# Resilient Placement of Virtual Process Control Functions in Mobile Edge Clouds

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

***Abstract*—Virtual Process Control Functions (VPFs) are a promising solution for replacing hardware controllers in industrial control processes in order to improve operational efficiency. With the introduction of Mobile Edge Computing (MEC) in 5G networks, VPFs could even be executed on cloud resources close to the mobile network edge to further improve operational efficiency. Nonetheless, for this to happen, it is fundamental to ensure that the placement of VPFs be resilient to potential cyber-attacks and component failures, besides being efficient. In this paper we address this problem by considering that VPF placement costs are incurred by reserving MEC resources, executing VPF instances, and by data transmission. We formulate the VPF placement problem as an integer programming (IP) problem, with resilience as a constraint. We propose a VPF placement algorithm based on generalized Benders decomposition and based on linear relaxation of the resulting sub-problems, which effectively reduces the number of integer variables to that of the number of MEC nodes. We evaluate the proposed solution with respect to operational cost, efficiency, and scalability, and compare it with a greedy baseline algorithm. Extensive simulations show that our algorithm performs well in realistic scenarios.**

## I. Introduction

Various industries are embracing softwarization with the objective of improving operational efficiency and flexibility, and to reduce the capital expenditures for their industrial control processes [1]–[3]. Examples include energy distribution, manufacturing, healthcare, and the automotive industry. Legacy industrial control processes consist of real-time data collection from sensors, data processing in standalone hardware controllers, and control command execution through actuators. With softwarization, hardware controllers are replaced by software instances, referred to as Virtual Process Control Functions (VPFs). VPFs can be placed in commodity servers, can be flexibly provisioned on demand, and are easier to upgrade than legacy hardware controllers.

In today's industrial network architectures a natural choice for placing VPFs would be self-managed servers on the shop-floor or in a private cloud [4]. A promising future alternative could, however, be Mobile Edge Computing (MEC), which is expected to be a key enabler of 5G [5]. MEC brings distributed computing and storage resources close to end-users in mobile networks, with low latency and high bandwidth. By relying on MEC for VPF placement, together with low latency 5G wireless access, industries could further reduce their operational cost and improve the flexibility of their control processes.

This work was partly funded by the EU H2020 SUCCESS project, grant agreement No. 700416.

ISBN 978-3-901882-94-4 © 2017 IFIP

A critical aspect for the adoption of MEC for VPF placement is resilience, both to cyber attacks (e.g.,denial-of-service attacks and advanced persistent threats) and to the potential failure of communication (wireless access and the mobile backhaul) and computing resources (virtual machines or servers in MEC nodes). One way to achieve resilience is to execute redundant copies of VPF at multiple MEC nodes, akin to 1+1 redundancy, but this solution would lead to high operational cost, as the communication and computing resources used by the redundant VPF instances need to be paid for. The alternative approach is to restore a VPF instance on a different MEC node in case of a failure, depending on the failure scenario. Restoring a VPF instance on a different MEC node may, however, be limited due to other VPF instances running there already, and hence may require other VPF instances to be migrated, which makes the problem of optimal resilient VPF placement challenging.

In this paper, we address the problem of resilient VPF placement with the objective of minimizing the expected operational cost. We consider a system in which costs stem from making a particular MEC node available for use, from executing a VPF instance on a MEC node, and from transmitting data from the sensors to the VPF and from the VPF to the actuator, which depends on the location of the VPF. The set of available MEC nodes and communication links is determined by a given set of failure scenarios, and the objective is to find the minimum cost VPF placement subject to capacity and to resilience constraints, in expectation for all possible scenarios. We formulate the VPF placement problem as an integer programming (IP) problem, and propose an efficient iterative algorithm for computing the optimal solution based on generalized Benders decomposition and linear relaxation. Through linear relaxation the proposed Resilient VPF Placement (RVP) algorithm converts the IP problem into a mixed integer linear programming (MILP) with only a few integral variables. We prove convergence of the RVP algorithm, and numerical results show that the algorithm can reduce the expected cost significantly compared to a greedy baseline algorithm and scales well to moderate instances of the problem.

The rest of this paper is organized as follows. We review related work in Section II and introduce the system model in Section III. In Section IV we present our solution to the resilient VPF placement problem. We provide numerical results in section V, and we conclude the paper in section VI.

## II. Related Work

There has been a significant interest in cloud computing for industrial use cases, both concerning potential application areas and security aspects [6], [7], but the focus of these works is on architectures and requirements, rather than resource management.

Related to our work are recent works on resource allocation in MEC for sensor networks [8], [9]. Authors in [8] considered the problem of allocating visual sensors with correlated measurements to computing resources so as to maximize system capacity, and proposed a 3-approximation. The problem of allocating health sensors to health cloud servers to maximize system utility is proposed in [9], and was solved by an auction theory based approach. These papers consider resource allocation, but do not capture resilience requirements.

Also closely related to ours are recent works on the Virtual Network Function (VNF) placement problem [10]–[13]. These works consider the VNF placement problem in wireline networks, and focus on individual functions or on function chains.

Authors in [10] modeled the individual VNF placement problem as a generalized assignment problem, where the network functions are assigned to different cloud servers to minimize the total assignment cost, and provided and approximation algorithm based on linear relaxation and rounding. [11] formulated the on-demand individual VNF placement problem as a simple lazy facility location (SLFL) problem, and proposed two heuristic algorithms for on-line optimization.

The placement of chains of VNFs was modeled in [12] as an Integer Linear Problem (ILP), and numerical solutions were provided. [13] considered minimizing the number of computing units that each function chain is distributed over, and proposed a heuristic algorithm. The joint placement and path selection problem for VNFs chains was addressed in [14] to maximize the service capacity, and proposed algorithms for estimating link and processing capacity demands, and for allocating VNFs.

Resilience under link and node failure for VNF placement was considered in [15], the problem was formulated as an ILP, and numerical results were provided. Unlike this, in our work we consider a set of stochastic resilience scenarios caused by MEC nodes failure and communication links failure. We compute computational resource allocation for each resilience scenario and minimize the overall cost.

Closest to our work in terms of methodology is [16], where the problem of capacitated facility location subject to facility disruptions was addressed using Benders decomposition. Unlike [16], we also use infeasibility cuts to handle the infeasible region, which makes our algorithm more efficient.

To the best of our knowledge, ours is the first work to consider resilience constraints in optimizing the placement of VPFs, and to propose an iterative algorithm for computing the optimal VPF placement in a network of sensors, actuators and computing resources.

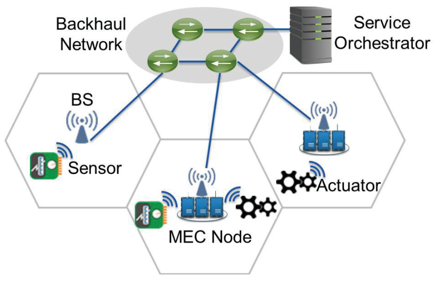

Figure 1. The communication and computing infrastructure consists of BSs, MEC nodes and a backhaul network. A service orchestrator monitors the status of the system and coordinates the allocation of communication and computing resources.

## III. System Model and Problem Formulation

We consider a system that consists of a set $\mathcal{B}$ of base stations (BSs), a set $\mathcal{S}$ of sensors, and a set $\mathcal{A}$ of actuators. A subset $\mathcal{M} \subseteq \mathcal{B}$ of the BSs is equipped with computational and storage resources and serve as MEC nodes [4], [17]. We denote by $\omega_m$ the computing capacity of MEC node $m \in \mathcal{M}$, and assume it to be an integer. This assumption is reasonable, for example, if capacity is the number of Virtual Machines (VMs), as industrial applications have tight delay requirements and require isolation for performance and security reasons; hence a single VPF would be allocated per VM.

The BSs are interconnected by a backhaul network (e.g., software defined mobile backhaul network). Via BSs, sensors and actuators can send and receive data wirelessly. Data communication through the wireless links and the backhaul network incurs cost, which we define in Section III-B.

Within the infrastructure above, we consider a set $\mathcal{F}$ of VPFs that process the data from sensors and send commands to actuators, and need to be allocated to MEC nodes for execution. For each function $f \in \mathcal{F}$ there is a set of sensors that capture data needed by $f$, and there is a set of actuators that VPF $f$ sends commands to. We use $y_{f,s} \in \mathbf{y}$ and $z_{f,a} \in \mathbf{z}$ to indicate whether sensor $s \in \mathcal{S}$ and actuator $a \in \mathcal{A}$ are required by VPF $f$. We consider that each VPF requires one unit of computing resource, i.e., 1 VM. This model is reasonable, among others, for smart grid control functions that share similar control structures, e.g., Volt/Var Control (VVC) [18].

We assume that a service orchestrator monitors the status of all system components and is able to coordinate the allocation of communication and computing resources. The specification of the required backhaul network and MEC APIs is outside of the scope of our work, but it could be implemented by combining the Openflow and the OpenStack APIs [19]. Figure 1 illustrates the components of the considered communication and computing infrastructure.

### A. Failure Scenarios

We consider that the communication and computing infrastructure is subject to the occasional failure of its components. We use the term failure scenario to refer to the system when a set of its components has failed, and we denote by $\mathcal{L}$ the set of all failure scenarios. A failure scenario can include a combination of communication and computing resources.

The failure of a wireless communication link, either due to equipment failure, due to jamming, or due to a denial of service attack, results in the failure of the communication between a sensor or an actuator and its associated BS. To recover from the failure, the sensor or actuator needs to be re-associated to another BS. We consider that BS association is taken care of by the mobile network, and we denote by $b_{l,i}$ the associated BS for a sensor or actuator $i \in \mathcal{A} \cup \mathcal{S}$ in scenario $l$. Similarly, the failure of a component of the backhaul network (e.g., an SDN switch or an optical cable) is handled by the mobile network, but it could result in increased delay.

The failure of MEC nodes, due to a communication failure, hardware failure or a DoS attack, results in a MEC node to be unsuitable for VPF placement. We use the binary variable $r_{l,m}$ to indicate the suitability of MEC node $m$ for VPF placement in scenario $l$.

We assume that the system operator is able to estimate the occurrence probability of each failure scenario, and we denote the estimated occurrence probability of scenario $l$ by $\pi_l$. The occurrence probabilities of failure scenarios can be estimated by using the mean time between failures and the mean time to repair, which can be obtained by the historical failure and maintenance records of the components of the system [20]. By definition $\sum_{l\in\mathcal{L}} \pi_l = 1$. Note that this model allows to capture correlated failures, and is thus able to capture various link layer, network layer and cloud failure recovery mechanisms.

## B. Cost Model

Our cost model for VPF placement accounts for costs in terms of computing and storage resources and in terms of the data transmission between the MEC nodes and the sensors and actuators. We denote by $F_m$ the availability cost of MEC node $m$, which has to be paid if the node is to be available for VPF placement in any scenario. The availability cost $F_m$ is justified by the cost of storing the virtual machine images in the MEC nodes (storage cost) and by the potential need for reserving computational power and memory for the eventual execution of the VPFs (availability fee). We denote by $p_{m,f}$ the placement cost of an instance of VPF $f$ on node $m$. This cost is justified by the computational and memory resources needed for the execution of the VPF.

The cost of data transmission consists of the wireless transmission cost and the backhaul network transmission cost. We denote by $c_{i,b}$ the cost of wireless data transmission between a sensor or an actuator $i \in \mathcal{S} \cup \mathcal{A}$ and a BS $b$. In general it is reasonable to assume that $c_{i,b}$ is inverse proportional to the achievable rate; this cost model is well suited for capturing the delay introduced by data transmission and the usage of radio resource blocks.

We denote by $\bar{c}_{l,b,b'}$ the data transmission cost over the backhaul network between BSs $b$ and $b'$ in scenario $l$. Note that in different failure scenarios the routing between BSs $b$ and $b'$ may be different, thus $\bar{c}_{l,b,b'}$ is scenario dependent.

The total data transmission cost $c_{i,b,m}$ between a sensor or an actuator $i \in \mathcal{A} \cup \mathcal{S}$ and a MEC node $m \in \mathcal{M}$ if the sensor or actuator is associated to BS $b \in \mathcal{B}$ is then the sum of the wireless transmission cost and the backhaul network data transmission cost, $c_{l,m,i} = c_{i,b} + \bar{c}_{l,b,m}$.

## C. Problem Formulation

We are now ready to formulate the resilient VPF placement problem, which consists of deciding which MEC nodes to keep available and on which MEC node to place each VPF, subject to resilience and MEC resource capacity constraints. We use the decision variable $v_m \in \{0,1\}$ to denote whether MEC node $m$ is kept available, and let $\mathbf{v} = \{v_1, \ldots, v_m\}$. Furthermore, we use the decision variable $x_{l,f,m} \in \{0,1\}$ to denote whether VPF $f$ is placed in MEC node $m$ in scenario $l$, and let $\mathbf{x} = \{x_{1,1,1}, \ldots, x_{l,f,m}\}$.

VPF placement is subject to resilience and capacity constraints. For resilience we require that at each VPF $f$ should be placed at a MEC node in each scenario $l \in \mathcal{L}$,

$$\sum_{m\in\mathcal{M}} x_{l,f,m} \geq 1,\ \forall l \in \mathcal{L},\ \forall f \in \mathcal{F}. \tag{1}$$

Furthermore, we consider two MEC resource capacity constraints. First, a VPF $f$ can only be placed on a MEC node that is made available and is accessible in failure scenario $l$,

$$x_{l,f,m} \leq r_{l,m} v_m,\ \forall l \in \mathcal{L},\ \forall f \in \mathcal{F},\ \forall m \in \mathcal{M}. \tag{2}$$

Second, the sum of the computing resource requirements of the VPFs placed on MEC node $m$ cannot exceed its computing resource capacity,

$$\sum_{f\in\mathcal{F}} x_{l,f,m} \leq \omega_m v_m,\ \forall l \in \mathcal{L},\ \forall m \in \mathcal{M}. \tag{3}$$

The VPF placement problem is then to minimize the VPF placement cost subject to resilience and capacity constraints,

$$\underset{\mathbf{x},\mathbf{v}}{\text{minimize}}$$

$$O(\mathbf{v},\mathbf{x}) = \underbrace{\sum_{m\in\mathcal{M}} v_m F_m}_{\text{Availability cost}} + \sum_{l\in\mathcal{L}} \pi_l \sum_{m\in\mathcal{M}} \Bigg( \underbrace{\sum_{f\in\mathcal{F}} x_{l,f,m} p_{m,f}}_{\text{Execution Cost}} +$$

$$\underbrace{\sum_{s\in\mathcal{S}} \Bigg( c_{l,m,s} \sum_{f\in\mathcal{F}} y_{f,s} x_{l,f,m} \Bigg) + \sum_{a\in\mathcal{A}} \Bigg( c_{l,m,a} \sum_{f\in\mathcal{F}} z_{f,a} x_{l,f,m} \Bigg)}_{\text{Data transmission cost}} \Bigg) \tag{4}$$

$$\begin{aligned} \text{s.t.} \quad & (1) - (3) \\ & x_{l,f,m} \in \{0,1\},\ \forall l \in \mathcal{L},\ \forall f \in \mathcal{F},\ \forall m \in \mathcal{M} \end{aligned} \tag{5}$$

Our focus is on instances when the VPF placement problem (4) is feasible, that is, there exists at least one feasible VPF placement $\mathbf{x}$ when all MEC nodes are kept available.

## IV. Resilient VPF Placement Algorithm

Observe that the resilient VPF placement problem (4) is an IP problem. IP problems are in general computationally

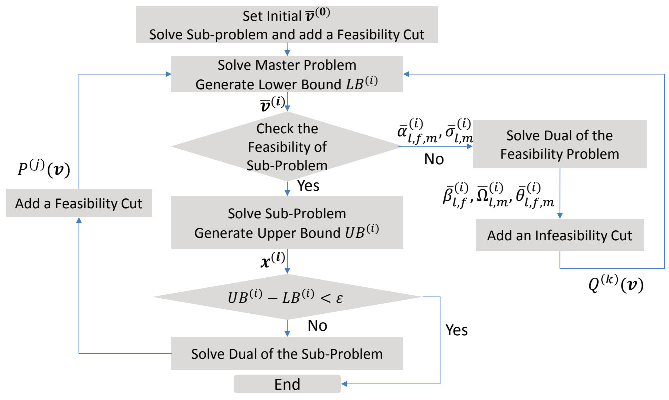
Set Initial $\bar{\boldsymbol{v}}^{(\mathbf{0})}$
Solve Sub-problem and add a Feasibility Cut
Solve Master Problem
Generate Lower Bound $LB^{(i)}$
$\bar{\boldsymbol{v}}^{(i)}$
Check the Feasibility of Sub-Problem
$\bar{\alpha}^{(i)}_{l,f,m}, \bar{\sigma}^{(i)}_{l,m}$
No
Solve Dual of the Feasibility Problem
$\bar{\beta}^{(i)}_{l,f}, \bar{\Omega}^{(i)}_{l,m}, \bar{\theta}^{(i)}_{l,f,m}$
Add an Infeasibility Cut
$Q^{(k)}(\boldsymbol{v})$
$P^{(j)}(\boldsymbol{v})$
Add a Feasibility Cut
Yes
Solve Sub-Problem
Generate Upper Bound $UB^{(i)}$
$\boldsymbol{x}^{(i)}$
$UB^{(i)} - LB^{(i)} < \varepsilon$
No
Yes
Solve Dual of the Sub-Problem
End

Figure 2. Flowchart of Resilient VPF Placement Algorithm

hard, but we can observe that the constraint matrix in (4) has a special block structure, known as block-ladder structure. Constraint matrices with a block-ladder structure typically arise in stochastic programming problems, for which an efficient solution method is the Generalized Benders Decomposition (GBD) [21]. In what follows we leverage this insight to propose a solution to the resilient VPF problem. The overall structure of the proposed RVP algorithm is shown in Figure 2.

Following the idea of GBD, RVP partitions the variables of the optimization problem into two sets, and decomposes the original problem into a master problem and a sub-problem. The master problem and the sub-problem are solved iteratively over different partitions, and in each iteration they generate an upper bound and a lower bound of the original problem's objective value, respectively. The iteration stops when the upper bound and the lower bound match the termination condition. In what follows we formulate the master problem, the sub-problem, introduce the feasibility check and discuss how to initialize the algorithm. Finally, we prove that the VPF algorithm terminates in a finite number of iterations.

Throughout the section we use $\bar{\mathbf{v}}^{(i)} = \{\bar{v}_1^{(i)}, \ldots, \bar{v}_m^{(i)}\}$ and $\bar{\mathbf{x}}^{(i)} = \{\bar{x}_{1,1,1}^{(i)}, \ldots, \bar{x}_{l,f,m}^{(i)}\}$ to denote the optimal solution for the master problem and for the sub-problem in iteration $i$, respectively.

## A. Master Problem: Availability Cost Minimization

The purpose of the master problem is to choose the MEC nodes to be made available. In iteration $i$, we formulate the master problem by fixing the variable $\mathbf{x} = \bar{\mathbf{x}}^{(i-1)}$ and optimizing the expected total cost over $\mathbf{v}$,

$$\begin{aligned} \min_{\mathbf{v}} \quad & z && (6)\\ \text{s.t.} \quad & z \geq P^{(j)}(\mathbf{v}),\ j = 1, \ldots, J && (7)\\ & 0 \geq Q^{(k)}(\mathbf{v}),\ k = 1, \ldots, K && (8)\\ & v_m \in \{0,1\},\ \forall m \in \mathcal{M} && (9) \end{aligned}$$

The constraints (7) are so called feasibility cuts, $P^{(j)}(\mathbf{v})$ is a support function obtained in a previous iteration by solving the dual of the sub-problem, which we will introduce in section IV-C. Note that $P^{(j)}(\mathbf{v})$ provides a lower bound on the total cost, and is a function of $\mathbf{v}$. The constraints (8) are so called infeasibility cuts. They are used to form a region of $\mathcal{V}$ that satisfies constraints (2)-(3) of the VPF placement problem, and will be introduced in Section IV-B. In each iteration the algorithm adds either a feasibility cut or an infeasibility cut, hence $J + K = i$.

In what follows we denote by $LB^{(i)} = \bar{z}^{(i)}$ the optimal objective function value of the master problem obtained in iteration $i$. Observe that the master problem is an IP in $|\mathcal{M}|$ binary variables.

## B. Feasibility Check of Available MEC Nodes

As a next step, before we proceed with placing the VPFs to the available MEC nodes, we need to check if the available MEC nodes meet the capacity requirements (2) and (3). We do this by formulating the feasibility problem,

$$\begin{aligned} \min_{\boldsymbol{\alpha},\, \boldsymbol{\sigma},\, \mathbf{x}} \quad & \sum_{l\in\mathcal{L}} \sum_{f\in\mathcal{F}} \sum_{m\in\mathcal{M}} \alpha_{l,f,m} + \sum_{l\in\mathcal{L}} \sum_{m\in\mathcal{M}} \sigma_{l,m} && (10)\\ \text{s.t.} \quad & (1) \\ & x_{l,f,m} \leq r_{l,m}\bar{v}_m^{(i)} + \alpha_{l,f,m},\ \forall l,\ \forall f,\ \forall m && (11)\\ & \sum_{f\in\mathcal{F}} x_{l,f,m} \leq \omega_m \bar{v}_m^{(i)} + \sigma_{l,m},\ \forall l,\ \forall m && (12)\\ & x_{l,f,m} \in [0,1],\ \forall l,\ \forall f,\ \forall m && (13)\\ & \alpha_{l,f,m}, \sigma_{l,m} \geq 0,\ \forall l,\ \forall f,\ \forall m && (14) \end{aligned}$$

Note that variables $\boldsymbol{\alpha} = \{\alpha_{1,1,1}, \ldots, \alpha_{l,f,m}\}$ and $\boldsymbol{\sigma} = \{\sigma_{1,1,1}, \ldots, \sigma_{l,m}\}$ are added to the constraints involving $\bar{v}_m^{(i)}$ to ensure that the constraints (2) and (3) can be satisfied, and hence the problem has an optimal solution. Notice that the feasibility check does not require $x_{l,f,m}$ to be binary; as we will show in Lemma 1 linear relaxation does not change the solution set (and hence it does not change feasibility either).

Let us denote by $\bar{\alpha}^{(i)} = \{\bar{\alpha}_{1,1,1}^{(i)}, \ldots, \bar{\alpha}_{l,f,m}^{(i)}\}$ and $\bar{\sigma}^{(i)} = \{\bar{\sigma}_{1,1}^{(i)}, \ldots, \bar{\sigma}_{l,m}^{(i)}\}$ the optimal solution of the feasibility problem. If the optimal value of the objective function (10) is zero then the algorithm continues with the sub-problem described in Section IV-C.

Otherwise, we continue with formulating the dual of the feasibility problem,

$$\begin{aligned} \min_{\boldsymbol{\beta},\, \boldsymbol{\theta},\, \boldsymbol{\Omega}} \quad & \sum_{l\in\mathcal{L}} \sum_{f\in\mathcal{F}} \beta_{l,f} - \sum_{l\in\mathcal{L}} \sum_{m\in\mathcal{M}} \Omega_{l,m}\omega_m \bar{v}_m^{(i)} \\ & - \sum_{l\in\mathcal{L}} \sum_{f\in\mathcal{F}} \sum_{m\in\mathcal{M}} \theta_{l,f,m}\bar{v}_m^{(i)} r_{l,m} && (15)\\ \text{s.t.} \quad & \Omega_{l,m} + \theta_{l,f,m} - \beta_{l,f} \geq 0,\ \forall l,\ \forall f,\ \forall m && (16)\\ & 1 - \theta_{l,f,m} \geq 0,\ \forall l,\ \forall,\ \forall m && (17)\\ & 1 - \Omega_{l,m} \geq 0,\ \forall l,\ \forall m && (18) \end{aligned}$$

Let us denote by $\bar{\beta}^{(i)} = \{\bar{\beta}_{1,1}^{(i)}, \ldots, \bar{\beta}_{l,f}^{(i)}\}$, $\bar{\theta}^{(i)} = \{\bar{\theta}_{1,1,1}^{(i)}, \ldots, \bar{\theta}_{l,f,m}^{(i)}\}$ and $\bar{\mathbf{\Omega}}^{(i)} = \{\bar{\Omega}_{1,1}^{(i)}, \ldots, \bar{\Omega}_{l,m}^{(i)}\}$ the optimal multipliers of the dual of the feasibility problem. Based on the solution above, we construct an infeasibility cut

$$Q^{(k)}(\mathbf{v})=\sum_{l\in\mathcal{L}}\sum_{f\in\mathcal{F}}\bar{\beta}_{l,f}^{(i)}\left(1-\sum_{m\in\mathcal{M}}\bar{x}_{l,f,m}^{(i)}\right)$$
$$+\sum_{l\in\mathcal{L}}\sum_{m\in\mathcal{M}}\bar{\Omega}_{l,m}^{(i)}\left(\sum_{f\in\mathcal{F}}\bar{x}_{l,f,m}^{(i)}-\omega_m v_m\right)$$
$$+\sum_{l\in\mathcal{L}}\sum_{f\in\mathcal{F}}\sum_{m\in\mathcal{M}}\bar{\theta}_{l,f,m}^{(i)}\left(\bar{x}_{l,f,m}^{(i)}-r_{l,m}v_m\right), \quad (19)$$

which we add to the master problem (Chapter 6.3.5, [22]). After adding the infeasibility cut, the algorithm continues with the master problem. Observe that the infeasibility cut tightens the master problem, and ensures the feasibility of the sub-problem in the next iteration.

## C. Sub-problem: Placement cost minimization

While the master problem chooses the set of MEC nodes that are made available to minimize the total cost, the sub-problem minimizes the placement cost and the data transmission cost over $\mathbf{x}$ for $\bar{\mathbf{v}}^{(i)}$ computed by the master problem,

$$\min_{\mathbf{x}}\sum_{l\in\mathcal{L}}\pi_l\sum_{m\in\mathcal{M}}\left(\sum_{s\in\mathcal{S}}\left(c_{l,m,s}\sum_{f\in\mathcal{F}}y_{f,s}x_{l,f,m}\right)+\right.$$
$$\left.\sum_{a\in\mathcal{A}}\left(c_{l,m,a}\sum_{f\in\mathcal{F}}z_{f,a}x_{l,f,m}\right)+\sum_{f\in\mathcal{F}}x_{l,f,m}p_{m,f}\right) \quad (20)$$

$$\text{s.t.} \qquad (1)-(3), \text{ and } (5)$$

To avoid solving the sub-problem as an IP we use linear relaxation and replace constraint (5) by

$$x_{l,f,m}\in[0,1],\ \forall l\in\mathcal{L},\ \forall f\in\mathcal{F},\ \forall m\in\mathcal{M}. \quad (21)$$

As we show next, linear relaxation does not affect the result of the algorithm, while reducing its computational complexity.

**Lemma 1.** *The coefficient matrix of constraints* (1)-(3) *is totally unimodular. Furthermore, the linear relaxation of the sub-problem has integral solutions, which are optimal for the sub-problem.*

*Proof.* First we prove that the coefficient matrix of constraints (1)-(3) is totally unimodular. We write constraints (1)-(3) according to the failure scenarios in matrix form,

$$R_l\mathbf{x}_l\le-\mathbf{1}, \quad (22)$$
$$I\mathbf{x}_l\le\mathbf{s}_l, \quad (23)$$
$$T_l\mathbf{x}_l\le\mathbf{t}_l, \quad (24)$$

where

$$R_{l,f,m}=\begin{cases}-1, & \text{if } (f-1)|\mathcal{F}|+1\le m\le f|\mathcal{F}|\\ 0, & \text{otherwise}\end{cases} \quad (25)$$
$$\mathbf{x}_l=[x_{l,1,1},\dots,x_{l,1,m},\dots,x_{l,f,m}]^T, \quad (26)$$
$$\mathbf{s}_{l,f}=[r_{l,1}v_1,\dots,r_{l,m}v_m]\forall f\in\mathcal{F}, \quad (27)$$
$$\mathbf{s}_l=[\underbrace{\mathbf{s}_{l,1}\dots\mathbf{s}_{l,|\mathcal{F}|}}_{|\mathcal{F}|}]^T, \quad (28)$$
$$T_l=[\underbrace{I_{m,m}\dots I_{m,m}}_{|\mathcal{F}|}], \quad (29)$$
$$\mathbf{t}_l=[\omega_1v_1,\dots,\omega_mv_m]^T. \quad (30)$$

We can now sort the coefficient matrix of constraints (1)-(3) according to failure scenarios, which provides us the following block-ladder structure

$$G=\begin{bmatrix}G_1 & \mathbf{0} & \dots & \mathbf{0}\\ \mathbf{0} & G_2 & \dots & \mathbf{0}\\ \vdots & \vdots & \ddots & \vdots\\ \mathbf{0} & \mathbf{0} & \dots & G_{|\mathcal{L}|}\end{bmatrix}, \quad (31)$$

where $G_l$ is the constraint matrix of scenario $l$ and

$$G_l=\begin{bmatrix}G'_l\\ \mathbf{I}\end{bmatrix}, \text{ and } G'_l=\begin{bmatrix}R_l\\ T_l\end{bmatrix} \quad (32)$$

For convenience, let us define

$$\bar{G}'_l=\begin{bmatrix}-R_l\\ T_l\end{bmatrix}. \quad (33)$$

According to the matrix formulation above, each column of $-R_l$ contains only one non-zero element with value 1, and each column of $T_l$ contains only one non-zero element with value 1. Thus matrix $\bar{G}'_l$ has two non-zero elements in each column. To show that $\bar{G}'_l$ is totally unimodular, it suffices to show that the rows of $\bar{G}'_l$ can be partitioned into two disjoint sets such that if there are two non-zero entries in a column of $\bar{G}'_l$ then the two rows where the two entries are should be in different sets of rows (Theorem 2.3.3 in [23]). This clearly holds, if one set of the rows of $\bar{G}'_l$ consists of the rows of matrix $-R_l$ and the other set of rows of $\bar{G}'_l$ consists of the rows of matrix $T_l$.

Now, since multiplying some rows of a totally unimodular matrix by $-1$ preserves total unimodularity, $G'_1$ is also a totally unimodular matrix. Following Theorem 19.3 in [24], $G_l$ and $G$ are totally unimodular matrices too.

Finally, since each element of $\mathbf{s}_l$ and $\mathbf{t}_l$ is an integer and the constraint coefficient matrix $G$ is totally unimodular, according to Corrollary 19.2a in [24], the sub-problem (20) with linear relaxation has integral optimal solutions, which are optimal for the original sub-problem (20). □

Given the optimal solution $\bar{\mathbf{x}}^{(i)}=\{\bar{x}_{1,1,1}^{(i)},\dots,\bar{x}_{l,f,m}^{(i)}\}$ of the sub-problem, we can express the upper bound (UB) of the total cost in iteration $i$ as

$$\text{UB}^{(i)}=\sum_{m\in\mathcal{M}}\bar{v}_m^{(i)}F_m+\sum_{l\in\mathcal{L}}\pi_l\sum_{m\in\mathcal{M}}\left(\sum_{s\in\mathcal{S}}\left(c_{l,m,s}\sum_{f\in\mathcal{F}}y_{f,s}\bar{x}_{l,f,m}^{(i)}\right)\right.$$
$$\left.+\sum_{a\in\mathcal{A}}\left(c_{l,m,a}\sum_{f\in\mathcal{F}}z_{f,a}\bar{x}_{l,f,m}^{(i)}\right)+\sum_{f\in\mathcal{F}}\bar{x}_{l,f,m}^{(i)}p_{m,f}\right) \quad (34)$$

The algorithm terminates here if $\mathrm{UB}^{(i)} - \mathrm{LB}^{(i)} < \epsilon$, where $\epsilon > 0$ is the termination threshold.

Otherwise the algorithm continues with adding one more constraint to the master problem. We obtain the constraint by formulating the dual of the sub-problem,

$$\max_{\boldsymbol{\lambda},\boldsymbol{\gamma},\boldsymbol{\mu}} \quad \sum_{l\in\mathcal{L}}\sum_{f\in\mathcal{F}} \lambda_{l,f} - \sum_{l\in\mathcal{L}}\sum_{m\in\mathcal{M}} \gamma_{l,m}\omega_m \tag{35}$$
$$- \sum_{l\in\mathcal{L}}\sum_{f\in\mathcal{F}} \left( \sum_{m\in\mathcal{M}} \mu_{l,f,m} r_{l,m} \bar{v}_m^{(i)} \right)$$
$$\text{s.t.} \quad \pi_l \left( c_{l,m,s} \sum_{s\in\mathcal{S}} y_{f,s} + c_{l,m,a} \sum_{a\in\mathcal{A}} z_{f,a} + p_{m,f} \right) \geq$$
$$\lambda_{l,f} - \mu_{l,f,m} - \gamma_{l,m}, \ \forall l \in \mathcal{L}, \forall f \in \mathcal{F}, \ m \in \mathcal{M} \tag{36}$$
$$\lambda_{l,f}, \mu_{l,f,m}, \gamma_{l,m} \geq 0, \ \forall l \in \mathcal{L}, \forall f \in \mathcal{F}, \ m \in \mathcal{M} \tag{37}$$

where multiplier $\lambda_{l,f}$ is the marginal revenue for a placed VPF $f$ in scenario $l$, $\gamma_{l,m}$ is the penalty for violation of the capacity of MEC node $m$ in scenario $l$, and $\mu_{l,f,m}$ is the penalty for each violated suitability constraint in scenario $l$ for node $m$. Let us denote the optimal multipliers obtained by solving the dual in iteration $i$ by $\bar{\boldsymbol{\lambda}}^{(i)} = \{\bar{\lambda}_{1,1}^{(i)}, \ldots, \bar{\lambda}_{l,f}^{(i)}\}$, $\bar{\boldsymbol{\gamma}} = \{\bar{\gamma}_{1,1}^{(i)}, \ldots, \bar{\gamma}_{l,m}^{(i)}\}$, and $\bar{\boldsymbol{\mu}} = \{\bar{\mu}_{1,1}^{(i)}, \ldots, \bar{\mu}_{l,f,m}^{(i)}\}$.

Based on the solution of the sub-problem and the dual of the sub-problem, we build the support function $P^{(j)}(\mathbf{v})$. In GBD the support function can be built in various ways. Since the objective function and the constraints of the VPF placement problem (4) are linearly separable in $\mathbf{v}$ and $\mathbf{x}$, we build the support function based on the Lagrange function (Chapter 6.3.5, [22]),

$$P^{(j)}(\mathbf{v}) = O(\mathbf{v}, \bar{\mathbf{x}}^{(i)}) + \sum_{l\in\mathcal{L}}\sum_{f\in\mathcal{F}} \bar{\lambda}_{l,f}^{(i)} \left( 1 - \sum_{m\in\mathcal{M}} \bar{x}_{l,f,m}^{(i)} \right)$$
$$+ \sum_{l\in\mathcal{L}}\sum_{m\in\mathcal{M}} \bar{\gamma}_{l,m}^{(i)} \left( \sum_{f\in\mathcal{F}} \bar{x}_{l,f,m}^{(i)} - \omega_m v_m \right)$$
$$+ \sum_{l\in\mathcal{L}}\sum_{f\in\mathcal{F}}\sum_{m\in\mathcal{M}} \bar{\mu}_{l,f,m}^{(i)} \left( \bar{x}_{l,f,m}^{(i)} - r_{l,m} v_m \right). \tag{38}$$

Finally, we add the support function $P^{(j)}(\mathbf{v})$ as a feasibility cut to the master problem in the next iteration.

**Remark 1.** *Observe that since the VPF placement in different failure scenarios is independent, one could formulate one sub-problem for each failure scenario instead of a single sub-problem, and then generate the feasibility cut by combining the optimal multipliers. By doing so, the sub-problem* (20) *is divided into* $|\mathcal{L}|$ *sub-problems with smaller dimensions, essentially allows to scale the RVP algorithm for an arbitrary number of failure scenarios. The same remark applies to the feasibility check. We choose to formulate a single feasibility check and a single sub-problem for conciseness.*

**Remark 2.** *Observe that through linear relaxation of the sub-problem RVP reduces the number of integer decision variables from* $|\mathcal{M}|(|\mathcal{L}||\mathcal{F}|+1)$ *to* $|\mathcal{M}|$*. As we will see this allows us to solve up to moderate instances of the VPF problem.*

## *D. Initializing RVP*

The algorithm is initialized with $\bar{\mathbf{v}}^{(0)} = \mathbf{e}$, i.e., all MEC nodes made available. Since $\bar{\mathbf{v}}^{(0)}$ is assumed to be feasible, the algorithm can start with solving the sub-problem and its dual to obtain $\mathrm{UB}^{(0)}$ and the feasibility cut $P^{(1)}(\mathbf{v})$, which can be added to the master problem.

## *E. Convergence of the RVP Algorithm*

Since RVP is iterative, a fundamental question is whether it is guaranteed to terminate. The following result shows that this is the case.

**Theorem 2.** *The RVP algorithm terminates in a finite number of iterations for any* $\epsilon > 0$.

*Proof.* According to Theorem 6.4.3 in [22], the following are the sufficient conditions for the RVP algorithm to terminate in a finite number of iterations:

1) The domain of $\mathbf{x}$ should be a nonempty, convex set.
2) The objective function $O(\mathbf{v}, \mathbf{x})$, and constraints (1)-(3) are convex for each fixed $\mathbf{v}$. The constraints are convex real functions on the domain of $\mathbf{x}$ for each fixed $\mathbf{v}$.
3) $\mathbf{x}$ is bounded and closed, and the constraint functions are continuous on the domain of $\mathbf{x}$ for each fixed $\mathbf{v}$.
4) The original problem with linear relaxation has a finite solution.
5) The original problem with linear relaxation has optimal multipliers for the constraints.

Condition 1) holds due to linear relaxation, which as we showed, provides the optimal integral solution to the subproblem (Lemma 1). Due to $O(\mathbf{v}, \mathbf{v})$, constraints (1)-(3) are linear functions, they are convex, and continuous in $\mathbf{x}$ for each fixed $\mathbf{v}$, therefore condition 2) holds. According to the problem formulation the domain of $\mathbf{x}$ is bounded, together with the linearities of the constraints (1)-(3), condition 3) is true. Since all the parameters are finite and the domains of $\mathbf{x}$ and $\mathbf{v}$ are bounded, condition 4) holds. Since we assumed there is at least one feasible solution $(\mathbf{v}, \mathbf{x})$, thus the first sub-problem is feasible. As a result, there exist optimal multipliers for the constraints (1)-(3).

Thus, the RVP algorithm satisfies conditions 1) to 5), and according to Theorem 6.4.3 in [22] it terminates in a finite number of iterations for any $\epsilon > 0$. □

## V. Numerical Results

In what follows we evaluate the RVP algorithm with respect to operational cost, efficiency, and scalability. For the evaluation we simulated a $50km \times 50km$ metropolitan area, based on a map of the city of Milan, Italy, including locations of the BSs (http://opencellid.org/), and used the k-means algorithm for selecting the subset of MEC nodes. Sensors and actuators were uniformly placed in the simulation area. Each sensor and actuator sends and receives data through the nearest BSs. Each

Table I
SYSTEM PARAMETERS

| Parameter | Value |
|---|---|
| Number of BSs | $\|\mathcal{B}\| = 15$ |
| Number of MEC nodes | $\|\mathcal{M}\| = 15$ |
| Number of Sensors | $\|S\| = 100$ |
| Number of VPFs | $\|F\| = 1, 2, \ldots, 30$ |
| Termination Threshold | $\epsilon = 2\%$ of UB |

VPF is associated with an actuator and with 5 sensors chosen at random. The probability that sensor $s$ is associated with $f$ is inversely proportional to the square of the distance between the sensor $s$ and the actuator $a$ associated to $f$,

$$P\left(y_{f,s} = 1 \mid z_{f,a} = 1\right) = \frac{1/d_{a,s}^2}{\sum_{s' \in \mathcal{S}} 1/d_{a,s'}^2}. \tag{39}$$

We model the wireless transmission cost $c_{i,b}$ between a sensor or an actuator $i \in \mathcal{A} \cup \mathcal{S}$ and BS $b \in \mathcal{B}$ to be inversely proportional to the unit capacity of the link given by its Shannon capacity,

$$c_{i,b} = \frac{n}{\log_2\left(1 + \frac{p_0 d_{i,b}^{-h}}{N_0}\right)},$$

where $n$ is a positive cost coefficient, $p_0$ is the fixed transmit power of the transmitter, $d_{i,b}$ is the distance between $i$ and $b$, $h$ is the path loss factor, and $N_0$ is the Additive White Gaussian Noise (AWGN). This cost model is reasonable since a radio link with a lower unit capacity requires more radio resource (e.g., radio spectrum) for the same bitrate.

In terms of failure scenarios, we include the single node failure of each MEC node and the scenario that all the MEC nodes are suitable. The results shown are the averages of 100 simulations, the confidence intervals are at the 95% confidence level. Table I shows the system parameters.

## A. Cost Performance

As a baseline for comparison for the RVP algorithm we use a greedy algorithm, which for each scenario makes all the MEC nodes available, solves the sub-problem for each scenario to get a VPF allocation that minimizes the sub-problem cost, which is the sum of the execution cost and the data transmission cost, and it then makes available the MEC nodes that are used by any VPF in any scenario. The pseudo-code of the greedy algorithm is shown in Algorithm 1.

**Algorithm 1:** Greedy VPF Placement Algorithm

```
1 Let $\bar{\mathbf{v}}^{(0)} = \mathbf{e}$.
2 Solve the sub-problem (20).
3 Let $v_m = \min(1, \sum_{l \in \mathcal{L}} \sum_{f \in \mathcal{F}} x_{l,f,m})$
  Output: $x_{l,f,m}, v_m$
```

We start with comparing the cost performance of the RVP algorithm and the greedy algorithm with respect to the number of VPFs. Figure 3 shows the average total cost and the average second-stage cost for 15 MEC as a function of the number of VPFs. In terms of the total cost, the RVP algorithm outperforms the baseline algorithm by up to about 110 percent. It is interesting to observe that in terms of the sub-problem cost the greedy algorithm performs better than the RVP algorithm. This is because the greedy algorithm minimizes the sub-problem cost at the price of increasing the availability cost, while the RVP algorithm performs joint optimization over the availability cost and the sub-problem cost to minimize total cost. Clearly, in practice the performance gain of the RVP algorithm compared to the greedy algorithm depends on the system parameters, mainly $F_m$ and $p_{m,f}$.

Besides the total cost, we consider the average utilization ratio of capacities (AURC) as a second performance metric,

$$AURC = \frac{\mid \mathcal{F} \mid}{\sum_{l \in \mathcal{L}} \pi_l \sum_{m \in \mathcal{M}} r_{l,m} \omega_m}. \tag{40}$$

The AURC shows how well the algorithms utilize the available MEC nodes. Figure 4 shows the AURC of the available MEC nodes for the scenario shown in Figure 3. The results show that the RVP algorithm outperforms the greedy algorithm by up to 74%. As a remark, the AURC for RVP drops when we increase the number of VPFs from 3 to 4, from 6 to 7, from 10 to 11, etc. This is because in these cases, the RVP algorithm makes an additional MEC node available on average to support an additional VPF, while an additional MEC node can serve more than one VPF.

Figure 5 shows the average number of MEC nodes that are made available for the scenarios shown in Figure 3. We observe that on average the RVP algorithm makes less MEC nodes available than the greedy algorithm. Also, the RVP algorithm can accommodate more VPFs on the same number of MEC nodes, which explains the higher AURC of the RVP algorithm shown in Figure 4.

## B. Convergence and Efficiency of the RVP algorithm

Figure 6 shows the convergence of the RVP algorithm for 21 VPFs and 15 MEC nodes. The black dots and the dotted line show the upper bound and the lower bound in different iterations, respectively. The solid line shows the lowest upper bound up to a certain iteration. The absence of the upper

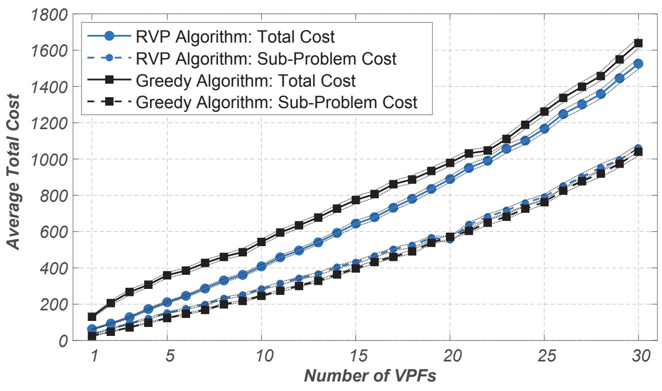

Figure 3. Total cost and sub-problem cost vs. the number of VPFs for a system of 15 MEC nodes.

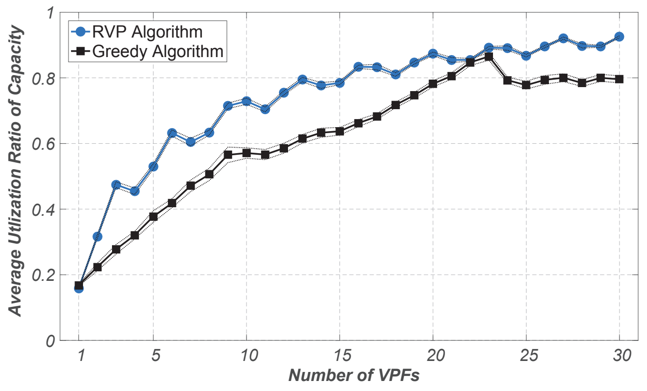

Figure 4. Average capacity utilization ratio by the RVP algorithm and by the Greedy algorithm vs. the number of VPFs for a system of 15 MEC nodes.

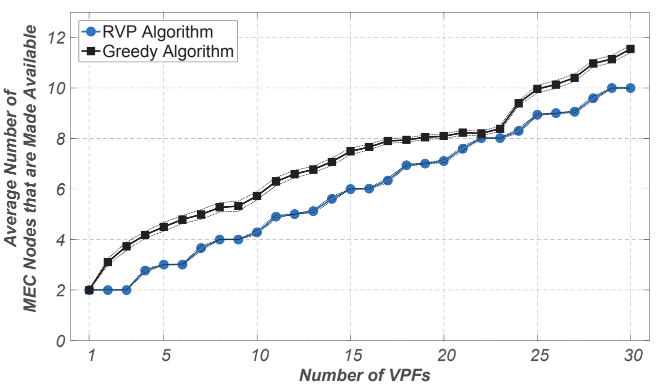

Figure 5. Average number of MEC nodes that are made available by the RVP algorithm and by the Greedy algorithm vs. the number of VPFs for a system of 15 MEC nodes.

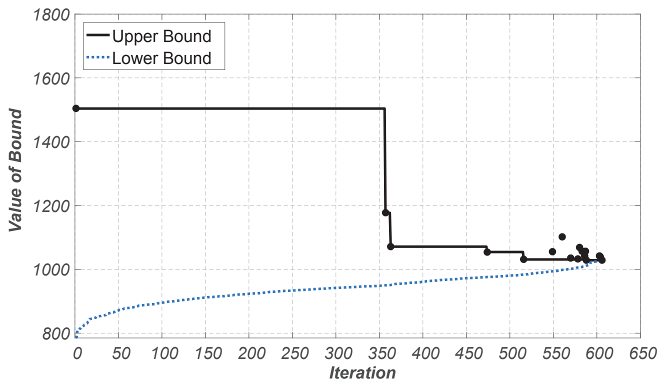

Figure 6. The convergence performance of the RVP algorithm for solving a sample VPF placement problem with 15 VPFs and 15 MEC nodes. The RVP algorithm converges at iteration 606.

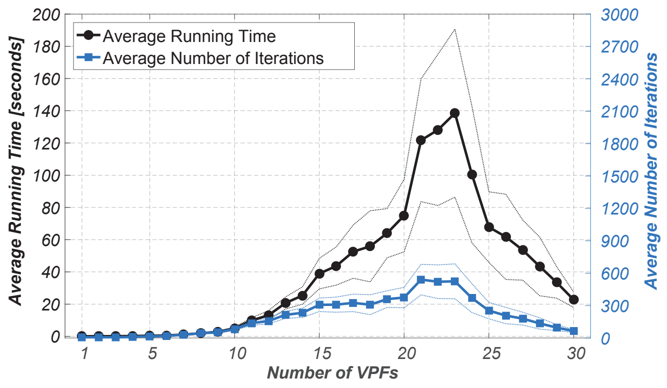

Figure 7. Average running time and average number of iterations for the RVP algorithm for 15 MEC nodes vs. number of VPFs.

bound in an iteration indicates that the algorithm encountered an infeasible set of MEC nodes made available by the master problem, and thus an infeasibility cut has been added. In the example convergence happens after 606 iterations, i.e., after considering 606 subsets of $\mathcal{M}$. As a comparison, using an exhaustive search over all subsets of MECs nodes would have to consider $2^{15} = 32768$ subsets of $\mathcal{M}$.

Figure 7 shows the average running time and the average number of iterations of RVP as a function of the numbers of VPFs for 15 MEC nodes. The total running time is determined by the running time per iteration, and the total number of iterations. The running time per iteration is affected by the dimension of the sub-problems. For example, the average number of iterations for the case of 13 and 26 VPFs are about the same (i.e., 214.2 and 207.7, respectively). However, the average running time of RVP for 26 VPFs is about three times of that for 13 VPFs. A similar reasoning explains the phenomenon that when the number of VPFs increases from 21 to 22, the corresponding average number of iterations are about the same, whereas the average running time increases. Recall that as the number of iterations increases, so does the number of feasibility and infeasibility cuts, which increases the complexity of the master problem as well. Note, however, that one could reduce the running times by formulating one feasibility check and one sub-problem per scenario and solving them in parallel, the speedup could be up to a factor of $|\mathcal{L}|$.

To further analyze the impact of the number of iterations on the running time, Figure 8 shows the distribution of the running time for 15 MEC nodes, while Figure 9 shows the distribution of the number of iterations for the same scenario. The markers on the curves in Figure 8 and 9 show the average values of the corresponding quantities. Figure 9 shows that an increase of the number of VPFs by a factor of two approximately doubles the percentile values of the number of iterations. At the same time, Figure 8 shows that the percentiles of the running time increase by more than a factor of two. The comparison of the two figures allows for a similar conclusion as in Figure 7, that is the average running time per iteration indeed increases as the number of VPFs increases.

Besides showing the impact of the number of iterations on the running time, Figure 8 and 9 show that the average running time and the average number of iterations for the RVP algorithm are higher than the corresponding median values. This is caused by a number of problem instances that take a very long time to converge. In practice, a larger convergence threshold $\epsilon$ at the price of relatively higher cost can shorten the running time.

Overall, our results show that joint optimization of the available MEC nodes and the placement of VPFs can provide

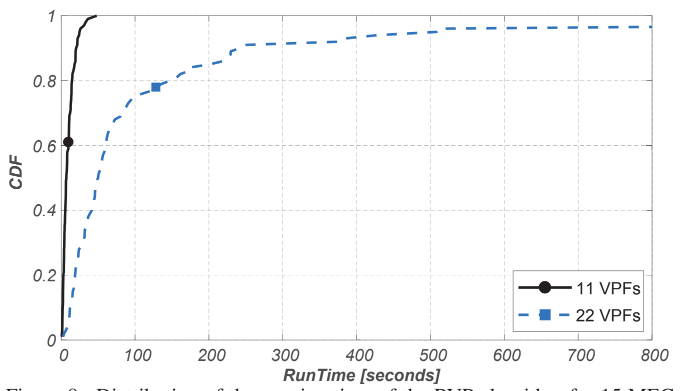

Figure 8. Distribution of the running time of the RVP algorithm for 15 MEC nodes, and 11 and 22 VPFs.

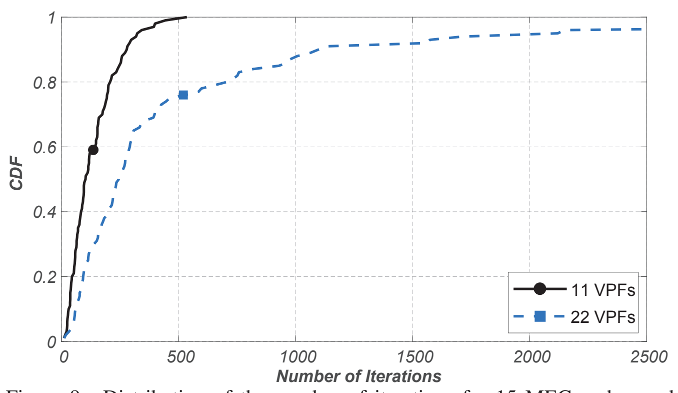

Figure 9. Distribution of the number of iterations for 15 MEC nodes, and 11 and 22 VPFs.

a significant cost reduction compared to a greedy allocation, and the proposed RVP algorithm is a computationally efficient solution for moderate instances of the problem, especially if one sub-problem is implemented and executed per scenario.

## VI. Conclusion

We have proposed a framework and an algorithm for solving problem of the resilient placement of virtual process control functions with the objective to minimize the computing and communication cost subject to capacity and resilience constraints. The iterative algorithm we proposed makes use of the special structure of the optimization problem, and together with a linear relaxation, the Benders decomposition based approach effectively reduces the search space. Extensive numerical results show that the proposed algorithm reduces the total cost significantly compared to a greedy baseline algorithm. Furthermore, the numerical results show that the proposed algorithm can scale to larger problem instances, independent of the number of resilience scenarios, if the feasibility and infeasibility cuts are computed in parallel.

Our work is a first step towards the study of resilient process control function placement in 5G mobile networks. Interesting extensions of our work include improving the proposed algorithm through different Benders cuts, and considering classes of virtual process functions, requiring that at least one function of each class be placed in each resilience scenario.

## References

[1] S. A. Boyer, *SCADA: supervisory control and data acquisition*. International Society of Automation, 2009.

[2] G. Di Orio, J. Barata, C. Sousa, and L. Flores, "Control system software design methodology for automotive industry," in *Proc. of IEEE International Conference on Systems, Man, and Cybernetics*, 2013, pp. 3848–3853.

[3] I. S. Acharyya and A. Al-Anbuky, "Towards wireless sensor network softwarization," in *Proc. of IEEE Conference on Network Softwarization*, 2016, pp. 378–383.

[4] 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.

[5] Q. C. Li, H. Niu, A. T. Papathanassiou, and G. Wu, "5G network capacity: key elements and technologies," *IEEE Vehicular Technology Magazine*, vol. 9, no. 1, pp. 71–78, 2014.

[6] M. Yigit, V. C. Gungor, and S. Baktir, "Cloud computing for smart grid applications," *Computer Networks*, vol. 70, pp. 312–329, 2014.

[7] G. Dán, R. B. Bobba, G. Gross, and R. H. Campbell, "Cloud computing for the power grid: from service composition to assured clouds," in *Proc. of Usenix Workshop on Hot Topics in Cloud Computing (HotCloud)*, 2013.

[8] E. Emil, G. Dán, and F. Viktoria, "Radio and computational resource management for fog computing enabled wireless camera netoworks," in *Proc. of IEEE GlobeCom Workshop on Internet of Everything*, 2016, pp. 1–6.

[9] S. Das, S. Misra, M. Khatua, and J. J. Rodrigues, "Mapping of sensor nodes with servers in a mobile health-cloud environment," in *Proc. of IEEE International Conference on e-Health Networking, Applications & Services (Healthcom)*, 2013, pp. 481–485.

[10] R. Cohen, L. Lewin-Eytan, J. S. Naor, and D. Raz, "Near optimal placement of virtual network functions," in *Proc. of IEEE INFOCOM*, 2015, pp. 1346–1354.

[11] M. Ghaznavi, A. Khan, N. Shahriar, K. Alsubhi, R. Ahmed, and R. Boutaba, "Elastic virtual network function placement," in *Proc. of IEEE CloudNet*, 2015, pp. 255–260.

[12] H. Moens and F. De Turck, "VNF-P: A model for efficient placement of virtualized network functions," in *Proc. of IEEE International Conference on Network and Service Management (CNSM)*, 2014, pp. 418–423.

[13] M. Xia, M. Shirazipour, Y. Zhang, H. Green, and A. Takacs, "Network function placement for NFV chaining in packet/optical datacenters," *Journal of Lightwave Technology*, vol. 33, no. 8, pp. 1565–1570, 2015.

[14] T.-W. Kuo, B.-H. Liou, K. C.-J. Lin, and M.-J. Tsai, "Deploying chains of virtual network functions: on the relation between link and server usage," in *Proc. of IEEE INFOCOM*, 2016, pp. 1–9.

[15] A. Hmaity, M. Savi, F. Musumeci, M. Tornatore, and A. Pattavina, "Virtual network function placement for resilient service chain provisioning," in *Proc. of IEEE International Workshop on Resilient Networks Design and Modeling (RNDM)*, 2016, pp. 245–252.

[16] P. Garcia-Herreros, J. M. Wassick, and I. E. Grossmann, "Design of resilient supply chains with risk of facility disruptions," *Industrial & Engineering Chemistry Research*, vol. 53, no. 44, pp. 17 240–17 251, 2014.

[17] M. T. Beck, M. Werner, S. Feld, and S. Schimper, "Mobile edge computing: A taxonomy," in *Proc. of the Sixth International Conference on Advances in Future Internet*, 2014, pp. 48–54.

[18] R. Uluski, "VVC in the smart grid era," in *IEEE PES General Meeting*, 2010, pp. 1–7.

[19] H. Li, G. Shou, Y. Hu, and Z. Guo, "Mobile edge computing: progress and challenges," in *Proc. of IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud)*, 2016, pp. 83–84.

[20] Q. Zhang, M. F. Zhani, M. Jabri, and R. Boutaba, "Venice: Reliable virtual data center embedding in clouds," in *Proc. of IEEE INFOCOM*, 2014, pp. 289–297.

[21] A. M. Geoffrion, "Generalized benders decomposition," *Journal of optimization theory and applications*, vol. 10, no. 4, pp. 237–260, 1972.

[22] C. A. Floudas, *Nonlinear and mixed-integer optimization: fundamentals and applications*. Oxford University Press, 1995.

[23] R. A. Brualdi and H. J. Ryser, *Combinatorial matrix theory*. Cambridge University Press, 1991, vol. 39.

[24] A. Schrijver, *Theory of linear and integer programming*. John Wiley & Sons, 1998.