Incentivizing Online Edge Caching via Auction-Based Subsidization

Youmei Song[1], Lei Jiao[2], Renyu Yang[3], Tianyu Wo[1], Jie Xu[3]
[1]Beihang University, China [2]University of Oregon, USA [3]University of Leeds, UK

***Abstract*—There exists a practical need for incentivizing content providers to cache contents at distributed network edges closer to users. However, this is a particularly challenging problem due to system environments that are uncertain, content placements that couple adjacent time slots, and economic properties that are desired but hard to ensure. In this paper, we present our design of an auction-based incentive mechanism for online edge caching. We formulate the long-term social cost minimization problem as a nonlinear mixed-integer program that addresses bid selections, user request dispatching, content placements, and payment determination in repetitive auctions. To solve this problem online, we devise a greedy approximation algorithm for solving each auction individually, and a lazy-replacement-based online algorithm that ties the series of auctions over time while dynamically pursuing the balance between downloading contents to new cache locations and keeping them at existing locations. We formally prove the approximation ratio for each single auction, the competitive ratio for the long-term social cost, as well as the truthfulness, the individual rationality, and the computational efficiency of our approach. Evaluations with real-world data have also validated and confirmed the practical superiority of our approach over multiple alternative algorithms.**

I. Introduction

Edge caching refers to caching or placing contents at distributed "edges", such as micro data centers or server clusters co-located with cellular base stations, WiFi access points, and neighbourhood spots in close proximity to end users [1], [2]. It promises ultra-low latency and traffic localization for content access, benefiting both end users and *Content Providers (CPs)*. In lots of cases, the edge caching infrastructures are often built and managed by the *Edge Network Operator (ENO)* that owns and operates the edge networks, charging end users for network access [3], [4]. This scenario is illustrated in Fig. 1.

Unfortunately, in reality, CPs rarely, if not never, leverage such emerging edge facilities for content caching. First, CPs already have sophisticated business models and experiences with Content Delivery Networks (CDNs) consisting of caches across locations [5], [6]. While CDNs have seen great success in reducing content access delay and improving end users' perceived service quality, using edge caching seems to result in only negligible further delay reduction and limited added-value for CPs. Second, there exists no well-established revenue sharing mechanism between CPs and the ENO [3]. While CPs often pay Internet Service Providers (ISPs) (which are not necessarily the ENOs) for network connections, they do not typically pay the ENO that operates access networks. The ENO does not pay CPs either, and may even limit the bandwidth as users can generate enormous traffic when accessing contents.

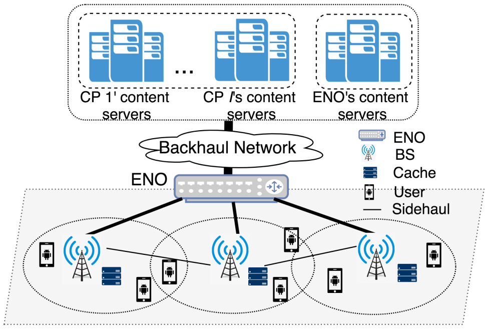

Fig. 1: Edge networks with content caching

Therefore, there is a strong case in practice for the ENO to incentivize CPs to cache their contents at distributed edges via (monetary) subsidization. This could potentially lead to a win-win situation and global optimization: CPs can gain direct economic benefits and be motivated to utilize edge caching, while the ENO can make the optimal use of cache and network resources, offering best services to CPs and attracting more end users. However, designing and implementing such an incentive mechanism is non-trivial and particularly challenging.

First, the incentivization is not a one-time transaction, but needs to be continuously conducted over time while adapting to unpredictable system dynamism and uncertainties. While bandwidth, delay, storage overhead, and end users' requests may all be time-varying, the ENO needs to determine on the fly which "packages" of contents to purchase from the CPs, and for each content, where to place and replicate it within the edge caching networks. Sometimes, the ENO may also have its own supplementary content servers [7], which need to be jointly managed for global optimal performance. More importantly, content caching decisions are intrinsically coupled over time—for example, in the case of no user requests toward a cache location, continuing to keep a content there incurs storage cost in the current time slot, but will save downloading cost for the next time slot (as the content is already on premises) if it is requested by users then; however, if it is not requested in the next time slot, the storage cost incurred for the current time slot will be wasted. Without knowing future inputs, it is not easy to irrevocably determine content caching at each time slot for long-term total cost optimization.

Second, each incentive transaction needs to guarantee the desired economic properties. In contrast to fixed pricing, we focus on the auction-based approach in this paper as auctions can easily reflect real-time market dynamics and match demands from the ENO (i.e., the auctioneer) and supplies from the CPs (e.g., the bidders) agilely. For market efficiency, each auction is expected to be truthful (i.e., each bid can maximize its utility only by bidding its true price) and individually rational (i.e., each bid always achieves non-negative utility), where "utility" of a bid refers to the difference between the received payment and the bidding price it wants to charge. The classic Vickrey–Clarke–Groves (VCG) auctions [8] achieve these properties, but entail solving the underlying cost minimization problem optimally; our problem turns out to be NP-hard, and contains other control decisions beyond winning-bid selections in each auction, making VCG computationally infeasible and calling for the novel auction mechanism design.

Although there exist substantial research on edge caching algorithms and optimizations [1], [2], [9]–[14], they generally lack systematic studies of the economic interactions between different entities in the system. Prior works on edge caching incentives [3], [4], [8], [15]–[17] do not specifically focus on the interplay between the ENO and the CPs, or do not explore the auction perspective, not to mention continuous dynamic auctions, failing to address the challenges above. See Section VI for details. In this paper, we make multiple contributions:

We formulate a long-term social cost minimization problem for the ENO to dynamically cache contents from the CPs in distributed edge networks to serve end users. We optimize the sum of (i) the ENO's cost, including cost of user requests dispatching, storage overhead of content caching, downloading cost of content transference from content servers to edge caches, cost of accessing the ENO's own content servers, and the payments made to CPs, and (ii) the CPs' cost, mainly the cost of bids minus the received payments (i.e., treated as negative cost). Our formulation is expressive and can capture arbitrary system dynamics. Our problem is an intractable, nonlinear mixed-integer program defined via a series of auctions corresponding to consecutive time slots.

We design polynomial-time algorithms to solve this problem. We first temporarily ignore the content downloading cost that could couple adjacent time slots, and design a greedy-based approximation algorithm that determines the winning bids of the CPs, the access to ENO's own content servers, and the user request dispatching for each individual auction assuming that contents' caching locations in the edge networks are given. We then design an online algorithm to determine such content caching locations dynamically without knowing future inputs and tie these single auctions over time for continuous incentivization. Our approach postpones changing content locations across edges until the cumulative storage cost since the last location-changing operation exceeds a pre-specified parameter times the last content downloading cost, preventing frequent content replacements while serving users.

We rigorously analyze the worst-case theoretical guarantees provided by our algorithms. We prove that our single-auction mechanism achieves a constant approximation ratio towards each auction's own optimal social cost. Satisfying the sufficient conditions of bidding monotone and critical payment [18], we also prove that our mechanism guarantees the desired economic properties of truthfulness and individual rationality. Absorbing the approximation ratio of each auction, we finally prove that our online algorithm achieves a parameterized-constant competitive ratio, i.e., the long-term social cost incurred online does not exceed this ratio times the optimal long-term social cost offline where inputs to all the auctions are envisaged to be known before the entire time horizon starts.

We finally carry out extensive evaluations to validate the practical performance of our approach using real-world inputs based on Youtube contents [19], hot spot edge locations [20], and Amazon EC2 prices [21] in different settings. We find that our approach (i) saves social cost by $5\% \sim 45\%$ compared to a greedy caching approach and the series of single-shot optimums, (ii) achieves empirically an approximation ratio of about 2 for each individual auction and a competitive ratio of $1.2 \sim 1.4$ with regards to the offline optimum, (iii) is robust for cache capacities, adjustable to seek trade-offs among different cost components, (iv) preserves truthful bidding and individual rationality in practice, and (v) runs very fast compared to the length of the time horizon under consideration.

II. Modeling and Formulation

A. System Models

We summarize the key notations in Table 1.

Cache Infrastructures: We consider a set $\mathcal{N}$ of distributed base stations managed by an Edge Network Operator (ENO). These base stations connect to one another via sidehaul links (e.g., X2) [22] and also connect to the Internet via backhaul networks. We study the system over a series of consecutive time slots $\mathcal{T} = \{1, 2, .., T\}$. Each base station $n \in \mathcal{N}$ has a co-located cache that can host at most C_n contents. At the time slot $t \in \mathcal{T}$, we use s_n^t to denote the content hosting cost (e.g., storage occupancy, maintenance overhead, or energy consumption) of caching a content at the base station n, use b_n^t to denote the downloading or replacement cost (e.g., ingress cost for edge networks, delay of loading the content to storage) for the base station n, and also use $b_{n,m}^t$ to denote the unit sidehaul transference cost between the base stations n and m (e.g., delay or bandwidth consumption within edge networks).

Contents and Sources: We consider a set $\mathcal{F} = \{1, 2, ..., F\}$ of contents for the entire system. Without loss of generality, we assume that every single content has equal size. In reality, this assumption can be removed as contents can be divided into blocks of the same size with coding techniques and then each block can be considered as a content. We consider a set $\mathcal{I} = \{1, 2, ..., I\}$ of Content Providers (CPs) with their content servers that host contents. In this paper, we also consider that the ENO can have its own content servers, for the cases where CPs may not offer every single content requested by users or CPs may offer some contents at very expensive price. The ENO's own content servers may be hosted in a public or private cloud. Then, we use c_f^t to denote the access cost (e.g., cloud egress cost) for accessing the content f from the ENO's own content servers [7] at the time slot t.

User Requests and Processing: We use $\lambda_{f,n}^t$ to denote the number of requests from end users sent to the base station n to access the content f at the time slot t. We also use an indicator

TABLE I: Notations

Inputs	Meaning
$\mathcal{T}$	Set of time slots
$\mathcal{N}$	Set of Base Stations (BSes)
$\mathcal{F}$	Set of contents
$\mathcal{I}$	Set of Content Providers (CPs)
C_n	Capacity of BS (i.e., cache) n
s_n^t	Per-content hosting cost at BS n at the time slot t
b_n^t	Per-content downloading cost for BS n at t
$b_{n,m}^t$	Cost of sidehaul transference between BSes n and m at t
c_f^t	Cost of accessing content f from ENO at t
$\lambda_{f,n}^t$	Number of requests sent to BS n for content f at t
$\delta_{f,n}^t$	1 if $\lambda_{f,n}^t > 0$, and 0 if $\lambda_{f,n}^t = 0$
b_i^t	Bidding price of the bid i at t
$\mathcal{Q}_i^t$	Set of contents offered by the bid i at t

Decisions	Meaning
x_i^t	Whether to choose the bid i as a winning bid at t
w_f^t	Whether to fetch the content f from ENO at t
$z_{f,n,m}^t$	Proportion of requests for content f received at BS n and dispatched to BS m at t
$y_{f,n}^t$	Whether to cache the content f at BS n at t
p_i^t	Payment made to the bid i at t

$\delta_{f,n}^t$, with $\delta_{f,n}^t = 1$ if $\lambda_{f,n}^t > 0$ and $\delta_{f,n}^t = 0$ if $\lambda_{f,n}^t = 0$. Requests received at one base station may be dispatched to and served by (the cache of) another base station via sidehaul links, if the requested contents are not found locally.

Auction Model: The ENO is the auctioneer and conducts an auction at every time slot t where the CPs act as the bidders. Fig. 2 depicts the operation of the auction at any t. First, the ENO solicits bids and the CPs submit bids. Each CP $i \in \mathcal{I}$ submits a bid $\{\mathcal{Q}_i^t, b_i^t\}$, where $\mathcal{Q}_i^t$ represents the set of contents offered by this bid (where we denote $|\mathcal{Q}_i^t| = Q_i^t$) and b_i^t is the bidding price, i.e., the amount of money this bid wants to charge for granting access to the contents $\mathcal{Q}_i^t$ during the time slot t. Second, the ENO decides the set of bids to purchase, i.e., the "winning" bids, and the cache location in the edge network to place each content contained in the winning bids. The ENO considers bids from both sources—from the CPs and from the ENO's own content servers—to purchase, download, and cache contents. Each downloaded content can be replicated and cached at multiple base stations simultaneously. Third, the ENO decides the payment for each winning bid and makes the payment to each corresponding CP. Note that the payment for a bid is not necessarily the same as the bidding price of a bid, because the calculation of the payment is often expected to satisfy the economic properties of "individual rationality" and "truthfulness", which will be elaborated later. Fourth, the ENO transfers each content from either the CPs or its own content servers to the decided location in the edge network. The fourth step may alternatively happen before the third step, depending on the agreement between the ENO and the CPs.

Control Decisions: The ENO makes the following control decisions: (1) $x_i^t \in \{1, 0\}$, denoting whether or not the bid i wins the auction at the time slot t; (2) $w_f^t \in \{1, 0\}$, denoting whether or not the ENO accesses the content f from its own content servers for caching at the time slot t; (3) $z_{f,n,m}^t \in [0, 1]$, denoting the proportion of user requests received at the base station n for the content f and dispatched to the base station m at the time slot t; (4) $y_{f,n}^t \in \{1, 0\}$, denoting whether or not to place or cache the content f at the base station n at the time slot t; (5) $p_i^t \in [0, +\infty)$, denoting the payment made by the ENO corresponding to the bid i at the time slot t.

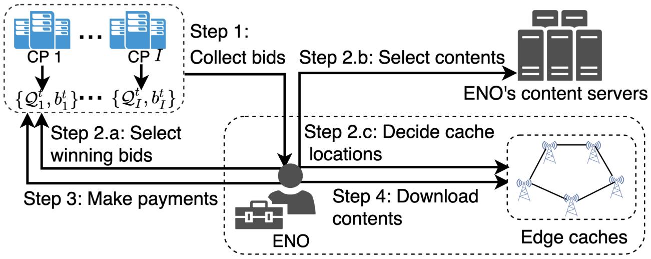
CP 1
CP I
$\{\mathcal{Q}_1^t, b_1^t\}$
$\{\mathcal{Q}_I^t, b_I^t\}$
Step 1: Collect bids
Step 2.b: Select contents
ENO's content servers
Step 2.a: Select winning bids
Step 2.c: Decide cache locations
Step 3: Make payments
Step 4: Download contents
ENO
Edge caches

Fig. 2: Auction in each time slot

Total Cost of ENO: At each time slot t, the ENO incurs the following types of costs: (1) the sidehaul transfer cost of dispatching users' requests across base stations, i.e., $\sum_f \sum_n \sum_m b_{n,m}^t \lambda_{f,n}^t z_{f,n,m}^t$; (2) the cost of hosting contents in edge caches, i.e., $\sum_f \sum_n s_n^t y_{f,n}^t$; (3) the replacement cost incurred by downloading contents from their original sources, either the CPs' or the ENO's content servers, and writing such contents into new cache locations, i.e., $\sum_f \sum_n b_n^t [y_{f,n}^t - y_{f,n}^{t-1}]^+$, where $[\cdot]^+ = \max\{\cdot, 0\}$ (note that this means there is no replacement cost if cache locations are unchanged, as there is no need to re-download the same content to the same location); (4) the cost of accessing contents on the ENO's own content servers, i.e., $\sum_f c_f^t w_f^t$; (5) the payment from the ENO to the CPs, i.e., $\sum_i p_i^t x_i^t$. The total cost of the ENO at t is

$$C_{ENO}^t = \sum_f \sum_n \sum_m b_{n,m}^t \lambda_{f,n}^t z_{f,n,m}^t + \sum_f \sum_n s_n^t y_{f,n}^t + \sum_f \sum_n b_n^t [y_{f,n}^t - y_{f,n}^{t-1}]^+ + \sum_f c_f^t w_f^t + \sum_i p_i^t x_i^t.$$

Total Cost of CPs: From the CPs' perspective, at each time slot t, there exist two types of costs: (1) the cost incurred by making contents available to the ENO via bidding, i.e., $\sum_i b_i^t x_i^t$, and (2) the received payment (treated as negative cost), i.e., $\sum_i p_i^t x_i^t$. Therefore, the total cost of all CPs at t is

$$C_{CP}^t = \sum_i (b_i^t - p_i^t) x_i^t.$$

B. Problem Formulation and Algorithmic Challenges

Social Cost Minimization: The "social cost" of the system is the sum of the total cost of the ENO and the total cost of the CPs. We formulate the social cost minimization problem $\mathbb{P}_0$ below (note that payments are cancelled automatically, but still need to be decided later):

$$\begin{aligned}
\mathbb{P}_0: \ & \min \sum_t C^t = \sum_t (C_{ENO}^t + C_{CP}^t) && \text{(2a)}\\
& s.t. \ y_{f,n}^t \le \sum_{i: f \in Q_i} x_i^t + w_f^t, \forall f, \forall n, \forall t, && \text{(2b)}\\
& z_{f,n,m}^t \le y_{f,m}^t, \forall f, \forall n, \forall m, \forall t, && \text{(2c)}\\
& \sum_f y_{f,n}^t \le C_n, \forall n, \forall t, && \text{(2d)}\\
& \sum_m z_{f,n,m}^t \ge \delta_{f,n}^t, \forall f, \forall n, \forall t, && \text{(2e)}\\
& x_i^t \in \{0, 1\}, w_f^t \in \{0, 1\}, \forall i, \forall f, \forall t, && \text{(2f)}\\
& z_{f,n,m}^t \ge 0, y_{f,n}^t \in \{0, 1\}, \forall f, \forall n, \forall m, \forall t. && \text{(2g)}
\end{aligned}$$

Constraint (2b) ensures that a content can only be cached if the content is contained in any purchased bid or fetched from

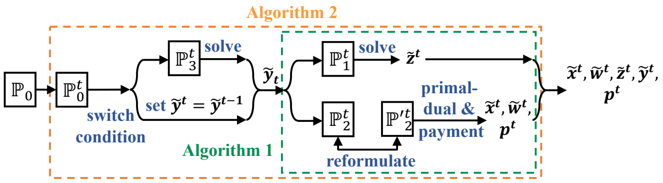

Fig. 3: Algorithm design

the ENO's own content servers. Constraint (2c) ensures that requests can only be dispatched to a base station if the content to be accessed by such requests is cached at that base station. Constraint (2d) ensures the capacity of each cache is respected. Constraint (2e) indicates that all the original requests received at each base station from users are fully dispatched.

Algorithmic Challenges: Solving the social cost minimization problem *in an online manner* confronts critical challenges. First, it is non-trivial to optimize the cache replacement cost online, which couples every pair of adjacent time slots. For example, at the time slot $t-1$, we can determine $y_{f,n}^{t-1}$ to minimize $b_n^t[y_{f,n}^{t-1} - y_{f,n}^{t-2}]^+$ as we already know $y_{f,n}^{t-2}$; however, as we do not know $y_{f,n}^t$ yet, which will only be determined at the next time slot t, our $y_{f,n}^{t-1}$ determined at $t-1$ can therefore hardly minimize $b_n^t[y_{f,n}^t - y_{f,n}^{t-1}]^+$. Second, the social cost minimization problem is in fact NP-hard and difficult to handle in the offline setting, not to mention online. Even without the cache replacement cost, our problem consists of a series of single-round problems, each of which can be reduced from the weighted set cover problem, known to be NP-hard. Third, while solving this problem to determine the winning bids, we need to determine the payments accordingly and ensure truthfulness and individual rationality, which are the keys to designing efficient auctions. This is not straightforward, particularly given the existence of the former two challenges.

III. Single-Auction Mechanism

In this section, we design a polynomial-time approximation algorithm to determine winning bids, payments, content access for different sources, and user request dispatching for a single auction, assuming content placements are already determined. We rigorously prove the approximation ratio, the truthfulness, and the individual rationality as the performance guarantees for our algorithm. We then determine content placements strategically in an online manner in the next section. Our entire approach can be structured as in Fig. 3.

A. Problem Decomposition

At t, if the value of $\mathbf{y}_t$ is given, denoted by $\tilde{\mathbf{y}}_t$, then we can split the original problem at t into the two subproblems $\mathbb{P}_1^t$ regarding $\mathbf{z}_t$ and $\mathbb{P}_2^t$ regarding $\mathbf{x}_t$ and $\mathbf{w}_t$, respectively:

$$\begin{aligned}\mathbb{P}_1^t:\ &\min\ h^t(\mathbf{z}^t) = \textstyle\sum_f \sum_n \sum_m b_{n,m}^t \lambda_{f,n}^t z_{f,n,m}^t\\ &s.t.\ z_{f,n,m}^t \le \tilde{y}_{f,m}^t, \forall f, \forall n, \forall m,\\ &\textstyle\sum_m z_{f,n,m}^t \ge \delta_{f,n}^t, \forall f, \forall n,\\ &z_{f,n,m}^t \ge 0, \forall f, \forall n, \forall m.\end{aligned}$$

$$\mathbb{P}_2^t:\ \min\ g^t(\mathbf{x}^t, \mathbf{w}^t) = \textstyle\sum_i b_i^t x_i^t + \sum_f w_f^t c_f^t \tag{4a}$$

$$s.t.\ \textstyle\sum_{i:f\in\mathcal{Q}_i} x_i^t + w_f^t \ge \max_n \tilde{y}_{f,n}, \forall f, \tag{4b}$$

$$x_i^t, w_f^t \in \{0,1\}, \forall i, \forall f. \tag{4c}$$

While $\mathbb{P}_1^t$ is in fact a linear program which can be optimally solved in polynomial time by any standard linear program solver, $\mathbb{P}_2^t$ is NP-hard, which can be shown by exhibiting that the weighted set cover problem is in fact a special case of $\mathbb{P}_2^t$. For the ease of presentation, we omit the time index t of variables and parameters for the time being. For $\mathbb{P}_2^t$, consider $\{\mathcal{Q}_1, ..., \mathcal{Q}_I, \mathcal{Q}_{I+1}, ..., \mathcal{Q}_{I+F}\}$, with its indices $\mathcal{L} = \{1, ..., I, I+1, ..., I+F\}$. For $1 \le l \le I$, $\mathcal{Q}_l$ refers to the set of contents offered by the CP l's bid; for $I < l \le I+F$, $\mathcal{Q}_l$ refers to the single-element set $\{f\}$, where $f = (l - I) \in \mathcal{F}$ represents the offered content, from the ENO. Accordingly, each set l has its cost e_l, i.e., $e_l = b_l$ if $1 \le l \le I$ and $e_l = c_{l-I}$ if $I < l \le I+F$. Then, $\mathbb{P}_2^t$ is to determine whether to select each set l in order to minimize the total cost while covering every content f denoted by $\tilde{y}_f = \max_n \tilde{y}_{f,n}$. This is actually the weighted set cover problem, which is NP-hard. Thus, $\mathbb{P}_2^t$ is also NP-hard. Based on what has been discussed above, the single-round problem of $\mathbb{P}_2^t$ involving the variables $\mathbf{x}^t, \mathbf{w}^t$ can be reformulated as below.

$$\mathbb{P}_2'^t:\ \min\ \textstyle\sum_l e_l \gamma_l \tag{5a}$$

$$s.t.\ \textstyle\sum_{l:f\in\mathcal{Q}_l} \gamma_l \ge \tilde{y}_f, \forall f, \tag{5b}$$

$$\gamma_l \in \{0,1\}, \forall l. \tag{5c}$$

B. Primal-Dual-Based Auction Mechanism

We focus on designing a polynomial-time approximation algorithm for $\mathbb{P}_2'^t$, with not only a provable approximation ratio but also the provable truthfulness and individual rationality.

We relax and reformulate $\mathbb{P}_2'^t$ as below, where γ_l is the decision variable:

$$P:\ \min\ \textstyle\sum_l e_l \gamma_l \tag{6a}$$

$$s.t.\ \textstyle\sum_{l:f\in\mathcal{Q}_l} \gamma_l \ge \tilde{y}_f, \forall f, \tag{6b}$$

$$0 \le \gamma_l \le 1, \forall l. \tag{6c}$$

We refer to this new formulation as our *primal* problem and correspondingly derive the Lagrange *dual* problem below:

$$D:\ \max\ \textstyle\sum_f v_f \tilde{y}_f \tag{7a}$$

$$s.t.\ \textstyle\sum_{f\in\mathcal{Q}_l} v_f \le e_l, \forall l, \tag{7b}$$

$$v_f \ge 0, \forall f. \tag{7c}$$

where v_f, $\forall f$ are the dual variables. In this derivation, note that $\gamma_l \le 1$ in (6c) is not needed, as it is inherently captured by (6b). With the help of duality, we can derive an approximation ratio for our algorithm, as elaborated next.

We design Algorithm 1 which is a greedy-based primal-dual algorithm. Our key idea is selecting the set that has the lowest "cost density" in each iteration until all the elements (i.e., contents) are covered, while calculating the corresponding payment based on the second lowest cost density and maintaining a feasible dual solution to connect the primal objective to the dual objective for proving the approximation ratio. We introduce the following additional notations. We define $\mathcal{U} = \{f | \tilde{y}_f = 1, \forall f\}$ and $\mathcal{F}_l = \mathcal{Q}_l \cap \mathcal{U}$, $\forall l$ where

Algorithm 1: Single-Auction Mechanism

```
1  Inputs: $\mathcal{U}$, $\mathcal{F}_l$, $e_l$, $\forall l$; $\widetilde{\mathbf{y}}$, $b_{n,m}$, $\lambda_{f,n}$, $\delta_{f,n}$, $\forall n, m, f$.
2  Initialization: $r_l = 0, x_i = 0, w_f = 0$, $v_f = 0$, $z_{f,n,m} = 0$,
   $\hat{\mathcal{F}}_l = \mathcal{F}_l$, $\forall l, i, f, n, m$; $\mathcal{S} = \emptyset$.
3  while $\exists f \in \mathcal{U}$ not covered do
4      $l^* = \arg\min_{l:\hat{\mathcal{F}}_l \neq \emptyset} \frac{e_l}{\hat{F}_l}$;
5      if $l^* \leq I$ then
6          $\gamma_{l^*} = x_{l^*} = 1$, $\mathcal{S} = \mathcal{S} \cup \{l^*\}$;
7          $l^- = \arg\min_{l \in \mathcal{L} \setminus l^*} \frac{e_l}{\hat{F}_l}$;
8          $p_{l^*} = F_{l^*} \cdot \frac{e_{l^-}}{\hat{F}_{l^-}}$;
9      else
10         $\gamma_{l^*} = w_{l^*-I} = 1, S = S \cup \{l^*\}$;
11     end
12     $v_f = \frac{1}{H_U} \frac{e_{l^*}}{\hat{F}_{l^*}}$, $\forall f \in \hat{\mathcal{F}}_{l^*}$;
13     $\hat{\mathcal{F}}_l = \hat{\mathcal{F}}_l \setminus \mathcal{F}_{l^*}$, $\forall l$;
14 end
15 Given $\widetilde{\mathbf{y}}$, obtain the optimal solution $\widetilde{\mathbf{z}}$ for $\mathbb{P}_1^t$;
16 return $\widetilde{\mathbf{x}}, \widetilde{\mathbf{w}}, \widetilde{\mathbf{z}}$.
```

we further denote $|\mathcal{U}| = U$ and $|\mathcal{F}_l| = F_l$. In each iteration (starting in Line 4), the algorithm selects the set with the lowest "cost density", i.e., the ratio of the cost or weight of the set over the number of the uncovered elements in this set. If the selected set l is offered by the CPs (Line 5), then l becomes a winning bid and x_l is updated (Line 6). Accordingly, the ENO calculates the payment for l via the lowest cost density excluding the set l (Lines 7-8). This is the key to ensuring truthfulness and individual rationality, which will be elaborated later. Otherwise, if the selected set l is offered by the ENO (Line 9), then w_{l^*-I} is updated (Line 10). After the winner determination, the dual variables of each covered element are updated based on the cost density (Line 12), where $H_U = 1 + \frac{1}{2} + \frac{1}{3} + ... + \frac{1}{U}$. These covered elements are also removed from the unselected sets, ensuring that each element can only be covered once (Line 13). Finally, we also obtain the optimal user request dispatching solution by invoking any standard linear program solver (Line 15).

C. Performance Analysis

Theorem 1. Complexity and Feasibility. Algorithm 1 produces feasible solutions to the primal problem P and the dual problem D in polynomial time.

Proof. Polynomial Time: The "while" loop of Algorithm 1 contains at most L iterations, where $|\mathcal{L}| = L$, to select one set in each iteration. Lines 4-11 and Line 13 take at most $O(LF)$; Line 12 takes at most $O(F)$. Hence, the time complexity of Algorithm 1 is $O(L(LF + F))$, plus the complexity of Line 15, depending on the solver. For example, the interior point method converges in $O(L)^{3.5}$ in general for L variables.

Primal Feasibility: Line 3 ensures that the algorithm only terminates when all elements in $\mathcal{U}$ are covered, satisfying constraints (4b). Constraints (4c) holds, as variables are initialized to 0 (Line 2) and some are updated to 1 (Lines 6 and 10).

Dual Feasibility: Suppose $\mathcal{F}_l$ is not selected by Algorithm 1. We let the elements in $\mathcal{F}_l$ be listed in the reverse order in which they were covered by Algorithm 1. As the "while" loop covers $\mathcal{F}_l$'s element f_k, $\mathcal{F}_l$ has at least k elements uncovered. Then, the cost density of $\mathcal{F}_l$ is at most $\frac{e_l}{k}$. Recall that Algorithm 1 always selects the set with the smallest cost density, so the cost density of the set selected by Algorithm 1 to cover f_k cannot be greater than $\frac{e_l}{k}$. Suppose $\mathcal{F}_j^{'}$ with the cost e_j is the set selected Algorithm 1 to cover f_k, then we have

$$\begin{aligned} \textstyle\sum_{f \in \mathcal{Q}_l} v_f &= \tfrac{1}{H_U} \textstyle\sum_{f_k \in \mathcal{F}_l} \tfrac{e_j}{F_j^{'}} \\ &\leq \tfrac{1}{H_U} \textstyle\sum_{k=1}^{F_l} \tfrac{e_l}{k} \\ &\leq \tfrac{e_l}{H_U} H_U \\ &\leq e_l. \end{aligned} \tag{8a}$$

If $\mathcal{F}_l$ is selected, the inequality (8a) still holds. Thus, the right-hand side of (7b) is always bounded by e_l. □

Theorem 2. Approximation Ratio. Algorithm 1 outputs a solution which incurs the cost at most α times the optimal cost of the problem $\mathbb{P}_2^t$, where $\alpha = H_U$.

Proof. Let p be the value of (6a) evaluated using the solutions returned by Algorithm 1. Suppose Algorithm 1 requires K ($K \leq L$) iterations in total in the "while" loop to cover all the elements in $\mathcal{U}$. Denote by $\mathcal{F}_k^{'}$ the set picked up by Algorithm 1 at each the iteration k ($k = \{1, 2, ..., K\}$). Then

$$p = \textstyle\sum_l e_l \gamma_l = \sum_{k=1}^{K} e_k \gamma_k = \sum_{k=1}^{K} e_k.$$

Denote by d the value of (7a). According to Line 12 of Algorithm 1, we have

$$\begin{aligned} d &= \textstyle\sum_{f \in \mathcal{F}} v_f \widetilde{y}_f \\ &= \textstyle\sum_{f \in \mathcal{U}} v_f \\ &= \textstyle\sum_{k=1}^{K} \sum_{f \in \mathcal{F}_k^{'}} \tfrac{1}{H_U} \tfrac{e_k}{F_k^{'}} \\ &= \tfrac{1}{H_U} \textstyle\sum_{k=1}^{K} e_k. \end{aligned} \tag{9a}$$

Therefore, we have $d = \frac{1}{H_U} p$. By duality, we further have $d \leq OPT_{cost}$, where OPT_{cost} is the optimal cost of $\mathbb{P}_2^t$. Joining them together, we have $p = d \cdot H_U \leq H_U \cdot OPT_{cost}$. □

In the rest of this section, we concentrate on the economic properties. We firstly define utility, and then based on it, we define truthfulness and individual rationality, respectively, and prove that our proposed Algorithm 1 achieves both of them.

Definition 1. Utility. The utility of a winning bid i is $u_i(b_i, \mathbf{b}_{-i}) = p_i - vc_i$, where b_i is the bidding price of the bid i, $\mathbf{b}_{-i}$ represents the bidding prices of all the other bids, p_i represents the received payment for the bid i, and vc_i represents the true cost of the bidder for composing the bid i or making the bid i available. The utility of a losing bid i is $u_i(b_i, \mathbf{b}_{-i}) = 0$.

Theorem 3. Truthfulness. An auction is truthful if for any bid i using its true cost as the bidding price leads to its maximum utility, i.e., $u_i(vc_i, \mathbf{b}_{-i}) \geq u_i(b_i, \mathbf{b}_{-i})$, for any $b_i \neq vc_i$ and any given $\mathbf{b}_{-i}$, $\forall i$. Algorithm 1 achieves truthfulness.

Proof. According to Myerson's theorem [18], an auction with bidding prices $\{e_l | \forall l \in \mathcal{I}\}$ and payments $\{p_l | \forall l \in \mathcal{I}\}$ is truthful if and only if *(i)* the auction result x_l is monotonically non-increasing in $e_l, \forall l \in \mathcal{I}$; and *(ii)* the winners are paid with the "critical" value. Hence, with Lemmas 1 and 2 as below and their proofs, we can conclude that Algorithm 1 is truthful. □

Lemma 1. Our auction is bid-monotonic. That is, for any bid $l \in \mathcal{I}$, compared to its original bidding price e_l, bidding a lower price $\widetilde{e}_l \le e_l$ leads to $\widetilde{x}_l = 1$ accordingly, if $x_l = 1$.

Proof. Suppose CP l submits $\mathcal{F}_l$ with the price e_l and wins the auction, i.e., $x_l = 1$. Then, if CP l submits $\mathcal{F}_l$ with a cheaper price $\widetilde{e}_l \le e_l$, we will have $\frac{\widetilde{e}_l}{F_l} \le \frac{e_l}{F_l}$. Thus, according to Algorithm 1, the new bid $\{\mathcal{F}_l, \widetilde{e}_l\}$ will also win, i.e., $\widetilde{x}_l = 1$. Hence, our auction is bid-monotonic. □

Lemma 2. The payments to winning bids are "critical". That is, for any bid $l \in \mathcal{I}$, compared to the received payment p_l when $x_l = 1$, bidding a lower price $\widetilde{e}_l \le p_l$ leads to winning the auction, i.e., $\widetilde{x}_l = 1$; bidding a higher price $\widetilde{e}_l > p_l$ leads to losing the auction, i.e., $\widetilde{x}_l = 0$.

Proof. According to Algorithm 1, we always have $\frac{e_l}{F_l} \le \frac{e_{l-}}{F_{l-}}$, because $\{\mathcal{F}_{l-}, e_{l-}\}$ is the first bid selected by the algorithm when $\{\mathcal{F}_{l^*}, e_{l^*}\}$ is excluded. Via setting $p_l = F_l \frac{e_{l-}}{F_{l-}}$, the algorithm guarantees $\frac{\widetilde{e}_l}{F_l} \le \frac{e_{l-}}{F_{l-}}$ when CP l bids a new price $\widetilde{e}_l \le p_l$, and guarantees $\frac{\widetilde{e}_l}{F_l} > \frac{e_{l-}}{F_{l-}}$ when $\widetilde{e}_l > p_l$. Therefore, the payment to every winner is at the critical value. □

Theorem 4. Individual Rationality. An auction is individually rational if every bid always has a non-negative utility, i.e., $u_l(b_l, \mathbf{b}_{-l}) \ge 0$, for any given $\mathbf{b}_{-l}$, $\forall l \in \mathcal{I}$. Algorithm 1 achieves individual rationality.

Proof. If CP l wins, we have $\frac{e_l}{F_l} \le \frac{e_{l-}}{F_{l-}}$, and $p_l = F_l \frac{e_{l-}}{F_{l-}} \ge e_l$. The utility of l is

$$u_l(b_l, \mathbf{b}_{-l}) = F_l \frac{e_{l-}}{F_{l-}} - vc_l = F_l \frac{e_{l-}}{F_{l-}} - e_l \ge 0,$$

where $vc_l = e_l$ is because of Theorem 3. If CP l loses, then we have $u_l(b_l, \mathbf{b}_{-l}) = 0$ by definition. Therefore, the utility of every bid is non-negative. □

IV. Long-Term Online Mechanism

In this section, we design an online algorithm to make the content placement decisions regarding cache locations while invoking our previous greedy-based primal-dual approximation algorithm to tie the series of single auctions altogether over time. We also rigorously prove the competitive ratio as the overall performance guarantee of our entire approach.

A. Lazy-Replacement-Based Online Algorithm

We can split the objective function of the single-shot problem $\mathbb{P}_0^t$ into two components: $C^t = C^t_{-RC} + C^t_{RC}$, where

$$\begin{aligned} C^t_{-RC}(\mathbf{x}^t, \mathbf{w}^t, \mathbf{z}^t, \mathbf{y}^t) = & \textstyle\sum_f \sum_n \sum_m b^t_{n,m} \lambda^t_{f,n} z^t_{f,n,m} \\ & + \textstyle\sum_f c^t_f w^t_f + \sum_i b^t_i x^t_i \\ & + \textstyle\sum_f \sum_n s^t_n y^t_{f,n} \end{aligned}$$

is the non-replacement cost and

$$C^t_{RC}(\mathbf{y}^t, \mathbf{y}^{t-1}) = \textstyle\sum_f \sum_n b^t_n [y^t_{f,n} - y^{t-1}_{f,n}]^+$$

Algorithm 2: Online Edge Caching Algorithm

```
1  Initialize $\hat{t} = 1$, $\widetilde{\mathbf{y}}^0$, $0 < \beta \le 1$;
2  Get the optimal solution $\widehat{\mathbf{y}}^1$ to $\mathbb{P}_3^1$;
3  Given $\widetilde{\mathbf{y}}^1 = \widehat{\mathbf{y}}^1$, get $\widetilde{\mathbf{x}}^1, \widetilde{\mathbf{w}}^1, \widetilde{\mathbf{z}}^1$ by Algorithm 1;
4  for $t = 2, 3, ..., T$ do
5      if $C^{\hat{t}}_{RC}(\widetilde{\mathbf{y}}^{\hat{t}}, \widetilde{\mathbf{y}}^{\hat{t}-1}) \le \beta \sum_{\tau=\hat{t}}^{t-1} C^{\tau}_{-RC}(\widetilde{\mathbf{x}}^t, \widetilde{\mathbf{w}}^t, \widetilde{\mathbf{z}}^t, \widetilde{\mathbf{y}}^t)$ or
        $\mathbb{P}_t^1$ or $\mathbb{P}_t^2$ infeasible given $\widetilde{\mathbf{y}}^{t-1}$ then
6          Get the optimal solution $\widehat{\mathbf{y}}^t$ to $\mathbb{P}_3^t$;
7          Given $\widetilde{\mathbf{y}}^t = \widehat{\mathbf{y}}^t$, get $\widetilde{\mathbf{x}}^t, \widetilde{\mathbf{w}}^t, \widetilde{\mathbf{z}}^t$ by Algorithm 1;
8          if $\widetilde{\mathbf{y}}^t \ne \widetilde{\mathbf{y}}^{t-1}$ then
9              $\hat{t} = t$;
10         end
11     end
12     if $\hat{t} < t$ then
13         $\widetilde{\mathbf{y}}^t = \widetilde{\mathbf{y}}^{t-1}$;
14         Given $\widetilde{\mathbf{y}}^t$, get $\widetilde{\mathbf{x}}^t, \widetilde{\mathbf{w}}^t, \widetilde{\mathbf{z}}^t$ by Algorithm 1;
15     end
16 end
```

is the replacement cost. Now, by C^t_{-RC} and the corresponding constraints, we define the following problem:

$$\begin{aligned} \mathbb{P}_3^t: \ \min \ & \textstyle\sum_f \sum_n s^t_n y^t_{f,n} \\ s.t. \ & \textstyle\sum_f y^t_{f,n} \le C_n, \forall n, \\ & \textstyle\sum_n y^t_{f,n} \ge \max_n \delta^t_{f,n}, \forall f, \\ & y^t_{f,n} \ge 0, \forall f, \forall n. \end{aligned}$$

We denote the optimal solution to $\mathbb{P}_3^t$ as $\widehat{\mathbf{y}}^t$. Note that even though we use a standard linear program solver to solve $\mathbb{P}_3^t$ in polynomial time, the optimal solutions $\widehat{y}^t_{f,n}, \forall f, \forall n$ we get are automatically integers in $\{0, 1\}$. This is because the coefficient matrix of the constraints in $\mathbb{P}_3^t$ is a "totally unimodular matrix" [23]. Also, according to Constraints (2c), (2d), and (2e), given $\widehat{\mathbf{y}}^t$, the other decision variables $\mathbf{x}^t$, $\mathbf{w}^t$, and $\mathbf{z}^t$ are all feasible.

Based on this, we design Algorithm 2, an online algorithm which balances C^t_{-RC} and C^t_{RC} dynamically in real time. Our key idea is to postpone changing content caching locations until the cumulative non-replacement cost times a pre-specified constant (i.e., β) exceeds the most recent replacement cost, or until the current cache locations lead to infeasible bid selection or request dispatching decisions (i.e., making $\mathbb{P}_1^t$ or $\mathbb{P}_2^t$ infeasible). This is actually Line 5 of the algorithm, where we denote the time slot of changing content caching locations as $\hat{t}$. If we need to change content caching locations, we solve $\mathbb{P}_3^t$ to get the new locations (Line 6) and given such new locations, we invoke Algorithm 1 to get all the other control decisions (Line 7) and update $\hat{t}$ if needed (Lines 8-10). Otherwise, if it turns out that we do not change the content caching locations (Line 12), we keep contents at current locations (Line 13) and given such locations, invoke Algorithm 1 to get all the other control decisions (Line 14).

B. Performance Analysis

We analyze the competitive ratio, defined as the maximum ratio of the objective function's value evaluated with the solutions produced by an online algorithm over the objective function's value evaluated with the offline optimum solutions, under all possible inputs. An online algorithm observes inputs that sequentially and dynamically arrive and makes irrevocable control decisions at each time slot on the fly, while the offline optimum observes the inputs over the entire time horizon at hindsight and solves the problem over time optimally at once.

Theorem 5. Competitive Ratio. The social cost achieved via Algorithm 2 is at most $\alpha \cdot (1+\max\{\beta, \max_t \gamma^t\}) \cdot \frac{\max_n s_n^t}{\min_n s_n^t} \cdot \max_t \left\{\frac{\sum_n C_n}{\max_n \sum_f \delta_{f,n}^t}\right\}$ times the offline optimal social cost, where α is defined in Theorem 2; β is the parameter introduced in Algorithm 2; and $\gamma^t = \frac{\sum_n b_n^t C_n}{\sum_f \sum_n \lambda_{f,n}^t \max_m b_{n,m}^t}$, $\forall t$.

Proof. We use $\widetilde{\mathbf{y}}^t$ to denote the value of $\mathbf{y}^t$ we obtain from Algorithm 2, and use $\widetilde{\mathbf{x}}^t$, $\widetilde{\mathbf{w}}^t$, and $\widetilde{\mathbf{z}}^t$ to denote the values of $\mathbf{x}^t$, $\mathbf{w}^t$, and $\mathbf{z}^t$, respectively, which we obtain from Algorithm 1 given $\widetilde{\mathbf{y}}^t$. We use $\{\widehat{\mathbf{x}}^t, \widehat{\mathbf{w}}^t, \widehat{\mathbf{z}}^t\}$ to denote the optimal solution to the problem of minimizing $g^t(\mathbf{x}^t, \mathbf{w}^t)+h^t(\mathbf{z}^t)$ at t given $\widetilde{\mathbf{y}}^t$. Further, we use $\{\mathbf{x}^{*t}, \mathbf{w}^{*t}, \mathbf{z}^{*t}, \mathbf{y}^{*t}\}$ to denote the joint optimal solution to the single-shot problem of minimizing C_{-RC}^t at t, and use $\left\{\{\bar{\mathbf{x}}^t, \bar{\mathbf{w}}^t, \bar{\mathbf{z}}^t, \bar{\mathbf{y}}^t\}, \forall t\right\}$ to denote the offline optimal solution to our social cost minimization problem.

First, due to Algorithm 2, we have

$$\begin{aligned}
&\textstyle\sum_{t=1}^T C^t(\widetilde{\mathbf{x}}^t, \widetilde{\mathbf{w}}^t, \widetilde{\mathbf{z}}^t, \widetilde{\mathbf{y}}^t) \\
=&\textstyle\sum_t \left(C_{-RC}^t(\widetilde{\mathbf{x}}^t, \widetilde{\mathbf{w}}^t, \widetilde{\mathbf{z}}^t, \widetilde{\mathbf{y}}^t) + C_{RC}^t(\widetilde{\mathbf{x}}^t, \widetilde{\mathbf{w}}^t, \widetilde{\mathbf{z}}^t, \widetilde{\mathbf{y}}^t)\right) \\
\leq&\textstyle\big(1+\max\{\beta, \max_t \gamma^t\}\big) \sum_t C_{-RC}^t(\widetilde{\mathbf{x}}^t, \widetilde{\mathbf{w}}^t, \widetilde{\mathbf{z}}^t, \widetilde{\mathbf{y}}^t). \quad (12a)
\end{aligned}$$

We explain (12a). Consider those time slots recorded by the variable $\hat{t}$ when executing Algorithm 2. Let us denote those time slots as $\{\hat{t}_1, \hat{t}_2, ...\}$. Consider any $k \geq 1$. For the consecutive time slots $\{\hat{t}_k, \hat{t}_k+1, ..., \hat{t}_{k+1}-1\}$, we either have

$$C_{RC}^{\hat{t}_k}(\widetilde{\mathbf{y}}^{\hat{t}_k}, \widetilde{\mathbf{y}}^{\hat{t}_{k-1}}) \leq \beta \cdot \textstyle\sum_{t=\hat{t}_k}^{\hat{t}_{k+1}-1} C_{-RC}^t(\widetilde{\mathbf{x}}^t, \widetilde{\mathbf{w}}^t, \widetilde{\mathbf{z}}^t, \widetilde{\mathbf{y}}^t)$$

or have

$$C_{RC}^{\hat{t}_k}(\widetilde{\mathbf{y}}^{\hat{t}_k}, \widetilde{\mathbf{y}}^{\hat{t}_{k-1}}) \leq \gamma^t \cdot C_{-RC}^{\hat{t}_k}(\widetilde{\mathbf{x}}^{\hat{t}_k}, \widetilde{\mathbf{w}}^{\hat{t}_k}, \widetilde{\mathbf{z}}^{\hat{t}_k}, \widetilde{\mathbf{y}}^{\hat{t}_k})$$

due to Line 5 of Algorithm 2. This is because, from the formulation of the original social cost minimization problem, we have $C_{RC}^t \leq \frac{\sum_n b_n^t C_n}{\sum_f \sum_n \lambda_{f,n}^t \max_m b_{n,m}^t} \cdot C_{-RC}^t = \gamma^t C_{-RC}^t$, $\forall t$. Summing up these inequalities for all k leads to (12a).

Then, recall $C_{-RC}^t(\mathbf{x}^t, \mathbf{w}^t, \mathbf{z}^t, \mathbf{y}^t) = g^t(\mathbf{x}^t, \mathbf{w}^t) + h^t(\mathbf{z}^t) + \sum_f \sum_n s_n^t y_{f,n}^t$. At each t, we have the following inequalities:

$$\begin{aligned}
&g^t(\widetilde{\mathbf{x}}^t, \widetilde{\mathbf{w}}^t) + h^t(\widetilde{\mathbf{z}}^t) + \textstyle\sum_f \sum_n s_n^t \widetilde{y}_{f,n}^t \\
\leq&\alpha\Big(g^t(\widehat{\mathbf{x}}^t, \widehat{\mathbf{w}}^t)+h^t(\widehat{\mathbf{z}}^t)+\textstyle\sum_f \sum_n s_n^t \widetilde{y}_{f,n}^t\Big) \quad (13a) \\
\leq&\alpha\Big(g^t(\mathbf{x}^{*t}, \mathbf{w}^{*t})+h^t(\mathbf{z}^{*t})+\textstyle\sum_f \sum_n s_n^t \widetilde{y}_{f,n}^t\Big) \quad (13b) \\
\leq&\alpha\Big(g^t(\mathbf{x}^{*t}, \mathbf{w}^{*t})+h^t(\mathbf{z}^{*t})+\max_n s_n^t \tfrac{\sum_n C_n}{\max_n \sum_f \delta_{f,n}^t} \textstyle\sum_{f,n} y_{f,n}^{*t}\Big) \\
&\qquad (13c) \\
\leq&\alpha \tfrac{\max_n s_n^t}{\min_n s_n^t} \tfrac{\sum_n C_n}{\max_n \sum_f \delta_{f,n}^t} \Big(g^t(\mathbf{x}^{*t}, \mathbf{w}^{*t})+h^t(\mathbf{z}^{*t})+\textstyle\sum_{f,n} s_n^t y_{f,n}^{*t}\Big) \\
&\qquad (13d) \\
\leq&\alpha \tfrac{\max_n s_n^t}{\min_n s_n^t} \tfrac{\sum_n C_n}{\max_n \sum_f \delta_{f,n}^t} \cdot C_{-RC}^t(\bar{\mathbf{x}}^t, \bar{\mathbf{w}}^t, \bar{\mathbf{z}}^t, \bar{\mathbf{y}}^t). \quad (13e)
\end{aligned}$$

(13a) follows from Algorithm 1 and also Theorem 2, where we have $g^t(\widetilde{\mathbf{x}}^t, \widetilde{\mathbf{w}}^t) \leq \alpha g^t(\widehat{\mathbf{x}}^t, \widehat{\mathbf{w}}^t)$, $\alpha \geq 1$, and $h^t(\widetilde{\mathbf{z}}^t) = h^t(\widehat{\mathbf{z}}^t)$. (13b) holds because $\{\widehat{\mathbf{x}}^t, \widehat{\mathbf{w}}^t, \widehat{\mathbf{z}}^t\}$ minimizes $g^t(\mathbf{x}^t, \mathbf{w}^t) + h^t(\mathbf{z}^t)$. (13c) is due to the following. By Constraints (2c), (2d), and (2e), we have $\max_n \sum_f \delta_{f,n}^t \leq \sum_f \sum_n y_{f,n}^t \leq \sum_n C_n$, and therefore have $\sum_f \sum_n \widetilde{y}_{f,n}^t \leq \frac{\sum_n C_n}{\max_n \sum_f \delta_{f,n}^t} \sum_f \sum_n y_{f,n}^{*t}$. (13d) constructs the term $\sum_f \sum_n s_n^t y_{f,n}^{*t}$. (13e) holds because $\{\mathbf{x}^{*t}, \mathbf{w}^{*t}, \mathbf{z}^{*t}, \mathbf{y}^{*t}\}$ minimizes C_{-RC}^t at t. □

V. Experimental Evaluations

A. *Experimental Settings*

Edges: We utilize the taxi data which contain over 10,000 records in 2013∼2014 regarding the trajectories of 442 taxis in Porto, Portugal [20]. We select the 25 most-visited locations in this dataset as "hot spots", and for each of such locations, we envisage a cellular base station (i.e., an edge) is deployed. The connections among these edges are generated as small-world networks [24]. We use this approach to simulate the edges because the true locations of real-world base stations are often proprietaries of mobile network carriers. We set the total length of the time horizon to 100 consecutive time slots, and obtain the content requests for each time slot (which can correspond to one hour in reality, for example) as follows.

Content Requests: We adopt the Youtube data [19]. We rank all the 1,500 contents in this dataset in terms of popularity, i.e., the accumulated number of user views. We only consider the 800 most popular videos in our evaluations. We get the average number of views of all the videos, and use it as the mean of a Poisson distribution. We further draw 25 random values from this distribution, and use these values as the corresponding total number of user requests received at the edges. At each edge, in proportion to the popularity, we then generate the number of user requests for each single video of the 800 videos out of the total number of views at this edge.

Costs: We set the per-content hosting cost at edge caches as within the range of $[0.1, 1]$, and set the sidehaul transference cost between two base stations in proportion to the corresponding geographic distance, within $[0.01, 0.1]$ which is the same order of magnitude as Amazon EC2 pricing [21]. We set the per-content downloading cost as being uniformly distributed in $[0.5, 5]$, considering the delay to remote servers is higher than the delay within edge networks. We generate the accessing cost for the ENO's own servers by scaling the average CP bidding price, described next. We note that we can associate proper non-negative weights to these different types of costs to indicate how the importance of each type of cost can impact the results. In reality, ENOs can tune such weights and run our approaches to seek trade-offs among different types of costs.

Bids: We consider 9 CPs. The bidding price of each CP's bid in each time slot is proportional to the sum of the view counts of all the contents (i.e., videos) contained in the bid [3]. To compose each CP's bid in each time slot, we randomly select at least half of all the contents under consideration.

Algorithms: We implement multiple algorithms in Python:

(i) *Greedy Caching with Auction (GCA)*: This approach places contents following the local demands at each base

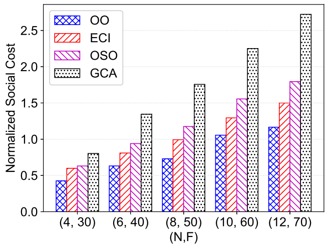

Fig. 4: Social cost

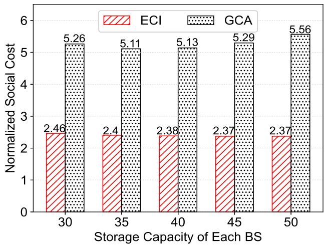

Fig. 5: Impact of cache cap

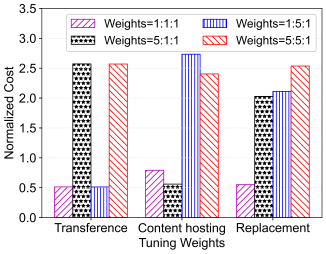

Fig. 6: Component cost

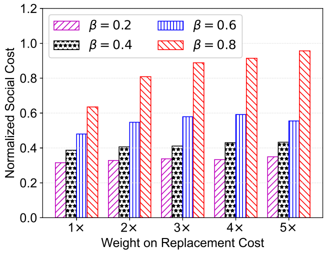

Fig. 7: Impact of laziness

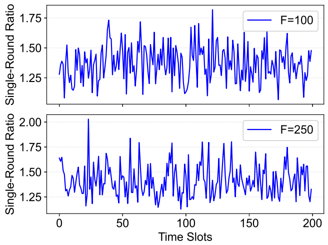

Fig. 8: Approximation ratio

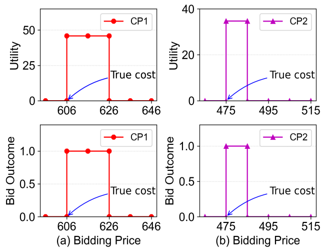

Fig. 9: Truthfulness

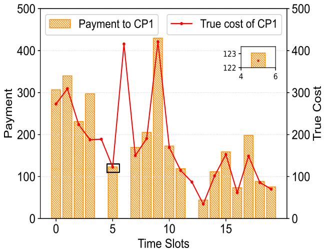

Fig. 10: Individual rationality

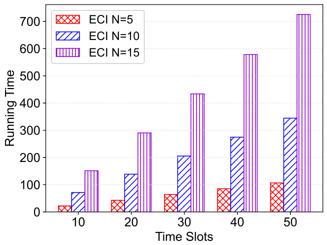

Fig. 11: Running time

station in each time slot [25], and then solves all the other control variables using our proposed action component.

(ii) *Online Single-shot Optimum (OSO)*: This approach ignores the downloading cost that couples adjacent time slots, and solves the problem (i.e., all the control variables) in each corresponding time slot optimally through invoking the Gurobi [26] optimization solver.

(iii) *Edge Caching Incentive (ECI)*: This is our holistic online approach as proposed in this paper.

(iv) *Offline Optimum (OO)*: This approach treats the long-term problem as an offline problem and solves it optimally over the entire time horizon all at once by Gurobi.

B. Experimental Results

Fig. 4 depicts the social cost over time, where N and F represent the number of edges and the number of contents considered in the system, respectively. In this figure, we focus on smaller-scale settings, as computing the offline optimum often takes an unacceptably long time even with the state-of-the-art optimization solver. Our approach ECI outperforms OSO and GCA by reducing $5\% \sim 45\%$ social cost, with the competitive ratio of $1.2 \sim 1.4$ regarding the offline optimum.

Fig. 5 exhibits how the social cost is impacted by edge cache capacity. In this figure, our approach ECI saves $53\% \sim 60\%$ social cost compared to GCA. Although capacity impacts content placements, ECI always balances the cost of downloading contents to new caches and the cost of maintaining contents at existing caches, incurring only slightly different social costs.

Fig. 6 visualizes the sidehaul transference cost, the content hosting cost, and the content downloading cost accumulated over time in our proposed approach, as the controlling weights for them vary. Increasing sidehaul transference cost incurs less content hosting cost while increasing the content hosting cost incurs slightly less sidehaul transference cost. The content downloading cost increases because, with the increase of the other two types of costs, it becomes easier to trigger the operation of downloads according to Algorithm 2.

Fig. 7 illustrates the impact of the control parameter β as in Algorithm 2 on the social cost. When the weight of the replacement (or downloading) cost is given, the increase of β enables the control condition in Algorithm 2 to be met more easily. Thus, more frequent content downloads from remote servers can happen and lead to excessive downloading cost (and thus excessive social cost). When the parameter β is given, a higher weight on the replacement cost may delay the necessary content downloading, resulting in an increase in the accumulated non-replacement cost (and thus in social cost).

Figs. 8, 9, and 10 verify the different properties provably achieved by our single-auction mechanism. Fig. 8 demonstrates that our approach actually achieves a very low approximation ratio of about less than 2 for any single time slot, where the number of contents in the system are set to 100 and 250, respectively. Fig. 9 verifies the truthfulness where CP1 and CP2 are two of the CPs in our evaluations selected randomly. The highest utilities are achieved as they bid their true costs (606 and 474, respectively). Since our payment depends on the second smallest cost density (the critical value) according to Algorithm 1, a bid would win if its bidding price is between the true cost and the second smallest value, and would lose otherwise. Fig. 10 verifies the individual rationality. When CP1 wins the auction, the payment is always no lower than the bidding price (assuming true cost) and the utility is non-negative; when it loses, the payment and the utility are zero.

Fig. 11 displays the execution time of our approach. As the length of the entire time horizon in terms of the total number of time slots increases, the total execution time of our approach grows moderately. This execution time is measured on a lab desktop, up to 725 seconds, and is much shorter than the time horizon under consideration, often hours to days in reality.

VI. Related Work

Edge Caching Algorithms: A substantial body of research have been done for edge caching algorithms. Xia et al. [1] developed a caching strategy to optimize edge transmit power and data retrieval rates. Asheralieva et al. [2] investigated content sharing in wireless CDNs with edge caching and device-to-device (D2D) communications. Abolhassani et al. [9] proposed freshness-driven content and update-rate selection algorithms for edge networks. Zhang et al. [10] balanced service delay and content freshness in mobile edge caching. Multiple works [11]–[14] exploited reinforcement learning and deep learning for edge caching to address privacy, spatial-temporal dynamics, popularity prediction, and D2D assistance.

These works focus on the various caching algorithms and optimizations for performance and cost from purely the system operator's or users' perspective. They do not typically study the interactions between different entities, i.e., the caching system and the content owners, thus inapplicable to our case.

Edge Caching Incentives: There are also some existing research on edge caching incentives. Wang et al. [15] set CPs' pricing to maximize utilities for edge contents by Stackelberg and stochastic games. Xiong et al. [16] studied the interplay among wireless network operators, CPs, and users as a three-stage non-cooperative game. Zhang et al. [8] cached contents among users with reduced redundancy via auctions, and decided payments via a bargaining game. Cao et al. [4] proposed to cache the most profitable contents by using auctions to allocate cache space and payments to maximize revenue. Ahmadi et al. [3] applied coalition games to design subsidization from access network operators to CPs. Liu et al. [17] investigated the scenario where the network operator leased small-cell resources to CPs via contract theory.

These works do not often focus on the specific interplay between the ENO and the CPs. The majority take a game-theoretic perspective, lacking auction-related economic properties. One other unique differentiator of our work is the auction intertwined with the switching-cost-aware online optimization, which never appeared in existing edge caching incentives.

VII. Conclusion

The economic interaction between the ENO and the CPs is the key to realizing commercial edge caching in reality, but has been largely ignored by existing research. In this paper, we fill this gap by devising an online auction-based mechanism which incentivizes CPs to cache contents continuously in distributed edge caches while enabling the ENO to determine bid selection, request dispatching, and cache updating to adapt to dynamic and uncertain environments. We rigorously prove multiple theoretical properties and guarantees, and conduct extensive evaluations using real-world data to validate the superior performance of our approach in practice. For future work, we intend to utilize the algorithmic techniques in this paper to study more online caching scenarios (e.g., caching machine learning models to serve users in edge networks).

Acknowledgment

This work is supported by the Ministry of Industry and Information Technology of China (2105-370171-07-02-860873), by the U.S. National Science Foundation (CNS-2047719), by the UK Engineering and Physical Sciences Research Council (EP/T01461X/1), and by the UK Alan Turing Pilot Project.

References

[1] X. Xia, F. Chen, Q. He, G. Cui, J. C. Grundy, M. Abdelrazek, X. Xu, and H. Jin, "Data, user and power allocations for caching in multi-access edge computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 5, pp. 1144–1155, 2022.

[2] A. Asheralieva and D. Niyato, "Combining contract theory and lyapunov optimization for content sharing with edge caching and device-to-device communications," *IEEE/ACM Transactions on Networking*, vol. 28, no. 3, pp. 1213–1226, 2020.

[3] M. Ahmadi, J. Roberts, E. Leonardi, and A. Movaghar, "Cache subsidies for an optimal memory for bandwidth tradeoff in the access network," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 4, pp. 736–749, 2020.

[4] X. Cao, J. Zhang, and H. V. Poor, "An optimal auction mechanism for mobile edge caching," in *IEEE ICDCS*, 2018.

[5] "Google CDN," https://cloud.google.com/cdn.

[6] "Netflix Open Connect," https://openconnect.netflix.com.

[7] "China Mobile launches new media arm to provide content," https://www.chinadaily.com.cn/business/2015-01/16/content_19333325.htm.

[8] T. Zhang, X. Fang, Y. Liu, G. Y. Li, and W. Xu, "D2d-enabled mobile user edge caching: A multi-winner auction approach," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 12, pp. 12 314–12 328, 2019.

[9] B. Abolhassani, J. Tadrous, and A. Eryilmaz, "Single vs distributed edge caching for dynamic content," *IEEE/ACM Transactions on Networking*, vol. 30, no. 2, pp. 669–682, 2022.

[10] S. Zhang, L. Wang, H. Luo, X. Ma, and S. Zhou, "Aoi-delay tradeoff in mobile edge caching with freshness-aware content refreshing," *IEEE Transactions on Wireless Communications*, vol. 20, no. 8, pp. 5329–5342, 2021.

[11] S. Liu, C. Zheng, Y. Huang, and T. Q. Quek, "Distributed reinforcement learning for privacy-preserving dynamic edge caching," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 3, pp. 749–760, 2022.

[12] X. Zhang, G. Zheng, S. Lambotharan, M. R. Nakhai, and K.-K. Wong, "A reinforcement learning-based user-assisted caching strategy for dynamic content library in small cell networks," *IEEE Transactions on Communications*, vol. 68, no. 6, pp. 3627–3639, 2020.

[13] Y. Jiang, H. Feng, F.-C. Zheng, D. Niyato, and X. You, "Deep learning-based edge caching in fog radio access networks," *IEEE Transactions on Wireless Communications*, vol. 19, no. 12, pp. 8442–8454, 2020.

[14] X. Wang, R. Li, C. Wang, X. Li, T. Taleb, and V. C. Leung, "Attention-weighted federated deep reinforcement learning for device-to-device assisted heterogeneous collaborative edge caching," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 1, pp. 154–169, 2020.

[15] Q. Wang, S. Guo, J. Liu, C. Pan, and L. Yang, "Motishare: Incentive mechanisms for content providers in heterogeneous time-varying edge content market," *IEEE Transactions on Services Computing*, 2021.

[16] Z. Xiong, S. Feng, D. Niyato, P. Wang, A. Leshem, and Z. Han, "Joint sponsored and edge caching content service market: A game-theoretic approach," *IEEE Transactions on Wireless Communications*, vol. 18, no. 2, pp. 1166–1181, 2019.

[17] T. Liu, J. Li, F. Shu, H. Guan, Y. Wu, and Z. Han, "Incentive mechanism design for two-layer wireless edge caching networks using contract theory," *IEEE Transactions on Services Computing*, vol. 14, no. 5, pp. 1426–1438, 2018.

[18] A. Archer and E. Tardos, "Truthful mechanisms for one-parameter agents," in *IEEE FOCS*, 2001.

[19] "Youtube request trace data," http://netsg.cs.sfu.ca/youtubedata.

[20] "Porto taxi trajectory," https://www.kaggle.com/crailtap/taxi-trajectory.

[21] "Amazon ec2," https://aws.amazon.com/cn/ec2/pricing/.

[22] "Nr and ng-ran overall description, 2021," https://www.3gpp.org.

[23] B. H. Korte, J. Vygen, B. Korte, and J. Vygen, *Combinatorial Optimization*. Springer, 2011.

[24] R. H. S. Duncan J. Watts, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, pp. 440–442, 1998.

[25] V. G. S. Borst and A. Walid, "Distributed caching algorithms for content distribution networks," in *IEEE INFOCOM*, 2010.

[26] "GUROBI OPTIMIZER," https://www.gurobi.com/.