Reliability-Aware Network Service Provisioning in Mobile Edge-Cloud Networks

Jing Li, Weifa Liang, *Senior Member, IEEE*, Meitian Huang, and Xiaohua Jia, *Fellow, IEEE*

Abstract—The Mobile Edge-Cloud (MEC) network has emerged as a promising networking paradigm to address the conflict between increasing computing-intensive applications and resource-constrained mobile Internet-of-Thing (IoT) devices with portable size and storage. In MEC environments, Virtualized Network Functions (VNFs) are deployed for provisioning network services to users to reduce the service cost on top of dedicated hardware infrastructures. However, VNFs may suffer from failures and malfunctions while network service providers have to guarantee continuously reliable services to their consumers to meet the ever-growing service demands of users, thereby securing their revenues for the service. In this article, we focus on reliable VNF service provisioning in MECs, by placing primary and backup VNF instances to cloudlets in an MEC network to meet the service reliability requirements of users. We first formulate a novel VNF service reliability problem with the aim to maximize the revenue collected by admitting as many as user requests while meeting their different reliability requirements, assuming that requests arrive into the system one by one without the knowledge of future arrivals, and the admission or rejection decision must be made immediately. We then develop two efficient online algorithms for the problem under two different backup schemes: the on-site (local) and off-site (remote) schemes, by adopting the primal-dual updating technique. Both algorithms achieve provable competitive ratios with bounded moderate resource capacity violations. We finally evaluate the proposed algorithms through experimental simulations. Experimental results demonstrate that the proposed algorithms are promising, compared with existing baseline algorithms.

Index Terms—Mobile edge computing (MEC), virtualized network function (VNF), virtualized service functions (VNFs), revenue maximization, reliability-aware service provisioning, Fault-tolerance, software failure, cloudlet failure, online algorithms, the primal-dual dynamic updating technique, mobile edge-cloud networks, competitive ratio analysis, resource allocation and optimization

1 Introduction

The recent advancement of Internet of the Things (IoTs) has further flourished new applications of mobile IoT devices [23]. However, due to their portable sizes and limited computing capacities, mobile IoT devices are unable to meet the demands of both computing and storage resources of these applications [10]. To meet ever-growing mobile users' resource demands, MEC as a promising paradigm has been introduced, which extends cloud computing services to the edge of mobile networks by utilizing cloudlets (servers, or clusters of servers) co-located with Access Points (APs) within the proximity of mobile users [17]. In addition, Network Functions Virtualization (NFV) is a promising technique that contributes to applications of MEC [19], and the instances of NFVs are employed to replace dedicated hardware equipment due to not only the cost reduction but also the advantage of adjusting services with agility to cope with rapid-changing user demands without the hassle of deployment and maintenance of physical infrastructures [25], where each network function is implemented in a virtual machine as a piece of software that is referred to as an instance of the Virtualized Network Function (VNF).

Meeting user service reliability requirements is critical to any network service provider. A mobile user usually not just requests a specific VNF service but also has a certain reliability requirement for the service [21]. The reliability of a network is defined as the ability of the network to provide stable services that ensure an agreed level of operational performance facing the risk of failures of underlying network components [4]. In other words, the reliability of a network refers to the ability of the network to recover from failures through some specific precautionary measures. The service reliability of a network is determined by its hardware and software components, and the broken down of any its components will result in the unavailability of the network [20]. Even a temporary malfunction of the service can potentially cause the loss of data and compromise the service security integration [7]. As the service in MEC is implemented by VNF instances which are pieces of software running in virtual machines in cloudlets, the software cannot be guaranteed to be 100 percent reliable, and instead they are prone to fail due to bugs. On the other hand, the cloudlets running VNF instances are also not always reliable either, they may fail occasionally. To deal with such failures efficiently, several prevention and recovery mechanisms are developed in the past. Among them, a common and practical approach is utilizing redundancy [2]. Specifically, a single VNF instance failure can be mitigated by deploying its other VNF backup instances in the same cloudlet as its primary VNF instance [4]. We refer to this recovering scheme as

• *J. Li, W. Liang, and M. Huang are with the Research School of Computer Science, The Australian National University, Canberra, ACT 2601, Australia. E-mail: {Jing.Li, meitian.huang}@anu.edu.au, wliang@cs.anu.edu.au.*
• *X. Jia is with the Department of Computer Science, City University of Hong Kong, Hong Kong, P.R. China. E-mail: csjia@cityu.edu.hk.*

Manuscript received 18 Nov. 2019; revised 21 Jan. 2020; accepted 24 Jan. 2020. Date of publication 29 Jan. 2020; date of current version 25 Feb. 2020.
(Corresponding author: Weifa Liang.)
Recommended for acceptance by Y. Yang.
Digital Object Identifier no. 10.1109/TPDS.2020.2970048

the on-site scheme that utilizes the local redundancy to meet the required reliability, which guarantees low switching latency from primary VNF instances to their backups. However, the VNF reliability under the on-site scheme is restricted by the reliability of the cloudlet at which the VNF instances located, because all VNF instances in a cloudlet will not function if the cloudlet fails. Instead, another recovering scheme - *the off-site scheme*, is proposed, which can mitigate this restriction. The off-site scheme instantiates backup VNF instances at cloudlets that are physically separated from the cloudlet hosting the primary VNF instances [24]. Although the off-site scheme can improve VNF reliabilities, it suffers drawbacks. Since it utilizes geographic redundancy, the recovery time will be slightly longer, and will incur extra costs of network traffic between the cloudlets hosting the primary and its backup VNF instances for each service [10]. In this paper, VNF replicas are deployed as backups to meet the requested reliability requirements of mobile users. This poses a challenge to maximize the revenue of network service providers by admitting as many as user requests through developing efficient online algorithms for user request admissions.

The novelty of this paper is that we study the VNF service reliability problem in MEC networks with the aim of maximizing the revenue collected by network service providers. We are the first to consider the provisioning of reliable VNF services in an MEC by joint considering both the VNF instance reliability (software) and the cloudlet reliability (hardware), and propose the very first online algorithms with provable competitive ratios for the problem under two different VNF placement backup schemes: the on-site and off-site schemes.

The main contributions of this paper are as follows.

- We formulate a VNF service reliability problem in an MEC environment, with the aim to maximize the revenue collected by network service providers through admitting as many as user requests with different reliability requirements.
- We show the NP-hardness of the problem, and formulate an Integer Linear Programming (ILP) for the off-line setting of the problem.
- We devise online algorithms for the problem under both on-site and off-site schemes. Both online algorithms can achieve provable competitive ratios at the expense of moderate resource capacity violations.
- We evaluate the performance of the proposed algorithms through experimental simulations. Experimental results demonstrate that the proposed algorithms are promising, and outperform baseline algorithms.

The remainder of the paper is organized as follows. Section 2 reviews the related work. Section 3 introduces the system model and defines the problem under both on-site and off-site schemes precisely. The NP-hardness of the defined problem is also shown in this section. Sections 4 and 5 formulate ILP solutions and devise online algorithms for the problem under both on-site and off-site schemes, respectively. Section 6 evaluates the performance of the proposed algorithms through experimental simulations, and Section 7 concludes the paper.

2 Related Work

Previous studies of improving VNF instance reliability are surveyed in [10]. These include state management, VNF migration, and rollback recovery. For example, Kanizo *et al.* [15] addressed the problem of optimizing the allocation of VNF backups. They proposed a survivability architecture and adopted the maximum matching method for resource allocation to VNFs. However, they assumed only one backup VNF instance for each primary VNF instance, and did not consider multiple backup VNF instances for each primary VNFs due to the difference in their reliabilities to meet reliability requirements. Especially, infrastructures, where VNF instances run, could also suffer from failures and malfunctions. Therefore, the awareness of infrastructure reliability poses challenges on reliability provisioning for VNF instance deployments. Kong *et al.* [16] proposed a coordinated protection mechanism that employs both network-layer protection and VNF replicas. They focused on minimizing the total computing resource consumption. Ding *et al.* [4] studied the problem of how to enhance the reliability of Service Function Chains (SFCs) more efficiently, and proposed the Cost-aware Importance Measure (CIM) in the selection of qualified backups. Beck *et al.* [1] jointly considered reliabilities of both VNFs and dedicated infrastructures, and focused on the resource cost savings by resource sharing among the backup VNF instances. Chen *et al.* [3] developed a task offloading scheduling utilizing Lyaponuv Optimization Approach to minimize the energy consumption. Qiu *et al.* [20] proposed a stochastic model to enhance the network reliability and reduce the response time, through leveraging a concurrent replication mechanism with cancelling. Huang *et al.* [13] studied the reliability-aware network services provisioning problem and proposed an approximation algorithm with a logarithmic approximation ratio in the off-line setting. Hu *et al.* [12] investigated the impacts of extent intermittent connectivity on the MEC network, and proposed a Markov chain based framework in a discrete-time manner to express the processing time of tasks under various MEC network conditions. Fan *et al.* [5] introduced an on-site pooling mechanism which improves the resource utilization, and thus reduces the resource consumption in the MEC network. Hmaity *et al.* [11] studied the VNF placement problem under the off-site scheme to provision reliable Service Chains (SCs). Li *et al.* [17] recently considered the VNF service reliability problem under both on-site or off-site schemes by proposed an online algorithm for the on-site scheme and a heuristic for the off-site scheme. This paper is an extension of this conference paper.

Unlike the aforementioned studies, in this paper we study the VNF service reliability problem for IoT applications in MEC environments, by jointly considering the reliabilities of both VNF instances and cloudlets. We aim to maximize the revenue collected by the network service provider through admitting as many as user requests, assuming that user requests arrive one by one without the knowledge of future service request arrivals. We will develop performance guaranteed online algorithms for this dynamic request problem.

3 Preliminaries

In this section, we first introduce the system model and then define the problem precisely.

3.1 Network System Model

We consider a Mobile Edge-Cloud (MEC) network consisting of APs, cloudlets, and links that connect APs. Each cloudlet is

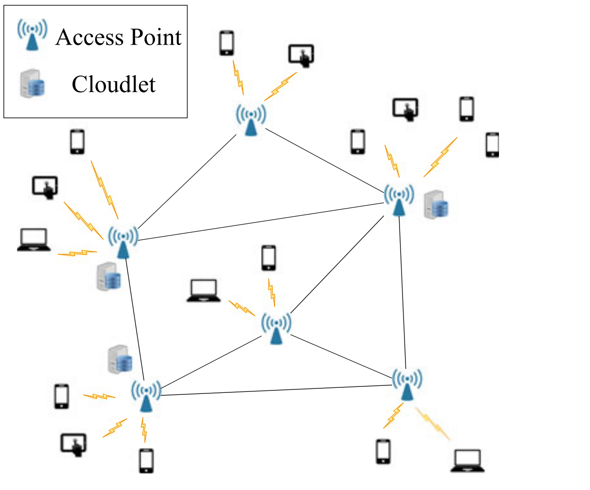


Fig. 1. An illustrative example of an MEC network consisting of five APs and three cloudlets co-located with APs.

co-located with an AP while some APs may not be co-located with any cloudlet. The MEC network is represented by an undirected graph $G = (V, E)$, where V is the set of APs and E is the set of links between APs. Let C be the set of cloudlets in G and $C \subseteq V$. Mobile users issue their service requests with reliability requirements to the MEC network through their nearby APs.

Let $F = \{f_1, f_2, \ldots, f_{|F|}\}$ be a set of different types of Virtualized Network Functions (VNFs) offered by the network G, e.g., Virtualized Load Balancers, Firewalls, Intrusion-Detection Systems, and so on. We assume that different VNFs need different amounts of computing resource for their implementations. Without loss of generality, we assume that the amount of computing resource demanded by a specific type of VNF, $f_i \in F$, is measured by computing units, denoted by $c(f_i)$. Furthermore, different VNFs have different reliabilities, and denote by $r(f_i)$ the reliability of f_i with $0 < r(f_i) < 1$.

Let $C = \{c_1, c_2, \ldots, c_{|C|}\}$ be the set of cloudlets in G with $|C| \leq |V|$. Each cloudlet $c_j \in C$ has a computing capacity cap_j. For a specific type of VNF, $f_i \in F$, the cloudlets can implement its VNF as a piece of software in a virtual machine with the demanded computing resource $c(f_i)$. In addition, different cloudlets may have different reliabilities, and denote by $r(c_j)$ the reliability of cloudlet $c_j \in C$ with $0 < r(c_j) < 1$. When a cloudlet c_j fails, all VNF instances in it will not be able to offer their services anymore, and thus become unavailable. Fig. 1 presents an illustration of a mobile edge cloud network consisting of five APs and three cloudlets co-located with APs.

3.2 The Competitive Ratio of an Online Algorithm

When considering an online maximization problem $\mathcal{P}$, we say an online algorithm with competitive ratio α for the problem if the solution delivered by the online algorithm is no less than $OPT(\mathcal{P})_{off}/\alpha$, where $OPT(\mathcal{P})_{off}$ is the optimal solution of the offline version of problem $\mathcal{P}$ with $\alpha > 1$.

3.3 User Request Scheduling

We consider that the MEC network is in a discrete-time fashion and assume that time is slotted into equal time slots with each a time unit, and denote by $\mathbb{T} = \{1, 2, \ldots, |\mathbb{T}|\}$ the set of time slots. Consequently, the arrival time of a request can be represented by a time slot label, while the processing duration of the request in MEC is expressed by the number of time slots needed.

We assume that each mobile user requests only one VNF instance service [14], [18], [22], and each user request ρ_i is defined by a tuple $(f_i, R_i, a_i, d_i, pay_i)$, where $f_i \in F$ is the requested VNF instance, R_i is the reliability requirement of the request with $0 < R_i < 1$, $a_i \in \mathbb{T}$ is the arrival time slot of the request, d_i is the execution duration of the request, and pay_i is the payment received by implementing the request. We further assume that d_i is a positive integer.

We assume that requests will be processed (accepted or rejected) in the beginning of each time slot, and we focus on a given time horizon $\mathbb{T}$. Denote by $\mathbb{R}$ the set of all requests within the time horizon $\mathbb{T}$. That is, for a request ρ_i arriving at time slot a_i, we treat it as an event lasting d_i time slots. For a better presentation of both the arrival time and the requested execution duration, we introduce T_i, which is a binary vector with the length of $\mathbb{T}$, and generated by both a_i and d_i. In a certain time slot t, $T_i[t] = 1$ represents that the requested execution duration of request covers time slot t, and $T_i[t] = 0$ otherwise, e.g., in the case $\mathbb{T} = 3$, $a_i = 1$ and $d_i = 2$, then we let $T_i = [1, 1, 0]$.

Suppose that the admission scheduling of requests is in an online manner. There is a hypervisor in the system to deal with incoming requests, where requests arrive one by one without the knowledge of future request arrivals. If the hypervisor accepts request ρ_i, it will allocate the resource from one or multiple cloudlets to accommodate its primary and backup VNF instances for its requested service and to find a routing path for its data traffic.

3.4 Problem Definition

We deal with *the VNF service reliability problem* under both on-site and off-site schemes. Given an MEC network $G(V, E)$ and a finite time horizon $\mathbb{T}$, there are a set C of cloudlets with each $c_j \in C$ having a computing capacity cap_j and reliability $r(c_j)$, requests arrive one by one without the knowledge of future arrivals. Assume that each request ρ_i requests a specific network service f_i with a reliability requirement R_i and a payment pay_i, the VNF service reliability problem is to maximize the cumulative revenue for the given time horizon, by admitting as many as requests while meeting the reliability requirements of admitted requests, subject to computing capacity on each cloudlet in G.

To meet the reliability requirement of a request, we place redundant VNF instances to cloudlets. Depending on different VNF instance placements for the request, we have two different placement schemes: the on-site placement in which all VNF instances of a request must be placed to a single cloudlets; and the off-site-placement in which the VNF instances of a request can be placed to different cloudlets while only one of its VNF instances can be placed to a cloudlet. This implies that no more than $|C|$ VNF instances for any request to be placed in the network to meet its reliability requirement.

3.5 NP-Hardness of the VNF Service Reliability Problem

Theorem 1. *The VNF service reliability problem in an MEC network $G = (V, E)$ is NP-hard.*

Proof. We prove the NP-hardness of the VNF service reliability problem by a reduction from a well known NP-hard problem - the knapsack problem, as follows. Given a bin with a capacity W and n items, item i has a profit p_i with weight w_i, $1 \leq i \leq n$. The knapsack problem aims to maximize the collected profit through packing as many as items into the bin, subject to the bin capacity W.

We consider a special case of the VNF service reliability problem, where there is only one cloudlet with a capacity W in the MEC network, and one VNF instance for each request will satisfy its reliability requirement. Also, all requests arrive at the same time slot with the same execution duration. Then, if we admit request ρ_i, we can collect payment p_i at the expense of computing resource of w_i. To this end, the VNF service reliability problem aims to maximize the revenue collected through admitting as many requests as possible, subject to the cloudlet capacity W. It can be seen that the special case of the VNF service reliability problem is equivalent to the knapsack problem. Hence, the VNF service reliability problem is NP-hard, due to the NP-hardness of the knapsack problem. □

4 Online Algorithm for the VNF Service Reliability Problem Under the On-Site Scheme

In this section we study the VNF service reliability problem under the on-site scheme, where all backup VNF instances and the primary VNF instance of each request must be placed to a single cloudlet. In the following we first formulate an Integer Linear Programming (ILP) for the offline version of the problem. We then devise an online algorithm for the problem, based on the ILP solution by adopting the primal-dual dynamic updating technique [8].

4.1 Integer Linear Programming Formulation

Let N_{ij} be the minimum number of primary and backup VNF instances needed to be placed in cloudlet $c_j \in C$ for request ρ_i to meet its reliability requirement R_i. We have

$$r(c_j) \cdot (1 - (1 - r(f_i))^{N_{ij}}) \geq R_i, \tag{1}$$

where $(1 - r(f_i))^{N_{ij}}$ is the failure probability of all primary and backup VNF instances of request ρ_i, while $1 - (1 - r(f_i))^{N_{ij}}$ is the probability of at least one VNF instance survival. If $r(c_j) > R_i$, N_{ij} can be calculated as follows.

$$N_{ij} = \left\lceil \log_{1-r(f_i)} \left(1 - \frac{R_i}{r(c_j)}\right) \right\rceil. \tag{2}$$

Otherwise $(r(c_j) \leq R_i)$, the reliability requirement of ρ_i cannot be met if the VNF instances of ρ_j sis placed to cloudlet c_j. For the sake of simplicity, in the rest of discussions, we assume that $r(c_j) > R_i, \forall c_j \in C, \forall \rho_i \in \mathbb{R}$, under the on-site scheme.

Let $X_i \in \{0, 1\}$ denote whether request ρ_i is admitted, i.e., $X_i = 1$ if the request is admitted; and $X_i = 0$ otherwise. If request ρ_i is admitted, then the probability of that at least one of its primary and backup VNF instances is available is no less than its specified reliability requirement R_i.

Let $Y_{ij} \in \{0, 1\}$ denote whether at least one VNF instance of request ρ_i will be placed to cloudlet $c_j \in C$. That is, $Y_{ij} = 1$ if at least one VNF instance of ρ_i is placed to cloudlet c_j; $Y_{ij} = 0$ otherwise. Associated with each request ρ_i, we assume that its payment pay_i and reliability requirement R_i are given, too.

The VNF service reliability problem under the on-site scheme can be formulated as an ILP with the objective to

$$\mathbf{P1:} \quad \text{maximize} \quad \sum_{\rho_i \in \mathbb{R}} X_i \cdot pay_i,$$

subject to

$$\sum_{\rho_i \in \mathbb{R}} T_i[t] \cdot N_{ij} \cdot c(f_i) \cdot Y_{ij} \leq cap_j, \quad \forall t \in \mathbb{T}, \forall c_j \in C. \tag{3}$$

$$\sum_{c_j \in C} Y_{ij} = X_i, \quad \forall \rho_i \in \mathbb{R}. \tag{4}$$

$$X_i \in \{0, 1\}, \quad \forall \rho_i \in \mathbb{R}, \tag{5}$$

$$Y_{ij} \in \{0, 1\}, \quad \forall \rho_i \in \mathbb{R}, \forall c_j \in C, \tag{6}$$

where Constraint (3) is the computing capacity constraint on each cloudlet in each time slot t, and Constraint (4) ensures that only one cloudlet is chosen to accommodate all VNF instances of request ρ_i under the on-site scheme if request ρ_i is admitted.

4.2 Online Algorithm

We now deal with the VNF service reliability problem under the on-site scheme, denoted by **P1**, by devising an online algorithm for it based on the linear relaxation of the ILP formulation.

The general strategy for **P1** proceeds as follows. We first perform linear relaxation of **P1**, and let **P2** be the relaxed Linear Programming (LP) of **P1**. We then find the dual **P3** of **P2**. A solution to the online version of **P1** can then be obtained from a feasible solution to **P3**, which has a provable competitive ratio with moderate resource capacity violations. Specifically, **P2** is obtained by performing the LP relaxation on **P1**, which is expressed as follows.

$$\mathbf{P2:} \quad \text{Maximize} \sum_{\rho_i \in \mathbb{R}} X_i \cdot pay_i,$$

subject to

$$\begin{aligned} &(3),\ (4), \\ &X_i \leq 1, \quad \forall \rho_i \in \mathbb{R}, \end{aligned} \tag{7}$$

$$X_i \geq 0, Y_{ij} \geq 0, \quad \forall \rho_i \in \mathbb{R}, \forall c_j \in C. \tag{8}$$

It is noted that any feasible solution for **P1** is a feasible solution for **P2**. Recall that **P3** is the dual of **P2**, **P3** is formulated as follows.

$$\mathbf{P3:} \quad \text{Minimize} \sum_{t \in \mathbb{T}, c_j \in C} cap_j \cdot \lambda_{tj} + \sum_{\rho_i \in \mathbb{R}} \delta_i, \tag{9}$$

subject to

$$\beta_i + \delta_i - pay_i \geq 0, \quad \forall \rho_i \in \mathbb{R}, \tag{10}$$

$$\sum_{t \in \mathbb{T}} T_i[t] \cdot N_{ij} \cdot c(f_i) \cdot \lambda_{tj} - \beta_i \geq 0, \ \forall \rho_i \in \mathbb{R}, \forall c_j \in C, \tag{11}$$

$$\lambda_{tj} \geq 0,\ \beta_i \geq 0,\ \delta_i \geq 0,\ \forall \rho_i \in \mathbb{R}, \forall c_j \in C, \forall t \in \mathbb{T}, \tag{12}$$

where λ_{tj}, β_i, and δ_i are dual variables for constraints (3), (4) and (7) in **P2**, respectively.

Algorithm 1. Algorithm for Reliability-Aware Network Service Provisioning Under the On-Site Scheme

```
Input: An MEC network $G = (V, E)$ and requests come one by one.
Output: An online scheduling of incoming requests for a given time horizon $\mathbb{T}$.
 1: Initialize $X_i$, $Y_{ij}$, $\lambda_{tj}$, $\beta_i$ =0, $\forall \rho_i \in \mathbb{R}$, $\forall c_j \in C$, $\forall t \in \mathbb{T}$;
 2: while Upon arrival of a request $\rho_i$ do
 3:   for cloudlet $c_j \in C$ do
 4:     Calculate $N_{ij}$ and $\sum_{t\in\mathbb{T}} T_i[t] \cdot N_{ij} \cdot c(f_i) \cdot \lambda_{tj}$ for $c_j$;
 5:   end for ;
 6:   select cloudlet $c_{j'}$ such that $\sum_{t\in\mathbb{T}} T_i[t] \cdot N_{ij'} \cdot c(f_i) \cdot \lambda_{tj'} = \min_{c_j\in C}$
      $\{\sum_{t\in\mathbb{T}} T_i[t] \cdot N_{ij} \cdot c(f_i) \cdot \lambda_{tj}\}$;
 7:   if $pay_i - \sum_{t\in\mathbb{T}} T_i[t] \cdot N_{ij'} \cdot c(f_i) \cdot \lambda_{tj'} > 0$ then
 8:     Admit request $\rho_i$;
 9:     $X_i \leftarrow 1$; $Y_{ij'} \leftarrow 1$;
10:     $\delta_i \leftarrow pay_i - \sum_{t\in\mathbb{T}} T_i[t] \cdot N_{ij'} \cdot c(f_i) \cdot \lambda_{tj'}$;
11:     $\lambda_{tj'} \leftarrow \lambda_{tj'} \cdot (1 + \frac{N_{ij'} \cdot c(f_i)}{cap_{j'}}) + \frac{N_{ij'} \cdot c(f_i) \cdot pay_i}{d_i \cdot cap_{j'}}$, according to the chosen cloudlet $c_{j'}$ and the execution time slot $t \in \mathbb{T}'_i$;
12:   else
13:     Reject request $\rho_i$;
14:   end if ;
15: end while .
```

We then rewrite (10) and (11) as follows.

$$\beta_i \geq pay_i - \delta_i, \tag{13}$$

$$\beta_i \leq \sum_{t\in\mathbb{T}} T_i[t] \cdot N_{ij} \cdot c(f_i) \cdot \lambda_{tj}. \tag{14}$$

From Inequality (14), we have

$$\beta_i \leq \min_{c_j\in C} \sum_{t\in\mathbb{T}} T_i[t] \cdot N_{ij} \cdot c(f_i) \cdot \lambda_{tj}. \tag{15}$$

Combining Inequalities (13) and (15), we have

$$pay_i - \delta_i \leq \min_{c_j\in C} \sum_{t\in\mathbb{T}} T_i[t] \cdot N_{ij} \cdot c(f_i) \cdot \lambda_{tj}. \tag{16}$$

We now eliminate the dual variable β_i by Inequality (15). We then only consider dual variables δ_i and λ_{tj} to ensure that Inequality (16) holds. We perform a transformation on Inequality (16) as follows.

$$\delta_i \geq pay_i - \min_{c_j\in C} \sum_{t\in\mathbb{T}} T_i[t] \cdot N_{ij} \cdot c(f_i) \cdot \lambda_{tj}. \tag{17}$$

The rest is to solve **P3**, we adopt the primal-dual dynamic updating technique [8] to update the variables in both primal and dual LP simultaneously. Specifically, for request ρ_i, the set of execution time slots is denoted as $\mathbb{T}'_i \subset \mathbb{T}$, and its duration is $d_i = |\mathbb{T}'_i|$. Dual variables λ_{tj} and β_i are 0 initially. Upon the arrival of request ρ_i, the dual variables are required to be set properly to satisfy Inequality (17). To this end, we first calculate N_{ij} for each cloudlet $c_j \in C$. We then calculate the value of $\sum_{t\in\mathbb{T}} T_i[t] \cdot N_{ij} \cdot c(f_i) \cdot \lambda_{tj}$ associated with each cloudlet $c_j \in C$. We finally identify the cloudlet c_j with $\min_{c_j\in C}\{\sum_{t\in\mathbb{T}} T_i[t] \cdot N_{ij} \cdot c(f_i) \cdot \lambda_{tj}\}$. If $pay_i - \min_{c_j\in C}\{\sum_{t\in\mathbb{T}} T_i[t] \cdot N_{ij} \cdot c(f_i) \cdot \lambda_{tj}\} \leq 0$, request ρ_i will be rejected. Otherwise, it will be admitted and all its VNFs will be accommodated in cloudlet c_j, and the value of δ_i is updated as follows.

$$\delta_i := pay_i - \min_{c_j\in C}\left\{\sum_{t\in\mathbb{T}} T_i[t] \cdot N_{ij} \cdot c(f_i) \cdot \lambda_{tj}\right\}. \tag{18}$$

The value of λ_{tj} associated with the chosen cloudlet $c_j \in C_i$ and $t \in \mathbb{T}'_i$ is updated as follows.

$$\lambda_{tj} := \lambda_{tj} \cdot \left(1 + \frac{N_{ij} \cdot c(f_i)}{cap_j}\right) + \frac{N_{ij} \cdot c(f_i) \cdot pay_i}{d_i \cdot cap_j}. \tag{19}$$

The proposed online algorithm for **P1** under the on-site scheme is given in `Algorithm` 1.

Let $a_{ij} = N_{ij} \cdot c(f_i)$ for any $\rho_i \in \mathbb{R}$ and for any $c_j \in C$, $a_{max} = \max_{\rho_i\in\mathbb{R}, c_j\in C}\{a_{ij}\}$, and $a_{min} = \min_{\rho_i\in\mathbb{R}, c_j\in C}\{a_{ij}\}$.

Lemma 1. *Let P_{on} and D_{on} be the values of the solutions delivered by the proposed algorithm for* **P2** *and* **P3**, *respectively, Then, $(1 + a_{max}) \cdot P_{on} \geq D_{on}$, where $a_{max} = \max_{\rho_i\in\mathbb{R}, c_j\in C}\{N_{ij} \cdot c(f_i)\}$.*

Proof. It can be seen that $P_{on} = D_{on} = 0$ initially.

We now prove the claim by showing that $(1 + a_{max}) \cdot \Delta P_{on} \geq \Delta D_{on}$ when request ρ_i arrives, where ΔP_{on} and ΔD_{on} are the value differences of the objective functions for **P2** and **P3** before and after a request arrives.

If request ρ_i is rejected, $\Delta P_{on} = \Delta D_{on} = 0$, and $(1 + a_{max}) \cdot \Delta P_{on} \geq \Delta D_{on}$.

Otherwise $\Delta P_{on} = pay_i$ and $\Delta D_{on} = \sum_{t\in\mathbb{T}'_i} cap_j \cdot \Delta\lambda_{tj} + \delta_i$, by (9), where $\Delta\lambda_{tj}$ is the difference before and after updating the value of λ_{tj}, which is associated with the chosen cloudlet c_j. By the update function (19) of λ_{tj}, we have

$$\begin{aligned}
\Delta D_{on} &= \sum_{t\in\mathbb{T}'_i} cap_j \cdot \Delta\lambda_{tj} + \delta_i \\
&= \sum_{t\in\mathbb{T}'_i} cap_j \cdot \left(\frac{N_{ij} \cdot c(f_i)}{cap_j} \cdot \lambda_{tj} + \frac{N_{ij} \cdot c(f_i) \cdot pay_i}{d_i \cdot cap_j}\right) + \delta_i \\
&= \sum_{t\in\mathbb{T}'_i} (N_{ij} \cdot c(f_i) \cdot \lambda_{tj}) + \sum_{t\in\mathbb{T}'_i} \frac{N_{ij} \cdot c(f_i) \cdot pay_i}{d_i} + \delta_i \\
&= \sum_{t\in\mathbb{T}} T_i[t] \cdot N_{ij} \cdot c(f_i) \cdot \lambda_{tj} + \delta_i + N_{ij} \cdot c(f_i) \cdot pay_i \\
&= pay_i + N_{ij} \cdot c(f_i) \cdot pay_i
\end{aligned} \tag{20}$$

$$\begin{aligned}
&= (1 + N_{ij} \cdot c(f_i)) \cdot pay_i \\
&\leq (1 + a_{max}) \cdot pay_i \\
&= (1 + a_{max}) \cdot \Delta P_{on}.
\end{aligned} \tag{21}$$

Notice that Inequality (20) holds, because from the update function (18), λ_{tj} is the value derived by identifying a cloudlet c_j with $\min \sum_{t\in\mathbb{T}} T_i[t] \cdot N_{ij} \cdot c(f_i) \cdot \lambda_{tj}$ and $pay_i = \sum_{t\in\mathbb{T}} T_i[t] \cdot N_{ij} \cdot c(f_i) \cdot \lambda_{tj} + \delta_i$. Hence, the lemma follows. □

Lemma 2. *`Algorithm` 1 delivers a feasible solution for **P3**.*

Proof. Following `Algorithm` 1, Constraint (17) is satisfied by the update function of δ_i (18) when request ρ_i arrives. Notice that the update function (19) of λ_{tj} is non-decreasing, $pay_i - \min_{c_j \in C}\{\sum_{t\in\mathbb{T}} T_i[t]\cdot N_{ij}\cdot c(f_i)\cdot \lambda_{tj}\}$ thus is non-increasing. Constraint (17) still holds when updating the value of λ_{tj}. The lemma then follows. □

Denote by pay_{max} and pay_{min} the maximum and minimum payments, d_{max} and d_{min} the maximum and minimum execution durations among requests, and cap_{max} and cap_{min} the maximum and minimum capacities of cloudlets.

Lemma 3.

$$\lambda_{tj} \ge \frac{pay_{min}}{d_{max}}\cdot\left(\left(1+\frac{a_{min}}{cap_{max}}\right)^{\sum_{\rho_i\in\mathbb{R}} T_i[t]\cdot Y_{ij}} -1\right). \tag{22}$$

The proof can be found in Appendix A, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/TPDS.2020.2970048.

Lemma 4.

$$\lambda_{tj} < \frac{pay_{max}}{a_{min}}\cdot\left(1+\frac{a_{max}}{cap_{min}}\right)+\frac{a_{max}\cdot pay_{max}}{d_{min}\cdot cap_{min}}. \tag{23}$$

The proof can be found in Appendix B, available in the online supplemental material.

Lemma 5. *In the solution delivered by `Algorithm` 1 for **P1**, the capacity constraint violation on each cloudlet is bounded by ξ, where $\xi = \frac{a_{max}}{cap_{min}\cdot\ln(1+\frac{a_{min}}{cap_{max}})}\cdot\ln(\frac{pay_{max}\cdot d_{max}}{pay_{min}}\cdot(\frac{1}{a_{min}}+\frac{a_{max}}{a_{min}\cdot cap_{min}}+\frac{a_{max}}{d_{min}\cdot cap_{min}})+1)$.*

Proof. Combining Lemmas 3 and 4, we have

$$\frac{pay_{min}}{d_{max}}\cdot\left(\left(1+\frac{a_{min}}{cap_{max}}\right)^{\sum_{\rho_i\in\mathbb{R}} T_i[t]\cdot Y_{ij}} -1\right) < \frac{pay_{max}}{a_{min}}\cdot\left(1+\frac{a_{max}}{cap_{min}}\right)+\frac{a_{max}\cdot pay_{max}}{d_{min}\cdot cap_{min}}.$$

Then,

$$\sum_{\rho_i\in\mathbb{R}} T_i[t]\cdot Y_{ij} < \frac{\ln(\frac{pay_{max}\cdot d_{max}}{pay_{min}}\cdot(\frac{1}{a_{min}}+\frac{a_{max}}{a_{min}\cdot cap_{min}}+\frac{a_{max}}{d_{min}\cdot cap_{min}})+1)}{\ln(1+\frac{a_{min}}{cap_{max}})}. \tag{24}$$

To calculate the capacity violation, we have

$$\begin{aligned}&\sum_{\rho_i\in\mathbb{R}} T_i[t]\cdot N_{ij}\cdot c(f_i)\cdot Y_{ij}\\ \le&\sum_{\rho_i\in\mathbb{R}} T_i[t]\cdot a_{max}\cdot Y_{ij}\\ \le&\frac{a_{max}}{\ln(1+\frac{a_{min}}{cap_{max}})}\cdot\ln\left(\frac{pay_{max}\cdot d_{max}}{pay_{min}}\cdot\right.\\ &\left.\left(\frac{1}{a_{min}}+\frac{a_{max}}{a_{min}\cdot cap_{min}}+\frac{a_{max}}{d_{min}\cdot cap_{min}}\right)+1\right), \text{by } (24).\end{aligned}$$

Considering the capacity constraint (3), the capacity violation in the solution is bounded by ξ. Hence, the lemma follows. □

Theorem 2. *Given an MEC network $G=(V,E)$, there is an online algorithm `Algorithm` 1 for the VNF service reliability problem under the on-site scheme, which delivers a solution with a $(1+a_{max})$-competitive ratio at the expense of moderate computing capacity violation on any cloudlet bounded by ξ. The algorithm takes $O(|\mathbb{R}|\cdot|C|)$ time, where $a_{max}=\max_{\rho_i\in\mathbb{R},c_j\in C}\{N_{ij}\cdot c(f_i)\}$, ξ is defined in Lemma 5, and $\mathbb{R}$ is the set of requests arrived during the monitoring period of $\mathbb{T}$.*

Proof. Let OPT_1, OPT_2 and OPT_3 be the optimal solutions of **P1**, **P2**, and **P3**, respectively. Following Lemma 1, we have $P_{on}\ge\frac{D_{on}}{1+a_{max}}$. By Lemma 2, D_{on} is a feasible solution to **P3** and **P3** is a minimization problem, thus, $D_{on}\ge OPT_3$. By the weak duality property, $OPT_3\ge OPT_2$. Furthermore, $OPT_2\ge OPT_1$ since **P2** is the LP relaxation on **P1**. We thus have

$$P_{on}\ge\frac{D_{on}}{1+a_{max}}\ge\frac{OPT_3}{1+a_{max}}\ge\frac{OPT_2}{1+a_{max}}\ge\frac{OPT_1}{1+a_{max}}.$$

By Lemma 5, the violation of the computing capacity of any cloudlet is upper bounded by ξ.

The time complexity of `Algorithm` 1 is analyzed as follows. Since this is an online algorithm, its running time is proportional to the number of requests it dealt with for the finite time horizon $\mathbb{T}$. The amount of time of admitting a request ρ_i is determined by calculating N_{ij} and $\sum_{t\in\mathbb{T}} T_i[t]\cdot N_{ij}\cdot c(f_i)\cdot\lambda_{tj}$ for each cloudlet $c_j\in C$. Selecting the cloudlet with $\min_{c_j\in C}\{\sum_{t\in\mathbb{T}} T_i[t]\cdot N_{ij}\cdot c(f_i)\cdot\lambda_{tj}\}$ takes time of $O(|C|)$. The variable updating takes $O(1)$ time. As there are $|\mathbb{R}|$ incoming requests for the monitoring period, the running time of `Algorithm` 1 is $O(|\mathbb{R}|\cdot|C|)$. The theorem then follows. □

5 Online Algorithm for the VNF Service Reliability Problem Under the Off-Site Scheme

In this section, we deal with the VNF service reliability problem under the off-site scheme, where the VNF instances of each request can be placed to different cloudlets, while each cloudlet can host at most one VNF instance of the request.

In the following, we first formulate an ILP solution for the offline version of the problem. We then devise an online algorithm for the problem, based on the linear relaxation of the ILP formulation.

5.1 Integer Linear Programming Formulation

To admit request ρ_i, we need to identify which cloudlets among all cloudlets $c_j\in C$ to accommodate the primary and backup VNF instances of ρ_i to meet its reliability requirement R_i, i.e.,

$$1-\prod_{c_j\in C}(1-r(f_i)\cdot r(c_j)\cdot Y_{ij})\ge R_i. \tag{25}$$

Recall Y_{ij} is a binary variable, denoting whether at least one VNF instance of request ρ_i will be placed to cloudlet $c_j\in C$.

If $Y_{ij} = 1$, the failure probability of a VNF instance of request ρ_i at cloudlet c_j is $1 - r(f_i) \cdot r(c_j) \cdot Y_{ij} = 1 - r(f_i) \cdot r(c_j)$, or $1 - r(f_i) \cdot r(c_j) \cdot Y_{ij} = 1$ otherwise. Thus, $\prod_{c_j \in C}(1 - r(f_i) \cdot r(c_j) \cdot Y_{ij})$ is the probability that both the primary and backup VNF instances of ρ_i fail, while $1 - \prod_{c_j \in C}(1 - r(f_i) \cdot r(c_j) \cdot Y_{ij})$ is the reliability of ρ_i to be admitted. However, if ρ_i is rejected, Y_{ij} should be 0 for each cloudlet $c_j \in C$, i.e., no VNF instance of request ρ_i will be placed to any cloudlet c_j, and Inequality (25) does not hold.

We now modify Inequality (25) through a transformation, using the following technique,

$$R_i \cdot X_i \leq 1 - \prod_{c_j \in C}(1 - r(f_i) \cdot r(c_j) \cdot Y_{ij}) \leq X_i. \tag{26}$$

Lemma 6. *Inequality (26) meets the reliability requirement of request ρ_i if it is admitted.*

Proof. If $X_i = 1$, request ρ_i is admitted, Inequality (26) is shown as follows.

$$R_i \leq 1 - \prod_{c_j \in C}(1 - r(f_i) \cdot r(c_j) \cdot Y_{ij}) \leq 1. \tag{27}$$

As $0 < 1 - r(f_i) \cdot r(c_j) \cdot Y_{ij} \leq 1$, it can be seen that inequalities (27) and (25) are equivalent, satisfying the reliability requirement of request ρ_i. Otherwise ($X_i = 0$), request ρ_i will be rejected and

$$1 \leq \prod_{c_j \in C}(1 - r(f_i) \cdot r(c_j) \cdot Y_{ij}) \leq 1. \tag{28}$$

It can be seen that $\prod_{c_j \in C}(1 - r(f_i) \cdot r(c_j) \cdot Y_{ij}) = 1$, since $0 < r(f_i) < 1$ and $0 < r(c_j) < 1$, we have $Y_{ij} = 0$ for each cloudlet $c_j \in C$, i.e., if request ρ_i is rejected, none of its VNF instances will be placed in any cloudlet. The lemma thus follows. □

Unlike the solution to the problem under the on-site scheme, it is noted that the primal-dual dynamic updating technique cannot be applied to the problem under the off-site scheme directly due to the fact that Inequality (26) is nonlinear. However, we will convert this nonlinear inequality into an equivalent linear one through the following non-trivial equivalent transformation. We then formulate an ILP solution for the VNF service reliability problem under the off-site scheme. Specifically, from Inequality (25), we have

$$\prod_{c_j \in C}(1 - r(f_i) \cdot r(c_j) \cdot Y_{ij}) \leq 1 - R_i, \tag{29}$$

since $0 < r(c_i) < 1$, $0 < r(c_j) < 1$, $0 < R_i < 1$ and $Y_{ij} \in \{0, 1\}$, then $\prod_{c_j \in C}(1 - r(f_i) \cdot r(c_j) \cdot Y_{ij}) > 0$, and $1 - R_i > 0$. Inequality (29) can then be transformed to the following Inequality.

$$\ln \prod_{c_j \in C}(1 - r(f_i) \cdot r(c_j) \cdot Y_{ij}) \leq \ln(1 - R_i). \tag{30}$$

Lemma 7. *Let $Y_{ij} \in \{0, 1\}$, we have*

$$\ln(1 - r(f_i) \cdot r(c_j) \cdot Y_{ij}) = \ln(1 - r(f_i) \cdot r(c_j)) \cdot Y_{ij}.$$

Proof. Because $0 < 1 - r(f_i) \cdot r(c_j)) < 1$ and Y_{ij} is the decision variable, i.e., $Y_{ij} \in \{0, 1\}$, we have two cases to deal with. That is,

if $Y_{ij} = 0$, we have

$$\ln(1 - r(f_i) \cdot r(c_j) \cdot Y_{ij}) = \ln 1 = 0$$
$$= \ln(1 - r(f_i) \cdot r(c_j)) \cdot 0 = \ln(1 - r(f_i) \cdot r(c_j)) \cdot Y_{ij}.$$

Otherwise ($Y_{ij} = 1$), we have

$$\ln(1 - r(f_i) \cdot r(c_j) \cdot Y_{ij}) = \ln(1 - r(f_i) \cdot r(c_j))$$
$$= \ln(1 - r(f_i) \cdot r(c_j)) \cdot Y_{ij}.$$

Hence, the lemma follows. □

By Lemma 7 and Inequality (30), we have

$$\sum_{c_j \in C} \ln(1 - r(f_i) \cdot r(c_j)) \cdot Y_{ij} \leq \ln(1 - R_i). \tag{31}$$

Inequality (31) then can be rewritten as follows.

$$L \cdot X_i \leq \sum_{c_j \in C} \ln(1 - r(f_i) \cdot r(c_j)) \cdot Y_{ij} \leq \ln(1 - R_i) \cdot X_i, \tag{32}$$

where $L = \min_{\rho_i \in \mathbb{R}}\{\sum_{c_j \in C} \ln(1 - r(f_i) \cdot r(c_j))\}$ is negative, which is a lower bound on $\sum_{c_j \in C} \ln(1 - r(f_i) \cdot r(c_j)) \cdot Y_{ij}$.

Lemma 8. *Inequality (32) meets the reliability requirement R_i if request ρ_i is admitted; otherwise, no VNF instance of the request will be placed to any cloudlet, where $L = \min_{\rho_i \in \mathbb{R}}\{\sum_{c_j \in C} \ln(1 - r(f_i) \cdot r(c_j))\}$.*

Proof. We show the claim as follows.

If $X_i = 1$, request ρ_i is admitted, Inequality (32) can be rewritten as follows.

$$L \leq \sum_{c_j \in C} \ln(1 - r(f_i) \cdot r(c_j)) \cdot Y_{ij} \leq \ln(1 - R_i).$$

It can be seen that $L \leq \sum_{c_j \in C} \ln(1 - r(f_i) \cdot r(c_j)) \cdot Y_{ij}$ always holds, and $\sum_{c_j \in C} \ln(1 - r(f_i) \cdot r(c_j)) \cdot Y_{ij} \leq \ln(1 - R_i)$ by Inequality (31).

Otherwise ($X_i = 0$), request ρ_i is rejected. We have

$$0 \leq \sum_{c_j \in C} \ln(1 - r(f_i) \cdot r(c_j)) \cdot Y_{ij} \leq 0$$
$$\Rightarrow \sum_{c_j \in C} \ln(1 - r(f_i) \cdot r(c_j)) \cdot Y_{ij} = 0.$$

Since $0 < r(f_i) < 1$ and $0 < r(c_j) < 1$, we have $Y_{ij} = 0$ for each cloudlet $c_j \in C$, i.e., no VNF instance of ρ_i will be placed to any cloudlet in C. The lemma then follows. □

The VNF service reliability problem under the off-site scheme thus can be formulated as an ILP with the optimization objective to

$$\textbf{P4:} \quad \text{maximize} \quad \sum_{\rho_i \in \mathbb{R}} X_i \cdot pay_i,$$

subject to

$$\sum_{\rho_i \in \mathbb{R}} T_i[t] \cdot c(f_i) \cdot Y_{ij} \leq cap_j, \quad \forall t \in \mathbb{T}, \forall c_j \in C, \tag{33}$$

$$\sum_{c_j \in C} \ln(1 - r(f_i) \cdot r(c_j)) \cdot Y_{ij} \leq \ln(1 - R_i) \cdot X_i, \forall \rho_i \in \mathbb{R}, \tag{34}$$

$$\sum_{c_j \in C} \ln(1 - r(f_i) \cdot r(c_j)) \cdot Y_{ij} \geq L \cdot X_i,\ \forall \rho_i \in \mathbb{R}, \tag{35}$$

$$X_i \in \{0, 1\}, \quad \forall \rho_i \in \mathbb{R}, \tag{36}$$

$$Y_{ij} \in \{0, 1\}, \quad \forall \rho_i \in \mathbb{R}, \forall c_j \in C, \tag{37}$$

where Constraint (33) is the computing capacity constraint on each cloudlet. Constraints (34) and (35) are the reliability constraints on each request by Inequality (32).

5.2 Online Algorithm

Recall that **P4** is referred to as the VNF service reliability problem under the off-site scheme. The strategy adopted for solving **P4** is similar to the one for **P1**. That is, we first perform the LP relaxation on **P4**, and denote this linear relaxation as **P5**. Let **P6** be the dual of **P5**. A feasible solution to **P6** ultimately returns a feasible solution to **P4** with performance guarantees.

The LP relaxation **P5** of **P4** is expressed as follows.

$$\mathbf{P5:} \quad \text{Maximize} \quad \sum_{\rho_i \in \mathbb{R}} X_i \cdot pay_i,$$

subject to

$$\begin{aligned} &(33),\ (34),\ (35), \\ &X_i \leq 1, \quad \forall \rho_i \in \mathbb{R}, \end{aligned} \tag{38}$$

$$Y_{ij} \leq X_i, \quad \forall \rho_i \in \mathbb{R},\ \forall c_j \in C \tag{39}$$

$$X_i \geq 0,\ Y_{ij} \geq 0, \quad \forall \rho_i \in \mathbb{R},\ \forall c_j \in C. \tag{40}$$

Notice that Constraint (39) always holds, because if request ρ_i is admitted ($X_i = 1$), we could choose to place a VNF instance in cloudlet c_j or not (i.e., Y_{ij} could be 0 or 1); otherwise (request ρ_i is rejected and $X_i = 0$), no VNF instance of ρ_i will be placed to cloudlet c_j, i.e., Y_{ij} must be 0. Also, Constraint (39) implicitly ensures that $Y_{ij} \leq 1$ with Constraint (38).

The rest is to solve **P5**, and any feasible solution for **P4** is a feasible solution for **P5**. Let **P6** be the dual of **P5**, which can be written as follows.

$$\mathbf{P6:} \quad \text{Minimize} \quad \sum_{t \in \mathbb{T}, c_j \in C} cap_j \cdot A_{tj} + \sum_{\rho_i \in \mathbb{R}} K_i, \tag{41}$$

subject to

$$-\ln(1 - R_i) \cdot B_i + L \cdot H_i + K_i - \sum_{c_j \in C} M_{ij} \geq pay_i, \forall \rho_i \in \mathbb{R}, \tag{42}$$

$$\begin{aligned} &\sum_{t \in \mathbb{T}} T_i[t] \cdot c(f_i) \cdot A_{tj} + \ln(1 - r(f_i) \cdot r(c_j)) \cdot (B_i - H_i) + M_{ij} \geq 0, \\ &\forall \rho_i \in \mathbb{R}, \forall c_j \in C, \end{aligned} \tag{43}$$

$$\begin{aligned} &A_{tj} \geq 0, B_i \geq 0, H_i \geq 0, K_i \geq 0, M_{ij} \geq 0, \\ &\forall \rho_i \in \mathbb{R}, \forall c_j \in C, \forall t \in \mathbb{T}, \end{aligned} \tag{44}$$

where A_{tj}, B_i, H_i, K_i and M_{ij} are the dual variables for constraints (33), (34), (35), (38) and (39) in **P5**, respectively.

Let us examine Constraint (42) first. Because L is a lower bound of $\sum_{c_j \in C}(\ln(1 - r(f_i) \cdot r(c_j))) \cdot Y_{ij}$, it can be treated as $-\infty$ in an extreme case.

To satisfy Constraint (42), with $H_i \geq 0$, we set $H_i = 0$, $\forall \rho_i \in \mathbb{R}$. Then, constraints (42) and (43) can be rewritten as follows.

$$\sum_{c_j \in C} M_{ij} \leq -\ln(1 - R_i) \cdot B_i + K_i - pay_i, \tag{45}$$

$$M_{ij} \geq -\sum_{t \in \mathbb{T}} T_i[t] \cdot c(f_i) \cdot A_{tj} - \ln(1 - r(f_i) \cdot r(c_j)) \cdot B_i. \tag{46}$$

From Inequality (45), with $M_{ij} \geq 0$, we have

$$-\ln(1 - R_i) \cdot B_i + K_i - pay_i \geq 0. \tag{47}$$

Then,

$$B_i \geq \frac{pay_i - K_i}{-\ln(1 - R_i)}. \tag{48}$$

Similarly, with $B_i \geq 0$ and $-\ln(1 - R_i) > 0$, $(0 < R_i < 1)$, we have

$$K_i \leq pay_i. \tag{49}$$

From Inequality (46), we have

$$\sum_{c_j \in C} M_{ij} \geq -\sum_{\substack{c_j \in C, \\ t \in \mathbb{T}}} T_i[t] \cdot c(f_i) \cdot A_{tj} - \sum_{c_j \in C} \ln(1 - r(f_i) \cdot r(c_j)) \cdot B_i. \tag{50}$$

Combining Inequalities (45) and (50), we have

$$\begin{aligned} K_i \geq\ & pay_i - \sum_{\substack{c_j \in C, \\ t \in \mathbb{T}}} T_i[t] \cdot c(f_i) \cdot A_{tj} \\ &+ (\ln(1 - R_i) - \sum_{c_j \in C} \ln(1 - r(f_i) \cdot r(c_j)) \cdot B_i. \end{aligned} \tag{51}$$

As we assume that the service reliability of any request can be met with at most $|C|$ VNF instances it requested (i.e., $\ln(1 - R_i) - \sum_{c_j \in C} \ln(1 - r(f_i) \cdot r(c_j)) \geq 0, \quad \forall c_j \in C$), by Inequalities (48) and (51), we have

$$K_i \geq pay_i - \frac{\ln(1 - R_i) \cdot c(f_i)}{\sum_{c_j \in C} \ln(1 - r(f_i) \cdot r(c_j))} \cdot \sum_{\substack{c_j \in C, \\ t \in \mathbb{T}}} T_i[t] \cdot A_{tj}. \tag{52}$$

Lemma 9. *If Inequalities (49) and (52) hold, there always exist feasible values for B_i and M_{ij} to ensure that constraints (42) and (43) hold.*

Proof. Because $T_i[t] \geq 0,\ c(f_i) > 0,\ A_{tj} \geq 0,\ 0 < r(f_i) < 1,$ $0 < r(c_j) < 1$ and $c < R_i < 1, \forall \rho_i \in \mathbb{R},\ \forall c_j \in C,\ \forall t \in \mathbb{T}$, then $pay_i - \frac{\ln(1 - R_i)}{\sum_{c_j \in C} \ln(1 - r(f_i) \cdot r(c_j))} \sum_{\substack{c_j \in C, \\ t \in \mathbb{T}}} T_i[t] \cdot c(f_i) \cdot A_{tj} \leq pay_i$.

Thus, Inequality (49) and (52) complies with each other.

As mentioned above, if Inequality (49) holds, the feasible values for B_i, can be found with Inequality (48). Then, combining (52), the feasible value for M_{ij} can be found to satisfy Inequalities (45) and (46). Thus, Constraints (42) and (43) hold, the lemma then follows. □

Algorithm 2. An Online Algorithm for the VNF Service Reliability Problem Under the Off-Site Scheme

```
Input: An MEC network $G = (V, E)$ and requests arrive one by one without the knowledge of future arrivals.
Output: An online scheduling of the incoming requests.
1: $X_i \leftarrow 0; Y_{ij} \leftarrow 0; A_{tj} \leftarrow 0; K_i \leftarrow 0, \forall \rho_i \in \mathbb{R},\ \forall c_j \in C,\ \forall t \in \mathbb{T}$;
2: while Upon arrival of a request $\rho_i$ do
3:   if $pay_i - \frac{\ln(1-R_i)c(f_i)}{\sum_{c_j \in C}\ln(1-r(f_i)\cdot r(c_j))}\cdot\sum_{t\in\mathbb{T}, c_j\in C} T_i[t]\cdot A_{tj} > 0$ then
4:     Admit request $\rho_i$;
5:     $X_i \leftarrow 1$;
6:     $K_i \leftarrow pay_i - \frac{\ln(1-R_i)c(f_i)}{\sum_{c_j \in C}\ln(1-r(f_i)\cdot r(c_j))}\cdot\sum_{t\in\mathbb{T}, c_j\in C} T_i[t]\cdot A_{tj}$;
7:     Sort cloudlets in $C$ in non-decreasing order of $\sum_{t\in\mathbb{T}} T_i[t]\cdot A_{tj}$;
8:     $S(i) \leftarrow \emptyset$; /* $S(i)$ is the chosen set of cloudlets for hosting its VNF instances of request $\rho_i$ */
9:     while consider cloudlet $c_j$ in the sorted cloudlet sequence do
10:      $S(i) \leftarrow S(i) \cup \{c_j\}$;
11:      $Y_{ij} \leftarrow 1$;
12:      $A_{tj} \leftarrow A_{tj}\cdot\left(1 + \frac{\ln(1-R_i)\cdot c(f_i)}{cap_j\cdot\sum_{c_j\in C}\ln(1-r(f_i)\cdot r(c_j))}\right) + \frac{\ln(1-R_i)\cdot c(f_i)\cdot pay_i}{cap_j\cdot d_i\cdot\sum_{c_j\in C}\ln(1-r(f_i)\cdot r(c_j))}$;
13:      if the VNF instance placements in cloudlets of $S(i)$ is no less than $R_i$ then
14:        EXIT;
15:      end if ;
16:    end while ;
17:  else
18:    Reject request $\rho_i$;
19:  end if ;
20: end while .
```

By Lemma 9, we consider Inequalities (49) and (52), and the two dual variables $A_{tj} \geq 0$ and $K_i \geq 0$ in order to solve **P6**. We then update the variables in both the primal and dual LP simultaneously, by adopting the primal-dual updating technique in the previous section.

The detailed online algorithm for **P4** is given in `Algorithm` 2. Specifically, for an incoming request ρ_i, if $pay_i - \frac{\ln(1-R_i)c(f_i)}{\sum_{c_j\in C}\ln(1-r(f_i)\cdot r(c_j))}\cdot\sum_{t\in\mathbb{T}, c_j\in C} T_i[t]\cdot A_{tj} \leq 0$ for any cloudlet c_j, request ρ_i will be rejected; otherwise, it will be admitted, and the value of K_i is updated as follows.

$$K_i := pay_i - \frac{\ln(1-R_i)\cdot c(f_i)}{\sum_{c_j\in C}\ln(1-r(f_i)\cdot r(c_j))}\cdot\sum_{\substack{t\in\mathbb{T},\\ c_j\in C}} T_i[t]\cdot A_{tj}. \tag{53}$$

The rest is to deal with the placements of the $\sum_{j=1}^{|C|} Y_{ij}$ VNF instances for request ρ_i to which cloudlets assuming $\sum_{j=1}^{|C|} Y_{ij} \leq |C|$.

To this end, we first sort the cloudlets in non-decreasing order of $\sum_{t\in\mathbb{T}} T_i[t]\cdot A_{tj}$. We then identify a set of cloudlets for hosting the VNF instances of request ρ_i in their sorted order until there exists a scheduling such that the reliability requirement of the request is met. We finally put one VNF instance to each of the chosen cloudlets and update the value of A_{tj} of each chosen cloudlet $c_j \in C$ and $t \in \mathbb{T}'_i$ as follows.

$$A_{tj} := A_{tj}\cdot\left(1 + \frac{\ln(1-R_i)\cdot c(f_i)}{cap_j\cdot\sum_{c_j\in C}\ln(1-r(f_i)\cdot r(c_j))}\right) + \frac{\ln(1-R_i)\cdot c(f_i)\cdot pay_i}{cap_j\cdot d_i\cdot\sum_{c_j\in C}\ln(1-r(f_i)\cdot r(c_j))}, \tag{54}$$

where $\mathbb{T}'_i$ is the set of the execution time slots of request ρ_i and d_i is the execution duration. Denote by $S(i)$ the set of the chosen cloudlets when request ρ_i is admitted, and let $b_i = \frac{\ln(1-R_i)\cdot c(f_i)\cdot|S(i)|}{\sum_{c_j\in C}\ln(1-r(f_i)\cdot r(c_j))}$, $b_{min} = \min_{\rho_i\in\mathbb{R}}\{b_i\}$, and $b_{max} = \max_{\rho_i\in\mathbb{R}}\{b_i\}$, respectively.

Lemma 10. *Let P_{off} and D_{off} be the values of the objective functions delivered by the proposed algorithm for* **P5** *and* **P6**, *respectively, Then, $(1 + b_{max})\cdot P_{off} \geq D_{off}$.*

Proof. We show the claim as follows. $P_{off} = D_{off} = 0$ initially, the claim holds.

We prove the claim by showing that $(1 + b_{max})\cdot\Delta P_{off} \geq \Delta D_{off}$ still holds after a request ρ_i arrives, where ΔP_{off} and ΔD_{off} are the value differences of the objective functions for **P5** and **P6** before and after the request arrives, as follows.

If request ρ_i will be rejected, $\Delta P_{off} = \Delta D_{off} = 0$, and $(1 + b_{max})\cdot\Delta P_{off} \geq \Delta D_{off}$. Otherwise, $\Delta P_{off} = pay_i$ and $\Delta D_{off} = \sum_{t\in\mathbb{T}'_i, c_j\in S(i)} cap_j\cdot\Delta A_{tj} + K_i$, by (41), where ΔA_{tj} is the difference before and after the update to A_{tj}, which is associated with the chosen cloudlet c_j. From the update function (54) of A_{tj}, we have

$$\begin{aligned}
\Delta D_{off} &= \sum_{t\in\mathbb{T}'_i, c_j\in S(i)} cap_j\cdot\Delta A_{tj} + K_i \\
&= \sum_{\substack{t\in\mathbb{T}'_i,\\ c_j\in S(i)}} cap_j\cdot\left(\frac{\ln(1-R_i)\cdot c(f_i)\cdot A_{ij}}{cap_j\cdot\sum_{c_j\in C}\ln(1-r(f_i)\cdot r(c_j))}\right. \\
&\qquad \left. + \frac{\ln(1-R_i)\cdot c(f_i)\cdot pay_i}{cap_j\cdot d_i\cdot\sum_{c_j\in C}\ln(1-r(f_i)\cdot r(c_j))}\right) + K_i \\
&= \left(\frac{\ln(1-R_i)\cdot c(f_i)}{\sum_{c_j\in C}\ln(1-r(f_i)\cdot r(c_j))}\cdot\sum_{\substack{t\in\mathbb{T},\\ c_j\in S(i)}} T_i[t]\cdot A_{tj}\right. \\
&\qquad \left. + \sum_{\substack{t\in\mathbb{T}'_i,\\ c_j\in S(i)}} \frac{\ln(1-R_i)\cdot c(f_i)\cdot pay_i}{d_i\cdot\sum_{c_j\in C}\ln(1-r(f_i)\cdot r(c_j))}\right) + K_i \\
&= \frac{\ln(1-R_i)\cdot c(f_i)}{\sum_{c_j\in C}\ln(1-r(f_i)\cdot r(c_j))}\cdot\sum_{\substack{t\in\mathbb{T},\\ c_j\in S(i)}} T_i[t]\cdot A_{tj} \\
&\qquad + K_i + \frac{\ln(1-R_i)\cdot c(f_i)\cdot|S(i)|}{\sum_{c_j\in C}\ln(1-r(f_i)\cdot r(c_j))}\cdot pay_i \\
&\leq pay_i + b_i\cdot pay_i \\
&\leq (1 + b_{max})\cdot pay_i \\
&= (1 + b_{max})\cdot\Delta P_{off}.
\end{aligned} \tag{55}$$

Notice that Inequality (55) holds, because we have $\frac{\ln(1-R_i)\cdot c(f_i)}{\sum_{c_j\in C}\ln(1-r(f_i)\cdot r(c_j))}\cdot\sum_{t\in\mathbb{T},c_j\in C}T_i[t]\cdot A_{tj}+K_i=pay_i$ from the update function (53), and $S(i)\subseteq C$. Hence, the lemma follows. □

Lemma 11. *`Algorithm` 2 delivers a feasible solution for* ***P6***.

Proof. Following `Algorithm` 2, Inequalities (49) and (52) hold, by the update function (53) of K_i when a new request ρ_i arrives. Since the update function (54) of A_{tj} is non-decreasing, $pay_i-\frac{\ln(1-R_i)\cdot c(f_i)}{\sum_{c_j\in C}\ln(1-r(f_i)\cdot r(c_j))}\cdot\sum_{t\in\mathbb{T},c_j\in C}T_i[t]\cdot A_{tj}$ is non-increasing. Inequalities (49) and (52) will still hold when updating the value of A_{tj}. The lemma then follows. □

Denote by $z_i=\frac{\ln(1-R_i)\cdot c(f_i)}{\sum_{c_j\in C}\ln(1-r(f_i)\cdot r(c_j))}$, $z_{min}=\min_{\rho_i\in\mathbb{R}}\{z_i\}$, and $z_{max}=\max_{\rho_i\in\mathbb{R}}\{z_i\}$.

Lemma 12.

$$A_{tj}\geq\frac{pay_{min}}{d_{max}}\cdot\left(\left(1+\frac{z_{min}}{cap_{max}}\right)^{\sum_{\rho_i\in\mathbb{R}}T_i[t]\cdot Y_{ij}}-1\right).\tag{56}$$

The proof can be found in Appendix C, available in the online supplemental material.

Lemma 13.

$$A_{tj}<\frac{pay_{max}}{z_{min}}\cdot\left(1+\frac{z_{max}}{cap_{min}}\right)+\frac{z_{max}\cdot pay_{max}}{d_{min}\cdot cap_{min}}.\tag{57}$$

The proof can be found in Appendix D, available in the online supplemental material.

Lemma 14. *In the solution for* ***P5*** *delivered by `Algorithm` 2, the violation on the capacity constraint of each cloudlet is upper bounded by* Λ, *where* $\Lambda=\frac{c(f)_{max}}{cap_{min}\cdot\ln(1+\frac{z_{min}}{cap_{max}})}\cdot\ln(\frac{pay_{max}\cdot d_{max}}{pay_{min}}\cdot(\frac{1}{z_{min}}+\frac{z_{max}}{z_{min}\cdot cap_{min}}+\frac{z_{max}}{d_{min}\cdot cap_{min}})+1)$, *and* $c(f)_{max}=\max_{\rho_i\in\mathbb{R}}\{c(f_i)\}$.

Proof. Combining Lemmas 12 and 13, we have

$$\frac{pay_{min}}{d_{max}}\cdot\left(\left(1+\frac{z_{min}}{cap_{max}}\right)^{\sum_{\rho_i\in\mathbb{R}}T_i[t]\cdot Y_{ij}}-1\right)<\frac{pay_{max}}{z_{min}}\cdot\left(1+\frac{z_{max}}{cap_{min}}\right)+\frac{z_{max}\cdot pay_{max}}{d_{min}\cdot cap_{min}}$$

We then have

$$\sum_{\rho_i\in\mathbb{R}}T_i[t]\cdot Y_{ij}<\frac{\ln\left(\frac{pay_{max}d_{max}}{pay_{min}}\cdot\left(\frac{1}{z_{min}}+\frac{z_{max}}{z_{min}cap_{min}}+\frac{z_{max}}{d_{min}cap_{min}}\right)+1\right)}{\ln\left(1+\frac{z_{min}}{cap_{max}}\right)}.\tag{58}$$

To calculate the capacity violation, we have

$$\begin{aligned}&\sum_{\rho_i\in\mathbb{R}}T_i[t]\cdot c(f_i)\cdot Y_{ij}\\&\leq\sum_{\rho_i\in\mathbb{R}}T_i[t]\cdot c(f)_{max}\cdot Y_{ij}\\&\leq\frac{c(f)_{max}}{\ln(1+\frac{z_{min}}{cap_{max}})}\cdot\ln\Bigg(\frac{pay_{max}\cdot d_{max}}{pay_{min}}\cdot\\&\left(\frac{1}{z_{min}}+\frac{z_{max}}{z_{min}\cdot cap_{min}}+\frac{z_{max}}{d_{min}\cdot cap_{min}}\right)+1\Bigg),\text{ by (58).}\end{aligned}$$

Considering the capacity constraint (33), the capacity violation on any cloudlet is upper bounded by Λ. Hence, the lemma follows. □

Theorem 3. *Given an MEC network* $G=(V,E)$, *there is an online algorithm, `Algorithm` 2, for the VNF service reliability problem under the off-site scheme, which delivers a solution with a* $(1+b_{max})$*-competitive ratio while the violation of the computing capacity on any cloudlet is upper bounded by* Λ. *The algorithm takes* $O(|\mathbb{R}|\cdot|C|\cdot\log|C|)$ *time, where* $b_{max}=\max_{\rho_i\in\mathbb{R},c_j\in C}\{\frac{\ln(1-R_i)\cdot c(f_i)\cdot|S(i)|}{\sum_{c_j\in C}\ln(1-r(f_i)\cdot r(c_j))}\}$, $\mathbb{R}$ *is the number of requests arrived during a finite monitoring period of* $\mathbb{T}$, *and* Λ *is defined in Lemma 14.*

Proof. Let OPT_4, OPT_5 and OPT_6 be the optimal solutions to **P4**, **P5**, and **P6**, respectively. By Lemma 10, $P_{off}\geq\frac{D_{off}}{1+b_{max}}$. Meanwhile, By Lemma 11, D_{off} is a feasible solution to **P6**, and **P6** is a minimization problem, thus, $D_{off}\geq OPT_6$. By the weak duality, $OPT_6\geq OPT_5$. Also, $OPT_5\geq OPT_4$ because **P5** is the LP relaxation of **P4**. We then have

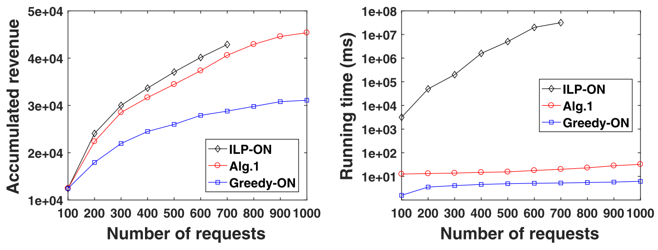
$$P_{off}\geq\frac{D_{off}}{1+b_{max}}\geq\frac{OPT_6}{1+b_{max}}\geq\frac{OPT_5}{1+b_{max}}\geq\frac{OPT_4}{1+b_{max}}.$$

The violation of the computing capacity of any cloudlet is upper bounded by Λ, following Lemma 14.

The time complexity of `Algorithm` 2 is analyzed as follows. Since this is an online algorithm, its running time is proportional to the number of requests dealt with for the finite monitoring period $\mathbb{T}$. The amount of time taken for admitting a request ρ_i is dominated by sorting cloudlets in non-decreasing order of $\sum_{t\in\mathbb{T}}T_i[t]\cdot A_{tj}$, which is $O(|C|\cdot\log|C|)$, where C is the set of cloudlets in G. Selecting the cloudlet with the sorted cloudlet sequence takes time of $O(|C|)$. The updating of variables takes $O(1)$ time. As there are $|\mathbb{R}|$ incoming requests, the running time of `Algorithm` 2 is $O(|\mathbb{R}|\cdot|C|\cdot\log|C|)$. The theorem then follows. □

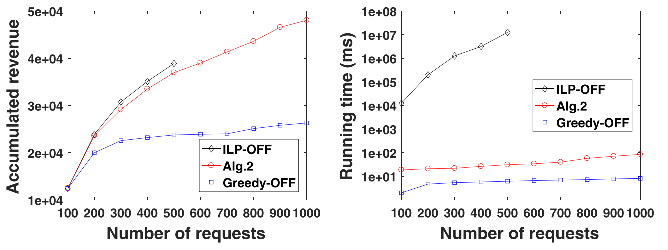
6 Performance Evaluation

In this section, we study the performance of the proposed algorithms for the VNF service reliability problem under both on-site and off-site schemes. We also investigate the impact of important parameters on the performance of the proposed algorithms.



(a) The accumulated revenue (b) The running time

Fig. 2. Performance of different algorithms under the on-site scheme by varying the number of requests from 100 to 1,000.



(a) The accumulated revenue (b) The running time

Fig. 3. Performance of different algorithms under the off-site scheme by varying the number of requests from 100 to 1,000.

6.1 Environment Settings

We consider an MEC network $G = (V, E)$ consisting of 100 APs and 10 percent of APs are co-located with cloudlets [18]. Each network is generated by the widely used tool GT-ITM [9]. The computing capacity of each cloudlet is randomly drawn from 2 GHz to 6 GHz [14]. while the reliability of each cloudlet is a value randomly drawn from 0.99999 to 0.999999 [10]. We assume that there are 10 types of VNFs offered by the network service provider with each having a reliability between 0.9 and 0.99 and the amounts of computing resource demanded for the implementation of a VNF is ranged from 40 MHz to 400 MHz [13]. For an incoming request, a random VNF in F is requested, and its reliability requirement is randomly drawn from 0.9999 to 0.99999 [10]. The running time of each algorithm is obtained based on a desktop with a 3.4 GHz quad-core Intel i7 CPU and 16 GB RAM. These parameters are adopted as the default setting unless otherwise specified.

To evaluate the performance of the proposed algorithms, `Algorithm` 1 and `Algorithm` 2 under on-site and off-site schemes, respectively, we introduce one benchmark against the proposed algorithms, which is a greedy algorithm that always admits incoming requests by placing their VNF instances to the cloudlets with high reliabilities. We refer to this algorithm as algorithms `Greedy-ON` and `Greedy-OFF`, respectively. Also, we formulated the Integer Linear Programming (ILP) solution to the problem under both on-site and off-site schemes, and we refer to these algorithms as algorithms `ILP-ON` and `ILP-OFF`, respectively. The exact solutions delivered by these ILP algorithms will be used for the benchmark of the proposed algorithms when the problem size is small.

6.2 Performance Evaluation of Different Algorithms

We first evaluate the performance of `Algorithm` 1 against algorithm `Greedy-ON` and the optimal solution `ILP-ON` for the problem under the on-site scheme, by varying the number of requests from 100 to 1,000. We also evaluate the performance of `Algorithm` 2 against algorithm `Greedy-OFF` and the optimal solution `ILP-OFF` for the problem under the off-site scheme, by varying the number of requests from 100 to 1,000. Figs. 2 and 3 depict the accumulated revenues and running times of different algorithms under both on-site and off-site schemes, respectively. Fig. 4 depicts the maximum computing resource violation ratios of `Algorithm` 1 and `Algorithm` 2, where the computing resource violation ratio is the ratio of the over-consumed computing resource of a cloudlet in a time slot to its computing resource capacity. From Fig. 2a, we can see that the revenue collected by algorithm `Greedy-ON` is 68.5 percent of that of `Algorithm` 1 when the number of requests is 1,000. From Fig. 3a, we can see that the revenue collected by `Greedy-OFF` is 54.7 percent of that of `Algorithm` 2 when the number of requests is 1,000. It can be seen from Figs. 2a and 4 that `Algorithm` 1 achieves 94.6 percent of the optimal one by algorithm `ILP-ON`, but causes the maximum computing resource violation ratio of 8.6 percent when the number of requests is 700. While Fig. 3a indicates that `Algorithm` 2 can achieve 95.1 percent of the optimal one by algorithm `ILP-OFF` but causes the maximum computing resource violation ratio of 10.4 percent when the number of requests is 500. However, from Fig. 2b, we can see that algorithm `ILP-ON` takes a prohibitively long time while `Algorithm` 1 takes a much shorter time. And when the number of requests reaches 800, algorithm `ILP-ON` fails to deliver any solution within a reasonable time. The similar performance behaviors can be found in Fig. 3b, too. From Figs. 2a and 3a, it can be seen that `Algorithm` 1 under the on-site scheme achieves only 93.9 percent of the performance compared with the performance by `Algorithm` 2 under the off-site scheme when the number of requests is 1,000. However, it can be seen from Fig. 4 that `Algorithm` 2 causes more computing resource violation than `Algorithm` 1. The rationale behind is that the off-site scheme utilizes multiple cloudlets to meet the reliability requirements of incoming requests which facilitates the corporation of different cloudlets to maximize the collected revenue at the expense of more computing resource violation.

6.3 Impact of Parameters on the Performance of Different Algorithms

We then evaluate the impact of important parameters on the proposed algorithms by fixing the number of requests at 1,000. We first investigate the impact of various network

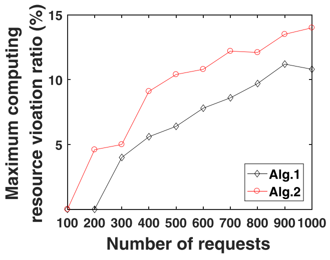


Fig. 4. The maximum computing resource violation ratios of `Algorithm` 1 and `Algorithm` 2 by varying the number of requests from 100 to 1,000.

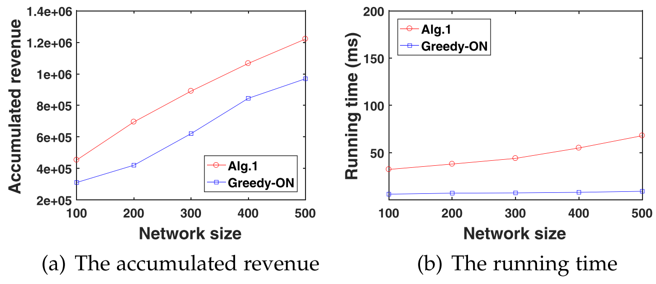


Fig. 5. Performance of different algorithms under the on-site scheme by varying the network size from 100 to 500.

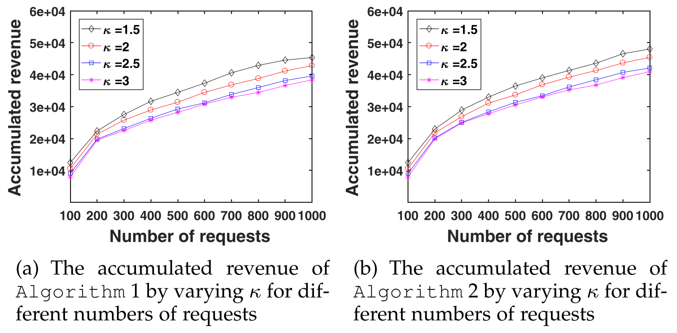


Fig. 7. Performance impact of parameter κ on the proposed algorithms, where κ is the ratio of the maximum payment to the minimum payment among requests.

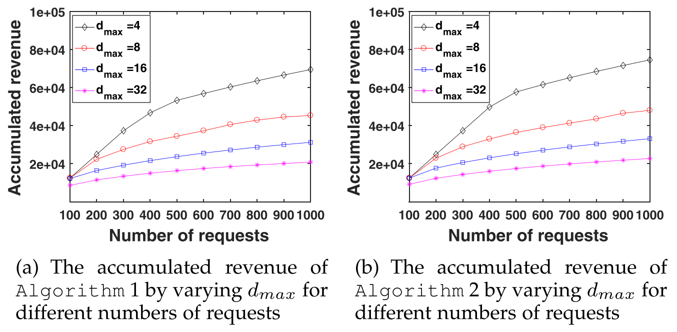


Fig. 8. Performance impact of parameter d_{max} on the proposed algorithms, where d_{max} is the maximum requested execution duration among requests.

sizes on the performance of the proposed algorithms. We then investigate the impact of various request payments on the performance of the proposed algorithms. Denote by $\kappa = \frac{pay_{max}}{pay_{min}}$ the ratio of the maximum payment pay_{max} to the minimum payment pay_{min}. We finally study the impact of the maximum execution duration d_{max} among requests on the performance of the proposed algorithms as follows.

We start with evaluating the impact of the network size on the performance of the proposed algorithms against the benchmarks `Greedy-ON` and `Greedy-OFF` under the on-site scheme and off-site scheme, respectively, by varying the network size from 100 to 500 while fixing the number of requests at 1,000. We also fix the ratio of the number of cloudlets to the network size at 0.1. Fig. 5 depicts the accumulated revenues and running times of different algorithms for the problem under the on-site scheme. While Fig. 6 depicts the accumulated revenues and running times of different algorithms for the problem under the off-site scheme. From Fig. 5a, it can be seen that `Greedy-ON` achieves 79.3 percent of the solution delivered by `Algorithm` 1 with the network size of 500, while `Greedy-ON` achieves 68.5 percent of the solution delivered by `Algorithm` 1 with the network size of 100. This can be justified that with a small number of network size (i.e., limited computing resource), a smarter scheduling strategy is supposed to be deployed to deal with a large number of incoming requests to maximize the collected revenue. The similar performance behaviors can be found in Fig. 6b.

We then study the impact of parameter κ on the performance of the proposed algorithms. Fig. 7 shows the performance impact of κ on the proposed algorithms `Algorithm` 1 and `Algorithm` 2, by varying κ from 1.5 to 3 and the number of requests from 100 to 1,000. As κ is the ratio of the maximum payment pay_{max} to the minimum payment pay_{min}. We varies κ by varying pay_{min} but fixing pay_{max}. E.g., with $\kappa = 1.5$, the payment of each request is randomly drawn from 100 dollars to 150 dollars. While with $\kappa = 2$, the payment of each request is randomly drawn from 75 dollars to 150 dollars. It can be seen from Fig. 7a that when the number of requests is 100, `Algorithm` 1 with $\kappa = 3$ achieves 63.8 percent of the performance of itself with $\kappa = 1.5$. While when the number of requests is 1,000, `Algorithm` 1 with $\kappa = 3$ achieves 84.4 percent of the performance of itself with $\kappa = 1.5$. The rationale behind is that when the number of requests is large, `Algorithm` 1 has better performance in admitting the requests with high payments but low computing resource consumption. The similar performance behaviors can be found in Fig. 7b.

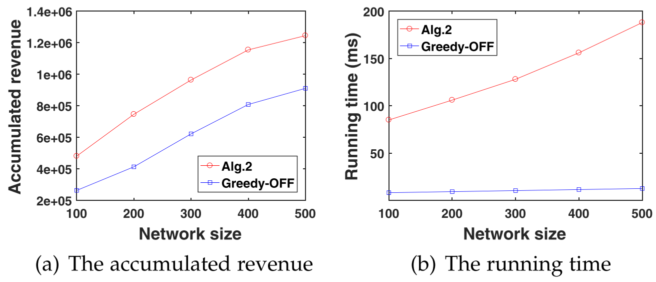


Fig. 6. Performance of different algorithms under the off-site scheme, by varying the network size from 100 to 500.

We finally investigate the impact of parameter d_{max} on the performance of the proposed algorithms. The execution duration of each incoming request is randomly drawn from $[1, d_{max}]$. Fig. 8 shows the performance of the proposed algorithms, by varying d_{max} from 4 to 32 and the number of requests from 100 to 1,000. It can be seen from Figs. 8a and 8b that the accumulated revenue decreases with the growth of the value of d_{max}. This is because users request more execution duration (i.e., more computing resource) while the payments do not change. The similar performance behaviors can be found in Fig. 8b.

7 Conclusion

In this paper, we studied reliability-aware VNF service provisioning for IoT applications in an MEC environment to meet the service reliability requirements of mobile users. We first formulated a novel VNF service reliability problem with the aim to maximize the revenue collected for a given time horizon by admitting as many as user requests, assuming that user requests arrive one by one, and each incoming request must be responded by admitting or rejecting it immediately. We then developed online algorithms with provable competitive ratios for the problem at the expense of moderate computing resource violations, through adopting the primal-dual dynamic updating technique. We finally evaluated the proposed algorithms through experimental simulations. Experimental results demonstrated that the proposed algorithms are promising, and outperform the mentioned benchmark.

Acknowledgments

The authors would like to thank the three anonymous referees for their expertise comments and constructive suggestions, which have helped us improve the presentation of this article greatly. The work by Jing Li and Weifa Liang is supported by the Australian Research Council Discovery Project No. DP200101985. The work by Xiaohua Jia is supported by Research Grants Council of Hong Kong [Project No. CityU11214316].

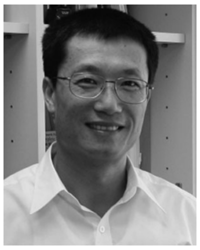
References

[1] M. T. Beck, J. F. Botero, and K. Samelin, "Resilient allocation of service function chains," *Proc. Int. Conf. Netw. Function Virtualization Softw. Defined Netw.*, 2016, pp. 128–133.
[2] H. Chantre and N. Fonseca, "Redundant placement of virtualized network functions for LTE evolved multimedia broadcast multicast services," in *Proc. IEEE Int. Conf. Commun.*, 2017, pp. 1–7.
[3] W. Chen, D. Wang, and K. Li, "Multi-user multi-task computation offloading in green mobile edge cloud computing," *IEEE Trans. Services Comput.*, vol. 12, no. 5, pp. 726–738, Sep./Oct. 2019.
[4] W. Ding, H. Yu, and S. Luo, "Enhancing the reliability of services in NFV with the cost-efficient redundancy scheme," in *Proc. IEEE Int. Conf. Commun.*, 2017, pp. 1–6.
[5] J. Fan, M. Jiang, and C. Qiao, "Carrier-grade availability-aware mapping of service function chains with on-sites," in *Proc. IEEE/ACM 25th Int. Symp. Quality Service*, 2017, pp. 1–10.
[6] Q. Fan and N. Ansari, "Cost aware cloudlet placement for big data processing at the edge," in *Proc. IEEE Int. Conf. Commun.*, 2017, pp. 1–6.
[7] P. Gill, N. Jain, and N. Nagappan, "Understanding network failures in data centers: Measurement, analysis, and implications," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 41, no. 4, pp. 350–361, 2011.
[8] M. X. Goemans and D. P. Williamson, "The primal-dual method for approximation algorithms and its application to network design problems," in *Book Chapter of Approximation Algorithms for NP-Hard Problems*, Boston, MA, USA: PWS, 1997, pp. 144 –191.
[9] GT-ITM, 2019. [Online]. Available: http://www.cc.gatech.edu/projects/gtitm/
[10] B. Han, V. Gopalakrishnan, G. Kathirvel, and A. Shaikh, "On the resiliency of virtual network functions," *IEEE Commun. Mag.*, vol. 55, no. 7, pp. 152–157, Jul. 2017.
[11] A. Hmaity, M. Savi, F. Musumeci, M. Tornatore, and A. Pattavina, "Virtual network function placement for resilient service chain provisioning," in *Proc. 8th Int. Workshop Resilient Netw. Des. Model.*, 2016, pp. 245–252.
[12] M. Hu, D. Wu, W. Wu, J. Cheng, and M. Chen, "Quantifying the influence of intermittent connectivity on mobile edge computing," *IEEE Trans. Cloud Comput.*, to be published, doi: 10.1109/TCC.2019.2926702.
[13] M. Huang, W. Liang, X. Shen, Y. Ma, and H. Kan, "Reliability-aware virtualized network function services provisioning in mobile edge computing," *IEEE Trans. Mobile Comput.*, to be published, doi: 10.1109/TMC.2019.2927214.
[14] M. Jia, W. Liang, and Z. Xu, "QoS-aware task offloading in distributed cloudlets with virtual network function services," in *Proc 20th Int. Conf. Model. Anal. Simul. Wireless Mobile Syst.*, 2017, pp. 109–116.
[15] Y. Kanizo, O. Rottenstreich, I. Segall, and J. Yallouz, "Optimizing virtual backup allocation for middleboxes," *IEEE/ACM Trans. Netw.*, vol. 25, no. 5, pp. 2759–2772, Oct. 2017.
[16] J. Kong *et al.*, "Guaranteed-availability network function virtualization with network protection and VNF replication," in *Proc IEEE Global Commun. Conf.*, 2017, pp. 1–6.
[17] J. Li, W. Liang, M. Huang, and X. Jia, "Providing reliability-aware virtualized network function services for mobile edge computing," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst.*, 2019, pp. 732–741.
[18] Y. Ma, W. Liang, Z. Xu, and S. Guo, "Profit maximization for admitting requests with network function services in distributed clouds," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 5, pp. 1143–1157, May 2019.
[19] J. Pan and J. McElhannon, "Future edge cloud and edge computing for internet of things applications," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 439–449, Feb. 2018.
[20] Z. Qiu and J. Pérez, "Enhancing reliability and response times via replication in computing clusters," in *Proc. IEEE Conf. Comput. Commun.*, 2015, pp. 1355–1363.
[21] Y. Sang, B. Ji, G. Gupta, X. Du, and L. Ye, "Provably efficient algorithms for joint placement and allocation of virtual network functions," in *Proc IEEE Conf. Comput. Commun.*, 2017, pp. 1–9.
[22] Z. Xu, W. Liang, M. Jia, M. Huang, and G. Mao, "Task offloading with network function services in a mobile edge-cloud network," *IEEE Trans. Mobile Comput.*, vol. 18, no. 11, pp. 2672–2685, Nov. 2019.
[23] F. Wang, J. Xu, X. Wang, and S. Cui, "Joint offloading and computing optimization in wireless powered mobile-edge computing systems," in *Proc. IEEE Int. Conf. Commun.*, 2017, pp. 1–6.
[24] J. Zhang, Z. Wang, C. Peng, L. Zhang, T. Huang, and Y. Liu, "RABA: Resource-aware backup allocation for a Chain of virtual network functions," in *Proc. IEEE Conf. Comput. Commun.*, 2019, pp. 1918–1926.
[25] J. Zhang, D. Zeng, L. Gu, H. Yao, and M. Xiong, "Joint optimization of virtual function migration and rule update in software defined NFV networks," in *Proc. IEEE Global Commun. Conf.*, 2017, pp. 1–5.

Jing Li received the BSc degree with the first class honours in computer science from the Australian National University, in 2018. He is currently working toward the PhD degree in the Research School of Computer Science, Australian National University. His research interests include mobile edge computing, network function virtualization, and combinatorial optimization.

Weifa Liang (Senior Member, IEEE) received the BSc degree in computer science from Wuhan University, China, in 1984, the ME degree in computer science from the University of Science and Technology of China, in 1989, and the PhD degree in computer science from the Australian National University, in 1998. He is currently a professor with the Research School of Computer Science, Australian National University. His research interests include design and analysis of energy efficient routing protocols for wireless ad hoc and sensor networks, Mobile Edge Computing (MEC), Network Function Virtualization (NFV), Software-Defined Networking (SDN), design and analysis of parallel and distributed algorithms, approximation algorithms, combinatorial optimization, and graph theory.

Meitian Huang received the BSc degree with the first class honours in computer science from the Australian National University, in 2015. He is currently working toward the PhD degree in the Research School of Computer Science, Australian National University. His research interests include software-defined networking, algorithm design and analysis, and cloud computing.



Xiaohua Jia (Fellow, IEEE) received the BSc and MEng degrees from the University of Science and Technology of China, in 1984 and 1987, respectively, and the DSc degree in information science from the University of Tokyo, in 1991. He is currently a chair professor with the Department of Computer Science, City University of Hong Kong. His research interests include cloud computing and distributed systems, computer networks, wireless sensor networks, and mobile wireless networks. He is an editor of the *IEEE Transactions on Parallel and Distributed Systems* (2006–2009), *Journal of World Wide Web*, *Wireless Networks*, *Journal of Combinatorial Optimization*, and so on. He is the general chair of ACM MobiHoc 2008, TPC co-chair of IEEE MASS 2009, area-chair of IEEE INFOCOM 2010, TPC co-chair of IEEE GlobeCom 2010, Ad Hoc and Sensor Networking Symposium, and panel co-chair of IEEE INFOCOM 2011.