Formulating Interference-aware Data Delivery Strategies in Edge Storage Systems

Xiaoyu Xia
The University of Adelaide
Australia
xiaoyu.xia@adelaide.edu.au

Feifei Chen
Deakin University
Australia
feifei.chen@deakin.edu.au

Qiang He
Swinburne University of Technology
Australia
qhe@swin.edu.au

Guangming Cui
Swinburne University of Technology
Australia
gcui@swin.edu.au

John Grundy
Monash University
Australia
john.grundy@monash.edu

Mohamed Abdelrazek
Deakin University
Australia
mohamed.abdelrazek@deakin.edu.au

Fang Dong
Southeast University
China
fdong@seu.edu.cn

ABSTRACT

Networked edge servers constitute an edge storage system in edge computing (EC). Upon users' requests, data must be delivered from edge servers in the system or from the cloud to users. Existing studies of edge storage systems have unfortunately neglected the fact that an excessive number of users accessing the same edge server for data may impact users' data rates seriously due to the wireless interference. Thus, users must first be allocated to edge servers properly for ensuring their data rates. After that, requested data can be delivered to users to minimize their average data delivery latency. In this paper, we formulate this **I**nterference-aware **D**ata **D**elivery at the network **E**dge (IDDE) problem, and demonstrate its NP-hardness. To tackle it effectively and efficiently, we propose IDDE-G, a novel approach that first finds a Nash equilibrium as the strategy for allocating users. Then, it finds an approximate strategy for delivering requested data to allocated users. We analyze the performance of IDDE-G theoretically and evaluate its performance experimentally to demonstrate the effectiveness and efficiency of IDDE-G on solving the IDDE problem.

KEYWORDS

edge computing, edge storage system, data delivery, interference-aware, user allocation

ACM Reference Format:
Xiaoyu Xia, Feifei Chen, Qiang He, Guangming Cui, John Grundy, Mohamed Abdelrazek, and Fang Dong. 2022. Formulating Interference-aware Data Delivery Strategies in Edge Storage Systems. In *Proceedings of ICPP'51.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The rapid growth of internet-of-things (IoT) and mobile devices has fueled the emergence of various online applications that require real-time responsiveness [14, 39], e.g., autonomous driving [2], video streaming [30], etc. The long latency in communicating with the remote cloud servers is hindering further advances in these applications [9]. To tackle this challenge, edge computing (EC) pushes computing and storage capacities to edge servers near users at the network edge [21]. The resources on edge servers can be hired by app vendors like Facebook and Nintendo for storing popular data to serve nearby users [4, 37]. This minimizes data retrieval latency and reduces the traffic pressure on the back-haul network significantly [37]. Fig. 1 compares the end-to-end network latency when a mobile device accesses data from an edge server and remote cloud servers deployed in Amazon's Singapore, London and Frankfurt data centers. It shows the remarkable advantage offered by edge computing for latency-sensitive applications. For example, high-quality VR can be facilitated by edge computing as it requires a 20ms end-to-end latency or lower to prevent motion sickness [13].

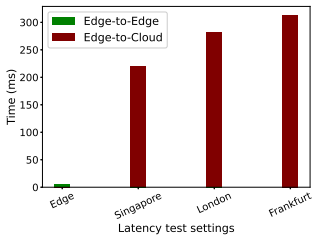


Figure 1: End-to-end network latency test. The results are collected hourly and averaged over a week in March 2022.

Networked edge servers constitute a novel storage system at the edge of network. Such *edge storage systems* are fundamentally different from conventional cloud-based storage systems [34]. Adjacent edge servers in an edge storage system can communicate via high-speed links [26]. It overcomes the single-point failures and performance bottlenecks encountered by the edge-cloud architecture, where edge servers communicate via a macro base station or the back-haul network [31].

Unlike in cloud-based storage systems where data are always delivered from a remote cloud server, app vendors pursue to deliver data to users from within the edge storage system to minimize their data delivery latency. Fig. 2 illustrates an exemplar edge storage system involving four edge servers $\{v_1, ..., v_4\}$ and nine users

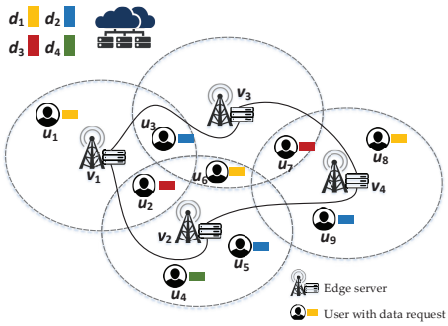


Figure 2: Example edge storage system

$\{u_1, ..., u_9\}$ in a specific area. At the moment, users u_1, u_6 and u_8 request data d_1; users u_3, u_5 and u_9 request data d_2; users u_2 and u_6 request data d_3; and users u_4 requests data d_4. Once the requested data has been stored in this system, it can be delivered to the user from within the system. Otherwise, it will be delivered from the remote cloud.

The new challenges raised by edge storage systems start to attract researchers' attentions recently, aiming to minimize storage cost and/or maximize system performance [24, 37]. However, existing studies have unfortunately neglected three major and unique issues in real-world edge storage systems.

Resource constraint. App vendors must reserve storage spaces on edge servers in advance. Existing studies assume that app vendors can hire storage resources on any edge server on demand at any time, similar to resource hiring in cloud-based storage systems. However, an edge server usually has limited storage capacities due to its physical size limit [8, 11, 29, 40]. App vendors must not assume that there will always be adequate storage resources on edge servers for hire . In fact, app vendors have to compete for storage resources for storing their own data [35]. A practical strategy is to reserve storage resources in the edge storage system in advance based on their budget [34]. Take Fig. 2 as an example. An app vendor, e.g., Facebook, needs to reserve storage spaces on these edge servers in advance for storing popular data that may be requested by the users in the area.

Edge server collaboration. The collaboration among edge servers is unfortunately ignored in many existing studies, such as [6, 20, 36]. Via high-speed links, adjacent edge servers can share resources and transmit data to each other [26]. Assuming that data d_1 is stored on edge server v_3 but not on v_1 and v_4 in the edge storage system presented in Fig. 2. User u_6 can access d_1 directly, while users u_1 and u_8 can access d_1 via the link between v_1 and v_3, and the link between v_3 and v_4, respectively. In this way, d_1 does not need to be delivered to u_1 and u_8 from the remote cloud server. Edge servers' ability to collaborate can reduce data delivery latency without incurring excessive storage overheads.

Communication interference. Existing studies have focused on the computing or storage aspects, and neglected or oversimplified the communication aspect in the "last mile" during data delivery in the EC environment, i.e., the communication between user devices and edge servers. Multiple users can access edge servers for data via wireless communication concurrently. The wireless interference incurred may significantly impact their achievable data rates and lower their quality of experience [5, 6]. Thus, interference must be considered during data delivery in edge storage systems. Assuming that there are two channels on each edge server in Fig. 2. Allocating too many users to the same channel on an edge server tends to incur severe interference and lowers users' average data rates. For example, user u_7 can access to either edge server v_3 or v_4. In the case that u_7 is allocated to v_4, $\{u_7, u_8, u_9\}$ will have to share the two channels on v_4. At least 2 users will be allocated to a same channel on v_4 and interference incurs. Please note this is an simplified example. In the real-world, the number of users involved is much larger, which incurs much greater interference and significantly complicates the problem. Fortunately, in the EC environment, app vendors have access to various communication information at the edge, such as received power and signals, neighbor cells, throughput and QoS [28]. This allows them to make informed decisions, considering not only the computation and storage aspects, but also the communication aspect [6].

Given limited reserved storage spaces in an edge storage system, upon a set of user requests, it is critical for an app vendor to formulate an edge data delivery strategy with consideration of wireless interference to 1) maximize users' average data rate and 2) minimize their average data delivery latency. In this study, we investigate this **I**nterference-aware **D**ata **D**elivery at the network **E**dge (IDDE) problem. The key contributions are:

- We formally model the IDDE problem and prove that it is $\mathcal{NP}$-complete.
- We propose IDDE-G, a game-based approach, for formulating IDDE strategies. First, it formulates the IDDE problem as a game and finds a Nash equilibrium as the strategy for user allocation. Then, it employs an approximate algorithm to determine how the requested data will be delivered to corresponding users.
- We theoretically and experimentally evaluate IDDE-G's performance to demonstrate its effectiveness and efficiency.

The organization of the paper is as follows. Section 2 first introduces the system model and then models the IDDE problem. Section 3 presents IDDE-G and analyzes its performance theoretically. Section 4 evaluates IDDE-G experimentally. Section 5 reviews the related work. Section 6 concludes this study and points out the future work.

2 PROBLEM STATEMENT

In this section, we first present the main concepts and models. Then, we formulate the IDDE problem and analyze its hardness.

2.1 System Models

Given the remote cloud server *cloud*, edge servers $V = \{v_1, \cdots, v_N\}$, users $U = \{u_1, \cdots, u_M\}$ and requested data $D = \{d_1, \cdots, d_K\}$ in an area, we need to formulate an IDDE strategy including: 1) a user allocation profile for allocating U to the proper channels on V so that U's average data rate is maximized; and 2) a data delivery profile for delivering D to a subset of V that cover U to serve U. Take Fig. 2 for example. A user allocation profile is formulated first to allocate $U = \{u_1, ..., u_9\}$ to $V = \{v_1, ..., v_4\}$ (or a subset of V) on appropriate channels. Then, a data delivery profile is formulated to determine how $D = \{d_1, ..., d_4\}$ are delivered to V (or the subset of V) to serve V, from other edge servers in V or from *cloud*.

Definition 1 (User Allocation Profile). *Denote $\alpha_j = (i, x)$ as the user allocation decision, indicating whether user u_j is allocated to edge server v_i's xth channel, denoted as $c_{i,x}$. If u_j is not allocated, there is $\alpha_j = (0, 0)$. The user allocation profile, denoted by $\alpha = \{\alpha_1, \cdots, \alpha_M\}$, consists of all users' allocation decisions.*

Let $U_{i,x}(\alpha)$ denote the users allocated to $c_{i,x}$ by α, and $U_i(\alpha) = \{U_{i,x}(\alpha), \forall c_{i,x} \in C_i\}$ denote the users allocated to v_i by α, where C_i is the set of channels on v_i. Let V_j represent the set of edge servers covering u_j, u_j can only be allocated to an edge server in V_j:

$$\alpha_j = (i, x) \neq (0, 0), \text{ iff } v_i \in V_j \tag{1}$$

Definition 2 (Data Delivery Profile). *A data delivery profile is represented by $\sigma = \{\sigma_{1,1}, \cdots \sigma_{1,K}, \cdots, \sigma_{N,K}\}$, where $\sigma_{i,k} \in \{0, 1\}$ is the data delivery decision indicating whether data d_k $(1 \leq k \leq K)$ is delivered to edge server v_i $(1 \leq i \leq N)$.*

Unlike cloud-based storage systems with virtually unlimited storage capacities, storage capacities are constrained on edge servers [8, 11, 29, 40]. As mentioned in Section 1, many app vendors may need to hire those resources in the same edge storage system for storing their own data. Due to the competition among app vendors, it is impractical to store plenty of data on every edge server for every app vendor [34]. Reserving edge servers' storage spaces in advance is a standard way for app vendors. Thus, the storage resource cost incurred for an app vendor to store data in an edge storage system remains unchanged unless it changes its storage reservation, and it is omitted in the models presented in this section.

2.2 User-Server Communication Model

In the EC environment, the signals received by users from edge servers are subject to wireless communication interference [4]. The channel gain between u_j and $c_{i,x}$, denoted by $g_{i,x,j}$ [7]:

$$g_{i,x,j} = \eta \cdot H_{i,j}^{-loss} \tag{2}$$

where η is the frequency dependent factor, $H_{i,j}$ is the distance between v_i and u_j, and $loss$ is the loss exponent. Allocating a set of users to different channels on the same edge server can significantly reduce the interference. According to [5, 38], the Signal-to-Interference-plus-Noise Ratio (SINR) for user u_j's with edge server v_i on $c_{i,x}$ is:

$$r_{i,x,j} = \frac{g_{i,x,j} \cdot p_j}{g_{i,x,j} \sum_{u_t \in U_{i,x}(\alpha) \setminus u_j} p_t + F_{i,x,j} + \omega} \tag{3}$$

where p_j is the signal transmission power required by u_j, $I_{i,x,j}$ is the interference from other edge servers covering u_j, calculated by $F_{i,x,j} = \sum_{v_o \in V_j \setminus v_i} \sum_{u_t \in U_{o,x}(\alpha)} g_{i,x,t} \cdot p_t$ and ω is the additive white Gaussian noise [4, 6]. Please note that the SINR can be calculated based on other wireless communication models based on the actual networking environment. It will not impact the IDDE problem or the performance of the proposed approaches fundamentally.

According to [7, 16, 41], u_j's actual data rate received from edge server v_i on $c_{i,x}$ is:

$$R_{i,x,j} = B_{i,x} \cdot log_2(1 + r_{i,x,j}) \tag{4}$$

where $B_{i,x}$ is the bandwidth of $c_{i,x}$.

Under the Shannon capacity constraint, a user's data rate is limited by a upper bound in any mobile network [17]. Let $R_{j,max}$ denote this maximum data rate of user u_j. In this way, we can obtain the actual data rate of u_j allocated to edge server v_i on channel $c_{i,x}$, denoted by R_j:

$$R_j = \min\{R_{j,max}, I_{\{\alpha_j \neq (0,0)\}} \sum_{v_i \in V} \sum_{c_{i,x} \in C_i} R_{i,x,j}\} \tag{5}$$

where $I_{\{condition\}}$ equals to 0 if $condition$ is false, and 1 otherwise.

Given M users, their average data rate can be calculated:

$$R_{ave} = \frac{\sum_{u_j \in U} R_j}{M} \tag{6}$$

Table 1: Summary of Notations

Notation	Description
A_i	available storage spaces on v_i
$B_{i,x}$	bandwidth of $c_{i,x}$
C_i	set of channels on v_i
$c_{i,x}$	channel x on v_i
$cloud$	remote cloud
D	set of data
d_k	data k
$g_{i,x,j}$	channel gain between u_j and $c_{i,x}$
K	number of data
L_{ave}	users' average data delivery latency
$L_{k,o,i}$	the latency from v_o to v_i for delivering d_k
M	number of users
N	number of edge servers
p_j	power required by u_j
R_{ave}	users' average data rate
R_j	u_j's data rate
$r_{i,x,j}$	u_j's SINR on $c_{i,x}$
$R_{i,x,j}$	u_j's data rate on $c_{i,x}$
s_k	data size of d_k
U	set of users
U_i	set of users covered by v_i
$U_{i,x}(\alpha)$	set of users allocated to $c_{i,x}$ based on α
u_j	user j
V	set of edge servers
V_j	set of edge servers covering u_j
v_i	edge server i
α	user allocation profile
$\alpha_j = (i, x)$	binary variable indicating whether u_j is allocated to $c_{i,x}$
α_j	user allocation decision of u_j
σ	data delivery profile
$\sigma_{i,k}$	binary variable indicating whether d_k is stored on v_i
$\zeta_{j,k}$	binary variable indicating whether u_j requests d_k
ω	additive white Gaussian noise

2.3 Problem Formulation and Hardness

As discussed in Section 2.2, an inappropriate allocation of users can impact users' data rates by severe interference. Thus, for app vendors, the first optimization of the IDDE problem, i.e., IDDE Objective #1, is to maximize users' average data rate $\max R_{ave}$, calculated with Eq. (6).

On edge server v_i, the volume of data saved must not exceed the storage spaces reserved on v_i, denoted as A_i. This is referred as to the *storage constraint*:

$$\sum_{d_k \in D} \sigma_{i,k} \cdot s_k \leq A_i, \forall v_i \in V \tag{7}$$

where s_k is the size of data d_k.

Any data can be retrieved from the app vendor's remote cloud server, represented as:

$$\sigma_{cloud,k} = 1, \forall d_k \in D \tag{8}$$

Let $L_{k,o,i}$ denote the lowest latency of delivering data d_k from edge server v_o to edge server v_i in the system. When user u_j retrieves d_k, the corresponding latency is:

$$L_{j,k}(\alpha_j, \sigma) = \min\{L_{k,o,i} | \sigma_{o,k} = 1, \forall v_o \in V \bigcup cloud\} \tag{9}$$

Please note that (9) ensures that delivering the data from an edge server in the system must not take longer than from the remote cloud. This is referred to the *latency constraint*.

All the M users' average data delivery latency can be calculated as follows:

$$L_{ave} = \frac{\sum_{u_j \in U} \sum_{d_k \in D} \zeta_{j,k} \cdot L_{j,k}(\alpha_j, \sigma)}{\sum_{u_j \in U} \sum_{d_k \in D} \zeta_{j,k}} \tag{10}$$

where $\zeta_{j,k} \in \{0, 1\}$ indicates whether data d_k is requested by user u_j.

Given the user allocation profile α, the data delivery profile σ must minimize users' average data delivery latency, i.e., IDDE objective #2: min L_{ave}. In this way, the IDDE problem can be formulated as follows:

$$\textbf{Objective \#1} \quad \max \quad R_{ave} \tag{11}$$

$$\textbf{Objective \#2} \quad \min \quad L_{ave} \tag{12}$$

$$s.t.: \quad (1), (5), (7), (8), (9)$$

To ensure users' quality of experience, app vendors need to maximize their users' data rate (Objective #1) and minimize their users' data delivery latency (Objective #2).

Next, we prove that the NP-hardness of this IDDE problem.

Theorem 1. *The IDDE problem is NP-hard.*

Proof. The IDDE problem is a multiple-objective optimization problem. To prove its NP-hardness, we need to prove that it is NP-hard to solve any one of its objectives [10].

Now, we first prove that the IDDE problem with Objective #1 is NP-hard. Here, we introduce the NP-hard problem named minimum routing cost spanning tree (MRCS) [33]. The MRCS problem aims to cover all vertices with minimum total distance via a spanning tree in a weighted graph. Given an instance of the IDDE problem and the number of users, Objective #1 becomes max $\sum_{u_j \in U} R_j$, i.e., maximizing users' overall data rate. Then, we convert this objective equally to min $\sum_{u_j \in U}(R_{j,max} - R_j)$. Let us assume that there is an edge between an edge server and each of the users within its coverage area and the distance of this edge is calculated by $R_{j,max} - R_j$. In this way, the IDDE problem with Objective #1 is to cover all the users with the minimum total distance via a spanning tree. Thus, the MRCS problem is reduced to an instance of the IDDE problem with Objective #1, and the IDDE problem with Objective #1 is thus NP-hard.

Without Objective #1, the IDDE problem with Objective #2 is to store the required data in reserved storage spaces with the aim to minimize users' average data delivery latency. The NP-hardness of this problem can be proved based on the weighted k-set packing (WKSP) problem [1], similar to the proof presented in [34]. Thus, the details are omitted here.

According to the above proofs, the multiple-objective optimization problem, IDDE, is NP-hard. □

3 ALGORITHM DESIGN AND ANALYSIS

As the IDDE problem scales up, it is intractable to optimally solve the IDDE problem due to its NP-hardness proved in Section 2.3. Pursuing optimal IDDE strategies is particularly infeasible in the EC environment where low latency is a fundamental requirement. To solve this problem, we present a game-based algorithm named IDDE-G to find sub-optimal IDDE strategies efficiently. After that, the performance of IDDE-G is analyzed theoretically.

3.1 IDDE-G Design

As introduced in Section 2.3, the IDDE problem has two optimization objectives: maximizing users' average data rate (11) and minimizing their average data delivery latency (12). IDDE-G solves the IDDE problem via two phases: Phase #1) constructing a game, named IDDE-U, to find a Nash equilibrium as the user allocation profile to maximize their average data rate (Objective #1); and Phase #2) finding a data delivery profile to deliver data to edge servers with reserved storage spaces with the minimum average data delivery latency (Objective #2).

The IDDE-U game aims to maximize users' average data rate. Here, we define the benefit function for user u_j:

$$\beta_{\alpha_{-j}}(\alpha_j) = \frac{g_{i,x,j} \cdot p_j}{g_{i,x,j} \sum_{u_t \in U_{i,x}(\alpha)} p_t + F_{i,x,j}} \tag{13}$$

According to (3) and (4), this benefit function will drive the allocation of u_j to the edge server that can offer the highest data rate.

In the IDDE-U game, there might be conflicts among users. Take Fig. 2 as an example. For easy of exposition, let us assume that there is only one channel available on each server. If u_3 is allocated to v_1, it is subject to interference from u_1 because u_1 can only be allocated to v_1. This may lower u_3 and u_1's benefits according to (13). As discussed in Section 2.2, any users allocated to the same channel are subject to the interference. The goal of the IDDE-U game is to mitigate the conflicts among users and maximize the average data rate. The key is whether IDDE-U admits to a Nash equilibrium [25].

Definition 3 (Nash Equilibrium). *A user allocation profile α^* is a Nash equilibrium if $\beta_{\alpha^*_{-j}}(\alpha^*_j) \geq \beta_{\alpha^*_{-j}}(\alpha_j), \forall u_j \in U, \alpha_j \in \delta_j$, where δ_j is the set of possible allocation decisions for u_j.*

The Nash equilibrium found in the IDDE-U game can be enforced by edge servers as the user allocation profile without a centralized control [12]. To investigate its existence, a possible method is to demonstrate that IDDE-U is a potential game [23].

Definition 4 (Potential Game). *For a potential function $\pi(\alpha)$, a potential game must satisfy $\beta_{\alpha_{-j}}(\alpha_j) < \beta_{\alpha_{-j}}(\alpha'_j) \Rightarrow \pi(\alpha_j, \alpha_{-j}) < \pi(\alpha'_j, \alpha_{-j})$, for any $u_j \in U$, $\alpha_j, \alpha'_j \in \delta_j$ and $\alpha_{-j} \in \prod_{l \neq j} \delta_l$.*

Here, we introduce Lemma 2 for proving that the IDDE-U game is a potential game:

Lemma 2. *Given a user allocation profile α, the interference received by a user u_j is not higher than T_j, if this user can be allocated*

to channel $c_{i,x}$. Let U_i denote the set of users covered by v_i, the value of T_j is $\frac{g_{i,x,j}p_j}{2^{\frac{R_{j,min}}{B_{i,x}}}-1} - \omega$, where $R_{j,min} = \min\{R_{i,x,j}, \forall c_{i,x} \in C_i, v_i \in V_j\}$.

According to (4), Lemma 2 can be easily proved, and we omit the details here. Based on Lemma 2, the potential function of the IDDE-U game is below:

$$\pi(\alpha_j, \alpha_{-j}) = \frac{1}{2} \sum_{u_j \in U} \sum_{u_q \in U \setminus u_j} I_{\{\alpha_j \neq (0,0)\}} \cdot I_{\{\alpha_q \neq (0,0)\}} \cdot \frac{g_{i,x,j}p_j}{g_{i,x,j} \sum_{u_t \in U_{i,x}(\alpha)} \cdot p_t + F_{i,x,t}} \cdot \frac{g_{i_q,x_q,q}p_q}{g_{i_q,x_q,t} \sum_{u_t \in U_{i_q,x_q}(\alpha)} p_t + F_{i_q,x_q,t}} - T_j \cdot I_{\{\alpha_j = (0,0)\}} \cdot \frac{g_{i_q,x_q,q}p_q}{g_{i_q,x_q,t} \sum_{u_t \in U_{i_q,x_q}(\alpha)} p_t + F_{i_q,x_q,t}} \quad (14)$$

where $\alpha_q = (i_q, x_q)$.

Theorem 3. *With the potential function $\pi(\alpha)$ (14), the IDDE-U game is a potential game.*

Proof. Supposing two user allocation decisions $\alpha_j = (i, x)$ and $\alpha'_j = (i', x')$, fulfilling $\beta_{\alpha_{-j}}(\alpha_j) < \beta_{\alpha_{-j}}(\alpha'_j)$, for user u_j. As $g_{i,x,j}$ is decided by edge server's physical nature and does not impact IDDE-U, we assume that $g_{i,x,j}$ is the same for different users in the proof, i.e., $g_{i,x,j} = g$. We will evaluate the performance with heterogeneous edge servers in Section 4. To prove this theorem, there are two cases according to (13): 1) $\alpha_j \neq (0,0)$ and $\alpha'_j \neq (0,0)$; and 2) $\alpha_j = (0,0)$ and $\alpha'_j \neq (0,0)$.

Case 1: $\alpha_j \neq (0,0)$ and $\alpha'_j \neq (0,0)$.

Given $\beta_{\alpha_{-j}}(\alpha_j) < \beta_{\alpha_{-j}}(\alpha'_j)$, we can obtain the following inequality based on (13):

$$\frac{p_j}{\sum_{v_i \in V_j} \sum_{u_q \in U_{i,x}(\alpha)} p_q} < \frac{p_j}{\sum_{v'_i \in V_j} \sum_{u_q \in U_{i',x'}(\alpha)} \cdot p_q} \quad (15)$$

Thus, the difference between the potentials produced by α_j and α'_j calculated with Eq. (14) can be calculated with:

$$\pi(\alpha_j, \alpha_{-j}) - \pi(\alpha'_j, \alpha_{-j}) = \sum_{u_q \in U \setminus u_j} p_q \cdot I_{\{\alpha_q \neq (0,0)\}} \cdot \left(\frac{p_j}{\sum_{v_i \in V_j} \sum_{u_q \in U_{i,x}(\alpha)} p_q} - \frac{p_j}{\sum_{v'_i \in V_j} \sum_{u_q \in U_{i',x'}(\alpha)} \cdot p_q} \right) < 0 \quad (16)$$

Case 2: $\alpha_j = (0,0)$ and $\alpha'_j \neq (0,0)$.

Similar to Case 1, we can obtain:

$$\pi(\alpha_j, \alpha_{-j}) - \pi(\alpha'_j, \alpha_{-j}) = -\left(\frac{p_j}{\sum_{v_i \in V_j} \sum_{u_t \in U_{i',x'}(\alpha)} \cdot p_t} + T_j \right) \cdot \sum_{u_q \in U} \frac{p_q}{\sum_{v_i \in V_j} \sum_{u_t \in U_{i_q,x_q}(\alpha)} p_t} < 0 \quad (17)$$

Therefore, Theorem 3 holds. □

Now that the IDDE-U game is proved to a potential game, a Nash equilibrium can be reached within finite iterations [23] as the user allocation profile at Phase #1 of IDDE-G. After finding the user allocation profile to maximize the average data rate at Phase #1, IDDE-G applies heuristic process to store data in reserved spaces on edge servers at Phase #2, aiming to minimize the average delivery latency.

The IDDE-G algorithm starts with initializing the user allocation profile α and the data delivery profile σ (Lines 1-4). In the IDDE-U game (Phase #1 of the IDDE-G algorithm), the algorithm creates an empty set of allocation decisions δ_j for each user $u_j \in U$, then includes all possible decisions into δ_j (Lines 7-12). After that, it attempts to find u_j's optimal decision $\alpha'_j \in \delta_j$ that produces the highest benefit calculated by (13) (Line 13). If u_j's benefit produced by this decision is higher than that produced by the current one, it is submitted as a update request (Lines 14-19). This iteration process ends if no any decision needs to be updated (Line 21). When Phase #1 ends, the IDDE-G algorithm starts to produce a data delivery process (Phase #2 of the IDDE-G algorithm). In each iteration of Phase #2, it always includes the data delivery decision $\sigma_{i,k}$ with the highest ratio of latency reduction over used spaces for all users, i.e., $\frac{L(\sigma)-L(\sigma \bigcup \sigma_{i,k})}{s_k}$, into σ. This process ends when the storage space constraint (7) is violated or there is no available decision $\sigma_{i,k}$ (Lines 23-26). At the end of Algorithm 1, α and σ constitute the final IDDE strategy as the user allocation profile and the data delivery profile, respectively (Line 27).

Algorithm 1 IDDE-G Algorithm

1: initialization
2: set the user allocation profile $\alpha = \{\alpha_1, \cdots, \alpha_M\}$, where the allocation decision for each user $u_j \in U$ is $\alpha_j = (0,0)$
3: set an empty data delivery profile $\sigma \leftarrow \emptyset$
4: end of initialization
– **Phase #1** IDDE-U Game for User Allocation Profile –
5: **repeat**
6: **for all** $u_j \in U$ **do**
7: create $\delta_j \leftarrow \emptyset$
8: **for all** $v_i \in V_j$ **do**
9: **for all** $c_{i,x} \in C_i$ **do**
10: $\delta_j \leftarrow \delta_j \bigcup (i,x)$
11: **end for**
12: **end for**
13: find $\alpha'_j \in \delta_j$ producing the highest benefit for user u_j according to the current α
14: **if** $B_{\alpha_{-j}}(\alpha_j) < B_{\alpha_{-j}}(\alpha'_j)$ **then**
15: submit α'_j as a update decision candidate
16: **if** u_j is the winner **then**
17: $\alpha_j \leftarrow \alpha'_j$
18: **end if**
19: **end if**
20: **end for**
21: **until** no decision updates submitted
– **Phase #2** Heuristic Process for Data Delivery Profile –
22: create $\sigma_{i,k} \leftarrow \emptyset$
23: **repeat**
24: $\sigma \leftarrow \sigma \bigcup \sigma_{i,k}$
25: according to α, obtain the data delivery decision $\sigma_{i,k}$ producing the highest ratio of latency reduction over used storage spaces under the storage constraint (7):

$$\sigma_{i,k} = \arg\max\{\frac{L(\sigma) - L(\sigma \bigcup \sigma_{i,k})}{s_k}, \forall v_i \in V, d_k \in D\} \quad (18)$$

26: **until** no feasible delivery decision $\sigma_{i,k}$
27: **return** α and σ

3.2 Algorithm Complexity Analysis

As a potential game, an IDDE-U game will complete after finite iterations and achieve a Nash equilibrium [23]. Let $T_{max} \triangleq \max(T_j)$, $T_{min} \triangleq \min(T_j)$, $Q_j \triangleq g_{i,x,j}p_j$, $Q_{min} \triangleq \min(Q_j)$, $Q_{max} \triangleq \max(Q_j)$, and Y be the total number of iterations, Theorem 4 quantifies the upper bound of Y.

Theorem 4. *For any $u_j \in U$, the maximum number of iterations Y is no more than $\frac{M(Q^2_{max} - Q^2_{min})}{2Q_{min}}$.*

Proof. According to Eq. (14), we can obtain $\frac{1}{2}\sum_{u_j\in U} Q_{min}\cdot Q_{min} - T_{max}\sum_{u_j\in U} Q_{max} \le \pi(\alpha_j,\alpha_{-j})$ $\le \frac{1}{2}\sum_{u_j\in U} Q_{max}\cdot Q_{max} - T_{min}\sum_{u_j\in U} Q_{min}$. That is,

$$\frac{1}{2}MQ_{min}^2 - MT_{max}Q_{max} \le \pi(\alpha_j,\alpha_{-j}) \le \frac{1}{2}MQ_{max}^2 - MT_{min}Q_{min} \tag{19}$$

If the allocation decision for user u_j is updated from α_j to α'_j, u_j's benefit should increase, i.e., $\beta_{\alpha_{-j}}(\alpha_j) < \beta_{\alpha_{-j}}(\alpha'_j)$. Based on Definition 4, there is also an increase in the potential with $\pi(\alpha_j,\alpha_{-j})$, denoted by ε_i:

$$\pi(\alpha_j,\alpha_{-j}) + \varepsilon_i \le \pi(\alpha'_j,\alpha_{-j}) \tag{20}$$

Now we try to prove $\varepsilon_i = Q_j$ for obtaining $\min_{u_j\in U}(\varepsilon_i) = Q_{min}$, where $\min_{u_j\in U}(\varepsilon_i)$ is the minimum potential increase by updating an individual allocation decision. Based on the proof of Theorem 3, two cases need to be discussed when a decision is updated for a user: 1) $\alpha_j \neq (0,0)$ and $\alpha'_j \neq (0,0)$; and 2) $\alpha_j = (0,0)$ and $\alpha'_j \neq (0,0)$.

Case 1: $\alpha_j \neq (0,0)$ and $\alpha'_j \neq (0,0)$. Based on (16), we can obtain:

$$\pi(\alpha'_j,\alpha_{-j}) - \pi(\alpha_j,\alpha_{-j}) = Q_j \cdot \Big(\sum_{u_q\in U_{i,x}(\alpha)\setminus u_j} Q_q \cdot I_{\{\alpha_j\neq(0,0)\}} - \sum_{u_q\in U_{i',x'}(\alpha)\setminus u_j} Q'_q \cdot I_{\{\alpha'_j\neq(0,0)\}}\Big) > 0 \tag{21}$$

Since $Q_j > 0$ for $u_j \in U$ and Q_j is an integer, we can obtain $\sum_{u_q\in U_{i,x}(\alpha)\setminus u_j} Q_q\cdot I_{\{\alpha_j\neq(0,0)\}} - \sum_{u_q\in U_{i',x'}(\alpha)\setminus u_j} Q'_q\cdot I_{\{\alpha'_j\neq(0,0)\}} \ge 1$. Thus, according to Eq. (21), there is:

$$\pi(\alpha'_j,\alpha_{-j}) \ge \pi(\alpha_j,\alpha_{-j}) + Q_j \ge \pi(\alpha_j,\alpha_{-j}) + Q_{min}$$

Case 2: $\alpha_j = (0,0)$ and $\alpha'_j \neq (0,0)$. According to (17), we obtain:

$$\pi(\alpha'_j,\alpha_{-j}) - \pi(\alpha_j,\alpha_{-j}) = Q_j \cdot \Big(T_j - \sum_{u_q\in U_{i',x'}(\alpha)\setminus u_j} Q_q \cdot I_{\{\alpha'_j\neq(0,0)\}}\Big) > 0 \tag{22}$$

Similar to Case 1, there is:

$$\pi(\alpha'_j,\alpha_{-j}) \ge \pi(\alpha_j,\alpha_{-j}) + Q_j \ge \pi(\alpha_j,\alpha_{-j}) + Q_{min} \tag{23}$$

Therefore, according to (19) and (20), there is:

$$Y \le \Big(\big(\frac{1}{2}MQ_{max}^2 - MT_{min}Q_{min}\big) - \big(\frac{1}{2}MQ_{min}^2 - MT_{max}Q_{max}\big)\Big)/Q_{min} = M(T_{min}Q_{min} - T_{max}Q_{max}))/Q_{min} + \frac{1}{2Q_{min}}M(Q_{max}^2 - Q_{min}^2) \le \frac{M(Q_{max}^2 - Q_{min}^2)}{2Q_{min}} \tag{24}$$

That is, $Y \le \frac{M(Q_{max}^2-Q_{min}^2)}{2Q_{min}}$ and Theorem 4 holds. □

Now, we analyze the computational complexity of IDDE-G. In Phase #1, the computational complexity of the iteration process in Lines 7-19 is $O(NK)$. Since the maximum number of iterations is at most $\frac{M(Q_{max}^2-Q_{min}^2)}{2Q_{min}}$ in Phase #1, where Q_{max} and Q_{min} are constants, the computational complexity of Phase #1 is $O(NMK)$. In Phase #2, the computational complexity of finding delivery decisions in Lines 25 is at most $O(NK)$. Since there are at most $\sum_{v_i\in V} A_i$ iterations in Lines 23-26, the computational complexity of Phase #2 is $O(NK\sum_{v_i\in V} A_i) = O(N^2K)$. Thus, the computational complexity of IDDE-G is $O(N^2K + NMK) = O(NK\max\{N,M\})$.

3.3 Algorithm Performance Analysis

Here, we first evaluate the performance of IDDE-G in maximizing users' average data rate in Phase #1 (IDDE Objective #1) by analyzing its Price of Anarchy (POA), measured by the distance between the global optimal user allocation profile and the worst utility of any Nash equilibrium [27]. After that, we analyze the approximation rate of IDDE-G in minimizing users' average data delivery latency in Phase #2 (IDDE Objective #2), measured by the ratio of the average data delivery latency incurred by IDDE-G over that achieved with the optimal data delivery profile.

Theorem 5 (POA in Average Data Rate). *Given the global optimal user allocation profile α^* and the user allocation profile α in the IDDE-U game, the POA of IDDE-U game in terms of the average data rate, denoted by ρ, fulfills $\frac{R_{min}}{R_{max}} \le \rho \le 1$, where $R_{min} = \min\{R_{j,min}, u_j \in U\}$ and $R_{max} = \min\{R_{j,max}, u_j \in U\}$.*

Proof. Given R_{min} and R_{max}, the POA of IDDE-U game in terms of the average data rate can be calculated by:

$$\rho = \frac{\sum_{u_j\in\mathcal{U}} \beta_{\alpha_{-j}}(\alpha_j)}{\sum_{u_j\in\mathcal{U}} \beta_{\alpha^*_{-j}}(\alpha_j)^*} \ge \frac{R_{min}\sum_{s_i\in S} num_{s_i}(\alpha)}{R_{max}\sum_{s_i\in S} num_{s_i}(\alpha^*)} \tag{25}$$

where $num_{s_i}(\alpha)$ is the number of users allocated by α.

Since all the users can be allocated in IDDE scenarios, we can obtain $\rho \ge \frac{R_{min}}{R_{max}}$. Moreover, the average data rate achieved by IDDE-U game cannot be higher than that achieved by the optimal user allocation profile α^*. Thus, Theorem 5 holds. □

Now, we analyze the performance of IDDE-G on IDDE Objective #2, i.e., minimizing users' average data delivery latency. Let σ^* denote the optimal data delivery profile that incurs the lowest average data delivery latency according to α, and σ denote the data delivery profile produced by IDDE-G. In this way, the total latency reduction achieved by σ, denoted by $\Delta L(\sigma)$, can be calculated with:

$$\Delta L(\sigma) = \varphi - L(\sigma) \tag{26}$$

where φ is the total data delivery latency incurred by all users retrieving data from the remote cloud.

Here, we divide each data delivery decision $\sigma_{i,k}$ from IDDE-G to obtain a set of $|s_k|$ sub-decisions. Let σ' and σ'^* denote the sets of sub-decisions obtained from IDDE-G's data delivery profile σ and the optimal data delivery profile σ^*, respectively. Let σ'_z denote the set of sub-decisions with the z^{th} sub-decision included, and $L(\sigma)$ denote the total latency incurred by the data delivery profile σ. In this way, we can obtain $L(\sigma^*) = L(\sigma'^*)$.

Theorem 6. *The latency reduced by σ'_z is at least $\frac{e-1}{2e}$ times that reduced by σ'^*_z if $z = \sum_{v_i\in V} A_i$.*

Proof. Here ΔL_z denotes the latency reduced by including the z^{th} sub-decision into σ_z. The first sub-decision in $\sigma'^*_z \bigcap \neg\sigma'_{z-1}$ can reduce at most ΔL_z from the total latency. This is because, with the z^{th} sub-decision included into σ', the sub-decision reducing the maximum latency is selected by σ'_z. Since the total reserved storage spaces are $\sum_{v_i\in V} A_i$ in the edge storage system, the set of decisions $\sigma'^*_z \bigcap \neg\sigma'_{z-1}$ can reduce at most $\sum_{v_i\in V} A_i\cdot\Delta L_z$ from the total latency. Thus, we can obtain:

$$\sum_{v_i\in V} A_i\cdot\Delta L_z \ge \Delta L(\sigma'^*_z) - \Delta L(\sigma'_{z-1}) \tag{27}$$

According to (27), the latency reduced by σ'_z can be calculated as follows:

$$\begin{aligned}\Delta L(\sigma'_z) = \Delta L_z + \Delta L(\sigma'_{z-1}) \geq \frac{\Delta L(\sigma'^*_z) - \Delta L(\sigma'_{z-1})}{\sum_{v_i \in V} A_i} \\ + \Delta L(\sigma'_{z-1}) = \frac{\Delta L(\sigma'^*_z)}{\sum_{v_i \in V} A_i} + \left(1 - \frac{1}{\sum_{v_i \in V} A_i}\right)\Delta L(\sigma'_{z-1})\end{aligned} \tag{28}$$

Based on mathematical induction, (29) can be easily proved, according to (28). The details of the proof are omitted here.

$$\frac{\Delta L(\sigma'_z)}{\Delta L(\sigma'^*_z)} = \left(1 - \left(1 - \frac{1}{\sum_{v_i \in V} A_i}\right)^{z-1}\right) \tag{29}$$

Let $z = \sum_{v_i \in V} A_i$. The latency reduced by σ'_{z+1} can be calculated by:

$$\begin{aligned}\frac{\Delta L(\sigma'_{z+1})}{\Delta L(\sigma'^*_z)} \geq \left(1 - \left(1 - \frac{1}{\sum_{v_i \in V} A_i}\right)^{z}\right) \\ = \left(1 - \left(1 - \frac{1}{\sum_{v_i \in V} A_i}\right)^{\sum_{v_i \in V} A_i}\right) \geq 1 - \frac{1}{e}\end{aligned} \tag{30}$$

Since the latency reduced by σ'_{z+1} is not higher than the total latency reduced by σ'_z, we can obtain:

$$\Delta L(\sigma'_z) \geq \frac{1}{2}\Delta L(\sigma'_{z+1}) \geq \frac{(e-1)\Delta L(\sigma'^*_z)}{2e} \tag{31}$$

Therefore, the theorem is proved based on (31). □

Please note that the sub-decision number of σ' is not always as many as $\sum_{v_i \in V} A_i$. In the worst case, the largest data, i.e., the one with s_{max}, is to be delivered to every edge server while no edge server has enough available storage spaces. In this case, a total of $N \cdot s_{max}$ sub-decisions cannot be included into σ.

Theorem 7. *The actual average data delivery latency incurred by data delivery profile σ fulfills:*

$$L(\sigma) \leq \left(\frac{e+1}{2e} + \frac{e-1}{2e} \cdot \frac{N \cdot s_{max}}{\sum_{v_i \in V} A_i}\right)\varphi + \left(1 - \frac{N \cdot s_{max}}{\sum_{v_i \in V} A_i}\right)\frac{e-1}{2e}L(\sigma^*) \tag{32}$$

where $s_{max} = \max\{s_k | \forall d_k \in D\}$.

Proof. As discussed above, a total of $N \cdot s_{max}$ sub-decisions cannot be included into σ in the worst case. In this way, the total data delivery latency reduced by σ in this case is:

$$\begin{aligned}\Delta L(\sigma) \geq \Delta L(\sigma'_{\sum_{v_i \in V} A_i}) - \frac{N \cdot s_{max}}{\sum_{v_i \in V} A_i}\Delta L(\sigma'_{\sum_{v_i \in V} A_i}) \\ \overset{(31)}{\geq} \left(1 - \frac{N \cdot s_{max}}{\sum_{v_i \in V} A_i}\right)\frac{e-1}{2e}\Delta L(\sigma^*)\end{aligned} \tag{33}$$

Based on (26) and (33), we can obtain:

$$\begin{aligned}L(\sigma) \leq \varphi - \left(1 - \frac{N \cdot s_{max}}{\sum_{v_i \in V} A_i}\right)\frac{e-1}{2e}\Delta L(\sigma^*) = \left(\frac{e+1}{2e} + \right. \\ \left.\frac{e-1}{2e} \cdot \frac{N \cdot s_{max}}{\sum_{v_i \in V} A_i}\right)\varphi + \left(1 - \frac{N \cdot s_{max}}{\sum_{v_i \in V} A_i}\right)\frac{e-1}{2e}L(\sigma^*)\end{aligned} \tag{34}$$

Thus, (32) stands and Theorem 7 is proved. □

Let $L_{avg}(\sigma)$ and $L_{avg}(\sigma^*)$ denote the average data retrieval latencies achieved by IDDE-G and the optimal strategy, respectively. The total number of data requests is $\sum_{u_j \in U} \sum_{d_k \in D} \zeta_{j,k}$ and the average data delivery latency (IDDE Objective #2) achieved by IDDE-G is at most:

$$\begin{aligned}L_{avg}(\sigma) \leq \left(\frac{e+1}{2e} + \frac{e-1}{2e} \cdot \frac{N \cdot s_{max}}{\sum_{v_i \in V} A_i}\right)\frac{\varphi}{\sum_{u_j \in U} \sum_{d_k \in D} \zeta_{j,k}} \\ + \left(1 - \frac{N \cdot s_{max}}{\sum_{v_i \in V} A_i}\right)\frac{e-1}{2e}L_{avg}(\sigma^*)\end{aligned} \tag{35}$$

4 EXPERIMENTAL EVALUATION

All experiments are conducted on a machine equipped with an Intel i7 processor (4 cores) and 8GB RAM.

4.1 Benchmark Approaches

The performance of IDDE-G is evaluated against four representative approaches:

- **IDDE-IP**: This approach solves the IDDE model formulated in Section 2.3 with IBM CPLEX[1]. However, due to the NP-hardness of the IDDE problem, it is intractable to find the optimal IDDE strategy within a reasonable time period. The execution time of the CP Optimizer consists of searching time and tuning time[2]. In the experiments, its maximum searching time is limited by 100 seconds.
- **SAA (Sample Average Approximation approach)**: This approach origins from [24]. Each edge server makes data delivery decisions based on the data requests coming from its coverage, aiming to maximize storage utility including data delivery latency and user coverage.
- **CDP (Centralized Data Placement approach)** [19]: This greedy approach finds a near-optimal strategy to minimize the data delivery latency, based on the same communication model presented in Section 2.2.
- **DUP-G**: This game-theoretical approach origins from [36]. It aims to maximize users' average data rate. It always finds a Nash equilibrium as the near-optimal strategy by allocating each user to the edge server directly covering the user, based on the communication model presented in Section 2.2.

4.2 Dataset

The real-world EUA dataset[3] is used to conduct the experiments. An EC environment is simulated with 125 edge servers and 816 users, extracted from the EUA dataset. In the experiments, each requested data is randomly sized from {30, 60, 90}MB while the reserved storage spaces on individual edge servers is randomly sized from 30MB to 300MB. The transmission speed among edge servers is selected randomly from $[2,000, 6,000]$MBps and that between edge servers and cloud is 600MBps. Following similar experiment settings as [5, 32], each edge server has 3 channels, each with a bandwidth of 200MBps and background noise $\omega = -174dBm$. Each user device's power is randomly selected from $[1, 5]$ Watts. In

[1]https://www.ibm.com/au-en/analytics/cplex-cp-optimizer
[2]https://www.ibm.com/docs/en/SSSA5P_12.8.0/ilog.odms.studio.help/pdf/usrcpoptimizer.pdf
[3]https://github.com/swinedge/eua-dataset

Table 2: Parameter Settings

	N	M	K	$density$
Set #1	20, 25, 30, 35, 40, 45, 50	200	5	1.0
Set #2	30	50, 100, 150, 200, 250, 300, 350	5	1.0
Set #3	30	200	2, 3, 4, 5, 6, 7, 8	1.0
Set #4	30	200	5	1.0, 1.4, 1.8, 2.2, 2.6, 3.0

addition, the setting of $\eta = 1$ and $loss = 3$ is employed for IDDE-G to calculate the channel gain $g_{i,x,j}$ with $g_{i,x,j} = \eta \cdot H_{i,j}^{-loss}$.

4.3 Parameter Settings

To evaluate the performance of IDDE-G in various IDDE scenarios, we conduct four experiment sets. In each set, one setting parameter varies while the other three are fixed, to analyze the impacts of these parameters on IDDE-G, which indicates its performance in various environments. Table 2 summarizes the parameter settings.

- Number of edge servers (N).
- Number of users (M).
- Number of data (K).
- Network density ($density$).

Given $density$ and N in a simulated edge storage system, $density \cdot N$ links are generated randomly to connect edge servers. Each experiment is run 50 times each time a parameter changes and the average results are reported.

4.4 Performance Metrics

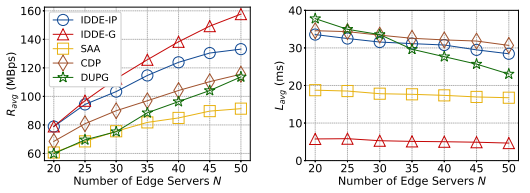
To evaluate all the approaches, three performance metrics are employed:

- Average data rate (R_{avg}). This metric is the mobile users' average data rate. It measures the ability of an approach to achieve IDDE objective #1, the higher the better.
- Average data delivery latency (L_{avg}). This metric is the mobile users' average data delivery latency. It measures the ability of an approach to achieve IDDE objective #2, the lower the better.
- Computation time ($time$). This metric is the time taken by an approach for formulate an IDDE strategy, the lower the better.

4.5 Experimental Results

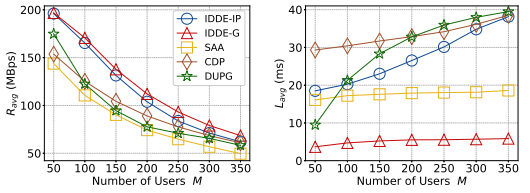
4.5.1 Effectiveness. Figures 3 - 6 demonstrate the performance of IDDE-G through comparison with IDDE-IP, SAA, CDP and DUP-G, and the impacts of the four parameters. **IDDE-G always achieves the highest average data rate with the lowest average data delivery latency.** Across all the experiments, the average advantage of IDDE-G in terms of data rate is 9.20% over IDDE-IP, 53.27% over SAA, 29.40% over CDP and 41.56% over DUP-G. In terms of the average data delivery latency, IDDE-G significantly outperforms IDDE-IP, SAA, CDP and DUP-G by 82.61%, 71.60%, 84.60% and 85.04%, respectively.

Fig. 3 depicts the experimental results of Set #1. In terms of data rate and delivery latency, IDDE-G's advantages are 10.36% and 83.16% over IDDE-IP, 55.55% and 70.42% over SAA, 28.99% and 84.05% over CDP, 41.51% and 82.76% over DUP-G on average. When N increases from 20 to 50, the average data rates achieved by five approaches increase. Since M here is fixed at 200, each



(a) Average Data Rate (R_{avg}) vs. N (b) Average Data Delivery Latency (L_{avg}) vs. N

Figure 3: Effectiveness in Set #1



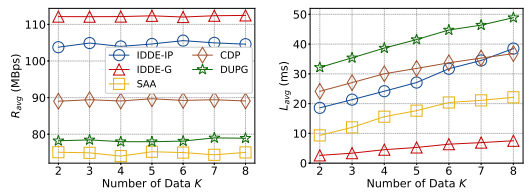
(a) Average Data Rate (R_{avg}) vs. M (b) Average Data Delivery Latency (L_{avg}) vs. M

Figure 4: Effectiveness in Set #2

edge server serves fewer users on average when more edge servers are available in the edge storage system. With a more dispersed distribution, the interference among users is reduced and their data rates is increased in **Fig. 3(a)**. In **Fig. 3(b)**, the average data retrieval latencies incurred by IDDE-G, SAA, CDP and DUP-G decrease when N increases. As N increases, more storage spaces can be reserved in the edge storage system to store more data, and more users can access requested data from nearby edge servers with the lowest data delivery latency.

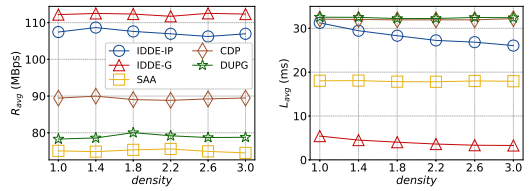
The results obtained from Set #2 are shown in **Fig. 4**. In terms of both average data rate and data delivery latency, IDDE-G outperforms others again with significant margins. In **Fig. 4(a)**,IDDE-G's advantages on average data rate are 5.47% over IDDE-IP, 45.43% over SAA, 26.32% over CDP and 29.15% over DUP-G. When M increases from 50 to 350, the average data rates achieved by IDDE-G, IDDE-IP, SAA, CDP and DUP-G decrease from 196.71MBps to 68.48MBps by 65.19%, from 196.06MBps to 62.01MBps by 68.37%, from 143.75MBps to 49.60MBps by 65.50%, from 153.62MBps to 60.87MBps by 60.38% and 174.76MBps to 58.26MBps by 66.66%, respectively. The reason is straightforward - more users result in high interference among them, which reduces their average data rate. When M increases, more data need to be delivered to each edge server to serve their nearby users. However, constrained by the fixed number of storage spaces reserved on individual edge servers, more data have to be delivered to users from edge servers faraway in the system. This causes the increase in the data delivery latencies incurred by all the approaches, as shown in **Fig. 4(b)**.

Fig. 5 demonstrates the results in Set #3. In **Fig. 5(a)**, IDDE-G consistently achieves the highest average data, 7.25%, 50.03%, 25.69%



(a) Average Data Rate (R_{avg}) vs. K (b) Average Data Delivery Latency (L_{avg}) vs. K

Figure 5: Effectiveness in Set #3



(a) Average Data Rate (R_{avg}) vs. $density$ (b) Average Data Delivery Latency (L_{avg}) vs. $density$

Figure 6: Effectiveness in Set #4

and 43.19% higher than IDDE-IP, SAA, CDP and DUP-G. In Fig. 5(a), the number of data K makes insignificant impacts on users' average data in this set of experiments. This is because K is not considered in the user allocation profile. However, K impacts users' average data delivery latency, as clearly illustrated in **Fig. 5(b)**. With the increase of K increases, the average data retrieval latencies incurred by all five approaches increase, from 2.61ms to 7.52ms for IDDE-G, from 18.58ms to 38.50ms for IDDE-IP, from 9.33ms to 22.12ms for SAA, from 24.12ms to 36.80ms for CDP and from 32.16ms to 48.88ms for DUP-G. A large number of data requested by users challenges the ability of the approaches to deliver data with low latency, in particular when the average number of storage spaces reserved on individual edge servers is fixed. Fig. 5(b) shows that IDDE-G is the clear winner in ensuring low data delivery latency for users. The average data delivery latency incurred by IDDE-G is multiple times lower than those incurred by the other approaches, i.e., 5.22ms (IDDE-G) versus 27.98ms (IDDE-IP), 16.88ms (SAA), 31.26ms (CDP) and 41.10ms (DUP-G).

Fig. 6 shows the results in Set #4. Again, IDDE-G is the clear winner, achieving the highest average data rate and the lowest average data delivery latency. In **Fig. 6(a)**, IDDE-G's performance in maximizing users' average data rate outperforms IDDE-IP, SAA, CDP and DUP-G significantly by 13.94%, 62.92%, 36.87% and 54.91% respectively. As shown in **Fig. 6(b)**, IDDE-G's performance advantage in terms of the average data delivery latency is immense in Set #4, similarly to Sets #1 - #3. On average, it outperforms IDDE-IP, SAA, CDP and DUP-G by 90.38%, 75.91%, 89.63% and 86.72%, respectively. The increase in *density* impacts users' average data delivery latency slightly but not on their average data rate. The main difference made by a larger *density* is a lower average latency between edge servers. This allows IDDE-IP to deliver data with a lower average latency. However, IDDE-G is capable of delivering data to most users with minimum latency even with a network density as low as 1.0. Thus, the increase in *density* decreases users' average data delivery latency only mildly when data are delivered by IDDE-G.

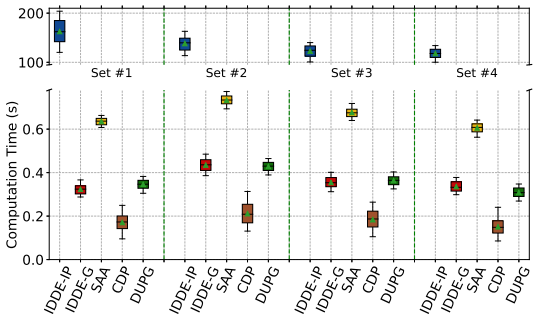


Figure 7: Computation Time (s)

4.5.2 Efficiency. The computation time taken across all the four experiment sets is shown in **Fig. 7**. Apparently and as expected, IDDE-IP always takes the most time to solve the IDDE problem in all the experiments, between 117.9126 seconds and 162.6129 seconds on average. SAA takes the second most time, between 0.6056 seconds and 0.7333 seconds on average. Across all the four experiment sets, IDDE-G, CDP and DUP-G can find a solution rapidly, compared with IDDE-IP and SAA, i.e. 0.3620 seconds (IDDE-G), 0.1691 seconds (CDP) and 0.3716 seconds (DUP-G) vs. 0.6626 seconds (SAA) and 135.3881 seconds (IDDE-IP). CDP takes the least time to solve the IDDE problem, followed by IDDE-G and DUP-G. However, such slight differences are negligible, compared with IDDE-G's profound advantage over CDP in effectiveness demonstrated above in Fig. 3 - Fig. 6.

4.5.3 Conclusion. As discussed in Section 1, the key objective of edge computing is to pursue the low latency. IDDE strategies must be formulated rapidly to ensure that users' data demands are served efficiently. Based on the analysis above, *among the five approaches, only IDDE-G can solve the IDDE problem both efficiently and effectively.*

5 RELATED WORK

In edge computing (EC), edge servers enable access to resources and services in the closest possible proximity to end-users [21]. As the same time, EC poses many new challenges, including request reliability [18], resource sharing [3].

In an edge storage system comprised of networked edge servers, data can be stored to serve end-users with low latency. Many researchers have investigated new ideas and techniques for this new edge storage system. The authors of [22] aimed to minimize the transmission delay and the data traffic to cloud in edge storage systems. They designed a two-phases approach with Gibbs sampling for solving this problem within a reasonable time. In [15], Huang et al. studied the fairness problem in edge storage systems. They introduced a data trading protocol based on the smart-contract and formulated a Stackelberg game-theoretical approach for ensuring fairness between producer and consumer. Xia et al. [37] introduced a Lyapunov-based approach named CDEC-O to solve the data storage problem in edge storage systems. The proposed CDEC-O approach minimized system cost while ensuring low data delivery latency for end-users.

Considering the unique proximity constraint in EC, a user can only retrieve data stored on its nearby edge servers covering this user [37]. Thus, users must be properly allocated to edge servers

deployed in the same area to access data with low latency. Focused on minimizing data delivery latency, existing studies have unfortunately ignored or oversimplified the interference incurred by users in the "last mile" during data delivery in edge storage systems. As a result, users' data rates cannot be ensured, which may significantly impact their quality of experience. Data rate assurance for users in the "last mile" is in fact a key difference between edge storage systems and conventional cloud storage systems. In very recent years, researchers start to take into account the impact of interference on users' quality of experience in the EC environment, such as user allocation [6] and computation offloading [41]. Specifically, the authors of [36] consider the impact of wireless interference on the data caching problem in the multi-access edge computing environment. However, the problem studied in [36] ignores edge servers' ability to collaborate.

In edge storage systems, both users' data rates and data delivery latency must be optimized to ensure their quality of experience. To the best of our knowledge, this paper is the first attempt at leveraging edge servers' ability to tackle the interference-aware data delivery (IDDE) problem.

6 CONCLUSION

In this paper, we studied a unique and critical problem in edge storage systems from the app vendor's perspective, named the Interference-aware Data Delivery at the network Edge (IDDE) problem. Our aim was to find a cost-effective IDDE strategy that optimizes users' data rate and data delivery latency. We first identified the major challenges, modeled the problem formally, and proved its NP-hardness. Then, we proposed IDDE-G, an innovative hybrid approach that combines game theory and approximation algorithm to solve this IDDE problem. According to theoretical and experimental evaluations, IDDE-G is able to solve the IDDE problem efficiently and effectively. In the future work, we will investigate the dynamics of user movements and data migrations in IDDE scenarios.

7 ACKNOWLEDGMENT

This research is partially funded by Australian Research Council Discovery Projects No. DP200102491 and Laureate Fellowship FL190100035. Feifei Chen and Qiang He are the corresponding authors of this paper.

REFERENCES

[1] Esther M Arkin and Refael Hassin. 1998. On local search for weighted k-set packing. *Mathematics of Operations Research* 23, 3 (1998), 640–648.
[2] Yunhao Bai, Zejiang Wang, Xiaorui Wang, and Junmin Wang. 2020. AutoE2E: End-to-End Real-time Middleware for Autonomous Driving Control. In *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 1101–1111.
[3] Xiaoqing Cai, Jiuchen Shi, Rui Yuan, Chang Liu, Wenli Zheng, Quan Chen, Chao Li, Jingwen Leng, and Minyi Guo. 2020. OVERSEE: Outsourcing Verification to Enable Resource Sharing in Edge Environment. In *49th International Conference on Parallel Processing-ICPP*. 1–11.
[4] Lixing Chen, Sheng Zhou, and Jie Xu. 2018. Computation peer offloading for energy-constrained mobile edge computing in small-cell networks. *IEEE/ACM Transactions on Networking* 26, 4 (2018), 1619–1632.
[5] Xu Chen, Lei Jiao, Wenzhong Li, and Xiaoming Fu. 2016. Efficient multi-user computation offloading for mobile-edge cloud computing. *IEEE/ACM Transactions on Networking* 24, 5 (2016), 2795–2808.
[6] Guangming Cui, Qiang He, Feifei Chen, Yiwen Zhang, Hai Jin, and Yun Yang. 2021. Interference-aware Game-theoretic Device Allocation for Mobile Edge Computing. *IEEE Transactions on Mobile Computing* (2021). https://doi.org/10.1109/TMC.2021.3064063
[7] Guangming Cui, Qiang He, Xiaoyu Xia, Phu Lai, Feifei Chen, Tao Gu, and Yun Yang. 2020. Interference-aware SaaS user allocation game for edge computing. *IEEE Transactions on Cloud Computing* (2020).
[8] Alexandre da Silva Veith, Felipe Rodrigo De Souza, Marcos Dias de Assuncao, Laurent Lefèvre, and Julio Cesar Santos Dos Anjos. 2019. Multi-objective reinforcement learning for reconfiguring data stream analytics on edge computing. In *Proceedings of the 48th International Conference on Parallel Processing*. 1–10.
[9] Yongheng Deng, Feng Lyu, Ju Ren, Yongmin Zhang, Yuezhi Zhou, Yaoxue Zhang, and Yuanyuan Yang. 2021. SHARE: Shaping data distribution at edge for communication-efficient hierarchical federated learning. In *2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 24–34.
[10] Christian Glaßer, Christian Reitwießner, Heinz Schmitz, and Maximilian Witek. 2010. Approximability and hardness in multi-objective optimization. In *Conference on Computability in Europe*. Springer, 180–189.
[11] Yeting Guo, Fang Liu, Zhiping Cai, Li Chen, and Nong Xiao. 2020. Feel: A federated edge learning system for efficient and privacy-preserving mobile healthcare. In *49th International Conference on Parallel Processing-ICPP*. 1–11.
[12] Charles A Holt and Alvin E Roth. 2004. The Nash equilibrium: A perspective. *Proceedings of the National Academy of Sciences* 101, 12 (2004), 3999–4002.
[13] Xueshi Hou and Sujit Dey. 2020. Motion prediction and pre-rendering at the edge to enable ultra-low latency mobile 6DoF experiences. *IEEE Open Journal of the Communications Society* 1 (2020), 1674–1690.
[14] Yaodong Huang, Xintong Song, Fan Ye, Yuanyuan Yang, and Xiaoming Li. 2019. Fair and efficient caching algorithms and strategies for peer data sharing in pervasive edge computing environments. *IEEE Transactions on Mobile Computing* 19, 4 (2019), 852–864.
[15] Yaodong Huang, Yiming Zeng, Fan Ye, and Yuanyuan Yang. 2020. Fair and Protected Profit Sharing for Data Trading in Pervasive Edge Computing Environments. In *IEEE Conference on Computer Communications*. IEEE, 1718–1727.
[16] Junghoon Kim, Taejoon Kim, Morteza Hashemi, Christopher G Brinton, and David J Love. 2020. Joint optimization of signal design and resource allocation in wireless D2D edge computing. In *IEEE Conference on Computer Communications*. IEEE, 2086–2095.
[17] Tae-Suk Kim, Hyuk Lim, and Jennifer C Hou. 2006. Improving spatial reuse through tuning transmit power, carrier sense threshold, and data rate in multihop wireless networks. In *Proceedings of the 12th annual international conference on Mobile computing and networking*. 366–377.
[18] Weifa Liang, Yu Ma, Wenzheng Xu, Xiaohua Jia, and Sid Chi-Kin Chau. 2020. Reliability augmentation of requests with service function chain requirements in mobile edge-cloud networks. In *49th International Conference on Parallel Processing-ICPP*. 1–11.
[19] Juan Liu, Bo Bai, Jun Zhang, and Khaled B Letaief. 2017. Cache placement in Fog-RANs: From centralized to distributed algorithms. *IEEE Transactions on Wireless Communications* 16, 11 (2017), 7039–7051.
[20] Ying Liu, Qiang He, Dequan Zheng, Xiaoyu Xia, Feifei Chen, and Bin Zhang. 2020. Data Caching Optimization in the Edge Computing Environment. *IEEE Transactions on Services Computing*. https://doi.org/10.1109/TSC.2020.3032724
[21] Feng Lyu, Ju Ren, Nan Cheng, Peng Yang, Minglu Li, Yaoxue Zhang, and Xuemin Shen. 2020. LEAD: Large-scale edge cache deployment based on spatio-temporal WiFi traffic statistics. *IEEE Transactions on Mobile Computing* (2020).
[22] Xiao Ma, Ao Zhou, Shan Zhang, and Shangguang Wang. 2020. Cooperative service caching and workload scheduling in mobile edge computing. In *IEEE Conference on Computer Communications*. IEEE, 2076–2085.
[23] Dov Monderer and Lloyd S Shapley. 1996. Potential games. *Games and economic behavior* 14, 1 (1996), 124–143.
[24] Zhaolong Ning, Peiran Dong, Xiaojie Wang, Shupeng Wang, Xiping Hu, Song Guo, Tie Qiu, Bin Hu, and Ricky Kwok. 2020. Distributed and Dynamic Service Placement in Pervasive Edge Computing Networks. *IEEE Transactions on Parallel and Distributed Systems* (2020).
[25] Martin J Osborne and Ariel Rubinstein. 1994. *A course in game theory*. MIT press.
[26] Stephen Pasteris, Shiqiang Wang, Mark Herbster, and Ting He. 2019. Service placement with provable guarantees in heterogeneous edge computing systems. In *IEEE Conference on Computer Communications*. IEEE, 514–522.
[27] Tim Roughgarden. 2005. *Selfish routing and the price of anarchy*. Vol. 174. MIT press Cambridge.
[28] Dario Sabella, Vadim Sukhomlinov, Linh Trang, Sami Kekki, Pietro Paglierani, Ralf Rossbach, Xinhui Li, Yonggang Fang, Dan Druta, Fabio Giust, et al. 2019. Developing software for multi-access edge computing. *ETSI white paper* 20 (2019), 1–38.
[29] Tanmoy Sen and Haiying Shen. 2021. Context-aware Data Operation Strategies in Edge Systems for High Application Performance. In *50th International Conference on Parallel Processing*. 1–10.
[30] Jiacheng Shang and Jie Wu. 2020. Protecting Real-time Video Chat against Fake Facial Videos Generated by Face Reenactment. In *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 689–699.

[31] Liang Tong, Yong Li, and Wei Gao. 2016. A hierarchical edge cloud architecture for mobile computing. In *IEEE Conference on Computer Communications*. IEEE, 1–9.

[32] Kaidi Wang, Yuanwei Liu, Zhiguo Ding, Arumugam Nallanathan, and Mugen Peng. 2019. User association and power allocation for multi-cell non-orthogonal multiple access networks. *IEEE Transactions on Wireless Communications* 18, 11 (2019), 5284–5298.

[33] Bang Ye Wu, Giuseppe Lancia, Vineet Bafna, Kun-Mao Chao, Ramamurthy Ravi, and Chuan Yi Tang. 2000. A polynomial-time approximation scheme for minimum routing cost spanning trees. *SIAM J. Comput.* 29, 3 (2000), 761–778.

[34] Xiaoyu Xia, Feifei Chen, John Grundy, Mohamed Abdelrazek, Hai Jin, and Qiang He. 2021. Constrained App Data Caching over Edge Server Graphs in Edge Computing Environment. *IEEE Transactions on Services Computing* (2021). https://doi.org/10.1109/TSC.2021.3062017

[35] Xiaoyu Xia, Feifei Chen, Qiang He, Guangming Cui, John Grundy, Mohamed Abdelrazek, Athman Bouguettaya, and Hai Jin. 2021. OL-MEDC: An Online Approach for Cost-effective Data Caching in Mobile Edge Computing Systems. *IEEE Transactions on Mobile Computing* (2021). https://doi.org/10.1109/TMC.2021.3107918

[36] Xiaoyu Xia, Feifei Chen, Qiang He, Guangming Cui, John Grundy, Mohamed Abdelrazek, Xiaolong Xu, and Hai Jin. 2021. Data, User and Power Allocations for Caching in Multi-Access Edge Computing. *IEEE Transactions on Parallel and Distributed Systems* (2021). https://doi.org/10.1109/TPDS.2021.3104241

[37] Xiaoyu Xia, Feifei Chen, Qiang He, John Grundy, Mohamed Abdelrazek, and Hai Jin. 2021. Online Collaborative Data Caching in Edge Computing. *IEEE Transactions on Parallel and Distributed Systems* 32, 2 (2021), 281–294.

[38] Ji-Hoon Yun. 2015. Intra and inter-cell resource management in full-duplex heterogeneous cellular networks. *IEEE Transactions on Mobile Computing* 15, 2 (2015), 392–405.

[39] Shan Zhang, Peter He, Katsuya Suto, Peng Yang, Lian Zhao, and Xuemin Shen. 2017. Cooperative edge caching in user-centric clustered mobile networks. *IEEE Transactions on Mobile Computing* 17, 8 (2017), 1791–1805.

[40] Sai Qian Zhang, Jieyu Lin, and Qi Zhang. 2020. Adaptive distributed convolutional neural network inference at the network edge with ADCNN. In *49th International Conference on Parallel Processing-ICPP*. 1–11.

[41] Tongxin Zhu, Jianzhong Li, Zhipeng Cai, Yingshu Li, and Hong Gao. 2020. Computation scheduling for wireless powered mobile edge computing networks. In *IEEE Conference on Computer Communications*. IEEE, 596–605.