Reputation-Based Service Provisioning for Vehicular Fog Computing

Chaogang Tang[a], Huaming Wu[b,*]

[a]School of Computer Science and Technology, China University of Mining and Technology, 221116, Xuzhou, China
[b]Center for Applied Mathematics, Tianjin University, 300072, Tianjin, China

Abstract

The startling rise in smart vehicles stimulates the rapid development of new paradigms such as Social Internet of Vehicle (SIoV) and Vehicular Fog Computing (VFC). Trustworthiness has been regarded as a dominating issue in all the have-to-be-addressed issues in SIoV, and many reputation-based countermeasures have been adopted to solve the trustiness-related issues in IoV. However, little attention has been paid to the reputation of vehicles when they provision computational resources in VFC, which is worthy of further investigation since some fog vehicles pursue more revenues or fewer costs at the expense of delivering poor-quality computing services. Such selfish behaviors should be discouraged. In this paper, we put forward a reputation-based service provisioning scheme, and a reputation management scheme consisting of the decentralized reputation updating and global reputation synchronization in VFC, aiming to prevent the fog vehicles from delivering low-quality computing services by maximizing the accumulated reputation of all the serving fog vehicles in the optimization period. An online approach is adopted to handle the requests in a slot-by-slot way. The simulation results show its effectiveness and advantages when compared to other existing approaches.

Keywords:
Social internet of vehicle, Vehicular fog computing, Reputation, Computational resources, Fog vehicles

1. Introduction

There are around 1.5 billion vehicles in the world and the number of them may skyrocket to 2.8 billion by 2036 [1]. The rapid development of Internet of Vehicles (IoV) makes vehicles inter-connected and interactive with each other. IoV that integrates internal vehicle network, inter-vehicle network and mobile Internet can perceive information pertaining to vehicular state and the surroundings [2]. In the past few years, smart vehicles have made up a huge part of the market in the automobile industry. Apart from sensing ability, smart vehicles are further capable of storage, calculation and analysis, owing to various vehicle-mounted facilities. In this context, several newly emergent paradigms have gained widespread attention in both industry and academia, typically exemplified by SIoV [3] and VFC [4]. In particular, the former is devoted to the development of vehicular social abilities such as social communication and low-cost infotainment service provisioning, which are driven primarily by strong social instincts in people, even if they become vehicle travelers on the road [5]. The latter concentrates on computational service provisioning by leveraging the computing power of vehicles.

Among all the have-to-be-addressed issues in SIoV, trustworthiness is regarded as the most urgent one, because wireless links for information dissemination and multimedia content sharing are established among unacquainted vehicles/drivers in vehicular social networks. Many hostile behaviors may exist, attempting to damage vehicular social networks, e.g., some malicious vehicles may send bogus or improper messages uninterruptedly to undermine the trustiness of vehicles towards each other. Against this background, many reputation-based countermeasures have been adopted to solve the trustiness-related issues in IoV and acquired satisfactory effects such as [6, 7, 8, 9, 10]. On the other hand, however, researchers seldom apply reputation-based mechanisms to service provisioning in VFC.

The explosive growth in the Internet of Things (IoT) devices has led to staggering demands for computational resources, because these size-limited devices generate a huge amount of data but cannot process and analyze them by themselves owing to their limited computational capabilities [11]. Vehicles with idle computational resources can become a tempting choice for service provisioning in this context, and such an observation also stimulates the fast development of VFC. For instance, smart vehicles deploying services (e.g., related libraries and databases) can respond and serve the offloading requests from IoT devices (e.g., smartwatches and wearable health devices). Owing to the profit-driven factors, vehicles may display selfish behaviors, e.g., pursuing more revenues or fewer costs at the expense of delivering poor-quality computing services. Such irresponsible service delivery from selfish vehicles not only degrades the Quality of Service (QoS) and Quality of Experience (QoE) [12], but also exerts a negative influence on unselfish vehicles delivering high-quality services as consistently claimed.

Unfortunately, current works seldom consider how to prevent selfish vehicles from delivering low-quality services in VFC, although there is extensive literature that applies blockchain-based technologies to the security, privacy and trust issues that

*Corresponding author
Email address: whming@tju.edu.cn (Huaming Wu)

arise in VFC [13, 14, 15, 16]. Despite the merits of blockchain-based technologies, they do not perfectly suit the scenario of computation outsourcing and service provisioning in VFC, because the offloading requests from IoT devices usually have strict delay requirements. To tackle the above issue, we put forward a lightweight reputation-based mechanism to prevent selfish vehicles from delivering low-quality services in VFC. Particularly, the contributions of the paper are threefold, as given below:

- We propose a reputation-based service provisioning scheme in VFC, aiming to prevent fog vehicles from delivering low-quality computing services. The defined reputation has considered multiple impressions which come from both the IoT devices sending the requests and the local server that allocates the service requests.
- A reputation management scheme with the decentralized reputation updating and global reputation synchronization is put forward, which tries to prevent the fog vehicles from tampering with their own reputation values to mislead the service requestors. The reputation is updated based on multiple factors to ensure the fairness and objectivity in the paper.
- We try to maximize the accumulated reputation of all the serving fog vehicles in the optimization period. Owing to the feature of the sequential arrival of service requests, it is difficult to optimally solve it in an off-line way. We put forward an online approach to serve the service request sequentially. The simulation evaluation shows the advantages compared to other existing approaches.

The rest of this paper is arranged as follows. The literature review is conducted in Section 2. We give some preliminaries about the hierarchical end-fog-cloud system architecture where the reputation-based service provisioning strategy is applied in Section 3. The system model is introduced in Section 4 and the optimization problem is formulated in Section 5. Extensive simulation is conducted in Section 6, followed by the conclusion in Section 7.

2. Related Works

Reputation has been extensively studied in a wide range of fields [17, 18, 19, 20], including computer science, sociology, economics, and psychology. The study has already yielded extremely useful and fruitful results, and thus it is pretty difficult to survey all the representative works from various fields [21], owing to the limitation of space. Accordingly, we only pay attention to the recent works related to our reputation-based service provisioning in this paper.

Smart vehicles, which are empowered with computing capabilities, can shoulder more responsibilities in Vehicular Ad Hoc Networks (VANETs), e.g., task calculation, event evaluation and information forwarding [22, 23]. Atwa *et al.* [1] leveraged fog nodes to collect the trust evaluations from vehicles, and proposed a notion of Task-based Experience Reputation (TER), such that different types of tasks can be allocated to the most appropriate vehicles for execution based on the reputation values of the vehicles. Engoulou *et al.* [24] analyzed and summarized some locally perceived factors that can affect the behaviors of vehicles. Such parameters and factors usually include speed, acceleration, transmission range, direction, frequency of a DoS attack, and so on. Then, they strive to construct a decentralized reputation framework based on these parameters with the aim to identify malicious vehicles and prevent them from getting access to the internet of vehicles (IoV) network. A trust game was proposed in [7], wherein investors, trustworthy trustees, and untrustworthy trustees compete for assets. The assets are owned and supervised by a third party. The third party has been authorized to modify the reputation value of each participant.

Indeed, there are malicious vehicles in IoV which try to undermine the IoV network. For instance, some vehicular nodes will silently drop messages or packets. Such passive response to the information forwarding is also called a black-hole attack. Despite solutions applied to handling this black-hole attack, most of them are either centralized or dependent upon other nodes' opinions. Thus, Nabais *et al.* [8] put forward a decentralized reputation framework in the hope to detect and punish vehicular nodes with black-hole behaviors in the IoV network.

In addition to the black-hole attack in IoV and VANETs, there are also other malicious behaviors, e.g., vehicular nodes can transmit and forward useless and even wrong traffic information, so as to let other nodes make wrong decisions on route planning. To tackle this issue, a reputation-based algorithm is put forward in [25] to guide reliable route planning in IoV. The proposed algorithm can detect suspicious information.

Service caching has gained extensive attention recently for its advantages in improving the QoS of requested computing services hosted at vehicles and also QoE of service requestors, by caching related source codes and data beforehand at the edge server. However, it requires incentives to motivate the first requestor offloading the tasks, since as the first one, the requestor has to offload the task and pay for the task execution. To tackle this issue, an SDN-based cache-enabled VEC framework was proposed in [26]. In particular, they evaluated the contributions of each vehicular node by its reputation value. In the meanwhile, they leveraged Stackelberg game to model the incentive mechanism and proved the existence and uniqueness of Stackelberg equilibrium for the proposed game.

Reputation-based mechanisms are usually important for securing communications in IoV networks. However, security and efficiency can seldom be achieved at the same time. Therefore, Su *et al.* [27] proposed a centralized reputation management framework to identify malicious vehicles in IoV networks. They have made a bold step in exploring the feasibility of this framework as well as the potential threats to it. Simulation results have shown that their scheme can take effect more quickly and is better than current trust management schemes.

In spite of the explosive growth in the number of vehicles that can participate in Vehicular Crowd-Sensing (VCS), includ-

ing data analysis, information forwarding, and task calculation, not all vehicles on the roads are willing to contribute to VCS systems. In view of this, Yu *et al.* [9] put forward a reputation-based incentive approach in a VCS system in which both the utility of the cloud center and the participants are considered. In particular, they design a reputation-based reverse combination auction incentive method. For instance, the reputation of each participant is incorporated into the incentive approach to avoid maliciously raising bidding prices. Huang *et al.* [28] put forward a global trust evaluation framework to accurately eliminate malicious mobile data collectors (MDCs) for clean data collection environment. In particular, UAVs are adopted in their work to validate the data submitted by MDCs. Extensive experiments have revealed the advantages of their approach compared to existing works.

In VANETs, content request and delivery have become the norm, as the rapid development of smart vehicles [29, 30]. Zhu *et al.* [10] categorized the entities in VANETs into Parked Vehicles (PVs), Roadside Unit (RSU) and Moving Vehicles (MVs), respectively. RSU is responsible for delivering the content requested by MVs and PVs are leveraged for assisting RSU by storing the content in advance. Owing to the selfishness of PVs, a reputation-based scheme is used for identifying malicious PVs. Meanwhile, a two-layer auction game is utilized to model the cooperation among PVs, MVs, and RSUs. This approach can maximize the utility of RSUs as well as the throughput of content transmission to a great extent. Nevertheless, it needs to take a relatively long time to achieve mutually satisfactory results and it also does not consider the privacy of vehicles. Therefore, it is not suitable for our scenario in this paper.

The above literature does not consider how to evaluate fog vehicles when they are contributing the computational resources for serving the service requests. The majority of vehicles assume that the services can be delivered as claimed. However, malicious vehicles can deliver low-quality services for pursuing their own profits. To the best of our knowledge, this is the first effort to concentrate on computing service provisioning, considering the possibility of potentially malicious fog nodes delivering low-quality services in VFC.

3. Preliminaries

A hierarchical end-fog-cloud system model is presented in Fig. 1. The end layer mainly consists of size-limited and computing capacity-restricted IoT devices such as smartwatches, smart bracelets and wearable health devices. A great deal of data is gathered and plenty of tasks from the IoT devices are formed. When the IoT devices need to perform these tasks, they have to request the corresponding services from the fog layer. The fog layer, as the intermediate layer between the cloud layer and the end layer, consists of two entities. One is the roadside units (RSUs) deployed with the fog servers (FS). The other is the vehicles with idle computational resources and willing to contribute them in the form of service provisioning. We thus call them fog vehicles or fog nodes. The fog vehicles can directly respond to the service requests from the IoT devices. Meanwhile, they can also mitigate the pressure of the fog

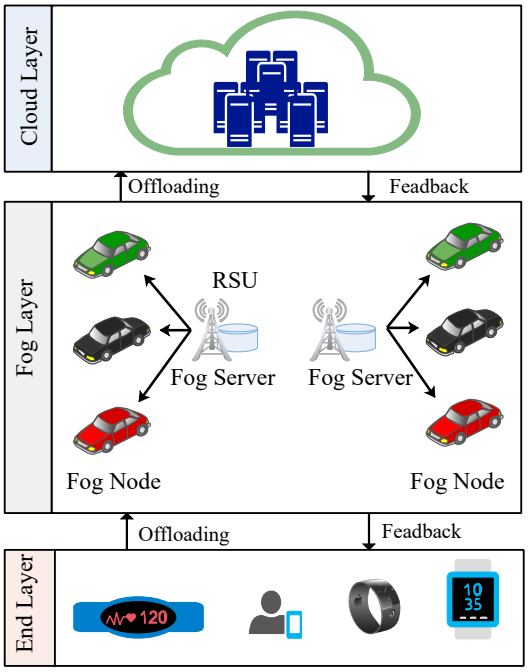


Figure 1: A hierarchical end-fog-cloud system architecture

server when the latter is overloaded. The cloud layer consists of a remote cloud center residing in the core/backbone network. Usually, the fog layer and cloud layer work cooperatively to provide computational resources. For example, the advantage of the cloud layer is that the powerful computing capabilities there can support even unlimited resource requests. However, it comes at the expense of a relatively long response delay. On the other hand, the fog layer can satisfy the strict delay requirement, and thus perfectly suits the time-critical service requests that the cloud layer does not suit.

In our system model, the two layers assume more responsibilities, e.g., to manage the reputation scheme proposed in this paper. Computing services can be provisioned in a decentralized and centralized way, respectively. For the former, the offloading links can be directly established after initial beacon exchanging between IoT devices and fog vehicles. However, this way cannot prevent selfish vehicles from delivering poor-quality services, and they can even tamper with their own reputation values to attract nearby IoT devices. On the other hand, the centralized service provisioning in this paper means that fog vehicles only accept the service requests designated by RSU covering them. This type of service provisioning may take time to determine which vehicles are suitable for the requests, thus yielding relatively long response latency. Nevertheless, the advantage is that the service execution at fog vehicles can be supervised by RSUs. Thus, it is difficult for fog vehicles to tamper with their reputation values.

Combining the merits of the two ways for service provisioning, we put forward a reputation-based service provisioning in VFC, aiming to prevent selfish fog vehicles from delivering low-quality services in this architecture. Particularly, the reputation for each fog vehicle cannot be manipulated deliberately

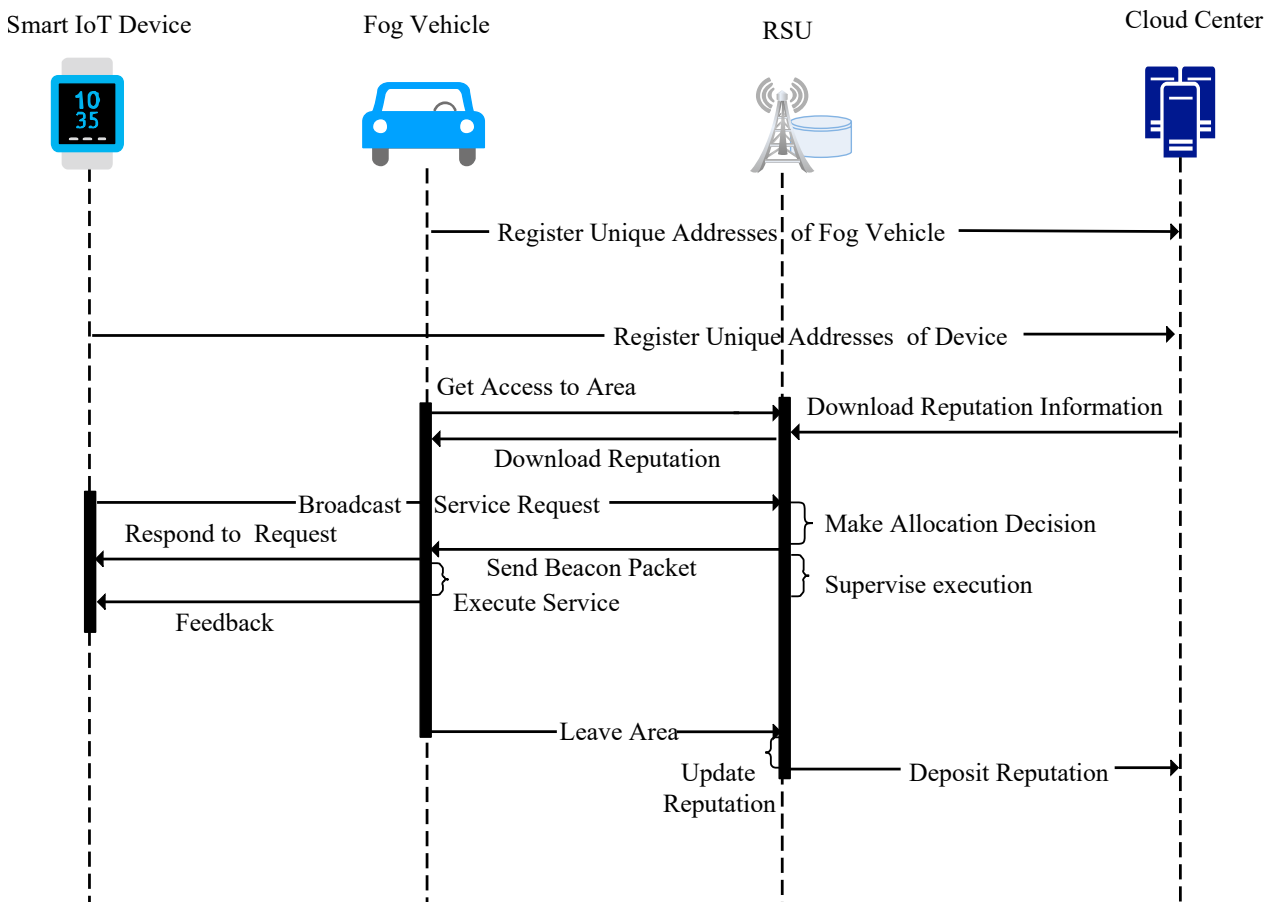


Figure 2: Sequence of interactions to display reputation management and service execution in end-fog-cloud architecture.

by themselves under the supervision of the RSUs. Each time a fog vehicle finishes serving the service request, it will obtain a score from RSU to assess its performance during service provisioning. Such a score has considered multiple factors such as the impressions from IoT devices and RSU, respectively, which will be elaborated later. Then the score is integrated into the reputation of the fog vehicle. The reputation of the fog vehicle is updated by the covering RSU every time it serves the service request. Generally, the fog vehicle delivering poor-quality computing service will be punished and one delivering high-quality computing service will be rewarded.

Note that one RSU together with the deployed fog server can act as a local server and manage the reputation for its covering fog vehicles. We assume that two neighboring RSUs have no overlaps, such that a fog vehicle can only be covered by at most one RSU at the same time. Owing to the high mobility of vehicles, they may enter or leave the area one RSU serves. Thus, when one fog vehicle leaves the area, RSU stops updating its reputation and sends the final value of reputation back to the cloud center for global reputation synchronization. On the other hand, when a fog vehicle gets access into the area, the serving RSU will retrieve the reputation of the vehicle from the cloud center and dynamically update it according to its performance. Specifically, Fig. 2 shows the sequence of interactions for reputation management and reputation-based service provisioning in the end-fog-cloud architecture. The presented interactions among the four entities can be sketched out as follows:

- Each fog vehicle and IoT device register to the cloud center with unique identifications (ID) such as Ethernet addresses. The fog vehicle is assigned with an initial reputation value after registration.
- Local server downloads the reputation value from the cloud center for each fog vehicle that gets access to its serving area. The reputation of all the fog vehicles is maintained by the local server. The local server will decide which fog vehicle to respond to the offloading request when the service request from IoT device arrives.
- IoT device sends the related data to the fog vehicle and waits for the execution. Meanwhile, the fog vehicle executes the service upon arrival of the service request.
- When the fog vehicle leaves the area, the local server writes back its reputation value immediately to the cloud center for global synchronization.

The above decentralized reputation management and service provisioning are feasible based on the following assumptions. First, the reputation stored in the cloud center cannot be

Table 1: Notations

Notation	Description
m	Number of fog vehicles
T	Number of discrete time slots
ϖ	The length of each time slot
$\mathcal{I}(t)$	The size of service-input data for the request at time slot t
$\mathcal{U}(t)$	The number of CPU cycles needed to accomplish the service at time slot t
$\mathcal{L}(t)$	Expected response delay from the IoT device at time slot t
$\mathcal{V}$	The set of fog vehicles
$\mathcal{P}_i$	The global reputation value of V_i
$\mathcal{P}_i(t)$	The current reputation of V_i in time slot t
$f_{i,min}$	The minimal processing frequency of V_i
$f_{i,max}$	The maximal processing frequency of V_i
B_{dv}	The wireless channel bandwidth
$H_{dv}(t)$	The channel gain between the device and fog vehicle
$P_{dv}(t)$	The transmission power from the device
$\delta^2(t)$	The noise power
$\lambda_i(t)$	The arrival rate of service requests at V_i in time slot t
$f_i(t)$	The processing frequency of V_i in time slot t
ϑ	The effectively switched capacitance coefficient
ς	The number of cycles to perform one service-input bit at V_i
β_t	The weight attached to the impression of the IoT devices in time slot t on V_j

forged, which is possible since various lightweight encryption techniques can be applied. Second, the local server (i.e., RSU with the deployed fog server) is trustworthy, which is also possible since they are usually deployed by the local government without selfish interest driving. Third, the fog vehicles are willing to accept the supervisor of the local server, which means their information such as speed, destination, waiting queue, and the amount of computational resources are known to the local server.

One pending issue during the above interaction needs to be addressed, i.e., how to schedule the service requests for the fog vehicles to guarantee that 1) the service requests are served without violating the constraints such as the energy consumption; 2) The fog vehicles are encouraged to deliver high-quality services, e.g., in terms of response latency, reliability or success rate. We will expatiate upon it in what follows.

4. System Model

The considered model in this paper consists of m fog vehicles and one local server. The optimization period $\mathcal{T}$ is divided into T discrete time slots, indexed by $\mathcal{T} = \{0, \cdots, T-1\}$, and each slot has a duration ϖ. ϖ is small enough so that there is only one service request from IoT devices arriving at the local server within ϖ. Denote the service request in time slot t by $\mathcal{S}(t) = (\mathcal{I}(t), \mathcal{U}(t), \mathcal{L}(t))$, where $\mathcal{I}(t)$ denotes the size of service-input data (e.g., the related processing codes) to be transmitted over the wireless channel, $\mathcal{U}(t)$ is the number of CPU cycles needed to accomplish the service, and $\mathcal{L}(t)$ is the expected response delay from the perspective of the IoT device. If the real response delay is not longer than $\mathcal{L}(t)$, then the IoT device sending $\mathcal{S}(t)$ is satisfied. Otherwise, the QoE begins to decline. Assume that the fog vehicles can provide computing services under the supervision of the local server, indexed by $\mathcal{V} = \{V_0, \cdots, V_{m-1}\}$. Each V_i can be represented by a vector $(\mathcal{P}_i, f_{i,min}, f_{i,max})$, where $\mathcal{P}_i$ is its global reputation value which can be downloaded from the cloud enter or the local server, $f_{i,min}$ and $f_{i,max}$ are the minimal and maximal processing frequencies of V_i, respectively. Usually, the processing frequency is an important factor to indicate the amount of computational resources leveraged for serving the service request. For instance, a larger processing frequency means more computational resources to be used for the service request, as well as more costs such as energy consumption. As a result, selfish fog vehicles pursue profit maximization by reducing costs, thus delivering low-quality services, e.g., in terms of response latency. Specifically, some notations of key variables to be used hereinafter are summarized in Table 1.

4.1. Delay Model

From the perspective of IoT devices, their QoE for the requested services mainly depends on one metric, i.e., the response delay. The devices are satisfied, if the execution result is returned before the expected response delay. Actually, the shorter the response delay, the more they are satisfied. In this paper, the response delay includes three parts in this paper and they are the transmission delay, the calculation delay and the returning delay.

4.1.1. Transmission Delay

The transmission delay $d_{trs}(t)$ for the requested service in time slot t denotes the time taken to transmit the service-input data from the IoT device to the fog vehicle, and it can be given as:

$$d_{trs}(t) = \frac{\mathcal{I}(t)}{r(t)}, \tag{1}$$

where $r(t)$ is the transmission rate for the service-input data in time slot t, given below [31]:

$$r(t) = B_{dv}(t) \log_2(1 + \frac{P_{dv}(t) H_{dv}(t)}{\delta^2(t)}), \tag{2}$$

where B_{dv}, $H_{dv}(t)$ and $P_{dv}(t)$ are the wireless channel bandwidth, the channel gain between the device and fog vehicle, and the transmission power from the device, respectively. $\delta^2(t)$ is the noise power.

4.1.2. Calculation Delay

The calculation delay $d_{clt}(t)$ for the requested service in time slot t denotes the time taken for the fog vehicle to accomplish

the calculation of the service. This delay includes the queueing delay and the execution delay. We denote the two parts by $d_q(t)$ and $d_e(t)$, respectively. To simplify the analysis, we assume that the arrival of service requests at each fog vehicle in time slot t follows a Poisson process with the arrival rate $\lambda_i(t)$, $i \in \{0, \cdots, m-1\}$. $\lambda_i(t)$ can be easily estimated based on the historical statistics. The service rate is $f_i(t)/\mathcal{U}(t)$, where $f_i(t)$ is the processing frequency of fog vehicle V_i in time slot t. Based on the M/M/1 queueing model, the average queuing delay in the waiting queue can be expressed as [32]:

$$d_q(t) = \frac{\lambda_i(t)\mathcal{U}^2(t)}{f_i(t)(f_i(t) - \lambda_i(t)\mathcal{U}(t))}. \tag{3}$$

The execution delay (i.e., the service time) $d_e(t)$ is given as:

$$d_e(t) = \frac{\mathcal{U}(t)}{f_i(t)}. \tag{4}$$

Thus, the calculation delay (i.e., the sojourn time) $d_{clt}(t)$ is [32]:

$$d_{clt}(t) = \frac{\lambda_i(t)\mathcal{U}^2(t)}{f_i(t)(f_i(t) - \lambda_i(t)\mathcal{U}(t))} + \frac{\mathcal{U}(t)}{f_i(t)}. \tag{5}$$

Meanwhile, the energy consumption $e_i(t)$ for accomplishing the service in time slot t at the fog node V_i is expressed as:

$$e_i(t) = \vartheta\varsigma\mathcal{U}(t)f_i^2(t), \tag{6}$$

where ϑ is the effectively switched capacitance coefficient, and ς is the number of cycles needed to perform one service-input bit at V_i. From this equation, it is obviously observed that the larger the processing frequency, the more the energy consumption. In other words, more computational resources to be used bring about more costs for the fog vehicles and fewer profits.

The returning delay denotes the time taken to send back the execution result to the IoT device. Similar to other works [33, 32], we also assume that the size of the execution result is much smaller than that of the service-input data. Hence, we have omitted the returning delay of the execution result. The response delay $d_{rsp}(t)$ for the service request in time slot t is:

$$d_{rsp}(t) = d_{trs}(t) + d_{clt}(t). \tag{7}$$

To depict the quality of service that one fog vehicle delivers, we give the following definitions:

Definition 1. ***Utility Function***. *A utility function is attached to a fog vehicle to represent its value to a service requestor. The corresponding utility value of fog vehicle V_i that serves the service request in time slot t is defined as a measurable gain of accomplishing the service, given as:*

$$\mathcal{F}_i(t) = \begin{cases} 1 & \text{if } d_{rsp}(t) \le \mathcal{L}(t) \\ \frac{\mathcal{F}_{i,max}(t) - \delta(d_{rsp}(t) - \mathcal{L}(t))}{\mathcal{F}_{i,max}(t)} & \text{if } \mathcal{L}(t) < d_{rsp}(t) < \mathcal{L}(t) + \mathcal{F}_{i,max}(t)/\delta \\ 0 & \text{if } d_{rsp}(t) \ge \mathcal{L}(t) + \mathcal{F}_{i,max}(t)/\delta \end{cases} \tag{8}$$

Note that $\mathcal{F}_{i,max}(t) > 0$, and $\delta > 0$. In particular, fog vehicle V_i will obtain a maximum utility if the service request is accomplished within its expectation $\mathcal{L}(t)$. Otherwise, the utility will decay linearly with a slope of δ until it equals to 0, as the response latency increases. In other words, the quality of service that the fog vehicle delivers declines as the value of its utility function decreases. Generally, the larger the utility value, the higher the quality of the delivered service.

Definition 2. ***Expected Utility***. *The expected utility $\mathcal{E}_i(t)$ is defined as the average utility value of the fog vehicle V_i for accomplishing the service requests coming from the last K time slots, given as:*

$$\mathcal{E}_i(t) = \frac{1}{K}\sum_{j=t-K}^{t-1}\mathcal{F}_i(j). \tag{9}$$

Definition 3. ***Utility Difference***. *The utility difference of the fog vehicle V_i, denoted by $\triangle\mathcal{F}_i(t)$, is defined as the deviation of $\mathcal{F}_i(t)$ from $\mathcal{E}_i(t)$, given as:*

$$\triangle\,\mathcal{F}_i(t) = \mathcal{F}_i(t) - \mathcal{E}_i(t). \tag{10}$$

4.2. Reputation Model

Reputation-based strategies have played an important role in making service providers constantly provide high-quality services. For example, the reputation usually represents the accountability for maintaining their service levels and can thus affect the offloading decisions to a great extent in VFC. A typcial description about reputation is that it represents an opinion that people have towards someone or something based on the observed behaviors or displayed character. In this paper, we need to think about what the opinion about a fog vehicle is like. Generally speaking, a computing service is usually provisioned on demand by a fog vehicle in VFC, and thus the opinion about the fog vehicle from the outside is typically built upon its performance during service provisioning. Particularly, if a service request is served better than what it has been expected (e.g., in terms of response latency), it will bring more value or utility to the requestors and even irrelevant ones who will in turn have a better opinion about the fog vehicle. Therefore, we can actually use utility difference to indicate the opinion about the fog vehicle. Then, we give the following definitions in what follows.

Definition 4. ***Impression***. *The impression $IMP_i(t)$, which denotes the opinion of the IoT device that sends $\mathcal{S}(t)$ towards the fog vehicle V_i that undertakes $\mathcal{S}(t)$, is equal to the utility difference, given as:*

$$IMP_i(t) = \triangle\mathcal{F}_i(t). \tag{11}$$

Given the above definition, several interesting observations can be made as follows. First, $IMP_i(t)$ is a real number ranging from -1 to 1. Second, $IMP_i(t)$ will increase if the quality of delivered service becomes higher, and $IMP_i(t)$ will decrease if the quality of delivered service becomes lower. Third, $IMP_i(t)$ is 0 if the current utility value equals the average one.

Definition 5. ***Service Offloading Decision***. *Service offloading in this paper refers to the allocation of service requests in VFC by the local server. Let φ_i^t be a binary variable to indicate whether the service request in time slot t is allocated to fog vehicle V_i, $i \in \{0, \cdots, m-1\}$. $\varphi_i^t = 1$, if the request is distributed to V_i; and 0, otherwise. Define $\boldsymbol{\varphi}_i = \{\varphi_i^0, \cdots, \varphi_i^{T-1}\}$ as the offloading decisions of V_i for all the service requests along the timeline $\mathcal{T}$. Define $\boldsymbol{\varphi} = \{\boldsymbol{\varphi}_0, \cdots, \boldsymbol{\varphi}_{m-1}\}$ as the offloading decisions of all the vehicles for the service requests along the timeline $\mathcal{T}$.*

It shall be noted that a fog vehicle may enter or leave the serving area of the local server in the middle of the optimization period, owing to its high mobility. For instance, one fog node V_j enters the area after the time slot k. In this case, for the sake of easy expression and discussion, we assume that the binary variables of this vehicle for these nonexistent time slots still exist, i.e., $\{\varphi_j^0, \cdots, \varphi_j^k\}$, and just let them equal zero instead.

Definition 6. ***Single Reputation***. *The single reputation of a fog vehicle V_j, denoted by $R_j(t)$, is an individual impression of the current IoT device sending the service request in time slot t:*

$$R_j(t) = \varphi_j^t \cdot IMP_j(t). \tag{12}$$

Based on the above definition, it is obvious that for an arbitrary time slot, there is only one nonzero value of the single reputation, since we have assumed that there is only one service request within each time slot. Particularly, $R_j(t)$ ranges from -1 to 1. The case $R_j(t) < 0$ indicates that the current service provisioning is worse and deviates a lot from the expected utility. Each time the fog node serves the service request, the single reputation should be updated. In addition, we have the following definition.

Definition 7. ***Current Reputation***. *The current reputation of a fog vehicle V_j in time slot t, denoted by $\mathcal{P}_j(t)$, is an overall reputation that comprehensively evaluates the performance when V_j provision computing services so far, and it is iteratively defined as:*

$$\mathcal{P}_j(t) = \mathcal{P}_j(t-1) + \beta_t \cdot R_j(t), \tag{13}$$

where β_t ($0 < \beta_t < 1$) is the weight attached to the impression of the IoT devices sending the service request in time slot t on the fog vehicle V_j.

Considering the fact that one fog node in VFC may serve the service requests multiple times, and at the same time, one IoT device may send service requests multiple times in the optimization period, the above weights $\beta_t (t \in \{0, \cdots, T-1\})$ can be leveraged for multiple purposes. First, if one service request allocated to V_j is not urgent enough in terms of response latency, the priority level of the service request is supposed to be relatively low; on the other hand, if one service request allocated to V_j is very urgent from the perspective of IoT device, then the value earned by serving it is supposed to be more than serving other requests which are not as urgent as it. The local server, depending upon different situations, can achieve the above purpose by adjusting the corresponding weight β_t. Second, if the IoT device sending service request $\mathcal{S}(t)$ is malicious, the local server can reduce the value earned by serving $\mathcal{S}(t)$, e.g., by means of lowering the weight, although the identification of malicious IoT devices is beyond the scope of this paper. Last but not least, from the perspective of fog vehicles, they may show different preferences towards the service requests in the optimization period, and we can also adjust the weights for this purpose. It shall be noted that all the impressions can hypothetically have the same weight, since it is not our focus in this paper.

Eq. (13) can also be regarded as the update of the reputation value. Usually, there are two ways to update the reputation values of fog vehicles. One is that the reputation values of all the fog vehicles remain unchanged during the optimization period and are updated only at the end of the optimization period. Such an update avoids frequent interactions between the local server and fog vehicles, thus reducing the great pressure on the front-haul links between them. However, this way cannot prevent selfish vehicles from delivering low-quality computing services. In this paper, we tend to distribute the service requests to the fog vehicles based on their reputation values which will be elaborated later. Assume that one fog vehicle V_j which has the highest reputation value gets access to the serving area of the local server, and V_j wants to pursue more profits in the incoming optimization period by delivering low-quality services. Since the service requests are distributed based on their reputations and the reputation values remain unchanged during the timeline, V_j with the highest reputation value can easily be assigned with more requests than other fog vehicles. Unfortunately, there are no efficient countermeasures to cope with this unfair requests distribution, let alone the measures adopted to prevent this selfish V_j.

The other way is to update the reputation values for all the fog vehicles every time the service request $\mathcal{S}(t)$ is served. Although this way undoubtedly incurs frequent interactions between the local server and vehicle fogs, it can prevent the fog vehicle say V_j from delivering low-quality services. For instance, $R_j(t)$ will gradually become smaller with the increasing number of time slots, since $R_j(t)$ has been nonpositive along the time slots. The decreasing reputation value will hinder $R_j(t)$ from obtaining more service requests. Based on this analysis, we adopt the second way to update the reputation values for all the fog vehicles in the optimization period.

In the meanwhile, the local server needs to write the reputation back to the cloud center based on the following two cases. One is that fog vehicle V_j leaves the serving area of the local server in the middle of the optimization period. In this case, the local server writes back the updated reputation value immediately to the cloud center for global synchronization, such that another local server can download it when V_j gets access to its serving area. The other case is that the optimization period is over. In this case, the local server also writes back the updated reputation values immediately to the cloud center for global synchronization. For those vehicles which still want to stay and make a contribution, they still need to download their own reputation values again either from the local server or the

cloud center.

5. Problem Formulation

The main goal of the reputation-based service provisioning in VFC is to prevent fog vehicles from delivering low-quality computing services, when the vehicles are serving the requests. In the meanwhile, the constraints should be considered such as the energy consumption and the processing frequencies. Intuitively, the fog vehicle with a large value of current reputation can, as always, deliver high-quality computing services. Therefore, we strive to maximize the reputation value of each fog vehicle along the time slots. Specifically, the optimization problem in this paper can be formulated as follows:

$$(\mathcal{Q}) \quad \max_{\boldsymbol{\varphi}} \sum_{i=0}^{m-1} \sum_{t=0}^{T-1} \mathcal{P}_i(t)$$

$$\text{s.t.} \quad \sum_{i=0}^{m-1} \varphi_i^t \leq 1 \quad \forall t \in \{0, \cdots, T-1\} \tag{14}$$

$$e_i(t) \leq e_{i,max} \quad \forall i \in \{0, \cdots, m-1\}\ \forall t \in \{0, \cdots, T-1\} \tag{15}$$

$$f_{i,min} \leq f_i(t) \leq f_{i,max}\ \forall i \in \{0, \cdots, m-1\} \forall t \in \{0, \cdots, T-1\} \tag{16}$$

$$\varphi_i^t \in \{0, 1\} \quad \forall i \in \{0, \cdots, m-1\}\ \forall t \in \{0, \cdots, T-1\} \tag{17}$$

where the constraint (14) guarantees that a service request from any time slot should be served by at most one fog vehicle. An extreme case is that none of the fog vehicles are qualified for the service request, e.g., lack of enough computational resources. In such a case, the local server will undertake the computation of the service. Although the fog vehicles are encouraged to deliver high-quality services, we allow for the case that the fog vehicles reserve the computational resources and energy supply for an emergency. We can achieve this goal by using the constraints (16) and (17).

Exhaustive search over the potential solution space is prohibitively costly, since it takes the exponential time to determine the best allocation scheme for the service requests in the entire optimization period. Even worse, to optimally solve problem $\mathcal{Q}$ requires complete information including future information about service requests, which can be only realized in an off-line way. However, it does not suit our scenario in this paper, since the service requests arrive sequentially, and they are supposed to be handled right upon its arrival, instead of being handled in batches. Meanwhile, it is pretty hard to predict service requests in the future time slots, which indeed necessitates an online approach to solve the optimization problem.

We notice that the above problem $\mathcal{Q}$ is equal to the following problem $\mathcal{K}$ after a straightforward transformation:

$$(\mathcal{K}) \quad \max_{\boldsymbol{\varphi}} \sum_{t=0}^{T-1} \sum_{i=0}^{m-1} \mathcal{P}_i(t). \tag{18}$$

Note that we do not list the constraints any longer since they are the same as the problem $\mathcal{Q}$. This problem indicates that we can optimize the reputation values of all the fog vehicles within each time slot at the beginning, and then we maximize these values along the time slots. On one hand, we have transformed this optimization problem spanning the entire optimization period into a series of per-slot deterministic optimization subproblems, such that the service request can be allocated in a slot-by-slot way. On the other hand, this problem can be solved by the greedy approach. The locally optimal solutions can constitute the globally optimal solution in this problem, which can be easily proven by reductio. Furthermore, it seems easy to find the optimal solution to the above problem with the time complexity of $O(TM)$.

However, this problem becomes very complicated if we take into account the selfishness of vehicles, since we do not know, a priori, whether the fog vehicle will deliver a high-quality service if the service request is distributed to it. Accordingly, there is a great deal of uncertainty during service provisioning. To tackle this issue, we put forward a two-phase-based service request allocation scheme within each time, aiming to answer the following two questions: 1) which fog vehicle is the most qualified for the service request among all the fog nodes; 2) How many computational resources are allocated to the service request for the chosen fog node?

5.1. Fog Node Selection

Our goal in this paper is to encourage fog vehicles to always deliver high-quality computing services, so the determination of fog nodes should display the advantages of those vehicles which deliver high-quality services as consistently claimed. Therefore, we tend to designate the fog node with the highest reputation value to respond to the service request. One may argue that it may not be fair enough, since the service requests arriving in sequence in the optimization period may be allocated to the same fog node with the highest reputation value, thus dampening the enthusiasm of other vehicles willing to contribute computational resources. We should admit that such an extreme case indeed exists. For instance, if one fog vehicle has the highest reputation value and much more computational resources than other fog nodes, this fog node may earn more service requests.

However, the proposed reputation-based service provisioning has taken into account the fairness issue to a great extent, based on the following reasons. First, the utility attached to the fog vehicle is updated each time the fog node finishes serving the service request. Meanwhile, the reputation based on the utility value is recorded per time slot. If a fog vehicle with high original reputation value has been delivering low-quality computing services in the past K time slots, its reputation will increasingly decline. On the other hand, if a fog vehicle with low original reputation value has been delivering high-quality computing services in the past K time slots, its reputation will increasingly rise. Second, the optimization period is short enough so that the reputation values of the fog vehicles can be updated timely. In this context, each fog vehicle has an opportunity to respond to the service request. Last but not least, in response

Algorithm 1: Procedure for Fog Node Determination and Resource Distribution (PDD)

Input: $\mathcal{T}$, K, V, ϑ, ς, $\mathcal{S}_t$, m, δ, $\boldsymbol{B_{dv}}$, $\boldsymbol{P_{dv}}$, $\boldsymbol{H_{dv}}$, $\boldsymbol{\beta}$
Output: The reputation sum of all the fog nodes

Gather status information from fog vehicles;
Download reputation values from cloud center;
Sort fog nodes in descending order of reputation values;
Initialize a list $\mathcal{H}$ to store the already sorted fog nodes;
$Sum = 0$;
for each time slot t in $\mathcal{T}$ **do**
  Get the fog vehicle V_i from the list $\mathcal{H}$ in sequence;
  Calculate $f_i^*(t)$ using Eq. (20);
  if $f_i^*(t) > f_{i,max}$ **then**
    `// V_i is not qualified`
    Repeat steps 6-7;
  else
    Calculate $e_i^*(t)$ based on Eq. (6);
    if $e_i^*(t) > e_{i,max}$ **then**
      `// V_i is not qualified`
      Repeat steps 6-7;
    else
      Disseminate the beacon information to V_i;
      Designate V_i in response to $\mathcal{S}(t)$;
      Supervise the process of serving the requests;
      Calculate and update its reputation value $\mathcal{P}_i$;
    end
    if V_j leave area **then**
      Update the reputation of V_j, i.e., $\mathcal{P}_j$;
      Write $\mathcal{P}_j$ back to the cloud center;
    end
  end
  Record the service offloading decision on $\mathcal{S}_t$;
  Form per-slot offloading decisions $\{\varphi_0^t, \cdots, \varphi_{m-1}^t\}$;
  $S = 0$;
  for each fog vehicle V_i in $\mathcal{V}$ **do**
    Calculate $\mathcal{P}_i(t)$ based on Eq. (13);
    $S += \mathcal{P}_i(t)$;
  end
  $Sum += S$;
end
Return Sum;

to the issue of all the service requests that are allocated to the same fog node, we admit such an extreme case, but there is not too much to worry about. Each service request has an expected response delay. If all the services are allocated to the same fog node, the response latency will increase for some service requests, owing to the increasing queueing delay and the uneven computational resources distribution. Thus, the utility value of the vehicle may decline due to the constraint violation (see Eq. (8)), which eventually leads to the decline of the reputation. As a result, the above situation will give other vehicles opportunities.

5.2. Computational Resources Distribution

When the fog vehicle that serves the service request is determined, another issue comes naturally, i.e., how many computational resources are supposed to be distributed to the service request such that a good reputation can be maintained as always but there are no more costs incurred by service provisioning such as the energy consumption.

Note that the utility value of one fog node only depends upon its response delay and the utility value will not increase anymore as long as the response delay is shorter than the expected response delay. In other words, it is meaningless to shorten the response delay after the real response delay reaches the expected response delay. We can thus calculate the critical value of the amount of computational resources when the response delay is equal to the expected response delay. Combining Eq. (7) and Eq. (8), it can be inferred that this critical value for the computational resources is:

$$f_i^*(t) = \lambda_i(t)\mathcal{U}(t) + \frac{r(t)\mathcal{U}(t)}{r(t)\mathcal{L}(t) - \mathcal{I}(t)}, \tag{19}$$

which indicates that the best result can be achieved when the processing frequency of the fog vehicle equals $f_i^*(t)$ and there are no more energy consumption incurred for extra computational resources distribution. Accordingly, the procedure for fog node determination and computing resource distribution, denoted by PDD, is shown in Alg. 1. Before the optimization period begins, the algorithm PDD should conduct some initializations. For instance, the local server needs to gather the information on the fog vehicles dwelling on its serving area. Such information usually includes the amount of computational resources, the destination, the dwelling time and even the reputation values, provided that these vehicles have obtained their reputation somehow. Then the local server downloads the reputation values of these vehicles from the authorized cloud center. It shall be noted that, the local server can check whether the two kinds of reputation values are equal, so as to tag those potentially selfish vehicles. For instance, if the reputation value gathered from one vehicle (e.g., V_i) is higher than that from the cloud center, the local server can consider V_i as a potentially selfish vehicle. The local server sorts the vehicles in descending order of reputation values and then stores them by a list $\mathcal{H}$ for subsequent operations.

For the service request in each time slot t, the local server gets the first fog vehicle (e.g., V_0) in $\mathcal{H}$ and checks whether

it is qualified for the request by the following several validations. First, the local server calculates the critical value of the processing frequency $f_0^*(t)$ based on Eq. (19). If this critical value is larger than the threshold of the processing frequency (i.e., $f_{0,max}$), we regard this vehicle as an unqualified fog node for the request. Otherwise, given $f_0^*(t)$, we calculate $e_0^*(t)$ based on Eq. (6) to check whether the critical energy consumption exceeds the threshold of energy consumption (i.e., $e_{0,max}$). If $e_0^*(t) > e_{0,max}$, this fog vehicle is still considered to be an unqualified fog node. In this case, the local server will try another fog vehicle by getting the vehicle just behind V_0 in $\mathcal{H}$ and repeat the above validations.

The fog vehicle passing the validation process will be noticed by the local server, e.g., by beacon packet dissemination (line 15). This vehicle will serve the request by allocating the computational resources to it under the supervision of the local server. After completing the service request, its reputation is updated and recorded at the local server. In the meanwhile, if one vehicle, say V_j, leaves the serving area, the local server will update the reputation and write it back to the cloud center. Actually, this procedure can be adopted to solve the problem $\mathcal{K}$. In particular, the local server records the service offloading decision in each time slot $\varphi_i^t(i \in \{0, \cdots, m-1\})$ and calculate the sum of the reputation values of all the fog nodes within this time slot (lines 28-32). Then, the total reputation of all the fog nodes in the entire optimization period can be obtained (line 33).

Table 2: Parameter Settings

Parameter	Value	Parameter	Value
T	[100,300]	m	[20,30]
K	[1, 100]	δ	[1, 5]
$f_{i,min}$	[1000,1500]	$f_{i,max}$	[2000,2500]
$\mathcal{I}(t)$	[1,50]	$\mathcal{U}(t)$	[40, 70]
$\mathcal{L}(t)$	[0,1]	$\mathcal{P}_i$	[0,1]
$\mathcal{F}_{i,max}$	[0,1]	β_t	[0,1]

6. Simulation Evaluation

In this section, we validate the proposed reputation-based service provisioning scheme via extensive simulation under different scenarios.

6.1. Experimental Settings

For the main parameters involved in the simulation, we initialize them as follows. The number of fog vehicles varies from 20 to 30, and the number of time slots ranges from 100 to 300. Each vehicle is assigned with an initial reputation value ranging from 0 to 1. Specifically, these parameters are listed in Tab. 2. All the simulation is run on a notebook with a 1.8 GHz Intel(R) Core(TM) i5-8250U CPU, 8 GB of RAM, Microsoft Windows 10 Operating System, Python 3.7.

Our simulation includes two parts. On one hand, we will validate the efficiency and effectiveness of our approach in terms of the effects of the involved parameters upon the performance, given different scenarios. Such parameters usually include the number of fog vehicles, the slope of the utility function and so on. On the other hand, we will compare our approach with the following two approaches:

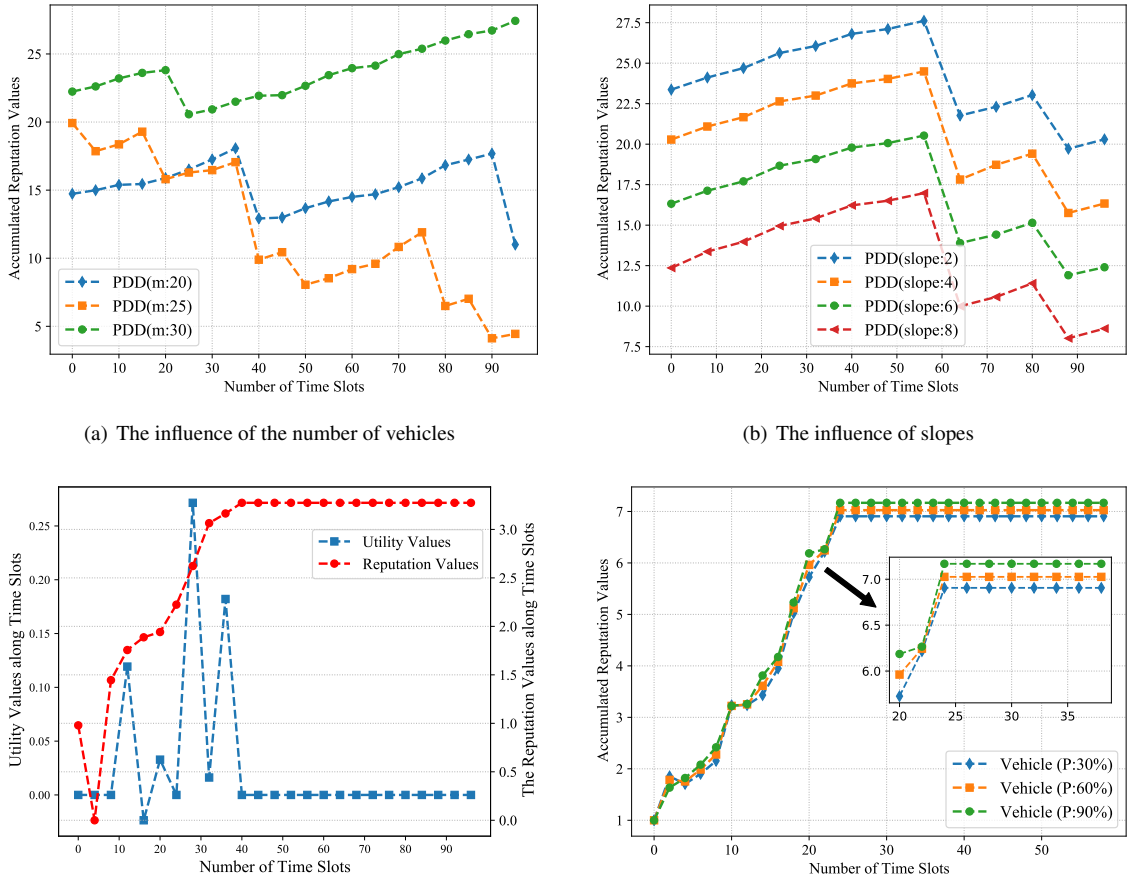
- Task-based Experience Reputation (TER) [1]: This approach assigns different tasks with different weights to indicate their emergency as well as importance. The reputation is calculated using a weighted mean of previous reputation values.
- Iterative Reputation Management (IRM) [27]: This approach iteratively manages the reputation for a vehicle, which takes into consideration the accumulated reputation at each iteration by a constant ratio.

6.2. Parameters Evaluation

The first set of experiments is conducted to validate the efficiency and effectiveness of the proposed reputation-based service approach. The simulation results are shown in Fig. 3. The influence of the number of vehicles upon the performance is depicted in Fig. 3(a). It is obvious that the number of vehicles affects the performance of PDD greatly in terms of the accumulated reputation. Particularly, given the number of service requests, the accumulated reputation values for all the fog vehicles will become larger generally. For instance, when the number of fog vehicles is 30, the overall reputation values are much better than the two cases when the number of fog vehicles is 20 and 25.

It is interesting that the performance of the case when the number of fog vehicles is 20 is better than the case when the number of fog vehicles is 25 most of the time. It is mainly due to the fact that all the data including the service requests and fog vehicles in the simulation is generated randomly. Although the service requests are the same, the fog vehicles are all different in the three cases. As a result, the above situation may occur in the simulation. On another hand, as far as one case (e.g., $m = 20, 25$, or 30) is concerned, the reputation values may decrease as the number of time slots increases. For instance, the reputation values decrease when the number of time slots is 20 for the case with $m = 20$ and 25 for the case with $m = 30$. Recall that our purpose is to prevent the fog vehicles from delivering low-quality services by distributing the service requests based on the reputation values of fog vehicles. If selfish vehicles deliver low-quality services, the reputation will decline as the number of time slots increases. Such punishment will hinder the fog vehicles from obtaining more service requests.

We investigate the influence of the slopes defined in Eq. (8) upon the performance of proposed strategy. The simulation results are shown in Fig. 3(b). Based on the definition of the utility function for a fog vehicle, the larger the slope, the faster the utility value declines. As a result, it is understandable that the performance of the proposed strategy is better than others when the corresponding slope is smaller than others, which can be easily observed from the figure. Meanwhile, we can also find that the accumulated reputation values fluctuate a lot. For instance, when the number of time slots increases from 55 to 62,



(a) The influence of the number of vehicles

(b) The influence of slopes

(c) The utility values versus the reputation values

(d) Reputation versus the service provisioning

Figure 3: The validation of the influence of involved parameters upon the performance of the approach

the accumulated values decrease sharply for all the four cases. Such a fluctuation has the same reason as shown in Fig. 3(a). The fog vehicles assigned with service requests will be punished if they deliver the poor-quality services along the time slots.

In the next, we investigate the relationships between the utility values and the reputation values of the fog vehicles. The results are shown in Fig. 3(c). Specifically, we have taken one from thirty vehicles as the observation object. Several conclusions can be drawn from this figure. First, the reputation of one fog vehicle goes up and down, depending on its performance in each time slot. Second, a sharp rise or decline in the utility values in the current time slot does not bring about a similar fluctuation in the reputation values in the current time slot. The reason is that the definition for reputation is based on the utility difference (see definition 3), which adopts the expected utility (i.e., the average utility value of the fog vehicle coming from the last K time slots) to prevent a sudden rise or decline in the reputation value. This countermeasure can efficiently handle the situation where a fog node continuously delivering poor-quality computing services wants to rise its reputation rapidly by delivering high-quality computing services several times.

We have conducted another set of experiments to validate the relationships between the reputations and the quality of delivered computing services. The simulation results are shown in Fig. 3(d). In the simulation, we assume that one vehicle with a good reputation will provide high-quality services along the time slots with the probabilities of 30%, 60%, and 90%, respectively. All the settings are the same except the probability of provisioning high-quality computing services. Obviously, the higher the probability that the fog vehicle provisions high-quality services, the larger the reputation, and vice versa.

6.3. Performance Comparison

We first evaluate our approach compared to the approach proposed in [1]. Herein, we call the approach TER directly for the sake of easy discussion and reference. As mentioned at the beginning, TER incorporates the emergency and importance into the reputation of the tasks, e.g., by assigning them with different weights. The reputation is a weighted mean of previous reputation values. Actually, we also consider the emergency and importance of the tasks in our reputation model by using the weights $\beta_t, (t \in \{0, \cdots, T-1\})$. Furthermore, we adopt utility difference-based impressions to define the reputation instead of a simple weighted mean of previous reputation values. The simulation results are shown in Fig. 4 and Fig. 5, respectively.

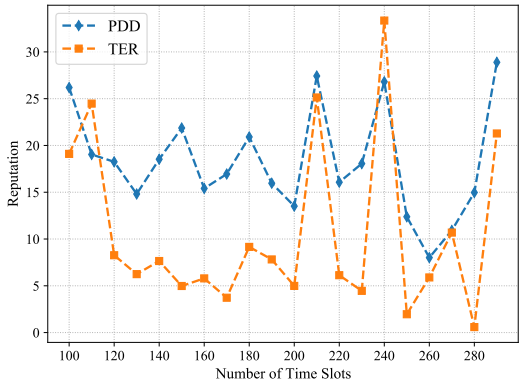


Figure 4: Performance comparison with TER w.r.t. reputation

Fig. 4 indicates that our reputation values are generally higher than TER along the time slots. The following reason can account for this result. The definition for reputation in this paper is relatively smooth and steady compared to TER. As illustrated above, it can prevent selfish vehicles which continuously deliver poor-quality computing services from rising their reputation rapidly by delivering high-quality computing services several times. On the other hand, it can also prevent some vehicles with good reputations from lowering their reputation rapidly by delivering poor-quality computing services several times. In contrast, TER is unable to achieve such a purpose. Hence, our approach is better than TER with regards to (w.r.t.) the achieved reputation values.

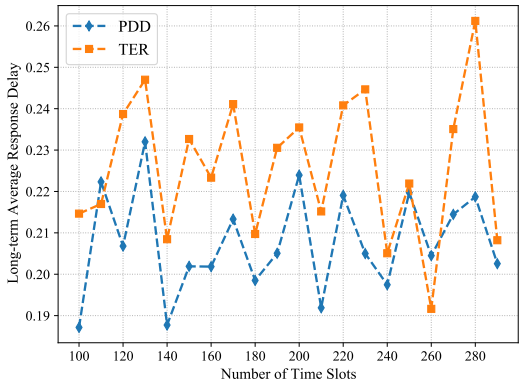


Figure 5: Performance comparison with TER w.r.t. the long-term average response delay

In addition, we have evaluated how they respectively function in preventing selfish vehicles from delivering low-quality services. The results are shown in Fig. 5, where the y-coordinate represents the long-term average response delay. In this simulation, we vary the number of time slots from 100 to 300, which means that there are hundreds of service requests waiting to be handled. From the figure, we can observe that our approach is much better than TER in terms of the long-term average response delay. In other words, our reputation-based service provisioning can better prevent selfish vehicles and encourage them to deliver the claimed computing service so as to maintain a good reputation.

In the next, we have conducted another set of experiments to evaluate our approach compared to the approach proposed in [27]. Similarly, we call the approach IRM directly for the sake of easy discussion and reference. IRM iteratively updates the reputation for a vehicle, which considers not only the accumulated reputation but also the impression from other vehicles. To suit our scenario, we need to tailor the reputation update as follows.

$$\mathcal{P}_j(t) = \mathcal{P}_j(t-1) + R_j(t)\rho_j(t)x_j(t), \tag{20}$$

where $\rho_j(t)$ is the validated result of vehicle V_j in time slot t, and $x_j(t)$ denotes the trustworthiness of the local server towards the fog vehicle V_j and the trustworthiness can be regarded as a probability ranging from 0 to 1. $\rho_j(t)$ is a binary variable. As mentioned earlier, the local server will check the two reputation values of which one is downloaded from the cloud center, and the other comes from the vehicle during the beacon packet dissemination. $\rho_j(t) = 1$, if the two values equal; and 0, otherwise.

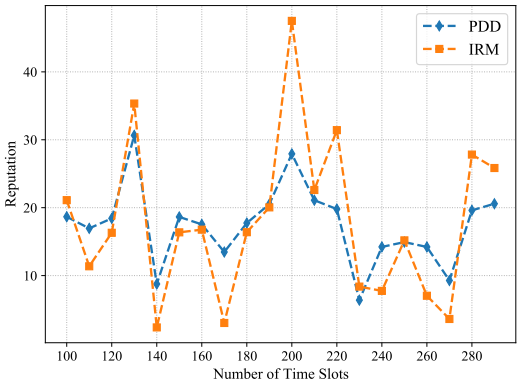


Figure 6: Performance comparison with IRM w.r.t. reputation

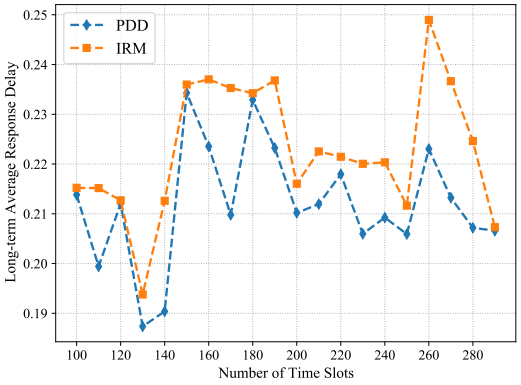


Figure 7: Performance comparison with IRM w.r.t. the long-term average response delay

The simulation results are shown in Fig. 6 and Fig. 7, respectively. The evaluation is similar to the evaluation of TER. Note that we have compared our approach with TER and IRM, separately in the simulation, because during the comparison between TER and PDD, we have assigned different weights to the impression of the IoT devices sending the service request in each time slot, and for the comparison between IRM and

PDD, all the weights are assumed to be the same. From Fig. 6, we can observe that there are no obvious relationships between IRM and PDD w.r.t. the reputation values. The two different ways to update the reputation make their reputation values respectively fluctuate a lot. Furthermore, the reputation values of TER sometimes are larger than those of PDD, and sometimes aren't. However, as far as the long-term average response delay is concerned, our approach is obviously better than IRM, which can be easily observed from Fig. 7. As the data on service requests is generated randomly in each optimize period, the resulting reputation values often go up and down, which is acceptable in our opinion.

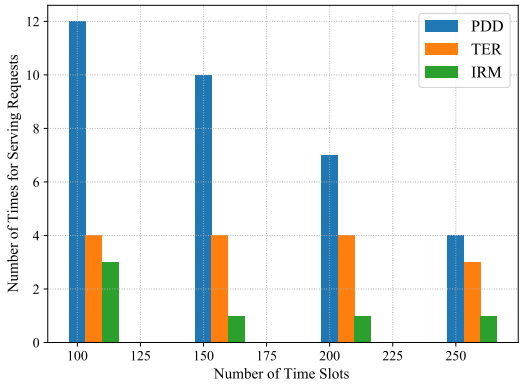


Figure 8: Performance comparison with selfish vehicle delivering poor-quality services

We have conducted the last set of experiments to evaluate the performance of our approach in comparison with TER and IRM. The goal is to check their performance in preventing fog vehicles from delivering the low-quality services. In particular, we still choose one from thirty fog vehicles as the observation object. Along the time slots, the probability that this fog vehicle delivers low-quality services increasingly rises in a random way. Intuitively, the higher the probability that the fog vehicle delivers low-quality services, the lower the reputation values. However, due to the difference in reputation updates, the reputation values cannot be compared directly. We then evaluate the number of times that this fog vehicle serves the service requests in each optimization period. Generally, the selfish vehicle will obtain gradually reduced service requests as time goes by. The corresponding simulation results are shown in Fig. 8, where the y-coordinate represents the number of times for the fog vehicle serving the service requests. The results have validated our expectations. Moreover, our approach can achieve better performance than the other two approaches in terms of the decline.

7. Conclusion

Trustworthiness has attracted extensive attention in SIoV, and many reputation-based approaches are applied to solving trustworthiness-related issues. Nevertheless, existing works seldom adopt reputation-based mechanisms to handle the trustworthiness related issues that arise in VFC. The fog vehicles in VFC may pursue improper revenues by delivering poor-quality computing services. Unfortunately, most of the existing works just assume that vehicles are unselfish and deliver their computing services as claimed. Such an assumption does not always hold in reality. Considering the selfishness of vehicles, a reputation-based service provisioning scheme is proposed to prevent fog vehicles from delivering low-quality computing services. In the meanwhile, a reputation management scheme is adopted, which consists of the decentralized reputation updating and global reputation synchronization in VFC. We have formulated the optimization problem to maximize the accumulated reputation of all the serving fog nodes in the optimization period. We have also carried out extensive simulation and the results reveal that our approach outstands other existing approaches. For the future work, more efficient strategies are required for improving the fairness and enthusiasm of newly registered fog vehicles, e.g., by avoiding the selection of the same fog node in the consecutive time slots.

Acknowledgment

This work was supported by the National Natural Science Foundation of China under Grant Number 62071327.

References

[1] Rasha Jamal Atwa, Paola Flocchini, and Amiya Nayak. A fog-based reputation evaluation model for vanets. In *International Symposium on Networks, Computers and Communications, ISNCC 2021, Dubai, United Arab Emirates, October 31 - November 2, 2021*, pages 1–7. IEEE, 2021.

[2] Chen Chen, Lei Liu, Shaohua Wan, Xiaozhe Hui, and Qingqi Pei. Data dissemination for industry 4.0 applications in internet of vehicles based on short-term traffic prediction. *ACM Trans. Internet Technol.*, 22(1), 2021.

[3] Tom H. Luan, Rongxing Lu, Xuemin Shen, and Fan Bai. Social on the road: enabling secure and efficient social networking on highways. *IEEE Wirel. Commun.*, 22(1):44–51, 2015.

[4] Chaogang Tang, Xianglin Wei, Chunsheng Zhu, Yi Wang, and Weijia Jia. Mobile vehicles as fog nodes for latency optimization in smart cities. *IEEE Trans. Veh. Technol.*, 69(9):9364–9375, 2020.

[5] Chaogang Tang, Xianglin Wei, Chong Liu, Haifeng Jiang, Huaming Wu, and Qing Li. Uav-enabled social internet of vehicles: Roles, security issues and use cases. In Yang Xiang, Zheli Liu, and Jin Li, editors, *Security and Privacy in Social Networks and Big Data - 6th International Symposium, SocialSec 2020, Tianjin, China, September 26-27, 2020, Proceedings*, volume 1298 of *Communications in Computer and Information Science*, pages 153–163. Springer, 2020.

[6] Giancarlo Fortino, Lidia Fotia, Fabrizio Messina, Domenico Rosaci, and Giuseppe M. L. Sarné. A blockchain-based group formation strategy for optimizing the social reputation capital of an iot scenario. *Simul. Model. Pract. Theory*, 108:102261, 2021.

[7] Zhengyang Hu, Xiaopeng Li, Juan Wang, Chengyi Xia, Zhen Wang, and Matjaz Perc. Adaptive reputation promotes trust in social networks. *IEEE Trans. Netw. Sci. Eng.*, 8(4):3087–3098, 2021.

[8] Catarina Nabais, Paulo Rogério Pereira, and Naércio Magaia. Birep: A reputation scheme to mitigate the effects of black-hole nodes in delay-tolerant internet of vehicles. *Sensors*, 21(3):835, 2021.

[9] Haiyang Yu, Yang Yang, Haoyang Zhang, Runkun Liu, and Yilong Ren. Reputation-based reverse combination auction incentive method to encourage vehicles to participate in the VCS system. *IEEE Trans. Netw. Sci. Eng.*, 8(3):2469–2481, 2021.

[10] Zihao Zhu, Yulin Xu, and Zhou Su. A reputation-based cooperative content delivery with parking vehicles in vehicular ad-hoc networks. *Peer-to-Peer Netw. Appl.*, 14(3):1531–1547, 2021.

[11] Huaming Wu, Ziru Zhang, Chang Guan, Katinka Wolter, and Minxian Xu. Collaborate edge and cloud computing with distributed deep learning for smart city internet of things. *IEEE Internet of Things Journal*, 7(9):8099–8110, 2020.

[12] Wenting Wei, Ruying Yang, Huaxi Gu, Weike Zhao, Chen Chen, and Shaohua Wan. Multi-objective optimization for resource allocation in vehicular cloud computing networks. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–10, 2021.

[13] Nurzhan Zhumabekuly Aitzhan and Davor Svetinovic. Security and privacy in decentralized energy trading through multi-signatures, blockchain and anonymous messaging streams. *IEEE Trans. Dependable Secur. Comput.*, 15(5):840–852, 2018.

[14] Haijun Liao, Yansong Mu, Zhenyu Zhou, Meng Sun, Zhao Wang, and Chao Pan. Blockchain and learning-based secure and intelligent task offloading for vehicular fog computing. *IEEE Trans. Intell. Transp. Syst.*, 22(7):4051–4063, 2021.

[15] Xiaolong Xu, Xuyun Zhang, Honghao Gao, Yuan Xue, Lianyong Qi, and Wanchun Dou. Become: Blockchain-enabled computation offloading for iot in mobile edge computing. *IEEE Trans. Ind. Informatics*, 16(6):4187–4195, 2020.

[16] Ming Li, Jian Weng, Anjia Yang, Jia-Nan Liu, and Xiaodong Lin. Toward blockchain-based fair and anonymous ad dissemination in vehicular networks. *IEEE Trans. Veh. Technol.*, 68(11):11248–11259, 2019.

[17] Qiaohong Hu, Hongju Cheng, Xiaoqi Zhang, and Chengkuan Lin. Trusted resource allocation based on proof-of-reputation consensus mechanism for edge computing. *Peer-to-Peer Netw. Appl.*, 15(1):444–460, 2022.

[18] An Liu, Qing Li, Liusheng Huang, Shiting Wen, Chaogang Tang, and Mingjun Xiao. Reputation-driven recommendation of services with uncertain qos. In *5th IEEE Asia-Pacific Services Computing Conference, APSCC 2010, 6-10 December 2010, Hangzhou, China, Proceedings*, pages 115–122. IEEE Computer Society, 2010.

[19] Xianglin Wei, Ming Chen, Chaogang Tang, Huali Bai, Guomin Zhang, and Zhanfeng Wang. irep: indirect reciprocity reputation based efficient content delivery in bt-like systems. *Telecommun. Syst.*, 54(1):47–60, 2013.

[20] Shiting Wen, Qing Li, Lihua Yue, An Liu, Chaogang Tang, and Farong Zhong. CRP: context-based reputation propagation in services composition. *Serv. Oriented Comput. Appl.*, 6(3):231–248, 2012.

[21] Diego de Siqueira Braga, Marco Niemann, Bernd Hellingrath, and Fernando Buarque de Lima Neto. Survey on computational trust and reputation models. *ACM Comput. Surv.*, 51(5):101:1–101:40, 2019.

[22] Chen Chen, Yuru Zhang, Zheng Wang, Shaohua Wan, and Qingqi Pei. Distributed computation offloading method based on deep reinforcement learning in ICV. *Appl. Soft Comput.*, 103:107108, 2021.

[23] Su Liu, Jiong Yu, Xiaoheng Deng, and Shaohua Wan. Fedcpf: An efficient-communication federated learning approach for vehicular edge computing in 6g communication networks. *IEEE Trans. Intell. Transp. Syst.*, 23(2):1616–1629, 2022.

[24] Richard Gilles Engoulou, Martine Bellaïche, Talal Halabi, and Samuel Pierre. A decentralized reputation management system for securing the internet of vehicles. In *International Conference on Computing, Networking and Communications, ICNC 2019, Honolulu, HI, USA, February 18-21, 2019*, pages 900–904. IEEE, 2019.

[25] Muhammad Awais Javed and Sherali Zeadally. Repguide: Reputation-based route guidance using internet of vehicles. *IEEE Commun. Stand. Mag.*, 2(4):81–87, 2018.

[26] Feng Zeng, Yaojia Chen, Lan Yao, and Jinsong Wu. A novel reputation incentive mechanism and game theory analysis for service caching in software-defined vehicle edge computing. *Peer-to-Peer Netw. Appl.*, 14(2):467–481, 2021.

[27] Shen Su, Zhihong Tian, Siyu Liang, Shuang Li, Sha-sha Du, and Nadra Guizani. A reputation management scheme for efficient malicious vehicle identification over 5g networks. *IEEE Wirel. Commun.*, 27(3):46–52, 2020.

[28] Mingfeng Huang, Anfeng Liu, Neal N. Xiong, and Jie Wu. A uav-assisted ubiquitous trust communication system in 5g and beyond networks. *IEEE J. Sel. Areas Commun.*, 39(11):3444–3458, 2021.

[29] Miaojiang Chen, Tian Wang, Kaoru Ota, Mianxiong Dong, Ming Zhao, and Anfeng Liu. Intelligent resource allocation management for vehicles network: An A3C learning approach. *Comput. Commun.*, 151:485–494, 2020.

[30] Miaojiang Chen, Tian Wang, Shaobo Zhang, and Anfeng Liu. Deep reinforcement learning for computation offloading in mobile edge computing environment. *Comput. Commun.*, 175:1–12, 2021.

[31] Chaogang Tang and Huaming Wu. Optimal computational resource pricing in vehicular edge computing: A stackelberg game approach. *J. Syst. Archit.*, 121:102331, 2021.

[32] Chaogang Tang, Chunsheng Zhu, Huaming Wu, Qing Li, and Joel J. P. C. Rodrigues. Toward response time minimization considering energy consumption in caching-assisted vehicular edge computing. *IEEE Internet Things J.*, 9(7):5051–5064, 2022.

[33] Xu Chen, Lei Jiao, Wenzhong Li, and Xiaoming Fu. Efficient multi-user computation offloading for mobile-edge cloud computing. *IEEE/ACM Trans. Netw.*, 24(5):2795–2808, 2016.