# Incentivizing Microservices for Online Resource Sharing in Edge Clouds

Amit Samanta[1] Lei Jiao[2] Max Mühlhäuser[1] Lin Wang[1,3]
[1]TU Darmstadt, Germany [2]University of Oregon, USA [3]VU Amsterdam, The Netherlands

***Abstract*—The microservice architecture provides high agility, making it a suitable choice for implementing edge cloud services. Provisioning microservices at the network edge requires the dynamic allocation of resources. However, due to the resource limitation in the edge cloud environment, there is no guarantee that enough resources are always available upon a microservice's requests. In this paper, we design an online auction-based mechanism to incentivize microservices to spare their occupied resources so that the edge cloud platform can reclaim them and reallocate them to other microservices that need resources. We firstly design a single-stage auction that determines the winning bids to satisfy the resource demands in polynomial time, while calculating the payments. Then, we design an online framework to tie a series of such single-stage auctions into a multi-stage online mechanism without requiring the knowledge of future bids and demands. Via rigorous analysis, we exhibit that our mechanism design achieves truthful bidding and individual rationality, with a constant competitive ratio regarding the social cost of the system in the long run. Finally, we verify the practical performance of our mechanism through extensive simulations.**

## I. Introduction

As an emerging evolution of cloud computing, edge computing provides small-scale clouds or server clusters at the network edge, with ultra-low access latency to end users [1]. The services provisioned at such edge clouds can often be architected as sets of "microservices" [2], which are loosely coupled, independently deployable, and highly maintainable, as implemented in several commercial products such as IBM Watson IoT and Amazon Greengrass. In a typical Function-as-a-Service (FaaS) scenario, multiple different tenants (i.e., service providers) deploy different microservices colocated at the edge cloud to serve their end users, and pay to the edge cloud operator for the resources they actually use.

One fundamental problem in the FaaS-based edge clouds is the resource allocation for the microservices. The resource allocation here differentiates itself from that in the conventional, gigantic clouds by the fact that the resources that can be allocated are usually very limited, due to the small capacities of the edge clouds. Therefore, there is no guarantee that the resource requests of a microservice, which may be due to a scaling-up operation to accommodate more end-user requests for example, can always be satisfied, especially when the requested resources are currently used by the microservices of other tenants. This can be a significant problem for the edge cloud operator, as failing to meet the resource demands may result in tenant dissatisfaction and eventually revenue loss.

The challenge of resource limitation motivates the introduction of appropriate mechanisms that can incentivize the microservices to yield their occupied resources proactively, so that the edge cloud operator can reallocate such resources to those microservices that need them, as illustrated in Figure 1. One approach to achieve this may be "pricing", i.e., letting the edge cloud operator repurchase those resources from the microservices at fixed or flat prices. However, how to set such price is often tricky for the edge cloud operator, and it may end up with an inefficient trial-and-error process; also, it is easy to under-price or over-price when the demand-supply relation among the microservices varies, which could hurt the edge cloud operator's revenue and the system-wide social welfare. We consider an alternative approach using "auction". In contrast to pricing, the auction mechanism enables the market efficiency and agility by resource pricing based on the real-time supply-demand directly; it also reduces the chance of mis-pricing, and better matches resources with the microservices that value or need them most, thus increasing the resource utilization efficiency and the edge cloud operator's profit.

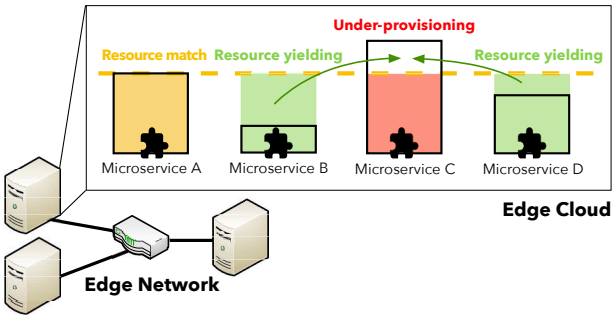



Figure 1: Resource reclaiming and reallocation in edge cloud.

In this paper, we design a dynamic, online auction-based mechanism to enable microservices to spare their resources in the form of bids so that the edge cloud operator can buy the bids to reclaim the resources. Our mechanism features multiple specific designs for the edge cloud environment: (1) we capture the heterogeneity of the microservices, i.e., at any given time instant, different microservices may be willing to yield different amounts of resources at different bidding prices; (2) we capture the limited resources that the microservices may be willing to share in the long run, i.e., the participation in the auction mechanism may be restricted over time; (3) we take into account the dynamic uncertainty in the resource requirements, i.e., the resource demands from those microservices that need resources may vary arbitrarily as time elapses. Our mechanism has four neat properties: (1) it is truthful, in the sense that only bidding the true cost of the

resources can each microservice achieve its maximum utility; (2) it is individual rational with non-negative utility, in the sense that the payment each winner microservice receives from the edge cloud operator always covers the cost of the bid; (3) it is computationally efficient, as it executes in polynomial time; (4) it is online and has a guaranteed competitive ratio, i.e, the long-term total social cost of the edge cloud operator and the microservices incurred by the online auction mechanism that only knows the dynamic resource demands on the fly, is always provably upper-bounded by a constant times of the corresponding offline optimum that knows all the system dynamics at once in advance. All these properties are the keys to ensure the economic efficiency and the successful operation of our mechanism for microservices.

While there exist substantial previous works on resource management for edge computing [3]–[10], they do not consider a market perspective, but rather a "managed" environment where the edge cloud platform controls everything, including both resource allocation and revocation. Meanwhile, there are auction-based mechanisms designed for cloud and edge [11]–[17]; however, they never study the resource sharing incentive mechanisms for edge clouds with limited resources. To the best of our knowledge, we are the first to study microservice resource reclaiming in edge clouds from an economic perspective. We summarize our contributions in this paper as follows:

• We build a demand estimation model to characterize the resource requirements of the microservices that need resources by primarily considering three indicators for serving end users' requests: the request waiting time, the request execution time, and the request processing rate.

• We design a single-stage auction mechanism incorporating our demand estimation model. We formulate an Integer Linear Program (ILP) to determine the winning bids and minimize the "social" cost, i.e., the total cost of both the edge cloud operator and the microservices that make bids, while satisfying the resource demands in the current stage. We propose a primal-dual-based approximation algorithm to solve this NP-hard ILP in polynomial time while calculating the payments.

• We design an online framework that ties a series of single-stage auctions into an online multi-stage auction mechanism. Our mechanism strategically protects a microservice's potential of its future participation in the auctions by avoiding depleting its resource offers too early, which is in turn achieved by using a dual variable for each microservice to augment its cost based on the remaining level of its resource offers.

• We formally analyze and prove the performance and properties of both our single-stage and multi-stage mechanisms, including NP-hardness, feasibility, truthfulness, individual rationality, and competitive ratio. We also evaluate the practical performance of the two mechanisms through extensive simulations with real-world data traces and parameter settings.

## II. System Settings

We consider an edge computing system comprised of a set of edge clouds $\mathcal{L}$, a set of microservices $\mathcal{S}$, and a set of end users $\mathcal{R}$ generating application-specific requests. Each edge cloud generally hosts some resources used by the microservices to process user requests received. The edge clouds are connected to each other through a backhaul network and every edge cloud is reachable from every network access point. We consider a time-slotted system where time is divided into time-slots and a time-slot $T$ is further divided into several rounds, i.e., $T = \{1, 2, \cdots, t\}$.

Let $\mathcal{S} = \{1, 2, \cdots, n\}$ denote the set of microservices, where $i$ denotes the $i^{th}$ microservice; $\mathcal{R} = \{1, 2, \cdots, m\}$ denotes the set of users, where $f$ is the $f^{th}$ user; $\mathcal{L} = \{1, 2, \cdots, l\}$ denotes the set of edge clouds, where $l$ denotes the $l^{th}$ edge cloud. Each user issues requests, which are processed in one of the edge clouds through a specific microservice. Each microservice requires a certain amount of resources from the edge cloud to process the requests received. Suppose, a microservice $i$ is getting $a_i^t$ amount of resources from the edge cloud $l$ following a *fair sharing* policy to process their available requests at time $t$, but sometimes the resources may not be sufficient to fulfill the actual resource demand $X_i^t$ of the microservice, where $X_i^t > a_i^t$. This is because the edge clouds are generally resource-constrained and have limited resources to share. Therefore, it is not always possible to fulfill the dynamic resource requirements of each microservice. Thus, the microservices in need may ask for a help from other coexisting microservices to share their extra resources within the same edge cloud. To this end, the microservices that may be willing to help need some incentives in order to share their resources with those microservices in need. Hence, the edge cloud platform circulates the financial payment to microservices with extra resources in each round. However, it is very tough to estimate the actual resource demand of microservices under different network dynamics, such as, traffic load, application types, waiting time, execution time etc. In order to overcome such issues, we have designed a microservice demand estimation scheme in edge clouds to quantify the actual resource demand of the microservice. Here, we consider that $\bar{\mathcal{S}}$ denotes a subset of microservices, which actually requires extra resources from the edge cloud (where $\bar{\mathcal{S}} \subset \mathcal{S}$). The auction mechanism comprised of three steps – (a) online demand estimation, (b) winner selection problem among microservices, and (c) online algorithm designs. Each microservice $i$ can participate in the incentive mechanism process at most once in a particular round and share its resources to other microservices.

## III. Microservice Demand Estimation

To estimate the actual resource demand of microservices, here we present a microservice demand estimation scheme. At each round, each edge cloud chooses a set of microservices and reports its demand requests to the edge platform, while having the minimum waiting time. Therefore, the edge platform is aware of the complete progress of all microservices at the end of each round. To estimate the demand, we formalize a demand indicator function to characterize the actual demand request of microservices. Let $\mathbb{X} = \{X_1^t, X_2^t, \cdots, X_n^t\}$ denote

a set of actual resource demands of all microservices at $t^{th}$ round, where $X_i^t$ denotes the demand of microservice $i$ at $t^{th}$ round. We propose to characterize the actual resource demand of a microservice by three main factors, i.e., the request waiting time, the request processing time of a microservice and the request rate. Intuitively, the smaller the waiting time, the larger the demand; the smaller the processing time, the larger the demand; the higher the request rate, the larger the demand. Thus, we have used these three factors to determine the demand for a microservice $i$. Mathematically,

$$X_i^t = \frac{1}{w_\gamma}\gamma_i^t + \frac{1}{w_\mathbb{R}}\mathbb{R}_i^t + \frac{1}{w_\mathbb{T}}\mathbb{T}_i^t \tag{1}$$

where $\gamma_i^t$, $\mathbb{R}_i^t$ and $\mathbb{T}_t^t$ represent the waiting time, execution time, and request rate of microservice $i$ at time $t$, respectively. $\frac{1}{w_\gamma}$, $\frac{1}{w_\mathbb{R}}$ and $\frac{1}{w_\mathbb{T}}$ are scaling factors to represent the relative importance of these three factors. In our scheme, the rewards are given to microservices according to their actual demands. Therefore, higher demand means higher the reward. However, the actual value of a demand at time $t$ actually does not have too much interpretation, but instead, the demands of all microservices at time $t-1, t-2, \cdots$ are more important in order to design a fair demand estimation scheme. This will assist our scheme to use accurate and appropriate rewards to balance the universality of microservices. The factors used in designing the demand estimation scheme are given as:

**Waiting time.** The waiting time is defined as the time each microservice needs to wait for processing their requests. The waiting time is a factor that can affect the demand of a microservice, which is defined as $\theta_i/\pi_i$, where $\theta_i$ is the number of served responses and $\pi_i$ is the received number of responses for microservice $i$. The larger the response time, the smaller demand will be required. Moreover, the larger the response time, the faster the reduction rate of demand will be required. Therefore, we have, $\gamma_i^t = \zeta\theta_i/\pi_i$. Here, $\zeta$ is a coefficient that scales the value of demand affected by the waiting time. We can see that the demand decreases as the waiting time increases and is lower bounded by 0. Furthermore, the reduction rate of demand $\gamma_i^t$ increases as the completing progress approaches to 1.

**Processing time.** In order to complete a request within some expected time, we need the microservice processing rate to be larger than or equal to the expected rate, $\varsigma_i - \varpi_i \geq 0$. We relax this constraint with its long term time-average. Mathematically, we have, $\mathbb{R}_i^t = \frac{\varsigma_i - \varpi_i}{t}$. It indicates that each microservice that needs the processing rate $\varsigma_i$ is on average $\varpi_i$ larger than $\varsigma_i$ in order for the request to complete within the expected time.

**Request rate.** The request rate is defined as the number of incoming requests from each user. Microservices with higher request rate would have higher resource demands to process all the requests from the users. Thus, we have

$$\mathbb{T}_i^t = \Delta\frac{a_i^t}{a_{max}}\frac{\mathbb{L}_i^t \times t}{\mathcal{V}(\bar{n})}\frac{1}{1-\mathbb{L}_i^t} \tag{2}$$

where $\Delta$ is a coefficient that scales the value of the demand

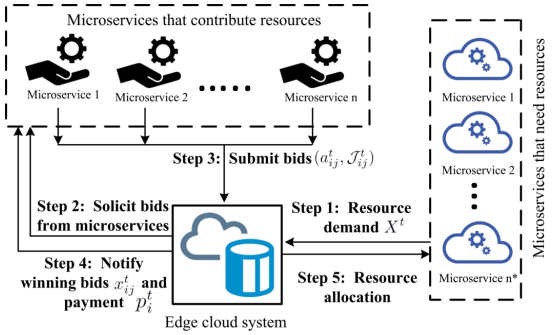



Figure 2: Online auction design for microservices.

affected by the request rate. $a_i^t$ denotes the amount of resources received from an edge cloud to process a request with microservice $i$ at time $t$ and $a_{max} = \max(a_i^t)$ is the maximum resource demand among all microservices. We can see that the less request rate, then less demand is required. $\mathcal{V}(\bar{n})$ denotes the density of neighbouring microservices served and $\mathbb{L}_{it}$ denotes the execution rate. By combining all these factors, we have

$$X_i^t = \sum_{i\in\hat{\mathcal{S}}}\frac{1}{w_\gamma}\zeta\theta_i/\pi_i + \frac{1}{w_\mathbb{R}}\frac{\varsigma_i - \varpi_i}{t} + \frac{1}{w_\mathbb{T}}\frac{\Delta a_i^t}{a_{max}}\frac{\mathbb{L}_i^t \times t}{\mathcal{V}(\bar{n})}\frac{1}{1-\mathbb{L}_i^t}.$$

The scaling factors are fixed contants and note that these scaling factors can be decided by the analytical hierarchy process (AHP) [18].

## IV. Online Mechanism Design

Next, we design an offline single-stage auction mechanism for resource sharing among microservices, taking into consideration of actual resource requirements of microservices. Then, we design an online algorithmic framework, while utilizing a single-stage auction mechanism in each round. In an online mechanism, we consider that the microservice requests arrive one by one in an arbitrary order, and their demand and pricing policies are drawn independently.

When a microservice request $i \in \mathcal{S}$ arrives at the edge platform, the platform estimates the resource demand of that microservice. The platform also determines the resource sharing price $p_i^t$ among microservices using the auction theory [11], [12], [19]. After observing the demand factor $X_i^t$, microservice $i$ reports $g_i \in \{0, 1\}$ to the platform, where $g_i$ denotes whether microservice $i$ accepts the price for resource sharing ($g_i = 1$ for acceptance). The mechanism adds $i$ to the winner set $\bar{\mathcal{S}}$ if and only if $g_i = 1$. This process continues until either the total budget $\mathcal{W}$ is depleted or the last microservice has been processed. However, achieving optimal profit is difficult in an online scenario than when offline, as we have to guarantee that the platform achieves optimality any time a microservice request arrives.

### A. Preliminaries for Truthful Mechanisms

The scheme is designed in multi-round manner over total $T$ rounds. A microservice $i$ needs a set of cooperative microservices $\widetilde{\mathcal{S}}$ to achieve desired resources in the resource sharing

process at round $t \in [T]$. At the beginning of round $t$, the edge platform circulates all the available resources to microservices present in the edge cloud following a fair sharing policy. If no microservices require any extra resources in round $t$, then $\bar{\mathcal{S}}$ is set to empty. Note that microservice requests may arrive in any round. Suppose, a request of microservice $i \in \mathcal{S}$ arrives at the $t_i^-$ round. First, it submits an integer pair $(t_i^+, X_i^t)$, where $X_i^t$ is the microservice $i$'s resource requirement, i.e., the maximum amount of resources required by microservice $i$ to process its request during the time period from $t_i^-$ to $t_i^+$. Without loss of generality, we assume that each microservice's resource demand $X_i^t$ can change at each time period and the values of $X_i^t$ and $t_i^+$ depend on a number of factors, e.g., resource demand and processing time. Thereafter, microservice $i$ may submit up to $F$ alternative bids at the beginning of round $t$ within $[t_i^-, t_i^+]$, each of which is a microservices-bid pair $(\hat{\mathcal{S}}, \mathcal{J}_{ij}^t)$: $\hat{\mathcal{S}}$ is a set of microservices willing to participate in the resource sharing mechanism at round $t$ and $\mathcal{J}_{ij}^t$ is the bidding price that microservice $i$ charges for sharing its resources.

In each round $t$, after receiving microservices' bids, the edge platform broadcasts the auction result. We consider a binary variable $x_{ij}^t$. If the edge platform considers microservice $i$'s bid $j$ in round $t$ then $x_{ij}^t$ equals to 1, otherwise 0. The edge platform also calculates the remuneration $p_i^t$ for a winner $i$ and pays $p_i^t$ to the winner $i$ after checking whether the desired microservices' resource demands can be satisfied. Let $G_{ij}^t$ denote the actual resource price for microservice $i$ to process its requests, the utility of that bid with bidding price $\mathcal{J}_{ij}^t$ is:

$$U_{ij}^t(\mathcal{J}_{ij}^t) = \begin{cases} p_i^t - G_{ij}^t, & \text{if } x_{ij}^t = 1, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

The strategical behavior of all the microservices is assumed to be rational with a common motive of maximizing their respective utilities. We instead value the rationality of the entire edge computing system and follow up the minimum social cost possible. In order to do so, the microservices are expected to provide truthful bids. To describe our auction mechanism, we introduce a few definitions in the following.

**Definition 1.** *A microservice strategy is dominant if and only if the strategy of a microservice can achieve a higher utility value than any other microservice strategies.*

**Definition 2.** *A microservice is called individually rational if the microservice always obtains a nonnegative utility value. Mathematically,*

$$U_{ij}^t(\mathcal{J}_{ij}^t) \geq 0. \tag{4}$$

**Definition 3.** *The microservice incentivization scheme is truthful in bidding price if, for any microservice $i$, bidding its actual price $G_{ij}^t$ forms its dominant strategy. We have*

$$U_{ij}^t(G_{ij}^t) \geq U_{ij}^t(\mathcal{J}_{ij}^t), \forall \mathcal{J}_{ij}^t \neq G_{ij}^t. \tag{5}$$

**Definition 4.** *The social welfare in the online auction mechanism is the aggregate utility of the edge platform $-\sum_{t\in T}\sum_{i\in\hat{\mathcal{S}}}^{t} x_{ij}^t$ and of all the microservices $\sum_{t\in T}\sum_{i\in\hat{\mathcal{S}}}(p_i^t - G_{ij}^t)x_{ij}^t$. The social welfare is $-\sum_{t\in T}\sum_{i\in\mathcal{S}} G_{ij}^t x_{ij}^t$ when payments cancel themselves. Maximizing the social welfare is equivalent to minimizing the social cost $\sum_{t\in T}\sum_{i\in\hat{\mathcal{S}}}\sum_{j\in J} G_{ij}^t x_{ij}^t$.*

**Definition 5.** *There should not be any economic loss in the designed auction mechanism, i.e., the expense that each auctioneer charges all its winning buyers can afford the total payment that the sellers receive. Formally, we have*

$$\sum_{i\in\widetilde{\mathcal{S}}} x_{ij}^t X_i^t \geq \sum_{i\in\bar{\mathcal{S}}} x_{ij}^t X_i^t. \tag{6}$$

**Definition 6.** *The competitive ratio is defined as the upper-bound ratio of the social cost generated by the designed online auction to the social cost produced by an optimal solution of the offline winner selection problem in (7).*

### *B. Optimization Problem Formulation*

By applying the truthful bidding, the winner selection problem for social cost minimization can be modeled by the following Integer Linear Program (ILP).

$$\min \quad \sum_{t\in T}\sum_{i\in\hat{\mathcal{S}}}\sum_{j\in J} \mathcal{J}_{ij}^t x_{ij}^t \tag{7}$$

$$\text{s.t.} \quad x_{ij}^t \in \{0,1\}, \forall i \in \hat{\mathcal{S}}, \forall j \in J, \forall t \in T \tag{8}$$

$$\sum_{j\in J} x_{ij}^t \leq 1, \forall i \in \hat{\mathcal{S}}, \forall t \in T \tag{9}$$

$$\sum_{j\in J, i\in\hat{\mathcal{S}}} a_{ij}^t x_{ij}^t \geq X^t, \forall t \in T \tag{10}$$

$$\sum_{j\in J, t\in T} x_{ij}^t \leq \sigma_i, \forall i \in \hat{\mathcal{S}} \tag{11}$$

Constraint (9) specifies that each microservice wins at most one bid in each round. Constraint (10) guarantees that each microservice $i \in \hat{\mathcal{S}}$ is covered by at least $|\mathcal{S}|$ participants in round $t$ for resource sharing and getting the required amount of resources $X^t$. Here, $a_{ij}^t$ denotes the amount of resources to be shared by microservice $i$ in bid $j$ at time $t$. Each microservices's resource capacity constraint is modeled in (11), which practically limits microservice $i$'s participation.

### *C. Single-Stage Auction Design*

We first formulate the single-stage winner selection problem (WSP). Later, we will introduce our single-stage auction mechanism design, based on which we design our final online algorithm. The single-stage winner selection problem is designed including the same constraints associated with the current time-slot from (7). We reform the price function $J_{ij}^t$ to a scaled price $\nabla_{ij}^t$ according to the level of remaining resource capacity and incorporate it in this objective function. We remove the constraints (11); they are contemplated in the design of scaled price (discussed in the online algorithm part).

$$\min \quad \sum_{t\in T}\sum_{i\in\hat{\mathcal{S}}}\sum_{j\in J} \nabla_{ij}^t x_{ij}^t \tag{12}$$

$$\text{s.t.} \sum_{j\in J, i\in\hat{\mathcal{S}}} a_{ij}^t x_{ij}^t \geq X^t, \forall t \in T \tag{13}$$

$$\sum_{j\in J} x_{ij}^t \leq 1, \forall i \in \hat{\mathcal{S}}, \forall t \in T \tag{14}$$

$$x_{ij}^t \in \{0,1\}, \forall i \in \hat{\mathcal{S}}, \forall j \in J, \forall t \in T \tag{15}$$

By relaxing the binary constraints of $x_{ij}^t$ into $0 \leq x_{ij}^t \leq 1, i \in \hat{\mathcal{S}}, j \in J$,, the dual of the LP relaxation can be formulated as follows. Here, $g_{ij}^t$, $h_{ij}^t$ and $\beta_i^t$ are the same dual variables as in the dual of (7), corresponding to constraints (13), $x_{ij}^t \leq 1$ and (14), respectively.

$$\max \quad \sum_{t\in T}\sum_{i\in\hat{\mathcal{S}}}\sum_{j\in J}\left(X^t g_{ij}^t - h_{ij}^t - \beta_i^t\right) \tag{16}$$

$$\text{s.t.} \sum_{t\in T, i\in\hat{\mathcal{S}}, j\in J} g_{ij}^t - h_{ij}^t - \beta_i^t \leq \nabla_{ij}^t \tag{17}$$

$$g_{ij}^t, h_{ij}^t, \beta_i^t \geq 0, \forall i \in \hat{\mathcal{S}}, \forall j \in J, t \in T \tag{18}$$

We consider a special case of the primal problem in (12) by setting $\mathcal{S} = 1, \forall \sum_{i\in\hat{\mathcal{S}}}$ and remove the constraint (14).

**Theorem 1.** *The winner selection problem (WSP) among microservices is NP-hard.*

*Proof.* The designed problem can be reduced from the minimum weighted set cover problem that is known to be NP-hard [20]. Because the minimum weighted set cover problem is a special case of winner selection problem (12), hence winner selection problem in Equation (12) is also NP-hard. □

We first briefly introduce the main idea of our approximation algorithm. We consider that a microservice $i$ is active in round $t$ until it reaches its resource requirement $X_i^t$. The algorithm iteratively selects winning bids based on a greedy strategy. In each iteration, a bid is picked to cover active microservices at the least average price. We next introduce some variables. Denote by $\mathbb{E}^t = \{(i_1, j_1), (i_2, j_2), \cdots\}$ a subset of bids submitted at round $t$, e.g., $(i_1, j_1)$ represents bid $j_1$ submitted by microservice $i_1$. Let $\theta_i(\mathbb{E}^t) = \sum_{(i,j)\in\mathbb{E}^t: i\in\hat{\mathcal{S}}} i$ be the total number of microservices $\hat{\mathcal{S}}$ participating in resource sharing mechanism among all bids in $\mathbb{E}^t$. The utility of set $\mathbb{E}^t$ is defined as $U(\mathbb{E}^t) = \sum_{i\in\hat{\mathcal{S}}} \min(\theta_i(\mathbb{E}^t), X^t)$, representing the valid contribution of bids in $\mathbb{E}^t$ to all microservices $\hat{\mathcal{S}}$. The minimum contribution of microservice $i$'s bid $j$ is the utility increase from adding $i$ into $\mathbb{E}^t$, given by

$$U_{ij}(\mathbb{E}^t) = \sum_{i\in\hat{\mathcal{S}}}\sum_{j\in J}\left(U(\mathbb{E}^t \cup (i,j)) - U(\mathbb{E}^t)\right) = \tag{19}$$
$$\big(\min(\theta_i(\mathbb{E}^t \cup (i,j)), X^t) - \min(\theta_i(\mathbb{E}^t), X^t)\big).$$

We present a single-stage auction mechanism – SSAM in Algorithm 1. SSAM selects one winning bid per iteration and appends it to the set of winning bids $\mathbb{E}^t$. Let $M_{ij}^t$ be a set that tracks current active microservice $i$'s bid $j$. Further, $U_{ij}(\mathbb{E}^t)$ also denotes the number of active microservices in that bid. To analyze the approximation ratio of SSAM, which is the upper-bound ratio of the social cost generated by SSAM to the social cost produced by an optimal solution of winner selection problem in Equation (12), we look at the dual problem in Equation (16). According to the weak duality theorem, the bound between the primal and dual objective values also bounds the approximation ratio.

**Algorithm 1:** SSAM: Single-Stage Auction Mechanism

**Input:** $(\nabla_{ij}^t, \hat{\mathcal{S}})$, $\mathbb{G}^t$, $\mathbb{F}^t$, $\mathbb{H}^t$, $\forall i, j, k$.
**Output:** $(\mathbb{E}^t)$.

1 Initialize $x_{ij}^t = 0$, $p_i^t = 0$, $g_{ij}^t = 0$, $h_{ij}^t = 0$, $\beta_i^t = 0$, $\forall i \in \hat{\mathcal{S}}, j \in \mathcal{S}, k \in K$;
2 Define $\mathbb{E}^t = \emptyset$;
3 **while** $U_{ij}(\mathbb{E}^t) < \sum_{i\in\hat{\mathcal{S}}} X^t$ **do**
4 $(i', j') = \arg\min_{(i,j)\in\mathbb{F}^t} \nabla_{ij}^t / U_{ij}(\mathbb{E}^t)$;
5 $x_{i'j'}^t = 1$; $f(i, \hat{\mathcal{S}}) = \nabla_{i'j'}^t / U_{i'j'}(\mathbb{E}^t)$;
6 $(i^-, j^-) = \arg\min_{(i,j)\in\mathbb{F}^t:(i,j)\neq(i',j')} \nabla_{ij}^t / U_{ij}(\mathbb{E}^t)$;
7 $p_{i'}^t = U_{i'j'}(\mathbb{E}^t) \nabla_{i^-j^-}^t / U_{i^-j^-}(\mathbb{E}^t)$;
8 $(i^+, j^+) = \arg\min_{(i,j)\in\mathbb{F}^t} \nabla_{ij}^t / U_{ij}(\mathbb{E}^t)$;
9 $f(i, \hat{\mathcal{S}})' = \nabla_{i^+j^+}^t / U_{i^+j^+}(\mathbb{E}^t)$;
10 $\mathbb{F}^t = \mathbb{F}^t \backslash (\bigcup_{j\in J}(i', j'))$;
11 $\mathbb{E}^t = \mathbb{E}^t \bigcup (i', j')$;
12 $\mathbb{H}^t = \mathbb{H}^t \backslash (\bigcup (i', j'))$;
13 $\max(i) = \max_{(\mathcal{S}_{ij}^t)}\{\{f(i, \hat{\mathcal{S}})\} \bigcup \{f(i, \mathcal{S})'\}\}, \forall i \in \hat{\mathcal{S}}$;
14 $\min(i) = \min_{(\mathcal{S}_{ij}^t)}\{\{f(i, \hat{\mathcal{S}})\} \bigcup \{f(i, \mathcal{S})'\}\}, \forall i \in \hat{\mathcal{S}}$;
15 $\Xi_i = \max(i)/\min(i), \forall i \in \hat{\mathcal{S}}; \Xi = \max_{i\in\hat{\mathcal{S}}} \Xi_i$;
16 $\Lambda(k) = \max_{(\mathcal{S}_{ij}^t)}\{\{f(i, \hat{\mathcal{S}})\}\}, \beta_i^t = \Lambda(k) / (\sum_{i=1}^{\hat{\mathcal{S}}} \frac{1}{i} \Xi)$;
17 **for** $x_{ij}^t == 1$ **do**
18 $h_{ij}^t = \sum_{i\in\hat{\mathcal{S}}} (\Lambda(i) - f(i, \hat{\mathcal{S}})) / (\sum_{i=1}^{\hat{\mathcal{S}}} \frac{1}{i} \Xi_i)$ ;
19 Accept microservice $i$'s bid $j$;
20 Pay $p_{i'}^t$ to microservice $i$;
21 **Return** $\mathbb{E}^t$;

In SSAM, $\mathbb{H}^t$ is a grand set of all bids in round $t$, $\mathbb{G}^t$ is the grand set of actual resource requirements of all microservices, and $\mathbb{F}^t$ is the microservice set of all valid bids, i.e., bids that satisfy capacity constraints (11). Here, $\mathbb{G}^t$ is a set of $X^t$, i.e., $\mathbb{G}^t = \{X_1^t, X_2^t, \cdots\}$. Line 1 initializes all primal and dual variables to zero. Line 2 initializes the set of winning bids as $\emptyset$. The `While` loop in lines 3-12 considers a greedy approach to select winning bids, calculates payments, and updates primal variables. Specifically, a winning bid with minimum average price is determined in line 4, with the corresponding primal variable $x_{i'j'}^t$ set to 1 in line 5. When a microservice set $\hat{\mathcal{S}}$ is accepted, its price $\nabla_{ij}^t$ is assigned to pairs $(i, \hat{\mathcal{S}})$ for each active microservice $i$, i.e., for each such pair, a price for microservice $i$: $f(i, \hat{\mathcal{S}}) = \nabla_{i'j'}^t / U_{i'j'}(\mathbb{E}^t)$ is defined. Lines 6-7 compute the payment to winner $i'$ based on the critical value rule [21], providing that if the winner announces a smaller price, it should win (see Lemma 3). Line 8 decides the winning bid if we relax the constraints (9) and (11). Further, another pricing factor $f(i, \hat{\mathcal{S}})'$ is considered in line 9 to assist

computing the dual variables. Other bids from microservice $i'$ are removed from active set $\mathbb{F}^t$. Line 11 appends the winning bid to set $\mathbb{E}^t$ and removes it from the bid set $\mathbb{H}^t$. We update dual variables based on the price of resource sharing for microservices to bound the approximation ratio. Lines 12-15 compute the values of dual variables $\beta_i^t$. The `for` loop in lines 17-20 assign values to dual variables $h_{ij}^t$, which correspond to winning bids, and announces auction result.

### *D. Theoretical Analysis*

We analyze the performance of SSA presented in Algorithm 1, in terms of correctness, complexity, approximation ratio, truthfulness, and individual rationality.

**Lemma 1 (Dual Feasibility).** *The SSAM algorithm produces a feasible solution to the dual problem of (12).*

*Proof.* **Case 1:** First, we assume that microservice $i$'s bid $j$ is not picked by SSAM. Let us sort the microservices in $\hat{\mathcal{S}}$ by the reverse order in which their resource requirement $\mathbb{G}^t$ is satisfied. When the microserice $i$'s participation requirement is satisfied (where $i \in \mathcal{S} - k$), $\hat{\mathcal{S}}$ has at least $k$ unsatisfied microservices. Therefore, it can satisfy the active microservices at a price no larger than $\nabla_{ij}^t/k$. This allows the maximal and minimal prices of $\hat{\mathcal{S}}$, i.e., $\max(k)$ and $\min(k)$, when we consider all bids and ignore constraints (14) and (11). When $i$'s requirement is satisfied (where $i \in \mathcal{S} - k$), its price $\Lambda(k)$ should be at most $(\nabla_{ij}^t/k \times \max(k)/\min(k))$. Also, we consider $\mathbb{W}_k = \sum_{k=1}^{\hat{\mathcal{S}}} \frac{1}{k}$

$$\sum_{k\in\hat{\mathcal{S}}:i\in\hat{\mathcal{S}}-k} g_k^t - h_{ij}^t - \beta_i^t = \sum_{k\in\hat{\mathcal{S}}:i\in\hat{\mathcal{S}}-k} g_k^t$$

$$= \frac{1}{\mathbb{W}_k\Xi} \sum_{k\in K:i\in\mathcal{S}-k} \Lambda(k) \leq \frac{\nabla_{ij}^t}{\mathbb{W}_k\Xi} \sum_{k=1}^{|\hat{\mathcal{S}}|} 1/k \frac{\max(k)}{\min(k)}$$

$$\leq \frac{\nabla_{ij}^t}{\mathbb{W}_k\Xi} \mathbb{W}_{|\hat{\mathcal{S}}|}\Xi \leq \frac{\nabla_{ij}^t}{\mathbb{W}_k\Xi} \mathbb{W}_k\Xi = \nabla_{ij}^t.$$

Hence, constraint (17) holds, if the microservice $i$'s bid $j$ fails.

**Case 2:** Further, we assume that the microservice $i$'s bid $j$ wins in the single-stage auction mechanism – SSAM. Then,

$$\sum_{k\in\hat{\mathcal{S}}:i\in\hat{\mathcal{S}}} g_k^t - h_{ij}^t - \beta_i^t = \sum_{k\in\hat{\mathcal{S}}:i\in\hat{\mathcal{S}}} g_k^t - h_{ij}^t$$

$$= \frac{1}{\mathbb{W}_k\Xi}\Bigg( \sum_{k\in\hat{\mathcal{S}}\setminus\bar{\mathcal{S}}} \Lambda(k) + \sum_{k\in\bar{\mathcal{S}}} f(k,\hat{\mathcal{S}}) \Bigg)$$

$$= \frac{1}{\mathbb{W}_k\Xi}\Bigg( \sum_{k\in\hat{\mathcal{S}}\setminus\bar{\mathcal{S}}} \Lambda(k) + \nabla_{ij}^t \Bigg).$$

Now, we first order microservices in $\bar{\mathcal{S}}$, and sort the remaining microservices in $\hat{\mathcal{S}}\setminus\bar{\mathcal{S}}$ in the reverse order of their participation requirement satisfaction. Then $\forall k \in \hat{\mathcal{S}}\setminus\bar{\mathcal{S}}$, when $i$'s requirement is satisfied, its price $\Lambda(k)$ should be at most $(\nabla_{ij}^t/(k + |\bar{\mathcal{S}}|) \times \max(k)/\min(k))$. Therefore, the right hand side of constraint (17) is bounded by $\nabla_{ij}^t$. □

**Theorem 2 (Complexity).** *The SSAM Algorithm provides a feasible solution to ILP (12) and (16) in polynomial time.*

*Proof.* **Polynomial time:** First, line 1 initializes all the variables within $O(nm)$ steps in linear time. The `while` loop runs at most $n$ times since it selects one winner during each iteration and there are at most $n$ microservices. The steps within the while loop (lines 4-12) take $O(nm)$ steps to determine the winner, calculate the payments, and also update all the variables. Hence, the computation complexity of the while loop is $O(n^2m)$. The values of $\max(i)$, $\min(i)$, $\Xi_i$, $\Lambda(i)$, and $\beta_i^t$ can be computed in $O(nm)$ steps (lines 13-16). The for loop (lines 17-20) iterates $nm$ times to update the dual variable $h_{ij}^t$, and its running time is $O(nmT)$. The last step in line 21 takes one step to return the winner set. Thus, the running time of SSAM in Algorithm 1 is $O(n^2m)$. For $l$ number of edge clouds, the running time of SSAM in Algorithm 1 is $O(n^2ml)$.

**Dual feasibility:** Lemma 1 proves that SSAM returns a feasible solution to linear program (16).

**Primal feasibility:** If the single-stage winner selection problem in Equation (12) is solvable, there exists at least a feasible solution by selecting one bid per microservice. Hence, Algorithm 1 terminates either before or when the candidate set $\mathbb{F}^t$ becomes empty. When it terminates $U(\mathbb{E}^t) \geq \sum_{i\in\hat{\mathcal{S}}} X^t$ and $\theta_i(\mathbb{E}^t) \geq X^t$ satisfying constraint (13). Constraint (14) holds true as line 4 and line 10 guarantee each microservice can have at most one bid accepted. Constraints (15) will not be violated as values $x_{ij}^t$ are initialized to 0 in line 1 and updated to 1 only in line 5. □

**Theorem 3.** *Let $\omega$ and $\psi$ be the primary objective value in (12) and the dual objective value in (16) returned by SSAM, respectively. Then, SSAM guarantees $\omega/\psi = \pi$ with $\pi = \mathbb{W}_i\Xi$, where the approximation ratio of SSAM is $\pi$.*

*Proof.* After the termination of SSAM, the objective value of the dual problem (16) is:

$$\psi = \frac{1}{\mathbb{W}_i\Xi} \sum_{i\in\hat{\mathcal{S}}} X^t\Lambda(i) - \frac{1}{\mathbb{W}_i\Xi} \sum_{i,j} \sum_{i\in\hat{\mathcal{S}}} \Big( \Lambda(i) - f(i,\hat{\mathcal{S}}) \Big)$$

$$= \frac{1}{\mathbb{W}_i\Xi} \sum_{i,j} \sum_{i} f(i,\hat{\mathcal{S}}), \tag{20}$$

where $\mathbb{W}_i = \sum_{i=1}^{\hat{\mathcal{S}}} \frac{1}{i}$. The second equality holds because for each microservice $i$, there are $X^t$ price variables $f(i,\hat{\mathcal{S}})$ corresponding to it. Furthermore, $f(i,\hat{\mathcal{S}})$ is assigned a value only when $i$ belongs to set $\hat{\mathcal{S}}$. Consequently, $\sum_{i,j}\sum_{i\in\hat{\mathcal{S}}} \big(\Lambda(i) - f(i,\hat{\mathcal{S}})\big)$ can be rewritten as $\sum_{i\in\hat{\mathcal{S}}} X^t\Lambda(i) - \sum_{i,j}\sum_{i\in\hat{\mathcal{S}}} \big(\Lambda(i) - f(i,\hat{\mathcal{S}})\big)$. Then, the objective value of the primal problem (12) is:

$$\omega = \sum_{(i,j)\in\mathbb{E}^t} \nabla_{ij}^t = \sum_{i,j} f(i,\hat{\mathcal{S}}). \tag{21}$$

The above equality follows because when bid $(\hat{\mathcal{S}}, \nabla_{ij}^t)$ is selected by SSAM, $\nabla_{ij}^t$ is distributed over variables $f(i,\hat{\mathcal{S}})$ of the microservices that are in $\hat{\mathcal{S}}$ and are still active.

Therefore, we have $\mathbb{W}_i \Xi \psi = \omega$. Let $\omega^*$ be the optimal objective value of ILP (12). By the weak duality theorem, $\omega^* \geq \psi$, then $\omega/\omega^* \leq \omega/\psi = \mathbb{W}_i \Xi = \pi$. Furthermore, if each microservice submits only one bid at each round, we can obtain $\mathbb{W}_k \psi = \omega$ and the approximation ratio is $\mathbb{W}_i$ under this typical scenario. □

**Lemma 2.** *The bidding process in SSAM is monotonic in nature, i.e., $\forall i \in \hat{\mathcal{S}}, \forall j, j' \in J$, if $\nabla_{ij'}^t < \nabla_{ij}^t$ and $\mathcal{S}_{ij'}^t = \mathcal{S}_{ij}^t$, $x_{ij}^t = 1$ implies $x_{ij'}^t = 1$.*

*Proof.* Suppose microservice $i$'s bid $j$ wins, i.e., $x_{ij}^t = 1$, then this bid has the minimum $\nabla_{ij}^t / U_{ij}(\mathbb{E}^t)$ in the current iteration. If microservice $i$ reports a smaller price $\nabla_{ij'}^t (which is smaller than \nabla_{ij}^t)$ to cover the same set of microservices, as $U_{ij'}(\mathbb{E}^t) = U_{ij}(\mathbb{E}^t)$ according to the definition in 6, $\nabla_{ij}^t / U_{ij'}(\mathbb{E}^t) < \nabla_{ij}^t / U_{ij}(\mathbb{E}^t)$ implies that bid $(\nabla_{ij'}^t, \mathcal{S}_{ij'}^t)$ wins in or even before the current iteration by our greedy algorithm in SSAM. Thus, the proof concludes. □

**Lemma 3.** *The payments to all winners estimated by the SSAM algorithm are critical under the following criteria: assume a winning bid, e.g., microservice $i'$'s bid $j$, reports a new price $\bar{\nabla}_{i'j'}^t$ instead of $\nabla_{i'j'}^t$; microservice $i'$'s bid $j$ will win if $\bar{\nabla}_{i'j'}^t < p_{i'}$, and will fail otherwise.*

*Proof.* By following SSAM, microservice $i^\circ$'s bid $j^\circ$ is the threshold bid for it since when we exclude $(i', j')$ from the candidate set, microservice $i^\circ$'s bid $j^\circ$ is the first bid that is accepted by SSAM. Intuitively, microservice $i'$'s bid $j'$ will win if $\bar{\nabla}_{i'j'}^t / U_{i'j'}(\mathbb{E}^t) \leq \nabla_{i^\circ j^\circ}^t / U_{i^\circ j^\circ}(\mathbb{E}^t)$ and will fail otherwise. By setting $p_{i't} = U_{i'j'}(\mathbb{E}^t).\nabla_{i^\circ j^\circ}^t / U_{i^\circ j^\circ}(\mathbb{E}^t)$, SSAM guarantees that $\bar{\nabla}_{i'j'}^t / U_{i'j'}(\mathbb{E}^t) \leq \nabla_{i^\circ j^\circ}^t / U_{i^\circ j^\circ}(\mathbb{E}^t)$ when $\bar{\nabla}_{i'j'}^t \leq p_{i'}$ and $\bar{\nabla}_{i'j'}^t / U_{i'j'}(\mathbb{E}^t) > \nabla_{i^\circ j^\circ}^t / U_{i^\circ j^\circ}(\mathbb{E}^t)$ when $\bar{\nabla}_{i'j'}^t > p_{i'}$. Thus, we conclude that each winner is paid with a critical value. □

**Theorem 4.** *The SSAM algorithm is truthful in bidding price.*

*Proof.* The Myerson's theorem [21] indicates that a reverse auction is truthful in bidding price if and only if 1) the auction result $(x_{ij}^t)$ is monotonically non-decreasing with the decrease of the reported price $\nabla_{ij}^t$ and 2) each winner is paid with a critical value. Hence, combining Lemma 2 and Lemma 3, we finish the proof. □

**Theorem 5.** *The SSAM algorithm achieves individual rationality.*

*Proof.* As discussed in Lemma 3, bid $(\nabla_{i^\circ j^\circ}^t, \mathcal{S}_{i^\circ j^\circ}^t)$ is the threshold bid for winning bid $(\nabla_{i'j'}^t, \mathcal{S}_{i'j'}^t)$, and $\nabla_{i'j'}^t / U_{i'j'}(\mathbb{E}^t) \leq \nabla_{i^\circ j^\circ}^t / U_{i^\circ j^\circ}(\mathbb{E}^t)$. It is clear that the payment $p_{i't}$ to winner $i'$ is at least $\nabla_{i'j'}^t$, as $p_{i't} = U_{i'j'}(\mathbb{E}^t) \nabla_{i^\circ j^\circ}^t / U_{i^\circ j^\circ}(\mathbb{E}^t)$. Further, when $\nabla_{i'j'}^t \geq J_{i'j'}^t$, the utility of microservice $i'$'s bid $j'$ with bidding price $J_{i'j'}^t$ is $U_{i'j'}^t(J_{i'j'}^t) = p_{it} - G_{i'j'}^t = p_{it} - J_{i'j'}^t \geq 0$ as Theorem 4 guarantees $G_{i'j'}^t = J_{i'j'}^t$. Therefore, the microservices always obtain non-negative utility. □

**Algorithm 2:** MSOA: Multi-Stage Online Auction

**Input:** $(J_{ij}^t, \mathcal{S}_{ij}^t)$, $\mathbb{G}^t$, $\Theta_i$, $t_{i-}$, $t_{i+}$, $\forall i, j, k, t$.
**Output:** $(\mathbb{E}^t), \forall t$.

1 Initialize $\psi_i^t = 0$, $\chi_i = 0$, $\forall i \in \mathcal{S}, t \in T$;
2 Define $\mathbb{E}^t = \cup_{i \in \mathcal{S}, j \in J}(i, j)$, $\mathbb{F}^t = \mathbb{E}^t$, $\forall t \in T$;
3 **for** $\forall 1 \leq t \leq T$ **do**
4  **for** $\forall i \in \mathcal{I}, j \in J$ **do**
5  **if** $t \in [t_{i-}, t_{i+}]$ *and* $\chi_i + |\mathcal{S}_{ij}^t| \Theta_i >$ **then**
6  $\nabla_{ij}^t = J_{ij}^t$; $\mathbb{F}^t = \mathbb{F}^t \backslash (i, j)$;
7  **else**
8  $\nabla_{ij}^t = J_{ij}^t + |\mathcal{S}_{ij}^t| \psi_i^{t-1}$;
9  $\mathbb{E}^t = SSAM\big((\nabla_{ij}^t, \mathcal{S}), \mathcal{H}_k^t, \mathbb{F}^t, \mathbb{H}^t, \forall i, j, k\big)$;
10  **for** $\forall (i, j) \in \mathbb{E}^t$ **do**
11  $\psi_i^t = \psi_i^{t-1}(1 + \frac{|\mathcal{S}_{ij}^t|}{\pi \Theta_i}) + \frac{J_{ij}^t |\mathcal{S}_{ij}^t|}{\pi \Theta_i^2}$;
 $\chi_i = \chi_i + |\mathcal{S}_{ij}^t|$;
12  **for** $\forall (i, j) \notin \mathbb{E}^t$ **do**
13  $\psi_i^t = \psi_i^{t-1}$;
14 $\psi_i = \psi_i^t, \forall i \in \mathcal{S}$;
15 **Return** $\mathbb{E}^t, \forall t \in T$;

### *E. Multi-Stage Online Mechanism Design*

We next design the online auction mechanism for resource sharing and participation incentivization, based on an online auction framework that decomposes the online winner selection problem (WSP) into a series of single-stage WSPs.

The key challenge of online auction lies in microservices' resource constraints over their demands: different overall social costs can be produced when the resources are well spent at different rounds. If a platform's bid with a large set of microservices is accepted at an early stage, that microservice may lose opportunities to participate later in the edge computing auction. The platform may be consequently forced to purchase resources from more expensive alternatives, leading to a higher social cost. The optimal strategy intends to let a microservice $i$ participate in all rounds between round $t^-$ and $t^+$ such that the platform can make the best decision among all microservices.

We present our online auction framework MSOA in Algorithm 2. Here, we introduce a new variable $\psi_i^t$ for each microservice in line 1. It increases with the decrease of a microservice's remaining resource requirement. $\chi_i$ tracks the number of satisfied microservices before the present round and is initialized to zero in line 1. We also initialize a grand set $\mathbb{E}^t$ that includes all bids defined in line 2 and a candidate set $\mathbb{F}^t$ that contains all valid bids for the current round initialized to $\mathbb{E}^t$ (satisfying all the constraints). In each round, the new scaled price $\nabla_{ij}^t$ will be used in the single-stage auction (SSAM). If the number of microservices in the current bid plus the number of previously performed microservices exceed the resource capacity (line 5), $\nabla_{ij}^t$ remains the same but that bid is excluded from the candidate set (line 6); $\nabla_{ij}^t$ equals

$J_{ij}^t + |\mathcal{S}_{ij}^t|\psi_i^{t-1}$ (line 8) otherwise. Following this, a bid with a smaller remaining resource requirement will be assigned a higher price, reducing its chance to win. $\psi_i^t$ is updated carefully for each winning bid in the current round (lines 11-13), and remains unchanged otherwise (lines 10-13). We set the dual variable $\psi_i = \psi_i^t$ in line 14. The adjustment of $\psi_i^t$ at each round is the increment of $\psi_i$ at each round, which will be used to bound the competitive ratio. Note that MSOA takes the single-stage auction SSAM in each round $t$, and winners are paid during the execution of SSAM (see line 17 of Algorithm 1).

**Theorem 6.** *MSOA computes a feasible solution for ILP (12) and its dual (16) in polynomial time.*

*Proof.* **Polynomial time:** Lines $1-2$ define three new variables and initialize one dual variable in $O(nml)$ steps. The outer `for` loop iterates $T$ times to determine winners at each round. The first inner `for` loop calculates the scaled cost in $O(IJ)$ steps. Then SSAM is executed in line 10 to select winners, and its running time is $O(n^2ml)$ according to Theorem 2. The second and third inner `for` loops can be executed in $O(n)$ steps to update dual variable $\psi_i^t$. Therefore, the running time of the outer `for` loop is $O(n^2mlT)$. The last line takes $O(nT)$ steps to assign values to $\psi_i$ and returns the winner sets. Therefore, the running time of MSOA in Algorithm 2 is $O(n^2mlT)$.

**Primal feasibility:** The constraints (8), (9) and (11) can be guaranteed by constraints in (12). Constraint (10) holds if the number of microservices in the current bid exceeds the microservice's remaining capacity. That bid is excluded from the microservice set $\mathbb{F}^t$ by MSOA (lines $5-6$) and is never accepted by the platform.

**Dual feasibility:** Since MSAO provides a feasible solution to liner program (16), constraint (17) holds. Based on the assignment of $\nabla_{ij}^t = J_{ij}^t + |\mathcal{S}_{ij}^t|\psi_i^{t-1}$, we have

$$\sum_{t\in T, i\in\hat{\mathcal{S}}, j\in J} g_{ij}^t - h_{ij}^t - \beta_i^t < \nabla_{ij}^t = J_{ij}^t + |\mathcal{S}_{ij}^t|\psi_i^{t-1} < J_{ij}^t + |\mathcal{S}_{ij}^t|\psi_i^T = J_{ij}^t + |\mathcal{S}_{ij}^t|\psi_i. \quad (22)$$

If $\nabla_{ij}^t = J_{ij}^t$, we get,

$$\sum_{t\in T, i\in\hat{\mathcal{S}}, j\in J} g_{ij}^t - h_{ij}^t - \beta_i^t < \nabla_{ij}^t = J_{ij}^t < J_{ij}^t + |\mathcal{S}_{ij}^t|\psi_i. \quad (23)$$

This satisfies constraint (17). Because $\psi_i > 0$, constraint (18) also holds true. □

We analyze the competitive ratio of MSOA. Let $\Delta\mu^t$ and $\Delta\Im^t$ be the primal and dual objective values in (7) and (16) returned by MSOA after $t$ rounds. Then $\Delta\mu^t = \mu^t - \mu^{t-1}$ and $\Delta\Im^t = \Im^t - \Im^{t-1}$ are the final primal and dual objective values achieved by MSOA. Let $\mu^*$ denote the optimal objective value of linear integer program (7).

**Lemma 4.** *Let $\Delta\mu^t$ and $\Delta\Im^t$ be the increase of the primal and dual objective values after iteration t, respectively, i.e., $\Delta\mu^t = \mu^t - \mu^{t-1}$ and $\Delta\Im^t = \Im^t - \Im^{t-1}$. For all $t \in T$, $\Delta\mu^t < \alpha\frac{\beta}{\beta-1}\Delta\Im^t$, where $\beta = min_{i\in\mathcal{S}, j\in J, t\in T}\frac{\Theta_i}{|\mathcal{S}_{ij}^t|}$.*

*Proof.* $\bar{\mu}$ and $\bar{\Im}$ are the primal objective value in (12) and the dual objective value in (16) returned by MSOA, respectively. Theorem 3 proves that $\alpha\bar{\Im} > \bar{\mu}$. At time $t$, $\Delta\mu^t = \sum_{(i,j)\in\mathbb{E}^t} J_{ij}^t$.

$$\Delta\Im^t = \sum_{i\in\mathbb{E}^t}\Theta_i(\psi_i^{t-1} - \psi_i^t) + \bar{\Im} \quad (24)$$

$$= \bar{\Im} - \frac{\sum_{(i,j)\in\mathbb{E}^t}|\mathcal{S}_{ij}^t|\psi_i^{t-1}}{\alpha} - \sum_{i\in\mathbb{E}^t}\frac{J_{ij}^t}{\alpha}\frac{|\mathcal{S}_{ij}^t|}{\Theta_i} \quad (25)$$

$$> \frac{\bar{\mu}}{\alpha} - \frac{\sum_{(i,j)\in\mathbb{E}^t}|\mathcal{S}_{ij}^t|\psi_i^{t-1}}{\alpha} - \frac{\sum_{i\in\mathbb{E}^t}J_{ij}^t}{\alpha\beta} \quad (26)$$

$$> \frac{\sum_{(i,j)\in\mathbb{E}^t}\nabla_{ij}^t}{\alpha} - \frac{\sum_{(i,j)\in\mathbb{E}^t}|\mathcal{S}_{ij}^t|\psi_i^{t-1}}{\alpha} - \frac{\sum_{i\in\mathbb{E}^t}J_{ij}^t}{\alpha\beta}$$

$$= (\frac{1}{\alpha} - \frac{1}{\alpha\beta})\sum_{\sum_{(i,j)\in\mathbb{E}^t}J_{ij}^t} = (\frac{1}{\alpha} - \frac{1}{\alpha\beta})\Delta\mu^t \quad (27)$$

□

**Theorem 7.** *MSOA is $\frac{\alpha\beta}{\beta-1}$-competitive in social cost.*

*Proof.* As $\mu^{(0)} = \Im^{(0)} = 0$ and $\Delta\mu^t < \frac{\alpha\beta}{\beta-1}\Delta\Im$, we can obtain $\mu^t = \sum_{t=0}^T \mu^t - \mu^{t-1} < \frac{\alpha\beta}{\beta-1}\sum_{t=0}^T \Im^t - \Im^{t-1} = \frac{\alpha\beta}{\beta-1}\Im^t$. By the weak duality theorem, $\mu^* > \Im^t$, thus $\frac{\mu}{\mu^*} < \frac{\mu^t}{\Im^t} < \frac{\alpha\beta}{\beta-1}$. Therefore, the competitive ratio of MSOA is $\frac{\alpha\beta}{\beta-1}$. □

**Theorem 8.** *MSOA is a truthful and individually rational online auction that solves the WDP (1) in polynomial time and is $\frac{\alpha\beta}{\beta-1}$-competitive in social cost.*

*Proof.* **Truthfulness in bidding:** MSOA is truthful in bidding price (see Theorem 4) running at each round with an input of the scaled cost. The scaled cost is based on the bidding price and is known only to the platform. Therefore, MSOA with SSAM ensures truthfulness in bidding price.

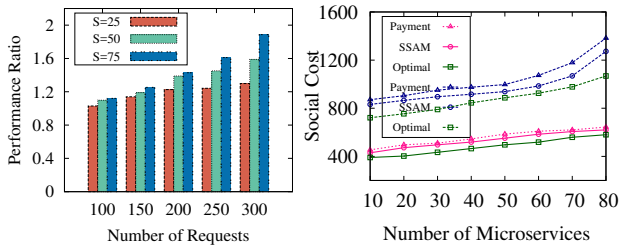
**Truthfulness in microservice set:** We need to show that a microservice will never bid for resources that are not within its resource demand. As indicated by line 18 in MSOA, if a winner cannot get its required resources, it will not receive any payment. Thus, microservices have no incentives to lie about the microservice set. □

## V. Performance Evaluation

We evaluate the performance of our single-stage and multi-stage online auction mechanisms in terms of performance ratio, price distribution, and percentage of winning tasks.

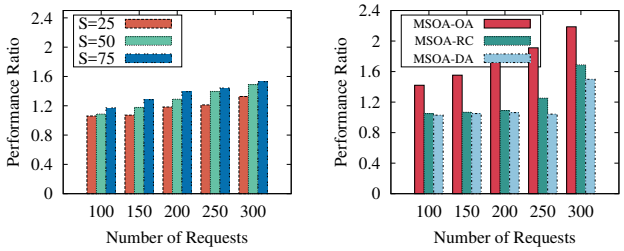
### A. Parameter Settings

We consider 300 edge users and 10 macro base stations each co-located with a computing server (considered as an edge cloud) [22]. We randomly deploy 25-75 microservices on different edge clouds. The prices of bids are uniformly



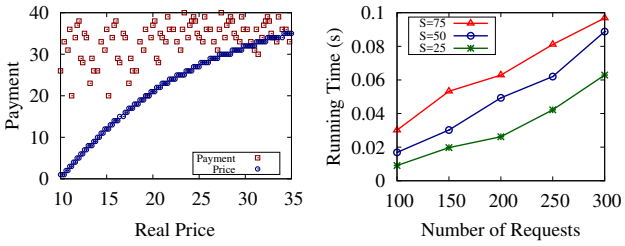

(a) Performance Ratio (b) Social Cost

Figure 3: Performance ratio and social cost for SSAM.



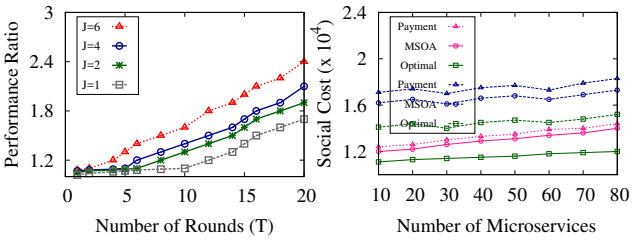

(a) Performance Ratio (b) Performance Ratio

Figure 5: Analysis of performance ratio for MSOA.




(a) Payment vs. actual price (b) Running Time

Figure 4: Payment vs. actual price, and running time for SSAM.




(a) Performance Ratio (b) Social Cost

Figure 6: Performance ratio under different bids and social cost.

distributed in the range of $[10, 35]$ and the value of $\mathbb{G}^t$ is set within the range of $[10, 40]$. We pick microservices randomly within the edge clouds to form the microservice set $\mathcal{S}$. For the single-stage auction, we iterate the SSAM algorithm for 10 minutes to get the average results. For the online auction, we vary the number of rounds $T$ from 1 to 15 and assume the length of each round is 10 minutes. $t_i^-$ and $t_i^+$ are set randomly within $[1, T]$. The default value for $T$, $\mathcal{S}$, $J$, and $\mathcal{L}$ is 10, 25, 2, and 10, respectively. For this work, we consider two types of microservices – delay-sensitive and delay-tolerant. To generate the workloads, we use the Poisson process with mean 5 and 10 for delay-sensitive and delay-tolerant microservice requests, respectively. Here, the higher priority is given to delay-sensitive microservices. To further understand the impact of MSOA on different performance metrics, we compare three different MSOA-based solutions: MSOA-DA (with optimal demand estimation scheme), MSOA-RC (with higher resource capacity values), and MSOA-OA (with both the demand and resource capacity constraints optimized).

### B. Results and Discussions

**Offline Algorithm (SSAM).** First, we examine the performance of our single-stage offline auction mechanism – SSAM, with respect to performance ratio. The performance ratio is defined as the ratio of the objective value of ILP in (12) produced by the Algorithm 1 to the optimal objective value of (12). Figure 3(a) shows the change of the ratio of SSAM with varying numbers of microservices $\mathcal{S}$. We can observe that with an increase in the number of microservices, the performance ratio also increases simultaneously. SSAM performs better with smaller $\mathcal{S}$, achieving a close-to-optimal ratio ($\approx 1$) in the presence of a small set of microservices and each microservice submits only one bid. The assessment is consistent with the theoretical analysis in Theorem 3 where it is shown that the performance ratio is upper bounded by the value of $\mathbb{W}_i\Xi$. The value of $\mathbb{W}_i$ increases if $\mathcal{S}$ becomes bigger and the value of $\Xi$ becomes larger when a microservice submits more bids.

We generate the price of bids following a uniform distribution and plot the social cost, payment, and optimal price returned by SSAM algorithm under different numbers of microservices in Figure 3(b). With the increase in the number of microservices, the edge platform must satisfy more requests, incurring a higher social cost. The social cost where the number of requests is set to 100 is lower than the that with 200 requests, as it generates low-price bids while distributing optimal resources to microservices. Further, the total payment to winners is always higher than the social cost, as SSAM guarantees that the payment to each winner is no smaller than its price but marginally higher than optimal price incurred by SSAM. This is because SSAM is able to choose more low-price bids to cover the same microservices from a larger set of bidders. We now validate individual rationality by comparing each winning bid's real price with its payment in Figure 4(a). Here, the payment is plotted by the brown color and the price is marked by the blue color. From the figure, we observe that the payment is always greater than the price. Figure 4(b) shows the running time of SSAM in different time slots. We use the Java platform to calculate the running time with `StopWatch` class. We run our experiments for ten times independently and present the average result. It can be observed that the running time is less than 100 milliseconds even with large data size and grows linearly with the increase of the number of microservices and of requests.

**Online Algorithm (MSOA).** Figure 5(a) illustrates the performance ratio calculated by dividing the social cost generated

by MSOA to the offline optimal social cost under different numbers of microservices and requests. Comparing the ratio to that of SSAM algorithm in Figure 3(a), we observe a small loss in performance. More importantly, the ratio slightly decreases with the increase in the number of microservices and of requests. This occurs, as MSOA selects more low-price bids from a larger microservices set, achieving a better performance ratio. Next, we compare MSOA with different extensions of MSOA. It can be observed that MSOA-DP achieves lower performance ratio compared to MSOA-RC and MSOA-OA, as MSOA-DP accurately estimates the microservices resource demand and based on that it participates in the auction process. Hence, it inherently minimizes the social cost and maintains a lower performance ratio. However, MSOA-RC and MSOA-OA cannot perform well in the process, as they restrict the resource capacity of each microservice, which automatically creates an economic imbalance in the process. Thus, MSOA-RC outperforms the other two solutions over a wide range of microservices and requests.

We next plot the ratio in Figure 6(a) with varying numbers of rounds $T$ and numbers of bids per user $J$. From the figure, we observe that a larger value of $J$ leads to worse performance and the expansion of $T$ has a negative impact on the performance ratio. If the edge platform permits microservices to submit multiple bids and the online auction consists of many rounds, then the performance ratio will become large. Figure 6(b) illustrates two different sets of results while varying the number of microservices. The three upper lines represent the social cost generated by MSOA, the total payment, and the optimal social cost when setting user requests to 100, respectively; the three lower lines are results with user requests set to 200. Form the figure, we observe a steep drop in social cost with different settings of requests.

## VI. Related Work

**Resource Management in Edge Clouds.** Resource management is critical to maintain optimal service quality and most economic operation of edge clouds. In recent years, this problem have been extensively studied [3]–[8], [23]–[25]. Wang et al. study resource allocation in mobile edge environments and propose a mobility-agnostic online algorithm [3]. Xu et al. propose a utility-aware resource allocation scheme for edge computing [7]. Some works also consider resource allocation coupled with service placement. For example, He et al. study a joint service placement and request scheduling problem in edge clouds for both sharable and no-sharable resources. For a specific application scenario – social virtual reality, Wang et al. devise an iterative algorithm for service entity placement in edge clouds [8]. Targeting machine learning use cases, Wang et al. propose an adaptive control mechanism for distributed machine learning at the edge [5]. Jiao et al. study the cost optimization problem in edge computing and propose a multi-granularity cloudlet control mechanism [6]. Generally speaking, all of the above solutions assume that the resources in edge clouds are fully managed by a central entity and the quality of service can be sacrificed whenever needed. However, such an assumption will unlikely follow in microservice-based edge computing environments.

**Auction Mechanisms for Cloud/Edge.** Auction-based approaches have been widely applied in Infrastructure-as-a-Service (IaaS) cloud environments. Zhang et al. propose a dynamic resource provisioning scheme for cloud computing using auction theory [11]. They also propose an online auctions in IaaS clouds for social welfare and profit maximization with server costs [12]. Shi et al. proposed an online auction framework for dynamic resource provisioning in cloud computing [13]. Another area of research that has extensively employed auction-based approaches is emergency demand response. Sun et al. propose an online incentive mechanism for emergency demand response in geo-distributed data centers [17]. Zhang et al. propose a truthful incentive mechanism for emergency demand response in colocation data centers [14]. He et al. propose an online demand response of GPU-based cloud computing with DVFS [15]. For the edge environment, Chen et al. first study emergency demand response in edge clouds and propose an online market mechanism [10]. Sbahreini et al. propose an envy-free auction mechanism for resource allocation in edge computing [16].

In summary, none of the existing works are able to provide an online incentive mechanism for resource sharing among microservices under uncertain demand request in edge clouds, while taking into consideration of profit and economical balance in the system.

## VII. Conclusion

In this paper, we targeted the dynamic resource sharing challenge in edge clouds and proposed an online incentive mechanism. We first removed the uncertainty of resource requirement of microservices through estimation and then employed an online auction framework to achieve microservice cooperation. In the online auction framework, we first studied a single-stage auction problem and proposed a winner selection algorithm achieving bounded approximation ratio, truthful bidding, and individual rationality. Based on the single-stage auction, we further proposed a multi-stage auction design for online optimization. We validated the performance of the proposed mechanism through both rigorous theoretical analysis and extensive simulations. We also like to extend the work while considering the diverse processing time of each task in our future work.

## Acknowledgement

This work was funded by the German Research Foundation (DFG) as part of the project C7 within the Collaborative Research Center (CRC) 1053 – MAKI and by the DFG and the National Natural Science Foundation of China (NSFC) joint Sino-German research project under Grant No. 392049569 (DFG) and No. 61761136014 (NSFC).

## References

[1] M. Satyanarayanan, "The emergence of edge computing," *IEEE Computer*, vol. 50, no. 1, pp. 30–39, 2017.

[2] N. Savage, "Going serverless," *Commun. ACM*, vol. 61, pp. 15–16, 2018.

[3] L. Wang, L. Jiao, J. Li, and M. Mühlhäuser, "Online resource allocation for arbitrary user mobility in distributed edge clouds," in *IEEE ICDCS*, 2017, pp. 1281–1290.

[4] T. He, H. Khamfroush, S. Wang, T. L. Porta, and S. Stein, "It is Hard to Share: Joint Service Placement and Request Scheduling in Edge Clouds with Sharable and Non-sharable Resources," in *IEEE ICDCS*, 2018.

[5] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "When edge meets learning: Adaptive control for resource-constrained distributed machine learning," in *IEEE INFOCOM*, 2018.

[6] L. Jiao, L. Pu, L. Wang, X. Lin, and J. Li, "Multiple granularity online control of cloudlet networks for edge computing," in *IEEE SECON*, 2018.

[7] J. Xu, B. Palanisamy, H. Ludwig, and Q. Wang, "Zenith: Utility-aware resource allocation for edge computing," in *IEEE EDGE*, 2017.

[8] L. Wang, L. Jiao, T. He, J. Li, and M. Mühlhäuser, "Service entity placement for social virtual reality applications in edge computing," in *IEEE INFOCOM*, 2018, pp. 1–9.

[9] A. Samanta, Y. Li, and F. Esposito, "Battle of microservices: Towards latency-optimal heuristic scheduling for edge computing," in *IEEE NetSoft*, 2019.

[10] S. Chen, L. Jiao, L. Wang, and F. Liu, "An online market mechanism for emergency demand response via cloudlet control," in *IEEE INFOCOM*, 2019, pp. 1–9.

[11] L. Zhang, Z. Li, and C. Wu, "Dynamic resource provisioning in cloud computing: A randomized auction approach," in *IEEE INFOCOM*, 2014.

[12] X. Zhang, Z. Huang, C. Wu, Z. Li, and F. Lau, "Online auctions in IaaS clouds: welfare and profit maximization with server costs," in *ACM SIGMETRICS*, 2015.

[13] W. Shi, L. Zhang, C. Wu, Z. Li, F. Lau, W. Shi, L. Zhang, C. Wu, Z. Li, and F. Lau, "An online auction framework for dynamic resource provisioning in cloud computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 4, pp. 2060–2073, 2016.

[14] L. Zhang, S. Ren, C. Wu, and Z. Li, "A truthful incentive mechanism for emergency demand response in colocation data centers," in *IEEE INFOCOM*, 2015.

[15] Y. He, L. Ma, and C. Huang, "Online Demand Response of GPU Cloud Computing with DVFS," in *IEEE IWQoS*, 2018.

[16] T. Bahreini, H. Badri, and D. Grosu, "An envy-free auction mechanism for resource allocation in edge computing systems," in *IEEE/ACM SEC*, 2018, pp. 313–322.

[17] Q. Sun, S. Ren, C. Wu, and Z. Li, "An online incentive mechanism for emergency demand response in geo-distributed colocation data centers," in *ACM e-Energy*, 2016.

[18] R. W. Saaty, "The analytic hierarchy process – what it is and how it is used," *Mathematical modelling*, vol. 9, no. 3-5, pp. 161–176, 1987.

[19] W. Shi, L. Zhang, C. Wu, Z. Li, and F. Lau, "An online auction framework for dynamic resource provisioning in cloud computing," in *ACM SIGMETRICS*, 2014.

[20] J. Kleinberg and E. Tardos, *Algorithm design*. Pear. Edu. Ind., 2006.

[21] R. B. Myerson, "Optimal auction design," *Mathematics of operations research*, vol. 6, no. 1, pp. 58–73, 1981.

[22] A. Samanta and Z. Chang, "Adaptive Service Offloading for Revenue Maximization in Mobile Edge Computing with Delay-Constraint," *IEEE Internet of Things Journal*, pp. 1–1, 2019.

[23] L. Wang, L. Jiao, J. Li, J. Gedeon, and M. Mühlhäuser, "Moera: Mobility-agnostic online resource allocation for edge computing," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2018.

[24] L. Pu, L. Jiao, X. Chen, L. Wang, Q. Xie, and J. Xu, "Online resource allocation, content placement and request routing for cost-efficient edge caching in cloud radio access networks," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 8, pp. 1751–1767, Aug 2018.

[25] L. Jiao, A. M. Tulino, J. Llorca, Y. Jin, and A. Sala, "Smoothed online resource allocation in multi-tier distributed cloud networks," *IEEE/ACM Transactions on Networking*, vol. 25, no. 4, pp. 2556–2570, 2017.