HA-EDD: Heterogeneity-Aware Edge Data Distribution with Bandwidth and Deadline Constraints

Guobing Zou[1], Lei Yang[1], Song Yang[1], Shengye Pang[1(✉)], Yanglan Gan[2], and Bofeng Zhang[3]

[1] School of Computer Engineering and Science, Shanghai University, Shanghai, China
{gbzou, leiy, yangsong, pangsy}@shu.edu.cn
[2] School of Computer Science and Technology, Donghua University, Shanghai, China
ylgan@dhu.edu.cn
[3] School of Computer and Information Engineering, Shanghai Polytechnic University, Shanghai, China
bfzhang@sspu.edu.cn

Abstract. Edge data distribution (EDD) is a core mechanism for reducing service latency and backhaul traffic in cloud–edge systems, but existing formulations often overlook data heterogeneity and temporal resource contention, leading to suboptimal scheduling under real-world capacity and hard deadline constraints. To address these challenges, we introduce the heterogeneity-aware edge data distribution (HA-EDD) problem to jointly minimize infrastructure cost and a heterogeneity-weighted delay penalty, subject to bandwidth, reachability, deadline, and temporal consistency constraints. We further show that HA-EDD is $\mathcal{NP}$-hard. To solve it efficiently, we develop a two-pronged solution framework: an exact CP-SAT-based solver, HA-EDD-OPT, for small-to-medium instances, and a topology-encoded genetic algorithm, HA-EDD-Evo, for larger instances. HA-EDD-Evo integrates heterogeneity-aware link scoring, deterministic repair, and schedule simulation to efficiently construct high-quality feasible distribution plans. Comprehensive experiments on real-world traffic and cluster traces show that HA-EDD-Evo achieves an average improvement of 12.9% in the overall objective over existing baselines, with up to 19.0% on large topologies, while effectively balancing infrastructure cost and delivery latency.

Keywords: Edge Data Distribution · Heterogeneity-Aware Optimization · Temporal Resource Contention · Hard Deadline · Heterogeneity-Aware Genetic Algorithm · Integer Programming

1 Introduction

Edge computing is widely adopted for IoT services in 5G/B5G networks, where data volumes and interactive workloads keep rising [12]. In cloud-centric architectures, requests and data traverse long-haul networks and multi-hop paths to

remote data centers, which increases end-to-end latency and can congest backhaul links [15]. Besides local execution, edge platforms cache application data (e.g., models, media assets, and software packages) to shorten response time and reduce core-network traffic [11]. In practice, these gains depend on the ability to deliver the right data from the cloud to a set of target edge servers under limited bandwidth and strict deadlines [19]. We refer to this process as *edge data distribution (EDD)*, which affects user latency and operational cost [6].

Despite extensive research on EDD, existing approaches exhibit two fundamental limitations. First, data heterogeneity is often neglected. Existing approaches treat data as homogeneous or reduce the problem to distributing a single item [18, 19], yet edge platforms serve heterogeneous content with different sensitivities and requirements for data transmission delay. For instance, real-time AI models for autonomous driving demand ultra-low delivery latency, whereas background software updates can tolerate much looser schedules. Second, bandwidth capacity and delivery latency are strictly coupled in time. Since each multi-hop transmission occupies link capacities for consecutive time slots, ignoring this temporal resource contention often yields dissemination plans that seem optimal in terms of infrastructure cost, but fail to satisfy hard deadlines under real-world bandwidth constraints [19].

To tackle these challenges, we formulate the *heterogeneity-aware edge data distribution (HA-EDD)* problem, which explicitly incorporates data heterogeneity and time-slotted bandwidth constraints. Based on the formulation of the HA-EDD problem to jointly minimize infrastructure cost and a heterogeneity-weighted delay penalty, we prove its $\mathcal{NP}$-hardness and propose a two-pronged approach based on problem scale to find optimal or near-optimal scheduling solutions. Specifically, we develop an exact CP-SAT-based solver (HA-EDD-OPT) for small- and medium-scale instances, and a scalable heuristic approach based on a topology-encoded genetic algorithm, called HA-EDD-Evo, for large-scale networks, which enhances the performance of data distribution for better satisfying the diverse delay requirements of heterogeneous data.

In summary, our main contributions are listed as follows:

- We formulate a novel edge data distribution problem called HA-EDD by considering data heterogeneity and temporal resource contention for better reflecting the situations in real applications. Also, we model it as a joint optimization problem that is proven to be $\mathcal{NP}$-hard.
- We propose a two-pronged approach to solve the $\mathcal{NP}$-hard HA-EDD problem based on problem scale. It includes an exact solver implemented with CP-SAT, named HA-EDD-OPT, for small-to-medium instances, and an improved genetic algorithm named HA-EDD-Evo, which enhances standard algorithms by incorporating heterogeneity-aware link scoring and deterministic repair mechanisms as heuristic information into initialization and scheduling operations.
- Our proposed methods achieve the best overall objective under real-world request and cluster workloads. Experimental results demonstrate the effectiveness and generality of our approaches for handling edge data distribution.

2 Related Work

In recent years, edge data distribution (EDD) has been studied as a cost-efficient dissemination service for edge caching systems. A central objective is to reduce delivery latency and infrastructure cost. Xia *et al.* formulate EDD with latency constraints and develop integer-programming and heuristic solutions [19], and later extend the formulation to the *online* setting with latency-aware control [18]. Li *et al.* propose EdgeDis to coordinate dissemination among edge servers in volatile MEC environments [6]. From an algorithmic perspective, Swain *et al.* relate EDD to network Steiner tree estimation and derive approximation algorithms for cost-effective multi-destination delivery [14]. Complementary formulations also incorporate end-to-end delay guarantees to better model strict delivery requirements [13]. However, these studies largely treat data as homogeneous, whereas practical edge platforms serve heterogeneous content with different delivery priorities and delay sensitivities.

Several studies move beyond uniform cost or latency minimization by considering heterogeneous importance or business value. User-centric approaches incorporate heterogeneous importance to maximize provider profit under resource constraints [3], and regional-importance based placement introduces topology-aware importance assessment that affects downstream latency and cache efficiency [16]. These results show that importance-aware system design can improve overall utility, but they do not directly address multi-type EDD scheduling with heterogeneous delay sensitivities.

Given the resource constraints of edge servers, scheduling under bandwidth scarcity and hard deadlines is another key challenge. Time-slotted scheduling mechanisms such as time-aware shaping provide practical support for temporal consistency in time-sensitive traffic [9], while limited link capacities make low-cost deadline-compliant delivery difficult in practice. Deadline-aware scheduling, offloading, and slack-reclamation techniques have also been explored to improve deadline compliance [1,10]. However, these studies do not fully capture the temporal resource contention in multi-hop EDD, where each transmission occupies bandwidth for consecutive time slots and can invalidate routing structures that appear efficient in static formulations.

Reliability is also important in volatile edge environments. Erasure-coding based operations improve availability while reducing storage or transmission overhead [5,8], and systems such as EdgeHydra further improve completion performance through early reconstruction [4]. Although these studies strengthen robustness and service quality, they are not designed for heterogeneity-aware EDD under hard deadlines. In particular, they do not jointly optimize data heterogeneity and time-coupled bandwidth constraints, which is the main focus of HA-EDD.

3 Approach

Figure 1 summarizes the overall flow of our approach. We first identify the system challenges in heterogeneity-aware edge data distribution, then define the HA-

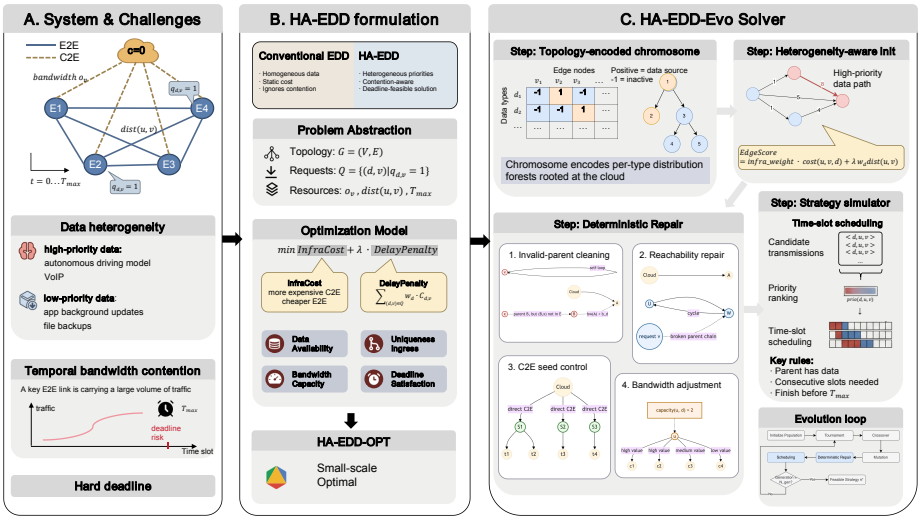

Fig. 1. Overview of HA-EDD, including system challenges, problem modeling, and solution methods.

EDD problem and derive its optimization model, after which we establish its computational hardness and develop exact and heuristic solvers for different problem scales.

3.1 HA-EDD Problem Definition

We formalize HA-EDD by specifying the cloud–edge network, data heterogeneity, request semantics, and transmission cost model.

Definition 1 (Cloud-Edge Graph Model). *The underlying infrastructure is modeled as an undirected graph $G = (V, E)$ [14, 18, 19]. The cloud is denoted by $c = 0 \in V$, and the remaining nodes $v \in V \setminus \{0\}$ represent edge servers, with $|V \setminus \{0\}| = n$. Each edge $(u, v) \in E$ is a high-speed link between servers u and v. If $(u, v) \in E$, we denote by $\mathrm{dist}(u, v) \in \mathbb{N}^+$ the number of time slots required to transmit a data item from u to v along this link (i.e., the propagation delay between the two servers).*

The cloud server can directly send any data type to any edge server through a dedicated cloud-to-edge (C2E) link [18, 19]. We use a unified C2E distance parameter δ to represent the latency between the cloud and each edge server in time slots. Each node $v \in V$ is associated with an outgoing bandwidth capacity o_v. For the cloud node, we assume o_0 is sufficiently large and treat it as unbounded, whereas for each edge server $v > 0$, o_v captures the maximum amount of bandwidth that can be used by v in every time slot.

We adopt a discrete time-slotted model. Time is indexed by t, where $t \in T = \{0, 1, \ldots, T_{\max}\}$. Transmitting a data type d from node u to v over edge (u, v)

occupies b_d units of outgoing bandwidth of node u for $\mathrm{dist}(u, v)$ *consecutive time slots. Similarly, a C2E transmission from the cloud to an edge server occupies b_d units of bandwidth at the cloud for δ consecutive time slots.*

This model captures two dissemination modes: cloud injection through expensive but always available C2E links, and forwarding through cheaper E2E links. The time-slotted view reflects continuous bandwidth occupation during transmission.

Definition 2 (Data Heterogeneity and Importance). *To capture* ***data heterogeneity and importance****, let D denote the set of data types (e.g., application packages, models, or content objects). For each type $d \in D$, let $b_d > 0$ be its bandwidth consumption (i.e., data size), $R_d > 0$ be its average request revenue, and $\beta_d > 0$ be its revenue sensitivity to delay. We define the composite weight $w_d = R_d \cdot \beta_d$, which reflects both the economic importance and latency sensitivity of type d.*

These attributes distinguish HA-EDD from homogeneous EDD formulations by capturing both bandwidth consumption and delay-sensitive business loss.

Definition 3 (User Requests and Hard Deadline). *Distribution schedules are driven by* ***user requests*** *and their temporal constraints. Requests are aggregated at edge servers, represented by a binary parameter $q_{d,v} \in \{0, 1\}$ indicating whether edge server v has at least one request for data type d in the considered time window. The set of request pairs is thus $\mathcal{Q} = \{(d, v) \mid d \in D, v \in V \setminus \{0\}, q_{d,v} = 1\}$. Let $T_{\max}$ denote the maximum tolerable end-to-end latency in time slots. For every request $(d, v) \in \mathcal{Q}$, the corresponding data type d must be delivered to edge server v no later than time slot $T_{\max}$.*

Definition 4 (Multi-Hop Transmission Cost). *Let $\gamma > 1$ denote the cost ratio between C2E and E2E transmissions [19]. For a transmission of data type d along a directed link (u, v), the infrastructure cost* $\mathrm{cost}(u, v, d)$ *is $\gamma \cdot b_d$ if $u = 0$, and b_d otherwise.*

Definition 5 (HA-EDD Problem). *Given a cloud–edge graph $G = (V, E)$, a heterogeneous data-type set D, request pairs $\mathcal{Q}$, node bandwidth capacities $\{o_v\}$, C2E delay δ, cost ratio γ, and deadline window $T_{\max}$, the heterogeneity-aware edge data distribution (HA-EDD) problem seeks a feasible time-slotted distribution schedule π that delivers every requested data type d to every target edge server v with $(d, v) \in \mathcal{Q}$ by the deadline. Feasibility requires that transmissions respect topology reachability, temporal causality, and per-slot bandwidth capacities. Among all feasible schedules, HA-EDD minimizes the total infrastructure cost together with the heterogeneity-weighted delay penalty induced by delivery completion times.*

Together, these definitions specify HA-EDD as a multi-destination dissemination problem with hard bandwidth and deadline constraints.

3.2 Optimization Modeling of HA-EDD Problem

We reformulate HA-EDD as a discrete-time optimization program that jointly determines cloud injection, edge propagation, and transmission timing.

Decision Variables. Let $\mathcal{A} = \{(u,v) \mid (u,v) \in E\} \cup \{(0,v) \mid v \in V \setminus \{0\}\}$ be the set of directed transmission arcs, where $\mathrm{dist}(0,v) = \delta$ for all $v > 0$. For every data type $d \in D$, arc $(u,v) \in \mathcal{A}$, and time slot $t \in T$, we define the binary scheduling variable

$$\mathcal{T}_{d,u,v}^{(t)} = \begin{cases} 1, & \text{if } d \text{ starts transmission from } u \text{ to } v \text{ at slot } t, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

We also use the binary state variable

$$z_{d,v}^{(t)} = \begin{cases} 1, & \text{if node } v \text{ already has } d \text{ at the start of slot } t, \\ 0, & \text{otherwise,} \end{cases} \tag{2}$$

to capture the temporal availability of each data type at each node.

Initial Availability and Temporal Consistency. The cloud initially stores all data types, while edge servers do not. Therefore, $z_{d,0}^{(t)} = 1$ for all $d \in D, t \in T$, and $z_{d,v}^{(0)} = 0$ for all $d \in D, v \in V \setminus \{0\}$. Any transmission must originate from a node that already possesses the corresponding data type:

$$\mathcal{T}_{d,u,v}^{(t)} \le z_{d,u}^{(t)}, \qquad \forall d \in D,\ \forall (u,v) \in \mathcal{A},\ \forall t \in T. \tag{3}$$

The state transition must also respect causality. A node can possess a type at slot t only if it already had it at slot $t-1$ or if some transmission to that node completes exactly at slot t:

$$z_{d,v}^{(t)} \le z_{d,v}^{(t-1)} + \sum_{u:(u,v)\in\mathcal{A}} \mathbb{I}\big(t \ge \mathrm{dist}(u,v)\big) \cdot \mathcal{T}_{d,u,v}^{(t-\mathrm{dist}(u,v))}, \quad \forall d \in D,\ \forall v \in V,\ \forall t \ge 1, \tag{4}$$

with the corresponding lower bounds

$$\begin{aligned} &z_{d,v}^{(t)} \ge z_{d,v}^{(t-1)}, \\ &z_{d,v}^{(t)} \ge \sum_{u:(u,v)\in\mathcal{A}} \mathbb{I}\big(t \ge \mathrm{dist}(u,v)\big) \cdot \mathcal{T}_{d,u,v}^{(t-\mathrm{dist}(u,v))}, \quad \forall d \in D,\ \forall v \in V,\ \forall t \ge 1. \end{aligned} \tag{5}$$

Transmission Exclusivity. To avoid conflicting arrivals, each node can receive a given data type from at most one predecessor in the same slot, and redundant transfers are disallowed once the destination already holds the data:

$$\sum_{u:(u,v)\in\mathcal{A}} \mathcal{T}_{d,u,v}^{(t)} \le 1,\ \forall v > 0; \qquad \mathcal{T}_{d,u,v}^{(t)} + z_{d,v}^{(t)} \le 1,\ \forall (u,v) \in \mathcal{A}, \tag{6}$$

for all $d \in D$ and $t \in T$.

Bandwidth Capacity Constraints. Because a transmission occupies the sender's outgoing bandwidth for multiple consecutive time slots, the effective bandwidth usage at slot t must aggregate all transmissions that are still in progress. For each edge node $u \in V \setminus \{0\}$,

$$\sum_{d \in D} \sum_{v:(u,v) \in E} \sum_{k=0}^{\min\{\mathrm{dist}(u,v)-1,\, t\}} b_d \cdot \mathcal{T}_{d,u,v}^{(t-k)} \leq o_u, \qquad \forall t \in T. \tag{7}$$

This constraint couples routing decisions with transmission timing.

Deadline Satisfaction. Every requested pair $(d, v) \in \mathcal{Q}$ must be completed no later than the deadline window: $z_{d,v}^{(T_{\max})} = 1$ for all $(d, v) \in \mathcal{Q}$. We define the completion time of each satisfied request as $C_{d,v} = \min\{t \in T \mid z_{d,v}^{(t)} = 1\}$ for all $(d, v) \in \mathcal{Q}$, which measures when the target node first obtains the requested type.

Joint Objective. The resulting optimization objective minimizes the total infrastructure cost and heterogeneity-aware delay penalty:

$$\min \sum_{d \in D, t \in T, (u,v) \in \mathcal{A}} \mathrm{cost}(u, v, d) \cdot \mathcal{T}_{d,u,v}^{(t)} + \lambda \sum_{(d,v) \in \mathcal{Q}} w_d \cdot C_{d,v}. \tag{8}$$

This yields a joint routing and scheduling problem.

3.3 $\mathcal{NP}$-Hardness Proving of HA-EDD Problem

Even without heterogeneity-awareness and latency penalties, EDD is already computationally hard [14, 19]. We formalize this by showing that HA-EDD remains $\mathcal{NP}$-hard even in a highly simplified special case.

Theorem 1 ($\mathcal{NP}$-hardness). *HA-EDD is $\mathcal{NP}$-hard, even when there is a single data type ($|D| = 1$), the objective ignores delay penalties ($\lambda = 0$), all E2E link delays are one time slot, and all bandwidth capacities are sufficiently large.*

Proof. We reduce from the unweighted Steiner Tree problem. Given an undirected graph $G' = (V', E')$, a terminal set $R \subseteq V'$, and an integer K, Steiner Tree asks whether there exists a connected subgraph that spans all vertices in R using at most K edges.

Construct a HA-EDD instance as follows. Let the node set be $V = \{0\} \cup V'$, where node 0 is the cloud. Use a single data type d with bandwidth $b_d = 1$ and set $\lambda = 0$. For each undirected edge $\{u, v\} \in E'$, add both directed E2E links (u, v) and (v, u) with unit delay. As defined in HA-EDD, allow C2E links from the cloud to every edge node with fixed delay and cost ratio γ, and set $\gamma = K+1$. Set the deadline window to $T_{\max} = |V'| + 1$ and create requests $q_{d,v} = 1$ for all terminals $v \in R$. Finally, set all outgoing bandwidth capacities large enough so that feasibility is determined only by reachability and time ordering.

If the Steiner Tree instance has a spanning subgraph with at most K edges, pick any node in that subgraph as a seed s, transmit d once from the cloud to s, and then forward along the chosen edges to reach all terminals. This yields a feasible HA-EDD schedule with total infrastructure cost $\gamma + K$. Conversely, if a HA-EDD schedule has objective at most $\gamma+K$ under $\lambda = 0$ and $\gamma = K+1$, it can include at most one C2E transmission and at most K E2E transmissions. The induced subgraph of E2E transmissions must be connected and must span all terminals (otherwise some requested node cannot obtain d), implying a Steiner Tree solution with at most K edges. Thus, Steiner Tree reduces to HA-EDD, and HA-EDD is $\mathcal{NP}$-hard.

3.4 Finding an Optimum Solution: HA-EDD-OPT

To obtain an exact baseline for HA-EDD on small-to-medium instances, we develop HA-EDD-OPT based on CP-SAT. We directly encode the optimization model in Section 3.2 using the binary transmission variables $\mathcal{T}^{(t)}_{d,u,v}$ and state variables $z^{(t)}_{d,v}$, together with the causality, exclusivity, bandwidth, and deadline constraints, so that cloud injection, edge forwarding, and time-slot scheduling are optimized within one unified framework.

The objective jointly minimizes infrastructure cost and the heterogeneity-weighted delay penalty. This allows HA-EDD-OPT to balance expensive but fast C2E delivery against cheaper E2E propagation according to λ, γ, and w_d. In our framework, HA-EDD-OPT is used as the exact method for small-scale instances and as a benchmark for evaluating the near-optimal performance of HA-EDD-Evo on larger networks.

3.5 Heuristic GA for Large-Scale Data: HA-EDD-Evo

For large-scale instances, we design a topology-encoded heterogeneity-aware genetic algorithm, called HA-EDD-Evo. It uses a graph-encoded chromosome and a repair-and-schedule pipeline to enforce bandwidth and deadline feasibility, as illustrated in Figure 1(C) and summarized in Algorithm 1.

Chromosome Encoding and Initialization. For each data type d and edge server $v > 0$, the chromosome maintains a parent pointer $g_{d,v}$, where $g_{d,v} = -1$ if v is inactive for type d, $g_{d,v} = 0$ if v receives d directly from the cloud, and $g_{d,v} = u$ if v receives d from neighbor u. Thus, a solution is a two-dimensional parent array `parents` encoding one cloud-rooted distribution forest per data type. For each type d, initialization builds cloud-to-target paths with a weighted shortest-path algorithm whose edge weight combines infrastructure cost and delay penalty:

$$\begin{aligned}\text{EdgeScore}(u, v, d) = \text{infra\_weight} \cdot b_d \cdot (\gamma \text{ if } u = 0 \text{ else } 1) \\ + \lambda \cdot w_d \cdot \text{dist}(u, v).\end{aligned} \tag{9}$$

We also turn a small fraction of targets into direct cloud seeds to increase population diversity.

Algorithm 1 Topology-Encoded Genetic Algorithm for HA-EDD (HA-EDD-Evo)

Require: Network graph $G = (V, E)$, data types D, request set $\mathcal{Q}$
Deadline window $T_{\max}$, GA parameters $(N_{\text{pop}}, N_{\text{gen}}, p_{\text{cr}}, p_{\text{mu}})$
Ensure: Feasible distribution strategy $\pi^\star$

Initialize a population $\Gamma = \{G_1, \ldots, G_{N_{\text{pop}}}\}$ using shortest paths based on EdgeScore;
for each chromosome $G \in \Gamma$ **do**
  $\textsc{Repair}(G)$
  $\pi_G \leftarrow \textsc{Schedule}(G)$
  $f(G) \leftarrow \textsc{Fitness}(\pi_G)$
end for
for $g = 1$ to N_{gen} **do**
  $\Gamma' \leftarrow \emptyset$
  while $|\Gamma'| < N_{\text{pop}}$ **do**
    Select two parents $G^{(1)}, G^{(2)}$ from Γ by tournament selection;
    $(H^{(1)}, H^{(2)}) \leftarrow \textsc{Crossover}(G^{(1)}, G^{(2)}, p_{\text{cr}})$
    $\textsc{Mutate}(H^{(1)}, p_{\text{mu}})$; $\textsc{Mutate}(H^{(2)}, p_{\text{mu}})$
    for each offspring $H \in \{H^{(1)}, H^{(2)}\}$ **do**
      $\textsc{Repair}(H)$; $\pi_H \leftarrow \textsc{Schedule}(H)$; $f(H) \leftarrow \textsc{Fitness}(\pi_H)$
      $\Gamma' \leftarrow \Gamma' \cup \{H\}$
    end for
  end while
  $\Gamma \leftarrow \textsc{SelectNextGen}(\Gamma \cup \Gamma', N_{\text{pop}})$
end for
$G^\star \leftarrow \arg\min_{G \in \Gamma} f(G)$
return $\pi^\star = \pi_{G^\star}$

Deterministic Solution Repair The raw chromosome may contain infeasibilities such as disconnected request nodes or overloaded parents. We apply a four-step deterministic repair before fitness evaluation. *Step 1: Invalid cleaning.* We remove edges with self-loops, missing physical links, or insufficient parent bandwidth. *Step 2: Reachability repair.* For each unfulfilled request, we reconstruct a path from the cloud using weighted shortest paths with EdgeScore as link weights. *Step 3: C2E seed control.* We retain only high-score direct cloud children and reattach others to reachable E2E parents. *Step 4: Bandwidth adjustment.* We cap each node's children by bandwidth, prioritizing larger and more important (w_d) subtrees and migrating excess children to the cloud.

Distribution Strategy Simulator Given a repaired chromosome, a discrete-time simulator evaluates the distribution strategy. We compute a transmission priority $\text{prio}(d, u, v) \propto (\text{subtree\_size}(d, v) \cdot w_d)/\,\text{cost}(u, v, d)$, favoring larger and more valuable subtrees. The simulator greedily places transmissions into the earliest contiguous time window with sufficient residual bandwidth while respecting availability and deadlines. Unserved requests fall back to direct C2E

Table 1. Parameter configurations for the three experiment scales.

Parameters	Small	Medium	Large
Edge servers n	25	50	60
Coverage	0.30	0.35	0.45
Requests per node	8	10	12
Data types $\|D\|$	15	15	20
$T_{\max}$	15	30	30

delivery when feasible. The final score combines infrastructure cost, weighted delay penalty, and static penalties for structural violations.

Genetic Operators HA-EDD-Evo uses tournament selection to choose parent chromosomes. For crossover, it processes each data type independently and performs one-point crossover on `parents[d][:]` with a given probability. Mutation selects a random (d, v) and perturbs $g_{d,v}$ to a neighbor, the cloud, or a non-participating state, with neighbors preferred to preserve locality. All offspring are passed through the repair-and-scheduling pipeline before fitness evaluation.

4 Experiments

4.1 Datasets and Experimental Setup

We evaluate our proposed methods using two real-world datasets:

- **China ISP Edge Computing Dataset [7]:** We build request workloads by aggregating records to edge-server locations and extracting destination domains from the collected mobile traffic logs.
- **Alibaba cluster trace [2]:** We model bandwidth demands by sampling empirical distributions from this trace and assigning each data type a demand after normalization.

All experiments were conducted on a workstation equipped with two NVIDIA GeForce 1080Ti GPUs and an Intel Xeon Gold 6132 CPU running at 2.60 GHz. We use three representative scales for the core comparison, summarized in Table 1. For HA-EDD-OPT, we cap the CP-SAT wall time to 180 seconds per instance. Each configuration is repeated with multiple random seeds: five seeds for the Small/Medium/Large comparison and three seeds for the sweep and ablation studies.

4.2 Competing Approaches and Evaluation Metrics

We compare our proposed methods (HA-EDD-OPT and HA-EDD-Evo) against the following baselines:

- Random: Randomly selects seed edge servers per data type, routes remaining targets via shortest paths, and falls back to direct cloud delivery when needed.
- Terminal-Bumper [17]: An online caching algorithm that places caches on a cloud-rooted tree once the accumulated delivery cost crosses a threshold.
- EDD-IP [19]: An integer-programming solver that minimizes EDD transmission cost under hop/latency constraints.
- EDD-A [19]: A heuristic that builds a connectivity-oriented Steiner tree and repairs it to satisfy hop limits.
- EDD-NSTE [14]: A network Steiner tree estimation approach with slicing and fine-tuning to meet latency constraints.
- LAO [18]: A latency-aware online method that selects cost-effective paths by balancing latency slack and cost.
- RVA [16]: A PageRank-inspired regional-importance method that seeds replicas based on regional importance and shortest-path reachability.

To comprehensively evaluate performance, we report the overall objective, infrastructure cost (InfraCost), and delay penalty (DelayPenalty), together with wall-clock solving time per instance. In the result tables, we use the abbreviations Obj, IC, DP, and CPU Time to denote objective, InfraCost, DelayPenalty, and runtime, respectively. For the ablation study, we additionally report two completion-time metrics. Let $\mathcal{Q}_{\text{done}} \subseteq \mathcal{Q}$ denote the set of completed requests in the simulator. The average completion time is

$$\text{AvgCompletion} = \frac{\sum_{(d,v)\in\mathcal{Q}_{\text{done}}} C_{d,v}}{|\mathcal{Q}_{\text{done}}|}. \tag{10}$$

Let the total request weight be

$$W = \sum_{(d,v)\in\mathcal{Q}} w_d. \tag{11}$$

The weighted average completion time is computed as

$$\text{WAvgCompletion} = \frac{\sum_{(d,v)\in\mathcal{Q}_{\text{done}}} w_d C_{d,v}}{W} = \frac{\text{DelayPenalty}}{W}, \tag{12}$$

All reported metrics are aggregated as mean values across multiple random seeds.

4.3 Experimental Results and Analyses

We systematically benchmark our proposed methods through three sets of experiments:

- A performance comparison across three scales: Small, Medium, and Large.
- An ablation study comparing heterogeneity-aware weights against uniform weights.
- A parameter-impact analysis over λ and γ. We test six λ values from 0 to 5 and four γ values: 1, 2, 5, and 10.

Table 2. Performance comparison on the main benchmark instances. Bold and underlined values indicate best and second-best results, respectively.

Methods	Small				Medium				Large			
	Obj	IC	DP	CPU Time	Obj	IC	DP	CPU Time	Obj	IC	DP	CPU Time
Random	22300	13821	8478	<u>0.003</u>	57716	34184	23531	<u>0.013</u>	99107	55773	43334	<u>0.025</u>
Terminal-Bumper	17988	11478	**6510**	**0.001**	48512	31410	**17102**	**0.003**	82347	50640	**31707**	**0.005**
EDD-IP	17254	10497	6757	1.031	45895	20211	25684	5.210	83049	44859	38189	25.505
EDD-A	17959	11425	<u>6534</u>	0.016	48738	27349	21389	0.056	82315	47014	<u>35302</u>	0.095
EDD-NSTE	18288	11777	**6510**	0.061	49284	32182	**17102**	0.189	83466	51759	**31707**	0.362
LAO	15465	<u>7415</u>	8050	0.016	46795	**8226**	38568	0.200	82996	**11888**	71108	0.596
RVA	24362	15088	9274	0.007	48487	19117	29370	0.017	86969	28502	58467	0.035
HA-EDD-Evo	<u>15130</u>	7657	7473	0.080	<u>37853</u>	17337	<u>20516</u>	0.122	<u>66684</u>	28000	38684	0.156
HA-EDD-OPT	**13330**	**5543**	7787	5.355	**33116**	<u>10166</u>	22950	180.896	**59736**	<u>17882</u>	41854	181.701

Performance Comparison Across Small, Medium, and Large Instances We evaluate all methods on Small, Medium, and Large instances using five random seeds per configuration, and report objective, InfraCost, DelayPenalty, and runtime.

Table 2 shows that HA-EDD-OPT consistently achieves the lowest objective across all three scales. Compared with the strongest baseline in each scale (LAO for Small, EDD-IP for Medium, and EDD-A for Large), HA-EDD-OPT reduces the overall objective by 13.8%, 27.8%, and 27.4%, respectively. Following closely, HA-EDD-Evo consistently outperforms all heuristic baselines, improving the objective by 2.2%, 17.5%, and 19.0% across the three scales. The cost and penalty breakdown reflects a fundamental trade-off: while proactive methods like Terminal-Bumper and routing heuristics like EDD-NSTE occasionally achieve a lower delay penalty, they incur noticeably higher infrastructure costs, thereby inflating the overall objective. In terms of runtime, HA-EDD-OPT approaches the 180-second time cap on Medium/Large instances, whereas HA-EDD-Evo stays within sub-second time budgets and is two to three orders of magnitude faster while achieving near-optimal objectives.

Heterogeneity-Aware Ablation Against Uniform Weights To isolate the benefit of incorporating data business importance and delay sensitivity into the optimization, we ablate our framework by comparing heterogeneity-aware weights against uniform weights under identical network configurations.

The ablation results in Figure 2 validate our weight modeling. Replacing uniform weights with heterogeneity-aware ones raises InfraCost for HA-EDD-OPT and HA-EDD-Evo by 69.6% and 14.7%, but drops their weighted average completion times by 34.0% and 17.6%, respectively. This confirms that incorporating business value and delay sensitivity steers the scheduler toward critical requests. Furthermore, unweighted average completion times also improve by 18.8% and 9.7% without incurring the excessive infrastructure costs seen in EDD-NSTE or Terminal-Bumper. Ultimately, HA-EDD reacts to request heterogeneity in the intended direction while maintaining a vastly superior cost–quality trade-off.

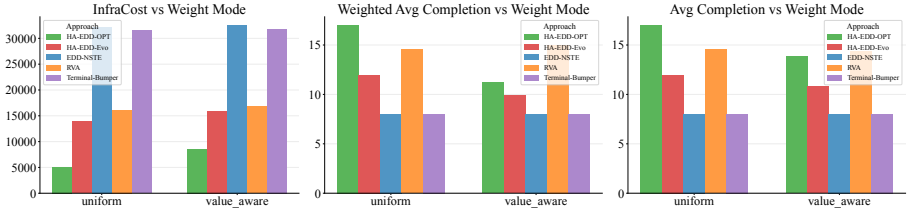

Fig. 2. Heterogeneity-aware ablation results (InfraCost, weighted average completion time, and average completion time).

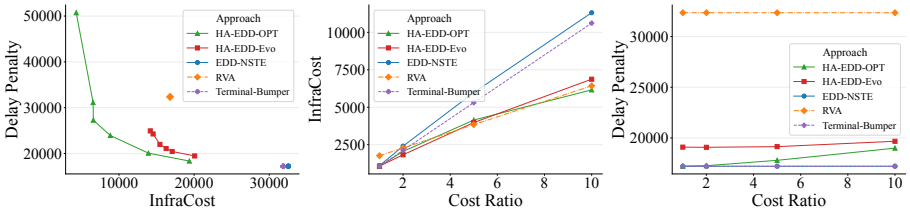

Fig. 3. Performance impact of key parameters (left: Pareto frontier under varying λ; middle/right: sensitivity to the C2E cost ratio γ in terms of InfraCost and DelayPenalty).

Performance Impact of Parameters We further examine how key objective and pricing parameters affect optimization behavior, focusing on both the trade-off coefficient λ and the C2E cost ratio γ.

As visualized in Figure 3, increasing λ causes both HA-EDD-OPT and HA-EDD-Evo to move along a clear cost–delay frontier: InfraCost rises as the system utilizes more resource-intensive links to expedite delivery, which in turn steadily reduces DelayPenalty. This indicates that both our exact and heuristic optimizers effectively respond to the parameterized objective. In contrast, the baseline curves for EDD-NSTE, RVA, and Terminal-Bumper are much flatter across λ, which is expected because they do not directly optimize the heterogeneity-weighted cost–delay objective. At larger λ, HA-EDD-OPT/Evo approach the low-delay region while still preserving a more favorable cost profile than these baselines, illustrating controllable trade-offs rather than strict Pareto dominance.

Impact of γ on Solution Performance. To study the impact of cloud-to-edge pricing on distribution decisions, we sweep the C2E cost ratio γ and analyze how the optimizers shift the load towards multi-hop E2E transmissions when direct C2E links become expensive. As summarized in Figure 3, higher C2E cost ratios intuitively push the InfraCost upward for all approaches, while the DelayPenalty experiences more moderate fluctuations. HA-EDD-OPT consistently attains the lowest objective across all tested configurations ($\gamma \in \{1, 2, 5, 10\}$), with HA-EDD-Evo pacing strictly behind it. This demonstrates that our heterogeneity-aware objective formulation remains robust and highly adaptive under diverse cloud pricing scenarios.

5 Conclusion and Future Work

To address the challenges of data heterogeneity and temporal resource contention, we studied heterogeneity-aware edge data distribution (HA-EDD) in cloud–edge systems, where the cloud disseminates heterogeneous data to edge servers under real-world bandwidth and strict deadline constraints. We formulated HA-EDD as an integer optimization problem and developed an exact CP-SAT-based solver (HA-EDD-OPT) and a scalable topology-encoded genetic algorithm (HA-EDD-Evo). Experiments show that HA-EDD-Evo achieves objective values close to the exact solver while scaling to larger networks, and that heterogeneity-aware weights steer the optimizers toward prioritizing high-priority deliveries. We also evaluate sensitivity to cloud egress cost ratios and find that the heterogeneity-aware objective remains robust across pricing regimes.

In future work, we will extend HA-EDD to online settings with time-varying bandwidth and demand, and study robustness under failures and partial/coded delivery.

Acknowledgments. This work was supported by National Natural Science Foundation of China (No. 62272290, 62572114).

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Aliyan Nejadi, R., Motameni, H., Barzegar, B., Akbari, E., Abbaszadeh Sori, A.: Energy-aware scientific real-time DAG scheduling using slack reclamation in heterogeneous DVFS cloud datacenters. IEEE Access **13**, 169825–169840 (2025)
2. Cheng, Y., Chai, Z., Anwar, A.: Characterizing Co-located datacenter workloads: An Alibaba case study. arXiv preprint arXiv:1808.02919 (2018)
3. Cong, P., Xu, G., Zhou, J., Chen, M., Wei, T., Qiu, M.: Personality-and value-aware scheduling of user requests in cloud for profit maximization. IEEE Transactions on Cloud Computing **10**(3), 1991–2004 (2020)
4. He, Q., Zhang, G., Wang, J., Luo, R., Dai, X., Hu, Y., Chen, F., Jin, H., Yang, Y.: EdgeHydra: Fault-tolerant edge data distribution based on erasure coding. IEEE Transactions on Parallel and Distributed Systems **36**(1), 29–42 (2024)
5. Jin, H., Luo, R., He, Q., Wu, S., Zeng, Z., Xia, X.: Cost-effective data placement in edge storage systems with erasure code. IEEE Transactions on Services Computing **16**(2), 1039–1050 (2022)
6. Li, B., He, Q., Chen, F., Lyu, L., Bouguettaya, A., Yang, Y.: EdgeDis: Enabling fast, economical, and reliable data dissemination for mobile edge computing. IEEE Transactions on Services Computing **17**(4), 1504–1518 (2023)
7. Liu, H., Li, Y., Wang, S.: Request scheduling combined with load balancing in mobile-edge computing. IEEE Internet of Things Journal **9**(21), 20841–20852 (2022)
8. Luo, R., Liang, Z., Nie, A., He, Q., Chen, F., Xiao, W., Yang, J., Gao, Y., Yang, Y.: Online caching replacement in erasure coding-Based edge storage system. IEEE Transactions on Services Computing **18**(6), 4069–4081 (2025)

9. Ma, Y., Zhang, T., Feng, X., Liu, P., Yang, H., Ren, F.: Improving robustness of Time-Aware Shaper in Time-Sensitive Networking. IEEE Internet of Things Journal **12**(13), 25211–25223 (2025)
10. Nguyen, N.H., Nguyen, V.D., Nguyen, A.T., Van Thieu, N., Nguyen, H.N., Chatzinotas, S.: Deadline-aware joint task scheduling and offloading in mobile-edge computing systems. IEEE Internet of Things Journal **11**(20), 33282–33295 (2024)
11. Niu, S., Liu, Y., Liao, K., Zhang, B., Zou, G.: Dynamic edge-caching through content popularity and crowd prediction for short video services. Scientific Reports **15**(1), 42147 (2025)
12. Pham, Q.V., Fang, F., Ha, V.N., Piran, M.J., Le, M., Le, L.B., Hwang, W.J., Ding, Z.: A survey of multi-access edge computing in 5G and beyond: Fundamentals, technology integration, and state-of-the-art. IEEE Access **8**, 116974–117017 (2020)
13. Shankar, R., Sahu, A.: Cost-effective edge data distribution with end-to-end delay guarantees in edge computing. arXiv preprint arXiv:2504.21070 (2025)
14. Swain, C.K., Shankar, R., Sahu, A.: Edge data distribution as a network steiner tree estimation in edge computing. Computing **106**(5), 1585–1609 (2024)
15. Taleb, T., Samdanis, K., Mada, B., Flinck, H., Dutta, S., Sabella, D.: On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration. IEEE Communications Surveys & Tutorials **19**(3), 1657–1681 (2017)
16. Wang, P., Qiao, J., Zhao, Y., Ding, Z.: Cost-effective and low-latency data placement in edge environment based on PageRank-inspired regional value. IEEE Transactions on Parallel and Distributed Systems **36**(2), 185–196 (2024)
17. Wu, B., Bao, W., Zhou, B.B.: Competitive analysis of online elastic caching of transient data in multi-tiered content delivery network. IEEE Transactions on Parallel and Distributed Systems **35**(12), 2449–2462 (2024)
18. Xia, X., Chen, F., He, Q., Grundy, J., Abdelrazek, M., Shen, J., Bouguettaya, A., Jin, H.: Formulating cost-effective data distribution strategies online for edge cache systems. IEEE Transactions on Parallel and Distributed Systems **33**(12), 4270–4281 (2022)
19. Xia, X., Chen, F., He, Q., Grundy, J.C., Abdelrazek, M., Jin, H.: Cost-effective app data distribution in edge computing. IEEE Transactions on Parallel and Distributed Systems **32**(1), 31–44 (2020)