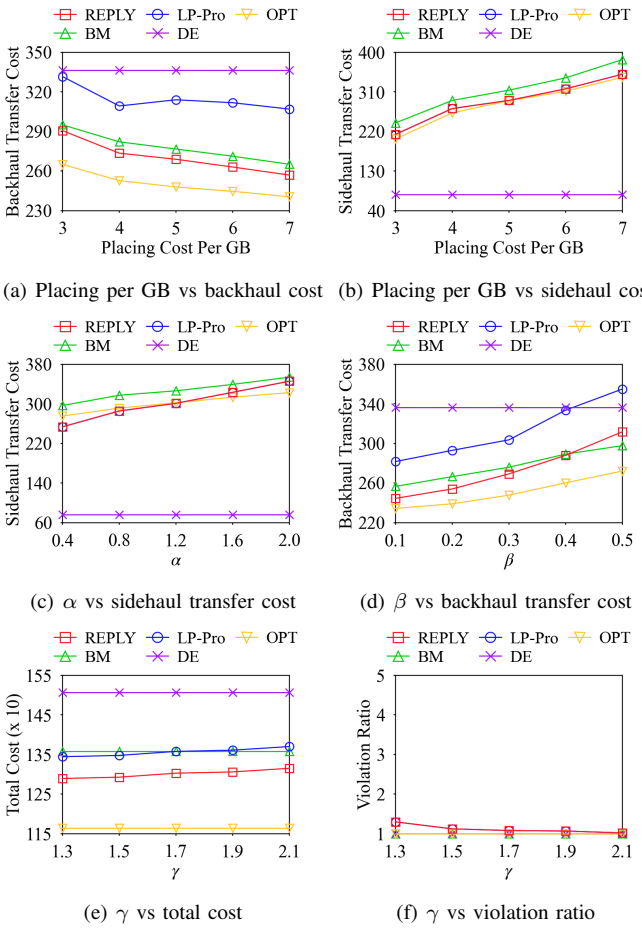
$$\begin{aligned}
\sum_{s\in S}(p_s+f_s)\cdot\hat{x}_s&\leq H_{|C|}\cdot\sum_{s\in S}(p_s+f_s)\cdot x_s^*\\
&\leq(\ln|C|+1)\cdot\sum_{s\in S}(p_s+f_s)\cdot x_s^*, \qquad (17)
\end{aligned}$$

where the harmonic series $H_{|C|}=1+\frac{1}{2}+...+\frac{1}{|C|}$.

Finally, by Eqs. (7), (8), (13), and (17), we obtain the total cost of $(\hat{x},\hat{y},\hat{z})$, as shown in Eq. (18). Thus, the approximation ratio is $\max\{\ln|C|+1,\frac{\gamma}{\gamma-1},3\gamma\}$. By Eq. (10), we have the storage capacity violation at most $\frac{\gamma}{\gamma-1}$. To the end, when $\gamma=\ln|C|$, the proposed REPLY is a $(\ln|C|+1,\min\{2,\frac{\ln|C|}{\ln|C|-1}\})$-approximation algorithm for the VIDEO.

Appendix D
Other Experimental Results

The effects of placing cost per GB, α, β, and γ on the different costs are shown in Fig. 5. Our proposed REPLY has the suitable content procurement, placement, and service, and also achieves the lowest total cost compared with other baselines. The detailed explanation is as follows.



(a) Placing per GB vs backhaul cost (b) Placing per GB vs sidehaul cost

(c) α vs sidehaul transfer cost (d) β vs backhaul transfer cost

(e) γ vs total cost (f) γ vs violation ratio

Fig. 5. Effects of different parameters.

A. *Effect of Placing Cost per GB*

The effect of placing cost per GB on backhaul transfer and sidehaul transfer costs is shown in Figs. 5(a) and 5(b). In Fig. 5(a), the backhaul transfer cost of all algorithms decreases as the placing cost per GB increases, since all of them will try to reduce the expensive placing cost, further leading to a lower backhaul transfer cost. In Fig. 5(b), the sidehaul transfer cost of all algorithms increases as the placing cost per GB increases, since the number of placed edge servers tends to reduce and thus the sidehaul transfer cost is enhanced. Specifically, the BM has a lower backhaul transfer cost but a higher sidehaul transfer cost, since it finds the minimum placing and backhaul transfer costs for each request but neglects the sidehaul transfer cost. The DE has a lower sidehaul transfer cost but a higher backhaul transfer cost, since it reduces the number of placed edge servers but neglects the backhaul transfer cost. Therefore, the proposed REPLY can balance two costs at the same time.

B. *Effects of α and β*

The effects of α and β on the backhaul transfer and sidehaul transfer costs are shown in Figs. 5(c) and 5(d). In Fig. 5(c), the sidehaul transfer cost of all algorithms increases as α increases, since the number of placed edge servers will reduce as the backhaul transfer cost increases, which causes a higher sidehaul transfer cost. In Fig. 5(d), the backhaul transfer cost of all algorithms increases as β increases, since the number

$$
\begin{aligned}
&\sum_{s\in S}(p_s+\alpha\cdot f_s)\cdot\hat{x}_s+\sum_{c\in C}\sum_{b\in B}(h_b+\alpha\cdot f_b)\cdot\hat{y}_{c,b}+\beta\sum_{c\in C}\sum_{b\in B}\sum_{b'\in B}f_{c,b',b}\cdot\hat{z}_{c,b',b}\\
&\leq \alpha\cdot\sum_{s\in S}(p_s+f_s)\cdot\hat{x}_s+\alpha\cdot\sum_{c\in C}\sum_{b\in B}(h_b+f_b)\cdot\hat{y}_{c,b}+\beta\sum_{c\in C}\sum_{b\in B}\sum_{b'\in B}f_{c,b',b}\cdot\hat{z}_{c,b',b}\\
&\leq (\ln|C|+1)\cdot\alpha\sum_{s\in S}(p_s+f_s)\cdot x_s^*+\frac{\gamma}{\gamma-1}\cdot\alpha\sum_{c\in C}\sum_{b\in B}(h_b+f_b)\cdot\tilde{y}_{c,b}+3\gamma\cdot\beta\sum_{c\in C}\sum_{b\in B}\sum_{b'\in B}f_{c,b',b}\cdot\tilde{z}_{c,b',b}\\
&\leq \max\{\ln|C|+1,\frac{\gamma}{\gamma-1},3\gamma\}\cdot(\alpha\sum_{s\in S}(p_s+f_s)\cdot x_s^*+\alpha\sum_{c\in C}\sum_{b\in B}(h_b+f_b)\cdot y_{c,b}^*+\beta\sum_{c\in C}\sum_{b\in B}\sum_{b'\in B}f_{c,b',b}\cdot z_{c,b',b}^*).
\end{aligned}
\tag{18}
$$

of placed edge servers will increase as the sidehaul transfer cost increases, which causes a higher backhaul transfer cost. Similarly, the proposed REPLY can balance two costs.

C. *Effect of* γ

The effect of the scope parameter γ on the total cost is shown in Fig. 5(e). As γ increases, the violation ratio will decrease, but the total cost will increase. Since the scope becomes larger, the REPLY will include more edge servers in the scope and reduce the violation ratio. However, the larger scope may incur a higher sidehaul transfer cost, further leading to a higher total cost. Overall, the proposed REPLY can achieve a lower total cost given a proper γ compared with other algorithms.

Without the modification of violation, the effect of the scope parameter γ on the violation ratio is shown in Fig. 5(f). As a result, we can find that the violation ratio of the proposed REPLY is significantly smaller than the theoretical analysis result. When $\gamma = 1.3$, the violation ratio is about 1.3. When $\gamma = 2.1$, there is almost no violation of storage capacity.