PDOP: Prediction-Driven Online Pricing for Live Edge Video Streaming

Anonymous ICME submission

***Abstract*—Live video streaming at the network edge requires coordinated optimization of limited bandwidth and computing resources to ensure low-latency delivery. We consider an online edge computing scenario where video streams must be multicast and transcoded in real time across resource-constrained routers. We formulate this as a social welfare maximization problem and show it is NP-hard. To tackle the challenge, we propose an auction-based framework that dynamically decides which video requests to serve, how to route and replicate their streams, and where to perform in-network transcoding. We further introduce a learning-augmented solution called PDOP, which integrates predicted future demand to enhance decision-making while preserving these guarantees. Extensive simulations demonstrate that our approach significantly outperforms baseline schemes in total welfare, resource utilization, and latency, and that PDOP achieves additional gains by anticipating demand. PDOP leverages dynamic pricing of bandwidth and CPU resources and constructs near-optimal multicast delivery trees to efficiently allocate resources. The framework provides strong theoretical guarantees: it is truthful, individually rational, budget-balanced, and achieves performance within a logarithmic factor of the optimal offline solution.**

***Index Terms*—Auction, Video Stream, Edge Computing, Online Pricing, Predictive-augmented**

I. Introduction

The proliferation of live video streaming presents significant challenges for modern networks. For example, consider a popular global sports event streamed live to millions of viewers across different regions and devices. Under a traditional architecture, each viewer's request might be served by a distant cloud data center, which must process the video and then deliver it over long-haul networks. This centralized approach can lead to high latency, congestion, and inefficient bandwidth usage, particularly when identical content is sent repeatedly over the core network.

Edge Computing Networks (ECNs) have emerged as a promising solution to these challenges. By embedding computational resources at network routers, ECNs enable in-network processing such as video transcoding and replication close to end users. In this paradigm, an edge router can transcode a high-bitrate live stream into multiple lower-bitrate versions and replicate it to local viewers, significantly reducing end-to-end latency and alleviating core-network traffic. However, the benefits of ECNs come with new resource-management challenges. Each edge node has limited processing power and transmission capacity, so the system must carefully coordinate these resources to serve multiple video requests concurrently.

Conventional content delivery systems do not fully exploit these edge capabilities. Typical cloud-centric or broadcast frameworks rely on fixed or coarse-grained scheduling policies and do not jointly optimize computing and communication resources. As a result, they may overload the core network with duplicate streams or underutilize expensive edge-computing resources, failing to meet stringent low-latency requirements. In particular, such approaches struggle to serve large-scale live events with dynamic demand patterns, as they cannot adaptively allocate edge processing and bandwidth in response to changing demand.

We consider the problem of online live video streaming in ECNs: requests for different video qualities arrive sequentially over time, and the system must immediately decide which streams to admit, how to route them, and where to transcode them at the edge. The objective is to maximize social welfare, defined as the total value of all served video streams minus the cost of computing and network resources consumed. Even in an offline setting with full knowledge of all requests, this joint routing–transcoding optimization can be formulated as a mixed-integer nonlinear program (MINLP), which we prove is NP-hard. The key difficulty is the interdependence of decisions: choosing to serve a particular stream requires deciding its multicast routing tree and where to perform any necessary transcoding, subject to link and node capacity constraints. These coupled decisions make the problem combinatorial and hard to decompose. In the online setting, uncertainty about future requests further complicates the problem: a naive admission decision could preclude serving a later high-value request.

To tackle these challenges, we propose an auction-based dynamic optimization framework. Our approach begins with an online mechanism that treats each arriving video request as a bid for service. DOP integrates primal–dual admission control, dynamic resource pricing, and efficient construction of Transcoding Multicast Trees (TMTs) for admitted streams. By comparing each bid against the current marginal cost of serving it, DOP ensures that resources are allocated to the most valuable streams under current network conditions. We show that DOP runs in polynomial time and achieves provable approximation guarantees for routing and scheduling.

Building on DOP, we introduce PDOP, a learning-augmented extension that incorporates demand predictions into the decision-making process. PDOP adjusts the pricing and admission rules based on short-term forecasts of future video demand. Crucially, PDOP preserves all of DOP's incentive and feasibility guarantees, while providing two key benefits: if predictions are accurate, PDOP approaches the performance of the offline optimum consistency; if predictions are unreliable, PDOP degrades gracefully to DOP's worst-case guarantees robustness. Experimental results show that PDOP significantly outperforms DOP and other non-predictive baselines under realistic streaming scenarios with forecasted demand.

Our main contributions are:

- **Problem formulation and reformulation:** We formulate the joint routing–scheduling–transcoding problem for online ECN video streaming as an MINLP that maximizes social welfare. Even the offline version of this problem is NP-hard. Using the optimization technique, we eliminate nonlinear coupling constraints to derive an equivalent pure 0–1 integer linear program (ILP), enabling an efficient online primal–dual solution approach.
- **Predictive extension (PDOP):** We design DOP, an online mechanism in which each arriving video request is treated as an auction bid. We develop PDOP by integrating demand forecasts into DOP's pricing and admission rules. PDOP retains all of DOP's desirable properties. Moreover, PDOP is consistent and robust. Our simulations demonstrate that PDOP significantly improves long-term social welfare compared to DOP and other baselines when prediction information is available.
- **Evaluation:** We conduct extensive simulations on representative network topologies and live-streaming scenarios. Results show that DOP significantly outperforms non-adaptive baselines in terms of social welfare and resource utilization, and that PDOP yields additional gains when predictive information is leveraged.

II. Related Work

A. AI-driven Video Streaming and Edge Integration

Recent surveys highlight the convergence of AI, cloud, and edge in video streaming. Darwich *et al.* review how artificial intelligence, cloud computing and edge technologies jointly transform video streaming systems [1]. Wang *et al.* emphasize the "added value of intelligent algorithms" for adaptive streaming, identifying emerging directions such as machine-learning-driven resource allocation and hybrid cloud–edge architectures [2]. These works underscore the importance of intelligent, adaptive management in streaming platforms. In contrast, the PDOP framework builds on these insights by focusing specifically on predictive offloading and resource optimization for video streams, rather than general surveys of technology trends.

B. Cloud–Edge Collaborative Analytics

A number of recent works propose cloud–edge hybrid architectures for video analytics tasks. For example, Chen *et al.* introduce a cloud-edge collaborative framework for traffic video analytics [3]. Qian *et al.* introduce a distributed streaming video analytics system that features optimized processing pipelines and multi-agent RL-based controllers for fast adapting the system configurations across the edge and the cloud [4]. Ye *et al.* present a novel distributed computation intelligent system with nearby edge devices, abbreviated as EdgeStreaming, to facilitate rapid and accurate analysis of streaming data [5]. Yuan *et al.* develop EOCA for cloud–edge video streaming (CEAVS), jointly optimizing accuracy, energy and latency through online offloading and redundancy compression under budget constraints [6]. These works focus on optimally partitioning video analytics tasks between edge and cloud. PDOP complements this line of work by specifically addressing predictive task offloading in streaming scenarios, leveraging future workload estimates to improve decision-making beyond the reactive or static schemes above.

C. Adaptive Streaming and Quality Optimization

Other works address quality-of-experience (QoE) optimization for interactive or high-resolution video. Hu *et al.* develop a dynamic scheduling and resource allocation algorithm to address the inherent randomness in data arrivals and resource availability under long-term energy constraints [7]. Liu *et al.* jointly optimize the four stages of prediction, caching, computing and transmission in mobile edge caching system, aimed to maximize the user's quality of experience [8]. Yau *et al.* optimize video quality and fairness using a novel throughput estimator, Weighted Harmonic Exponential Averaging, which improves robustness and accuracy under dynamic conditions [9].These works target smooth, fair, and high-quality playback. PDOP is related in that it also seeks to improve streaming performance, but it diverges by focusing on predictive offloading scheduling of analytics tasks rather than view synthesis or bitrate selection for playback.

D. Edge System Architectures and Partitioning

Several studies propose system-level techniques for edge streaming. Ding *et al.* propose a novel algorithm, called ClusPar, to address the problem of streaming edge partitioning [10]. Lu *et al.* present STREAMSYS, a light weight executable delivery system that loads code on demand by redirecting the local disk IO to the server through optimized network IO [11]. Gokarn *et al.* claim that high-throughput vision perception is even more challenging in multi-tenancy systems, where video streams from multiple such high-quality cameras need to share the same GPU resource on a single edge device [12]. Hu *et al.* propose VARFVV, a bandwidth- and computationally efficient system that enables real-time interactive FVV streaming with high QoE and low switching delay [13]. These system- and resource-focused approaches demonstrate the need for adaptive management of edge resources. PDOP differs from them by integrating predictive modeling into the offloading decision process, thereby anticipating future load and making proactive scheduling choices rather than reactive partitioning or code delivery.

III. Problem Formulation

We consider an edge computing network with node set V and directed link set E, serving a sequence of video multicast requests indexed by $i \in I$. Time is slotted (indexed by $t \in T$). Each request i has a baseline benefit b_i and must deliver a video stream to multiple destination users (possibly at differing quality levels). For each request i, let P_i be the set of feasible *transmission plans*, where each plan $j \in P_i$ specifies a particular routing and transcoding strategy that satisfies the quality requirements of all destinations. We introduce a binary decision variable $z_{ij} \in 0, 1$ to indicate whether plan j is selected for request i. Each plan j consumes $r_{ij}^{(e,t)}$ units of bandwidth on edge e and $c_{ij}^{(v,t)}$ units of CPU at node v during time slot t. We define the net utility of plan j for request i as $u_{ij} = b_i - cost_{ij}$, where $cost_{ij}$ represents the plan's cost. This plan-selection ILP follows the standard pattern of scheduling/flow ILPs with binary decision variables and capacity constraints.

The ILP formulation then aims to maximize the total utility of the chosen plans. Formally, we write:

$$\max \sum_{i \in I} \sum_{j \in P_i} u_{ij} z_{ij} \tag{1}$$

s.t.

$$\sum_{i \in I} \sum_{j \in P_i} r_{ij}^{(e,t)} z_{ij} \le B_e^{(t)}, \quad \forall e \in E, t \in T, \tag{2}$$

$$\sum_{i \in I} \sum_{j \in P_i} c_{ij}^{(v,t)} z_{ij} \le C_v^{(t)}, \quad \forall v \in V, t \in T, \tag{3}$$

$$\sum_{j \in P_i} z_{ij} \le 1, \quad \forall i \in I, \tag{4}$$

$$z_{ij} \in 0, 1, \quad \forall i \in I, j \in P_i. \tag{5}$$

The first two sets of constraints ensure that, in each time slot t, the aggregate bandwidth used on every edge and the CPU load on every node do not exceed the available capacities $B_e^{(t)}$ and $C_v^{(t)}$. The third constraint enforces that at most one plan is selected per request (a request may remain unserved if no plan is chosen). The last line declares that all decision variables z_{ij} are binary. This completes the 0–1 ILP for PDOP's plan-selection optimization.

A. Plan-Menu Linearization and LP Relaxation

Modeling per-packet multicast forwarding and in-network quality adaptation at the granularity of links, nodes, and time slots typically leads to coupled binary variables and nonconvexities. To keep the optimization aligned with the implementation of PDOP, we adopt a *plan-menu* view: for each request i, we predefine (or generate on the fly) a set of feasible delivery blueprints P_i. Each blueprint $j \in P_i$ fully specifies all low-level actions needed to satisfy i (routing, replication, and quality adaptation), and is summarized by its *resource footprint* over time:

$$\left\{ r_{ij}^{(e,t)} \right\}_{(e,t) \in E \times T}, \qquad \left\{ c_{ij}^{(v,t)} \right\}_{(v,t) \in V \times T}.$$

Thus, the offline admission-and-selection problem becomes a 0–1 packing ILP over the plan variables z_{ij}.

To connect to a primal–dual online method, we relax the integrality and consider the LP relaxation with $z_{ij} \in [0, 1]$. Let $\lambda_{e,t} \ge 0$ and $\pi_{v,t} \ge 0$ be the dual variables associated with the bandwidth and CPU constraints at each resource–time pair, and let $\nu_i \ge 0$ be the dual variable associated with the "at most one plan per request" constraint for request i. The dual program of the LP relaxation can be written as

$$\min \sum_{i \in I} \nu_i \; + \sum_{e \in E} \sum_{t \in T} B_e^{(t)} \lambda_{e,t} \; + \sum_{v \in V} \sum_{t \in T} C_v^{(t)} \pi_{v,t} \tag{6}$$

s.t.

$$\nu_i \;\ge\; b_i - \sum_{e \in E} \sum_{t \in T} r_{ij}^{(e,t)} \lambda_{e,t} - \sum_{v \in V} \sum_{t \in T} c_{ij}^{(v,t)} \pi_{v,t}, \forall i \in I, \; \forall j \in P_i, \tag{7}$$

$$\lambda_{e,t} \ge 0, \; \pi_{v,t} \ge 0, \; \nu_i \ge 0, \quad \forall e \in E, \; \forall v \in V, \; \forall t \in T, \; \forall i \in I. \tag{8}$$

Define the *priced cost* of plan j for request i under prices (λ, π) as

$$\mathrm{pcost}_{ij}(\lambda, \pi) \;=\; \sum_{e,t} r_{ij}^{(e,t)} \lambda_{e,t} \; + \; \sum_{v,t} c_{ij}^{(v,t)} \pi_{v,t}. \tag{9}$$

Then the corresponding *surplus* (net gain) is $s_{ij}(\lambda, \pi) \;=\; b_i - \mathrm{pcost}_{ij}(\lambda, \pi)$.

Constraint (7) ensures that ν_i upper-bounds the best achievable surplus of request i under the current prices. By complementary slackness, at a dual optimum, $\nu_i = \max\left\{0, \; \max_{j \in P_i} s_{ij}(\lambda, \pi)\right\}$, i.e., $\nu_i > 0$ precisely when request i admits at least one plan whose value dominates its priced resource cost.

B. Online Primal–Dual Admission via Best-Plan Surplus

The dual variables (λ, π) admit a natural economic interpretation as *time-dependent unit prices* for bandwidth and CPU. PDOP uses these prices online as follows. When a request i arrives, the algorithm evaluates the priced cost (9) for all candidate plans $j \in P_i$ and selects the best plan $j^\star(i) \in \arg\min_{j \in P_i} \; \mathrm{pcost}_{ij}(\lambda, \pi)$.

Equivalently, this maximizes the surplus (III-A). The admission rule is: accept i and set $z_{ij^\star(i)} = 1 \iff s_{ij^\star(i)}(\lambda, \pi) \ge 0$, and reject otherwise (set $z_{ij} = 0$ for all $j \in P_i$). Once accepted, the algorithm updates the *accumulated usage* on every resource–time pair: $\delta_{e,t} \leftarrow \delta_{e,t} + r_{ij^\star(i)}^{(e,t)}, \qquad \gamma_{v,t} \leftarrow \gamma_{v,t} + c_{ij^\star(i)}^{(v,t)}$.

These usage variables summarize the current consumption state and drive the next price update.

C. Exponential Price Update for Scarcity Awareness

To discourage over-commitment as resources approach saturation, PDOP employs an exponential pricing map from usage to dual prices. Let $B_e^{(t)}$ be the bandwidth capacity of edge e at time t, and $C_v^{(t)}$ be the CPU capacity of node v at time t. A convenient monotone price family is $\lambda_{e,t}(\delta_{e,t}) = (\kappa_e)^{\delta_{e,t}/B_e^{(t)}} - 1, \qquad \pi_{v,t}(\gamma_{v,t}) = (\kappa_v)^{\gamma_{v,t}/C_v^{(t)}} - 1,$

where $\kappa_e > 1$ and $\kappa_v > 1$ are resource-specific scale factors. This design yields $\lambda_{e,t}(0) = 0$ and $\pi_{v,t}(0) = 0$, while prices rise slowly under light load and accelerate as $\delta_{e,t} \to B_e^{(t)}$ or $\gamma_{v,t} \to C_v^{(t)}$. Consequently, the surplus test (III-B) becomes progressively stricter when the network is congested, naturally preserving feasibility.

Given acceptance decisions, the online loop repeats: update usage by (III-B), recompute prices by (III-C), then process the next arriving request.

D. Predictive Augmentation for PDOP

The distinguishing feature of PDOP is that it augments the price update using predictions of future demand. Suppose a predictor provides *anticipated residual load* on each resource–time pair, denoted by $\phi_{e,t} \ge 0$ for bandwidth and $\psi_{v,t} \ge 0$ for CPU. PDOP biases the prices as if a portion of the capacity were already reserved for the predicted future traffic:

$$\tilde{\lambda}_{e,t}(\delta_{e,t}) = (\kappa_e)^{(\delta_{e,t}+\phi_{e,t})/B_e^{(t)}} - 1, \tilde{\pi}_{v,t}(\gamma_{v,t}) = (\kappa_v)^{(\gamma_{v,t}+\psi_{v,t})/C_v^{(t)}} - 1. \quad (10)$$

More generally, one may introduce a confidence parameter $\theta \in [0,1]$ and use $\delta_{e,t} + \theta\phi_{e,t}$ and $\gamma_{v,t} + \theta\psi_{v,t}$ to interpolate between purely reactive pricing ($\theta = 0$) and fully prediction-driven biasing ($\theta = 1$).

Under predictive prices $(\tilde{\lambda}, \tilde{\pi})$, the best-plan computation (III-B) and the surplus test (III-B) are performed using $\mathrm{pcost}_{ij}(\tilde{\lambda}, \tilde{\pi})$. Intuitively, if a link or node is predicted to be heavily demanded in the near future, its current price is raised preemptively, making the algorithm more selective about admitting marginal requests that would consume that scarce resource. If the prediction signals low future demand, the prices remain close to the reactive baseline and PDOP behaves similarly to the non-predictive primal–dual policy. Importantly, when $\phi_{e,t} \equiv 0$ and $\psi_{v,t} \equiv 0$, (10) reduces to (III-C), recovering the standard exponential pricing mechanism.

IV. Algorithmic Framework

The Predictive Dynamic Optimization and Pricing (PDOP) framework extends the DOP model by incorporating forecasts of future demand into an online primal-dual auction mechanism. PDOP processes video requests sequentially in arrival order and maintains dynamic dual prices for network resources. The framework has three main components described in the following subsections.

A. Access Management via Primal-Dual Admission Control

When a video request i arrives with bid b_i, PDOP computes the minimum serving cost cost_i using the current resource prices (including any predictive adjustments). A subroutine solves for the cheapest transcoding-multicast plan for i under the weight functions. If $b_i - \mathrm{cost}_i > 0$, video i is admitted ($x_i = 1$) and we record the dual gain $\mu_i = b_i - \mathrm{cost}_i$. Otherwise ($b_i \leq \mathrm{cost}_i$) the request is rejected ($x_i = 0$). This admission rule is truthful: overstating b_i risks paying a higher price, while understating b_i could cause a beneficial request to be denied.

Once a video is accepted, its multicast plan is fixed and the primal solution is updated. For every edge e and node v used by this plan, the algorithm increments the usage counters $\delta_e(t)$ or $\gamma_v(t)$ (for all relevant times t) by the video's resource requirements. These counters start at zero and accumulate as more videos are admitted, keeping track of the current allocation.

B. Dynamic Exponential Price Function Design

After each acceptance, PDOP updates resource prices to reflect utilization. We use exponential pricing: for each edge e and time t,

$$\alpha_e(\delta_e(t)) = (\xi_e)^{\delta_e(t)/C_e} - 1, \beta_v(\gamma_v(t)) = (\xi_v)^{\gamma_v(t)/C_v} - 1. \quad (11)$$

where C_e (resp. C_v) is the capacity of edge e (resp. node v), and the base factors ξ_e, ξ_v are defined as

$$\xi_e = 1 + \max_{i,l} \frac{b_{il}}{\sum_{e'} r_{ie'}^{(l)}}, \qquad \xi_v = 1 + \max_{i,l} \frac{b_{il}}{\sum_{v'} w_{iv'}^{(l)}}. \quad (12)$$

This design ensures boundedness ($\alpha_e(0) = 0$ when unused, and α_e grows to $\xi_e - 1$ at full capacity) and monotonicity (prices increase with usage). Initially, unused resources have zero price, encouraging utilization. As $\delta_e(t)$ increases, α_e grows slowly at first and sharply near saturation, reflecting scarcity. At full capacity ($\delta_e(t) = C_e$), α_e becomes $\xi_e - 1$, effectively deterring further use. Dynamic pricing aligns each video's payment with the network's current load and prevents over-commitment of capacity.

C. Optimal TMT Design using Steiner Tree Heuristic

For each admitted video i, PDOP must compute a minimum-cost transcoding multicast tree (TMT) from the source to all its destinations. The cost of a tree is the sum of edge usage and transcoding costs: sending one unit of stream on edge e at time t costs $\alpha_e(t)$, and each transcoding at node v costs $\beta_v(t)$. Exactly solving this Steiner-tree problem with capacity constraints is hard, so we use a polynomial-time heuristic (Algorithm 2, Abah).

The Abah heuristic works as follows:

- **Type grouping:** Group users by requested video type k.
- **Steiner tree per type:** Compute a minimum Steiner tree from the source to all users of type k (using a known 2-approximation algorithm).
- **Transcoding attachment:** For any user requiring a lower-quality stream that is not present at the source, attach a transcoding node at the branching point to downgrade the stream.
- **Tree merging:** Merge the per-type trees into one TMT by reusing common edges (already-used edges have zero additional cost for later types).
- **Implementation:** Allocate capacity along the merged tree (increment $\delta_e(t), \gamma_v(t)$ on all its edges/nodes) and begin streaming.

This yields a near-optimal multicast tree that covers all destinations. The chosen tree l^* has low cost (within a constant factor of optimal) and is then installed in the network.

D. Predictive Augmentation (PDOP)

PDOP incorporates predictions of future demand into the pricing and admission steps. Let $\phi_e(t)$ and $\psi_v(t)$ be predicted future utilization of edge e and node v at time t (e.g. from a forecasting model). We define *predictive* price functions:

$$\tilde{\alpha}_e(\delta_e(t)) = (\xi_e)^{(\delta_e(t)+\phi_e(t))/C_e} - 1, \tilde{\beta}_v(\gamma_v(t)) = (\xi_v)^{(\gamma_v(t)+\psi_v(t))/C_v} - 1. \quad (13)$$

using the same ξ_e, ξ_v as above. This effectively treats each resource as if pre-loaded by $\phi_e(t)$ or $\psi_v(t)$: if future load is expected to be high, the resource's price starts at a higher value, discouraging low-value current requests.

These predictive prices are used in the admission check. When video i arrives, the algorithm computes its serving cost using $\tilde{\alpha}_e, \tilde{\beta}_v$ and sets $\mu_i = b_i - \mathrm{cost}_i$. Video i is admitted only if $\mu_i > 0$. As a result, PDOP may reject some videos that DOP would accept, preserving capacity for higher-valued videos that are predicted to arrive later. If the predictions are accurate, this strategy can improve total welfare by avoiding myopic overuse. If predictions are poor or unavailable, setting $\phi_e(t) = \psi_v(t) = 0$ recovers the original DOP behavior, so PDOP remains robust.

Finally, note that the Steiner-tree routing (Sec. 3.3) does not change under PDOP. Once a video is admitted, we construct its TMT using the same heuristic as in DOP, and then allocate resources according to that tree.

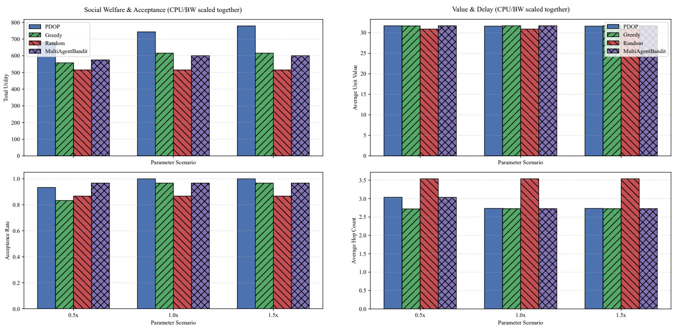
V. Simulation and Evaluation

We compare PDOP with three baselines (Greedy, Random, and MultiAgentBandit [14]). We report *total utility* (social welfare), *acceptance rate*, *average unit value* (utility per accepted task), and *average hop count* (path length, proxy for delay). We evaluate four settings: (i) Resource scaling (CPU/BW at 0.5×, 1.0×, 1.5×), (ii) Task load (20/35/50 tasks), (iii) Node count (10/15/20 nodes), and (iv) Edge count (20/25/30 edges).

Resource scaling (Fig. 1). PDOP achieves the highest utility at all resource levels. At 0.5× capacity, it improves welfare by $> 15\%$ over the best baseline while keeping ∼95% acceptance; at 1.0× and 1.5×, acceptance is near 100% and the utility gap persists (e.g., ∼20% over

Algorithm 1 Predictive DOP (PDOP) Algorithm

```
Initialize $\delta_e(t) = 0$, $\gamma_v(t) = 0$ for all $e, v, t$.
for each arriving video request $i$ do
    Compute $\text{cost}_i$ = minimum serving cost under prices $\tilde{\alpha}_e(\delta), \tilde{\beta}_v(\gamma)$.
    if $b_i \geq \text{cost}_i$ then
        Accept request ($x_i = 1$) and find multicast tree $l$ via the Steiner heuristic.
        Update $\delta_e(t), \gamma_v(t)$ along $l$ for all $t \in [a_i, t_i]$.
        Update prices $\alpha_e, \beta_v$ (via the exponential formula) based on new $\delta, \gamma$.
        Charge the operator $p_i = \text{cost}_i$.
    else
        Reject request ($x_i = 0$).
    end if
end for
```



(a): Utility and acceptance.

(b): Avg. value and hops.

Fig. 1: Resource scaling (0.5×/1.0×/1.5× CPU/BW). PDOP delivers the highest utility with near-100% acceptance and near-minimum hop count.

Greedy at 1.5×). PDOP also maintains short routes (within ∼0.2 hops of Greedy), whereas Random yields much longer paths.

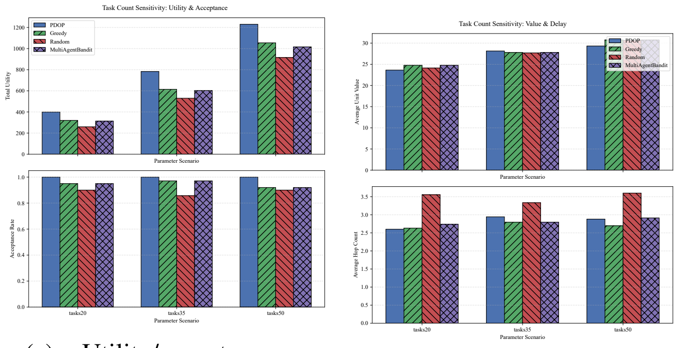
Task load (Fig. 2). As load increases, PDOP's utility grows almost linearly while keeping acceptance high. At 50 tasks, PDOP achieves ∼20% higher utility than Greedy and maintains ∼95% acceptance (baselines are lower). It also preserves higher unit value (about 30 vs. ∼27 for Greedy at 50 tasks) with low hop count (e.g., ∼2.9, ∼0.4 hops shorter than Greedy).

Node count (Fig. 3). PDOP remains best as the network grows. With 20 nodes, it delivers ∼25% higher utility than Greedy and accepts ∼95% of tasks (Greedy ∼88%). Despite admitting more tasks, PDOP still finds short routes (mean hops ∼2.5 vs. > 3.0 for Greedy).

Edge count (Fig. 4). Denser connectivity benefits all methods, but PDOP gains the most. At 30 edges, PDOP reaches utility ∼400 (vs. ∼340 for Greedy) with near-100% acceptance. It also exploits extra links to keep hop count low (around 2.5 vs. ∼3.2 for Greedy).

VI. Conclusion

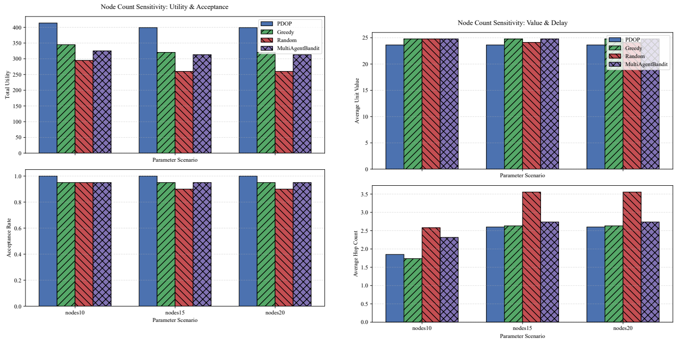
This work proposes PDOP, an online auction-based framework that jointly optimizes admission, pricing, and transcoding multicast routing in ECNs. By reformulating the original MINLP into an ILP via the CEO technique, PDOP enables an efficient primal–dual algorithm with exponential, utilization-aware resource prices and near-cost-minimal multicast tree construction. The framework provides strong theoretical guarantees (truthfulness, individual rationality, budget balance, and bounded online performance) and delivers higher social welfare and better network efficiency than baselines in simulations. Building on DOP, PDOP incorporates demand predictions to further improve welfare while remaining consistent under accurate forecasts and robust to prediction errors. Overall, the results highlight the benefits of co-designing economic incentives, routing, and learning for scalable, low-latency video delivery at the edge.



(a): Utility/acceptance vs. tasks.

(b): Value/hops vs. tasks.

Fig. 2: Task load (20/35/50 tasks). PDOP scales utility while sustaining high acceptance and low hop count under heavier demand.



(a): Utility/acceptance vs. nodes.

(b): Value/hops vs. nodes.

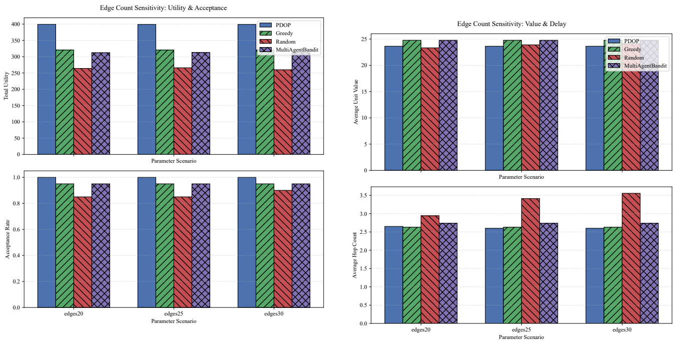
Fig. 3: Network size (10/15/20 nodes). PDOP yields the highest utility and acceptance, and achieves shorter average paths as the network scales.

VII. Appendix

Truthfulness. PDOP implements a truthful (incentive-compatible) admission and pricing rule. Let the true valuation of request i be v_i, and let b_i be the reported bid. Denote by p_i the charged payment if i is accepted, and $p_i = 0$ otherwise. The induced quasi-linear utility is

$$U_i(b_i; v_i) = \begin{cases} v_i - p_i, & \text{if } i \text{ is accepted under bid } b_i, \\ 0, & \text{otherwise.} \end{cases}$$

In PDOP, acceptance is determined by a threshold condition based on the minimum priced plan cost under the (predictively biased) resource prices: $\min_{j \in P_i} \text{pcost}_{ij}(\tilde{\lambda}, \tilde{\pi}) \leq b_i$, $\quad \text{pcost}_{ij}(\tilde{\lambda}, \tilde{\pi}) = \sum_{e,t} r_{ij}^{(e,t)} \tilde{\lambda}_{e,t} + \sum_{v,t} c_{ij}^{(v,t)} \tilde{\pi}_{v,t}$. Moreover, the payment is cost-based (threshold) and does not increase with b_i once the threshold is met. Hence bidding above v_i cannot increase U_i, while bidding below v_i may turn a winning outcome into a rejection and forfeit



(a): Utility/acceptance vs. edges.

(b): Value/hops vs. edges.

Fig. 4: Connectivity (20/25/30 edges). PDOP consistently achieves the best utility and acceptance, while maintaining the lowest (or near-lowest) hop count.

positive utility. Therefore, truthful reporting is a dominant strategy: $U_i(v_i; v_i) \geq U_i(b_i; v_i), \quad \forall\, b_i \neq v_i$.

Individual Rationality. If request i is accepted, PDOP selects a plan $j^\star(i)$ maximizing surplus $s_{ij}(\tilde{\lambda}, \tilde{\pi}) = b_i - \mathrm{pcost}_{ij}(\tilde{\lambda}, \tilde{\pi}), \quad j^\star(i) \in \arg\max_{j \in P_i} s_{ij}(\tilde{\lambda}, \tilde{\pi})$, and acceptance implies $s_{ij^\star(i)}(\tilde{\lambda}, \tilde{\pi}) \geq 0$. With the cost-based payment $p_i = \mathrm{pcost}_{ij^\star(i)}(\tilde{\lambda}, \tilde{\pi})$ (or an equivalent threshold payment), we have $U_i(v_i; v_i) = v_i - p_i \geq 0$ for every accepted request, i.e., PDOP is individually rational.

Budget Balance. PDOP is weakly budget-balanced: total collected payments cover total incurred resource cost under the implemented plans. Specifically, with $p_i = \mathrm{pcost}_{ij^\star(i)}(\tilde{\lambda}, \tilde{\pi})$ for each accepted request and $p_i = 0$ otherwise, the operator's revenue satisfies $\sum_i p_i \geq \sum_{e,t} \tilde{\lambda}_{e,t}\, \delta_{e,t} + \sum_{v,t} \tilde{\pi}_{v,t}\, \gamma_{v,t}$, where $\delta_{e,t}$ and $\gamma_{v,t}$ are the realized aggregate usage induced by the accepted plan selections. Thus PDOP does not require external subsidy.

Polynomial-Time Execution. Let $|P_i|$ be the number of candidate plans available for request i. Per arrival, PDOP evaluates $\mathrm{pcost}_{ij}(\tilde{\lambda}, \tilde{\pi})$ for all $j \in P_i$ and selects $j^\star(i)$, which is $O(|P_i|)$ given precomputed footprints $\{r_{ij}^{(e,t)}, c_{ij}^{(v,t)}\}$. If plans are generated online by a routing-and-transcoding heuristic, the per-arrival runtime is polynomial in $(|V|, |E|, |T|)$ for each generated plan, yielding overall runtime polynomial in the input size and $\sum_i |P_i|$.

Approximation Guarantee for Plan Construction. When P_i is produced via a Steiner-tree-based heuristic (e.g., a 2-approximation directed Steiner routine combined with a local branch-attach procedure for quality adaptation), the resulting plan cost is within a constant factor of the optimal per-request delivery cost. In particular, if OPT_i denotes the minimum feasible delivery cost for request i under the same prices, then the constructed plan satisfies

$$\mathrm{pcost}_{ij^\star(i)}(\tilde{\lambda}, \tilde{\pi}) \;\leq\; 2\,\mathrm{OPT}_i \;+\; \Delta_i,$$

where Δ_i captures the (bounded) overhead introduced by the attachment of quality-adaptation points.

Competitive Performance. Under adversarial arrivals, the primal–dual analysis for exponential pricing yields a logarithmic competitive ratio with respect to the offline optimum. Concretely, with exponential price scales κ_e, κ_v (or equivalently ξ_e, ξ_v), PDOP achieves welfare within an $O(\log \kappa)$ factor of the offline clairvoyant solution, where κ summarizes the range of value densities relative to capacities. This bound holds independent of prediction accuracy.

Learning-Augmented Consistency and Robustness. PDOP augments prices using predicted future load $(\phi_{e,t}, \psi_{v,t})$ through $\tilde{\lambda}_{e,t}(\delta_{e,t}) = (\kappa_e)^{(\delta_{e,t}+\phi_{e,t})/B_e^{(t)}} - 1, \tilde{\pi}_{v,t}(\gamma_{v,t}) = (\kappa_v)^{(\gamma_{v,t}+\psi_{v,t})/C_v^{(t)}} - 1$. If predictions are accurate (small error relative to capacities), the induced price bias steers admission toward near-optimal allocations and improves welfare, approaching the offline optimum up to the plan-construction approximation. If predictions are poor, PDOP remains robust: setting $\phi_{e,t} \equiv 0$ and $\psi_{v,t} \equiv 0$ recovers the non-predictive exponential pricing rule, and the worst-case logarithmic competitiveness is preserved. In all cases, PDOP maintains truthfulness, individual rationality, and budget balance because the acceptance threshold and payment depend on priced resource footprints and exogenous predictions, not on any bidder's report beyond meeting the threshold.

References

[1] M. Darwich and M. A. Bayoumi, *Enhancing Video Streaming with AI, Cloud, and Edge Technologies - Optimization Techniques and Frameworks*. Springer, 2025.

[2] W. Wang, X. Wei, W. Tao, M. Zhou, and C. Ji, "Quality of experience-oriented cloud-edge dynamic adaptive streaming: Recent advances, challenges, and opportunities," *Symmetry*, vol. 17, no. 2, p. 194, 2025.

[3] J. Chen, X. Li, L. Pan, and S. Liu, "A lyapunov optimization-based online algorithm for scheduling cloud-edge collaborative real-time video stream analytics tasks," *IEEE Internet Things J.*, vol. 12, no. 14, pp. 27 713–27 727, 2025.

[4] B. Qian, Y. Xuan, D. Wu, Z. Wen, R. Yang, S. He, J. Chen, and R. Ranjan, "Edge-cloud collaborative streaming video analytics with multi-agent deep reinforcement learning," *IEEE Netw.*, vol. 39, no. 1, pp. 165–173, 2025.

[5] P. Ye, W. Wang, B. Mi, and K. Chen, "Edgestreaming: Secure computation intelligence in distributed edge networks for streaming analytics," *ACM Trans. Multim. Comput. Commun. Appl.*, vol. 21, no. 8, pp. 222:1–222:15, 2025.

[6] S. Yuan, Y. Liu, S. Guo, J. Li, H. Chen, C. Wu, and Y. Yang, "Efficient online computing offloading for budget- constrained cloud-edge collaborative video streaming systems," *IEEE Trans. Cloud Comput.*, vol. 13, no. 1, pp. 273–287, 2025.

[7] C. Hu, Z. Chen, and E. G. Larsson, "Energy-efficient federated edge learning with streaming data: A lyapunov optimization approach," *IEEE Trans. Commun.*, vol. 73, no. 2, pp. 1142–1156, 2025.

[8] Q. Liu, H. Chen, Z. Li, Y. Bai, D. Wu, and Y. Zhou, "Online caching algorithm for VR video streaming in mobile edge caching system," *Mob. Networks Appl.*, vol. 30, no. 1, pp. 59–71, 2025.

[9] S. Yau, S. Awiphan, J. Bootkrajang, and J. Chaijaruwanich, "Edge-centric quality adaptation scheme for mobile video streaming," *IEEE Open J. Commun. Soc.*, vol. 6, pp. 6953–6965, 2025.

[10] Z. Ding, D. Kong, Z. Zhang, X. Xie, and J. Xu, "Cluspar: A game-theoretic approach for efficient and scalable streaming edge partitioning," *IEEE Trans. Computers*, vol. 74, no. 1, pp. 116–130, 2025.

[11] J. Lu, Z. Ma, Y. Gao, S. Yue, J. Ren, and Y. Zhang, "Streamsys: A lightweight executable delivery system for edge computing," *IEEE Trans. Cloud Comput.*, vol. 13, no. 1, pp. 213–226, 2025.

[12] I. Gokarn, Y. Hu, T. F. Abdelzaher, and A. Misra, "*RA-MOSAIC*: Resource adaptive edge AI optimization over spatially multiplexed video streams," *ACM Trans. Multim. Comput. Commun. Appl.*, vol. 21, no. 9, pp. 247:1–247:25, 2025.

[13] Q. Hu, Q. He, H. Zhong, G. Lu, X. Zhang, G. Zhai, and Y. Wang, "VARFVV: view-adaptive real-time interactive free-view video streaming with edge computing," *IEEE J. Sel. Areas Commun.*, vol. 43, no. 7, pp. 2620–2634, 2025.

[14] M. A. Raza, M. Abolhasan, J. Lipman, N. Shariati, W. Ni, and A. Jamalipour, "Multi-agent multi-armed bandit learning for grant-free access in ultra-dense iot networks," *IEEE Trans. Cogn. Commun. Netw.*, vol. 10, no. 4, pp. 1356–1370, 2024.