A Truthful Online Mechanism for Resource Allocation in Fog Computing

Fan Bi[1], Sebastian Stein[1], Enrico Gerding[1],
Nick Jennings[2], and Thomas La Porta[3]

[1] University of Southampton, Southampton, UK
{fb1n15,ss2,eg}@ecs.soton.ac.uk
[2] Imperial College London, London, UK
[3] Penn State University, State College, USA

Abstract. Fog computing is a promising Internet of Things (IoT) paradigm in which data is processed near its source. Here, efficient resource allocation mechanisms are needed to assign limited fog resources to competing IoT tasks. To this end, we consider two challenges: (1) near-optimal resource allocation in a fog computing system; (2) incentivising self-interested fog users to report their tasks truthfully. To address these challenges, we develop a truthful online resource allocation mechanism called flexible online greedy. The key idea is that the mechanism only commits a certain amount of computational resources to a task when it arrives. However, when and where to allocate resources stays flexible until the completion of the task. We compare our mechanism to four benchmarks and show that it outperforms all of them in terms of social welfare by up to 10% and achieves a social welfare of about 90% of the offline optimal upper bound.

Keywords: Mechanism design · Fog computing · IoT · Resource allocation.

1 Introduction

The Internet of Things (IoT) is developing rapidly, and it is estimated that by 2025, 22 billion active devices will be in the IoT (Lueth, 2018). Since it is impossible to let the often low-powered IoT devices perform all computing tasks, some of which are highly computationally demanding, a common solution is to combine IoT and cloud computing (Doukas and Maglogiannis, 2012; Sajid et al., 2016). However, cloud computing alone cannot satisfy all the computing requirements from the IoT (Bonomi et al., 2012). The main reason is that transferring all the data from the IoT to the cloud to analyse requires a huge amount of bandwidth, and many IoT applications, such as autonomous vehicles, augmented reality and virtual reality, need very low latency, which cloud computing cannot guarantee. Consequently, fog computing has been proposed to make up for these shortcomings (Bonomi et al., 2012). In simple terms, the key difference is that the fog is closer to the IoT devices than the cloud (CIS, 2015). To make the most of the fog resources and maximise the efficiency, good resource allocation mechanisms for fog computing are needed. However, unlike cloud computing, fog computing

cannot ignore bandwidth constraints because it is common to send large volumes of traffic between IoT devices and fog nodes (FNs). Another difference is that many tasks in the fog are time-oriented, which means that they need a certain amount of computational time to achieve their maximum value, but they can still achieve part of the value if they are allocated less time. For example, suppose a user wants to run a video surveillance application with facial recognition to surveil their shops for 24 hours. In this case, the large volume of video streams from the cameras in their shops will be sent to a nearby FN instead of a remote data centre to do the compute-intensive analysis. Furthermore, it is still of value to them if the surveillance lasts less than 24 hours, say, 18 hours.

To address these challenges, researchers have proposed many resource allocation mechanisms for fog computing (or similar computing paradigms such as cloud computing, edge computing or geo-distributed clouds) in order to save energy, reduce cost or improve quality of service (Aazam and Huh, 2015; Cardellini et al., 2015; Gu et al., 2018). However, most of these mechanisms were not specifically designed for settings where users act strategically to maximise their utility (e.g., users may misreport higher value for their tasks to increase their chances of acceptance). To address this problem, some researchers have proposed truthful mechanisms that incentivise users to truthfully reveal their private information. However, these approaches cannot be applied directly to our model due to subtle but important differences. For example, several truthful mechanisms are designed to schedule tasks in the cloud (Wang et al., 2012; Lucier et al., 2013; Wang et al., 2015; Chawla et al., 2017; Zhu et al., 2018). However, they all assume single-minded agents (i.e., agents who do not get any value for a partially executed task). In addition, the model by Lucier et al. (2013) assumes that each task requires a certain amount of resource to complete rather than a certain running time, which is very different from the time-oriented tasks in the fog. Furthermore, Zhang et al. (2015) and Shi et al. (2017) also propose truthful mechanisms for single-minded users in a geo-distributed cloud, and they assume users can specify all the details about the placement of resources among data centres for their tasks, which is not very practical mainly because users rarely have the knowledge of the system structure. Finally, Hayakawa et al. (2018) introduce the price-based mechanisms for homogeneous resource allocation, whereas there are several heterogeneous resources in the fog. So their resource allocation framework needs to be adapted in this case. In addition, we choose online greedy (OG) and Social Welfare Maximisation Online Auction 2 (SWMOA2), which are adapted from mechanisms in (Wang et al., 2012; Shi et al., 2017) respectively, as the state-of-the-art benchmarks that we will evaluate our mechanism against.

In this paper, we are the first to address these shortcomings. Specifically, we design *dominant-strategy incentive compatible* (DSIC) and *individually rational* (IR) mechanisms for realistic fog settings to maximise social welfare[4]. DSIC mechanisms guarantee that regardless of others' behaviours, users always maximise their utility by reporting truthfully. Furthermore, under an IR mechanism,

[4] We define social welfare as the difference between the value of all fog tasks and the operational costs of all fog tasks.

no user will get a negative utility by participation. Such mechanisms provide two major benefits. First, they can elicit the true information about the tasks. Second, fog users do not need to invest their resources into optimally manipulating their bids to increase their utility. In addition, we focus on improving social welfare in this paper and leave the objective of maximising the fog provider's revenue to future work.

To design a truthful mechanism which addresses these problems, we significantly extend the framework proposed by Hayakawa et al. (2018) to our problem model. This is because their resource allocation model is similar to ours, and they show that a well-defined price-based mechanism can achieve high efficiency. In brief, we extend the state of the art as follows:

- We are the first to formulate the resource allocation in fog computing (RAFC) problem as a constraint optimisation problem that considers the bandwidth constraints and allows flexible allocation of virtual machines (VMs) (i.e., emulations of real computers that contain all the necessary elements to run fog tasks) and of the bandwidth. We also show that it can be modelled as an online mechanism design problem where a fog user requests an amount of usage time with a given resource configuration.
- We design a DSIC and IR online mechanism called flexible online greedy (FlexOG) for RAFC and show by extensive simulations, that it achieves a social welfare better than that achieved by the state-of-the-art benchmarks (up to 10%) and is close to the offline optimal value (around 90%).

The remainder of the paper is organised as follows: In Section 2, we propose a formal model of the RAFC problem. In Section 3, we present our proposed resource allocation mechanism as well as other benchmark mechanisms. In Section 4, we show the results of simulations and evaluate the performance of our mechanism. Finally, in Section 5, we conclude the paper.

2 The Fog Resource Model

We briefly describe our model of RAFC. The fog computing system is owned by a fog provider. It contains a set P of geo-distributed FNs and a set L of locations, which are interconnected through a set $\mathbb{E}$ of data links, as shown in Figure 1. Furthermore, there is a set E_l of IoT devices in each location l. An IoT device (e.g., a smart TV, surveillance camera, smart speaker or smartphone) is denoted as $e \in E_l$. Every FN $p \in P$ has a set R of limited computational resources (e.g., CPU, RAM and disk storage). Moreover, there are $A_{p,r}$ units of type $r \in R$ resources in FN p, and the unit operational cost of resource r in FN p is $o_{p,r}$. In addition, the bandwidth capacity and the unit operational cost of link $(j,k) \in \mathbb{E}$ are $b_{j,k}$ and $o_{j,k}$ respectively. For simplicity, we assume that the bandwidth capacity and unit bandwidth costs are symmetrical for all links (i.e., $b_{j,k} = b_{k,j},\ o_{j,k} = o_{k,j},\ \forall (j,k) \in \mathbb{E}$). FNs and data links together offer their resources to satisfy the needs of fog users. In particular, we assume that VMs can be created in an FN to run fog tasks as long as there are enough computational resources in that FN, and the total resource requirements of several virtual machines are just the sum of their resource requirements. Furthermore, the fog provider controls the resource allocation of the fog through a central control

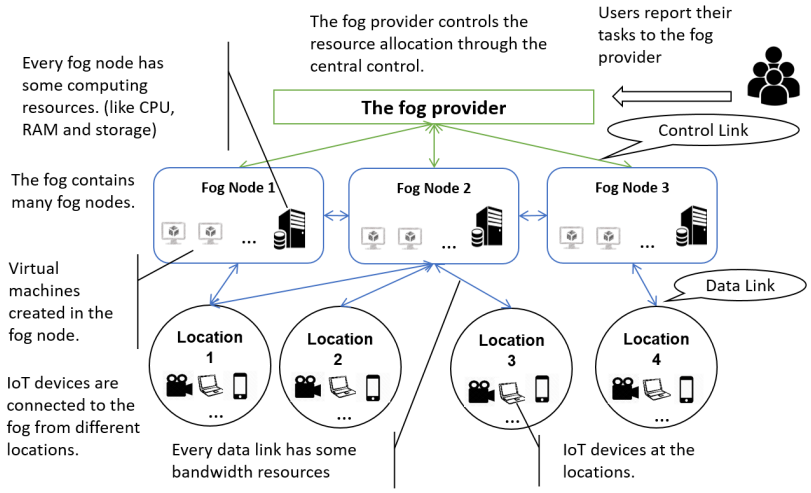

Fig. 1: General view of a fog computing system.

system, which is a server that receives reports of tasks from fog users, makes decisions of how to allocate resources and executes them through control links.

Fog users with tasks arrive over time and I denotes the set of all tasks. Note that we adopt a continuous time system, but the tasks can only start execution at discrete time steps, denoted by the set $T = \{1, 2, \ldots, |T|\}$. Each task $i \in I$ is owned by a user, which is also denoted as i for simplicity. In addition, the arrival time of task i is $T_i^a \in [0, |T|]$, which is the time when user i becomes aware of its task i, and the time interval that the task can run is from its start time T_i^s to its finish time T_i^f. Here, we assume that no tasks arrive at the exact same time. User i reports its task's type $\hat{\theta}_i$ (as defined in the following) at time $\hat{T}_i^a$ to run a certain application (e.g., a video surveillance application or a picture processing application). We assume that user i wants to know the number of time steps $\tilde{t}_i$ it will get and the payment $\tilde{p}_i$ for its task also by time $\hat{T}_i^a$ because users want to run the tasks locally or elsewhere if their tasks get rejected. The operational cost of task i is denoted as o_i, which is the sum of costs of all resources allocated to task i, including the cost of bandwidth. Furthermore, we also assume that every task only requires one VM to run but may require connections to several IoT devices $e \in E$ (in the same location or in different locations) because this is common in an IoT system. Users are also assumed to be stationary, which means that the IoT devices of users do not change locations over time. Furthermore, we also assume VMs can migrate between FNs and the migration costs are negligible, and all tasks are preemptible, which means that they can always be paused and resumed. Finally, we focus on one type of task called time-oriented tasks (e.g., video surveillance and video processing tasks), which are common in fog computing. Such a task i needs a certain capacity

of resources for a time length t_i to get its full value, but can still get part of the value if the processing time is less than t_i. Formally, the type of task i: θ_i is a tuple $(T_i^a, T_i^s, T_i^f, \boldsymbol{v}_i, \{a_{i,r}\}_{r\in R}, \{\Gamma_l^i\}_{l\in L})$, where $a_{i,r}$ denotes the amount of resource $r \in R$ required, and Γ_l^i denotes the bandwidth demand between its VM and location $l \in L$. For simplicity, bandwidth demands are symmetrical. That is, Γ_l^i denotes both the bandwidth demands to and from location $l \in L$. In this paper, the valuation function $\boldsymbol{v}_i = \{v_{i,0}, v_{i,1}, \ldots, v_{i,t_i}\}$, where $v_{i,t}$ is the value when task i gets usage time of t time steps and t_i denotes the usage time needed to get the full value of the task. We make a mild assumption that the value monotonically increases with usage time (i.e., $v_{i,t'} \geq v_{i,t''}, \forall t' \geq t''$). We choose this type of valuation function because it corresponds to many applications in the fog, which achieve better results as processing time increases. Moreover, the reported type of task i: $\hat{\theta}_i$ is a tuple $(\hat{T}_i^a, \hat{T}_i^s, \hat{T}_i^f, \hat{\boldsymbol{v}}_i, \{\hat{a}_{i,r}\}_{r\in R}, \{\hat{\Gamma}_l^i\}_{l\in L})$, and $\hat{\theta}^{\langle t\rangle}$ denotes the set of all reported types until and including time t.

Now, a key assumption in our work is that users are strategic, so $\hat{\theta}_i$ may not be equal to θ_i. Moreover, we assume limited misreports (Nisan et al., 2007) based on the nature of our problem (i.e. $\hat{T}_i^a \geq T_i^a, \hat{T}_i^s \geq T_i^s, \hat{T}_i^f \leq T_i^f, \hat{a}_{i,r} \geq a_{i,r}, \hat{\Gamma}_l^i \geq \Gamma_l^i$). This is reasonable because a user cannot bid for a task before it becomes aware of it, and cannot bid a looser time constraint ($\hat{T}_i^s < T_i^s$ or $\hat{T}_i^f > T_i^f$) because the provider can check whether i is ready to run at $\hat{T}_i^s$ and withhold the results for i until $\hat{T}_i^f$. So bidding $\hat{T}_i^s < T_i^s$ will be detected and penalised by cancelling the task and bidding $\hat{T}_i^f > T_i^f$ will get no value. Finally, user i will not misreport a lower resource requirement because its task cannot run in that case.

Next, when receiving the bid $\hat{\theta}_i$ for task i, the fog provider will decide the resource allocation scheme λ_i to this task, how much usage time $\tilde{t}_i$ will be allocated, and the payment $\tilde{p}_i$ right away because of the assumption we made earlier. Formally, the fog provider solves a constraint optimisation problem, and the decision variables are: (1) $\{z_{p,t}^i \in \{0,1\}\}_{i\in I, p\in P, t\in T}$, indicating that the VM of task i is placed in FN p ($z_{p,t}^i = 1$), or not ($z_{p,t}^i = 0$) at time step t. (2) $\{f_{l,p,j,k,t}^i \in \mathbb{R}^+\}_{i\in I, l\in L, p\in P, (j,k)\in\mathbb{E}, t\in T}$, indicating allocation of the bandwidth on each link for task i at time step t. (3) $\tilde{p}_i(\lambda_i, \hat{\theta}^{\langle \hat{T}_i^a\rangle}) \in \mathbb{R}^+$, denoting the payment of task i, which is a function of the allocation: λ_i and all information received by $\hat{T}_i^a$: $\hat{\theta}^{\langle \hat{T}_i^a\rangle}$. So, for task i, its usage time $\tilde{t}_i = \sum_{p\in P, t\in T} z_{p,t}^i$, resource allocation scheme $\lambda_i = \{z_{p,t}^i\}_{i\in I, p\in P, t\in T} \cup \{f_{l,p,j,k,t}^i\}_{i\in I, l\in L, p\in P, (j,k)\in\mathbb{E}, t\in T}$ and its utility is $u_i = \boldsymbol{v}_i(\tilde{t}_i) - \tilde{p}_i(\lambda_i, \hat{\theta}^{\langle \hat{T}_i^a\rangle})$. The objective function of this optimisation problem maximises the social welfare:

$$\underset{\lambda_i}{\text{maximise}} \sum_{i\in I} \boldsymbol{v}_i\Big(\sum_{p\in P, t\in T} z_{p,t}^i\Big) - o \tag{1}$$

where $o = \sum_{i\in I, r\in R, p\in P, t\in T} a_{i,r} z_{p,t}^i o_{p,r} + \sum_{i\in I, l\in L, p\in P, (j,k)\in\mathbb{E}, t\in T} 2o_{j,k} f_{l,p,j,k,t}^i$

The constraints of the optimisation problem include resource constraints in the fog system and time constraints for fog tasks. Please refer to Bi et al. (2019)

for details on these constraints. This is a mixed integer linear programming problem, and we use the IBM ILOG CPLEX optimiser to solve it in our simulations.

3 Allocation Mechanisms

In this section, we present the details of the mechanisms used in this paper.

3.1 Price-based Mechanisms

First, we introduce a class of online resource allocation mechanisms called price-based mechanisms that guarantee DSIC and IR for our resource allocation problem. Specifically, the properties that this class of mechanisms should have are:

Definition 1. *A monotonic payment function is (weakly) monotonically increasing over $\hat{T}_i^a$, $\hat{T}_i^s$, $\hat{t}_i$, $\hat{a}_{i,r}, r \in R$ and $\hat{\Gamma}_l^i, l \in L$, and (weakly) monotonically decreasing over $\hat{T}_i^f$.*

Definition 2. *An online mechanism belongs to the price-based mechanisms class if it has the following properties:*

1. *The mechanism computes the payment $\tilde{p}_i$ for any possible allocation λ_i to task i by using a payment function $\tilde{p}_i(\lambda_i, \hat{\theta}^{\langle \hat{T}_i^a \rangle})$ that is independent of $\hat{\boldsymbol{v}}_i$ and monotonic.*
2. *The payment for tasks with no resource allocated is zero.*
3. *The resource allocation scheme λ_i for task i maximises $\hat{\boldsymbol{v}}_i - \tilde{p}_i$ (over all λ_i that can be made to task i for any $\hat{\boldsymbol{v}}_i$).*

Then, the following theorem guarantees that any mechanism in the class of price-based mechanisms is DSIC and IR.

Theorem 1. *Any online mechanism that satisfies Definition 2 is DSIC and IR.*

This theorem can be proved in a similar way to Theorem 1 in (Hayakawa et al., 2018), and the proof is omitted for space reasons.

3.2 Benchmark Mechanisms

We describe the benchmark mechanisms used in this paper in detail below.

Offline Optimal Mechanism Under this mechanism, we assume that we know all the information about future tasks and allocate resources to optimise the social welfare with no need to incentivise fog users to bid truthfully. This theoretical and idealised case can be achieved by solving the optimisation problem 1.

Online Optimal Mechanism This mechanism is similar to the offline optimal except that the optimisation problem is solved at each time step with knowledge only of the tasks that have arrived so far (and not of future tasks). Note that this mechanism is non-truthful, but we use this to determine the social welfare that could be achieved in an online setting if all users report truthfully. In Section 4, we also evaluate this mechanism in settings where some users misreport.

Online Greedy Mechanism (OG) This mechanism is an extension of the greedy algorithm from (Wang et al., 2012), and greedily allocates resources to maximise the utility of a task when it arrives and commits to this allocation. Furthermore, it computes the payment as this task's corresponding operational costs ($\tilde{p}_i = o_i = \sum_{r \in R, p \in P, t \in T}(a_{i,r} z_{p,t}^i o_{p,r}) + \sum_{l \in L, p \in P, (j,k) \in \mathbb{E}, t \in T}(2 o_{j,k} f_{l,p,j,k,t}^i)$). Note that OG belongs to the price-based mechanisms and thus is DSIC and IR.

SWMOA2 Although the Social Welfare Maximisation Online Auction (SWMOA) mechanism from (Shi et al., 2017) cannot be directly applied to our model, we develop a variant of it called SWMOA2 as a suitable benchmark. The main difference between this mechanism (given in Algorithm 1) and OG is that it keeps a virtual cost instead of an operational cost for every resource. For convenience, we use M to denote the set of every computational resource at each FN and the bandwidth resource on each link, and m is one type of them. To compute the virtual costs, we define the load factor $\kappa_{m,t}$ to be the proportion of occupied resource m at time step t. Then, the virtual cost accordingly is: $c_{m,t} = \mu^{\kappa_{m,t}} - 1, \forall t \in T, m \in M$, where $\mu = 2|M|F + 2$, and F is the upper limit of the ratio between the highest and the lowest task valuation per time step. Then, the virtual cost of task i is $c_i = \sum_{r\in R,p\in P,t\in T}(a_{i,r}z^i_{p,t}c_{p,r,t}) + \sum_{l\in L,p\in P,(j,k)\in\mathbb{E},t\in T}(2c_{j,k,t}f^i_{l,p,j,k,t})$. SWMOA2 also belongs to the price-based mechanisms and is DSIC and IR.

Algorithm 1: The SWMOA2 mechanism

$\theta_{all} \leftarrow \emptyset$ ▷ The set of arrived tasks
$\Lambda \leftarrow \emptyset$ ▷ The set of committed allocation decisions
$\kappa_{m,t} \leftarrow 0, \forall m, t$ ▷ The load factors of resources
$c_{m,t} \leftarrow 0, \forall m, t$ ▷ The virtual costs of resources
for t *in* T **do**
  while *new tasks arrive within* t **do**
    When a new task i arrives ▷ Tasks arrive over time
    $\theta_{all} \leftarrow \theta_{all} \cup i$ ▷ Update the set of arrived tasks
    Solve the maximum virtual utility allocation for task i (i.e., $\arg\max_{\lambda_i}(\hat{\boldsymbol{v}}_i(\lambda_i) - c_i(\lambda_i))$) ▷ Find the allocation that maximises task i's virtual utility
    $\Lambda \leftarrow \Lambda \cup \lambda_i$ ▷ Commit this allocation
    $\tilde{p}_i \leftarrow c_i(\lambda_i)$ ▷ Compute the payment for task i
    $\kappa_{m,t} \leftarrow \kappa_{m,t} + z^i_{p,t}a_{i,r}/A_{p,r}, \forall m \in P \times R, t \in T$ ▷ Update load factors of computational resources
    $\kappa_{m,t} \leftarrow \kappa_{m,t} + \sum_{l\in L,p\in P} f^i_{l,p,j,k,t}/b_{j,k}, \forall m \in \mathbb{E}, t \in T$ ▷ Update load factors of bandwidth resources
    $c_{m,t} = \mu^{\kappa_{m,t}} - 1, \forall t \in T, m \in M$ ▷ Update the virtual costs of resources
  end
  Allocate resources for next time step $(t + 1)$ according to Λ
end

3.3 Flexible Online Greedy Mechanism (FlexOG)

Our mechanism, FlexOG (Algorithm 2), builds upon OG by allocating newly arrived tasks greedily but keeps their specific allocation schemes flexible. This gives it the DSIC property of OG but adds more flexibility. This also results in higher social welfare because there is more space for optimisation when high-value tasks arrive in the future. After receiving a report of task i, FlexOG finds the allocation that maximises the social welfare of all flexible tasks given the constraints of their committed usage time. Then, FlexOG computes the usage time $\tilde{t}_i$ for task i from its corresponding allocation scheme, and commits it to task i, which means that task i is guaranteed to get $\tilde{t}_i$ usage time before its reported finish time $\hat{T}^f_i$. Afterwards, FlexOG requires payment for task i as the marginal total operational cost, and task i is put to the set of flexible tasks. In addition, at the end of each time step, FlexOG allocates resources for the next time step according to the latest allocation schemes. Finally, if a task will get

Algorithm 2: The FlexOG mechanism

$\theta_{all} \leftarrow \emptyset$ ▷ The set of arrived tasks
$\theta_{flex} \leftarrow \emptyset$ ▷ The set of flexible tasks
$o \leftarrow 0$ ▷ The total operational costs
$\tilde{T} \leftarrow \emptyset$ ▷ The set of committed processing times
for t *in* T **do**
  while *new tasks arrive within* t **do**
    When a new task i arrives ▷ Tasks arrive over time
    $\theta_{all} \leftarrow \theta_{all} \cup i$ ▷ Update the set of arrived tasks
    $\theta_{flex} \leftarrow \theta_{flex} \cup i$ ▷ Update the set of flexible tasks
    Solve the maximum utility allocation for tasks in θ_{flex} (i.e., $\arg\max_{\lambda_j} \sum_{j \in \theta_{flex}} (\hat{\boldsymbol{v}}_j(\lambda_j) - o_j(\lambda_j))$) ▷ Find the allocation for tasks in θ_{flex} that maximise their social welfare, given their committed usage time
    $\tilde{T} \leftarrow \tilde{T} \cup \tilde{t}_i(\lambda_i)$ ▷ Commit the processing time to i
    $\tilde{p}_i \leftarrow \sum_{j \in \theta_{all}} o_j(\lambda_j) - o$ ▷ Compute the payment for i
    $o \leftarrow \sum_{j \in \theta_{all}} o_j(\lambda_j)$ ▷ Update the total operational costs
  end
  for i *in* θ_{flex} **do**
    Allocate resources for the next time step $(t+1)$ according to λ_i
    $\tilde{t}_i \leftarrow \tilde{t}_i - \sum_{p \in P} z^i_{p,t+1}$ ▷ Update the remaining processing time of task i
    if $\tilde{t}_i = 0$ **then**
      $\theta_{flex} \leftarrow \theta_{flex} \setminus i$ ▷ Delete task i from flexible tasks if it gets its comitted usage time
    end
  end
end

all of its committed usage time in the next time step, it will be removed from the set of flexible tasks. In summary, the key idea of our mechanism is that we only commit the usage time $\tilde{t}_i$ to task i but keep its allocation scheme flexible.

Theorem 2. *The FlexOG mechanism is DSIC and IR.*

We only give a proof sketch here because of space reasons. Obviously, FlexOG satisfies condition 2 in Definition 2 by charging zero to a rejected task. The payment $\tilde{p}_i(\lambda_i, \hat{\theta}^{\langle \hat{T}^a_i \rangle})$ is independent of $\hat{\boldsymbol{v}}$ because by maximising $\sum_{j \in \theta_{flex}} (\hat{\boldsymbol{v}}_j(\lambda_j) - o_j(\lambda_j))$ FlexOG actually minimises the total operational cost, which is independent of $\hat{\boldsymbol{v}}$. The payment function is also monotonic because increasing $\hat{T}^a_i, \hat{T}^s_i$, $\hat{t}_i$, $\{\hat{a}_{i,r}\}_{r \in R}$, $\{\hat{\Gamma}^i_l\}_{l \in L}$ or decreasing $\hat{T}^f_i$ can only increase the total operational cost $\sum_{j \in \theta_{flex}} o_j$. Hence, this mechanism satisfies condition 1. The mechanism also satisfies condition 3 because it maximises $\sum_{j \in \theta_{flex}} (\hat{\boldsymbol{v}}_j(\lambda_j) - o_j(\lambda_j))$, which is equivalent to maximise $(\hat{v}_i - \tilde{p}_i)$ according to how the payment is computed by FlexOG. From the above, FlexOG is DSIC and IR by Theorem 1.

4 Simulations and Analysis

In this section, we describe the setup of our experiments and evaluate our proposed mechanism by simulations. The aim is to compare the social welfare achieved by FlexOG to benchmark mechanisms in different situations.

4.1 Experimental Setup

We generate the following synthetic data to use in simulations because there currently exists no comprehensive data set of real-world fog computing tasks.

The basic parameters of the synthetic data are as follows. The time span of our discrete time period is $|T| = 12$. The fog provider has 6 FNs ($|P|$=6) and 6 locations ($|L|$=6). The topology of this setup is shown in Figure 2. Additionally, there are $|R| = 3$ types of computational resources (CPU, RAM, and disk storage) at each FN. We choose this small setting so that we can run more trials in a reasonable time for all mechanisms, and we get similar results in other settings on a similar scale.

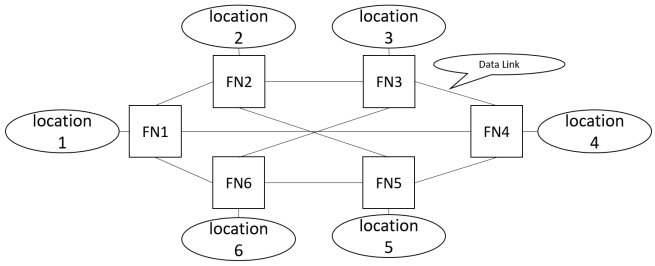

Fig. 2: The topology of the fog computing system.

The number of tasks in this time period is $|I| = 40$. The arrival time T_i^a follows a continuous uniform distribution $U(0, 10)$, so that no tasks arrive at exactly the same time. Moreover, the number of IoT devices for each task E_i is generated uniformly from $\{1, 2, \ldots, 6\}$. The location of each IoT device $u_{e,l}^i$ is chosen uniformly at random from all locations L with replacement.

Furthermore, we choose a special valuation function v_i in our simulation for simplicity, which is a non-decreasing linear function of the usage time $\tilde{t}_i$.

$$v_i(\tilde{t}_i) = \begin{cases} g_i \times \tilde{t}_i & \text{if } \tilde{t}_i \leq t_i \\ g_i \times t_i & \text{if } \tilde{t}_i > t_i \end{cases}$$

where the coefficient g_i represents task i's obtained value per usage time.

To make the resource allocation more realistic, there are two types of tasks in this synthetic data: low-value tasks and high-value tasks, and the proportion of high-value tasks is denoted as $q \in [0, 1]$. For task i of either type: $a_{i,r}, \forall r \in R$ and Γ_l^i are all generated from a Gaussian distribution $\mathcal{N}(1, 1)$ with negative results discarded. The usage duration t_i is a positive integer uniformly chosen from $\{1, 2, 3, 4\}$, and the start time T_i^s is an integer uniformly chosen within 2 time steps after the arrival time: $\{\lceil T_i^a \rceil, \lceil T_i^a \rceil + 1, \lceil T_i^a \rceil + 2\}$. Furthermore, the finish time T_i^f is an integer uniformly chosen between a and b time steps after the earliest finish time (not exceeding the last time step): $\{T_i^s + t_i - 1 + a, T_i^s + t_i + a, \ldots, min(T_i^s + t_i - 1 + b, |T|)\}$, and (a, b) defines the deadline slackness of the task, which is an important parameter because it reflects the task's flexibility. For low-value task i, g_i is uniformly chosen from a continuous interval: $[8, 30]$. However, for high-value task i, g_i is uniformly chosen from a continuous interval: $[180, 200]$. Thus, $F = 200/8 = 25$ in this case. Finally, users who misreport only misreport their valuation coefficient as one million.

Furthermore, the overall resource capacity of each computational resource r: $\sum_{p \in P} A_{p,r}$ is set to be a k fraction of the corresponding total resource demand:

$\sum_{i\in I} a_{i,r}$, and the overall bandwidth capacity: $\sum_{(j,k)\in\mathbb{E}} b_{j,k}$ is set to be a $2k$ fraction of the total bandwidth demands: $\sum_{i\in I, l\in L} \Gamma_l^i$. Then, each FN receives the same fraction of resource r: $\frac{\sum_{p\in P} A_{p,r}}{|P|}$, and each data link receives the same fraction of the available total bandwidth: $\frac{\sum_{(j,k)\in\mathbb{E}} b_{j,k}}{|\mathbb{E}|}$. Finally, the unit operational costs at different FNs and links: $o_{p,r}, p \in P, r \in R, o_{j,k}, (j,k) \in \mathbb{E}$ are all generated uniformly from [0.03, 0.1].

4.2 Simulation Results

We have tested the robustness of our mechanism by running simulations with different parameters, such as the number of tasks, the value distribution, the arrival time distribution, the operational costs of resources, deadline slackness, and resource scarcity in FNs and data links. We only show representative results below due to the space limitation. Across all of these settings, trends are similar. In particular, the FlexOG's performance in social welfare is typically around 90% of the offline optimal, and between $5-10\%$ better than OG's.

First, we compare the total social welfare achieved by FlexOG with other benchmarks under different resource coefficients k indicating the scarcity of the resources in Figure 3.[5] Note that we normalise the results to the performance of offline optimal so that it is easier to compare the performance of different mechanisms. The figure shows that FlexOG consistently achieves better social welfare than other truthful benchmark mechanisms. In particular, SWMOA2 always has the worst performance mainly because its virtual price function is exponential to the amount of occupied resource, and this hinders tasks from getting allocated even when there is enough resource for them. It is worth noting that, although the price function of SWMOA can guarantee that the allocation will not break the resource constraints for the problem model in (Shi et al., 2017), it no longer has this function in our model. The reason FlexOG performs better than OG is the way in which committed time steps are allocated to tasks is flexible, and so it can reschedule unfinished tasks to allocate more time steps for the newly arrived task. In addition, our mechanism also performs close to offline optimal, achieving around 90%, which indicates that our mechanism is efficient even though it is online. Although online optimal performs about 10% better than FlexOG, its performance drops below that of FlexOG when just 20% of users misreport. In addition, we have also tested whether users have the incentive to misreport by comparing utilities of truthful and non-truthful users, and the result shows on average non-truthful users get a higher utility. This means that, in a strategic setting where users can misreport, FlexOG can actually achieve significantly more social welfare than online optimal. The figure also shows that the performance difference between FlexOG and OG shrinks when the resource coefficient k is relatively low or high. Intuitively, this is because when there are few resources or there are abundant resources the performance of OG will be

[5] All figures are with 95% confidence intervals based on 200 trials, and the relative tolerance is set to 1% for offline optimal, and 5 % for others. (A 1% tolerance means that the CPLEX optimiser stops when a solution is within 1% of optimality)

closer to the optimal, and there is less space for FlexOG to improve social welfare by rescheduling tasks.

Next, we compare the performance in social welfare under different levels of task slackness in Figure 4. A task with more slackness has more time steps between its earliest and latest finish times. Such tasks are more flexible to allocate. As can be seen from the figure, the gap between FlexOG and OG increases as the tasks becomes more slack. This is because when tasks are more slack, FlexOG is more likely to reschedule low-value tasks to allocate more high-value tasks, while OG cannot benefit from this since its resource allocation schemes are fixed once they have been made.

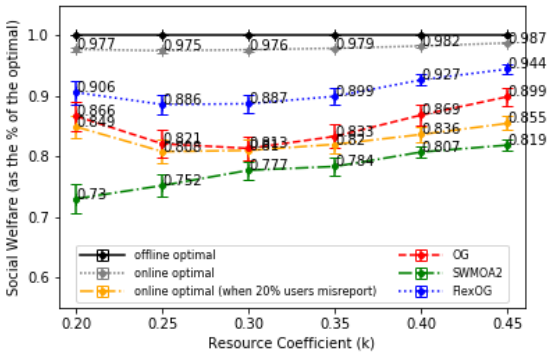

Fig. 3: Social welfare achieved by the mechanisms $((a, b) = (5, 10), F = 25, q = 0.1)$

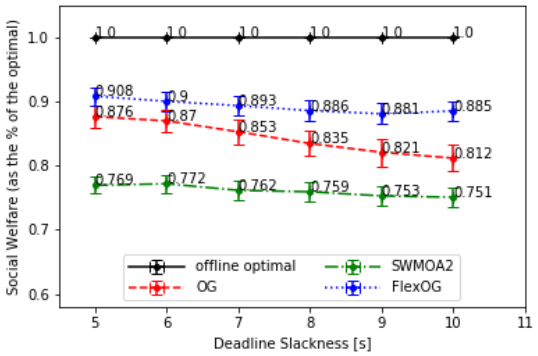

Fig. 4: Social welfare achieved by the mechanisms $((a, b) = \{(0, 5), (1, 6), (2, 7)(3, 8), (4, 9), (5, 10)\}, F = 25, q = 0.1, k = 0.3)$.

Finally, the evaluation of processing time is shown in Figure 5. We plot the processing time of all mechanism only under resource coefficient 0.25, 0.35 and 0,45 because the trend is similar under other coefficients. Note that the boxes

show the lower to upper 25% values of the data with whiskers showing $5-95$ percentile of the data, and the outliers are not shown in the figure. It can be seen from the figure that in general, offline optimal takes the least processing time, online optimal, OG, and SWMOA2 take more, and FlexOG uses the most time. This is mainly because offline optimal only needs to solve the optimisation problem once, while all other mechanisms need to solve the optimisation problem multiple times. FlexOG not only needs to solve the optimisation problem $|I| = 40$ times, but its optimisation problems also have more decision variables. Thus FlexOG is feasible for tasks where users can forecast their time constraints. Whereas for task requests that need immediate processing or task requests that come very frequently, the processing time of FlexOG would become an issue.

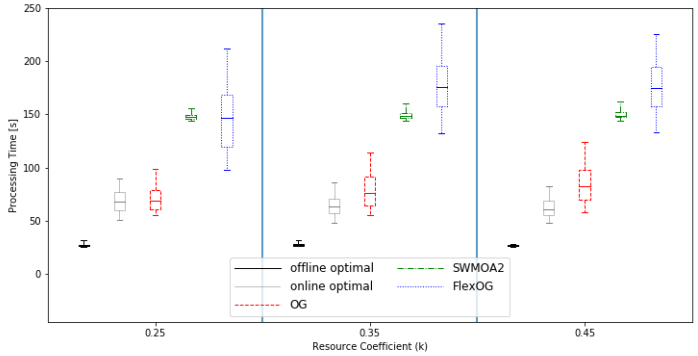

Fig. 5: Processing time of the mechanisms $((a, b) = (5, 10), F = 25, q = 0.1)$.

5 Conclusions

This paper formulates the RAFC problem as a constrained optimisation problem and proposes a novel truthful online mechanism for solving it. We made two key contributions. The first is that we extend price-based online mechanisms to our RAFC problem. The second is that we propose a truthful fog resource allocation mechanism called FlexOG, and we show its performance in terms of social welfare is significantly better than state-of-the-art mechanisms.

In the future, we plan to improve the scalability of FlexOG and to combine online mechanism design and machine learning to enhance social welfare further.

Acknowledgments. This research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence under Agreement Number W911NF-16-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

White paper: Fog computing and the internet of things: Extend the cloud to where the things are. https://www.cisco.com/c/dam/en_us/solutions/trends/iot/docs/computing-overview.pdf (2015), accessed: 2019-5-20

Aazam, M., Huh, E.N.: Fog computing micro datacenter based dynamic resource estimation and pricing model for iot. In: Proc. of 29th International Conference on AINA. pp. 687–694. IEEE (2015)

Bi, F., Stein, S., Gerding, E., Jennings, N., La Porta, T.: A truthful online mechanism for allocating fog computing resources. In: Proc. of the 18th International Conference on AAMAS. pp. 1829–1831 (2019)

Bonomi, F., Milito, R., Zhu, J., Addepalli, S.: Fog computing and its role in the internet of things. In: Proc. of the first MCC workshop. pp. 13–16. ACM (2012)

Cardellini, V., Grassi, V., Presti, F.L., Nardelli, M.: On qos-aware scheduling of data stream applications over fog computing infrastructures. In: ISCC. pp. 271–276. IEEE (2015)

Chawla, S., Devanur, N.R., Holroyd, A.E., Karlin, A.R., Martin, J.B., Sivan, B.: Stability of service under time-of-use pricing. In: Proc. of the 49th Annual ACM SIGACT Symposium on Theory of Computing. pp. 184–197. ACM (2017)

Doukas, C., Maglogiannis, I.: Bringing iot and cloud computing towards pervasive healthcare. In: Proc. of Sixth International Conference on IMIS. pp. 922–926. IEEE (2012)

Gu, Y., Chang, Z., Pan, M., Song, L., Han, Z.: Joint radio and computational resource allocation in iot fog computing. IEEE Transactions on Vehicular Technology 67(8), 7475–7484 (2018)

Hayakawa, K., Gerding, E.H., Stein, S., Shiga, T.: Price-based online mechanisms for settings with uncertain future procurement costs and multi-unit demand. In: Proc. of the 17th International Conference on AAMAS. pp. 309–317 (2018)

Lucier, B., Menache, I., Naor, J.S., Yaniv, J.: Efficient online scheduling for deadline-sensitive jobs. In: Proc. of the twenty-fifth annual ACM symposium on Parallelism in algorithms and architectures. pp. 305–314. ACM (2013)

Lueth, K.L.: State of the iot 2018: Number of iot devices now at 7b – market accelerating. https://iot-analytics.com/state-of-the-iot-update-q1-q2-2018-number-of-iot-devices-now-7b/ (August 2018), accessed: 2019-2-14

Nisan, N., Roughgarden, T., Tardos, E., Vazirani, V.V.: Algorithmic Game Theory. Cambridge University Press, New York, NY, USA (2007)

Sajid, A., Abbas, H., Saleem, K.: Cloud-assisted iot-based scada systems security: A review of the state of the art and future challenges. IEEE Access 4, 1375–1384 (2016)

Shi, W., Wu, C., Li, Z.: An online auction mechanism for dynamic virtual cluster provisioning in geo-distributed clouds. Proc. of IEEE Transactions on Parallel and Distributed Systems 28(3), 677–688 (2017)

Wang, C., Ma, W., Qin, T., Chen, X., Hu, X., Liu, T.Y.: Selling reserved instances in cloud computing. In: Proc. of the 24th IJCAI. pp. 224–230 (2015)

Wang, Q., Ren, K., Meng, X.: When cloud meets ebay: Towards effective pricing for cloud computing. In: Proc. of INFOCOM. pp. 936–944. IEEE (2012)

Zhang, X., Wu, C., Li, Z., Lau, F.C.M.: A truthful (1-ϵ)-optimal mechanism for on-demand cloud resource provisioning. In: Proc. of INFOCOM. pp. 1053–1061. IEEE (2015)

Zhu, Y., Fu, S.D., Liu, J., Cui, Y.: Truthful online auction toward maximized instance utilization in the cloud. IEEE/ACM Trans. Netw. 26(5), 2132–2145 (2018)