Joint Resource Dimensioning and Placement for Dependable Virtualized Services in Mobile Edge Clouds

Peiyue Zhao and György Dán
Division of Network and Systems Engineering, School of Electrical Engineering and Computer Science
KTH, Royal Institute of Technology, Stockholm, Sweden E-mail: {peiyue, gyuri}@kth.se

***Abstract*—Mobile edge computing (MEC) is an emerging architecture for accommodating latency sensitive virtualized services (VSs). Many of these VSs are expected to be safety critical, and will have some form of reliability requirements. In order to support provisioning reliability to such VSs in MEC in an efficient and confidentiality preserving manner, in this paper we consider the joint resource dimensioning and placement problem for VSs with diverse reliability requirements, with the objective of minimizing the energy consumption. We formulate the problem as an integer programming problem, and prove that it is NP-hard. We propose a two-step approximation algorithm with bounded approximation ratio based on Lagrangian relaxation. We benchmark our algorithm against two greedy algorithms in realistic scenarios. The results show that the proposed solution is computationally efficient, scalable and can provide up to 30% reduction in energy consumption compared to greedy algorithms.**

I. Introduction

Mobile edge computing (MEC) is an emerging technology for providing distributed computational and storage resources to end users in mobile networks. Due to its proximity to end users, MEC benefits operators and end users by reducing end-to-end service latency, and by increasing data and service capacity, and is a promising architecture for hosting mission critical services as well if end-to-end latency requirements can be met [1]–[6]. Notable applications include real-time machine learning services, data analytics services, computation offloading for Internet of Things (IoT) devices and for connected vehicles, and process control for critical infrastructures [7], [8].

For all these applications resource dimensioning is a prerequisite for MEC to be able to host the Virtualized Services (VSs) corresponding to end-user applications (VSs are often referred to as micro-services in the literature). Resource dimensioning involves determining the location and amount of MEC resources to be deployed. In practice, MEC resource dimensioning at a candidate site needs to consider the available bandwidth and power supply, together with the kinds of MEC node hardware suitable for the site. Resource dimensioning naturally affects the feasible set of locations for the placement of VSs, and thus dimensioning and placement together have a high impact on the end-to-end latency and performance provided to the VSs [1]–[3].

This work was partly funded by the Swedish Research Council through the project Resist (2020-03860), by the Swedish Civil Contingencies Agency through the Cerces project and by Sweden's Innovation Agency (Vinnova) through the TECoSA project.

While there has been significant attention paid to the latency and energy consumption aspects of MEC, resilience to the failure of computing and communication resources has received less attention even though for many VSs that would benefit from MEC it is an essential requirement [6], [9], [10]. Failures could be due to cyber attacks, faulty components, or mistakes in operations [11]. While in the case of traditional cloud environments the cloud infrastructure exposes a high level topological abstraction to applications (typically referred to as availability zones), in the case of MEC such an abstraction is unlikely to be feasible as the small scale of the backhaul network would not allow a confidentiality preserving abstraction [12]. Thus, we argue that resilience in MEC would have to be expressed as part of the service level agreement (SLA) between individual applications and the network operator.

Including resilience as a requirement in SLAs implies that the network infrastructure has to be able to cater for the diverse resilience requirements of different services. For critical services that require continuous operation 1+1 redundancy shall be considered, such that seamless failover is possible between the primary instance and the redundant instance [11]. Less costly than 1+1 redundancy but suitable for stateful services with lower availability requirements is to stream the state of the primary VS instance to a secondary VS on a different MEC node. In this case the secondary VS only requires shared resources and a small amount of computational resources. Clearly, the resource requirements of VSs depend on their resilience requirements, which implies that existing approaches to resource dimensioning and placement have to be revisited to enable efficient MEC infrastructures that respect the latency and resilience requirements set in the SLAs, and at the same time are energy efficient.

In this paper we address the problem of joint resource dimensioning and placement (RDP) of VSs with diverse availability requirements in MEC. We consider a set of hardware configuration for MEC nodes and a set of candidate locations for deploying MEC nodes. The RDP problem is to dimension the MEC nodes to be deployed at each location for placing a set of VSs, with the objective of minimizing the total energy consumption of the system. To ensure service

continuity despite potential failures, we consider three classes of VSs with different availability requirements, and we consider that VS placement is subject to computational resource and VS-dependent end-to-end latency constraints. We formulate the RDP problem as an integer programming (IP) problem, and prove that it is NP-hard. We propose an efficient approximation algorithm based on Lagrangian relaxation and provide a bound on its approximation ratio. By simulations we benchmark our algorithm against three greedy algorithms, and we show through numerical results that it is efficient and scalable.

The rest of the paper is organized as follows. Section II reviews the related work and Section III introduces the system model and the problem formulation. Section IV shows the complexity of the RDP problem and Section V presents our approximation scheme for the RDP problem. Section VI presents numerical results and Section VII concludes the paper.

II. Related Work

Closely related to our work is resource dimensioning in the area of cloud computing [13]–[17]. The recent work [13] considers the placement of Virtual Machines (VMs) in MEC for minimizing average response times of services. [14] focuses on minimizing the total cost for MEC node and Access Point (AP) placement and for traffic routing. Both [13] and [14] propose heuristic solutions. [15] considers resource dimensioning for improving resource utilization, and proposes a two-step algorithm that decides the location of data centers and allocates computational resources to the data centers separately. However, resilience is not considered in these works.

Resource dimensioning with resilience is addressed in [16] with the objective of minimizing infrastructure cost. A multi-step solution is proposed to protect the system against network link failures. However, in this work the number of data centers is assumed to be given a priori, which limits the flexibility of resource dimensioning. Compared to these works, our model considers opening MEC nodes with heterogeneous hardware configurations at different locations. In addition, compared to [13]–[16] our work also considers the placement of services among the opened computational resources.

[17] considers the joint dimensioning and placement problem for IoT applications. The proposed solution relies on solving two sub-problems iteratively without a performance bound, places at most one MEC node per location, and resilience is not considered.

Also related to ours are the works that focus on service placement within MEC [1]–[5]. [1] considers the joint service placement and routing problem in MEC subject to computational and communication resources and proposes a randomized rounding based approximation algorithm. The algorithm allows to exceed the capacity of MEC nodes, which limits its application in realistic scenarios. [2] considers the joint service placement and request scheduling in MEC with the objective of maximizing the number of served users, subject to available computational and network resources. The problem is proven to be NP-hard and approximation algorithms are proposed. [3] considers the placement of services in heterogeneous MEC to maximize the total reward for providing services, and proposes an approximation algorithm based on partitioning each edge node into multiple slots. [4] considers the placement of multi-component applications in MEC and proposes and approximation algorithm. However, these works mainly consider minimizing the maximal cost of using an individual MEC node instead of the total cost. [5] proposes a multi-stage stochastic programming model for service placement in MEC subject to energy budget, and maximizes the quality of service.

Closest to our work in terms of methodology are the works on the uncapacitated facility location (UFL) problems, which we use as a building block of our approximation solution.The UFL problem is NP-hard, and approximation algorithms with a tight approximation bound have been proposed for the UFL problem [18], [19]. However, those works assume costs on a metric space, i.e., the cost for serving a client by a facility is proportional to geometric distances. Non-metric costs make many approximation techniques inapplicable to the UFL problem. An approximation algorithm based on converting the UFL problem to a minimal weight set cover problem was proposed in [19]. Compared to the UFL problems above, we consider the dimensioning of resources, services with different service availability requirements, and constraints on network delay and compatibility, which are in general not considered in the UFL problems.

III. System Model and Problem Formulation

A. Edge Computing Infrastructure

We consider a mobile backhaul that contains a set $\mathcal{B}$ of base stations (BSs) and a set $\mathcal{L}$ of locations where MEC nodes can be deployed. The MEC nodes can be located within the radio access network or at different network edge elements, for example, at base stations and at core network functions [20], [21]. Practically, the locations $\mathcal{L}$ are constrained by the availability of power supply, backhaul connectivity, and the preferences of the operators. There is an underlying backhaul network topology connecting the BSs $\mathcal{B}$ and the locations $\mathcal{L}$, which we model as a weighted graph $\mathcal{G} = (\mathcal{B} \cup \mathcal{L}, \mathcal{E}, \mathcal{W})$. The set of edges $\mathcal{E}$ corresponds to the set of paths in the backhaul network, and the weight $w_{i,j} \in \mathcal{W}$ of edge $(i,j) \in \mathcal{E}$ models the latency of the corresponding path.

We consider to open MEC nodes at the set $\mathcal{L}$ of locations, and we denote by $\mathcal{T}$ the set of MEC node types (e.g., hardware configuration) available for use. We denote by $c_{l,t} \in \mathbb{Z}_{\geq 0}$ the number of MEC nodes of type $t \in \mathcal{T}$ to be opened at location $l \in \mathcal{L}$, and we denote by the tuple (l,t,n) the n^{th} instance of a MEC node of type t opened at location l. Finally, we denote by $\mathcal{M} = \{(l,t,n) | l \in \mathcal{L}, t \in \mathcal{T}, n \in \mathbb{Z}_{\geq 0}\}$ the set of MEC nodes that could be opened in the mobile backhaul (i.e., considering all combinations of locations, types and number of nodes). Our notation allows $\mathcal{M}$ to be infinite, we will provide an upper bound on n later to ensure that $\mathcal{M}$ is a finite set. Assuming that $\mathcal{M}$ is finite we can reindex the MEC nodes in $\mathcal{M}$ so as to simplify notation: in what follows we use a single

index i for the MEC nodes in $\mathcal{M}$, and we define the functions $T(i) \in \mathcal{T}$, $L(i) \in \mathcal{L}$, and $N(i) \in \mathbb{Z}_{\geq 0}$ that provide the type, the location and the index of MEC node i, respectively. For convenience, let $\mathcal{M}_{l,t} = \{i \in \mathcal{M} | T(i) = t, L(i) = l\}$ be the set of MEC nodes of type t that can be placed at location l. We will use binary variable c_i to indicate if MEC node i is open, and we define $\mathbf{c} = \{c_1, c_2, \ldots, c_{|\mathcal{M}|}\}$.

We denote by integer parameter ω_i the number of computational resources available at MEC node $i \in \mathcal{M}$. The assumption of integer resource availability is reasonable if we consider the number of virtual CPUs (e.g., cores) as the computational resource, as bare metal partitioning hypervisors, such as Jailhouse [22] allocate dedicated virtual CPUs to each process for isolation and security reasons. Clearly, the MEC nodes of a particular type at a particular location have the same capacity, i.e., $\omega_i = \omega_j$, $\forall i, j \in \mathcal{M}_{l,t}$, $\forall l \in \mathcal{L}$, $\forall t \in \mathcal{T}$.

B. Virtualized Services and Availability

We consider the placement of VSs at the set $\mathcal{L}$ of MEC node locations; a VS corresponds to an edge computing application, such as an industrial control process, an augmented reality service, or a visual analytics application. The application logic for a VS is implemented in a virtualized environment (e.g., a container), and receives data from (and possibly sends data to) a physical process. Each VS f is characterized by its end-to-end latency requirement D_f, and by its service availability requirement. We consider three types of VSs in terms of their requirement for service availability.

Non-critical stateless: The first type of services have low availability requirements and/or do not require to maintain state. In case of failure of the node the VS is running on, a new instance of the VS can be created from a stored image without major impact on the application performance. Typical examples include visual analytics services (e.g., object recognition), and proportional-derivative (PD) controllers used for process control. These services require a single running instance, and do not need a hot standby or a replica. We denote by $\mathcal{F}_l$ the set of non-critical services.

Stateful: The second type of services require state to be maintained despite node failure. Typical examples of stateful VSs are process control services, such as the widely used proportional-integral-derivative (PID) controller, the linear-quadratic-Gaussian (LQG) controller, and model predictive control (MPC). These VSs thus require a hot standby besides the primary VS: when the primary instance of a VS is available, only the primary instance provides service, while the state of the primary instance is streamed to the secondary instance(s). In case of failure of the primary instance the secondary instance can take over based on the most recent state. We denote by $\mathcal{F}_n$ the set of processes that are stateful.

Critical: The third type of services may be stateful or stateless, but they require continuous real-time operation despite failure. Thus, they require redundant VSs that provide service simultaneously. Examples include controllers for unstable plants and controllers for safety critical systems, e.g., autonomous vehicles. We denote by $\mathcal{F}_h$ the set of critical processes.

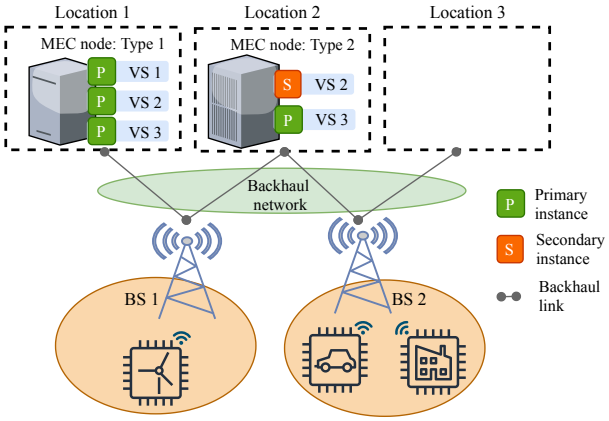

Figure 1: An example of resource dimensioning and VS placement for a system with two BSs ($|\mathcal{B}|= 2$), three locations ($|\mathcal{L}|= 3$), and two MEC node types ($|\mathcal{T}|= 2$). Three VSs are placed to serve three physical processes, with $\mathcal{F}_l = \{f_1\}$, $\mathcal{F}_n = \{f_2\}$ and $\mathcal{F}_h = \{f_3\}$.

We consider $n-1$ contingency as the planning criterion for reliability, that is, in case of the failure of a single MEC node the service continuity of stateful and critical VSs should be guaranteed (Section III, [23]). To satisfy the $n-1$ contingency criterion, we thus consider that each VS $f \in \mathcal{F}_n$ requires a primary instance to provide service, and a secondary instance as a hot standby. For critical services $f \in \mathcal{F}_h$ instead we have to ensure $1+1$ redundancy, i.e., two primary instances provide the service simultaneously.

An example of resource dimensioning and VS placement is shown in Figure 1. The mobile backhaul consists of two BSs ($|\mathcal{B}|= 2$) and three locations ($|\mathcal{L}|= 3$), where two types of MEC nodes ($|\mathcal{T}|= 2$) can be opened. In this mobile backhaul three VSs of different types (i.e., $\mathcal{F}_l = \{f_1\}$, $\mathcal{F}_n = \{f_2\}$ and $\mathcal{F}_h = \{f_3\}$) need to be placed to serve three physical processes. As an illustrative solution, a MEC node of type 1 and a MEC node of type 2 are opened at locations 1 and 2, respectively. VS instances are then placed as the figure shows to ensure service continuity for stateful and for critical VSs.

C. Placement Model

In order to provide a unified treatment of the different classes of VSs, we start with introducing replica instances for VSs $f \in \mathcal{F}_h$, which require two primary instances. We do so by introducing a replica f_c for each $f \in F_h$, and we define the set $\mathcal{F}_c$ of replica VSs. Note that $|\mathcal{F}_c|= |\mathcal{F}_h|$. Furthermore, for convenience we define the set $\mathcal{F} = \mathcal{F}_l \cup \mathcal{F}_n \cup \mathcal{F}_h \cup \mathcal{F}_c$ of all VSs. We use binary decision variables $x^p_{f,i}$ and $x^s_{f,j}$ to indicate the placement of the VS instances. Specifically, $x^p_{f,i} = 1$ if a primary instance of VS f is placed on MEC node $i \in \mathcal{M}$, and $x^p_{f,i} = 0$ otherwise. Similarly, $x^s_{f,j} = 1$ if a secondary instance of VS f is placed on MEC node $j \in \mathcal{M}$, and $x^s_{f,j} = 0$ otherwise. Furthermore, we define $\mathbf{x} = \{x^p_{1,1}, x^s_{1,1}, \ldots, x^p_{|\mathcal{F}|,|\mathcal{M}|}, x^s_{|\mathcal{F}|,|\mathcal{M}|}\}$. Since each VS $f \in \mathcal{F}$ requires one primary instance to be placed, and each VS in $\mathcal{F}_n$ also requires one secondary instance to be placed, we have

$$\sum\nolimits_{i\in\mathcal{M}} x^p_{f,i} = 1, \forall f \in \mathcal{F}, \quad (1)$$

$$\text{and } \sum\nolimits_{j\in\mathcal{M}} x^s_{f,j} = 1, \forall f \in \mathcal{F}_n. \quad (2)$$

Besides the general constraints above we introduce two constraints that are specific to the type of VS. First, for a VS $f \in \mathcal{F}_n$, the primary and secondary instances of VS f cannot be placed on the same MEC node, and thus

$$x^p_{f,i} + x^s_{f,i} \leq 1, \forall f \in \mathcal{F}_n, i \in \mathcal{M}. \quad (3)$$

The constraint above ensures that upon the failure of the MEC node that hosts the primary instance of $f \in \mathcal{F}_n$, the secondary instance of f can provide service, while the primary instance of f is not affected if the MEC node that hosts the secondary instance of f fails.

Furthermore, for a critical VS $f \in \mathcal{F}_h$, the primary instance of VS f and its replica f_c cannot be placed on the same MEC node, and therefore we have

$$x^p_{f,i} + x^p_{f_c,i} \leq 1, \forall f \in \mathcal{F}_h \text{ and } \forall i \in \mathcal{M}. \quad (4)$$

In order to ensure that the placement of VSs respects the end-to-end latency requirements, we consider the communication latency due to wireless transmission and backhaul network transmission. We denote by $b_f \in \mathcal{B}$ the BS that serves the edge computing application (plant, augmented reality device, etc) corresponding to VS f, and we denote by d_{f,b_f} the wireless transmission latency to BS b_f. Furthermore, we denote by $d_{b_f,i}$ the backhaul network transmission latency between BSs b_f and MEC node i. $d_{b_f,i}$ is calculated as the sum weight of the edges on the shortest path between b_f and i on the backhaul graph $\mathcal{G}$. Based on these the total communication latency between VS f and MEC node i is

$$d_{f,i} = d_{f,b_f} + d_{b_f,i}. \quad (5)$$

This model is simple, but can be used with any additive measure of latency (e.g., mean). We consider that the amount of traffic related to VSs is relatively small and therefore it does not affect the latency on a link. Thus the latency mainly depends on the locations of the VSs and the backhaul topology. Furthermore, $d_{f,i}$ can include the latency in the uplink and in the downlink, depending on the use case of VS f, determined by interference, radio network topology and the interference management solutions used.

Let us denote by D_f the end-to-end delay requirement of VS f. Then any instance of VS $f \in \mathcal{F}$ must be placed on a MEC that satisfies

$$x^p_{f,i} d_{f,i} \leq D_f, \forall f \in \mathcal{F}, i \in \mathcal{M} \quad (6)$$

$$x^s_{f,j} d_{f,j} \leq D_f, \forall f \in \mathcal{F}_n, j \in \mathcal{M}. \quad (7)$$

We consider that the MEC nodes of a particular type $t \in \mathcal{T}$ at a particular location $l \in \mathcal{L}$ also have the same total communication latency for serving a VS, i.e., $d_{f,i} = d_{f,j}, \forall i,j \in \mathcal{M}_{l,t}$ and $\forall f \in \mathcal{F}$.

Placement is also constrained by the availability of computational resources. We consider that each primary VS instance requires one unit of computational resource to provide service, while a secondary VS instance requires a unit of computational resource only if it is providing service (i.e., if the primary has failed). Our assumption of the resource requirements of the primary instances are motivated by the use case of critical applications, which require isolation for performance and security reasons and thus would prefer to be executed on a dedicated resource. The assumption about the secondary instances is reasonable, as when a secondary instance is not providing service, it only needs computational resources for the state to be synchronized with the primary instance, which requires much less resources than providing the service. We provide experimental results in Section VI-D that support our modeling assumption. We can thus consider that on each MEC node all secondary instances that are not providing service can be handled by a single computational resource on the MEC node, e.g., by the resource (vCPU) that executes the operating system (or the hypervisor).

For a VS f of any type its primary and secondary instances can be placed on MEC node i only if i is open, and therefore

$$x^p_{f,i} \leq c_i, \forall f \in \mathcal{F},\ i \in \mathcal{M}, \quad (8)$$

$$\text{and } x^s_{f,j} \leq c_j,\ \forall f \in \mathcal{F}_n,\ j \in \mathcal{M}. \quad (9)$$

When MEC node i is open, the number of primary VS instances placed on it can be calculated as $\sum_{f\in\mathcal{F}} x^p_{f,i}$, and therefore the available resources for placing the secondary instances on MEC node i are:

$$a_i = c_i\omega_i - \sum\nolimits_{f\in\mathcal{F}} x^p_{f,i}. \quad (10)$$

When a MEC node $j \in \mathcal{M}$ fails, the secondary instances of VSs in $\mathcal{F}_n$ with $x^p_{f,j} = 1$ will be providing service, among which $\sum_{f\in\mathcal{F}_n} x^s_{f,i} x^p_{f,j}$ secondary VS instances will be running on MEC node $i \neq j$. Therefore, MEC node i needs to have enough resources for those secondary instances,

$$a_i \geq \sum\nolimits_{f\in\mathcal{F}_n} x^s_{f,i} x^p_{f,j},\ \forall i,j \in \mathcal{M}. \quad (11)$$

Note that constraint (11) is non-linear. In order to solve the problem as an IP problem, we linearize (11) by introducing a new binary variable $y_{f,i,j}$ with the following constraints:

$$y_{f,i,j} \leq x^p_{f,i}, \forall f \in \mathcal{F}_n, \text{ and } i,j \in \mathcal{M}, \quad (12)$$

$$y_{f,i,j} \leq x^s_{f,j}, \forall f \in \mathcal{F}_n, \text{ and } i,j \in \mathcal{M}, \quad (13)$$

$$y_{f,i,j} \geq x^p_{f,i} + x^s_{f,j} - 1, \forall f \in \mathcal{F}_n, \text{ and } i,j \in \mathcal{M}. \quad (14)$$

When at least one of the the variables $x^p_{f,i}$ and $x^s_{f,j}$ is equal to 0, constraints (12) and (13) enforce that $y_{f,i,j} \leq 0$. Since $y_{f,i,j}$ is binary, $y_{f,i,j} = 0$. When $x^p_{f,i} = 1$, $x^s_{f,j} = 1$, the constraints above require that $y_{f,i,j} \leq 1$ and $y_{f,i,j} \geq 1$, and thus $y_{f,i,j} = 1$. Therefore, with the constraints (12)-(14), we can replace $x^p_{f,i} x^s_{f,j}$ by $y_{f,i,j}$, and thus (11) is equivalent to

$$a_i \geq \sum\nolimits_{f\in\mathcal{F}_n} y_{f,j,i}, \forall f \in \mathcal{F}_n, \text{ and } i,j \in \mathcal{M}. \quad (15)$$

Finally, we define $\mathbf{y} = \{y_{1,1,1}, \ldots, y_{|\mathcal{F}|,|\mathcal{M}|,|\mathcal{M}|}\}$.

D. Energy Consumption and Problem Formulation

When MEC node i is open for placing VSs, we consider that its energy consumption has two sources. The first source is the base energy consumption p_i that accounts for running the operating system and for hosting the secondary VS instances. The second source is the energy consumption for executing

Table I: Table of Notations

Sets	
$\mathcal{B}$	Set of base stations
$\mathcal{F}_l$	Set of non-critical stateless VSs
$\mathcal{F}_n$	Set of stateful VSs
$\mathcal{F}_h$	Set of critical VSs
$\mathcal{F}_c$	Set of replica VSs
$\mathcal{F}$	Set of all VSs
$\mathcal{L}$	Set of possible locations of MEC nodes
$\mathcal{T}$	Set of MEC node types
$\mathcal{M}$	Set of all MEC nodes
$\mathcal{M}_{l,t}$	Set of MEC nodes of type $t \in \mathcal{T}$ at $l \in \mathcal{L}$
Parameters	
$b_{f,i}$	Latency between $f \in \mathcal{F}$ and $i \in \mathcal{M}$
$c_{l,t}$	Number of possible MEC nodes of type $t \in \mathcal{T}$ at $l \in \mathcal{L}$
D_f	Delay requirement of $f \in \mathcal{F}$
p_i	Base energy consumption of $i \in \mathcal{M}$
$p_{f,i}$	Energy consumption for executing $f \in \mathcal{F}$ on $i \in \mathcal{M}$
ω_i	Capacity of $i \in \mathcal{M}$
Functions	
$T(i)$	MEC node type of $i \in \mathcal{M}$
$L(i)$	Location of $i \in \mathcal{M}$
$N(i)$	Index of $i \in \mathcal{M}$
Decision Variables	
c_i	$= \begin{cases} 1, \text{ MEC node } i \in \mathcal{M} \text{ is open} \\ 0, \text{ Otherwise} \end{cases}$
$x^p_{f,i}$	$= \begin{cases} 1, \text{ Primary instance of } f \in \mathcal{F} \text{ is placed on } i \in \mathcal{M} \\ 0, \text{ Otherwise} \end{cases}$
$x^s_{f,j}$	$= \begin{cases} 1, \text{ Secondary instance of } f \in \mathcal{F} \text{ is placed on } j \in \mathcal{M} \\ 0, \text{ Otherwise} \end{cases}$
$y_{f,i,j}$	$= \begin{cases} 1, \ x^p_{f,i} = 1 \text{ and } x^s_{f,j} = 1, \ \forall f \in \mathcal{F} \text{ and } i, j \in \mathcal{M} \\ 0, \text{ Otherwise} \end{cases}$

the primary VS instances; we denote by $p_{f,i}$ the energy consumption of hosting the VS $f \in \mathcal{F}$ on i. We consider that $p_{f,i}$ accounts for the increased energy consumption of the CPUs, memory, hard disks, cooling systems and other hardware components, due to hosting VS f. Furthermore, we make the reasonable assumption that MEC nodes of the same type $t \in \mathcal{T}$ and at the same location $l \in \mathcal{L}$ have the same energy consumption, i.e., $p_i = p_j, \forall i, j \in \mathcal{M}_{l,t}$, and $p_{f,i} = p_{f,j}, \forall i, j \in \mathcal{M}_{l,t}$. Using these notations we can express the total energy cost as

$$E(\mathbf{c}, \mathbf{x}) = \sum\nolimits_{i \in \mathcal{M}} \left(c_i p_i + \sum\nolimits_{f \in \mathcal{F}} p_{f,i} x^p_{f,i} \right). \tag{16}$$

We are now ready to formulate the joint resource dimensioning and placement problem for VSs within MEC, referred to as the RDP problem. The RDP problem is to decide the number of MEC nodes to be placed at each location of each MEC node type, and to compute the placement of the primary and secondary instances of VSs, with the objective of minimizing the total energy consumption of the MEC nodes.

$$\begin{aligned} \underset{\mathbf{c},\mathbf{x},\mathbf{y}}{\text{minimize}} \quad & E(\mathbf{c}, \mathbf{x}) \\ \text{subject to} \quad & (1) - (10) \text{ and } (12) - (15) \\ & c_i \in \{0, 1\}, \ \forall i \in \mathcal{M} \\ & x^p_{f,i} \in \{0, 1\}, \ \forall f \in \mathcal{F}, \text{ and } i \in \mathcal{M} \\ & x^s_{f,j}, y_{f,i,j} \in \{0, 1\}, \ \forall f \in \mathcal{F}_n, \text{ and } i, j \in \mathcal{M} \end{aligned} \tag{RDP}$$

In the RDP problem we observe that the number of binary variables $x^p_{f,i}$ and $x^s_{f,j}$ depends on the number of MEC nodes $|\mathcal{M}|$, which further depends on $c_{l,t}$, the number of MEC nodes of type t opened at location l. Therefore, in order to solve the RDP problem as an IP programming problem, we need to fix the number of variables. In the following result, we show that $c_{l,t}$ can in fact be bounded.

Lemma 1. *Consider a set $\mathcal{F}$ of VSs with delay constraints D_f. Denote by $\omega_{l,t} = \omega_i \ \forall i \in \mathcal{M}_{l,t}$ the capacity of MEC nodes of type $t \in \mathcal{T}$ at location $l \in \mathcal{L}$, and by $d_{f,l,t} = d_{f,i} \ \forall i \in \mathcal{M}_{l,t}, \forall f \in \mathcal{F}$ their communication latency. Then at location l we need to open at most $c^{max}_{l,t} = \lceil \frac{|\{f \in \mathcal{F} | d_{f,l,t} \leq D_f\}|}{\omega_{l,t}} \rceil + 1$ MEC nodes of type t.*

Proof. The set of VSs for which the delay constraints can be satisfied by MEC nodes in $\mathcal{M}_{l,t}$ is $\{f \in \mathcal{F} | d_{f,l,t} \leq D_f\}$. Therefore, to place the primary VS instances at most $c^{max}_{l,t} = \lceil \frac{|\{f \in \mathcal{F} | d_{f,l,t} \leq D_f\}|}{\omega_{l,t}} \rceil$ MEC nodes of type t need to be opened at location l. In addition, we need at most one more MEC node of type t at location l to satisfy the $n-1$ contingency criterion (i.e., place the secondary instances of the stateful VSs with primary instances placed on MEC nodes $\mathcal{M}_{l,t}$). This proves the result. □

By Lemma 1 we can thus set $c_{l,t} = c^{max}_{l,t}$ in the RDP problem. Then the problem becomes an IP problem. The most frequently used notations are summarized in Table I.

IV. Complexity Analysis

In what follows we analyze the computational complexity of the RDP problem. For this, we start with recalling the minimum set cover (MSC) problem.

MSC problem: Consider a set $\mathcal{E} = \{e_1, e_2, \ldots, e_{|\mathcal{E}|}\}$ of elements, and let $\mathcal{S} = \{S_1, S_2, \ldots, S_{|\mathcal{S}|}\}$ be a collection of subsets of $\mathcal{E}$ such that $\cup_{s=1}^{|\mathcal{S}|} S_s = \mathcal{E}$. For a sub-collection $\mathcal{R} \subseteq \mathcal{S}$ we say that $\mathcal{R}$ is a set cover of $\mathcal{E}$ if each element in $\mathcal{E}$ belongs to at least one subset in $\mathcal{R}$. The MSC problem is to find a set cover of $\mathcal{E}$ consisting of a minimal number of subsets. The MSC problem is known to be NP-hard (Chapter 16.1, [24]). We now introduce a variant of MSC as follows.

Minimum set cover with soft capacity constraints (MSCSC) problem: Consider a set $\mathcal{E} = \{e_1, e_2, \ldots, e_{|\mathcal{E}|}\}$ of elements. Let $\mathcal{S} = \{S_1, S_2, \ldots, S_{|\mathcal{S}|}\}$ be a collection of subsets of $\mathcal{E}$, such that $\cup_{s=1}^{|\mathcal{S}|} S_s = \mathcal{E}$. Furthermore, assume that each subset $S_i \in \mathcal{S}$ can be used to cover at most $g_i \in \mathbb{Z}_{\geq 0}$ elements of $\mathcal{E}$. For a sub-collection $\mathcal{R} \subseteq \mathcal{S}$ we say that $\mathcal{R}$ is a set cover of $\mathcal{E}$ if there exists a mapping $Q : \mathcal{E} \to \mathcal{R}$ and non-negative integers $h_i, 1 \leq i \leq |\mathcal{S}|$, such that Q is defined for all $e \in \mathcal{E}$, and there are at most $g_i h_i$ elements mapped to

subset $S_i \in \mathcal{R}$. The MSCSC problem is to find a set cover $\mathcal{R}$ with minimal $\sum_{S_i \in \mathcal{R}} h_i$. Our next result shows that the MSCSC problem is also NP-hard.

Lemma 2. *The MSCSC problem is NP-hard.*

Proof. We prove this result by reducing the MSC problem to the MSCSC problem. For an instance ($\mathcal{E}$, $\mathcal{S}$) of the MSC problem we create an instance of the MSCSC problem with ($\mathcal{E}$, $\mathcal{S}$,(g_i)), with $g_i = |S_i|$ $\forall S_i \in \mathcal{S}$. Since $g_i = |S_i|$, each subset in $\mathcal{S}$ needs to be included in $\mathcal{R}$ at most once, i.e., $h_i \in \{0, 1\}$ at optimality. Therefore, a solution to the MSCSC problem instance is a solution to the corresponding MSC problem instance, which proves the lemma. □

Now we can prove the complexity of the RDP problem.

Proposition 1. *The RDP problem is NP-hard.*

Proof. We prove the proposition by reducing the MSCSC problem to the RDP problem. For an instance ($\mathcal{E}$, $\mathcal{S}$,(g_i)) of the MSCSC problem we construct a corresponding instance of the RDP problem as follows. We create a set $\mathcal{F}_n = \{1, 2, \ldots, |\mathcal{E}|\}$ of stateful VSs, where $f \in \mathcal{F}_n$ corresponds to element $e_f \in \mathcal{E}$, and let sets $\mathcal{F}_l = \mathcal{F}_h = \mathcal{F}_c = \emptyset$. Clearly, $\mathcal{F} = \mathcal{F}_n$. We then create a mobile backhaul with a single BS and a single MEC location, i.e., $\mathcal{L} = \mathcal{B} = \{1\}$, and create a set $T = \{1, 2, \ldots, |\mathcal{S}|, |\mathcal{S}|{+}1\}$ of MEC node types. Each MEC node type $1 \leq t \leq |\mathcal{S}|$ corresponds to subset $S_t \in \mathcal{S}$. Furthermore, we set $d_{f,i} \leq D_f$ $\forall e_f \in S_{T(i)}$ if $1 \leq T(i) \leq |\mathcal{S}|$, and we set $d_{f,i} \leq D_f$ $\forall f$ if $T(i) = |\mathcal{S}|{+}1$. The capacity of each MEC node is set as $\omega_i = g_{T(i)}$ $\forall 1 \leq T(i) \leq |\mathcal{S}|$, and $\omega_i = |\mathcal{F}|$ if $T(i) = |\mathcal{S}|{+}1$. We set the base energy consumption $p_i = 1$ $\forall 1 \leq T(i) \leq |\mathcal{S}|$ and $p_i = 0$ if $T(i) = |\mathcal{S}|{+}1$. The energy consumption for executing a VS instance is set as $p_{f,i} = 0$ $\forall f$ if $\forall 1 \leq T(i) \leq |\mathcal{S}|$, and $p_{f,i} \gg 0$, $\forall f$ if $T(i) = |\mathcal{S}|{+}1$.

Let us denote by $(\mathbf{c}^\star, \mathbf{x}^\star)$ an optimal solution to the RDP problem above, and let $\mathcal{M}^\star = \{i \in \mathcal{M} : c_i^\star = 1, 1 \leq T(i) \leq |\mathcal{S}|\}$. Since MEC nodes of type $|\mathcal{S}|{+}1$ have much higher energy cost than the MEC nodes of other types for executing VS instances, in an optimal solution they are only suitable for the placement of the secondary instances, i.e., $x_{f,i}^{p\star} = 0, \forall\ T(i) = |\mathcal{S}|{+}1$ and $f \in \mathcal{F}$. Therefore, the objective function of the RDP problem above becomes

$$E(\mathbf{c}^\star, \mathbf{x}^\star) = \sum_{1 \leq T(i) \leq |\mathcal{S}|} (c_i^\star p_i + \sum_{f \in \mathcal{F}} p_{f,i} x_{f,i}^{p\star}) + \\ + \sum_{T(i) = |\mathcal{S}|+1} (c_i^\star p_i + \sum_{f \in \mathcal{F}} p_{f,i} x_{f,i}^{p\star}) = \sum_{1 \leq T(i) \leq |\mathcal{S}|} c_i^\star. \quad (17)$$

In what follows we construct a solution $(\mathcal{R}^\star, Q^\star, h_i^\star)$ to the MSCSC problem from $(\mathbf{c}^\star, \mathbf{x}^\star)$. For each VS f, by constraint (1) there exists i such that $x_{f,i}^{p\star} = 1$. Then we include subset $S_{T(i)}$ in $\mathcal{R}^\star$, add mapping $e_f \to S_{T(i)}$ to $Q^\star$. For each subset S_i, we set $h_i^\star = \sum_{j \in \mathcal{M}, T(j) = i} c_j^\star$. Clearly, $E(\mathbf{c}^\star, \mathbf{x}^\star) = \sum_{S_i \in \mathcal{R}^\star} h_i^\star$.

To prove the proposition, what remains is to prove that $(\mathcal{R}^\star, Q^\star, h_i^\star)$ is an optimal solution to the MSCSC problem. We prove this by contradiction. Let us assume that there exists another solution $(\mathcal{R}', Q', h_i')$ to the MSCSC problem such that $\sum_{S_i \in \mathcal{R}'} h_i' < \sum_{S_i \in \mathcal{R}^\star} h_i$. Then we can construct a solution $(\mathbf{c}', \mathbf{x}')$ to the RDP problem as follows. We open h_i' MEC nodes of type i, and then place the primary instance of VS f on one of the MEC nodes of type i if there exists a mapping $e_f \to S_i \in Q'$. Finally, we place the secondary instance of the VSs on the MEC node of type $|S|{+}1$, which does not increase the total energy consumption. Since $E(\mathbf{c}', \mathbf{x}') = \sum_{1 \leq T(i) \leq |\mathcal{S}|} c_i' = \sum_{S_i \in \mathcal{R}'} h_i'$, we have $E(\mathbf{c}', \mathbf{x}') < E(\mathbf{c}^\star, \mathbf{x}^\star)$, which contradicts the assumption that $(\mathbf{c}^\star, \mathbf{x}^\star)$ was optimal. This concludes the proof. □

V. Approximation Algorithm based on Lagrangian Relaxation

Our proof of NP-hardness of the RDP problem relies on proving that the classical set cover problem is a special case of the RDP problem. For set cover type problems branch-and-bound algorithms have been shown not to perform well (Table 5, [25]), thus in what follows we propose computationally efficient algorithms for computing approximate solutions to the RDP problem. We start with introducing an approximation algorithm for the UFL problem with delay and compatibility constraints, which we use as a building block in our approximation scheme, after that we develop the approximation algorithm, and prove its approximation ratio bound.

A. UFL problem with delay and compatibility constraints

We present an approximation algorithm for the UFL problem with delay and compatibility constraints, referred to as the UFL-DC problem. We are not aware of an approximation algorithm for the UFL-DC, but in the following we show that the algorithm for approximating the non-metric UFL problem proposed in [26] can be extended to the UFL-DC problem.

The UFL-DC problem is as follows. Consider a set of facilities $\mathcal{U}$ and a set of clients $\mathcal{K}$. Decision variable b_u denotes if facility u is open, and decision variable $z_{k,u}$ denotes if client k is served by facility u. The cost for opening facility u is q_u and the cost for serving a client k by facility u is $q_{k,u}$. In a feasible solution each client $k \in \mathcal{K}$ must be served by one facility,

$$\sum\nolimits_u z_{k,u} = 1, \forall k \in \mathcal{K}. \quad (18)$$

Parameter D_k denotes the delay requirement of client k, and parameter $d_{k,u}$ denotes the service delay when client k is served by facility u. A client k can be served by a facility u only if its delay requirement is satisfied, thus

$$z_{k,u} d_{k,u} \leq D_k b_u,\ \forall u \in \mathcal{U}, \forall k \in \mathcal{K}. \quad (19)$$

Due to the compatibility constraint two clients that are incompatible with each other cannot be served by the same facility. Parameter $r_{k,k'}$ indicates if two clients $k, k', k \neq k'$ are compatible, and thus

$$z_{k,u} + z_{k',u} \leq 1 + r_{k,k'}, \forall k, k' \in \mathcal{K}, u \in \mathcal{U}. \quad (20)$$

The objective of the UFL-DC problem is to find a feasible solution that minimizes the total cost,

Algorithm 1: Approximation algorithm for UFL-DC

Input : $\mathcal{U}, \mathcal{K}, q_k, q_{k,u}, r_{k,k'}$

1 $\Omega = \mathcal{K}$, and $n = 0$
2 **while** $\Omega \neq \emptyset$ **do**
3 $\quad n = n + 1$
4 $\quad [u^{(n)}, \mathcal{K}^{(n)}] =$ argmin$\{v(u, \mathcal{K}'_u) | u \in \mathcal{U}, \mathcal{K}'_u \subseteq \mathcal{K}_u \cap \Omega\}$
5 $\quad b_{u^{(n)}} = 1$
6 $\quad z_{u^{(n)},k} = 1, \forall k \in \mathcal{K}^{(n)}$
7 $\quad \Omega = \Omega \setminus \mathcal{K}_u^{(n)}$

Output : $b_u, z_{f,u}$

$$\begin{aligned} \underset{b_u, z_{k,u}}{\text{minimize}} \quad & \sum_u b_u q_u + \sum_u \sum_k z_{k,u} q_{k,u} \\ \text{subject to} \quad & (18) - (20) \\ & b_u \in \{0,1\}, \ \forall u \in \mathcal{U} \\ & z_{k,u}, \in \{0,1\}, \ \forall \ k \in \mathcal{K}, \ u \in \mathcal{U} \end{aligned} \tag{UFL-DC}$$

The UFL-DC problem is clearly NP-hard, as we can convert an instance of the UFL problem into an instance of the UFL-DC problem by setting $D_k = \infty$ and $r_{k,k'} = 1, \forall k, k' \in \mathcal{K}$.

To develop an approximation algorithm, for convenience, let us denote by $\mathcal{K}_u = \{k \in \mathcal{K} | d_{k,u} \leq D_k\}$ the set of clients that can be served by facility u. Let us consider a subset $\mathcal{K}'_u \subseteq \mathcal{K}_u$ of clients, and let us define the cost efficiency of serving clients $\mathcal{K}'_u$ by u as

$$v(u, \mathcal{K}'_u) = \frac{q_u + \sum_{k \in \mathcal{K}'_u} q_{k,u}}{|\mathcal{K}'_u|}. \tag{21}$$

In what follows we propose Algorithm 1 as an approximation scheme for the UFL-DC problem. The algorithm is an extension of an approximation algorithm for the UFL problem proposed in [26]. In line 1, we initialize the set of unserved clients $\Omega = \mathcal{K}$. Observe that $\mathcal{K}_u \cap \Omega$ is the set of unserved clients that can be served by facility u. Then in iteration n, among all the facilities, we find the facility $u^{(n)}$ and the subsets of clients $\mathcal{K}^{(n)}$ with the lowest cost efficiency $v(u^{(n)}, \mathcal{K}^{(n)})$ (line 4). Then we open the facility $u^{(n)}$ and let $u^{(n)}$ serve the clients $\mathcal{K}^{(n)}$ (lines 5-7). The algorithm continues until all the clients are served. The following result provides an approximation ratio bound for Algorithm 1.

Lemma 3. *For an instance of the UFL-DC problem let us denote by (b_u,$z_{k,u}$) the solution provided by Algorithm 1, and let us denote by Opt(UFL-DC) the optimal objective value of the UFL-DC problem. Then*

$$\sum_u b_u q_u + \sum_u \sum_k z_{k,u} q_{k,u} \leq H_{|\mathcal{K}|} \cdot \textit{Opt(UFL-DC)}, \tag{22}$$

where $H_{|\mathcal{K}|}$ is the harmonic series and $H_{|\mathcal{K}|} \approx log(|\mathcal{K}|)$.

The proof is provided in the appendix.

B. Approximation Algorithm

We now proceed with presenting our approximation scheme for the RDP problem. First, we apply Lagrangian relaxation to the RDP problem, and then we propose algorithms that place the primary and the secondary instances, referred to as the LRA (LRA) algorithm.

Lagrangian relaxation: We apply Lagrangian relaxation to constraint (15) of the RDP problem as follows. Let us denote by $\lambda_{i,j} > 0$ the Lagrangian multiplier of constraint (15), then the objective function of the relaxed problem becomes

$$\begin{aligned} & E(\mathbf{c}, \mathbf{x}) - \sum_{i \in \mathcal{M}} \sum_{j \in \mathcal{M}} \lambda_{i,j} (c_i \omega_i - \sum_{f \in \mathcal{F}} x^p_{f,i} - \sum_{f \in \mathcal{F}_n} y_{f,j,i}) \\ & = \sum_{i \in \mathcal{M}} c_i \left(p_i - \omega_i \sum_{j \in \mathcal{M}} \lambda_{i,j} \right) + \sum_{i \in \mathcal{M}} \sum_{f \in \mathcal{F}} x^p_{f,i} \left(p_{f,i} + \sum_{j \in \mathcal{M}} \lambda_{i,j} \right) \\ & + \sum_{i \in \mathcal{M}} \sum_{j \in \mathcal{M}} \sum_{f \in \mathcal{F}_n} \lambda_{i,j} y_{f,j,i} \end{aligned} \tag{23}$$

As a next step let us set $\lambda_{i,j} = \frac{\beta_i}{\omega_i}$, for some $\beta_i > 0$, whose value we will determine later to serve the derivation of the approximation bound of the proposed LRA algorithm. We can express the objective function of the relaxed problem as

$$\begin{aligned} E_R(\mathbf{c}, \mathbf{x}, \mathbf{y}) = & \underbrace{\sum_{i \in \mathcal{M}} c_i (p_i - |\mathcal{M}| \beta_i)}_{\text{base energy consumption}} \\ & + \underbrace{\sum_{f \in \mathcal{F}} \sum_{i \in \mathcal{M}} (p_{f,i} + |M| \frac{\beta_i}{\omega_i}) x^p_{f,i}}_{\text{placement of primary instances}} + \underbrace{\sum_{f \in \mathcal{F}_n} \sum_{i \in \mathcal{M}} \sum_{j \in \mathcal{M}} \frac{\beta_j}{\omega_j} y_{f,i,j}}_{\text{placement of secondary instances}} \end{aligned} \tag{24}$$

The relaxed-RDP (R-RDP) problem then becomes

$$\begin{aligned} \underset{\mathbf{c}, \mathbf{x}, \mathbf{y}}{\text{minimize}} \quad & E_R(\mathbf{c}, \mathbf{x}, \mathbf{y}) \\ \text{subject to} \quad & (1) - (9) \text{ and } (12) - (14) \\ & c_i \in \{0,1\}, \ \forall i \in \mathcal{M} \\ & x^p_{f,i} \in \{0,1\}, \ \forall f \in \mathcal{F}, \text{ and } i \in \mathcal{M} \\ & x^s_{f,j}, y_{f,i,j} \in \{0,1\}, \ \forall f \in \mathcal{F}_n, \text{ and } i, j \in \mathcal{M} \end{aligned} \tag{R-RDP}$$

Let us denote by Opt(RDP) and Opt(R-RDP) the optimal objective value of the RDP and the R-RDP problems, respectively. We relate Opt(RDP) and Opt(R-RDP) as follows.

Proposition 2. *Opt(R-RDP) $\leq$ Opt(RDP).*

Proof. Assume that $(\mathbf{c}^\star, \mathbf{x}^\star, \mathbf{y}^\star)$ is an optimal solution to an instance of the RDP problem. Since the constraints of the R-RDP problem constitute a subset of the constraints of the RDP problem, $(\mathbf{c}^\star, \mathbf{x}^\star, \mathbf{y}^\star)$ is also a feasible solution to the R-RDP problem. Thus,

$$\text{Opt(R-RDP)} \leq E_R(\mathbf{c}^\star, \mathbf{x}^\star, \mathbf{y}^\star). \tag{25}$$

Now, because $(\mathbf{c}^\star, \mathbf{x}^\star, \mathbf{y}^\star)$ is feasible for the RDP problem, the constraint (15) is satisfied, and thus

$$c^\star_i \omega_i - \sum_{f \in \mathcal{F}} x^{p\star}_{f,i} - \sum_{f \in \mathcal{F}_n} y^\star_{f,j,i} \geq 0, \forall i, j \in \mathcal{M}. \tag{26}$$

Furthermore, since $\lambda_{i,j} > 0$ and by (23) we have that

$$E_R(\mathbf{c}^\star, \mathbf{x}^\star, \mathbf{y}^\star) \leq E(\mathbf{c}^\star, \mathbf{x}^\star) = Opt(RDP). \tag{27}$$

This proves the statement. □

Based on the Lagrangian relaxation of constraint (15), in what follows we first focus on the placement of the primary instances, and then we handle the placement of the secondary instances.

Placement of the primary instances: For the placement of the primary instances we first construct the modified-RDP (M-RDP) problem by ignoring the terms $\frac{\beta_j}{\omega_j}|\mathcal{M}|y_{f,i,j}$ that are related to the placement of the secondary instances in the objective function of the R-RDP problem, and by only considering the constraints related to the primary placement variable $x^p_{f,i}$,

$$\begin{aligned} \underset{\mathbf{c},\mathbf{x}}{\text{minimize}} \quad & E_M(\mathbf{c},\mathbf{x}) = \sum_{i\in\mathcal{M}} c_i(p_i - |\mathcal{M}|\beta_i) \\ & + \sum_{f\in\mathcal{F}}\sum_{i\in\mathcal{M}} (p_{f,i} + |\mathcal{M}|\frac{\beta_i}{\omega_i})x^p_{f,i} \qquad \text{(M-RDP)} \\ \text{subject to} \quad & (1),(4),(6), \text{ and } (8) \\ & c_i \in \{0,1\},\ \forall i \in \mathcal{M} \\ & x^p_{f,i} \in \{0,1\},\ \forall f \in \mathcal{F}, \text{ and } i \in \mathcal{M} \end{aligned}$$

Let us denote by Opt(M-RDP) the optimal objective value of the M-RDP problem. The following result relates the optimal objective values of M-RDP and R-RDP, and will facilitate the derivation of the approximation ratio bound of the proposed algorithm.

Proposition 3. *Opt(M-RDP)≤ Opt(R-RDP)*

Proof. Let use denote by $(\mathbf{c}^\star, \mathbf{x}^\star, \mathbf{y}^\star)$ an optimal solution to the R-RDP problem. Observe that the constraints of the M-RDP problem are a subset of the constraints of the R-RDP problem, therefore $(\mathbf{c}^\star, \mathbf{x}^\star)$ is also a feasible solution to the M-RDP problem, and thus

$$\text{Opt(M-RDP)} \leq E_M(\mathbf{c}^\star, \mathbf{x}^\star).$$

Furthermore, observe that the objective function $E_M(\mathbf{c}^\star, \mathbf{x}^\star)$ contains the first two terms of the objective function $E_R(\mathbf{c}^\star, \mathbf{x}^\star, \mathbf{y}^\star)$, and therefore,

$$E_M(\mathbf{c}^\star, \mathbf{x}^\star) \leq E_R(\mathbf{c}^\star, \mathbf{x}^\star, \mathbf{y}^\star) = \text{Opt(R-RDP)},$$

which concludes the proof. □

In what follows we propose to solve the M-RDP problem by mapping it to the UFL-DC problem introduced in Section V-A. For an instance of the M-RDP problem we first create a set of facilities $\mathcal{U}$, with $|\mathcal{U}| = |\mathcal{M}|$, where facility $u_i \in \mathcal{U}$ corresponds to MEC node $i \in \mathcal{M}$, and $q_{u_i} = p_i - |\mathcal{M}|\beta_i$. We then create a set of clients $\mathcal{K}$, with $|\mathcal{K}| = |\mathcal{F}|$, where client $k_f \in \mathcal{K}$ corresponds to VS $f \in \mathcal{F}$. The cost for serving client $k_f \in \mathcal{K}$ by facility u_i is $q_{k_f,u_i} = p_{f,i} + \frac{\beta_i}{\omega_i}|M|$. The delay constrains are set to $d_{k_f,u_i} = d_{f,i}$. Finally, in the M-RDP problem any two primary instances of the VSs in $\mathcal{F}_l \cup \mathcal{F}_n \cup \mathcal{F}_h$ are compatible with each other, therefore we set $r_{k_f,k_{f'}} = 1, \forall f, f' \in \mathcal{F}_l \cup \mathcal{F}_n \cup \mathcal{F}_h$. Since VS $f \in \mathcal{F}_h$ and its replica $f_c \in \mathcal{F}_c$ cannot be placed on the same MEC node, $r_{k_f,k_{f_c}} = 0\ \forall f \in \mathcal{F}_h$.

Now we can use Algorithm 1 to solve the constructed UFL-DC problem instance $(\mathcal{U}, \mathcal{K}, d_{k,u}, r_{k,k'})$. Let us denote by $(b_u, z_{k,u})$ the solution given by Algorithm 1. Since the M-RDP problem is not constrained by MEC node capacity ω_i, in the solution $(b_u, z_{k,u})$ facility u_i that corresponds to MEC node i serves $\sum_{k_f} z_{k_f,u_i}$ clients, corresponding to $\sum_{k_f} z_{k_f,u_i}$ VSs. If we place the VSs to MEC node i as implied by z_{k_f,u_i} (i.e., set $x^p_{f,i} = z_{k_f,u_i}$), we may overload the MEC nodes (i.e., $\sum_f x^p_{f,i} > \omega_i$). Therefore, we have to construct a feasible placement of the primary VS as follows.

Algorithm 2: LRA(1)-Primary instance placement

Input : MEC nodes $\mathcal{M}$ and VSs $\mathcal{F}$
Create UFL-DC problem instance $(\mathcal{U}, \mathcal{K}, d_{k,u}, r_{k,k'})$
Compute a solution $(b_u, z_{k,u})$ by Algorithm 1
for $i \in \mathcal{M}$ **do**
  if $c'_{L(i),T(i)} > 0$ *and* $N(i) \leq c'_{L(i),T(i)}$ **then**
    $c_i = 1$
for $f \in \mathcal{F}$ **do**
  Find i with $f \in \mathcal{F}_{L(i),T(i)}$, and $\sum_i x^p_{f,i} < \omega_i c_i$
  $x^p_{f,i} = 1$
Output : $c_i, x^p_{f,i}$

The solution $(b_u, z_{k,u})$ implies that on MEC nodes of type $t \in \mathcal{T}$ at location $l \in \mathcal{L}$ the number of primary VS instances to be placed is $(\sum_{i\in\mathcal{M}_{l,t}} \sum_{k_f} z_{k_f,u_i})$. Thus the total number of MEC nodes of type t to be opened at location l is

$$c'_{l,t} = \lceil \sum_{i\in\mathcal{M}_{l,t}} \frac{\sum_{k_f} z_{k_f,u_i}}{\omega_i} \rceil \leq \sum_{i\in\mathcal{M}_{l,t}} \lceil \frac{\sum_{k_f} z_{k_f,u_i}}{\omega_i} \rceil. \quad (28)$$

Let us denote by $\mathcal{F}_{l,t} = \{f \in \mathcal{F} | z_{k_f,u_i} = 1, i \in \mathcal{M}_{l,t}\}$ the set of VSs that have their corresponding clients served by facilities that correspond to MEC nodes $\mathcal{M}_{l,t}$. We can then place the primary instances of VSs $f \in \mathcal{F}_{l,t}$ at any MEC node (l,t,n), as long as $\sum_f x^p_{f,i} < \omega_i$. The pseudo-code for placing the primary instances is shown in Algorithm 2.

Proposition 4. *Consider an M-RDP problem instance, and its corresponding UFL-DC problem instance. Let us denote by* $(b_u, z_{u,k})$ *the solution given by Algorithm 1 for the UFL-DC problem instance above. Then*

$$\begin{aligned} & \sum_i b_{u_i}(p_i - |\mathcal{M}|\beta_i) + \sum_f \sum_i \left(p_{f,i} + |M|\frac{\beta_i}{\omega_i}\right) z_{k_f,u_i} \\ & \leq H_{|\mathcal{F}|} \cdot \textit{Opt(UFL-DC)} = H_{|\mathcal{F}|} \cdot \textit{Opt(M-RDP)} \\ & \leq H_{|\mathcal{F}|} \cdot \textit{Opt(RDP)}. \end{aligned} \quad (29)$$

Proof. The first inequality holds as Algorithm 2 is a $H_{|\mathcal{F}|}$-approximation solution to the constructed UFL-DC problem (Lemma 3). Furthermore, the equality holds because the objective functions of the M-RDP problem and the constructed UFL-DC problem are the same. Finally, the last inequality holds by Propositions 2 and 3. □

Placement of the secondary instances: What remains to be solved is the placement of the secondary VSs instances of the stateful VSs $f \in \mathcal{F}_n$ as the other VSs in $\mathcal{F}_l \cup \mathcal{F}_h \cup \mathcal{F}_c$ do not require secondary instances. We handle the placement of the secondary VS instances by constructing and solving a maximal matching problem. For convenience, let us define the set $\mathcal{F}^p_i = \{f \in \mathcal{F} | x^p_{f,i} = 1\}$ of VSs that have their primary

Algorithm 3: LRA(2)-Secondary instance placement

Input : c_i, $x^p_{f,i}$

1 **for** $i \in \mathcal{M}$ *with* $c_i = 1$, $N(i) \leq c'_{L(i),T(i)}$ **do**
2 Construct bipartite graph $\mathcal{G}_i$
3 Compute $\mathcal{E}^{max}_i$ by Hopcroft-Karp Algorithm [27]
4 **if** $|\mathcal{E}^{max}_i| = |\mathcal{F}^p_i|$ **then**
5 **for** $(f, v_{i,m}) \in \mathcal{E}^{max}_i$ **do**
6 $x^s_{f,i} = 1$
7 **else**
8 $c_{i'} = 1$ with $i' \in \mathcal{M}_{L(i),T(i)}$, and $N(i') = c'_{L(i),T(i)} + 1$
9 $x^s_{f,i'} = 1 \; \forall f \in \mathcal{F}^p_i$

Output : c_i, $x^s_{f,j}$

instances placed on MEC node i. Recall that a_i defined in (10) is the amount of resources that are available for the placement of secondary VS instances on MEC node i. For each set of VSs $\mathcal{F}^p_i$, we create a bipartite graph $\mathcal{G}_i = (\mathcal{F}^p_i, \mathcal{V}_i, \mathcal{E}_i)$ as follows. For each MEC node that is open, we add a_i vertices to $\mathcal{V}_i$, and we denote by $v_{i,m}, m \leq a_i$ the m^{th} vertex added for MEC node i. We then add an edge $(f, v_{i,m}) \forall m \leq a_i$ to $\mathcal{E}_i$ if MEC node i satisfies the delay requirement of $f \in \mathcal{F}^p_i$ (i.e., $d_{f,i} \leq D_f$).

Now we are ready to place the secondary VS instances using Algorithm 3. For each MEC node that is opened by Algorithm 2 we construct a bipartite graph $\mathcal{G}_i$, and we compute a maximal matching by the Hopcroft-Karp algorithm [27] (lines 2-3). We denote by $\mathcal{E}^{max}_i$ the set of edges included in the maximal matching. For each bipartite graph $\mathcal{G}_i$, if the number of edges in $\mathcal{E}^{max}_i$ is equal to the number of VSs in $\mathcal{F}^p_i$, it indicates that each vertex that corresponds to a VS can be mapped to a vertex that corresponds to a MEC resource uniquely. For each edge $(f, v_{i,m}) \in \mathcal{E}^{max}_i$ we then place the secondary instance of f on MEC node i, i.e., $x^s_{f,i} = 1$ (lines 5-6). Otherwise, we open a MEC node i' of the same type and at the same location as MEC node i (line 8). Since $\omega'_i = \omega_i$, MEC node i' has enough capacity for placing the secondary instances of VSs $\mathcal{F}^p_i$, and thus we set $x^s_{f,i'} = 1 \; \forall f \in \mathcal{F}^p_i$ (line 9).

C. Approximation Ratio Bound

The above algorithm clearly executes in polynomial time, and as we show next, has a bounded approximation ratio.

Lemma 4. *Let* (**c**,**x**) *be the solution provided by the LRA algorithm for an instance of the RDP problem. Then*

$$E(\mathbf{c}, \mathbf{x}) \leq 3 \cdot H_{|\mathcal{F}|} Opt(RDP), \tag{30}$$

where $H_{|\mathcal{F}|}$ *is the harmonic series and* $H_{|\mathcal{F}|} \approx log(|\mathcal{F}|)$.

Proof. Recall that in the first part of the LRA algorithm, the algorithm opens $c'_{l,t}$ MEC nodes of type t at location l,

$$c'_{l,t} \leq \sum_{i \in \mathcal{M}_{l,t}} \lceil \frac{\sum_{k_f} z_{k_f,u_i}}{\omega_i} \rceil \leq \sum_{i \in \mathcal{M}_{l,t}} \left(\frac{\sum_{k_f} z_{k_f,u_i}}{\omega_i} + b_{u_i} \right) \tag{31}$$

The first inequality holds by (28). The second inequality holds because when $b_{u_i} = 0$ no client can be served by facility u_i and therefore $\lceil \frac{\sum_{k_f} z_{k_f,u_i}}{\omega_i} \rceil = 0$. Otherwise,

$$\lceil \frac{\sum_{k_f} z_{k_f,u_i}}{\omega_i} \rceil \leq \left(\frac{\sum_{k_f} z_{k_f,u_i}}{\omega_i} + 1 \right) = \left(\frac{\sum_{k_f} z_{k_f,u_i}}{\omega_i} + b_{u_i} \right).$$

For convenience, let us define the indicator

$$1_{|c'_{l,t}>0} = \begin{cases} 1, & c'_{l,t} > 0 \\ 0, & \text{Otherwise} \end{cases} \tag{32}$$

Clearly, when $c'_{l,t} = 0$, $\sum_{i \in \mathcal{M}_{l,t}} b_{u_i} = 0$ and $1_{|c'_{l,t}>0} = 0$. Otherwise, $1_{|c'_{l,t}>0} = 1 \leq \sum_{i \in \mathcal{M}_{l,t}} b_{u_i}$. Therefore

$$1_{|c'_{l,t}>0} \leq \sum_{i \in \mathcal{M}_{l,t}} b_{u_i}. \tag{33}$$

In the stage of the secondary instance placement, the algorithm may open one extra node for each MEC node type if $c'_{l,t} > 0$. Let us denote by $c_{l,t}$ the total number of MEC nodes of type t opened by the algorithm at location l,

$$\begin{aligned} c_{l,t} &\leq c'_{l,t} + 1_{|c'_{l,t}>0} \\ &\leq \sum_{i \in \mathcal{M}_{l,t}} \left(\frac{\sum_{k_f} z_{k_f,u_i}}{\omega_i} + b_{u_i} \right) + \sum_{i \in \mathcal{M}_{l,t}} b_{u_i} \\ &= \sum_{i \in \mathcal{M}_{l,t}} \left(\frac{\sum_{f} z_{f,u_i}}{\omega_i} + 2b_{u_i} \right) \end{aligned} \tag{34}$$

Recall that MEC nodes of the same type and at the same location have the same energy consumption. We can thus denote by $p_{l,t}$ the base energy consumption of MEC nodes of type t at location l. Then the cost of solution $(\mathbf{c}, \mathbf{x})$ is

$$\begin{aligned} E(\mathbf{c}, \mathbf{x}) &= \sum_l \sum_t c_{l,t} p_{l,t} + \sum_i \sum_f x^p_{f,i} p_{f,i} \\ &\leq \sum_l \sum_t \sum_{i \in \mathcal{M}_{l,t}} \left(\frac{\sum_f z_{k_f,u_i}}{\omega_i} + 2b_{u_i} \right) p_{l,t} + \sum_i \sum_f z_{k_f,u_i} p_{f,i} \\ &\leq \sum_i \left(\frac{\sum_f z_{k_f,u_i}}{\omega_i} + 2b_{u_i} \right) p_i + \sum_i \sum_f z_{k_f,u_i} p_{f,i} \\ &= 2\sum_i b_{u_i} p_i + \sum_i \sum_f \frac{p_i}{\omega_i} z_{k_f,u_i} + \sum_i \sum_f z_{k_f,u_i} p_{f,i} \end{aligned} \tag{35}$$

Finally, let us substitute $\beta_i = \frac{p_i}{3|\mathcal{M}|}$ into (29) to obtain

$$\begin{aligned} &2\sum_i b_{u_i} p_i + \sum_i \sum_f \frac{p_i}{\omega_i} z_{k_f,u_i} + 3\sum_i \sum_f z_{k_f,u_i} p_{f,i} \\ &\leq 3H_{|\mathcal{F}|} \cdot \text{Opt(RDP)}. \end{aligned} \tag{36}$$

Then, (30) follows by (35) and (36). □

Finally, we show that the LRA algorithm terminates after a finite number of iterations.

Lemma 5. *The LRA algorithm terminates after at most* $2(|\mathcal{F}|+|\mathcal{M}|)$ *iterations.*

Proof. To begin with, let us focus on the first part of LRA (Algorithm 2). Line 2 of Algorithm 2 relies on Algorithm 1 to compute the placement of the primary instances. Since

Figure 2: The simulated area, based on a map of the city of Milan, Italy. Triangles show the considered locations of the MEC nodes.

Algorithm 1 chooses one subset that corresponds to at least one VS at each iteration, Algorithm 1 terminates in at most $|\mathcal{F}|$ iterations. Furthermore, we observe that the remaining parts of Algorithm 2 require $|\mathcal{F}|+|\mathcal{M}|$ iterations. The second part of LRA (Algorithm 3) needs exactly $|\mathcal{M}|$ iterations to compute the placement of the secondary VS instances, as the pseudo-code shows. This proves the result. □

VI. Numerical Results

For the evaluation we simulated an urban area based on the map of the center of the city of Milan, Italy, covered by a set $\mathcal{B}$ of BSs of 581 LTE BSs (http://opencellid.org/). The set $\mathcal{L}$ of MEC locations is chosen as a subset of $\mathcal{B}$ using a local search heuristic for the K-means problem; the MEC locations are shown by triangles in Figure 2. The backhaul network topology is a minimum cost spanning tree of $\mathcal{B}$.

We used the number of CPU cores as the capacity of a hardware configuration t, and we used a set of four hardware configurations $\mathcal{T} = \{2, 4, 8, 16\}$. The maximal energy consumption per CPU core p_t^{core} was chosen uniform at random between 2W and 20W based on the technical specifications of recent Intel CPUs. We choose the base energy consumption P_t of MEC node type t uniformly between $[65\%, 230\%]$ of the maximal energy consumption of its CPU cores $p_t^{core}\omega_t$, which is based on the experimental results in [28]. In the simulations we consider that the CPUs are the main source of energy consumption for executing the VS instances and thus the energy consumption of executing a VS instance on a MEC node of type t is chosen uniformly on $[0.5, 1] \times p_t^{core}$. This assumption is reasonable, as CPUs usually account for a large portion of the energy consumption of a MEC server, e.g., measurement results in [28] show that the CPU may account for 30% to 60% of the energy consumption of a single CPU server; the ratio may be even higher in servers with multiple CPUs and solid state drives.

At each location $l \in \mathcal{L}$ we selected 3 hardware configurations from $\mathcal{T}$ as the configurations that can be deployed. This corresponds to typical technical constraints, such as the available energy supply at a site and constraints on physical dimensions. As baselines for comparison we use three greedy algorithms.

Least Opening Cost First: The first greedy algorithm attempts to open MEC nodes in ascending order of their opening cost; we refer to it as the least open cost first (LOC) algorithm. The pseudo-code of LOC is shown in Algorithm 4. In line 1, the algorithm initializes the set Ω of VSs that have not yet been placed. Then in each iteration it finds the MEC node type t and location l with minimal base energy consumption (line 3). Then the algorithm opens the minimal number of MEC nodes of type i at location l (i,e., $c'_{l,t} = \lceil \frac{|\Omega \cap \mathcal{F}_{l,t}|}{\omega_i} \rceil$ in lines 4-5), to place the primary instances of all the VSs in $\Omega \cap \mathcal{F}_{l,t}$. At the end, we use Algorithm 3 to compute the placement of secondary instances of the stateful VSs $\mathcal{F}_n$.

Algorithm 4: Least Opening Cost First Algorithm

1 $\Omega = \mathcal{F}$
2 **while** $\Omega \neq \emptyset$ **do**
3 $\quad [l, t] = \text{argmin}_{l,t}\{p_{l,t} | c'_{l,t} = 0\}$
4 $\quad c'_{l,t} = \lceil \frac{|\Omega \cap \mathcal{F}_{l,t}|}{\omega_i} \rceil$
5 $\quad c_i = 1, \forall i$ with $N(i) \leq c'_{l,t}, T(i) = t, L(i) = l$
6 $\quad$ **for** $f \in \Omega \cap \mathcal{F}_{l,t}$ **do**
7 $\quad\quad$ Find $i \in \mathcal{M}_{l,t}$ with $c_i = 1$, $\sum_f x^p_{f,i} < \omega_i$
8 $\quad\quad x^p_{f,i} = 1$, and $\Omega = \Omega \setminus \{f\}$
9 Use Algorithm 3 to compute $x^s_{f,j}$, $\forall f \in \mathcal{F}_n$

Algorithm 5: Least Execution Cost First Algorithm

1 **for** $f \in \mathcal{F}$ **do**
2 $\quad i = \text{argmin}_i\{p_{f,i} | i \in \mathcal{M}, \sum_f x^p_{f,i} < \omega_i, d_{f,i} \leq D_f\}$
3 $\quad c_i = 1$ and $x^p_{f,i} = 1$
4 Use Algorithm 3 to compute $x^s_{f,j} \forall\ f \in \mathcal{F}_n$

Least Execution Cost First: The second greedy algorithm attempts to minimize the execution cost; we refer to it as the least execution cost first (LEC) algorithm. The pseudo-code of LEC is shown in Algorithm 5. LEC places primary instance of VS f on MEC node i with the lowest execution energy consumption, among all the feasible MEC nodes that have available computational resources (i.e., $\sum_f x^p_{f,i} < \omega_i$) and can satisfy the delay requirement of f (i.e., $d_{f,i} \leq D_f$) in lines 2-3. At the end, LEC uses Algorithm 3 to compute the placement of secondary instances of the stateful VSs $\mathcal{F}_n$ (line 4).

Non-joint Dimensioning and Placement: The third greedy algorithm considers the MEC node dimensioning and VS instance placement in two separate steps, and we refer to it as the non-joint dimensioning and placement (NJDP) algorithm. The pseudo-code of NJDP is shown in Algorithm 6. The NJDP algorithm first dimensions the MEC resources (lines 1-2). At each location l, MEC nodes of type t can satisfy the delay requirement of $|\{f \in \mathcal{F} | d_{f,l,t} \leq D_f\}|$ VSs, and thus the algorithm opens $\lceil \frac{|\{f \in \mathcal{F} | d_{f,l,t} \leq D_f\}|}{\omega_{l,t}} \rceil$ MEC nodes of type t at location l. The NJDP algorithm then handles the VS placement (lines 3-6). The NJDP algorithm assigns the primary VS instance of f to the MEC node with the lowest execution cost, among all the MEC nodes that have available computational resources and can satisfy the delay requirement of f (lines 4-5), and then uses Algorithm 3 to place the secondary instances (line 6).

Algorithm 6: Non-joint Dimensioning and Placement Algorithm

// Dimensioning
for $l \in \mathcal{L}$ **do**
  Open $\lceil \frac{|\{f \in \mathcal{F} | d_{f,l,t} \leq D_f\}|}{\omega_{l,t}} \rceil$ MEC nodes of type t
// Placement
for $f \in \mathcal{F}$ **do**
  $i = \text{argmin}_i \{p_{f,i} | i \in \mathcal{M}, \sum_f x^p_{f,i} < \omega_i, d_{f,i} \leq D_f\}$
  $x^p_{f,i} = 1$
Use Algorithm 3 to compute $x^s_{f,j} \forall\ f \in \mathcal{F}_n$

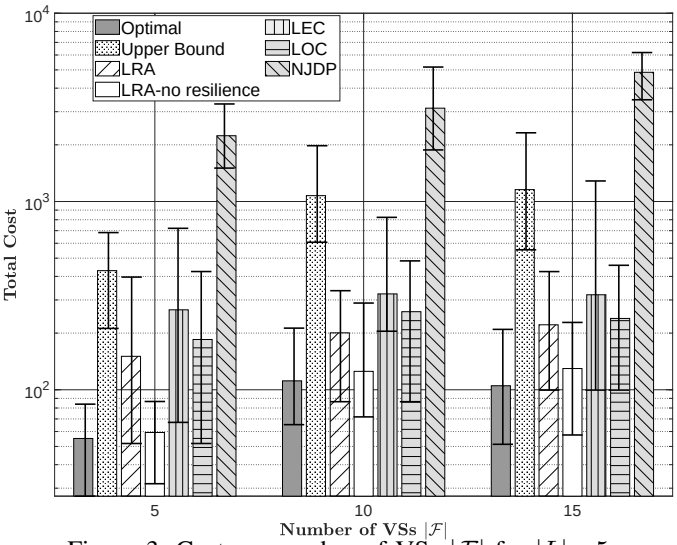

Figure 3: Cost vs. number of VSs $|\mathcal{F}|$ for $|L|= 5$

Furthermore, to show the cost of considering resilience, we also consider a variant of the LRA algorithm that consider all VSs $\mathcal{F}$ are non-critical stateless, referred to as the LRA-no resilience algorithm. The results shown are the averages of 500 simulations.

A. Cost Performance

To evaluate the cost performance of LRA, we start with a system with $|\mathcal{L}|= 5$ locations. Besides the baselines above, we also compute optimal solutions by solving the RDP problem as an IP problem, and then compute an upper bound as the product of the optimal cost and the approximation ratio $3H_{|\mathcal{F}|}$. Figure 3 shows the total energy cost of the algorithms as a function of the number of VSs $|\mathcal{F}|$. The error bars show the 5% and 95% percentiles. The results show that LRA performs close to the optimal solution. For example, when $|\mathcal{F}|= 10$ the total cost of LRA is within a factor of two of the optimal solution. It is also interesting to observe that the total cost of the NJDP algorithm is much higher than that of the other algorithms. For example, when $|\mathcal{F}|= 10$ the total cost of the NJDP algorithm is about fifteen times of the total cost of LRA. This is because VS placement is not considered during MEC resource dimensioning in NJDP, and thus the resource demand of each VS has been redundantly addressed at multiple locations. This observation shows that our joint approach is necessary for efficiently solving the dimensioning and placement problem. In the following, we will focus on LRA, LRA-no resilience, LEC and LOC, and evaluate them in larger systems to obtain more insight.

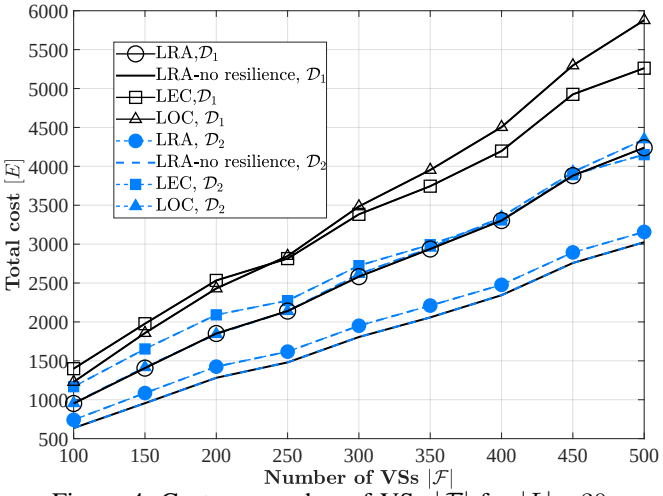

Figure 4: Cost vs. number of VSs $|\mathcal{F}|$ for $|L|= 30$

Figure 4 shows the total energy cost of LRA, LRA-no resilience, LEC and LOC as a function of the number of VSs $|\mathcal{F}|$ for two scenarios. The two scenarios differ in the fraction of the types of VSs, and allow to investigate the impact of different types of VSs on the performance of the algorithms. In the first scenario (distribution $\mathcal{D}_1$) VS types are uniformly distributed among $\{\text{non-critical stateless}, \text{stateful}, \text{critical}\}$, while in the second scenario (distribution $\mathcal{D}_2$) all the VSs are stateful. The results for $\mathcal{D}_1$ and $\mathcal{D}_2$ are shown using solid and dashed lines, respectively. It is interesting to observe that LRA, LEC and LOC achieve higher costs in $\mathcal{D}_1$ than in $\mathcal{D}_2$. This is mainly due to the critical VSs present in $\mathcal{D}_1$, as critical VSs require more computational resources than stateful VSs, i.e., each critical VS requires dedicated resources for its two primary instances, while each stateful VS only requires one dedicated resource for its primary instance and runs its secondary instance on shared computational resources. Note that since LRA-no resilience treats all VSs as non-critical stateless, LRA-no resilience performs exactly the same in $\mathcal{D}_1$ and $\mathcal{D}_2$.

In what follows, let us first focus on the performance of the algorithms in $\mathcal{D}_1$. Figure 4 shows that the LRA algorithm outperforms the two greedy algorithms, and the performance gap increases as $|\mathcal{F}|$ increases. For example, for $|\mathcal{F}|= 500$ the energy consumption of the LRA algorithm is about 27% and 20% less than that of LEC and LOC, respectively. This is because the LEC and the LOC algorithms only focus on minimizing the base energy consumption and the execution energy consumption, respectively, while the LRA algorithm considers both. Compared to LRA-no resilience, the energy consumption of LRA increases up to 34% due to providing resilience to the stateful and critical VSs. The increased cost comes from opening extra MEC nodes and from the energy consumption for executing the replicas of the critical VSs. Furthermore, the reasoning above also explains the phenomenon that in $\mathcal{D}_2$ the performance gaps of the four algorithms are

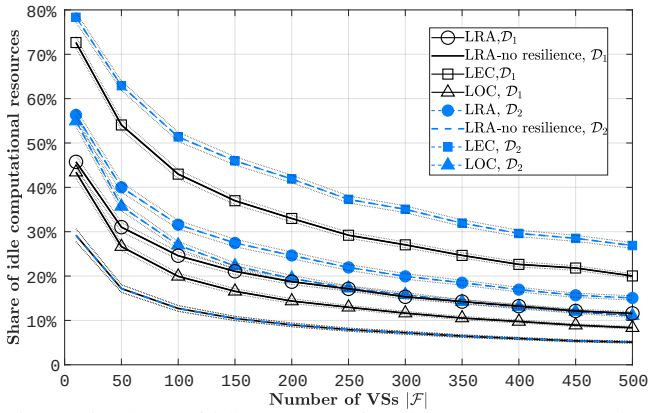

Figure 5: Share of idle computational resources vs. number of VSs $|\mathcal{F}|$ for $|L|= 30$

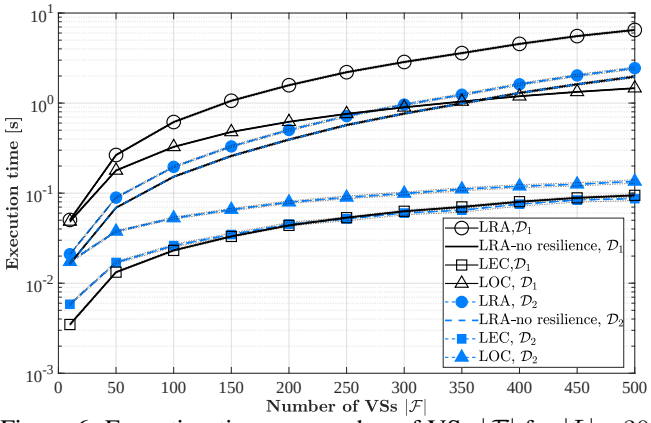

Figure 6: Execution time vs. number of VSs $|\mathcal{F}|$ for $|L|= 30$

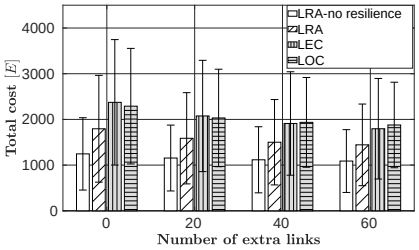

Figure 7: Total cost when adding extra links in the mobile backhaul $\mathcal{G}$.

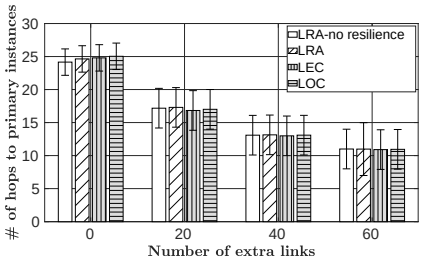

Figure 8: Number of hops between applications and the primary instances when adding extra links in the mobile backhaul $\mathcal{G}$.

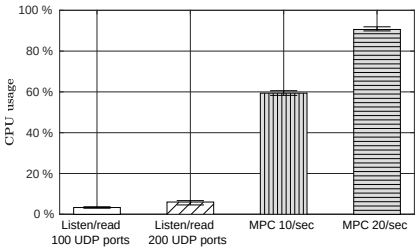

Figure 9: CPU usage for processes that receive data only and for model predictive control (MPC)

similar to that in D_1.

Besides the energy consumption, we also use the share of idle computational resources (SICR) as a second performance metric,

$$\text{SICR} = 1 - \frac{|\mathcal{F}|}{\sum_{i\in\mathcal{M}} \omega_i c_i}. \tag{37}$$

The SICR is a measure of how well the algorithms utilize the computational resources of the opened MEC nodes. Figure 5 shows the SICR of the algorithms as a function of the number of VSs $|\mathcal{F}|$ for the same scenario as Figure 4. The results show that for all the algorithms the SICR decreases as $|\mathcal{F}|$ increases under both VS type distributions $\mathcal{D}_1$ and $\mathcal{D}_2$. This is because when $|\mathcal{F}|$ is small, the VSs are sparsely distributed within the simulated area. Therefore, due to the delay requirement of the VSs, it could be the case that a VS is opened to serve much fewer VSs than its capacity allows. Among all the algorithms, the LEC algorithm has the highest SICR, as it minimizes the execution energy consumption of each VS individually without coordinating the VSs. On the contrary, LOC has a lower SICR than the LRA and LEC, as the LRA attempts to fill the opened MEC nodes before opening new MEC nodes. Further, the results show that the LRA-no resilience algorithm achieves lower SICR than the LRA algorithm in both $\mathcal{D}_1$ and $\mathcal{D}_2$. This is mainly because the LRA algorithm reserves idle resources for the secondary instances of the stateful VSs. The need for reserving idle resources for secondary instances also explains the observation that LRA, LEC and LOC have higher SICR under $\mathcal{D}_2$ than under $\mathcal{D}_1$, as there are more stateful VSs.

B. *Efficiency*

Figure 6 shows the execution time for the algorithms as a function of the number of VSs $|\mathcal{F}|$ for the same scenario as in Figure 4. The results show that the LEC algorithm has the lowest execution time under both distributions $\mathcal{D}_1$ and $\mathcal{D}_2$, as using this algorithm the number of iterations for placing the primary instances is determined by $|\mathcal{F}|$. The LRA-no resilience algorithm consumes less time than the LRA algorithm, as the LRA-no resilience algorithm does not consider replicas of the critical VSs, and it also skips the placement of secondary instances. Furthermore, the results show that LRA, LRA-no resilience and LOC have a higher execution time in $\mathcal{D}_1$ than in $\mathcal{D}_2$. This is because there are more primary instances to be placed in $\mathcal{D}_1$ than under $\mathcal{D}_2$ due to the critical instances present in $\mathcal{D}_1$, and using those algorithms the placement of primary instances has a bigger impact on the execution time than that of the secondary instances. It is worthwhile to note that the execution time of the LRA algorithm is arguably negligible compared to the timescale at which a MEC operator would in practice update the deployment of MEC nodes and services.

C. *Impact of backhaul topology*

Next we evaluate the impact of the backhaul topology on the cost and on the latency performance. For this we modified the backhaul topology by adding extra links to the original spanning tree using an iterative greedy algorithm: in each iteration we add an extra link between the first and the last nodes on the longest shortest path in the backhaul, thereby

reducing the diameter of the graph. Figure 7 shows the total cost for $|\mathcal{F}|= 200$ with 0, 20, 40, and 60 extra links added, the error bars show the 5% and 95% percentiles. In simulations we choose the types of VSs according to the distribution $\mathcal{D}_1$ considered in Figure 4, as $\mathcal{D}_1$ includes all VS types and thus is more general than $\mathcal{D}_2$. Overall, the total cost decreases as the number of extra links increases with a decreasing marginal gain. For example, compared to the spanning tree topology (i.e., 0 extra links) adding 20 extra links (i.e., about 2% increase in number of links) reduces the total cost of the LRA algorithm by about 20%. This is because adding extra links increases the number of MEC nodes that a VS can be placed on, which gives the algorithm more flexibility when solving the problem. Another interesting observation is that adding extra links to the network topology has a higher impact on the worst-case energy cost than on the average cost or on the 5% cost, i.e., it helps to make the energy consumption more predictable.

Second we consider number of network hops between the BS serving the application and the locations of the primary instances of the VSs as a performance metric for the backhaul delay. We consider the number of hops only, as the propagation delay in mobile backhaul networks is almost negligible (e.g., fiber), and thus the delay in the mobile backhaul mainly comes from packet processing in each hop. Figure 8 shows the average number of hops together with the 5% and 95% percentiles. The results show that as the number of extra links increases, the average number of hops decreases significantly for all the algorithms. The results indicate that improving the backhaul network is very important for MEC to serve VSs with stringent delay requirements. The result also show that the LRA and the LRA-no resilience algorithms perform close to the greedy algorithms and even outperform the greedy algorithms when no extra links are added. This shows that the LRA algorithm does not sacrifice the backhaul delay performance while reducing the energy consumption.

D. Experimental Results

To validate our resource consumption model for the primary and secondary instances, we measured the CPU usage for two different processes on a desktop computer with 4 CPU cores. The first process implements model predictive control (MPC) for balancing an inverted pendulum. MPC is an advanced process control method that uses dynamic linear models, and is widely used in the process industry in chemical plants and oil refineries. We use the MPC process to simulate the main activity of primary VS instances. The second process is a UDP server that listens to and reads packets from a number of UDP ports. Each UDP port receives about 60 packets per second. We use this process to capture the main activity of the secondary instances, which receive state update messages from the primary instances periodically. Figure 9 shows the average CPU usage of the UDP server with 100 and 200 UDP ports, and the MPC process with a control frequency of 10 and 20 per second, respectively. The results show that compared to the MPC process, the UDP server uses significantly less computational resources. For example, when there are 200 ports the CPU usage of the UDP server is about 5%. These results justify our assumption that properly implemented secondary instances can be hosted together on a shared computational resource (e.g., together with the operating systems).

VII. Conclusion

We have proposed an approximation algorithm for solving the joint problem of resource dimensioning and placement for dependable virtualized services, for minimizing energy consumption under resource and latency constraints. The proposed approximation algorithm is based on Lagrangian relaxation and solves the problem in two steps. Extensive simulations show that our proposed algorithm significantly outperforms three greedy baseline algorithms and is computationally efficient. Overall our results indicate that service availability requirements can effectively be incorporated into SLAs, and can be used for making MEC dimensioning and service placement efficient. An interesting extension of our work is to consider virtualized services that have heterogeneous resource requirements, and the case of shared computation resources subject to schedulability constraints. A further extension of the model would be to consider services that require data from devices connected to multiple base stations.

Appendix: Proof of Lemma 3

Proof. Algorithm 1 was applied to the non-metric UFL problem in [26]. Since the complete proof is missing in [26], we provide the complete proof as follows, based on an approach similar to Theorem 1.11 in [29]. Let us denote by $\Omega^{(n)}$ the set of clients that are not served at the beginning of iteration n, and by $\mathcal{U}^{(n^*)}$ the set of facilities that the optimal solution would use to serve the clients in $\Omega^{(n)}$. Let us denote by $(b^*_u, z^*_{k,u})$ the optimal solution of the UFL-DC problem. Then,

$$v(u^{(n)}, \mathcal{K}^{(n)}) \leq \frac{\sum_{u\in\mathcal{U}^{(n^*)}} (b^*_u q_u + \sum_k z^*_{k,u} q_{k,u})}{\sum_{u\in\mathcal{U}^{(n^*)}} \sum_k z^*_{k,u}} \leq \frac{\text{Opt(UFL-DC)}}{|\Omega^{(n)}|}. \tag{38}$$

The first inequality holds as $v(u^{(n)}, \mathcal{K}^{(n)})$ is the minimal cost efficiency at iteration n (line 4, Algorithm 1). The second inequality holds as $\sum_{u\in\mathcal{U}^{(n^*)}} (b^*_u q_u + \sum_k z^*_{k,u} q_{k,u}) \leq$ Opt(UFL-DC) and $\sum_{u\in\mathcal{U}^{(n^*)}} \sum_k z^*_{k,u} \geq |\Omega^{(n)}|$.

Let us define the cost efficiency of serving client k as $v(k) = v(u^{(n)}, \mathcal{K}^{(n)})$, where $k \in \mathcal{K}^{(n)}$. Assume that the Algorithm 1 converges in N iterations, then

$$\sum_{n=1}^{N} \sum_{k\in\mathcal{K}^{(n)}} v(k) \leq \sum_{n=1}^{N} \sum_{k\in\mathcal{K}^{(n)}} \frac{\text{Opt(UFL-DC)}}{|\Omega^{(n)}|} \tag{39}$$

$$\leq \sum_{n=1}^{N-1} \left(\frac{\text{Opt(UFL-DC)}}{|\Omega^{(n)}|} + \frac{\text{Opt(UFL-DC)}}{|\Omega^{(n)}|-1} + \ldots + \frac{\text{Opt(UFL-DC)}}{|\Omega^{(n)}|-|\Omega^{(n+1)}|+1} \right)$$

$$+ \left(\frac{\text{Opt(UFL-DC)}}{\Omega^{(N)}} + \frac{\text{Opt(UFL-DC)}}{\Omega^{(N)}-1} + \ldots + \frac{\text{Opt(UFL-DC)}}{1} \right)$$

$$= \text{Opt(UFL-DC)} \left(\frac{1}{|\mathcal{K}|} + \ldots \frac{1}{3} + \frac{1}{2} + \frac{1}{1} \right) = H_{|\mathcal{K}|}\text{Opt(UFL-DC)},$$

where $H_{|\mathcal{K}|}$ is the harmonic series and $H_{|\mathcal{K}|} \approx log(|\mathcal{K}|)$.

Finally, the cost of the solution given by Algorithm 1 is

$$\sum_u b_u q_u + \sum_u \sum_{k\in\mathcal{K}} z_{k,u} q_{k,u} \leq \sum_{n=1}^{N} \sum_{k\in\mathcal{K}^{(n)}} v(k) \leq H_{|\mathcal{K}|}\text{Opt(UFL-DC)}, \tag{40}$$

which proves the result. □

REFERENCES

[1] K. Poularakis, J. Llorca, A. M. Tulino, I. Taylor, and L. Tassiulas, "Joint service placement and request routing in multi-cell mobile edge computing networks," in *Proc. of IEEE INFOCOM*, 2019, pp. 10–18.

[2] T. He, H. Khamfroush, S. Wang, T. La Porta, and S. Stein, "It's hard to share: joint service placement and request scheduling in edge clouds with sharable and non-sharable resources," in *Proc. of IEEE ICDCS*, 2018, pp. 365–375.

[3] S. Pasteris, S. Wang, M. Herbster, and T. He, "Service placement with provable guarantees in heterogeneous edge computing systems," in *Proc. of IEEE INFOCOM*, 2019, pp. 514–522.

[4] S. Wang, M. Zafer, and K. K. Leung, "Online placement of multi-component applications in edge computing environments," *IEEE Access*, vol. 5, pp. 2514–2533, 2017.

[5] H. Badri, T. Bahreini, D. Grosu, and K. Yang, "Energy-aware application placement in mobile edge computing: A stochastic optimization approach," *IEEE Trans. Parallel Distrib. Syst*, 2019.

[6] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," *IEEE IoT Journal*, vol. 5, no. 1, pp. 450 – 465, 2017.

[7] C. Alippi, R. Fantacci, D. Marabissi, and M. Roveri, "A cloud to the ground: The new frontier of intelligent and autonomous networks of things," *IEEE Comm. Mag.*, vol. 54, no. 12, pp. 14–20, 2016.

[8] I. Jang, S. Choo, M. Kim, S. Pack, and G. Dán, "The software-defined vehicular cloud: A new level of sharing the road," *IEEE Vehicular Technology Magazine*, vol. 12, no. 2, pp. 78–88, June 2017.

[9] S. N. Shirazi, A. Gouglidis, A. Farshad, and D. Hutchison, "The extended cloud: Review and analysis of mobile edge computing and fog from a security and resilience perspective," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 11, pp. 2586–2595, 2017.

[10] R. Ford, A. Sridharan, R. Margolies, R. Jana, and S. Rangan, "Provisioning low latency, resilient mobile edge clouds for 5g," in *IEEE INFOCOM Workshops*, 2017, pp. 169–174.

[11] C. Colman-Meixner, C. Develder, M. Tornatore, and B. Mukherjee, "A survey on resiliency techniques in cloud computing infrastructures and applications," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 3, pp. 2244–2281, 2016.

[12] M. Unuvar, S. Tosi, Y. N. Doganata, M. G. Steinder, and A. N. Tantawi, "Selecting optimum cloud availability zones by learning user satisfaction levels," *IEEE Trans. Services Comput*, vol. 8, no. 2, pp. 199–211.

[13] L. Zhao, W. Sun, Y. Shi, and J. Liu, "Optimal placement of cloudlets for access delay minimization in sdn-based internet of things networks," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1334–1344, 2018.

[14] A. Ceselli, M. Premoli, and S. Secci, "Mobile edge cloud network design optimization," *IEEE/ACM Trans. Netw.*, vol. 25, no. 3, pp. 1818–1831, 2017.

[15] X. Wang, M. Razo, M. Tacca, and A. Fumagalli, "Planning and online resource allocation for the multi-resource cloud infrastructure," in *Proc. of IEEE ICC*, 2014, pp. 2938–2943.

[16] C. Develder, J. Buysse, B. Dhoedt, and B. Jaumard, "Joint dimensioning of server and network infrastructure for resilient optical grids/clouds," *EEE/ACM Trans. Netw.*, vol. 22, no. 5, pp. 1591–1606, 2013.

[17] N. Kherraf, H. A. Alameddine, S. Sharafeddine, C. Assi, and A. Ghrayeb, "Optimized provisioning of edge computing resources with heterogeneous workload in iot networks," *IEEE Trans. Netw. Service Manag*, 2019.

[18] M. Sviridenko, "An improved approximation algorithm for the metric uncapacitated facility location problem," in *Proc. of International Conference on Integer Programming and Combinatorial Optimization*. Springer, 2002, pp. 240–257.

[19] K. Jain, M. Mahdian, E. Markakis, A. Saberi, and V. V. Vazirani, "Greedy facility location algorithms analyzed using dual fitting with factor-revealing lp," *Journal of the ACM*, vol. 50, no. 6, pp. 795–824, 2003.

[20] S. Kekki, W. Featherstone, Y. Fang, P. Kuure, A. Li, A. Ranjan, D. Purkayastha, F. Jiangping, D. Frydman, G. Verin *et al.*, "Mec in 5g networks," *ETSI white paper*, vol. 28, pp. 1–28, 2018.

[21] Q.-V. Pham, F. Fang, V. N. Ha, M. J. Piran, M. Le, L. B. Le, W.-J. Hwang, and Z. Ding, "A survey of multi-access edge computing in 5g and beyond: Fundamentals, technology integration, and state-of-the-art," *IEEE Access*, vol. 8, pp. 116 974–117 017, 2020.

[22] V. Sinitsyn, "Jailhouse," *Linux Journal*, no. 252, p. 2, 2015.

[23] Z. Huang, Y. Chen, and J. Nieplocha, "Massive contingency analysis with high performance computing," in *Proc. IEEE power & energy society general meeting*, 2009, pp. 1–8.

[24] K. Bernhard and J. Vygen, *Combinatorial optimization: Theory and algorithms*, 3rd ed., 2008.

[25] A. Caprara, P. Toth, and M. Fischetti, "Algorithms for the set covering problem," *Annals of Operations Research*, vol. 98, no. 1-4, pp. 353–371, 2000.

[26] S. H. Vardhan. (2019, nov) Uncapacitated facility location. [Online]. Available: https://www.cs.cmu.edu/~anupamg/adv-approx/lecture4.pdf

[27] J. E. Hopcroft and R. M. Karp, "An n^5/2 algorithm for maximum matchings in bipartite graphs," *SIAM Journal on computing*, vol. 2, no. 4, pp. 225–231, 1973.

[28] T. Enokido, A. Aikebaier, and M. Takizawa, "An extended simple power consumption model for selecting a server to perform computation type processes in digital ecosystems," *IEEE Trans. Ind. Informat.*, vol. 10, no. 2, pp. 1627–1636, 2014.

[29] D. P. Williamson and D. B. Shmoys, *The design of approximation algorithms*. Cambridge university press, 2011.

Peiyue Zhao is a Ph.D. student at the Division of Network and Systems Engineering at KTH Royal Institute of Technology. He received his B.Sc degree in Automation from Beijing Institute of Technology, China in 2013, and his M.Sc. degree in Information and Communication Technology from KTH Royal Institute of Technology, Sweden in 2015. His research interests include resource management for mobile edge computing and radio resource management.

György Dán is Professor at KTH Royal Institute of Technology, Stockholm, Sweden. He received the M.Sc. in computer engineering from the Budapest University of Technology and Economics, Hungary in 1999, the M.Sc. in business administration from the Corvinus University of Budapest, Hungary in 2003, and the Ph.D. in Telecommunications from KTH in 2006. He worked as a consultant in the field of access networks, streaming media and videoconferencing 1999-2001. He was a visiting researcher at the Swedish Institute of Computer Science in 2008, a Fulbright research scholar at University of Illinois at Urbana-Champaign in 2012-2013, and an invited professor at EPFL in 2014-2015. He has been an area editor of Computer Communications since 2014 and of IEEE Trans. on Mobile Computing since 2019. His research interests include the design and analysis of content management and computing systems, game theoretical models of networked systems, and cyber-physical system security and resilience.