Joint Wireless and Edge Computing Resource Management With Dynamic Network Slice Selection

Sladana Jošilo and György Dán, *Senior Member, IEEE*

***Abstract*— Network slicing is a promising approach for enabling low latency computation offloading in edge computing systems. In this paper, we consider an edge computing system under network slicing in which the wireless devices generate latency sensitive computational tasks. We address the problem of joint dynamic assignment of computational tasks to slices, management of radio resources across slices and management of radio and computing resources within slices. We formulate the *Joint Slice Selection and Edge Resource Management* (JSS-ERM) problem as a mixed-integer problem with the objective to minimize the completion time of computational tasks. We show that the JSS-ERM problem is NP-hard and develop an approximation algorithm with bounded approximation ratio based on a game theoretic treatment of the problem. We use extensive simulations to provide insight into the performance of the proposed solution from the perspective of the whole system and from the perspective of individual slices. Our results show that the proposed slicing policy can achieve significant gains compared to the equal slicing policy, and that the computational complexity of the proposed task placement algorithm is approximately linear in the number of devices.**

***Index Terms*— Edge computing, network slicing, resource allocation, computation offloading, game theory, decentralized algorithms.**

I. Introduction

NETWORK slicing is emerging as an enabler for providing logical networks that are customized to meet the needs of different kinds of applications, mostly in 5G mobile networks. Horizontal network slices are designed for specific classes of applications, e.g., streaming visual analytics, real-time control, or media delivery, while vertical network slices are designed for specific industries. Slicing is expected to allow flexible and efficient end-to-end provisioning of bandwidth, composition of in-network processing, e.g., in the form of service chains composed of virtual network functions (VNF), and the allocation of dedicated computing resources. At the same time it provides performance isolation. Slicing is particularly appealing in combination with edge computing, as network slicing could allow low latency access to customized computing services located in edge clouds [1], [2].

Flexibility in network slicing is achieved through service orchestration. Orchestration focuses on the deployment and service-aware adaptation of VNFs and edge cloud services based on predicted workloads. Recent works in the area addressed the joint placement and routing of service function chains, formulated as a virtual network embedding problem [3], and the problem of joint resource dimensioning and routing [4], [5]. Typical objectives are maximization of the service capacity or profit under physical (bandwidth and computational power) resource constraints, or the minimization of the energy consumption subject to satisfying service demand.

Common to the works on service orchestration is that they assume that each application is mapped to a specific slice deterministically, and assume a static resource pool per slice so as to ensure performance isolation [3]–[5]. A deterministic mapping is, however, not mandatory in practice. While there may be a designated (default) slice for every application, most proposed architectures for network slicing define a set of allowed slices, and the assignment of an application to a slice can be decided dynamically based on the current workload and SLA requirements [6]. The dynamic assignment of applications to slices thus results in a mixture of workloads in the slices, and consequently calls for flexibility in allocating resources to slices.

The importance of resource management across slices has been widely accepted in the case of the radio access network (RAN) [6]. Such inter-slice resource allocation should happen at short time scales, taking into account slice-level service level agreements (SLAs) and technological constraints (e.g., available RAN technology, such as 5GNR or WiFi-Lic). Recent works in the area have focused on system aspects of virtualizing RANs [7]–[9], and on the allocation of virtual resource block groups to slices so as to maximize efficiency [10], [11]. These works have not addressed, however, the potential impact of inter-slice resource management on service orchestration and on the dynamic assignment of applications to slices, from an algorithmic point of view. It is thus so far unclear how to perform service orchestration and joint management of communication and computing resources in an edge computing system under network slicing. Yet, designing efficient inter-slice and intra-slice policies for allocating communication and computing resources together with

Manuscript received 13 March 2020; revised 3 May 2021 and 5 November 2021; accepted 21 February 2022; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor T. Spyropoulos. Date of publication 10 March 2022; date of current version 18 August 2022. This work was supported in part by the Vinnova Competence Center for Trustworthy Edge Computing Systems and Applications at KTH and in part by the Swedish Research Council under Project 2020-03860. *(Corresponding author: György Dán.)*

The authors are with the Division of Network and Systems Engineering, School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, 11428 Stockholm, Sweden (e-mail: josilo@kth.se; gyuri@kth.se).

Digital Object Identifier 10.1109/TNET.2022.3156178

developing low complexity algorithms for dynamic assignment of computational tasks is an inherently challenging problem.

In this paper we address this problem by considering joint dynamic slice selection, inter-slice radio resource management and intra-slice radio and computing resource management for latency sensitive workloads, and make three important contributions. First, we formulate the joint slice selection and edge resource management (JSS-ERM) problem, and show that it is NP-hard. Second, we analyze the optimal solution structure, and we develop an efficient approximation algorithm with bounded approximation ratio inspired by a game theoretic treatment of the problem. Third, we provide extensive numerical results to show that the resulting system performance significantly outperforms baseline resource allocation policies.

The rest of the paper is organized as follows. Section II introduces the system model and Section III the problem formulation. Sections IV and V provide the analytical results, and Section VI shows numerical results. Section VII discusses related work and Section VIII concludes the paper.

II. System Model

We consider a slicing enabled mobile backhaul including mobile edge computing (MEC) resources that serves a set $\mathcal{N}=\{1,2,\ldots,N\}$ of wireless devices (WDs) that generate computationally intensive tasks. WDs can offload their tasks through a set $\mathcal{A}=\{1,2,\ldots,A\}$ of access points (APs) to a set $\mathcal{C}=\{1,2,\ldots,C\}$ of edge clouds (ECs). APs and ECs form the set $\mathcal{E}\triangleq\mathcal{A}\cup\mathcal{C}$ of edge resources. We denote by $\mathcal{S}=\{1,2,\ldots,S\}$ the set of slices in the network, which include certain combinations of computing resources (e.g., CPUs, GPUs, NPUs and/or FPGAs), optimized for executing some types of tasks.

We characterize a task generated by WD i by the size D_i of the input data and by its complexity, which we define as the *expected* number of instructions required to perform the computation. Since the WDs and the slices may have different instruction set architectures, the number of instructions required to execute the same task may also differ. Hence, for a task generated by WD i we denote by L_i and $L_{i,s}$ the expected number of instructions required to perform the computation locally and in slice s, respectively. Similar to other works [12]–[14], we consider that D_i, L_i and $L_{i,s}$ can be estimated from measurements by applying the methods described in [15]–[17].

We consider that each WD i generates a computational task at a time; each task is atomic and can be either offloaded for computation or performed locally on the WD it was generated at. In the case of offloading, the WD will be assigned to exactly one slice $s\in\mathcal{S}$ and within the slice to exactly one AP $a\in\mathcal{A}_i\subseteq\mathcal{A}$ through which it can offload the computation to exactly one EC $c\in\mathcal{C}$. Therefore, we define the set of feasible decisions for WD i as $\mathfrak{D}_i\triangleq\{i\}\cup\{(a,c,s)|a\in\mathcal{A}_i,c\in\mathcal{C},s\in\mathcal{S}\}$ and we use variable $d_i\in\mathfrak{D}_i$ to indicate the decision for WD i's task (i.e., $d_i=i$ indicates that WD i performs the task locally and $d_i=(a,c,s)$ indicates that WD i should offload its task through AP a to EC c in slice s). Furthermore, we define a decision vector $\mathbf{d}\triangleq(d_i)_{i\in\mathcal{N}}$ as the collection of the decisions of all WDs and we define the set $\mathfrak{D}\triangleq\times_{i\in\mathcal{N}}\mathfrak{D}_i$, i.e., the set of all possible decision vectors.

For a decision vector $\mathbf{d}\in\mathfrak{D}$ we define the set $O_{a,s}(\mathbf{d})\triangleq\{i\in\mathcal{N}|d_i=(a,\cdot,s),a\in\mathcal{A}_i,s\in\mathcal{S}\}$ of all WDs that use AP a in slice s and the set $O_a(\mathbf{d})=\cup_{s\in\mathcal{S}}O_{a,s}(\mathbf{d})$ of all WDs that use AP a. Similarly, considering that the offloading decisions of WDs are given by $\mathbf{d}$, we define the set $O_{c,s}(\mathbf{d})\triangleq\{i\in\mathcal{N}|d_i=(\cdot,c,s),c\in\mathcal{C},s\in\mathcal{S}\}$ of all WDs that use EC c in slice s and the set $O_c(\mathbf{d})=\cup_{s\in\mathcal{S}}O_{c,s}(\mathbf{d})$ of all WDs that use EC c. Finally, we define the local computing singleton set $O_i(\mathbf{d})\subset\{i,\emptyset\}$ for WD i (i.e., $O_i(\mathbf{d})=\{i\}$ when WD i performs the computation locally and $O_i(\mathbf{d})=\emptyset$ otherwise) and the set $O_l(\mathbf{d})=\cup_{i\in\mathcal{N}}O_i(\mathbf{d})$ of all WDs that perform the computation locally in a given decision vector $\mathbf{d}$.

Figure 1 shows an example of a slicing enabled MEC system that consists of $N=7$ WDs, $C=2$ ECs and $A=3$ APs and $S=4$ slices; WDs 1 and 2 offload their tasks through AP a, WD 3 offloads its tasks through AP b, WDs 4 and 5 offload their tasks through AP c, and WDs 6 and 7 perform their computation locally. In what follows we discuss our models of communication and computing resources.

A. Communication Resources

Communication resources in the system are managed at two levels: at the network level and at the slice level.

At the network level, the radio resources of each AP $a\in\mathcal{A}$ are shared across the slices according to the *inter-slice radio resource allocation policy* $\mathcal{P}_b:\mathfrak{D}\rightarrow\mathbb{R}_{[0,1]}^{|\mathcal{A}|\times|\mathcal{S}|}$, which determines the inter-slice radio resource provisioning coefficients $b_a^{\mathrm{s}}\in[0,1]$, $\forall(a,s)\in\mathcal{A}\times\mathcal{S}$ such that $\sum_{s\in\mathcal{S}}b_a^{\mathrm{s}}\leq 1$, $\forall a\in\mathcal{A}$.

At the slice level, the radio resources assigned to each slice $s\in\mathcal{S}$ are shared among the WDs according to an *intra-slice radio resource allocation policy* $\mathcal{P}_{w_a}^{\mathrm{s}}:\mathfrak{D}\rightarrow\mathbb{R}_{[0,1]}^{|\mathcal{A}|\times|\mathcal{N}|}$, which determines the intra-slice radio resource provisioning coefficients $w_{i,a}^{\mathrm{s}}\in[0,1]$, $\forall a\in\mathcal{A}$ and $\forall i\in O_{a,s}(\mathbf{d})$ such that $\sum_{i\in O_{a,s}(\mathbf{d})}w_{i,a}^{\mathrm{s}}\leq 1$, $\forall(a,s)\in\mathcal{A}\times\mathcal{S}$.

We denote by $R_{i,a}$ the achievable PHY rate of WD i at AP a (i.e., if WD i was the only transmitter), which depends on the characteristics of the physical layer, such as the wireless medium, interference, and the modulation and coding scheme. We consider that $R_{i,a}$ captures the expected channel conditions, which can be estimated through historical measurements of the path loss, fading, and interference. Therefore, given $R_{i,a}$ and provisioning coefficients b_a^{s} and $w_{i,a}^{\mathrm{s}}$, we can express the uplink rate of WD i at AP a in slice s as

$$W_{i,a}^{\mathrm{s}}(\mathbf{d},\mathcal{P}_b,\mathcal{P}_{w_a}^{\mathrm{s}})=b_a^{\mathrm{s}}w_{i,a}^{\mathrm{s}}R_{i,a}. \tag{1}$$

The uplink rate (1) together with the input data size D_i determines the transmission time of WD $i\in O_{a,s}(\mathbf{d})$,

$$T_{i,a}^{\mathrm{tx,s}}(\mathbf{d},\mathcal{P}_b,\mathcal{P}_{w_a}^{\mathrm{s}})=\frac{D_i}{W_{i,a}^{\mathrm{s}}(\mathbf{d},\mathcal{P}_b,\mathcal{P}_{w_a}^{\mathrm{s}})}. \tag{2}$$

Similar to previous works [13], [18]–[20] we make the assumption that the time needed to transmit the results of the computation from the EC to the WD can be neglected because for many applications (e.g., face recognition and tracking) the size of the output data is significantly smaller than the size D_i of the input data.

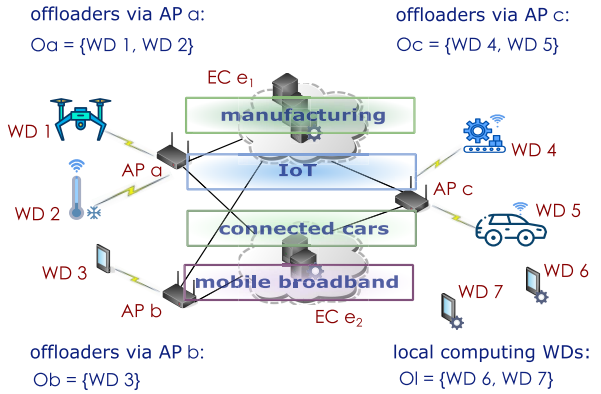

Fig. 1. An example of a slicing enabled MEC system that consists of $N = 7$ WDs, $C = 2$ ECs and $A = 3$ APs and $S = 4$ slices.

B. Computing Resources

Our system model distinguishes between edge cloud resources and local computing resources.

1) Edge Cloud Resources: We consider that each slice $s \in \mathcal{S}$ is equipped with a certain combination of computing resources optimized for executing specific types of tasks (e.g, CPUs, GPUs, NPUs, FPGAs), and we denote by F^{s}_c the computing capability of EC c in slice s. The computing resources within a slice are shared among the WDs according to the *intra-slice computing power allocation policy* $\mathcal{P}^{\mathrm{s}}_{w_c} : \mathfrak{D} \rightarrow \mathbb{R}^{|\mathcal{C}|\times|\mathcal{N}|}_{[0,1]}$, which determines the intra-slice computing power provisioning coefficients $w^{\mathrm{s}}_{i,c} \in [0,1]$, $\forall c \in \mathcal{C}$ and $\forall i \in O_{c,s}(\mathbf{d})$ such that $\sum_{i\in O_{c,s}(\mathbf{d})} w^{\mathrm{s}}_{i,c} \leq 1$, $\forall (c,s) \in \mathcal{C}\times\mathcal{S}$.

Given the computing capability F^{s}_c we can express the computing capability allocated to WD i in EC c in slice s as

$$F^{\mathrm{s}}_{i,c}(\mathbf{d}, \mathcal{P}^{\mathrm{s}}_{w_c}) = w^{\mathrm{s}}_{i,c} F^{\mathrm{s}}_c. \tag{3}$$

In order to account for the diversity of computing resources provided by different slices we use the coefficient $h_{i,s} \in \mathbb{R}_{\geq 0}$ to capture how well a slice s is tailored for executing a task generated by WD i and we express the expected number of instructions $L_{i,s}$ required to execute a task generated by WD i in slice s as $L_{i,s} = L_i/h_{i,s}$ (i.e., a high $h_{i,s}$ indicates that a task generated by WD i is a good match for the computing resources in slice s). Thus, in our model the computing capability (3) together with the expected task complexity $L_{i,s}$ determines the task execution time of WD $i \in O_{c,s}(\mathbf{d})$ as

$$T^{\mathrm{ex,s}}_{i,c}(\mathbf{d}, \mathcal{P}^{\mathrm{s}}_{w_c}) = \frac{L_{i,s}}{F^{\mathrm{s}}_{i,c}(\mathbf{d}, \mathcal{P}^{\mathrm{s}}_{w_c})}. \tag{4}$$

2) Local Computing Resources: We denote by F^l_i the computing capability of WD i and we express the local execution time T^{ex}_i of WD i as

$$T^{\mathrm{ex}}_i = \frac{L_i}{F^l_i}. \tag{5}$$

C. Cost Model

We define the system cost as the aggregate completion time of all WDs. Before providing a formal definition, we introduce the shorthand notation

$$\tau^{\mathrm{s}}_{i,e} = \begin{cases} \dfrac{D_i}{R_{i,e}} & \text{if } i \in \mathcal{N}, e \in \mathcal{E}\cap\mathcal{A} \\ \dfrac{L_{i,s}}{F^{\mathrm{s}}_e} & \text{if } i \in \mathcal{N}, e \in \mathcal{E}\cap\mathcal{C}. \end{cases} \tag{6}$$

Observe that for $e \in \mathcal{E}\cap\mathcal{A}$ we have that e is a communication resource and $\tau^{\mathrm{s}}_{i,e}$ is the minimum transmission time that WD i would achieve if it was the only WD offloading its computation through AP e in slice s. Similarly, for $e \in \mathcal{E}\cap\mathcal{C}$ we have that e is a computing resource and $\tau^{\mathrm{s}}_{i,e}$ is the minimum execution time that WD i would achieve if it was the only WD offloading its computation to EC e in slice s.

Finally, for notational convenience let us define the indicator function $I(d_i, d)$ for WD i as

$$I(d_i, d) = \begin{cases} 1 & \text{if } d_i = d, \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

Cost of WD i: When offloading, the task completion time consists of two parts: the time needed to transmit the data pertaining to a task through an AP and the time needed to execute a task in an EC. In the case of local computing, the task completion time depends only on the local execution time. Therefore, the cost of WD i can be expressed as

$$\begin{aligned} &C_i(\mathbf{d}, \mathcal{P}_b, \mathcal{P}_{w_a}, \mathcal{P}_{w_c}) \\ &\quad = T^{\mathrm{ex}}_i I(d_i, i) \\ &\qquad + \sum_{s\in\mathcal{S}}\sum_{c\in\mathcal{C}}\sum_{a\in\mathcal{A}} \left(\frac{\tau^{\mathrm{s}}_{i,a}}{b^{\mathrm{s}}_a w^{\mathrm{s}}_{i,a}} + \frac{\tau^{\mathrm{s}}_{i,c}}{w^{\mathrm{s}}_{i,c}}\right) I(d_i, (a,c,s)). \end{aligned} \tag{8}$$

where $(\mathcal{P}_{w_a}, \mathcal{P}_{w_c}) = ((\mathcal{P}^{\mathrm{s}}_{w_a}, \mathcal{P}^{\mathrm{s}}_{w_c}))_{s\in\mathcal{S}}$ denotes the collection of the slices' policies.

Cost per slice: We express the cost in slice s as a sum of transmission and executions times of all WDs offloading their tasks in slice s

$$C^{\mathrm{s}}(\mathbf{d}, \mathcal{P}_b, \mathcal{P}^{\mathrm{s}}_{w_a}, \mathcal{P}^{\mathrm{s}}_{w_c}) = \sum_{e\in\mathcal{E}} \sum_{i\in O_{e,s}(\mathbf{d})} \frac{\tau^{\mathrm{s}}_{i,e}}{b^{\mathrm{s}}_e w^{\mathrm{s}}_{i,e}}, \tag{9}$$

where $b^{\mathrm{s}}_e = 1$ if e is a computing resource (i.e., $e \in \mathcal{E}\cap\mathcal{C}$).

System cost: Finally, we express the system cost as

$$\begin{aligned} C(\mathbf{d}, \mathcal{P}_b, \mathcal{P}_{w_a}, \mathcal{P}_{w_c}) &= \sum_{i\in\mathcal{N}} C_i(\mathbf{d}, \mathcal{P}_b, \mathcal{P}^{\mathrm{s}}_{w_a}, \mathcal{P}^{\mathrm{s}}_{w_c}) \\ &= \sum_{s\in\mathcal{S}} C^{\mathrm{s}}(\mathbf{d}, \mathcal{P}_b, \mathcal{P}^{\mathrm{s}}_{w_a}, \mathcal{P}^{\mathrm{s}}_{w_c}) + \sum_{i\in O_l(\mathbf{d})} C^l_i. \end{aligned} \tag{10}$$

For ease of reference, the key notations used in the paper are summarized in Table I.

III. Problem Formulation

We consider that the network operator aims at minimizing the system cost $C(\mathbf{d}, \mathcal{P}_b, \mathcal{P}_{w_a}, \mathcal{P}_{w_c})$ by finding an optimal vector $\hat{\mathbf{d}}$ of offloading decisions, and an optimal collection $(\hat{\mathcal{P}}_b, \hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})$ of policies for sharing the edge resources across slices and within slices. We refer to the problem as the *Joint Slice Selection and Edge Resource Management* (JSS-ERM) problem. Since the WDs generate atomic tasks

TABLE I
SUMMARY OF KEY NOTATIONS

Notation	Description
$\mathcal{N}$	Set of N WDs
$\mathcal{A}$	Set of A APs
$\mathcal{A}_i$	Set of APs available for offloading to WD i
$\mathcal{C}$	Set of C ECs
$\mathcal{E}$	Set of APs and ECs, $\mathcal{E} = \mathcal{A} \cup \mathcal{C}$
$\mathcal{S}$	Set of S slices
$\mathcal{P}_b$	Inter-slice radio resource allocation policy
$\mathcal{P}^{\mathrm{s}}_{w_a}$	Intra-slice radio resource allocation policy
$\mathcal{P}^{\mathrm{s}}_{w_c}$	Intra-slice computing power allocation policy
b^{s}_a	Inter-slice radio resource provisioning coefficient
$w^{\mathrm{s}}_{i,a}$	Intra-slice radio resource provisioning coefficient
$w^{\mathrm{s}}_{i,c}$	Intra-slice computing power provisioning coefficient
D_i	Mean size of the input data for WD i
L_i	Mean WD i's task complexity when executing locally
$L_{i,s}$	Mean WD i's task complexity when offloading in slice s
F^l_i	Computational capability of WD i
T^{ex}_i	Local execution time of WD i
$R_{i,a}$	Uplink PHY rate of WD i towards AP a
F^{s}_c	Computing capability of EC c in slice s
$\mathfrak{D}_i$	Set of feasible offloading decisions for WD i
d_i	Offloading decision for WD i, $d_i \in \mathfrak{D}_i$
$\mathbf{d}$	Offloading decision vector, $\mathbf{d} = (d_i)_{i\in\mathcal{N}}$
$O_{a,s}(\mathbf{d})$	Set of WDs offloading in slice s through AP a in $\mathbf{d}$
$O_a(\mathbf{d})$	Set of WDs offloading through AP a in $\mathbf{d}$
$O_{c,s}(\mathbf{d})$	Set of WDs offloading in slice s to EC c in $\mathbf{d}$
$O_c(\mathbf{d})$	Set of WDs offloading to EC c in $\mathbf{d}$
$O_l(\mathbf{d})$	Set of WDs performing the computation locally in $\mathbf{d}$
$W^{\mathrm{s}}_{i,a}(\cdot)$	Uplink rate of WD $i \in O_{a,s}(\mathbf{d})$
$T^{\mathrm{tx,s}}_{i,a}(\cdot)$	Transmission time of WD $i \in O_{a,s}(\mathbf{d})$
$F^{\mathrm{s}}_{i,c}(\cdot)$	Computing capability of WD $i \in O_{c,s}(\mathbf{d})$
$T^{\mathrm{ex,s}}_{i,c}(\cdot)$	Task execution time of WD $i \in O_{c,s}(\mathbf{d})$
$\tau^{\mathrm{s}}_{i,a}$	Minimum transmission time of WD $i \in O_{a,s}(\mathbf{d})$
$\tau^{\mathrm{s}}_{i,c}$	Minimum execution time of WD $i \in O_{c,s}(\mathbf{d})$
$C_i(\cdot)$	Cost of WD i
$C^{\mathrm{s}}(\cdot)$	Cost in slice s
$C(\cdot)$	System cost

that cannot be further split, the JSS-ERM is a mixed-integer optimization problem, and can be formulated as

$$\min_{\mathbf{d},\mathcal{P}_b,\mathcal{P}_{w_a},\mathcal{P}_{w_c}} C(\mathbf{d},\mathcal{P}_b,\mathcal{P}_{w_a},\mathcal{P}_{w_c}) \tag{11}$$

$$\text{s.t.} \sum_{d\in\mathfrak{D}_i} I(d_i,d) = 1, \quad \forall i \in \mathcal{N}, \tag{12}$$

$$C_i(\mathbf{d},\mathcal{P}_b,\mathcal{P}_{w_a},\mathcal{P}_{w_c}) \le T^{\mathrm{ex}}_i, \quad \forall i \in \mathcal{N}, \tag{13}$$

$$\sum_{s\in\mathcal{S}} b^{\mathrm{s}}_a \le 1, \quad \forall a \in \mathcal{A}, \tag{14}$$

$$\sum_{j\in O_{e,s}(\mathbf{d})} w^{\mathrm{s}}_{j,e} \le 1, \quad \forall e \in \mathcal{E},\ \forall s \in \mathcal{S}, \tag{15}$$

$$b^{\mathrm{s}}_a \ge 0, \quad \forall a \in \mathcal{A}, \forall s \in \mathcal{S}, \tag{16}$$

$$w^{\mathrm{s}}_{i,e} \ge 0, \quad \forall i \in \mathcal{N},\ \forall e \in \mathcal{E},\ \forall s \in \mathcal{S}. \tag{17}$$

Constraint (12) enforces that each WD either performs the computation locally or offloads its task to exactly one logical resource $(a,c,s) \in \mathcal{A}\times\mathcal{C}\times\mathcal{S}$; constraint (13) ensures that the task completion time in the case of offloading is not greater than the task completion time in the case of local computing; constraint (14) enforces a limitation on the amount of communication resources of an AP that can be provided to each slice; constraint (15) enforces a limitation on the amount of communication resources of an AP and the amount of computing resources of an EC that can be provided to each WD in each slice.

Theorem 1: The JSS-ERM defined by (11)-(17) is NP-hard.

Proof: We provide the proof in Section IV-B. □

In the rest of the paper we develop an approximation scheme for the JSS-ERM problem based on a decomposition of the problem, and by adopting a game theoretic interpretation of one of the subproblems.

IV. NETWORK SLICE ORCHESTRATION AND EDGE RESOURCE ALLOCATION

In what follows we show that the JSS-ERM problem can be solved through solving a series of smaller optimization problems. To do so, we start with considering the problem of finding the collection $(\hat{\mathcal{P}}_b,\hat{\mathcal{P}}_{w_a},\hat{\mathcal{P}}_{w_c})$ of optimal resource allocation policies for a given vector $\mathbf{d}$ of offloading decisions. Our first result concerns the convexity of the resulting optimization problem.

Proposition 1: Consider an offloading decision vector $\boldsymbol{d}$ for which constraint (13) can be satisfied. Furthermore, define the problem of finding a collection $(\hat{\mathcal{P}}_b,\hat{\mathcal{P}}_{w_a},\hat{\mathcal{P}}_{w_c})$ of optimal resource allocation policies,

$$\min_{\mathcal{P}_b,\mathcal{P}_{w_a},\mathcal{P}_{w_c}} C(\boldsymbol{d},\mathcal{P}_b,\mathcal{P}_{w_a},\mathcal{P}_{w_c}) \tag{18}$$

$$s.t. (13) - (17). \tag{19}$$

The problem (18)-(19) is convex.

Proof: The proof is given in Appendix A. □

Proposition 2: The collection $(\hat{\mathcal{P}}_b,\hat{\mathcal{P}}_{w_a},\hat{\mathcal{P}}_{w_c})$ of optimal resource allocation policies that solves problem (18)-(19) sets the provisioning coefficients according to

$$\hat{w}^{\mathrm{s}}_{i,e} = \frac{\sqrt{\tau^{\mathrm{s}}_{i,e}}}{\sum_{j\in O_{e,s}(\boldsymbol{d})}\sqrt{\tau^{\mathrm{s}}_{j,e}}}, \forall e\in\mathcal{E},\ \ \forall s\in\mathcal{S},\ \forall i\in O_{e,s}(\boldsymbol{d}), \tag{20}$$

$$\hat{b}^{\mathrm{s}}_a = \frac{\sum_{j\in O_{a,s}(\boldsymbol{d})}\sqrt{\tau^{\mathrm{s}}_{j,a}}}{\sum_{s'\in\mathcal{S}}\sum_{j\in O a,s'(\boldsymbol{d})}\sqrt{\tau^{\mathrm{s}'}_{j,a}}}, \quad \forall a\in\mathcal{A}, \forall s\in\mathcal{S}. \tag{21}$$

Proof: The proof is given in Appendix B. □

As a first step in the decomposition, let us consider the problem of finding an optimal collection $(\mathcal{P}^*_{w_a},\mathcal{P}^*_{w_c}) = ((\mathcal{P}^{\mathrm{s},*}_{w_a},\mathcal{P}^{\mathrm{s},*}_{w_c}))_{s\in\mathcal{S}}$ of resource allocation policies of slices for fixed offloading vector $\mathbf{d}$ and inter-slice policy $\mathcal{P}_b$.

Proposition 3: Consider an offloading decision vector $\boldsymbol{d}$ for which constraint (13) can be satisfied and a policy $\mathcal{P}_b$ for setting the inter-slice radio resource provisioning coefficients $b^{\mathrm{s}}_a, \forall a\in\mathcal{A}, \forall s\in\mathcal{S}$. Then the solution to the problem

$$\min_{\mathcal{P}^{\mathrm{s}}_{w_a},\mathcal{P}^{\mathrm{s}}_{w_c}} C^{\mathrm{s}}(\boldsymbol{d},\mathcal{P}_b,\mathcal{P}^{\mathrm{s}}_{w_a},\mathcal{P}^{\mathrm{s}}_{w_c}) \tag{22}$$

$$s.t. (13), (15), (17). \tag{23}$$

is given by (20), i.e., $(\mathcal{P}^{\mathrm{s},*}_{w_a},\mathcal{P}^{\mathrm{s},*}_{w_c}) = (\hat{\mathcal{P}}^{\mathrm{s}}_{w_a},\hat{\mathcal{P}}^{\mathrm{s}}_{w_c}), \forall s\in\mathcal{S}$.

Proof: The result can be proved by following the approach presented in the proof of Proposition 2. □

As a second step, let us consider the problem of finding an optimal policy $\mathcal{P}^*_b$ for a fixed vector $\mathbf{d}$ of offloading decisions

and the optimal collection $(\mathcal{P}^*_{w_a}, \mathcal{P}^*_{w_c}) = (\hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})$ of the slices' policies.

Proposition 4: Consider an offloading decision vector $\boldsymbol{d}$ *for which the constraint (13) can be satisfied. Furthermore, let us substitute (20) into (11)-(17) and define the problem of finding an optimal inter-slice radio resource allocation policy* $\mathcal{P}^*_b$*, i.e., a solution to*

$$\min_{\mathcal{P}_b} \sum_{s' \in \mathcal{S}} \sum_{a' \in \mathcal{A}} \frac{1}{b^{s'}_{a'}} \Big(\sum_{j \in O_{a',s'}(\boldsymbol{d})} \sqrt{\tau^{s'}_{j,a'}} \Big)^2 \tag{24}$$

$$s.t. (13), (14) \text{ and } (16). \tag{25}$$

Then, the optimal inter-slice radio resource allocation policy $\mathcal{P}^*_b$ *sets the inter-slice provisioning coefficients according to (21), i.e.,* $\mathcal{P}^*_b = \hat{\mathcal{P}}_b$.

Proof: The result can be proved by following the approach presented in the proof of Proposition 2. □

By combining the above two results, we are now ready to show that the JSS-ERM problem can be decomposed into a sequence of optimization problems.

Theorem 2: The solution to problem (18)-(19) can be obtained by finding the optimal policies $(\hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})$ *first, and finding the optimal policy* $\hat{\mathcal{P}}_b$ *second, i.e.,*

$$\min_{\mathcal{P}_b, \mathcal{P}_{w_a}, \mathcal{P}_{w_c}} C(\boldsymbol{d}, \mathcal{P}_b, \mathcal{P}_{w_a}, \mathcal{P}_{w_c}) = \min_{\mathcal{P}_b} \min_{\mathcal{P}_{w_a}, \mathcal{P}_{w_c}} C(\boldsymbol{d}, \mathcal{P}_b, \mathcal{P}_{w_a}, \mathcal{P}_{w_c}) \tag{26}$$

Proof: The result follows from the proofs of Proposition 2, Proposition 3 and Proposition 4. □

Furthermore, as the next theorem shows, we can use this decomposition structure also for computing the optimal offloading decision vector.

Theorem 3: The solution to problem (11)-(17) can be obtained by finding the optimal collection $(\hat{\mathcal{P}}_b, \hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})$ *of resource allocation policies first, and finding an optimal offloading decision vector* $\hat{\boldsymbol{d}}$ *second, i.e.,*

$$\min_{\boldsymbol{d}, \mathcal{P}_b, \mathcal{P}_{w_a}, \mathcal{P}_{w_c}} C(\boldsymbol{d}, \mathcal{P}_b, \mathcal{P}_{w_a}, \mathcal{P}_{w_c}) = \min_{\boldsymbol{d}} \min_{\mathcal{P}_b} \min_{\mathcal{P}_{w_a}, \mathcal{P}_{w_c}} C(\boldsymbol{d}, \mathcal{P}_b, \mathcal{P}_{w_a}, \mathcal{P}_{w_c}) \tag{27}$$

Proof: It is easy to see that the exact values of the provisioning coefficients are functions of $\hat{\mathbf{d}}$. However, the optimal policies according to which the resources are shared are the same for every offloading decision vector $\mathbf{d} \in \mathfrak{D}$, as defined by (20) and (21). Therefore, one can solve the problem (18)-(19) first, assuming an arbitrary offloading decision vector $\mathbf{d}$, and then given the solution $(\hat{\mathcal{P}}_b, \hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})$ of (18)-(19) find the optimal offloading decision vector $\hat{\mathbf{d}}$ that will determine the exact values of the provisioning coefficients. This proves the result. □

A. Discussion and Practical Implications

So far we have shown that the JSS-ERM problem can be decomposed into a $S+2$ coupled resource allocation problems that can be solved sequentially. It is of interest to discuss the relationship between the decomposition and the potential implementation of a resource allocation and orchestration framework.

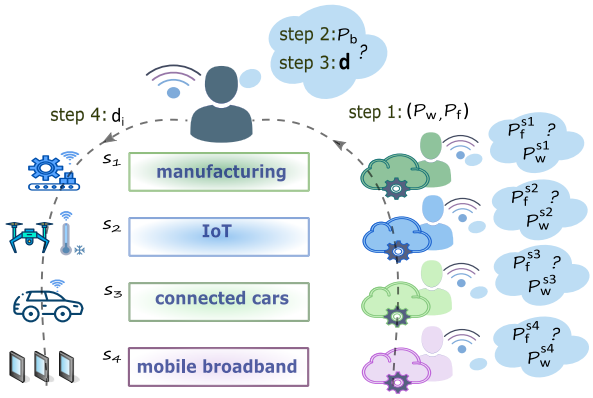

Fig. 2. An example of the potential implementation of a resource allocation and orchestration framework.

The proposed decomposition results in an optimization problem to be solved at the network level (eqns. (24)-(25)) and one in each slice (eqns. (22)-(23)), followed by the problem of finding an optimal offloading decision vector. This structure is aligned with the slice-based network architecture proposed in [6], where inter-slice radio resource allocation and service orchestration are performed by a centralized entity, while intra-slice radio and computing resource management is performed by the slices themselves, i.e., each slice manages its own radio and computing resources. We refer to the centralized entity that decides about the inter-slice radio resource allocation policy $\mathcal{P}_b$ and the offloading decision vector $\mathbf{d}$ as the *slice resource orchestrator* (SRO).

Figure 2 illustrates the interaction between the SRO and slices in the potential implementation of a resource allocation and orchestration framework. As shown in the figure, the slices can first decide about the collection $(\mathcal{P}_{w_a}, \mathcal{P}_{w_c})$ of their intra-slice radio and computing resource allocation policies as each slice can assume an arbitrary offloading decision vector $\mathbf{d}$ when solving the optimization problem (22)-(23). After the slices inform the SRO about their radio and computing resource allocation policies, the SRO can decide about its own inter-slice radio resource allocation policy $\mathcal{P}_b$ by solving the optimization problem (24)-(25) for an arbitrary offloading decision vector $\mathbf{d}$. Finally, given the collection $(\mathcal{P}_b, \mathcal{P}_{w_a}, \mathcal{P}_{w_c})$ of all resource allocation policies and the estimates of the average parameters that are the inputs to the algorithm, the SRO solves the service optimization problem through finding the offloading decision vector $\mathbf{d}$. The SRO then informs each WD i about its offloading decision d_i. Whenever the estimates of the average parameters that are the inputs to the algorithm are updated or when the set of WDs in the system changes, the SRO can recompute the vector of offloading decisions and send the updated decisions to the WDs. The proposed centralized implementation is based on the average system parameters only, and hence it requires low signaling overhead.

B. Problem Complexity

In what follows we provide a result concerning the complexity of the JSS-ERM problem. For notational convenience let us

first define the set of resources $\tilde{\mathcal{R}} \triangleq \{\{\mathcal{A}\times\mathcal{S}\}\cup\{\mathcal{C}\times\mathcal{S}\}\cup\mathcal{N}\}$ and let us introduce the following shorthand notation

$$q_{i,(a,s)} \triangleq \sqrt{\frac{D_i}{R_{i,a}}}, \quad q_{i,(c,s)} \triangleq \sqrt{\frac{L_i}{h_{i,s}}}, \quad q_{i,i} \triangleq \sqrt{L_i},$$
$$q_r(\mathbf{d}) \triangleq \sum_{j\in O_r(\mathbf{d})} q_{j,r}, \quad \forall r \in \tilde{\mathcal{R}}. \tag{28}$$

First, by substituting (20) into (8) and by using the notation introduced in (28), we can express the cost of WD i under a policy $\mathcal{P}_b$ and the collection $(\hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})$ of optimal allocation policies of slices as

$$\tilde{C}_i(\mathbf{d}) = \sum_{r\in\tilde{\mathcal{R}}_{d_i}} m_r q_{i,r} q_r(\mathbf{d}), \tag{29}$$

where $\tilde{\mathcal{R}}_{d_i}$ is the set of resources that WD i uses for performing its task in $\mathbf{d}$ (i.e., $\tilde{\mathcal{R}}_{d_i} = \{(a,s),(c,s)\}$ if $d_i = (a,c,s)$ and $\tilde{\mathcal{R}}_{d_i} = \{i\}$ if $d_i = i$) and $m_{(a,s)} = 1/b_a^{\mathrm{s}}$, $m_{(c,s)} = 1/F_c^{\mathrm{s}}$ and $m_i = 1/F_i^l$.

Second, by summing the expressions (29) over all WDs $i \in \mathcal{N}$ and by reordering the summations we can express the system cost (10) under a policy $\mathcal{P}_b$ and the collection $(\hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})$ of optimal allocation policies of slices as

$$\tilde{C}(\mathbf{d}) = \sum_{i\in\mathcal{N}} \tilde{C}_i(\mathbf{d}) = \sum_{r\in\tilde{\mathcal{R}}} m_r q_r^2(\mathbf{d}). \tag{30}$$

Next, let us define the set of resources $\bar{\mathcal{R}} \triangleq \{\mathcal{A}\cup\{\mathcal{C}\times\mathcal{S}\}\cup\mathcal{N}\}$ and a coefficient $q_{i,a} \triangleq q_{i,(a,s)} = \sqrt{D_i/R_{i,a}}$. By substituting (21) into (29), we can express the cost of WD i under the collection $(\hat{\mathcal{P}}_b, \hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})$ of optimal allocation policies as

$$\bar{C}_i(\mathbf{d}) = \sum_{r\in\bar{\mathcal{R}}_{d_i}} m_r q_{i,r} q_r(\mathbf{d}), \tag{31}$$

where $\bar{\mathcal{R}}_{d_i}$ is the set of resources that WD i uses for performing its task in $\mathbf{d}$ (i.e., $\bar{\mathcal{R}}_{d_i} = \{a,(c,s)\}$ if $d_i = (a,c,s)$ and $\bar{\mathcal{R}}_{d_i} = \{i\}$ if $d_i = i$) and $m_a = 1$.

Finally, by summing the expressions (31) over all WDs $i \in \mathcal{N}$ and by reordering the summations we can express the system cost (30) under the collection $(\hat{\mathcal{P}}_b, \hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})$ of optimal allocation policies as

$$\bar{C}(\mathbf{d}) = \sum_{i\in\mathcal{N}} \bar{C}_i(\mathbf{d}) = \sum_{r\in\bar{\mathcal{R}}} m_r q_r^2(\mathbf{d}). \tag{32}$$

Theorem 4: Consider the problem of finding the optimal vector $\hat{\boldsymbol{d}}$ of offloading decisions of WDs under the collection $(\hat{\mathcal{P}}_b, \hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})$ of optimal allocation policies that set provisioning coefficients according to (20) and (21)

$$\min_{\boldsymbol{d}} \bar{C}(\boldsymbol{d}) \tag{33}$$
$$\textit{s.t.}(12). \tag{34}$$

Problem (33)-(34) is NP-hard.

Proof: We prove the NP-hardness of the problem by reduction from the *Minimum Sum of Squares* problem (SP19 problem in [21]): given a finite set $\mathcal{B}$, a size $s(b) \in \mathbb{Z}^+, \forall b \in \mathcal{B}$ and positive integers $K \leq |\mathcal{B}|$ and J, the question is whether $\mathcal{B}$ can be partitioned into K disjoint subsets $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_K$ such that $\sum_{k=1}^{K} \Big(\sum_{b\in\mathcal{B}_k} s(b) \Big)^2 \leq J$.

For the reduction we set $S = 1$, $C = 0$ and $F_i^l = 0$, $\forall i \in \mathcal{N}$, i.e., in this simplified version of the problem $\mathcal{E} = \mathcal{A}$. Next, we let $\mathcal{N} = \mathcal{B}$, $|\mathcal{A}| = K$, $R_{i,a} = R_i$, $\forall i \in \mathcal{N}$, $\forall a \in \mathcal{A}$ and $\sqrt{D_i/R_i} = s(b)$. Then, it follows from (30) that the optimal solution of (33)-(34) provides the solution to the SP19 problem. As SP19 is NP-hard, problem (33)-(34) is also NP-hard, which proves the theorem. □

Proof: [Proof of Theorem 1] The result follows from Theorem 3 and Theorem 4. □

$\mathbf{d}^* = \mathrm{COS}(\mathbf{d}^0, \mathcal{P}_b, \mathcal{P}_w^*, \mathcal{P}_w^*)$

$\mathbf{d} \leftarrow \mathbf{d}^0$
while $\exists\, WD\, j \in \mathcal{N}$ *s.t.* $d_j \neq \arg\min_{d'_j\in\mathfrak{D}_j} \tilde{C}_j(d'_j, d_{-j})$
  $d_j^* = \arg\min_{d'_j\in\mathfrak{D}_j} \tilde{C}_j(d'_j, d_{-j})$
  $\mathbf{d} = (d_j^*, d_{-j})$
end
$\mathbf{d}^* = \mathbf{d}$

Fig. 3. Pseudo code of the COS algorithm.

V. Approximation Scheme for the JSS-ERM Problem

In what follows we propose the *choose offloading slice* (COS) algorithm that the SRO can use for computing an approximation to the optimal solution of the JSS-ERM problem. In particular, the algorithm serves as an approximation scheme to the problem of finding an optimal offloading decision vector. The algorithm starts from an offloading decision vector $\mathbf{d}^0$ in which all WDs perform computation locally, and it lets the SRO to update the offloading decisions of the WDs one at a time, based on their cost functions $\tilde{C}_i(\mathbf{d})$. We show the pseudo code of the algorithm in Figure 3.

Theorem 5: Consider an allocation policy $\mathcal{P}_b$ and the collection $(\hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})$ of optimal allocation policies of slices. The COS algorithm terminates after a finite number of the iterations.

Proof: The proof is based on a game theoretic treatment of the problem

$$\min_{\mathbf{d}} \tilde{C}(\mathbf{d}) \tag{35}$$
$$\text{s.t.}(12), \tag{36}$$

in which the inter-slice radio resource provisioning coefficients are set according to an arbitrary policy $\mathcal{P}_b$ and the intra-slice radio and computing power provisioning coefficients are set according to the optimal policies $\hat{\mathcal{P}}_{w_a}$ and $\hat{\mathcal{P}}_{w_c}$, respectively.

In what follows we show that the problem (35)-(36) can be interpreted as a congestion game $\Gamma(\mathcal{P}_b, \hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c}) =< \mathcal{N}, (\mathfrak{D}_i)_{i\in\mathcal{N}}, (\tilde{C}_i)_{i\in\mathcal{N}} >$ with resource-dependent weights $q_{i,r}$, $i \in \mathcal{N}$, $r \in \tilde{\mathcal{R}}$, and the cost of WD i in the resulting game is given by (29). First, observe that $q_{i,r}$ can be interpreted as

the weight that WD i contributes to the congestion when using resource $r \in \tilde{\mathcal{R}}$ and thus $q_r(\mathbf{d})$ can be interpreted as the total congestion on resource r in strategy profile $\mathbf{d}$. This in fact implies that the cost (29) of WD i in strategy profile $\mathbf{d}$ depends on its own resource-dependent weights $q_{i,r}$ and on the total congestion $q_r(\mathbf{d})$ on the resources it uses. Therefore, it follows from [22] that the problem (35)-(36) can be interpreted as a congestion game $\Gamma(\mathcal{P}_b, \hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})$ with resource dependent weights. Consequently, the COS algorithm terminates after a finite number of iterations iff the game $\Gamma(\mathcal{P}_b, \hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})$ has a pure strategy Nash equilibrium. [1]

Since the cost $c_r(\mathbf{d}) \triangleq m_r q_r(\mathbf{d})$ of sharing every resource $r \in \tilde{\mathcal{R}}$ is an affine function of the congestion $q_r(\mathbf{d})$ on resource r, it follows from Theorem 4.2 in [22] that the game $\Gamma(\mathcal{P}_b, \hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})$ has the exact potential function[2] given by

$$\Psi(\mathbf{d}) = \sum_{i \in \mathcal{N}} \sum_{r \in \tilde{\mathcal{R}}_{d_i}} q_{i,r} c_r^{\leq i}(\mathbf{d}), \tag{38}$$

where $c_r^{\leq i}(\mathbf{d}) = m_r q_r^{\leq i}(\mathbf{d})$ and $q_r^{\leq i}(\mathbf{d}) = \sum_{\{j \in O_r(\mathbf{d}) | j \leq i\}} q_{i,r}$.

It is well known that in a finite strategic game that admits an exact potential all improvement paths[3] are finite [23] and thus the existence of the exact potential function (38) allows us to use the COS algorithm for computing a pure strategy Nash equilibrium $\mathbf{d}^*$ of the game $\Gamma(\mathcal{P}_b, \hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})$, which proves the result. □

Theorem 6: The COS algorithm terminates after a finite number of the iterations for the collection $(\hat{\mathcal{P}}_b, \hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})$ of optimal allocation policies.

Proof: By following the same approach as in the proof of Theorem 5, it is easy to show that given the collection $(\hat{\mathcal{P}}_b, \hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})$ of optimal allocation policies, the problem (33)-(34) can be interpreted as a congestion game $\Gamma(\hat{\mathcal{P}}_b, \hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c}) =< \mathcal{N}, (\mathfrak{D}_i)_{i \in \mathcal{N}}, (\bar{C}_i)_{i \in \mathcal{N}} >$ with resource-dependent weights $q_{i,r}$, $i \in \mathcal{N}$, $r \in \bar{\mathcal{R}}$, and the cost of WD i in the resulting game is given by (31).

Since $m_a = 1, \forall a \in \mathcal{A}$, the cost $c_r(\mathbf{d}) \triangleq m_r q_r(\mathbf{d})$ of sharing every resource $r \in \bar{\mathcal{R}}$ is an affine function of the congestion on resource r. Therefore, the game $\Gamma(\hat{\mathcal{P}}_b, \hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})$ is also an exact potential game, and thus the COS algorithm computes a pure strategy Nash equilibrium $\mathbf{d}^*$ of the game $\Gamma(\hat{\mathcal{P}}_b, \hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})$, which proves the result. □

In general, the number of improvement steps can be exponential in a potential game, but as we show next the COS algorithm can compute an equilibrium $\mathbf{d}^*$ of offloading decisions efficiently.

[1]A pure strategy Nash equilibrium of a strategic game is a collection $\mathbf{d}^*$ of decisions (called a strategy profile) for which $\tilde{C}_i(d*_i, d*_{-i}) \leq \tilde{C}_i(d_i, d*_{-i})$, $\forall d_i$, where $d*_i$ and $d*_{-i}$ are standard game theoretical notations for an improvement step of player i and for the collection of decisions (strategies) of all players other than i, respectively.

[2]A function $\Psi : \times_i(\mathfrak{D}_i) \rightarrow \mathbb{R}$ is an exact potential for a finite strategic game if for an arbitrary strategy profile (d_i, d_{-i}) and for any improvement step d_i^* the following holds:

$$\Psi(d_i, d_{-i}) - \Psi(d_i^*, d_{-i}) = \tilde{C}_i(d_i, d_{-i}) - \tilde{C}_i(d_i^*, d_{-i}). \tag{37}$$

[3]An improvement path is a sequence of strategy profiles in which one player at a time changes its strategy through performing an improvement step.

Theorem 7: The COS algorithm terminates in $\mathcal{O}(\frac{NC^{\max}}{\epsilon} \log \frac{\sum_{i \in \mathcal{N}} T_i^{\text{ex}}}{\Psi^{\min}})$ iterations, where $C^{\max}$ and ϵ are system parameter dependent constants and $\Psi^{\min}$ is the minimum value of the potential function.

Proof: First, let us denote by $d_i' \in \mathcal{A} \times \mathcal{C} \times \mathcal{S}$ the offloading decision of WD i for which it would have the highest offloading cost if it was the only WD offloading, i.e., $d_i' = (a', c', s') = \arg\max_{(a,c,s) \in \mathcal{A} \times \mathcal{C} \times \mathcal{S}} (D_i / R_{i,a} + L_{i,s} / F_c^{\text{s}})$. Furthermore, let us denote by $T_i^{\text{off,max}}$ the highest offloading cost that a WD i can experience in the system. It is easy to see that WD i experiences offloading cost $T_i^{\text{off,max}}$ when all WDs $j \in \mathcal{N}$ offload their tasks through AP a' to EC c' in slice s', i.e., when $d_j = (a', c', s')$ for each $j \in \mathcal{N}$. Finally, let us define the maximum cost that a WD can experience in the system as $C^{\max} \triangleq \max_{i \in \mathcal{N}} \max\{T_i^{\text{ex}}, T_i^{\text{off,max}}\}$.

Now, let us denote by ϵ_i the smallest decrease in the cost that WD i can experience when its decision is updated using the COS algorithm and let us define the constant $\epsilon \triangleq \min_{i \in \mathcal{N}} \epsilon_i$. Next, let us consider an iteration of the COS algorithm where the offloading decision of WD i is updated from d_i to d_i*. We can then write

$$\begin{aligned} \Psi(d_i, d_{-i}) - \Psi(d*_i, d_{-i}) &= C_i(d_i, d_{-i}) - C_i(d*_i, d_{-i}) \\ &\geq \epsilon \geq \epsilon \frac{\Psi(d_i, d_{-i})}{NC^{\max}}, \end{aligned} \tag{39}$$

where $C_i(\cdot)$ is defined by (29) in the case of $\Gamma(\mathcal{P}_b, \hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})$ and by (31) in the case of $\Gamma(\hat{\mathcal{P}}_b, \hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})$. Observe that in (39) we have that the equality follows from the definition of the exact potential function (37), the first inequality follows from the definition of ϵ, and the last inequality follows from the facts that $C(\mathbf{d}) = \sum_{i \in \mathcal{N}} C_i(\mathbf{d}) \leq NC^{\max}$ and $\Psi(\mathbf{d}) \leq C(\mathbf{d})$ for any vector $\mathbf{d}$ of offloading decisions (c.f., (30), (32) and (38)).

We can rewrite (39) as

$$\Psi(d_i*, d_{-i}) \leq (1 - \frac{\epsilon}{NC^{\max}}) \Psi(d_i, d_{-i}). \tag{40}$$

Observe that from (40) it follows that the COS algorithm decreases the potential function by at least a factor of $(1 - \frac{\epsilon}{NC^{\max}})$. Since ϵ is defined such that $0 < \epsilon < C^{\max}$ holds, we have that $(1 - \frac{\epsilon}{NC^{\max}})^{\frac{NC^{\max}}{\epsilon}} \leq \frac{1}{e}$ and thus every $\frac{NC^{\max}}{\epsilon}$ iterations decrease the potential function by a constant factor.[4]

Next, let us recall that $\Psi(\mathbf{d}) \leq C(\mathbf{d})$ holds for any vector $\mathbf{d}$ of offloading decisions (c.f., (30), (32) and (38)) and that the COS algorithm starts from an offloading decision vector $\mathbf{d}^0$ in which all WDs perform computation locally. Consequently, we have that the potential function starts from a value $\Psi(\mathbf{d}^0) \leq C(\mathbf{d}^0) = \sum_{i \in \mathcal{N}} T_i^{\text{ex}}$ and it cannot drop lower than $\Psi^{\min}$. Therefore, assuming that the COS algorithm converges after $K \frac{NC^{\max}}{\epsilon}$ iterations we obtain from (40) that $\frac{\sum_{i \in \mathcal{N}} T_i^{\text{ex}}}{\Psi^{\min}} (\frac{1}{e})^K \geq \frac{\sum_{i \in \mathcal{N}} T_i^{\text{ex}}}{\Psi^{\min}} (1 - \frac{\epsilon}{NC^{\max}})^{K \frac{NC^{\max}}{\epsilon}} \geq \frac{\Psi(\mathbf{d}^0)}{\Psi^{\min}} (1 - \frac{\epsilon}{NC^{\max}})^{K \frac{NC^{\max}}{\epsilon}} \geq 1$ hold. After taking the logarithm, we obtain that $K \leq \log \frac{\sum_{i \in \mathcal{N}} T_i^{\text{ex}}}{\Psi^{\min}}$ and thus that the COS algorithm converges in $\mathcal{O}(\frac{NC^{\max}}{\epsilon} \log \frac{\sum_{i \in \mathcal{N}} T_i^{\text{ex}}}{\Psi^{\min}})$ iterations, which proves the result. □

[4]To see this, note that $(1 - x)^{\frac{1}{x}} \leq (e^{-x})^{\frac{1}{x}} \leq \frac{1}{e}$ for $x \in (0, 1)$.

In what follows we address the efficiency of the COS algorithm in terms of the cost approximation ratio.

Theorem 8: The COS algorithm is a 2.62*-approximation algorithm for the optimization problem (35)-(36) in terms of the system cost, i.e.,* $\frac{\tilde{C}(\mathbf{d}^*)}{\tilde{C}(\hat{\mathbf{d}})} \leq 2.62$.

Proof: Let us denote by $\mathfrak{D}* \subseteq \mathfrak{D}$ the set of all vectors of offloading decisions that can be computed using the COS algorithm given any policy $\mathcal{P}_b$ and the collection $(\hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})$ of the optimal resource allocation policies of slices. Furthermore, let us consider a vector $\mathbf{d}^* \in \mathfrak{D}*$ and an arbitrary vector $\hat{\mathbf{d}} \in \mathfrak{D}$ of offloading decisions. Since there is no WD i for which the cost $\tilde{C}_i(\mathbf{d}^*)$ can be decreased by unilaterally changing its offloading decision we have the following

$$\begin{aligned}\tilde{C}_i(\mathbf{d}^*) \leq & \sum_{r\in\tilde{\mathcal{R}}_{d_i^*}\cap\tilde{\mathcal{R}}_{\hat{d}_i}} m_r q_{i,r} q_r(\mathbf{d}^*) \\ & + \sum_{r\in\tilde{\mathcal{R}}_{\hat{d}_i}\setminus\tilde{\mathcal{R}}_{d_i^*}} m_r\big(q_r(\mathbf{d}^*)+q_{i,r}\big)q_{i,r} \\ & \leq \sum_{r\in\tilde{\mathcal{R}}_{\hat{d}_i}} m_r\big(q_r(\mathbf{d}^*)+q_{i,r}\big)q_{i,r}, \end{aligned} \tag{41}$$

where $\tilde{\mathcal{R}}_{d_i^*} \subset \tilde{\mathcal{R}}$ and $\tilde{\mathcal{R}}_{\hat{d}_i} \subset \tilde{\mathcal{R}}$ denote the the set of resources that WD i uses in $\mathbf{d}^*$ and $\hat{\mathbf{d}}$, respectively. By summing (41) over all WDs $i \in \mathcal{N}$ and by reordering the summations we obtain

$$\tilde{C}(\mathbf{d}^*) \leq \sum_{r\in\mathcal{R}} \sum_{i\in O_r(\hat{\mathbf{d}})} m_r\big(q_r(\mathbf{d}^*)q_{i,r} + q_{i,r}^2\big). \tag{42}$$

From the definition (28) of the total weight $q_r(\mathbf{d})$ on resource $r \in \tilde{\mathcal{R}}$ and from $\sum_{i\in O_r(\mathbf{d})} q_{i,r}^2 \leq q_r^2(\mathbf{d})$ we obtain

$$\tilde{C}(\mathbf{d}^*) \leq \sum_{r\in\mathcal{R}} m_r q_r(\mathbf{d}^*) q_r(\hat{\mathbf{d}}) + \sum_{r\in\mathcal{R}} m_r q_r^2(\hat{\mathbf{d}}).$$

Next, let us recall the Cauchy-Schwartz inequality $\sum_{r\in\mathcal{R}} a_r b_r \leq \sqrt{\sum_{r\in\mathcal{R}} a_r^2 \sum_{r\in\mathcal{R}} b_r^2}$. By defining $a_r \triangleq \sqrt{m_r} q_r(\mathbf{d}^*)$ and $b_r \triangleq \sqrt{m_r} q_r(\hat{\mathbf{d}})$ we obtain the following

$$\begin{aligned}\tilde{C}(\mathbf{d}^*) \leq & \sqrt{\sum_{r\in\mathcal{R}} m_r q_r^2(\mathbf{d}^*) \sum_{r\in\mathcal{R}} m_r q_r^2(\hat{\mathbf{d}})} \\ & + \sum_{r\in\mathcal{R}} m_r q_r^2(\hat{\mathbf{d}}). \end{aligned} \tag{43}$$

By dividing the right and the left side of (43) by $\sum_{r\in\mathcal{R}} q_r^2(\hat{\mathbf{d}}) > 0$ and by using (30) we obtain

$$\frac{\tilde{C}(\mathbf{d}^*)}{\tilde{C}(\hat{\mathbf{d}})} \leq \sqrt{\frac{\tilde{C}(\mathbf{d}^*)}{\tilde{C}(\hat{\mathbf{d}})}} + 1. \tag{44}$$

Since (44) holds for any vector $\mathbf{d}^* \in \mathfrak{D}*$ of offloading decisions computed by the COS algorithm and for any vector $\hat{\mathbf{d}} \in \mathfrak{D}$ of offloading decisions of the WDs, it holds for the worst vector $\mathbf{d}^* = \arg\max_{\mathbf{d}\in\mathfrak{D}*} \tilde{C}(\mathbf{d})$ of offloading decisions that can be computed using the COS algorithm and for the optimal $\hat{\mathbf{d}} = \arg\min_{\mathbf{d}\in\mathfrak{D}} \tilde{C}(\mathbf{d})$ solution too. Therefore, by solving (44) we obtain that the cost approximation ratio $\frac{\tilde{C}(\mathbf{d}^*)}{\tilde{C}(\hat{\mathbf{d}})}$ of the COS algorithm is upper bounded by $(3+\sqrt{5})/2 \cong 2.62$, which proves the theorem. □

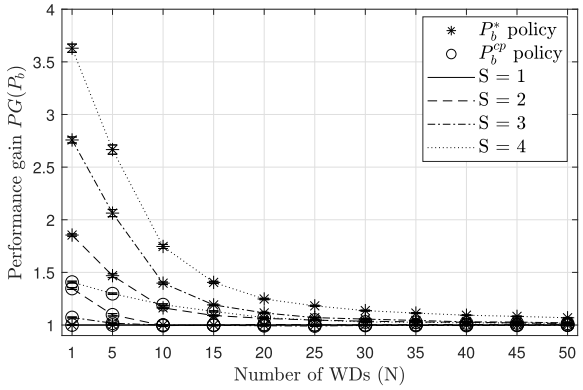

Fig. 4. Performance gain vs. number of WDs N.

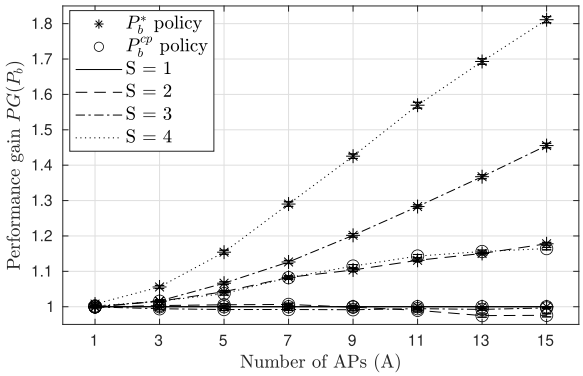

Fig. 5. Performance gain vs. number of APs A.

Theorem 9: The COS algorithm is a 2.62*-approximation algorithm for the optimization problem (33)-(34) in terms of the system cost, i.e.,* $\frac{\bar{C}(\mathbf{d}^*)}{\bar{C}(\hat{\mathbf{d}})} \leq 2.62$.

Proof: The result can be easily obtained by following the approach used to prove Theorem 8. □

Finally, from Theorem 3 and Theorem 9 we obtain the approximation ratio bound for the proposed decomposition-based algorithm.

Theorem 10: Given the collection $(\hat{\mathcal{P}}_b, \hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})$ *of optimal allocation policies, the proposed decomposition-based algorithm computes a* 2.62*-approximation solution to the JSS-ERM problem.*

VI. Numerical Results

We used extensive simulations to evaluate the cost performance of the proposed solution from the perspective of the operator, individual slices and devices, and for assessing the computational complexity of the proposed COS algorithm.

To capture the potentially uneven spatial distribution of ECs, WDs and APs in a dense urban area, we consider a square area of $1km \times 1\ km$ in which WDs and 3 ECs are placed uniformly at random and 5 APs are placed at random on a *regular grid* with 25 points. The channel gain of WD i to AP a depends on their Euclidean distance $d_{i,a}$ and on the path loss exponent α,

which we set to 4 according to the path loss model in urban and suburban areas [24]. Unless otherwise noted, we set the bandwidth B_a of 2 APs to 18*MHz* and the bandwidth of 3 APs to 27*MHz*, corresponding to 25 and 75 resource blocks that are 12×60*KHz* and 12×30*KHz* subcarriers wide [25], [26], respectively. We consider that the transmit power $P_{i,a}$ of every WD i is uniformly distributed on $[10^{-6}, 0.1]W$ according to [27]. We calculate the total thermal noise in a B_a*MHz* channel as $N_0(\text{dBm}) = -174 + 10\, log(B_a)$ according to [28] and the transmission rate $R_{i,a}$ achievable to WD i at AP a as $R_{i,a} = B_a log(1 + d_{i,a}^{-\alpha} \frac{P_{i,a}}{N_0})$.

To set the values for the computational capabilities of the WDs, we consider a line of Samsung Galaxy phones, from the oldest version with 1 core operating at 1*GHz* to the one of the newest versions with 8 cores operating at 2.84*GHz*. We consider that EC c_1 is equipped with 36 vCPUs operating at 2.3*GHz* and 96 vCPUs operating at 3.6*GHz*. We consider that EC c_2 and EC c_3 are equipped with 1 GPU each (with 2048 parallel processing cores operating at 557*MHz* and 2496 parallel processing cores operating at 560*MHz*, respectively). Given the measurements reported in [29]–[31] we assume that a WD, a CPU and a GPU can execute on average $2, 3$ and 1 instructions per cycle (*IPC*), respectively. Based on this, we consider that the computational capability F_i^l of every WD i is uniformly distributed on [2, 45.4]*GIPS*, where the lower and the upper bound correspond to the oldest and the newest version of the phone, respectively. Similarly, we calculate the computational capabilities of ECs, and set them to $F_{c1} = 1285.2$*GIPS*, $F_{c_2} = 1140.7$*GIPS* and $F_{c_3} = 1397.8$*GIPS*.

The input data size D_i is drawn from a uniform distribution on $[1.7, 10]$*Mb* according to measurements in [32]. The number X of instructions per data bit follows a Gamma distribution [33] with shape parameter $k = 75$ and scale $\theta = 50$. Given D_i and X, we calculate the complexity of a task as $L_i = D_i X$.

Motivated by Amazon EC2 instances [34] designed to support different kinds of applications (e.g., G3 and P2 instances for graphics-intensive and general-purpose GPU applications, and C5 and I3 instances for compute-intensive and non-virtualized workloads), we evaluate the system performance for the following four cases.

S= 1: The slice s_1 contains all ECs, and thus is able to support all of the above applications.

S= 2: The ECs are sliced such that slice s_1 supports the G3.4 instance and slice s_2 supports instances C5 and I3.

S= 3: The ECs are sliced such that slices s_1 and s_2 support P2 and G3s instances, respectively and slice s_3 supports instances C5 and I3.

S= 4: The ECs are sliced such that slices s_1, s_2, s_3 and s_4 support P2, G3s, C5 and I3 instances, respectively.

The coefficients $\frac{1}{h_{i,s}}$ were drawn from a continuous uniform distribution on $[0, 1]$ and unless otherwise noted, the results are shown for all of the above scenarios.

We use two bandwidth allocation policies $\mathcal{P}_b$ of the slice orchestrator as a basis for comparison. The first policy $\mathcal{P}_b^{cp}$ shares the bandwidth of each AP a among slices proportionally to the ECs' resources that slices have. The second policy $\mathcal{P}_b^{eq}$ gives an equal share of the bandwidth of each AP a to each slice s. Observe that the COS algorithm computes an approximation vector $\mathbf{d}^*$ of offloading decisions for both policies (c.f. Theorem 5 and Theorem 8). The results shown are the averages of 300 simulations, together with 95% confidence intervals.

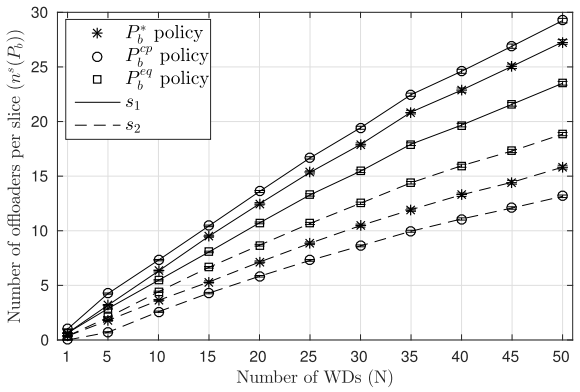

Fig. 6. Number of offloaders per slice vs. number of WDs N.

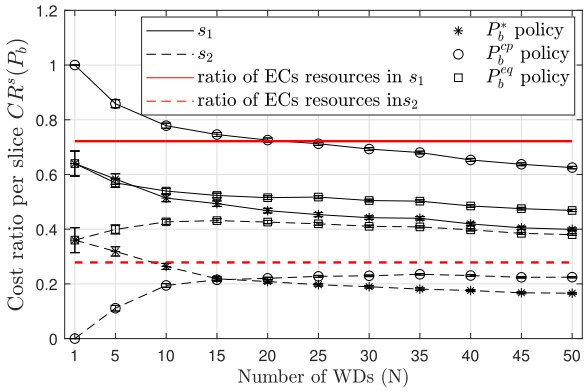

Fig. 7. Cost ratio per slice vs. number of WDs N.

A. System Performance

We start with considering the system performance from the point of view of the slice orchestrator. To do so we consider the collection $(\hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})$ of optimal intra-slice policies and define the system performance gain $PG(\mathcal{P}_b)$ for an inter-slice radio allocation policy $\mathcal{P}_b$ as the ratio between the system cost reached under policy $\mathcal{P}_b^{eq}$ and the system cost reached under policy $\mathcal{P}_b$

$$PG(\mathcal{P}_b) = \frac{C(\mathbf{d}^*, \mathcal{P}_b^{eq}, \hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})}{C(\mathbf{d}^*, \mathcal{P}_b, \hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})}.$$

We investigate the system performance gain $PG(\mathcal{P}_b)$ for the optimal $\mathcal{P}_b^*$ and for the cloud proportional $\mathcal{P}_b^{cp}$ allocation policies.

We first show $PG(\mathcal{P}_b)$ as a function of the number N of WDs in Figure 4. We observe that $PG(\mathcal{P}_b^*) = PG(\mathcal{P}_b^{cp}) = 1$ when $S = 1$ and this is because the three solutions are equivalent when there is no slicing. On the contrary, for $S > 1$ we observe that $PG(\mathcal{P}_b^*) > 1$ and $PG(\mathcal{P}_b^{cp}) > 1$, which is due to that the policy $\mathcal{P}_b^{eq}$ does not take into

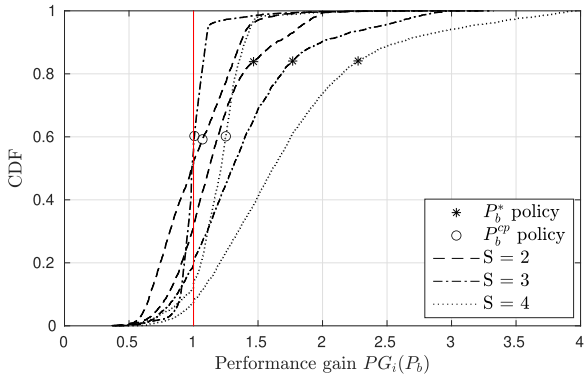

Fig. 8. Distribution of the individual performance gain for $N = 10$.

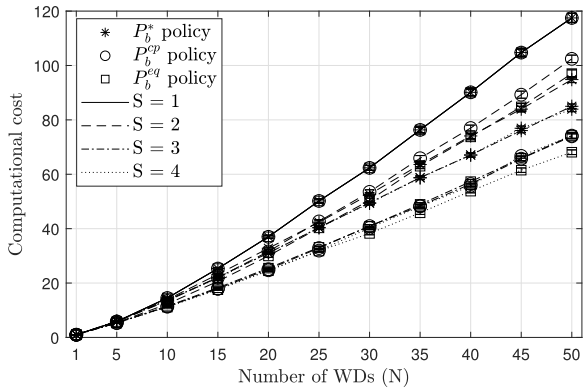

Fig. 9. Computational complexity vs. number of WDs N.

account that the slices might have different amounts of ECs' resources. We also observe that the policy $\mathcal{P}_b^*$ achieves better performance gain (up to 2.5 times greater) than the policy $\mathcal{P}_b^{cp}$ because $\mathcal{P}_b^*$ assigns the WDs to slices not only based on the amount of ECs' resources the slices have, but also based on how well the slices are tailored for executing their tasks. This effect is especially evident when there are few WDs, because in this case WDs tend to offload their tasks, and thus the system cost is mostly determined by the offloading cost. On the contrary, as the number N of WDs increases, the gap between considered inter-slice radio allocation policies vanishes because the system cost becomes mostly determined by the WDs that perform the computation locally.

Next, in Figure 5 we show $PG(\mathcal{P}_b)$ for $N = 25$ WDs as a function of the number A of APs, each of them with $B_a = 18MHz$. We first observe that $PG(\mathcal{P}_b^*)$ increases with both the number A of APs and the number S of slices and that $PG(\mathcal{P}_b^*) > 1$ already for $S = 2$. On the contrary, $PG(\mathcal{P}_b^{cp}) > 1$ only for $S = 4$, and $PG(\mathcal{P}_b^*)$ for $S = 2$ is approximately the same as $PG(\mathcal{P}_b^{cp})$ for $S = 4$, which illustrates the superior performance of the proposed policy $\mathcal{P}_b^*$.

B. Performance Within the Slices

We continue with considering the performance from the point of view of the slices. For an inter-slice radio allocation policy $\mathcal{P}_b$, we denote by $n^s(\mathcal{P}_b)$ the number of offloaders per slice in the vector $\mathbf{d}^*$ of offloading decisions computed by the COS algorithm and we define the cost ratio $CR^s(\mathcal{P}_b)$ per slice w.r.t. the system cost as

$$CR^s(\mathcal{P}_b) = \frac{C^s(\mathbf{d}^*, \mathcal{P}_b, \hat{\mathcal{P}}^s_{w_a}, \hat{\mathcal{P}}^s_{w_c})}{C(\mathbf{d}^*, \mathcal{P}_b, \hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})}.$$

Figure 6 and Figure 7 show $n^s(\mathcal{P}_b)$ and $CR^s(\mathcal{P}_b)$, respectively for the optimal $\mathcal{P}_b^*$, the cloud proportional $\mathcal{P}_b^{cp}$ and the equal $\mathcal{P}_b^{eq}$ inter-slice radio allocation policy of the slice orchestrator. The results are shown for $S = 2$ and the red lines in Figure 7 show the share of the ECs' resources among the slices s_1 and s_2 (i.e, slices s_1 and s_2 have approximately 72% and 28% of the resources, respectively). We observe from Figure 6 and Figure 7, respectively that the gap between $n^{s_1}(\mathcal{P}_b)$ and $n^{s_2}(\mathcal{P}_b)$ and the gap between $CR^{s_1}(\mathcal{P}_b)$ and $CR^{s_2}(\mathcal{P}_b)$ are highest in the case of the policy $\mathcal{P}_b^{cp}$ and lowest in the case of the policy $\mathcal{P}_b^{eq}$. Therefore, WDs whose tasks are a better match with the EC resources in slice s_2 than those in slice s_1 cannot fully exploit the ECs' resources in slice s_2 under the policy $\mathcal{P}_b^{cp}$, which allocates bandwidth resources proportionally to the ECs' resources. Similarly, WDs whose tasks are a better match with the EC resources in slice s_1 than in slice s_2 cannot fully exploit the ECs' resources in slice s_1 under the policy $\mathcal{P}_b^{eq}$, which allocates bandwidth resources equally. On the contrary, the results show that the optimal policy $\mathcal{P}_b^*$ finds a good match between the EC resources in the slices and the WDs' preferences for different types of computing resources, which makes it a good candidate for dynamic resource management for network slicing coupled with edge computing.

C. Performance Perceived Per Device

In order to evaluate the performance perceived per device, we define the *individual performance gain* $PG_i(\mathcal{P}_b)$ as

$$PG_i(\mathcal{P}_b) = \frac{C_i(\mathbf{d}^*, \mathcal{P}_b^{eq}, \hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})}{C_i(\mathbf{d}^*, \mathcal{P}_b, \hat{\mathcal{P}}_{w_a}, \hat{\mathcal{P}}_{w_c})}.$$

In Figure 8 we show the CDF of $PG_i(\mathcal{P}_b)$ for $N = 10$ and for the same set of parameters as in Figure 4. We omit the results for $S = 1$ since in this case the solutions achieved for policies $\mathcal{P}_b^*, \mathcal{P}_b^{cp}$ and $\mathcal{P}_b^{eq}$ are equivalent (c.f., Figure 4).

We observe that the proposed $\mathcal{P}_b^*$ policy outperforms the cloud proportional policy $\mathcal{P}_b^{cp}$ since for any value of S the probability that $PG_i(\mathcal{P}_b)$ is less than a fixed threshold is smaller in the case of policy $\mathcal{P}_b^*$ than in the case of policy $\mathcal{P}_b^{cp}$. Finally, we observe that in the case of both $\mathcal{P}_b^*$ and $\mathcal{P}_b^{cp}$ policies the probability that $PG_i(\mathcal{P}_b)$ is less than a fixed threshold decreases as the number S of slices increases, which confirms that network slicing can improve the performance of an edge computing system.

D. Computational Cost

Figure 9 shows the number of iterations in which the COS algorithm computes a decision vector $\mathbf{d}^*$ as a function of the number N of WDs under the optimal $\mathcal{P}_b^*$, the cloud proportional $\mathcal{P}_b^{cp}$ and the equal $\mathcal{P}_b^{eq}$ inter-slice radio allocation policy of the slice orchestrator.

Interestingly, the number of updates decreases with the number S of slices. This is due to that the congestion on the

logical resources decreases as S increases, and thus the COS algorithm updates the offloading decisions less frequently. We also observe that the number of updates scales approximately linearly with N under all considered policies of the slice orchestrator, and thus we can conclude that the COS algorithm is computationally efficient, which makes it a good candidate for computing an approximation $\mathbf{d}^*$ to the optimal vector $\hat{\mathbf{d}}$ of offloading decisions of WDs.

VII. Related Work

There is a significant body of works that consider computation offloading problems in edge computing systems in which multiple devices share both communication and computing resources [35]–[40]. The authors in [35] addressed the problem of minimizing the long-term average task completion times, and based on the decomposition of the original problem, proposed an iterative algorithm for which they provided a competitive ratio. In [36] the authors considered the problem of joint optimization of the task placement and the allocation of uplink and downlink communication resources, and proposed an algorithm that minimizes the average completion times of the tasks. The authors in [37] defined the cost of each device as a function of its energy consumption and task completion time, formulated the problem of minimizing the sum cost of devices, and based on the relaxation of the original problem, proposed a heuristic for computing the offloading decisions of devices and allocating communication and computing resources. In [38] and [39] the authors considered an edge computing system with limited communication, computing and storage resources. The authors in [38] formulated the joint computation offloading, content caching, and resource allocation problem, and based on the generalized Benders decomposition, proposed an iterative algorithm. In [39] the authors considered the joint optimization of service placement and request routing in a dense edge computing system, and proposed an approximation algorithm that leverages a randomized rounding technique. The authors in [40] addressed the problem of joint resource dimensioning and placement of virtualized services, and proposed an approximation algorithm based on Lagrangian relaxation. Different from these works, we consider an edge computing system in which the resources are managed both at the network and at the slice level, and we propose a novel game theory inspired approach for designing an approximate solution to the joint task placement and resource allocation problem.

Closest to our work are recent game theoretic treatments of the computation offloading problem [41]–[45]. In [41] the authors considered devices that compete for cloud resources so as to minimize their energy consumption, and proved that an equilibrium of offloading decision can be computed in polynomial time. In [42] the authors considered devices that maximize their performance and a profit maximizing service provider, and used backward induction for deriving near optimal strategies for the devices and the operator. In [43] the authors considered that devices can offload their tasks to a cloud through multiple identical wireless links, modeled the congestion on wireless links, and used a potential function argument for proposing a decentralized algorithm for computing an equilibrium. In [44] the authors considered that devices can offload their tasks to a cloud through multiple heterogeneous wireless links, modeled the congestion on wireless and cloud resources, showed that the game played by devices is not a potential game and proposed a decentralized algorithm for computing an equilibrium. In [45] the authors modeled the interaction between devices and a single network operator as a Stackelberg game, and provided an algorithm for computing a subgame perfect equilibrium. Unlike these works, we consider the computation offloading problem together with network slicing, provide a general model of computational tasks that accounts for the diversity of computing resources across the slices, and analyze the interaction between the network operator and the slices.

Another line of works considers the network slicing resource allocation problem [46]–[50]. In [46] the authors considered an auction-based model for allocating edge cloud resources to slices and proposed an algorithm for allocating resources to slices so as to maximize the total network revenue. In [47] the authors considered the radio resources slicing problem and proposed an approximation algorithm for maximizing the sum of the users' utilities. In [48] the authors modeled the interaction between slices that compete for bandwidth resources with the objective to maximize the sum of their users' utilities, and proposed an admission control algorithm under which the slices can reach an equilibrium. In [49] the authors proposed a deep learning architecture for sharing the resources among network slices in order to meet the users' demand within the slices. In [50] the authors considered a radio access network slicing problem and proposed two approximation algorithms for maximizing the total network throughput. Unlike these works, we consider a slicing enabled edge system in which the slice resource orchestrator assigns devices to slices and shares radio resources across slices, while the slices manage their own radio and computing resources with the objective to maximize overall system performance.

To the best of our knowledge, ours is the first work to consider the combinatorial problem of placing heterogeneous computational tasks together with the problem of optimizing the inter-slice and intra-slice resource allocation policies for joint management of communication and computing resources in an edge computing system.

VIII. Conclusion and Discussion

We have considered the computation offloading problem in an edge computing system under network slicing in which slices jointly manage their own communication and computing resources and the slice resource orchestrator manages communication resources among slices and assigns the WDs to slices. We formulated the problem of minimizing the sum over all WDs' task completion times as a mixed-integer problem, proved that the problem is NP-hard and proposed a decomposition of the problem into a sequence of optimization problems. We proved that the proposed decomposition does not change the optimal solution of the original problem, proposed an efficient approximation algorithm for solving

the decomposed problem and proved that the algorithm has bounded approximation ratio. Our numerical results show that the proposed algorithm is computationally efficient. They also show that dynamic allocation of slice resources is essential for maximizing the benefits of edge computing, and slicing could be beneficial for improving overall system performance.

We conclude with discussing two potential extensions of our work. First, we discuss the applicability of our model to three classes of emerging extended reality (XR) applications [51]. Our model captures well the class of uplink traffic dominated XR conversational and conference applications. For the class of downlink traffic dominated applications, the considered model and problem formulation could be applicable with minor modifications: the identity of WDs and ECs would have to be swapped in the model, and whether or not to offload and to which EC (i.e., WD) to send data to would not be a decision any longer. Our model does not fit the class of XR applications where downlink traffic and uplink traffic are equally important, we leave this as an interesting avenue of future work.

Second, we discuss the applicability of our solution to WDs that cannot perform the computational tasks locally, e.g., due to lack of computing resources. The proposed game theoretic approach extends nicely to this case, by defining the local execution cost to be the desired completion time of the WD. By doing so, a WD would only offload if it can meet the desired completion time, but would not offload if offloading violates the completion deadline, and it may thus decide not to perform the task. An interesting avenue for future research would be to consider a fairness objective, e.g., WDs that cannot perform the tasks locally would receive preferential treatment in resource allocation, but such a formulation would have to be combined with appropriate financial incentives, i.e., pricing, leading to a significantly different problem formulation from ours.

APPENDIX

A. Proof of Proposition 1

First, let us define the collections $\mathbf{b} \triangleq (b_a^{\mathrm{s}})_{s\in\mathcal{S},a\in\mathcal{A}}$ and $\mathbf{w} \triangleq (w_{i,e}^{\mathrm{s}})_{i\in\mathcal{N},e\in\mathcal{E},s\in\mathcal{S}}$ of inter-slice and intra-slice resource provisioning coefficients, respectively. Observe that the total number of provisioning coefficients in (18)-(19) is $p = NS(A+C+1)$. Next, let us consider a WD i such that $d_i = (a,c,s)$ and let us express the Hessian matrix of its cost (8)

$$\mathcal{H}_i = \begin{pmatrix} \frac{2\tau_{i,a}^{\mathrm{s}}}{b_a^{\mathrm{s}^3} w_{i,a}^{\mathrm{s}}} & \frac{\tau_{i,a}^{\mathrm{s}}}{b_a^{\mathrm{s}^2} w_{i,a}^{\mathrm{s}^2}} & 0 & \mathcal{O}_{1\times(p-3)} \\ \frac{\tau_{i,a}^{\mathrm{s}}}{b_a^{\mathrm{s}^2} w_{i,a}^{\mathrm{s}^2}} & \frac{2\tau_{i,a}^{\mathrm{s}}}{b_a^{\mathrm{s}} w_{i,a}^{\mathrm{s}^3}} & 0 & \mathcal{O}_{1\times(p-3)} \\ 0 & 0 & \frac{\tau_{i,c}^{\mathrm{s}}}{w_{i,c}^{\mathrm{s}^3}} & \mathcal{O}_{1\times(p-3)} \\ \mathcal{O}_{(p-3)\times 1} & \mathcal{O}_{(p-3)\times 1} & \mathcal{O}_{(p-3)\times 1} & \mathcal{O}_{(p-3)\times(p-3)} \end{pmatrix},$$

where $\mathcal{O}_{m\times n}$ is a zero matrix of m rows and n columns.

It is easy to see that $\mathcal{H}_i$ is a symmetric matrix, and thus it has p eigenvalues among which $p-3$ are equal to 0. The remaining three eigenvalues are given by $\lambda_1 = \frac{2\tau_{i,c}^{\mathrm{s}}}{w_{i,c}^{\mathrm{s}^3}}$, $\lambda_2 = \frac{\tau_{i,a}^{\mathrm{s}}(b_a^{\mathrm{s}^2}+w_{i,a}^{\mathrm{s}^2}-\sqrt{b_a^{\mathrm{s}^4}-b_a^{\mathrm{s}^2}w_{i,a}^{\mathrm{s}^2}+w_{i,a}^{\mathrm{s}^4}})}{b_a^{\mathrm{s}^3}w_{i,a}^{\mathrm{s}^3}}$ and $\lambda_3 = \frac{\tau_{i,a}^{\mathrm{s}}(b_a^{\mathrm{s}^2}+w_{i,a}^{\mathrm{s}^2}+\sqrt{b_a^{\mathrm{s}^4}-b_a^{\mathrm{s}^2}w_{i,a}^{\mathrm{s}^2}+w_{i,a}^{\mathrm{s}^4}})}{b_a^{\mathrm{s}^3}w_{i,a}^{\mathrm{s}^3}}$, respectively. Clearly, $\lambda_1 \geq 0$ since $\tau_{i,c}^{\mathrm{s}}, w_{i,c}^{\mathrm{s}} \geq 0$. Furthermore, since $\tau_{i,a}^{\mathrm{s}}, b_a^{\mathrm{s}}, w_{i,a}^{\mathrm{s}} \geq 0$ we have the following

$$\lambda_2 \geq \frac{\tau_{i,a}^{\mathrm{s}}\left(b_a^{\mathrm{s}^2}+w_{i,a}^{\mathrm{s}^2}-\sqrt{(b_a^{\mathrm{s}^2}+w_{i,a}^{\mathrm{s}^2})^2}\right)}{b_a^{\mathrm{s}^3}w_{i,a}^{\mathrm{s}^3}} \geq 0,$$

$$\lambda_3 \geq \frac{\tau_{i,a}^{\mathrm{s}}\left(b_a^{\mathrm{s}^2}+w_{i,a}^{\mathrm{s}^2}+\sqrt{(b_a^{\mathrm{s}^2}-w_{i,a}^{\mathrm{s}^2})^2}\right)}{b_a^{\mathrm{s}^3}w_{i,a}^{\mathrm{s}^3}} \geq 0.$$

Hence, the cost $C_i(\mathbf{d}, \mathcal{P}_b, \mathcal{P}_{w_a}, \mathcal{P}_{w_c})$ of WD i that offloads its computation is convex in $(\mathbf{b}, \mathbf{w})$. Since the cost of a WD i that performs the computation locally does not depend on provisioning coefficients $\mathbf{b}$ and $\mathbf{w}$ (c.f., equation (8)), we have that the convexity of (18)-(19) follows from the known results that the sum of convex functions is convex, and that the sublevel sets of convex functions are convex. This proves the result.

B. Proof of Proposition 2

First, observe that constraint (13) can be omitted since we assumed that the decision vector $\mathbf{d}$ is such that constraint (13) can be satisfied. From Proposition 1 we have that (18)-(19) is a convex problem, and thus its optimal solution must satisfy the Karush–Kuhn–Tucker (KKT) conditions. In order to formulate the corresponding Lagrangian dual problem, let us introduce non-negative Lagrange multiplier vectors $\boldsymbol{\alpha} = (\alpha_a)_{a\in\mathcal{A}}$, $\boldsymbol{\beta} = (\beta_e^{\mathrm{s}})_{e\in\mathcal{E},s\in\mathcal{S}}$, $\boldsymbol{\gamma} = (\gamma_a^{\mathrm{s}})_{a\in\mathcal{A},s\in\mathcal{S}}$ and $\boldsymbol{\delta} = (\delta_{i,e}^{\mathrm{s}})_{i\in O_{e,s}(\mathbf{d}),e\in\mathcal{E},s\in\mathcal{S}}$ for constraints in (19), respectively. Next, let us define the Lagrangian dual problem corresponding to problem (18)-(19) as $\max_{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{\delta}\succeq 0} \min_{\mathbf{b},\mathbf{w}\succeq 0} \mathcal{L}(\mathbf{b},\mathbf{w},\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{\delta})$, where the Lagrangian is given by

$$\begin{aligned}
&\mathcal{L}(\mathbf{b},\mathbf{w},\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{\delta}) \\
&= \sum_{s'\in\mathcal{S}}\sum_{e'\in\mathcal{E}} \frac{1}{b_{e'}^{\mathrm{s}'}}\Big(\sum_{j\in O_{e',s'}(\mathbf{d})} \frac{\tau_{j,e'}^{\mathrm{s}'}}{w_{j,e'}^{\mathrm{s}'}}\Big) \\
&\quad + \sum_{a'\in\mathcal{A}} \alpha_{a'}\Big(\sum_{s'\in\mathcal{S}} b_{a'}^{\mathrm{s}'} - 1\Big) + \sum_{e'\in\mathcal{E}}\sum_{s'\in\mathcal{S}} \beta_{e'}^{\mathrm{s}'}\Big(\sum_{j\in O_{e',s'}(\mathbf{d})} w_{j,e'}^{\mathrm{s}'} - 1\Big)\Big) \\
&\quad - \sum_{a'\in\mathcal{A}}\sum_{s'\in\mathcal{S}} \gamma_{a'}^{\mathrm{s}'} b_{a'}^{\mathrm{s}'} - \sum_{e'\in\mathcal{E}}\sum_{s'\in\mathcal{S}}\sum_{j\in O_{e',s'}(\mathbf{d})} \delta_{j,e'}^{\mathrm{s}'} w_{j,e'}^{\mathrm{s}'} + \sum_{j\in O_l(\mathbf{d})} C_j^l.
\end{aligned}$$

Now, we can express the KKT conditions as follows

$$\textbf{stationarity:} \quad \sum_{j\in O_{a,s}(\mathbf{d})} \frac{\tau_{j,a}^{\mathrm{s}}}{w_{j,a}^{\mathrm{s}}} \cdot \frac{1}{(b_a^{\mathrm{s}})^2} = \alpha_a - \gamma_a^{\mathrm{s}}, a\in\mathcal{A}, s\in\mathcal{S} \tag{45}$$

$$\frac{\tau_{i,e}^{\mathrm{s}}}{b_e^{\mathrm{s}}(w_{i,e}^{\mathrm{s}})^2} = \beta_e^{\mathrm{s}} - \delta_{i,e}^{\mathrm{s}}, e\in\mathcal{E}, s\in\mathcal{S}, i\in O_{e,s}(\mathbf{d}) \tag{46}$$

$$\textbf{pr. feasibility:} \ (19) \tag{47}$$

$$\textbf{du. feasibility:} \ \boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{\delta} \succeq 0, \tag{48}$$

$$\textbf{co. slackness:} \ \alpha_a\Big(\sum_{s'\in\mathcal{S}} b_a^{\mathrm{s}'} - 1\Big), a\in\mathcal{A} \tag{49}$$

$$\beta_e^{\mathrm{s}}\Big(\sum_{j\in O_{e,s}(\mathbf{d})} w_{j,e}^{\mathrm{s}} - 1\Big) = 0, e\in\mathcal{E}, s\in\mathcal{S} \quad (50)$$

$$-\gamma_a^{\mathrm{s}} b_a^{\mathrm{s}} = 0, a\in\mathcal{A}, s\in\mathcal{S} \quad (51)$$

$$-\delta_{i,e}^{\mathrm{s}} w_{i,e}^{\mathrm{s}} = 0, e\in\mathcal{E}, s\in\mathcal{S}, i\in O_{e,s}(\mathbf{d}). \quad (52)$$

We proceed with finding $\hat{w}_{i,e}^{\mathrm{s}}$. First, from the KKT dual feasibility condition $\boldsymbol{\delta} \succeq 0$ and complementary slackness condition (52) we obtain that $\delta_{i,e}^{\mathrm{s}} = 0$ must hold for every $e \in \mathcal{E}$, $s \in \mathcal{S}$ and $i \in O_{e,s}(\mathbf{d})$ as otherwise $w_{i,e}^{\mathrm{s}} = 0$ would lead to infinite value of the objective function. Then, from the KKT stationarity condition (46) and complementary slackness condition (50) we obtain the expression (20) for coefficients $\hat{w}_{i,e}^{\mathrm{s}}$. Finally, by substituting expression (20) into the KKT stationarity condition (45) and by following the same approach as for finding $\hat{w}_{i,e}^{\mathrm{s}}$ we obtain the expression (21) for coefficients $\hat{b}_a^{\mathrm{s}}$, which proves the result.

REFERENCES

[1] J. Ordonez-Lucena, P. Ameigeiras, D. Lopez, J. J. Ramos-Munoz, J. Lorca, and J. Folgueira, "Network slicing for 5G with SDN/NFV: Concepts, architectures, and challenges," *IEEE Commun. Mag.*, vol. 55, no. 5, pp. 80–87, May 2017.

[2] S. Kekki *et al.* (2018). *MEC in 5G networks*. Sophia Antipolis, France, ETSI, White Paper. [Online]. Available: https://portal.etsi.org/TB-SiteMap/MEC/MEC-White-Papers

[3] M. Rost and S. Schmid, "Virtual network embedding approximations: Leveraging randomized rounding," *IEEE/ACM Trans. Netw.*, vol. 27, no. 5, pp. 2071–2084, Oct. 2019.

[4] B. Farkiani, B. Bakhshi, and S. A. MirHassani, "A fast near-optimal approach for energy-aware SFC deployment," *IEEE Trans. Netw. Service Manage.*, vol. 16, no. 4, pp. 1360–1373, Dec. 2019.

[5] I. Jang, D. Suh, S. Pack, and G. Dán, "Joint optimization of service function placement and flow distribution for service function chaining," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 11, pp. 2532–2541, Nov. 2017.

[6] S. Redana *et al.*, "5G PPP architecture working group: View on 5G architecture," Version 3.0, Eur. Commission, Brussels, Belgium, Feb. 2020, Tech. Rep., doi: 10.5281/zenodo.3265031.

[7] X. Foukas, M. K. Marina, and K. Kontovasilis, "Orion: RAN slicing for a flexible and cost-effective multi-service mobile network architecture," in *Proc. 23rd Annu. Int. Conf. Mobile Comput. Netw.*, Oct. 2017, pp. 127–140.

[8] A. Rostami *et al.*, "Orchestration of RAN and transport networks for 5G: An SDN approach," *IEEE Commun. Mag.*, vol. 55, no. 4, pp. 64–70, Apr. 2017.

[9] D. Bega, M. Gramaglia, A. Garcia-Saavedra, M. Fiore, A. Banchs, and X. Costa-Perez, "Network slicing meets artificial intelligence: An AI-based framework for slice management," *IEEE Commun. Mag.*, vol. 58, no. 6, pp. 32–38, Jun. 2020.

[10] C.-Y. Chang, N. Nikaein, and T. Spyropoulos, "Radio access network resource slicing for flexible service execution," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Apr. 2018, pp. 668–673.

[11] A. Ksentini and N. Nikaein, "Toward enforcing network slicing on RAN: Flexibility and resources abstraction," *IEEE Commun. Mag.*, vol. 55, no. 6, pp. 102–108, Jun. 2017.

[12] J. Zheng, Y. Cai, Y. Wu, and X. Shen, "Dynamic computation offloading for mobile cloud computing: A stochastic game-theoretic approach," *IEEE Trans. Mobile Comput.*, vol. 18, no. 4, pp. 771–786, Apr. 2018.

[13] X. Chen, "Decentralized computation offloading game for mobile cloud computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 4, pp. 974–983, Apr. 2015.

[14] S. Jošilo and G. Dán, "Decentralized algorithm for randomized task allocation in fog computing systems," *IEEE/ACM Trans. Netw.*, vol. 27, no. 1, pp. 85–97, Feb. 2018.

[15] J. L. D. Neto, S. Yu, D. F. Macedo, J. M. S. Nogueira, R. Langar, and S. Secci, "ULOOF: A user level online offloading framework for mobile edge computing," *IEEE Trans. Mobile Comput.*, vol. 17, no. 11, pp. 2660–2674, Nov. 2018.

[16] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, "CloneCloud: Elastic execution between mobile device and cloud," in *Proc. 6th Conf. Comput. Syst.*, 2011, pp. 301–314.

[17] E. Cuervo *et al.*, "MAUI: Making smartphones last longer with code offload," in *Proc. 8th Int. Conf. Mobile Syst., Appl., Services (MobiSys)*, 2010, pp. 49–62.

[18] S. Jošilo and G. Dán, "Selfish decentralized computation offloading for mobile cloud computing in dense wireless networks," *IEEE Trans. Mobile Comput.*, vol. 18, no. 1, pp. 207–220, Jan. 2019.

[19] D. Huang, P. Wang, and D. Niyato, "A dynamic offloading algorithm for mobile computing," *IEEE Trans. Wireless Commun.*, vol. 11, no. 6, pp. 1991–1995, Jun. 2012.

[20] S. Josilo and G. Dan, "Joint management of wireless and computing resources for computation offloading in mobile edge clouds," *IEEE Trans. Cloud Comput.*, vol. 9, no. 4, pp. 1507–1520, Oct. 2021.

[21] M. R. Garey and D. S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness*. New York, NY, USA: W. H. Freeman & Co., 1990.

[22] T. Harks, M. Klimm, and R. H. Möhring, "Characterizing the existence of potential functions in weighted congestion games," *Theory Comput. Syst.*, vol. 49, no. 1, pp. 46–70, Jul. 2011.

[23] D. Monderer and L. S. Shapley, "Potential games," *Games Econ. Behavior*, vol. 14, no. 1, pp. 124–143, 1996.

[24] S. R. Saunders and A. Aragón-Zavala, *Antennas and Propagation for Wireless Communication Systems*. Hoboken, NJ, USA: Wiley, 2007.

[25] (2011). *LTE; Evolved Universal Terrestrial Radio Access (E-UTRA); Physical Channels and Modulation*. Antipolis, France, ETSI, White Paper, Version 10.0.0. [Online]. Available: https://www.etsi.org/deliver/etsi_ts/136200_136299/136211/newlin% e10.01.00_60/

[26] A. Zaidi, F. Athley, J. Medbo, U. Gustavsson, G. Durisi, and X. Chen, *5G Physical Layer: Principles, Models and Technology Components*, 1st ed. New York, NY, USA: Academic, 2018.

[27] M. Lauridsen, L. Noël, T. B. Sørensen, and P. Mogensen, "An empirical lte smartphone power model with a view to energy efficiency evolution," *Intel Technol. J.*, vol. 18, no. 1, pp. 172–193, 2014.

[28] N. Da Dalt and A. Sheikholeslami, *Understanding Jitter and Phase Noise: A Circuits and Systems Perspective*. Cambridge, U.K.: Cambridge Univ. Press, 2018.

[29] L. Codrescu *et al.*, "Hexagon DSP: An architecture optimized for mobile multimedia and communications," *IEEE Micro*, vol. 34, no. 2, pp. 34–43, Mar. 2014.

[30] D. Hackenberg, R. Schöne, T. Ilsche, D. Molka, J. Schuchart, and R. Geyer, "An energy efficiency feature survey of the Intel Haswell processor," in *Proc. IEEE Int. Parallel Distrib. Process. Symp. Workshop*, May 2015, pp. 896–904.

[31] Y. Takefuji, *GPU Parallel Computing for Machine Learning in Python: How to Build a Parallel Computer*. Independently published, Jun. 2017.

[32] L. Fletcher, L. Petersson, and A. Zelinsky, "Road scene monotony detection in a fatigue management driver assistance system," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2005, pp. 484–489.

[33] J. R. Lorch and A. J. Smith, "PACE: A new approach to dynamic voltage scaling," *IEEE Trans. Comput.*, vol. 53, no. 7, pp. 856–869, Jul. 2004.

[34] *Amazon EC2 Instance Types*. Accessed: Apr. 2021. [Online]. Available: https://aws.amazon.com/ec2/instance-types/

[35] B. Gao, Z. Zhou, F. Liu, and F. Xu, "Winning at the starting line: Joint network selection and service placement for mobile edge computing," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2019, pp. 1459–1467.

[36] K. Guo, M. Yang, Y. Zhang, and J. Cao, "Joint computation offloading and bandwidth assignment in cloud-assisted edge computing," *IEEE Trans. Cloud Comput.*, early access, Oct. 30, 2019, doi: 10.1109/TCC.2019.2950395.

[37] M.-H. Chen, B. Liang, and M. Dong, "Multi-user multi-task offloading and resource allocation in mobile cloud systems," *IEEE Wireless Commun.*, vol. 17, no. 10, pp. 6790–6805, Oct. 2018.

[38] J. Zhang *et al.*, "Joint resource allocation for latency-sensitive services over mobile edge computing networks with caching," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4283–4294, Jun. 2018.

[39] K. Poularakis, J. Llorca, A. M. Tulino, I. Taylor, and L. Tassiulas, "Service placement and request routing in MEC networks with storage, computation, and communication constraints," *IEEE/ACM Trans. Netw.*, vol. 28, no. 3, pp. 1047–1060, Jun. 2020.

[40] P. Zhao and G. Dan, "Joint resource dimensioning and placement for dependable virtualized services in mobile edge clouds," *IEEE Trans. Mobile Comput.*, early access, Feb. 18, 2021, doi: 10.1109/TMC.2021.3060118.

[41] Y. Ge, Y. Zhang, Q. Qiu, and Y.-H. Lu, "A game theoretic resource allocation for overall energy minimization in mobile cloud computing system," in *Proc. ACM/IEEE Int. Symp. Low Power Electron. Design (ISLPED)*, 2012, pp. 279–284.
[42] Y. Wang, X. Lin, and M. Pedram, "A nested two stage game-based optimization framework in mobile cloud computing system," in *Proc. IEEE 7th Int. Symp. Service-Oriented Syst. Eng.*, Mar. 2013, pp. 494–502.
[43] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct. 2016.
[44] S. Josilo and G. Dan, "A game theoretic analysis of selfish mobile computation offloading," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, May 2017, pp. 1–9.
[45] S. Josilo and G. Dan, "Wireless and computing resource allocation for selfish computation offloading in edge computing," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2019, pp. 2467–2475.
[46] M. Jiang, M. Condoluci, and T. Mahmoodi, "Network slicing in 5G: An auction-based model," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.
[47] P. Caballero *et al.*, "Multi-tenant radio access network slicing: Statistical multiplexing of spatial loads," *IEEE/ACM Trans. Netw.*, vol. 25, no. 5, pp. 3044–3058, Oct. 2017.
[48] P. Caballero, A. Banchs, G. de Veciana, X. Costa-Pérez, and A. Azcorra, "Network slicing for guaranteed rate services: Admission control and resource allocation games," *IEEE Trans. Wireless Commun.*, vol. 17, no. 10, pp. 6419–6432, Oct. 2018.
[49] D. Bega, M. Gramaglia, M. Fiore, A. Banchs, and X. Costa-Perez, "DeepCog: Cognitive network management in sliced 5G networks with deep learning," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2019, pp. 280–288.
[50] S. D'Oro, F. Restuccia, A. Talamonti, and T. Melodia, "The slice is served: Enforcing radio access network slicing in virtualized 5G systems," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2019, pp. 442–450.
[51] (2020). 3GPP. *Extended Reality (XR) in 5G*. [Online]. Available: https://www.3gpp.org/ftp/Specs/archive/26_series/26.928/

Sladana Jošilo received the M.Sc. degree in electrical engineering from the University of Novi Sad, Serbia, in 2012, and the Ph.D. degree in electrical engineering from the KTH Royal Institute of Technology, Stockholm, Sweden, in 2020. She worked as a Research Engineer with the Department of Power, Electronics and Communication Engineering, University of Novi Sad, from 2013 to 2014, and as a Post-Doctoral Researcher with the Division of Network and Systems Engineering, KTH Royal Institute of Technology, from 2020 to 2021. She currently works as a Researcher at Ericsson, Stockholm. Her research interests include 5G networks, edge computing systems, and applied game theory.

György Dán (Senior Member, IEEE) received the M.Sc. degree in computer engineering from the Budapest University of Technology and Economics, Hungary, in 1999, the M.Sc. degree in business administration from the Corvinus University of Budapest, Hungary, in 2003, and the Ph.D. degree in telecommunications from the KTH Royal Institute of Technology in 2006. He worked as a Consultant in the field of access networks, streaming media, and videoconferencing from 1999 to 2001. He was a Visiting Researcher with the Swedish Institute of Computer Science in 2008, a Fulbright Research Scholar at the University of Illinois at Urbana-Champaign from 2012 to 2013, and an Invited Professor at EPFL from 2014 to 2015. He is currently a Professor with the KTH Royal Institute of Technology, Stockholm, Sweden. 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. He served as an Area Editor for *Computer Communications* from 2014 to 2021, and he has been an Editor of IEEE TRANSACTIONS ON MOBILE COMPUTING since 2019.